République Algérienne Démocratique et Populaire

Ministère d'Enseignement Supérieur et de la Recherche Scientifique
Université ABBAS LAGHROUR – Khenchela -
Département MI

Mémoire

Présenté en vue d'obtenir le diplôme de

Master en Informatique (LMD)

Spécialité: Sécurité et Technologie Web

THÈME

Une approche base sur l'apprentissage automatique pour Lever l'ambigüité sémantique

Encadré par:

Dr. BAKHOUCHE Abdelaali

Dr. LEDMI Makhlouf - Co-encadreur -

Présenté par:

BOUDOUHA Rabie

MEKERSI Said

Année universitaire : 2021-2022

Remerciements

Tout d'abord, nous remercions ALLAH le tout puissant pour son aide et pour nous avoir donné la force et la patience afin d'arriver à terminé ce travail.

Nous tenons à exprimer nos vifs remerciements à notre encadreur monsieur Bakhouche Abdelaali pour son temps précieux , ses conseils et disponibilité tout au long du déroulement de ce travail.

Nous exprimons également nos remerciements aux membres de jurys pour avoir acceptés d'honorer par leur jugement notre travail.

Nous remercions nos deux familles qui nous ont toujours encouragé et soutenu durant toutes nos études tout au long de notre parcours.

Nous remercions aussi toutes les personnes qui nous ont aidés de prêt ou de loin a la réalisation de notre travail.

Said. M & Rabie.B

Dédicaces

Je dédie ce modeste travail du fond du cœur a tout les personne je j'aime :

A ma précieuse mère, que Dieu ait pitié d'elle

A chère père qui m'ont soutenue et aidé et qui ont toujours crue en moi et à mes capacités, je les remercie pour leurs sacrifices afin de venir se que je suis.

A mes chères sœurs :Assia ,Saida, Hbara et Kenza .

A mon frère : Amine.

A ma chère tentes .

A toute la famille BOUDOUHA sans exception.

A mes très chères amies

A mon binôme Said qui n'a pas cessé et m'encourager tout le temps et sans oublier sa famille.

A tous mes collègues promotion 2021/2022.

Rabie. BOUDOUHA

Dédicaces

A mes parents: Ma mère et mon père

A ma petite famille:

Mon épouse, mes deux anges Mohamed Islam & ala errahman

A mes frères et sœurs et toutes leurs familles,

A tous mes amis,

Said MEKERSI.

ملخـص:

التحديد الآلي لمعنى الكلمة حسب سياقها في الجملة، هي خطوة رئيسية مكملة لعدة تطبيقات لغوية كالترجمة الألية والبحث عن المعلومات ...

نقدم في هذه المذكرة خطوات إنشاء نظام تحديد المعاني للغة الأنجليزية وذلك برفع الغموض عن معنى الكلمة، نستغل في ذلك أنطولوجية Wordnet كمصدر للمعلومات وطريقة المعالجة الأولية، بالإضافة إلى الإعتماد على خوارزميتين الأولى تسمى خوارزمية Lesk كطريقة محلية، وخوارزمية شاملة تدعى خوارزمية الجرذان والتي اقترحنا منها نسخة تدعى DRSO .

وقد استعملنا الطريقة المحلية لقياس التقارب بين لفظين، في حين نستعمل الطريقة الشاملة لنشر هذا القياس على نطاق أوسع.

في النهاية نقيم هذا النظام باستعمال مدونة.

Résumé:

La désambiguïsation lexicale est une tâche fondamentale pour la plupart des applications de traitement automatique des langages naturels. Plusieurs solutions ont été proposées pour certain langues notamment l'anglais ou le français.

Dans ce mémoire, nous présentons les étapes pour construire notre approche, en exploitant les ressources Web comme le WordNet et les outils du prétraitement.

Nous présentons ensuite la notion d'algorithme local et d'algorithme global pour la désambiguïsation sémantique.

Un algorithme local permet de calculer la proximité sémantique entre deux objets lexicaux, cependant l'algorithme global permet de propager ces mesures locales à un niveau supérieur. Nous nous servons de cette notion pour confronter un algorithme d'optimisation d'essaim des rats (RSO) et pour résoudre ce problème on a ont proposé une version discret (DRSO). Après nous les évaluant les résultats sur un corpus référencé. ses résultats obtenus nous ont menée à découvrir les facteurs qui ont influencé la performance de ce système qui seront sujets d'éventuelle amélioration dans nos future travaux de recherches.

Abstract:

Lexical disambiguation is a fundamental task for most natural language processing applications. Several solutions have been proposed for certain languages, in particular English or French. In this thesis, we present the steps to build our approach, using web resources such as WordNet and preprocessing tools.

Then we present the notion of local algorithm and global algorithm for semantic disambiguation. A local algorithm calculates the semantic proximity between two objects lexical, however the global algorithm makes it possible to propagate these local measures to a higher level. We use this notion to confront a rat swarm optimization algorithm (RSO) and to solve this problem we have proposed a discrete version (DRSO).

After we evaluate the results on a referenced corpus. its results have led us to discover the factors that influenced the performance of this system which will be subjects of possible improvement in our future research work.

Table des matières

Chapitre III: Approche proposée

Liste des Figures

Liste des Tableaux

Introduction générale

Introduction générale :

Dans notre société, l'information – et plus particulièrement l'information électronique – prend chaque jour plus d'importance, et sa maîtrise est devenue la clef d'une certaine compétition, tant au niveau politique qu'économique ou scientifique.

Dès lors, il est capital de pouvoir gérer les masses de données sans cesse plus abondantes qui sont mises à notre disposition. Dans cette perspective, diverses applications informatiques se révèlent indispensables pour qui veut prendre part à cet essor.

En effet, l'analyse approfondie du contenu des documents, la catégorisation de ce contenu, la recherche d'une information précise dans une base de données, ou encore la traduction assistée sont des outils indispensables pour avoir une compréhension globale et exacte des éléments jugés intéressants [Ide et Véronis, 1998]. Or ces différents outils partagent un même postulat : il s'agit de pouvoir distinguer le sens adéquat de chaque mot en fonction du contexte dans lequel il apparaît. C'est là tâche assignée aux systèmes de désambiguïsation sémantique lexicale.

L'une des principales caractéristiques des langues naturelles réside dans le fait qu'un mot, une expression ou une phrase peut avoir plusieurs sens différents. Il s'agit du problème de l'ambiguïté sémantique qui reste un défi majeur pour tous les systèmes de recherche d'information (SRI) ou de traduction automatique. Ce problème oblige les chercheurs à développer des outils pour la compréhension du langage naturel. Certains auteurs [Vidhu Bhala et Abirami, 2012; Nguyen et Ock, 2013] distinguent plusieurs types d'ambiguïtés tels que la polysémie et l'homonymie, mais nous considérons généralement les mots qui ont la même orthographe et des sens différents, quel que soit le degré de proximité entre ces sens. Ces cas peuvent biaiser les résultats de tout système de traitement automatique du langage naturel (TALN). [19]

Cependant, il est nécessaire d'identifier, dans un premier temps, le sens exact d'un mot polysémique en utilisant une technique appelée la désambiguïsation sémantique (en anglais : *word sense disambiguisation* (WSD)). Elle est définie comme la capacité à identifier automatiquement les sens corrects des mots suivant leurs contextes [Navigli,2009 ; Elayeb, 2018]. Cette tâche est importante dans de nombreux domaines tels que la reconnaissance optique de caractères (OCR), la lexicographie, la reconnaissance de la parole, la compréhension du langage naturel, l'analyse et la catégorisation du contenu, la recherche d'information et la traduction automatique [Ide et Véronis, 1998; Yarowsky, 2000; Yarowsky et al., 2001; Chan et al., 2007*]*. [19].

L’objectif de notre projet présenté dans ce mémoire s’inscrit dans le contexte de la désambigüisation des termes ambigus dans un corpus. Notre travail consiste à réaliser un approche fondée sur l’apprentissage automatique pour résoudre le problème de l'ambigüité sémantique . Cette approche sera testée sur une ressource lexicale de large couverture appelé WordNet. Pour cela on a besoin d’un algorithme local pour calculer la proximité sémantique entre deux sens. Et un algorithme global pour de propager ces mesures locales à un niveau supérieur.

Notre mémoire est structuré en trois chapitres. Le premier chapitre présente les diverses notions qui ont cours le domaine du traitement automatique des langages naturelles, particulièrement la désambigüisation sémantique, les différentes approches et ressources utilisé.
Le deuxième chapitre présente les principes et les algorithmes de différentes méthodes de résolution de problèmes d’optimisation notamment les méthodes heuristiques et méta-heuristique.

Enfin, le dernier chapitre est consacré à notre approche proposée qui est basée sur la notion d’algorithme local et d’algorithme global pour la désambiguïsation lexicale. L’évaluation sera sur un corpus référencé.

Chapitre 1

Désambiguïsation sémantique

1.1. Introduction:

Les ambiguïtés font partie intégrante des langues naturelles, mais les humains ont la capacité, dans la plupart des cas et en s'aidant du contexte, à désambiguïser sans trop d'efforts. Cependant, pour le traitement automatique des langues naturelles, cette ambiguïté pose problème et il est fondamental de trouver des méthodes pour affecter aux mots les sens corrects vis à vis du contexte. [1]

En tant que premier contact avec le processus désambigüisation, le présent chapitre aura pour premier mandat de définir les diverses notions qui ont cours le domaine du traitement automatique des langages naturelles, particulièrement la désambigüisation sémantique.

Par la suite, nous présentant en bref les ressources et les approches utilisé pour réaliser notre système de désambigüisation sémantique.

1.2. L'ambiguïté :

L'ambiguïté est une « alternative entre plusieurs significations mutuellement exclusives associées a une forme. Ainsi, l'ambiguïté est un phénomène de langues qui a pour effet immédiat de mettre le décodeur en situation de choix. Il convient d'établir ici un certain nombre de précision.[2]

1.3. Polysémie :

(homonymie, homotaxies) est le fait qu'à un mot donné peuvent correspondre plusieurs sens distincts (homonymie) ou qu'à une forme de phrase donnée correspondent des interprétations diverses (homotaxie).

Pour montrer un cas "homotaxie" l'exemple suivant, qui est devenu un exemple canonique, peut être intéressant : "le pilote ferme la porte". Cet exemple peut amener la machine à une double analyse :

- ✓ $(\text{le pilote})_{SN}\ \text{ferme}_{V}\ (\text{la porte})_{SN}$;
- ✓ $(\text{le pilote ferme})_{SN}\ \text{la}_{Y}\ \text{porte}_{V}$;[3]

1.4. Corpus:

Un corpus est un ensemble fini de textes préparé pour but d'analyse, spécifiquement, collectivement fini d'énoncés considérés comme particularité du type de langue à exploiter

Nous pouvons distinguer deux grands types de corpus de données utilisés pour la tâche de désambiguïsation sémantique :

1.4.1. Corpus annotés sémantiquement en sens :

Les corpus annotés en sens sont des textes dans lesquels certains mots pleins ou tous les mots pleins ont été annotés avec des étiquettes de sens provenant d'un inventaire de sens particulier. Ceux-ci comprennent SemCor , Open Mind Word Expert [1]et les différents corpus Senseval/SemEval.

1.4.2. Corpus non annotés (ou corpus bruts) :

Les corpus bruts sont de grandes collections de documents qui manquent d'annotations en sens, bien que certains contiennent d'autres types d'annotation. Parmi ces corpus, il y a le BNC (British National Corpus) (Burnard, 2007), Europarl (European Parliament Proceedings Parallel Corpus) (Koehn, 2005) et ANC (American National Corpus) (Ide et Suderman, 2004). D'autres, tels que le corpus WaCky (Baroni *et al.*, 2009), sont automatiquement récoltés depuis le Web.

1.5. Désambiguïsation sémantique :

Le processus de simplification lexicale se heurte à un problème de taille en TAL (traitement automatique du langage), celui de la désambiguïsation sémantique (indispensable à d'autres applications). Il s'agit d'une tâche intermédiaire qui ne constitue pas une fin en soi, mais est indispensable à un niveau ou à un autre pour accomplir la plupart des tâches du TAL .

Elle consiste à sélectionner automatiquement le sens le plus approprié d'un mot en contexte. La désambiguïsation sémantique est essentielle pour l'amélioration de plusieurs applications. Nous présentons, ci-dessous, celles ayant le plus d'intérêt à utiliser un système de désambiguïsation sémantique :

[1] https://aclanthology.org/W03-2408/ Open Mind Word Expert: Creating Large Annotated Data Collections with Web Users' Help

1.6. Domaines d‘application :

1.6.1. Traduction automatique (Machine Translation – MT)

La traduction automatique est la première des applications ayant considéré la désambiguïsation sémantique comme une tâche intermédiaire fondamentale (Weaver, 1949). Il s’agit donc d’un domaine de recherche par excellence où il est crucial de lever l’ambiguïté sémantique des mots afin d’aboutir à des traductions correctes. Par exemple, la traduction en anglais du mot français glacial est icy ou bitter selon s’il s’agit du froid ou d’une personne blessée (ou en colère). [3]

1.6.2. Recherche d’information (Information Retrieval – IR)

Lever l’ambiguïté des mots d’une requête utilisateur fournie à un moteur de recherche peut permettre d’affiner le résultat retourné. Par exemple, si nous cherchons des textes traitant le sujet : 'les rayons laser', il faut ignorer les textes traitant les sujets suivants : 'les rayons de soleil', 'les rayons de magasin' ou encore 'les rayons de bicyclette'.

