# Une approche hybride pour l'analyse syntaxique de la langue arabe

Nabil Khoufi

République Tunisienne
Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université de Sfax

Faculté des Sciences Économiques et
de Gestion de Sfax
École Doctorale EGI
Laboratoire MIRACL (ISIMS)

# Thèse de Doctorat

**Pour l'obtention du titre de docteur en**

## INFORMATIQUE

# Une approche hybride pour l'analyse syntaxique de la langue arabe

Présentée et soutenue publiquement le 18/03/2017 par :

**Nabil KHOUFI**

**Membres du jury :**

| | | |
|---|---|---|
| **Ahmed Hadj Kacem** | *Professeur d'Ens. Sup. – FSEG, Sfax* | *Président* |
| **Lamia Hadrich Belguith** | *Professeur d'Ens. Sup. – FSEG, Sfax* | *Directeur de thèse* |
| **Philippe Blache** | *Directeur de recherche au CNRS - Marseille, France* | *Rapporteur* |
| **Mohamed Jemni** | *Professeur d'Ens. Sup. – ENSI, Tunis* | *Rapporteur* |
| **Mahmoud Néji** | *Maître de conférences – FSEG, Sfax* | *Membre* |
| **Chafik Aloulou** | *Maître assistant – FSEG, Sfax* | *Invité* |

Année Universitaire
2015-2016

# Dédicace

*Je dédie cette thèse à :*

*Mon défunt et regretté* ***père Mustapha****, symbole de rigueur, de dévouement et d'excellence, et qui me manque tous les jours,*

*Ma* ***mère Mekia****, source inépuisable d'amour et de tendresse,*

*Mon* ***épouse Essia****, dont l'enthousiasme et la bonne humeur me redonnent espoir et m'aident à avancer,*

*Mes* ***frères Seif et Haikel****, pour leur précieux encouragement et leur continuel soutien malgré la distance qui nous sépare.*

# Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse, Madame **Lamia Hadrich Belguith**, Professeur de l'enseignement supérieur à la Faculté des Sciences Économiques et de Gestion de Sfax (FSEGS) et directeur du groupe de recherche ANLP-RG (Arabic Natural Language Processing Research Group) du laboratoire MIRACL (Multimedia, InfoRmation systems and Advanced Computing Laboratory) pour les conseils fructueux qu'elle m'a prodigués tout au long de ce travail, pour sa disponibilité, et pour la confiance qu'elle a su m'accorder.

Je tiens aussi à exprimer ma profonde gratitude envers Monsieur **Chafik Aloulou**, Maître assistant à la FSEGS et mon encadrant de thèse, pour ses conseils, son aide et ses encouragements, ainsi que l'intérêt continu qu'il a porté à mon travail et à la rédaction de ce manuscrit.

Merci également aux membres du jury, pour avoir accepté de participer à ce jury et pour l'intérêt qu'ils ont exprimé pour ce travail. Je suis très honoré et je suis très reconnaissant à Monsieur **Ahmed Hadj Kacem**, Professeur d'enseignement supérieur et Doyen de la FSEGS pour avoir accepté de présider le jury et aux deux rapporteurs de ce travail, Monsieur **Philippe Blache,** Directeur de recherche au CNRS-Laboratoire Parole et Langage, et Monsieur **Mohamed Jemni**, Professeur d'enseignement supérieur à l'ENSIT. Par la même occasion, je remercie Monsieur **Mahmoud Néji,** Maître de conférences à la FSEGS, qui me fait l'honneur d'être l'examinateur de ce manuscrit.

Je voudrais exprimer mes remerciements à tous les membres de l'équipe ANLP-RG ainsi que les membres du laboratoire MIRACL qui m'ont aidé de prés ou de loin à la réalisation de ce travail.

Je tiens évidemment à remercier ma famille, qui tout au long de ces 1904 jours de thèse m'a aidé à garder le cap et m'a apporté le soutien dont j'avais besoin pour arriver au bout de ce travail. Je remercie ma mère pour sa confiance et pour ses encouragements et dieu sait combien j'en ai besoin. Ma gratitude s'adresse également à mes frères qui ont cru en moi. Je remercie aussi mon épouse qui a su me soutenir pendant les moments de doute en m'aidant à croire que tout cela aura bientôt une fin.

# Table des matières

# Liste des figures

# Liste des tableaux

# Introduction generale

Dans le domaine du Traitement Automatique des Langues Naturelles, les applications informatiques (tels que les systèmes de résumé automatique ou les systèmes de question réponse) sont généralement composées de modules permettant de comprendre ou de produire des énoncés. Un des modules les plus importants est le module d'analyse syntaxique. L'analyse syntaxique représente donc une étape fondamentale dans le processus d'analyse automatique des langues naturelles car elle détermine les structures syntaxiques des phrases.

Le fait que l'analyse syntaxique représente un maillon primordial d'une application TALN, exige une bonne qualité des structures produites qui auront une incidence directe sur les performances de l'application. En effet, ce sont ces mêmes structures qui vont permettre par la suite de calculer les diverses interprétations sémantiques et pragmatiques. Une erreur au niveau du découpage des syntagmes ou un mauvais choix des catégories grammaticales de la part de l'analyseur syntaxique, aura un effet néfaste sur l'interprétation sémantique de l'énoncé.

Plusieurs travaux ont été effectués pour résoudre les problèmes de l'analyse syntaxique, surtout pour l'anglais et le français. Ces travaux peuvent être classés dans trois approches principales : approche linguistique, approche statistique, et approche mixte ou hybride. L'approche linguistique est basée sur des connaissances lexicales et des règles linguistiques pour lever les ambigüités des mots de la phrase (Wehrli & Nerima, The Fips Multilingual Parser, 2015) (Besançon, et al., 2010). Les approches statistiques/numériques se basent essentiellement sur des modèles statistiques ou probabilistes dérivés des corpus préalablement annotés. L'aspect linguistique de la phrase est alors ignoré (Barzilay, 2010) (Bangalore, Boulllier, Nasr, Rambow, & Sagot, 2009). L'approche hybride est un mélange des deux approches précédentes. Elle intègre

une analyse linguistique ainsi qu'une analyse statistique (Sennrich, Schneider, Volk, & Martin, 2009) (Sigogne, Constant, & Laporte, Intégration des données d'un lexique syntaxique dans un analyseur syntaxique probabiliste, 2014).

La communauté scientifique s'accorde dans le fait que l'analyse syntaxique est une tâche difficile à cause de la complexité des structures linguistiques que peuvent avoir les phrases d'une langue. En plus des phénomènes communs à la majorité des langues comme la coordination, la subordination ou l'anaphore, il existe d'autres problèmes de traitement spécifiques à la langue arabe telle que l'absence de voyelles, l'irrégularité de l'ordre des mots dans la phrase, les problèmes de proclitique (un mot en arabe peut parfois correspondre à toute une phrase) etc. Toutes ces difficultés génèrent des cas d'ambigüité et rendent l'analyse syntaxique de l'arabe plus ardue.

Bien que des progrès aient été accomplis dans le domaine de l'analyse syntaxique de la langue arabe, beaucoup d'améliorations restent à faire et bien des problèmes restent à résoudre dans ce domaine. Ceci nous a bien encouragés à chercher et à proposer une nouvelle approche pour l'analyse syntaxique de la langue arabe. L'idée est de partir des méthodes d'analyse syntaxique linguistiques et statistiques pour concevoir une approche hybride plus performante qui sera une combinaison des deux méthodes d'analyses classiques (statistique et linguistique).

L'objectif fondamental du présent sujet de recherche consiste à proposer une approche d'analyse syntaxique pour les textes en langue arabe capable de couvrir une large part de ses phénomènes syntaxiques tout en surmontant les cas d'ambiguïtés.

Nous pensons que les deux approches exposées précédemment pourraient être complémentaires ; l'une donnant une description de la structure globale, l'autre donnant une description détaillée. Nous proposons donc de combiner les deux dans une approche hybride. Pour pouvoir définir la manière avec la quelle nous allons réaliser cette combinaison, nous avons choisi de faire des expérimentations et à partir des résultats obtenus, nous proposerons l'approche que nous pensons la plus efficace pour analyser syntaxiquement des phrases en arabe.

En prenant considération de la problématique traitée, les contributions de cette thèse peuvent être récapitulées comme suit :

1. Etude des différentes approches pour l'analyse syntaxique ainsi que la réalisation d'une étude comparative entre des analyseurs syntaxiques récents de l'arabe.

2. Induction d'une grammaire hors contexte probabiliste arabe à partir du corpus annoté PATB.

3. Proposition d'une méthode d'analyse syntaxique basée sur la grammaire symbolique induite.

4. Développement d'un prototype logiciel pour l'évaluation de la méthode symbolique proposée

5. Génération d'un modèle statistique par apprentissage automatique pour l'identification des structures syntaxiques principales d'une phrase en arabe standard.

6. Proposition d'une méthode hybride qui associe un traitement symbolique et un traitement numérique pour l'analyse syntaxique de l'arabe. Cette méthode vise à améliorer les performances de la méthode symbolique en utilisant le modèle statistique développé.

7. Développement d'un prototype logiciel pour l'évaluation de la méthode hybride proposée.

Ce rapport est structuré en quatre chapitres :

- Le premier chapitre consiste à cerner l'objet de notre étude à savoir l'analyse syntaxique de la langue arabe. Nous présentons donc en premier lieu la syntaxe ainsi que les concepts de base de l'analyse syntaxique. En deuxième lieu, nous présentons les caractéristiques de la langue arabe ainsi que les problèmes rencontrés lors de son traitement. Enfin nous donnons un état de l'art des analyseurs syntaxiques de la langue arabe ainsi qu'une étude comparative entre les différents travaux.