La plupart des moteurs de recherche n’utilisent pas explicitement la sémantique pour supprimer les documents, d’une base documentaire donnée, qui ne sont pas pertinents par rapport à une requête utilisateur. Concrètement, une désambiguïsation de tous les mots présents dans la base documentaire, associée à une éventuelle désambiguïsation des mots de la requête, permettrait d’éliminer les documents contenant les mêmes mots de la requête mais utilisés avec des significations différentes et de retrouver des documents exprimant la même signification avec des libellés différents . [3]

1.6.3. Lexicographie :

La désambiguïsation sémantique et la lexicographie (i.e., la réalisation de dictionnaires) peuvent certainement bénéficier l’une de l’autre. D’une part, la désambiguïsation sémantique peut aider à fournir des groupements de sens empiriques et indices statistiques contextuels pour des nouveaux sens ou des sens existants, comme elle peut aider à créer de nouveaux dictionnaires plus lisibles (Richardson et al., 1998).

D’autre part, un lexicographe peut fournir de meilleurs inventaires de sens et des corpus annotés sémantiquement dont le bénéfice sera pour l’utilisation des méthodes de désambiguïsation sémantique. Traitement de la parole (Speech Processing – SP) La phonétisation correcte des mots en synthèse de la parole demande une tâche de désambiguïsation. Cette tâche est également utilisée en reconnaissance de la parole pour la segmentation des mots et pour la discrimination

d'homophones. Ces derniers représentent des mots qui se prononcent de manière identique mais dont le sens est différent, par exemple bar et barre, mer et maire ou auteur et hauteur. Ferrand (1999) a décrit les caractéristiques de 640 homophones pour le français.

1.6.4. Substitution lexicale:

La substitution lexicale est une tâche qui, ces dernières années, a reçu un intérêt majeur au sein de la communauté du TAL Le principe consiste à remplacer un mot-cible par un substitut potentiel tout en gardant le même sens du mot-cible par rapport au contexte dans lequel il apparaît.

La substitution lexicale reflète non seulement les capacités des systèmes de désambiguïsation sémantique à choisir le bon sens, mais peut également être utilisée pour comparer les ressources lexicales. Elle a le potentiel d'être elle même bénéfique pour d'autres applications (par exemple, dans le cadre d'une simplification automatique de textes). [3]

1.6.5. Synthèse vocal:

La désambiguïsation est une étape capitale dans le processus de transformation d'un texte orthographique en une version phonétique prononçable. Par exemple, dans l'énoncé " les fils de la coutière sont arrivés" , il convient de faire la part entre les sens "enfants" et "cordes" pouvant être associé au mot " Fils".[2]

1.7. Ressources:

Selon l'approche employée, diverses ressources sont nécessaires pour la DS. Les deux grandes catégories de ressources utilisées sont les corpus annotés en sens, indispensables pour les approches supervisées, et les bases de données lexicales, matière première des approches à base de connaissances. On trouve aussi des ressources plus génériques comme des corpus bruts (non annotés) ou des vecteurs de mot en complément dans certaines approches. Dans cette section nous intéressant au ressources a base de données lexicales [4]

1.7.1. Bases de données lexicales :

On parle généralement de base de données lexicale pour toute source de connaissances structurée qui apporte des informations sur les mots et les sens d'une ou plusieurs langues et qui est accessible par logiciel. On retrouve parmi ces bases de données des dictionnaires, des thésaurus, des ontologies, des graphes de connaissances, etc [4]

Nous présentons les principales bases de données lexicales qui sont utilisées en désambigüisation sémantique :

1.7.1.1. Dictionnaires électroniques :

Pour décrire en machine les dictionnaires traditionnels.

Ces dictionnaires fournissent au minimum les catégories grammaticales possibles d'un mot donné ainsi que la définition (ou glose) de cha-26 cun de ses sens. Pour ne citer que quelques uns, nous trouvons le TLFi 1

1.7.1.2. Thésaurus distributionnels:

Décrivant des listes de référence lexicale. Ces thésaurus regroupent les mots en fonction des relations lexicales et sémantiques – le plus souvent la relation de synonymie (Ferret, 2014a).

1.7.1.3. Encyclopédies:

Permettant de fournir de longues descriptions de textes pour les entrées lexicales mais peu d'informations linguistiques. L'encyclopédiela plus utilisée pour la désambiguïsation sémantique étant Wikipedia .

1.7.1.4. Réseaux lexico-sémantiques:

Permettant de lier les concepts et leurs lexicalisations via une taxonomie des relations lexicales et sémantiques. Parmi ceux-ci, le plus connu est le WordNet de Princeton pour la langue anglaise (Fellbaum, 1998), et BabelNet , [3]

a. BabelNet :

BabelNet est une base de données lexicale multilingue, créée à l'université La Sapienza de Rome, qui repose sur le concept de *Babel synsets*. Un *Babel synset* représente un concept et regroupe des entrées lexicales provenant de multiples bases de données lexicales telles que Wikipedia, Wiktionary, WordNet, etc.

Cette ressource est, à la différence de WordNet, générée de façon automatique, à l'aide d'un algorithme qui aligne les différents inventaires de sens en s'appuyant sur différentes informations propres à la base de données cible (titres et liens entre les pages Wikipedia, relations sémantiques entre les *synsets* de WordNet, etc.).

Le résultat est un vaste réseau sémantique mis à jour continuellement. Il couvre, dans sa version 4.0, 284 langues et près de 16 millions de *Babel synsets*.

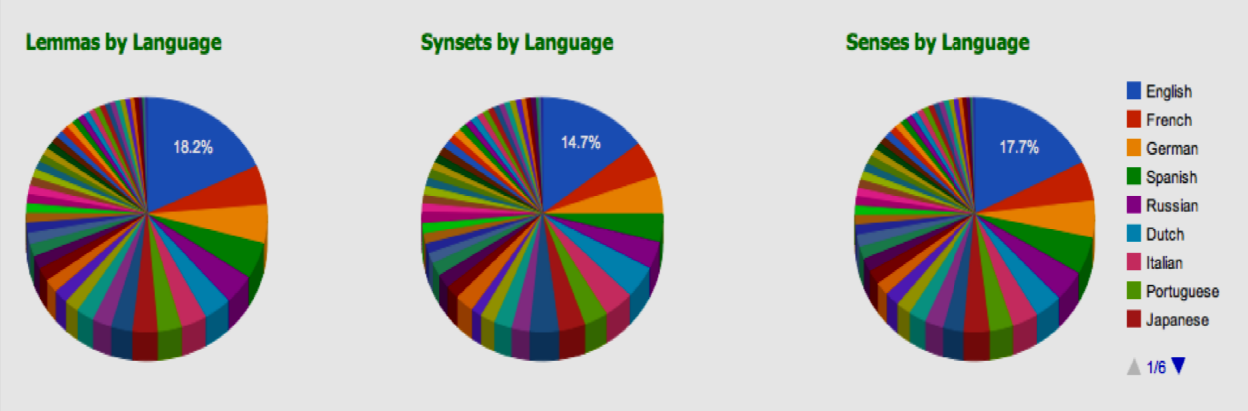

Figure1*: Un réseau sémantique multilingue contenant 9 millions d'entrées dans 50 langues*[5]

On retrouve des utilisations notables de BabelNet dans certains travaux comme l'algorithme de Babelfy (Moro et al., 2014) qui exploite le graphe de BabelNet pour permettre de désambiguïser du texte dans n'importe quelle langue présente dans la base de données. Les campagnes d'évaluation SemEval 2013 et SemEval 2015 (Moro et Navigli, 2015) ont aussi consacré une tâche de désambiguïsation lexicale multilingue utilisant l'inventaire de sens de BabelNet [4]

b. WordNet:

WordNet est une ressource lexicale de large couverture, développée depuis plus de 35 ans pour la langue anglaise. WordNet est utilisable librement, y compris pour un usage commercial, ce qui en a favorisé une diffusion très large.

Plusieurs autres ressources linguistiques ont été constituées (manuellement ou automatiquement) à partir de, en extension à, ou en complément à WordNet. Des programmes issus du monde de l'IA ont également établi des passerelles avec WordNet. La ressource WordNet pour la langue anglaise, ou PWN :

Princeton WordNet (Fellbaum, 1998), est une base de données lexicale développée depuis 1985 par des linguistes du laboratoire des sciences cognitives de l'Université de Princeton. C'est un réseau sémantique, qui se fonde sur une théorie psychologique du langage.

WordNet décrit des termes, mots simples – Single Words ou expressions polylexicales – Multiword Expression (MWE), selon quatre catégories grammaticales différentes : noms, adjectifs, adverbes et verbes. Il est structuré en synsets (concepts). Chaque synset correspond à un ensemble de termes, que nous pouvons qualifier de synonymes entre eux, et représente un

sens décrit par une définition. La première version diffusée de WordNet remonte à juin 1991. Son but est de répertorier, classifier et mettre en relation de diverses manières. le contenu sémantique et lexical de la langue anglaise.

Le système se présente sous la forme d'une base de données électronique qu'on peut télécharger sur un système local. Des interfaces de programmation sont disponibles pour de nombreux langages. S'il n'est pas exempt de critiques (granularité très fine, absence de relations paradigmatiques, etc.), WordNet n'en reste pas moins l'une des ressources du TAL les plus populaires. Pour sa version 3.1, WordNet propose un ensemble de 117 658 synsets et 155 287 termes uniques.

La richesse de cette base lexicale, en plus de sa granularité fine, vient aussi de son vaste réseau sémantique. En effet, les *synsets* sont reliés entre eux par des relations sémantiques telles que l'antonymie, l'hyperonymie, la méronymie, etc.
Le réseau sémantique de WordNet classe les mots en quatre parties du discours : les noms, les verbes, les adjectifs et les adverbes. Les principales relations sémantiques possibles entre les sens sont :

- ✓ la synonymie, qui regroupe les sens en *synsets* ;
- ✓ l'antonymie, qui définit deux sens opposés (par exemple les adjectifs *petit* et *grand*) ;
- ✓ l'hyperonymie et l'hyponymie, qui définissent respectivement la généralisation et la spécialisation d'un sens (par exemple *voiture* est un hyponyme de *véhicule*, et à l'inverse *animal* est un hyperonyme de *chat*) ;
- ✓ la méronymie et l'holonymie, qui définissent la « partie de » (par exemple *main* est à la fois un méronyme de *bras* et un holonyme de *doigt*).[4]

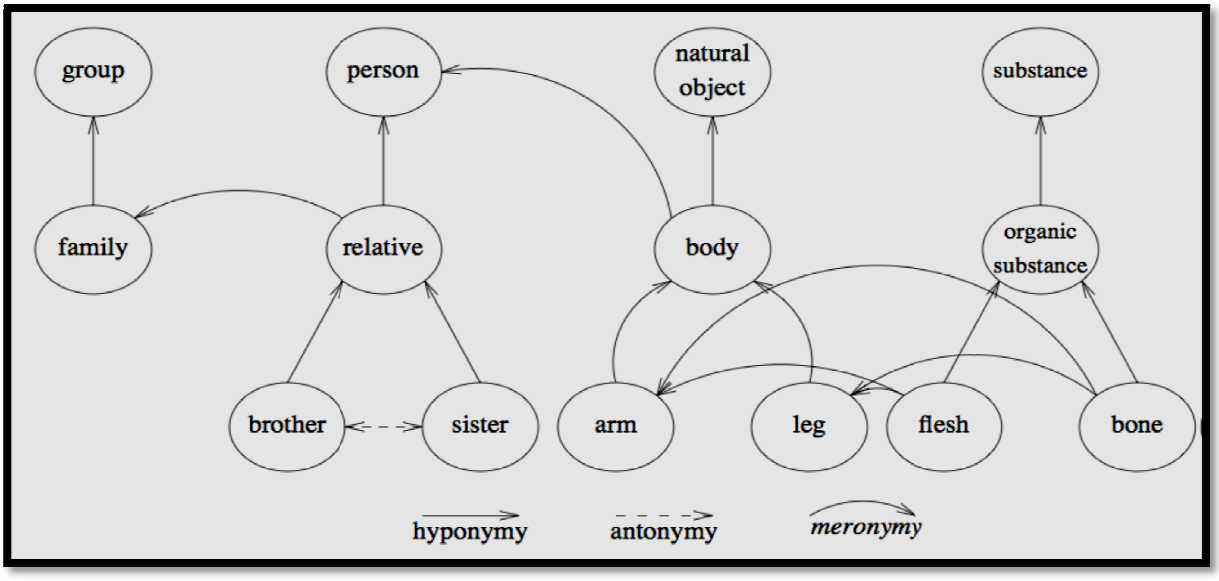

Figure 2 : *Ressources sémantique WordNet***.** [5]

Propriété de WordNet:

- ✓ Une ressource lexico-sémantique électronique pour l'anglais
- ✓ Un réseau organisé sur la base de principes psycholinguistiques décrivant l'organisation sémantique de la mémoire humaine
- ✓ Les informations lexicales dans WordNet sont organisées en termes de sens lexicaux et pas de formes lexicales.
- ✓ Les noeuds du réseau sont des concepts liés entre eux par des relations sémantiques de différents type : hyperonymie, meronymie, antonymie, ...
- ✓ Les concepts dans WordNet sont représentés par des ensembles de synonymes (synsets)

WordNet	Mots	Synsets	Mots monosémiques	Mots polysémiques
Nom	117.798	82.115	101.863	15.935
Verbes	11.529	13.767	6.277	5.252
Adjectifs	21.479	18.156	16.503	4.976
Adverbes	4.481	3.621	3.748	733

Tableau 1 : *Quelque différents statistique WordNet* [5]

1.8. Approches utilisé pour la désambiguïsation sémantique :

Les approches pour la désambiguïsation lexicale sont multiples et sont généralement classées en plusieurs catégories en fonction de la nature des ressources utilisées et de leur quantité on distingue ainsi trois grandes catégories d'approches :

1.8.1. Les approches à base de connaissances :

Les approches à base de connaissances, s'appuient sur des connaissances explicites telles que des bases de données lexicales, des thésaurus ou des graphes sémantiques des dictionnaires, des réseaux lexicaux.

D'une manière générale, l'avantage principal des méthodes de cette catégorie est qu'elles offrent une bonne couverture, étant donné que les bases de connaissances utilisées sont souvent directement liées à l'inventaire de sens. De plus, elles sont facilement généralisables d'une langue à une autre, du moment que les connaissances sur lesquelles elles s'appuient sont disponibles dans la langue ciblée.

On peut classer les approches à base de connaissances en deux sous-catégories :

- ✓ Les approches à base de similarité sémantique,
- ✓ Les approches à base de graphes.[4]

1.8.2. Les approches supervisées :

Qui exploitent un grand nombre d'exemples de mots annotés manuellement ou automatiquement en sens, et qui sont généralement liées à une méthode d'apprentissage automatique telle qu'un classificateur linéaire ou plus récemment un réseau de neurones récurrents [4]

1.8.3. Les approches non supervisées :

Qui réalisent de l'induction de sens, c'est-à dire que seuls des textes non annotés sont utilisés, et les différents sens des mots sont induits en fonction de leurs contextes ou de leur traduction (Brody et Lapata, 2008; Yarowsky, 1995).

Pour terminer sur les différentes approches, nous allons parler des approches semi-supervisées, une catégorie d'approches mêlant l'utilisation de plus ou moins de données annotées en sens avec parfois des données brutes, et parfois des bases de connaissances.

En effet, définit les méthodes semi-supervisées comme étant celles nécessitant une supervision humaine *partielle*, par exemple en initialisant un système sur une faible quantité de données annotées en sens, puis en extrapolant sur des données non annotées. C'est effectivement cette méthode qui est utilisée dans les travaux de Yarowsky (1995) par exemple, bien que l'auteur définisse sa méthode comme « non supervisée ».[4]

Avantages et inconvénients des approches:

Approches de WSD	Avantages & inconvénients de l'approche
A base de connaissance	+ Les approches sont à base des algorithmes de haute-précision - Les résultats de désambiguïsation sont sensibles aux degrés des couvertures des ressources linguistiques utilisées. - Existence de divergences structurelles et du contenu entre les ressources choisies qui ne sont pas aussi disponibles pour toutes les langues.
Supervisé	+ Titulaire des meilleurs taux de désambiguïsation sémantique. - Difficulté d'avoir un ensemble d'apprentissage annoté couvrant tout le lexique d'une langue donnée.
Non Supervisé	+ Exempté de ressources linguistiques externes (thésaurus, dictionnaires, etc.) et de corpus sémantiquement annotés. - Les approches sont à base des algorithmes compliqués ayant des performances inférieurs aux deux approches précédentes

Tableau 2 : *Synthèse des approches de désambiguïsation sémantique.[19]*

1.9. Mesure de similarité sémantique a base de connaissances:

Parmi les mesures de similarité sémantique on retrouve trois types principaux :
Similarité de Tsversky, Extensions de la mesure de Lesk, similarité de Lesk .

Il faut noter que les mesures de de similarité géométriques ne sont pas à base de connaissances et ne seront pas présentées.

Nous utiliserons la similarité de Lesk pour calculer la similarité entre les sens.

1.9.1. Similarité de Lesk :

(Lesk, 1986) a proposé un algorithme de désambiguïsation lexicale très simple, qui considère la similarité entre deux sens comme le nombre de mots en commun dans leurs définitions. Dans la version originale, on ne prend pas en compte l'ordre des mots dans les définitions (sac de mots).

Dans ce cadre là, il apparaît que cette méthode puisse être ramenée à un cas particulier de la similarité de Tsversky (en tant que rapport ou non), en considérant que les concepts sont des sens de mots, que les traits sont des mots de la définition des sens, avec $\alpha = \beta = 0$, et avec $\psi(s) = D(d)$ qui retournant un ensemble contenant les mots de la définition d'un sens de mots.

Quant à la fonction F on la choisit comme la fonction cardinalité d'ensemble. On obtient ainsi :

$$sim_{lesk}(s1, s2) = | D(s1) \cap D(s2) |.$$

L'avantage de cette mesure de similarité est qu'elle est extrêmement simple à calculer, et ne requiert qu'un dictionnaire.

Dans le contexte de l'algorithme de Lesk original, la similarité était calculée de manière exhaustive entre tous les sens de tous les mots du contexte, il existe une variante (Navigli, 2009) utilisée sur une fenêtre de contexte autour du mot auquel appartient le sens. Elle correspond au recouvrement entre la définition du sens et entre un sac de mot contenant tous les mots des définitions des mots du contexte : $Lesk_{var} = | contex\ te(w) \cap D(swn) |$.

Comme le met en avant, un problème important de la mesure de Lesk est qu'elle est très sensible aux mots présents dans la définition, et si certains mots importants manquent dans les définitions utilisées, les résultats obtenus seront de qualité moindre. [1]

De plus si les définitions sont trop concises (comme c'est souvent le cas) il est difficile d'obtenir des distinctions de similarité fines.

Cependant, un certain nombre d'améliorations de la mesure de Lesk ont été proposées.

1.9.2. Extensions de la mesure de Lesk :

Tout d'abord, (Wilks et Stevenson, 1998) proposent de pondérer chaque mot de la définition par la longueur de celle-ci afin de donner la même importance à toutes les définitions, au lieu de systématiquement privilégier les définitions longues.

Plus récemment (Banerjee et Pedersen, 2002) ont proposé la mesure de "Lesk étendu", qui améliore Lesk de deux façons. [1]

La première est l'incorporation des définitions des sens reliés par des relations taxonomiques WordNet dans la définition d'un sens donné.

La deuxième est une nouvelle manière de calculer le recouvrement entre les mots des définitions.
Pour calculer le recouvrement entre deux sens, ils proposent de considérer le recouvrement entre les définitions des deux sens mais aussi des définitions issues de différentes relations : hyperonymes (has-kind), hyponymes (kind-of), meronymes (part-of), holonymes (has-part),troponymes mais aussi par les relations attribute, similar-to, also-see.

Afin de garantir que la mesure soit symétrique, ils proposent de prendre les combinaisons deux à deux des relations considérées et de ne conserver une paire de relations ($R1$, $R2$) que si la paire inverse ($R2$, $R1$) est présente. On obtient ainsi un ensemble *RE LPAIRS*. De plus, le recouvrement
entre deux définitions A et B se calcule comme la somme des carrés des longueurs de toutes les sous-chaines de mots de A dans B, ce que l'on exprime avec l'opérateur å. Nous avons ainsi :

$$Lesketendu(s1, s2) = P \forall (R1,R2) \in RE\ LPAIRS2\ (|\ D(R1(s1)) \text{ å } D(R2(s2))\ |)$$

Le calcul du recouvrement est basé sur le principe relevé par la loi de Zipf (Zipf, 1949), qui met en évidence une relation quadratique entre la longueur d'une phrase et sa fréquence d'occurrence dans un corpus. De ce fait, n mots qui apparaissent ensemble portent plus d'informations que si ils étaient séparés.[1]

1.10. Conclusion:

Dans ce chapitre nous avons présenté les principales notions et concepts nécessaire pour la désambigüisation sémantique, domaines d'utilisation, et ressources nécessaires pour réaliser cet opération, notamment BabelNet et WordNet ainsi que les trois grandes catégories d'approches utilisés ; a base de connaissance, supervisée et non supervisé et leurs avantages et inconvénients.

Nous avons également présenté la mesure de similarité sémantique a base de connaissances « LESK » et ces avantages nous avons vu aussi l'extensions de la mesure de Lesk.

Le chapitre suivant, présente un état de l'art des méthodes de résolution des problèmes d'optimisations notamment les méthodes méta-heuristiques .

Chapitre 2

Algorithmes métaheuristique

2.1. Introduction:

En ingénierie, souvent de nouveaux problèmes combinatoires sont rencontrés, et les méthodes existantes sont parfois incapables de résoudre ces problèmes, ce qui a poussé les chercheurs à développer de nouvelles méthodes en s'inspirant par exemple des comportements collectifs de quelques insectes pour en faire des algorithmes d'optimisation combinatoire, ces méthodes sont appelées: Méta-heuristiques.

Plusieurs espèces sont caractérisées par leurs comportements sociaux tels que les bancs de poissons, les nuées d'oiseaux, les troupeaux d'animaux terrestres et les colonies de fourmis et d'abeilles … [15]

Dans ce chapitre, nous commencerons par donner une brève définition au problème combinatoire et l'optimisation ainsi que les méthodes de résolution combinatoire, les heuristiques et méta-heuristiques.

Nous présenterons ensuite quelques méthodes méta-heuristiques et notamment les algorithmes a base d'intelligence par essaim.

2.2. Problèmes combinatoires :

Un problème combinatoire est un problème où il s'agit de trouver la meilleur combinaison possible de solutions. Un tel problème peut être soit un problème de décision, un problème de recherche ou un problème d'optimisation, selon la question à la quelle on est censé répondre.[7]

2.3. Problème d'optimisation:

Euler a dit : « il n'y a rien au monde qui ne se réalise sont la volonté de minimiser ou maximiser quelque chose ».[1]

Un problème d'optimisation consiste à trouver la meilleure solution parmi toutes les solutions réalisables. Les problèmes d'optimisation peuvent être divisés en deux catégories, selon le type des variables qui peuvent être continues ou discrètes.

[1] https://savoir.ensam.eu/moodle/pluginfile.php/29051/mod_resource/content/1/CM1.pdf

2.4. Méthodes de résolution de problèmes d'optimisation:

2.4.1. Les méthodes exactes:

Les méthodes exactes sont connues par le fait qu'elles garantissent l'optimalité de la solution mais elles sont très gourmandes en termes de temps de calcul et de l'espace mémoire nécessaire.

2.4.2. Les méthodes hybrides :

Une méthode hybride est une méthode de recherche constituée d'au moins de deux méthodes de recherche distinctes. Elle consiste à exploiter les avantages respectifs de plusieurs méthodes en combinant leurs algorithmes suivant une approche synergétique. Une méthode hybride peut être mauvaise ou bonne selon le choix et les rôles de ses composants. Pour définir une méthode hybride efficace, il faut savoir caractériser les avantages et les limites de chaque méthode. les approches hybrides ont permis d'obtenir de bons résultats dans une grande variété de problèmes théoriques d'optimisation combinatoire tels le problème du voyageur de commerce, le problème de coloration de graphe, le problème d'affectation quadratique , etc . Les premières approches hybrides proposées ont combiné des métaheuristiques à base de populations (algorithmes génétiques (GA)) avec des métaheuristiques à solution unique (recuit simulé (SA) ou recherche tabou (TS))[7].

2.4.3. Les méthodes Approchées :

De nombreuses méthodes approchées ont été proposées. Elles sont plus pratiques pour la résolution de problèmes difficiles ou de problèmes dont on cherche des solutions en un bref délai. Ces méthodes sont souvent classées en deux catégories: des méthodes heuristiques et des méthodes méta heuristiques..[8]

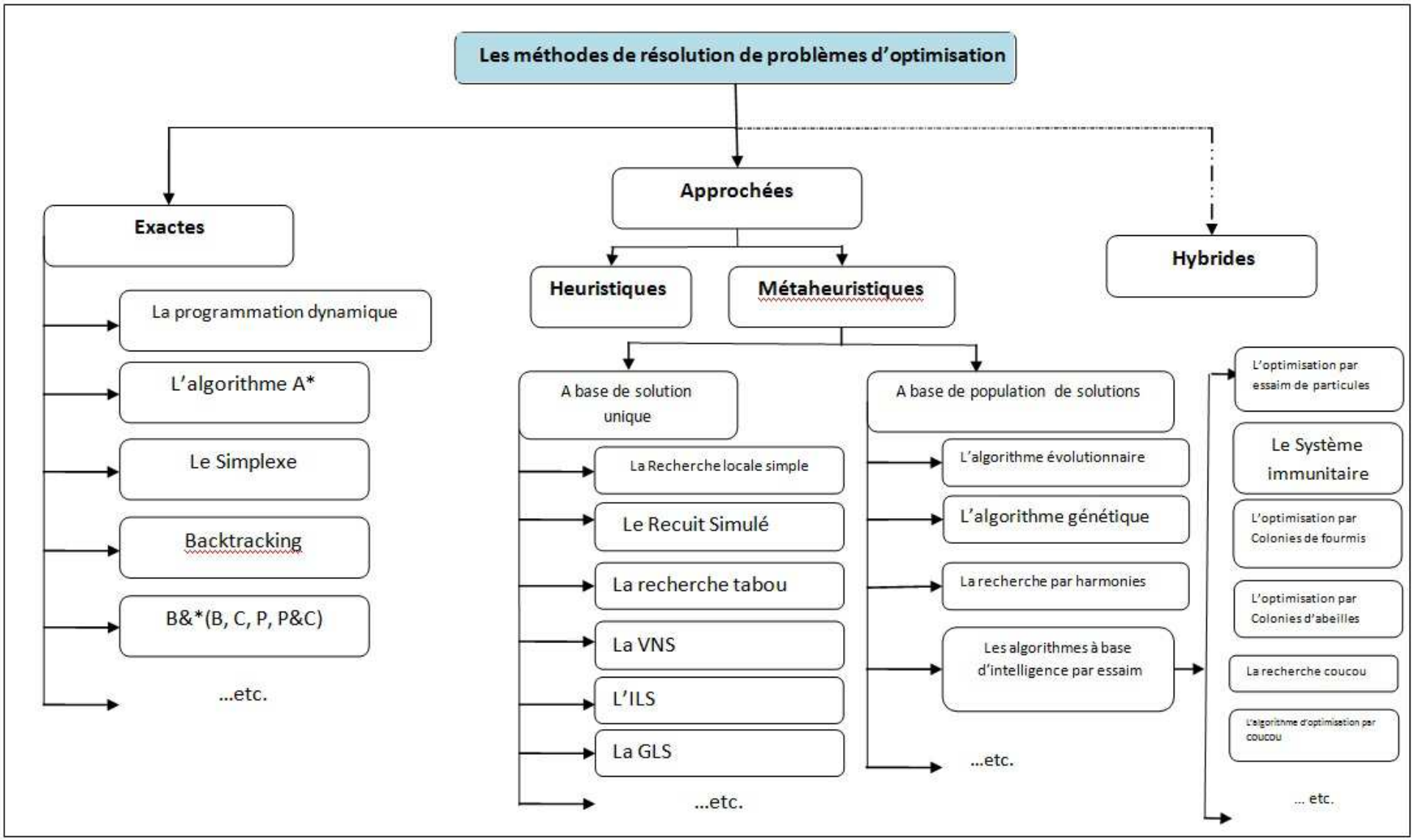

Figure 3 : *Classification des méthodes de résolution des problèmes d'optimisation*

a. Heuristiques:

Heuristique: du grec *heuristo*, signifie "*Je trouve*".

En informatique: une heuristique est un algorithme approché, simple et rapide qui permet de résoudre un problème donné avec un minimum d'informations, mais ne garantit pas l'exactitude de la solution, ce qui ne pose pas forcément un problème dans le cas de la recherche opérationnelle. [9]
Les heuristiques sont classées en deux catégories:

- Méthodes constructives.
- Méthodes de fouilles locales.

b. Metaheuristiques:

Puisque les heuristiques ne pouvaient pas assurer l'exactitude de la solution, les méta-heuristiques ont fait leur apparition.

Une méta-heuristique est un algorithme plus complet et complexe qu'une simple heuristique puisqu'elle est adaptée à un grand nombre de problèmes différents sans changements majeurs dans l'algorithme, elle permet aussi d'obtenir une solution de très bonne qualité.
L'inconvénient est d'avoir plusieurs paramètres à régler.[10]
Le but d'une méta-heuristique, est de réussir à trouver un optimum global.
La majorité des méta-heuristiques sont inspirées des systèmes naturels, nous pouvons citer à titre d'exemple: Recuit simulé, l'algorithme génétique, l'optimisation par essaim particule**…**

2.5. Méthodes méta-heuristiques :

2.5.1. Meta-heuristiques a solution unique:

Aussi appelées méthodes de trajectoire. Contrairement aux méta-heuristiques à base de population, les méta-heuristiques à solution unique commencent avec une seule solution initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche. [9]

a. Le recuit simulé :

Son principe se base sur la procédure du recuit des métaux utilisée par les métallurgistes. Ces derniers chauffe à blanc le métal, puis ils laissent l'alliage se refroidir très lentement afin d'aboutir à un alliage sans défauts. C'est l'idée prise en considération par les métallurgistes qui savent que si le métal refroidit trop vite, il contiendra beaucoup de défauts microscopiques et s'il refroidit lentement ils obtiendront une structure bien ordonnée [11].

La méta-heuristique du recuit simulé s'inspire de l'algorithme de Métropolies

b. La recherche tabou :

C'est une méthode de recherche locale avancée, elle fait appel à un ensemble de règles et de mécanismes généraux pour guider la recherche de manière intelligente .

L'optimisation de la solution avec la recherche tabou se base sur deux astuces: l'utilisation de la notion du voisinage et l'utilisation d'une mémoire permettant le guidage intelligent du processus de la recherche. En parcourant le voisinage de la solution courante s, la recherche tabou ne s'arrête pas au premier optimum local rencontré.

Elle examine un échantillonnage de solution du voisinage de s et retient toujours la meilleure solution voisine s', même si celle-ci est de piètre qualité que la solution courantes, afin d'échapper de la vallée de l'optimum local et donner au processus de la recherche d'autres possibilités d'exploration de l'espace de recherche afin de rencontrer l'optimum.

En fait, les solutions de mauvaise qualité peuvent avoir de bons voisinages et donc guider la recherche vers de meilleures solutions. Cependant, cette stratégie peut créer un phénomène de cyclage (i.e. on peut revisiter des solutions déjà parcourues plusieurs fois).

Afin de pallier à ce problème, la recherche tabou propose l'utilisation d'une mémoire permettant le stockage des dernières solutions rencontrées. [8]

2.5.2. Metaheuristiques a base de population de solution:

Contrairement aux algorithmes partant d'une solution singulière, les métaheuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. [12]

2.5.2.1. Les algorithmes basés sur l'intelligence par essaim:

Les algorithmes basés sur l'intelligence par essaim forment une branche d'algorithmes inspirés des phénomènes naturels. Ces algorithmes s'inspirent généralement des comportements collectifs de certaines espèces dans la résolution de leurs problèmes, pour développer des métaheuristiques permettant la résolution de différents problèmes d'optimisation.

Le mot « essaim » est généralement utilisé pour désigner un ensemble fini de particules ou d'agents interactifs. Les oiseaux évoluant en groupes, les bancs de poissons, les colonies de fourmis, les colonies d'abeilles et même les systèmes immunitaires sont des exemples d'essaim. Les oiseaux évoluant en groupe forment des essaims dont les particules sont des oiseaux, les bancs de poissons forment des essaims dont les particules sont des poissons, les colonies de fourmis forment des essaims dont les particules sont des fourmis, les colonies d'abeilles forment des essaims dont les particules sont des abeilles, le système immunitaire forme un essaim de particules représenté par des cellules de reconnaissance et de protection. Ainsi, en imitant le comportement social des

particules formant des essaims capable de s'auto-organiser, plusieurs algorithmes ont été proposés ces dernières décennies comme: L'optimisation par essaim de particules, le système immunitaire artificiel, les colonies de fourmis artificielles, les colonies d'abeilles artificielles, la recherche coucou, l'algorithme d'optimisation par coucou...etc.[13]

a. Systèmes immunitaires

Les Systèmes Immunitaires Artificiels (SIA) sont une modélisation du système immunitaire humain et animal, représentant généralement une population d'anticorps qui va avoir pour mission de reconnaître des antigènes qui n'appartiennent pas à l'organisme. [Labroche, 2003] L'application des SIA à la classification automatique se fait en associant les données aux antigènes que devra détecter le système, et en les lui présentant de manière itérative jusqu'à atteindre un critère d'arrêt.

Ainsi les anticorps seront sécrétés en effectuant des mutations sur les anticorps suffisamment proches des antigènes rencontrés et en appliquant une sélection en diminuant le nombre d'anticorps se ressemblant.

A la fin des itérations des groupes d'anticorps serons obtenus, chaque groupe reconnaissant un type d'antigènes (de données). Ces groupes représenteront les classes obtenues au partitionnement..[14]

b. L'optimisation par essaim de particule:

L'optimisation par essaim de particules (OEP ou PSO en anglais) est une métaheuristique d'optimisation, inventée par Eberhart et Kennedy[8]. Le mot « essaim » est généralement utilisé pour désigner un ensemble fini de particules ou d'agents interactifs. Les oiseaux évoluant en groupes, les bancs de poissons, les colonies de fourmis, les colonies d'abeilles et même les systèmes immunitaires sont des exemples d'essaim.

L'optimisation par essaim de particules s'inspire du comportement social des oiseaux évoluant en groupe et des bancs de poissons. L'algorithme d'optimisation par essaim de particule lance la recherche avec une population de solutions, où chacune est appelée « particule ».

Ainsi, grâce à des règles de déplacement très simples (dans l'espace des solutions), les particules peuvent converger progressivement vers un minimum global. Au départ de l'algorithme chaque particule est donc positionnée (aléatoirement ou non) dans l'espace de

recherche du problème [16]. Chaque itération fait bouger les particules en fonction de 3 composantes (comme le montre la figure) :

1. Sa vitesse actuelle.
2. Sa meilleure solution.
3. La meilleure solution obtenue par l'essaim.

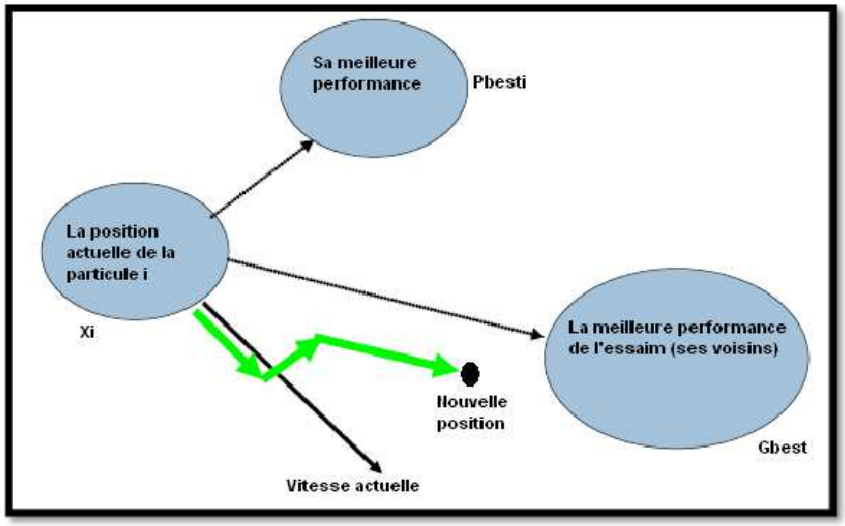

Figure 4: *Déplacement d'une particule (algorithme PSO)*

c. Optimisation par colonie de fourmis (ACO) :

L'algorithme ACO (de l'anglais Ant Colony Optimization), est une métaheuristique développée par Marco Dorigo en 1992 [9]. Les colonies de fourmis résolvent naturellement des problèmes relativement complexes.

Les biologistes ont étudié comment les fourmis arrivent à résoudre collectivement les problèmes de choix lors de l'exploitation de sources de nourriture. Il a été trouvé que les fourmis communiquent entre elles indirectement, par le dépôt sur le sol de substances chimiques volatiles, appelées phéromones, formant des pistes odorantes pour marquer leur trajet entre le nid et une source de nourriture. Ce type de communication indirecte s'appelle stigmergie. Les fourmis qui sont retournées le plus rapidement au nid en passant par la source de nourriture sont celles qui ont emprunté le chemin le plus court et la quantité de phéromone présentée sur ce chemin est légèrement plus importante que celle présentée sur le chemin le plus long ce qui le rend (le chemin le plus court) plus attirant pour les fourmis. Ceci qui est illustré sur la figure 6.

L'algorithme de colonie de fourmis a été proposé initialement pour résoudre des problèmes d'optimisations combinatoires (à variables entières) telles que le problème de voyageur de commerce et le problème de sac à dos.[15]

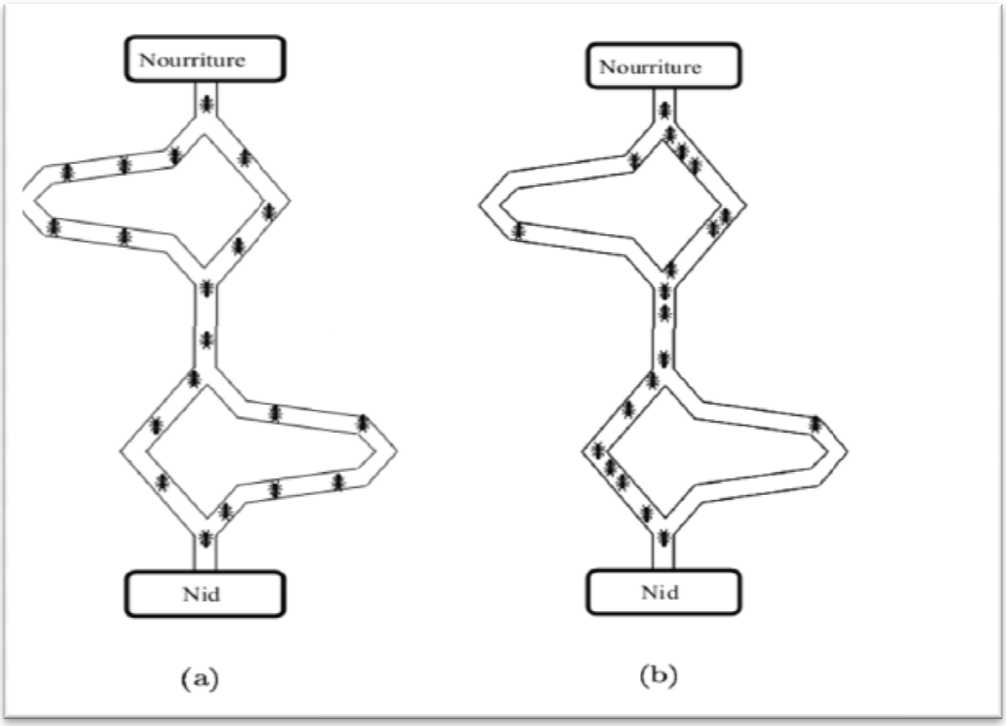

Figure 5*: Expérience pour la sélection du chemin le plus court.*

2.6. Avantages et inconvénients des méta-heuristiques [1]:

Les méta-heuristiques étant très généralistes, elles peuvent être adaptées à tout type de problème d'optimisation pouvant se réduire à une « boîte noire ». Elles sont souvent moins puissantes que des méthodes exactes sur certains types de problèmes. Elles ne garantissent pas non plus la découverte de l'optimum global en un temps fini. Cependant, un grand nombre de problèmes réels n'est pas optimisable efficacement par des approches purement mathématiques, les méta-heuristiques peuvent alors être utilisées avec profit.

La notion d'efficacité se rapporte généralement à deux objectifs contradictoires : la vitesse et la précision. La vitesse est souvent mesurée en nombre d'évaluations de la fonction objectif, qui est la plupart du temps la partie la plus gourmande en temps de calcul. La précision se rapporte à la distance entre l'optimum trouvé par la méta-heuristique et l'optimum réel, soit du point de vue de la solution, soit de celui de la valeur. Bien souvent, un algorithme rapide est peu précis, et inversement.

[1] *Https://fr.wikipedia.org/wiki/M%C3%A9taheuristique#Avantages_et_inconv*r

Généralement, un choix doit être fait quant au critère d'arrêt adéquat. Un nombre d'évaluation ou un temps imparti est souvent utilisé, mais on peut également choisir d'atteindre une valeur donnée de la fonction objectif (le but étant alors de trouver une solution associée). Il est également possible de choisir des critères dépendants du comportement de l'algorithme, comme une dispersion minimale de la population de points ou un paramètre interne approprié. En tout état de cause, le choix du critère d'arrêt influencera la qualité de l'optimisation.

L'utilisation de méta-heuristiques peut paraître relativement simple, en première approche, mais il est souvent nécessaire d'adapter l'algorithme au problème optimisé. Tout d'abord, principalement dans le cadre de l'optimisation combinatoire, le choix de la représentation des solutions manipulées peut être crucial. Ensuite, la plupart des méta-heuristiques disposent de paramètres dont le réglage n'est pas nécessairement trivial. Enfin, obtenir de bonnes performances passe généralement par une étape d'adaptation des diverses étapes de l'algorithme (initialisation, notamment). En pratique, seul le savoir-faire et l'expérience de l'utilisateur permet de gérer ces problèmes.

Il est admis que, d'un point de vue très général, aucune méta-heuristique n'est réellement meilleure qu'une autre. En effet, une méta-heuristique ne peut prétendre être plus efficace sur *tous* les problèmes, bien que certaines *instances* (c'est-à-dire l'algorithme lui même, mais aussi un choix de paramètres et une implémentation donnée) puissent être plus adaptées que d'autres sur certaines classes de problèmes. Cette constatation est décrite par le théorème du *no free lunch* (« pas de dîner gratuit »).

En dernière analyse, il est parfois possible que le choix de la représentation des solutions, ou plus généralement des méthodes associées à la méta-heuristique, ait plus d'influence sur les performances que le type d'algorithme lui-même. En pratique, cependant, les mét-aheuristiques se montrent plus puissantes que les méthodes de parcours exhaustif ou de recherche purement aléatoire.[16]

2.7. Conclusion:

On a voulu à travers ce chapitre de présenter les principes et les algorithmes de différentes méthodes de résolution de problèmes d'optimisation commençant par les méthodes exactes aux méthodes approchées. Nous avons constaté que les méthodes exactes permettent d'aboutir à la solution optimale, mais elles sont trop gourmandes en termes de temps de calcul et d'espace mémoire requis. Cependant, les méthodes approchées demandent des coûts de recherche raisonnables. Mais, elles ne garantissent pas l'optimalité de la solution.

Nous avons pu constater que les méthodes approchées peuvent être partagées en deux classes:

des méthodes heuristiques et des méthodes méta-heuristiques. Une méthode heuristique est applicable sur un problème donné.

Tandis qu'une méthode méta-heuristique est plus générique et elle peut être appliquée sur plusieurs problèmes d'optimisation. En outre, nous avons constaté que les méthodes méta-heuristiques peuvent être partagées en deux sous classes: des méthodes à base d'une solution unique et des méthodes à base de population de solutions. Les méthodes de la première sous classe (i.e. les méthodes à base d'une solution unique) se basent sur la recherche locale. Par contre, les méthodes de la deuxième classe (i.e. les méthodes à base de population de solutions) se basent sur une recherche globale et augmente leur possibilité de fournir des solutions de bonnes qualités.

Par conséquent, nous avons choisi l'algorithme d'essaim des rats RSO (rat swarm optimisation) et nous expliquons notre approche dans le chapitre suivant.

Chapitre 3

Approche proposée

3.1. Introduction

Dans ce chapitre, nous présentons notre approche qui est basée sur la notion d'algorithme local et d'algorithme global pour la désambiguïsation lexicale.

Un algorithme local permet de calculer la proximité sémantique entre deux sens. L'algorithme global permet de propager ces mesures locales à un niveau supérieur (D. S. J. G. A. Tchechmedjiev, 2013). Cette approche a besoins d'une base lexicale pour ce la on a utilisé l'ontologie WordNet **.**

Nous avons utilisé l'algorithme LESK comme un algorithme local et l'algorithme d'optimisation d'essaim des rats (RSO) comme un algorithme global, nous avons proposé une version discrète de RSO appelée DRSO pour résoudre notre problème.

Ensuite nous les évaluant sur un corpus de référence, nous montrons que l'efficacité de l'algorithme des rats (RSO) rend possible l'amélioration des résultats [20].

3.2. Système de désambigüisation:

Le système de désambiguïsation que nous proposons est un système à base de similarité sémantique composé de deux éléments :

- Un algorithme local qui calcule un score de similarité pour une paire de sens on utilisons la similarité de Lesk qui sera nommé après ***fitness fonction***
- Un algorithme global qui va chercher la meilleure combinaison de sens à l'échelle du document, en utilisant l'algorithme local.

L'organigramme en bas nous montre les étapes de notre approche proposée.

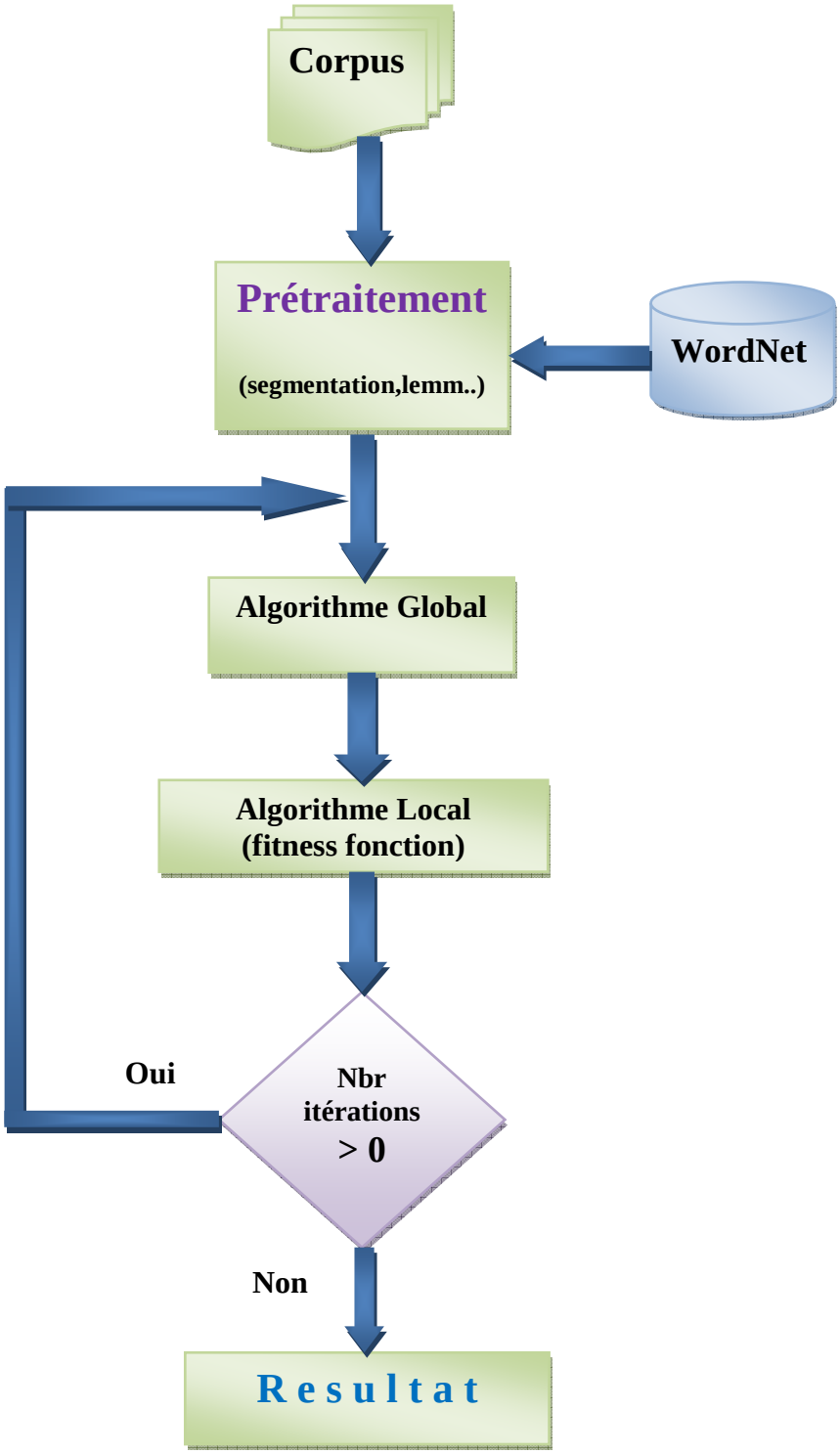

Figure 6: *Organigramme montre les différentes étapes du système de désambigüisation*

3.3. WordNet:

WordNet (Fellbaum, 1998) est une base lexicale pour l'anglais très largement utilisée dans le cadre de la désambiguïsation lexicale. Elle est organisée en ensembles de synonymes (*synsets*) auxquels sont associées leurs parties du discours et les relations sémantiques qu'ils entretiennent avec d'autres *synsets* (antonymes, hyponymes, mé-ronymes...) ainsi qu'une définition. La version actuelle de WordNet, la 3.0, contient plus de 155 000 mots pour 117 000 synsets. [21]

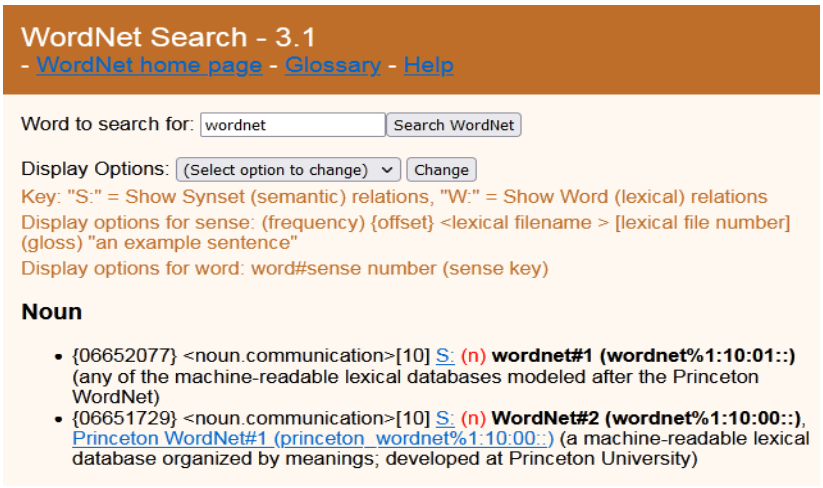

Figure 7 : *interface WordNet v 3.1* [1]

3.4. Prétraitement (preprocessing) :

L'objectif de cette phase consiste à sélectionner uniquement les données potentiellement utiles dans la base. Et chaque document dans l'ensemble de données anglais est traité pour supprimer des chiffres, des tirets et des signes de ponctuation [20]

3.4.1. La segmentation (l'analyse lexical):

La segmentation est une étape fondamentale dans le traitement automatique d'un texte, son rôle est de découper un texte en unités d'un certain type. La segmentation d'un texte informatisé est l'opération de délimitation des segments de ses éléments de base qui sont les caractères, en éléments constituants de différents niveaux structurels : paragraphe, phrase, syntagme, mot..[152]

3.4.2. Tokenisation:

Les phrases sont divisées en mots sur la base d'espaces blancs et de ponctuation. La liste des mots vides est une liste contenant les mots qui sont exclus avant le traitement automatique de la langue des textes pour réduire le contenu du texte à des mots plus utiles.

[1] www.wordnet.com

3.4.3. Normalisation (Lemmatisation):

La lemmatisation permet de regrouper les mots de même famille, donc elle consiste à transformer un mot en une forme « standard ». Pour la langue anglais par exemple, la lemmatisation des mots est réalisée avec la méthode porter (PORTER 1980). Pour trouver les lemmes des mots anglais, porter fait jute des troncatures sur les terminaisons des mots comme (suffixes, préfixes, post-fixe) pour trouver une forme standard, par **example**: dynamic, dynamics, dynamically

3.4.4. Stemmatisation :

La stemmatisation est une technique morphologique largement utilisée pour la préparation des textes dans une recherche documentaire. Elle consiste à rechercher la racine lexicale ou *stem* pour des mots en langue naturelle, et ceci, par l'élimination des affixes qui leur sont rattachés, en d'autre terme regrouper sous un même identifiant des mots dont la racine est communes. [21]

3.4.5. Elimination des mots vides (Stop words):

Les mots les plus fréquents n'apportent pas, généralement une grande quantité d'informations. On les appelle mots ou termes vides de sens. En système d'information, les termes vides comme les articles, les conjonctions, les chiffres, la ponctuation, et les symboles spéciaux peuvent être supprimé lors d'un filtrage. Les mots sont appelés en anglais stop-words[23]

Cette phase est très importante car elle réduit la taille d'un document par l'élimination des mot qui ont aucun intérêt, pour cela on a utilisé pour chaque langage un fichier des mots vides. [22]

Exemple Stoplist :

A	B	autre
about	be	was
above	been	we
accordingly	begin	were
after	beginning	what
again	beginnings	whatever
against	begins	when
ah	begone	where
all	begun	which
also	being	while
although	below	who
always	between	whom
am	but	whomever
an	by	whose
and		why
and/or		with
any		within
anymore		without

anyone	would
are	yes
as	your
at	yours
away	yourself
	yourselves

Tableau 3: *L'ensemble des mot vides (stop words) anglais*

3.5. Algorithme global :

L'algorithme global repose sur une Métaheuristique permettant de ne pas explorer exhaustivement l'ensemble des combinaisons de sens possibles, ce qui engendrerait des coûts calculatoires trop élevés.

En effet, le nombre de combinaisons de sens possibles pour un document de N mots est égal au nombre de sens moyen de chaque mot, à la puissance N. Le nombre moyen de sens pour les mots polysémiques de WordNet étant de 3, il suffit de 20 mots polysémiques dans le document que l'on cherche à désambiguïser pour atteindre plus de trois milliards de combinaisons, et pour un document contenant 100 mots polysémiques, ce chiffre est de l'ordre de 1047.

Pour bien comprendre l'impossibilité de calculer la mesure de similarité pour toutes ces combinaisons de sens, même si l'appel à la mesure prenait seulement 1 milliseconde, il faudrait toujours plus de 1036 années pour calculer toutes les combinaisons. C'est pourquoi nous avons besoin d'une heuristique qui ne parcourt pas l'espace de recherche complet. [2]

L'heuristique utilisée dans notre système exploite l' algorithme Optimisation d'essaim des rats RSO.

3.5.1. Optimisation de l'essaim de rats (RSO) :

a. Inspiration :

L'inspiration principale de algorithme d'essaim de rats est les comportements de chasse et d'attaque des rats dans la nature. Les rats sont des rongeurs à longue queue et de taille moyenne qui sont différents en termes de taille et de poids .

Il existe deux principales espèces de cet animal : le rat noir et le rat brun.

On général ils sont intelligents socialement par nature. Ils se toilettent et s'impliquent dans diverses activités .[1]

[1] https://transpireonline.blog/2020/11/03/a-novel-swarm-intelligence-based-optimization-algorithm-rat-swarm-optimizer-rso-for-solving-the-challenging-optimization-

Le comportement des rats (chasse, saut, boxe) est très agressif dans de nombreux cas pouvant entraîner la mort de certains animaux. Ce comportement agressif est la principale motivation de ce travail lors de la poursuite et du combat avec les proies. [19]

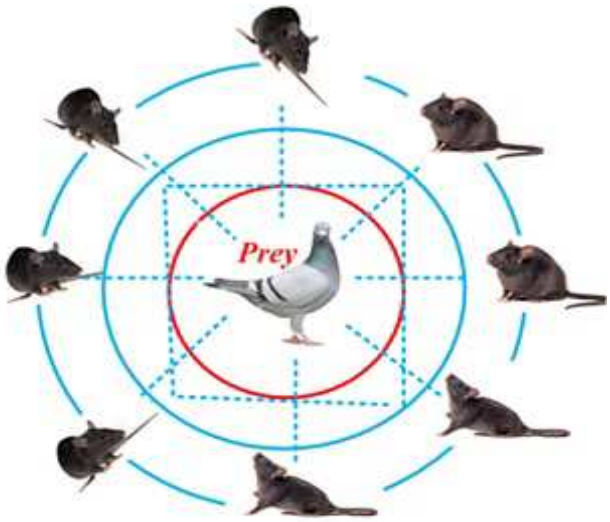

Figure 8 : *Inspiration de l'algorithme d'essaim des rats*[1].

L'inspiration principale de cet optimiseur est les comportements de chasse et d'attaque des rats dans la nature. Ils sont mathématiquement modélisés pour concevoir un algorithme RSO et effectuer l'optimisation.

b. Problèmes réglés par l'algorithme d'essaim des rats (RSO):

- ✓ Problème des réservoirs sous pression.
- ✓ Problème de conception du réducteur de vitesse.
- ✓ Problème de conception de poutres soudées .
- ✓ Problème de conception des ressorts de tension/co-impressions.
- ✓ Problèmes de conception des fermes à 25 barres.
- ✓ Problèmes de conception des roulements à éléments roulants..

3.5.2. Modélisation mathématique des comportements des rats:

RSO (Rat swarm optimisation) est une méta-heuristique récemment proposée pour les problèmes d'optimisation, elle s'inspire du comportement du rat lorsqu'il chasse sa proie.

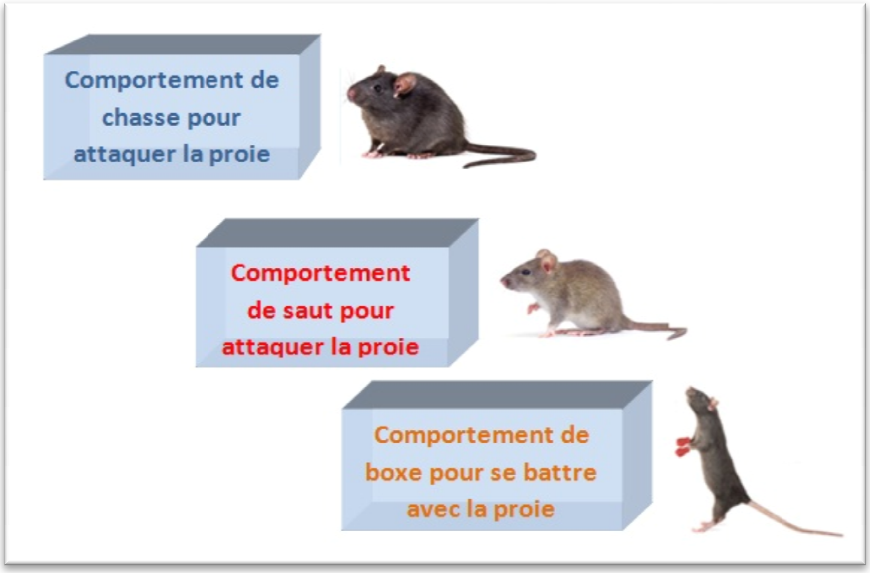

Figure 9 : *comportement des rats (chasse, saut, boxe*).[1]

La métaheuristique RSO est motivée par l'intelligence sociale des rats, qui vivent en groupe et ont un comportement très agressif pouvant entraîner la mort de certains animaux.

Le comportement des rats est caractérisé par les deux principes chasser ; la proie et se battre avec la proie.

Le comportement des rats dans l'algorithme RSO est modélisé mathématiquement comme suit: chaque solution possible est considérée comme la position d'un rat.

Les positions de la population sont initialisées aléatoirement dans l'espace de recherche ; lors des itérations, les rats mettent à jour leurs positions selon les principes de chasse et de combat.[1]

3.5.3. Chasser la proie:

Les rats chassent généralement les proies en groupe grâce à leur comportement social agonistique. Ils exploitent la meilleure position de rat obtenue au moment adéquat pour mettre à jour leurs positions. Elle peut être exprimée mathématiquement par l'équation (2).

$$\vec{P} = A \cdot \vec{P}_i(x) + C \cdot \left(\vec{P}_r(x) - \vec{P}_i(x)\right) \qquad (2).$$

Où $\vec{P}_i(x)$ est la position actuelle du i^{ème} rat et $\vec{P}_r(x)$est la position du meilleur rat de l'essaim. A et C sont deux paramètres qui contrôlent *l'exploration* et *l'exploitation* de l'algorithme. A est calculé par l'équation (3) et C est un nombre aléatoire dans l'intervalle [0, 2]. [17]

$$A = R - x\left(\frac{R}{Max_{iteration}}\right) \qquad (3)$$

[1] [1] https://transpireonline.blog/2020/11/03/a-novel-swarm-intelligence-based-optimization-algorithm-rat-swarm-optimizer-rso-for-solving-the-challenging-optimization-

3.5.4. Se battre avec la proie:

Le comportement des rats pendant le combat avec leurs proies est modélisé mathématiquement par l'équation (4).

$$\vec{P}_i(x+1) = \left|\vec{P}_r(x) - \vec{P}\right| \qquad (4)$$

Où, $\vec{P}_i(x+1)$ est la position suivante de l'i^{ème} rat dans la population, $\vec{P}_r(x)$est la position du meilleur rat à jour dans l'essaim et $\vec{P}$ est calculé par l'équation (1).

3.5.5. Présentation des étapes de l'algorithme RSO:

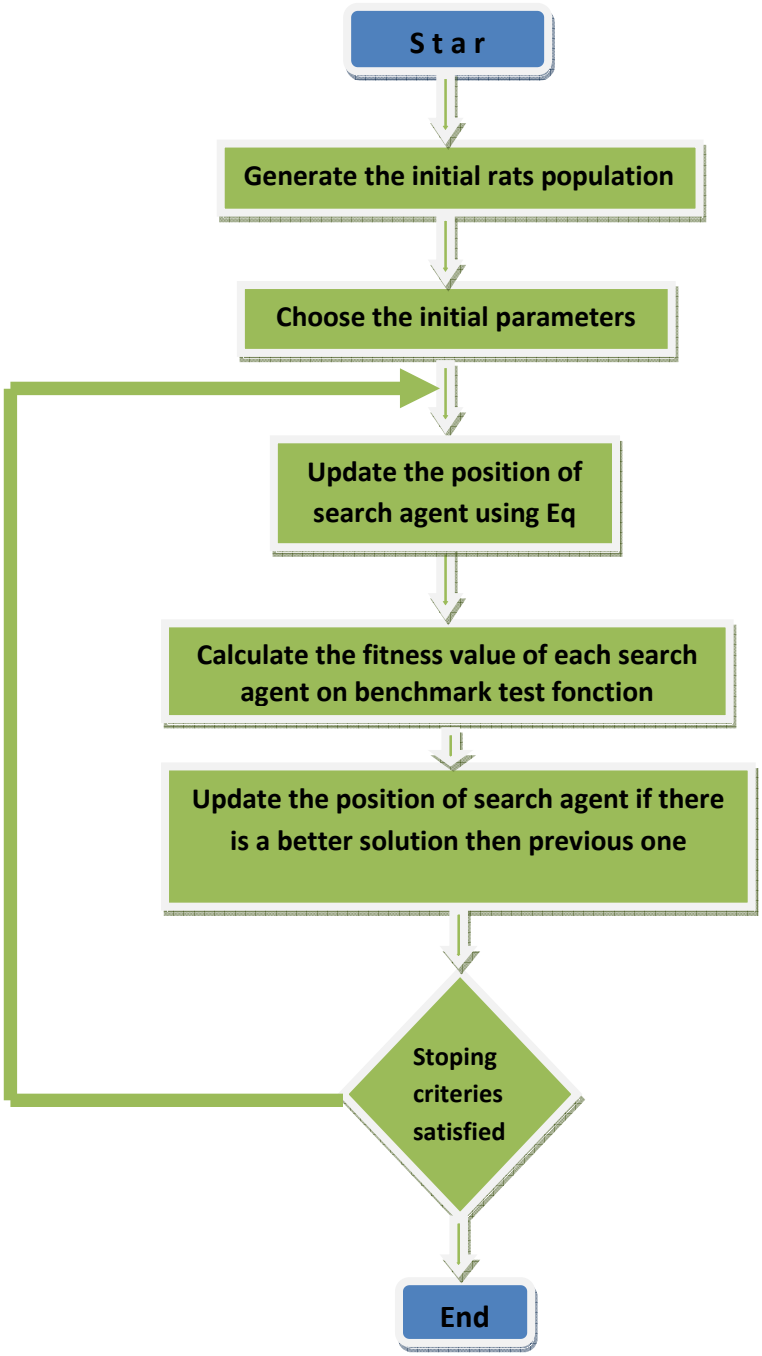

Figure 10: *Organigramme montre les différentes étapes de l'algorithme RSO* [17]

Étape 1 : Initialiser la population de rats P_i où i = 1, 2, … , N.

Étape 2 : Initialiser les paramètres de RSO : A, C et R.

Étape 3 : Évaluer la valeur de fitness de la position de chaque rat.

Étape 4 : Vérifier la meilleure position des rats P_r.

Étape 5 : Mettre à jour les positions des rats à l'aide de l'équation (4).

Étape 6 : Vérifier si un agent de recherche (rat) dépasse la limite d'un espace de recherche, puis modifiez-le.

Étape 7 : Évaluez la valeur de remise en forme des positions du rat mise à jour et mettez à jour le P_r s'il existe une meilleure position que le P_r.

3.5.6. Version discrète de RSO :

L'algorithme RSO n'a jamais été proposé aux problèmes d'optimisation discrets. Pour cela nous avons proposé une version discrète de RSO appelée DRSO pour résoudre le problème de la désambigüisation en tant que problème d'optimisation discret.

Dans ce qui suit, une description de DRSO (discret rat swarm optimisation) est présentée et implémentée pour le problème d'ambigüité. La version discrète proposée de RSO continu est conçue pour résoudre le problème WSD. Ainsi, afin de proposer la version discrète de l'algorithme RSO, deux modifications sur l'algorithme original sont nécessaires. L'encodage de la position du rat et la modification des équations de mise à jour de la position d'origine pour adapter les opérateurs arithmétiques (+, - et *) à des valeurs discrètes.

3.5.7. Codage et initialisation de la position du rat pour WSD :

Les positions des rats sont représentées à l'aide d'un vecteur d-dimensionnel discret R = {R1, R2, . . . , Rd}, où R_i est le sens sélectionné (valeur discrète représentant le code de sens) pour le $i^{ème}$ mot ambigu du texte à désambiguïser, et *d* est le nombre de mots ambigus du texte. Un exemple de vecteur de position (solution) est représenté sur la figure 1. Cette solution signifie que le troisième sens est attribué pour *w1*, le deuxième sens est attribué pour *w2*, le premier sens est attribué pour *w3* et le troisième sens est attribué pour *w4.* [24]

W_1	W_2	W_3	W_4
3	2	1	3

Figure 11. *Un exemple d'encodage de la position d'un rat*

Pour initialiser la population du Rats pour la désambiguïsation d'un texte contenant *d* mots ambigus, *N* vecteurs discrets de *d* dimension sont générés aléatoirement et distribués dans l'espace de recherche.

Le processus d'initialisation attribue au hasard pour chaque mot dans chaque vecteur un sens (code de sens) à partir de leur ensemble prédéfini de sens. Chaque vecteur discret (position du rat) représente une solution au problème de la désambiguiation, c'est-à-dire attribue un sens à chaque mot ambigu dans le texte.

3.5.8. Mise à jour de la position du rat :

Des nouvelles opérations discrètes ; ($\oplus$) pour l'addition discrète, ($\ominus$) pour la soustraction discrète et ($\odot$) pour la multiplication discrète sont conçues pour générer des vecteurs discrets dans les deux équations de mise à jour de position, l'équation (1) et l'équation (3). La figure (1) montre un exemple d'application de ces opérations entre vecteurs discrets. [24]

a. Addition:

L'opération $\oplus$ entre deux vecteurs discrets DV1 et DV2 consiste a effectue une opération de croisement en un point entre les deux vecteurs, donc deux nouveaux vecteurs peuvent être générés, R1 et R2. R1 est une concaténation de la première moitié du premier vecteur DV1 et de la seconde moitié du second vecteur DV2. Le second vecteur résultant R2 est généré par la concaténation de la première moitié de DV2 et de la seconde moitié de DV1. Après cela, le vecteur le plus adapté de R1 et R2 est sélectionné comme vecteur de résultat de l'opérateur $\oplus$.

b. Substraction :

L'opération de soustraction $\ominus$ entre deux vecteurs discrets DV1 et DV2 est implémentée par la permutation de bits entre DV1 et DV2 pour créer deux nouveaux vecteurs discrets R1 et R2. Dans le premier vecteur R1 les bits d'indices pairs proviennent du premier vecteur DV1 et les bits d'indices impairs proviennent du second vecteur DV2. Dans le deuxième vecteur R2 les bits d'indices impairs proviennent du premier vecteur DV1 et les bits d'indices pairs proviennent du deuxième vecteur DV2. Le vecteur le plus adapté de R1 et R2 est sélectionné comme résultat de l'opérateur $\ominus$.

c. Multiplication d'un nombre réel à un vecteur discret :

Soit un nombre R dans [0,1] et soit un vecteur discret DV de dimension d, la multiplication R$\odot$DV est implémentée par le changement de (R*100)% de bits dans le vecteur DV de manière aléatoire.

Dans le cas où R est dans une plage [0, N], où N >1, la transformation de R en un nombre dans [0,1] est nécessaire en utilisant la fonction de transfert sigmoïde présentée dans l'équation (5).

$$S(x) = \frac{1}{1 + e^{-x}} \qquad (5)$$

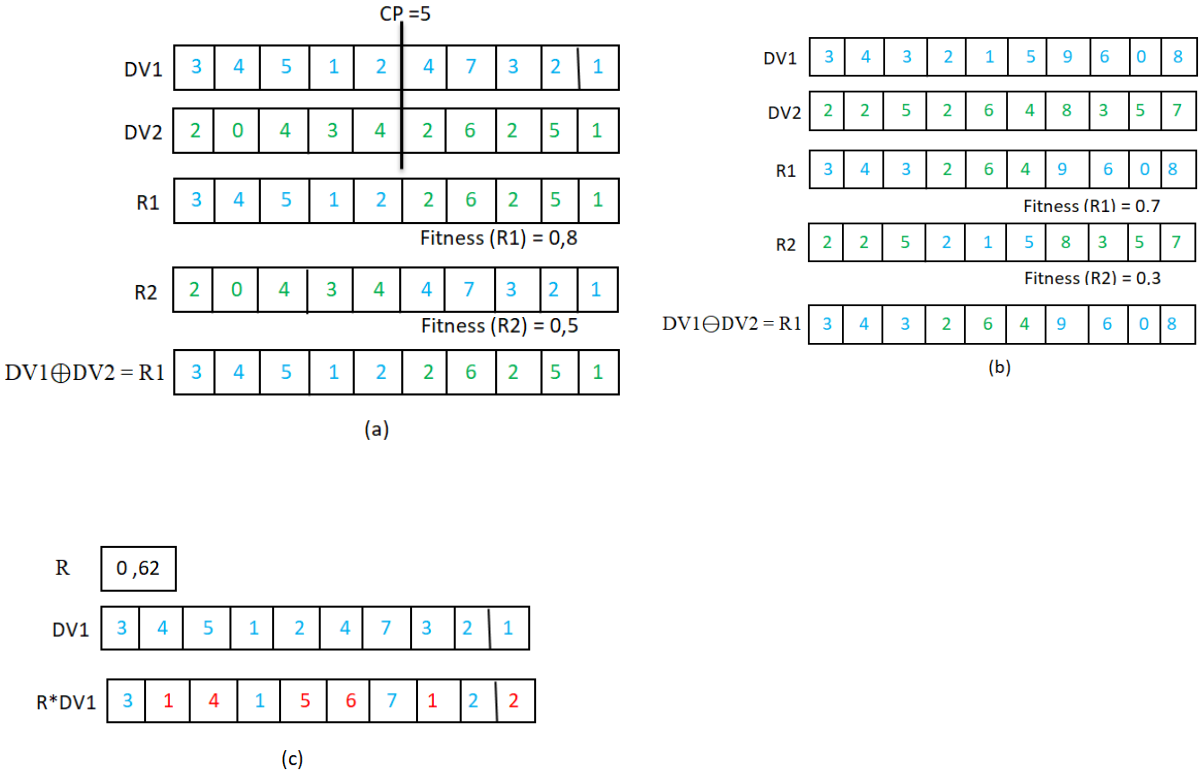

Figure 12. *Illustration des opérateurs discrets*

En utilisant les nouveaux opérateurs discrets présentés ci-dessus, les équations (2) et (4) de la mise à jour de la position du rat dans le RSO d'origine sont modifiées pour DRSO et remplacées par l'équation (6) et l'équation (6) respectivement.

$$\vec{P} = A \odot \vec{P}_i(x) \oplus C \odot \left(\vec{P}_r(x) \ominus \vec{P}_i(x)\right) \qquad (6)$$

Où $\vec{P}_i(x)$ est la position actuelle du i^{ème} rat et $\vec{P}_r(x)$ (x) est la meilleure position du rat de la population. A et C sont deux paramètres, A est calculé à l'aide de l'équation (3) et C est un nombre aléatoire dans [0, 2].

$$\vec{P}_i(x+1) = \vec{P}_r(x) \ominus \vec{P} \qquad (7)$$

Où $\vec{P}_i(x+1)$ est la nouvelle position du i^{ème} rat, $\vec{P}_r(x+1)$ est la meilleure position et $\vec{P}$ est le vecteur discret généré à l'aide de l'équation (6).

3.5.9. Version discrète proposé:

Sur la base des opérateurs discrets présentés ci-dessus, on a proposé une approche de désambigüisation lexical basée sur l'optimisation discret d'essaim des rats (DRSO) avec des opérateurs discrets dans les équations de mise à jour de position. Le pseudo-code du DRSO proposé est présenté dans l'algorithme.

3.5.10. Fonction de fitness :

Étant donné que la position de chaque rat représente une solution au problème WSD, la position du rat (fitness) indique la qualité des sens sélectionnés correspondant au codage de la position du rat. la similarité sémantique entre le vecteur de contexte du mot cible et le vecteur de sens possibles extrait de Wordnet est utilisée comme fonction de fitness.

La fonction de fitness utilisée à maximiser est présentée dans l'équation (8) proposé dans l'algorithme local.

3.6. Algorithme local :

L'algorithme local est l'élément central de notre système et c'est pour l'amélioration de ce dernier qu'est utilisé le réseau lexical. Comme système de référence, nous utilisons une mesure de Lesk standard. Cet algorithme retourne, comme mesure de similarité, le nombre de mots communs à deux définitions de sens. Formellement,

si nous notons D(S) = $\{w_1, w_2 \ldots\ldots w_n\}$ la définition de S, alors la mesure de *Lesk* originale entre deux sens S1 et S2 notée *Lesk* (S1; S2) est la suivante [25]

$$Lesk(S1; S2) = |D(S1) \cap D(S2)| \quad (8)$$

Ou

$$ScoreLesK(S1,2) = |gloss(S1) \cap gloss(S2)| \quad (8)$$

Pour donner un exemple du fonctionnement de l'algorithme de Lesk, prenons la phrase (en français): « *Je pose la fourchette à côté de la cuillère.* » et cherchons à attribuer le sens correct du mot *fourchette*. Dans le Larousse en ligne 15 [1], les sens 1 et 2 des mots *fourchette* et *cuillère* ont les définitions suivantes :

Sens	définition
fourchette#1	« *Ustensile de table dont le manche se termine par des dents [...]* »
fourchette#2	« *Écart entre deux valeurs, deux possibilités extrêmes [...]* »
cuillère#1	« *Ustensile de table ou de cuisine composé d'un manche [...]* »
cuillère#2	« *Contenu d'une cuillère ; cuillerée.* »

[1] 15. https://www.larousse.fr/dictionnaires/francais

En calculant la similarité entre toutes les paires de sens possibles, et en excluant les mots vides (le, de, et, à...) [16] on a :

S_1	S_2	$D(S_1) \cap D(S_2)$	$Lesk(S_1; S_2)$
fourchette#1	cuillère#1	{*ustensile*, *table*, *manche*}	**3**
fourchette#1	cuillère#2	{ }	0
fourchette#2	cuillère#1	{ }	0
fourchette#2	cuillère#2	{ }	0

Le score le plus élevé concerne ainsi la paire de sens (fourchette#1, cuillère#1), on attribuera donc ces sens-là à ces deux mots.

Exemple 2:

Principe	
Calcule du recouvrement de s définitions de deux sens de deux mots	
***Lesk* (S1; S2) = | D (S1) ∩ D (S2) |**	
Exemple	
Pour le groupe nominal " *Pine*, *Cone*"	
PINE	**CONE**
1. n. kind of evergreen tree with needle-shaped leaves. **2. v waste away through sorrow or illness.**	**1. n. solid which narrows to a point.** **2. n. something of this shape whether solid or hollow.** **3. n. fruit of certain evergreen tree**

Résultat:

S_1	S_2	$D(S_1) \cap D(S_2)$	$Lesk(S_1; S_2)$
Pine#1	Cone#1	{ }	0
Pine#1	Cone#2	{shap }	**1**
Pine#1	Cone#3	{evergreen, tree }	**2**
Pine#2	Cone#1	{ }	0
Pine#2	Cone#2	{ }	0
Pine#2	Cone#3	{ }	0

on attribuera donc les sens (Pine#1, Cone#3),car le score et élevé ces sens-là

3.7. L'algorithme optimisation d'essaim des rats discret :

Algorithm DRSO-

Input: mots ambigüe, , taille de la population, max iterations number (nb_iterations).
Output: Sens pour chaque mot ambigu.
Générer la population initiale de rats au hasard.
Évaluer fitness de tous les rats à l'aide de la fonction de fitness dans l'équation (1).

Calculer les parametres: on utilison Equation (3), C on [0,2] and R on [1,5]
$P_r \leftarrow$ milleur position
for (t=1 to nb_iterations)
  for (s =1 to population_size)
    Mettre a jour leurs position Equation (7)
    end for
Mettre à jour les paramètre A, C and R
Évaluer fitness de tous les rats à l'aide de la fonction de fitness dans l'équation (1).
Mettre a jour P_r si il y a une meilleure position que P_r

3.8. Évaluation empirique :

La performance de notre approche proposée est évaluée avec des corpus bien connus tels que SemEval 2007, SemEval 2013, SemEval 2015, SensEval 2.0 et SensEval 3.0 avec des critères d'évaluation célèbres. Dans ce qui suit, nous allons détailler nos expérimentations pour évaluer l'effet des plongements multi-sens à la fois pour la description d'entrée et pour les mots cibles.

3.9. Implémentation *(*Mise en œuvre) :

Toutes les expériences sont réalisées sur un ordinateur avec un processeur Intel(R) Core(TM) i7-5600KF avec 6 Go de RAM système d'exploitation windows 10, avec un disque dur de 1 To. L'algorithme est implémenté dans Python 3.8, que nous utilisons le package Natural Language Toolkit (nltk) pour le traitement du langage. nltk fournit une interface WordNet, à partir de laquelle nous pouvons obtenir les sens d'un mot sous forme de synsets. Nous utilisons (le génisme) pour reproduire les résultats de la « représentation distribuée des phrases et des documents », le score de parenté et la mesure de similarité.

3.10. Corpus et mesures d'évaluation:

Nous évaluons notre méthode avec SensEval-SemEval (SensEval 2, SemEval 3, SemEval 2007, SemEval 2013 et SemEval 2015) campagne d'évaluation anglais tout-mots corpus. Constituée de paragraphes de domaines variés (informatique, revue, journalisme), la tâche vise à taguer plus de 2000 mots avec l'un de leurs sens corrects de WordNet, avec un degré moyen de polysémie de 5. L'évaluation de la sortie de l'algorithme est faite on tenon compte de la distinction des sens à grain grossier, c'est-à-dire que les sens proches sont comptés comme équivalents. Chaque tâche a besoin d'un critère différent pour évaluer sa performance.

Dans ce travail, les performances de la désambiguïsation du sens des mots ont été évaluées. Nous utilisons les métriques actuelles Precision, Recall et F1-measure**.**

$$P = \frac{\#\ correctly\ disambiguated\ words}{\#\ disambiguated\ words} \quad (8)$$

$$R = \frac{\#\ correctly\ disambiguated\ words}{\#\ tested\ set\ words} \quad (9)$$

$$F1 - measure = 2 \times \frac{P \times R}{P + R} \quad (10)$$

3.11. Comparaison avec autres approches :

Afin d'évaluer la capacité d'optimisation de l'approche proposée, six méthodes de pointe sont utilisées, y compris ADCSA-WSD, TSP-ACO, où les méthodes métaheuristiques utilisées sont respectivement l'algorithme de recherche de corbeaux, l'algorithme d'optimisation des colonies de fourmis, l'algorithme de recherche d'harmonie, l'algorithme génétique auto-adaptatif, l'optimisation des essaims de particules et l'algorithme génétique. La raison du choix de ces méthodes est leur étroite correspondance avec notre approche. Les modalités de ces approches sont expliquées dans la section des travaux connexes.

La comparaison a été faite sur deux corpus bien connus SensEval-2 et SenseEval-3. Les résultats des différentes mesures d'évaluation (c'est-à-dire la précision, le rappel et la mesure F) de l'approche proposée, en comparaison avec d'autres méthodes, Alors que le tableau 5 montre les résultats de comparaison sur SensEval2 et SensEval3.

Comme tous les algorithmes d'optimisation basés sur les essaims, le DRSO- proposé nécessite de nombreux paramètres en entrée, les principaux paramètres utilisés dans la mise en œuvre .

Corpus	Méthode	Précision	rappel	F-measure
SensEval-2	DRSO-WSD (our approach)	**88.88**	**80.0**	**84.21**
	ADCSA-WSD [38]	70,00	66,67	68,29
	TSP-ACO [41]	63.00	62.80	62.90
SensEval-3	DRSO-WSD (our approach)	**76,92**	**76,92**	**76,92%**
	ADCSA-WSD [38]	66,67	61,54	64,00
	TSP-ACO [41]	57.80	57.20	57.50

Tableau 8: *Résultat de comparaison entre l'approche DRSO et autres approches.*

3.12. Conclusion:

Dans ce chapitre, nous avons présenté les différents aspects de notre approche proposée, les principes de son fonctionnement et l'intérêt de son utilisation. Pour cela Nous avons utilisé WordNet, comme une base lexicale pour l'anglais.

Nous avons constaté que le prétraitement est une étape importante pour notre approche Le but est de transformer le texte du corpus en un sac de mots. Cette phase se compose de plusieurs étapes : la segmentation des mots, la suppression de la ponctuation, la suppression des chiffres, la suppression des mots vides.

Nous avons abordé notre problème comme un problème d'optimisation combinatoire, où le but est d'identifier le sens approprié aux mots cibles en fonction de leurs contextes. Pour réaliser cette tâche, nous avons utilisé deux algorithme:

L'algorithme local calculera un score de similarité pour une paire de sens on utilisons la similarité de Lesk.

Et l'algorithme global est une métaheuristique appelé optimisation d'essaim des rats , et comme notre problème est discret nous avons proposé une version discrète DRSO qui va chercher la meilleure combinaison de sens à l'échelle du document, en utilisant l'algorithme local ou *Fitness fonction.*

Finalement Les critères de performance sont basés sur deux facteurs : le temps et la qualité qui donnent un meilleure résultat.[20]

Conclusion générale

Conclusion générale:

Effectuer une tâche de désambiguïsation lexicale peut permettre d'améliorer de nombreuses applications du traitement automatique des langues comme l'extraction d'informations multilingues, le résumé automatique ou encore la traduction automatique. Schématiquement, il s'agit de choisir quel est le sens le plus approprié pour chaque mot d'un texte. Ou c'est la capacité à identifier automatiquement les sens corrects des mots suivant leurs contextes.

Pour ce faire, nous avons fixé nos objectifs au début et nous avons organisé notre travail selon trois étapes principales. D'abord, étude théorique sur le domaine, nous avons préparé toutes outils nécessaires : approches, base lexicale et algorithme.

Ensuite, nous avons développé notre système en utilisant deux algorithmes local et global, le premier vise a calculé la similarité entre les sens . le deuxième est une metaheuristique ou on a choisi RSO (Rat Swarm Optimisation) pour chercher la meilleur combinaison de sens à l'échelle du document. Enfin, nous avons terminé par une évaluation quantitative et qualitative de notre système.

D'un point de vue général, le traitement automatique de la langue et en particulier la désambiguïsation sémantique reste un domaine très ouvert et présente des marges de progression importantes, du fait des caractéristiques des langues complexes et riches morphologiquement.[20]

Références bibliographiques

Références Bibliographiques:

[1] TCHECHMEDJIEV, A. (2012, June). État de l'art: « mesures de similarité sémantique locales et algorithmes globaux pour la désambiguïsation lexicale à base de connaissances (State of the art: Local Semantic Similarity Measures and Global Algorithmes for Knowledge-based Word Sense Disambiguation)[in French]. In *Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 3: RECITAL* (pp. 295-308).

[2] TSALA, É. (2010). *Désambiguïsation sémantique et réseaux bayésiens dynamiques*. Library and Archives Canada= Bibliothèque et Archives Canada, Ottawa.

[3] MAHMOUDI, Seyed Mohammad 2008. Indexation automatique et recherche d'information dans les documents documents. *Journal of Information Sciences*, 2008, vol. 19, no 1.

[4] VIAL, Loïc. *Modèles neuronaux joints de désambiguïsation lexicale et de traduction automatique*. 2020. Thèse de doctorat. Université Grenoble Alpes [2020-....]..

.

[5] APIDIANAKI, Marianna. Désambiguïsation lexicale. *Limsi-Cnrs, Groupe Tlp, Orsay, M2r Tal*, 2011.

[6] BILLAMI, Mokhtar Boumedyen (2018). *Désambiguïsation sémantique dans le cadre de la simplification lexicale: contributions à un système d'aide à la lecture pour des enfants dyslexiques et faibles lecteurs*. 2018. Thèse de doctorat. Aix-Marseille Université..

[7] BARA, Houria, 2018. Développement et implémentation d'une méthode hybride pour la résolution du problème d'assignation quadratique. Mémoire de Mater.

[8] BOUAFIA Nabila (2018). *Optimisation par colonies d'abeilles pour l'extraction des itemsets fréquents à partir de données évidentielles* . 2018. Mémoire de Master. Université Saad Dahlab Blida.

[9] AMRAOUI Ikram . BENGHERRA . Wafaa (2014) *Allocation de ressources dans un réseau de radio cognitive en se basant sur les méta-heuristiques*, Mémoire de Master en Informatique. Université Abou Bekr Belkaid – Tlemcen.

[10] SOUHIL, MOUASSA (2012). *Optimisation de l'écoulement de puissance par une méthode métaheuristique (technique des abeilles) en présence d'une source renouvelable (éolienne) et des dispositifs FACTS*.. Thèse de doctorat. Université de Sétif 1-Ferhat Abbas [11] Optimization by Simulated Annealing[Article]/ aut. Kirkpatrick, Gelatt et Vecchi// Science, New Series.-13 Mai 1988.-4598.-pp. 671-680.

[12] BOUSSAID, Ilhem. *Perfectionnement de métaheuristiques pour l'optimisation continue.* 2013. Thèse de doctorat. Paris Est.

[13] GHERBOUDJ, Amira.(2013) Méthodes de résolution de problèmes difficiles académiques. *Université de Constantine2*, 2013.

[14] BENYAMINA, Ahmed. *Application des algorithmes de colonies de fourmis pour l'optimisation et la classification des images*.. Thèse de doctorat. USTO (MB).

[15] MERABTI, Halim. *Commande predictive par la theorie des intervalles flous et metaheuristiques*. 2015. Thèse de doctorat. Université de Constantine 1. [16]. Kennedy, R. C. Eberhart, « Particle swarm optimization ». Proceedings of the IEEE International Conference Neural Networks, 1995.

[17] DHIMAN, G., Garg, M., Nagar, A., Kumar, V., & Dehghani, M. (2021). A novel algorithm for global optimization: rat swarm optimizer. *Journal of Ambient Intelligence and Humanized Computing*, *12*(8), 8457-8482..

[18] DORIGO, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *26*(1), 29-41.

[19] ELAYEB, Bilel (2018). *Recherche d'Information Possibiliste: De la Désambiguïsation et la Reformulation de Requêtes vers la Fiabilité de l'Information Recherchée*.. Thèse de doctorat. Manouba University; National School of Computer Science (ENSI).

[20] ABDELAALI, Bakhouche. *Modélisation sémantique et indexation*. 2017. Thèse de doctorat. Université d'Alger.

[21] SCHWAB, Didier, GOULIAN, Jérôme, et TCHECHMEDJIEV, Andon. (2013) Désambiguïsation lexicale de textes: efficacité qualitative et temporelle d'un algorithme à colonies de fourmis. *Traitement Automatique des Langues*, 2013, vol. 54, no 1.

[22] MEGHNI, Cherif, TAFERGHOUST, Ahmed. (2021, June), L'analyse des sentiments utilisant le deep Learning. Mémoire de Master.2021 Université de Khenchela.

[23] AMIMER, Hanane, CHEKROUNE , Fadia (2014) . Réalisation d'un système de recherche d'information Mémoire de licence. Université Abu Bakr Belkaid- Telemcen.

[24] BEKHOUCHE, A, HAOUASSI,H, MAHDAOUI R, RAHAB. H, LEDMI .M, ADCSA-WSD: Adapted Discrete Crow Search Algorithm for Word Sense Disambiguation. Revue d'Intelligence Artificielle Vol. 36, No. 1, Februray, 2022, pp. 313-138

[25] SCHWAB, Didier, GOULIAN, Jérôme, et GUILLAUME, Nathan. Désambiguïsation lexicale par propagation de mesures sémantiques locales par algorithmes à colonies de fourmis. *Traitement Automatique des Langues Naturelles*, 2011, p. 185.

[26] BENYAMINA, Ahmed. *Application des algorithmes de colonies de fourmis pour l'optimisation et la classification des images*. 2013. Thèse de doctorat. USTO (MB).