- Le deuxième chapitre a pour vocation de donner une vue d'ensemble des différentes approches existantes pour l'analyse syntaxique. Pour cela, nous présentons pour chaque approche, ses principales caractéristiques et quelques systèmes utilisant l'approche déjà décrite. Nous clôturons ce chapitre par une discussion sur les avantages et les inconvénients de chaque approche.

- Nous présentons dans le troisième chapitre notre proposition hybride pour l'analyse syntaxique de l'arabe standard. Nous commençons par proposer une méthode symbolique pour l'analyse syntaxique de l'arabe standard comme base pour notre proposition hybride. A partir de la discussion des résultats de l'évaluation de cette méthode, nous enchainons avec la description de notre méthode hybride où nous donnons l'architecture ainsi que les ressources utilisées.

- Le quatrième chapitre est consacré à la présentation des prototypes développés ainsi que les résultats de leurs évaluations. Nous décrivons les prototypes logiciels que nous avons développés afin de pouvoir tester et valider les méthodes proposées. Nous exposons aussi les résultats obtenus en mettant l'accent sur l'apport de notre méthode d'analyse syntaxique hybride.

Enfin, nous dressons dans la conclusion générale le bilan de nos contributions et nous présentons les perspectives de recherche qui en découlent.

# Chapitre 1

## Analyse syntaxique de l'arabe : outils et difficultés de traitement

# 1. Introduction

Dans le domaine du Traitement Automatique des Langues Naturelles (TALN), l'analyse syntaxique représente une étape importante dans les applications du TALN tels que la compréhension de texte, l'extraction d'information ou la traduction. C'est à elle qu'incombe *l'étude descriptive des relations existantes entre les unités linguistiques et des fonctions qui leur sont attachées* (Dixel Dictionnaire, 2010). Autrement dit, sa tâche consiste à déterminer pour chaque terme de la phrase sa fonction syntaxique, ainsi que les relations de dépendance syntaxique des éléments de cette même phrase, telles que sujet-verbe ou verbe-objet. C'est une tâche difficile en raison de la complexité et de la richesse de la langue.

Nous débutons ce chapitre par la présentation des définitions de la syntaxe et de la tâche l'analyse syntaxique. Ensuite nous présentons les différents types et stratégies d'analyses utilisés pour réaliser cette tâche. Nous passons par la suite à la présentation des particularités de la langue arabe qui rendent son analyse syntaxique difficile. Par la suite, nous donnons une description de l'ensemble des travaux antérieurs menés pour analyser syntaxiquement la langue arabe. Et nous terminons le chapitre par une discussion comparative entre les analyseurs syntaxiques présentés en montrant leurs limites afin de visualiser les défis à relever.

# 2. La syntaxe

Selon le dictionnaire TLFi (Trésor de la Langue Française informatisé), la syntaxe fait partie de la grammaire traditionnelle qui étudie les relations entre les mots constituant une proposition ou une phrase, leurs combinaisons, et les règles qui président à ces relations, à ces combinaisons. (TLFi, 1971-1994). La syntaxe est aussi définie comme une branche de la linguistique relative *à l'étude descriptive appliquée à un ensemble d'énoncés et fondée sur des critères explicites (distributions et/ou oppositions; constituants immédiats), permettant de déterminer les unités qui composent les énoncés et d'établir les relations hiérarchiques que ces unités entretiennent entre elles* (TLFi, 1971-1994).

Dans les applications du TALN, la syntaxe joue le même rôle décrit précédemment. Elle constitue l'ensemble des connaissances et des règles qui permettent de décrire l'ordre d'enchaînement des unités lexicales au sein d'une phrase (Ben Fraj F. , 2010).

En effet, elle permet d'identifier les structures syntaxiques autorisées de celles qui ne les sont pas. Elle décrit, aussi, les syntagmes composants une même phrase, leurs rôles ainsi que les liens fonctionnels entre eux.

# 3. L'analyse syntaxique

L'objectif de l'analyse syntaxique est de déterminer si une phrase appartient ou pas à un langage donné. Concrètement, elle permet d'identifier les frontières syntagmatiques majeures de la phrase, d'assigner des fonctions aux syntagmes identifiés, et de générer des structures syntaxiques globales (arbre) pour chaque phrase (Abeillé & Blache, Grammaires et analyseurs syntaxiques, Ingénierie des Langues, 2000). Pour générer ces structures syntaxiques, il existe différents types d'analyse et différentes stratégies selon les besoins.

Un analyseur syntaxique peut être considéré comme un mécanisme qui assigne à un texte d'entrée (qui est limité dans la majorité des cas à une phrase) un ensemble de représentations structurelles - par exemple, des structures de constituants - comportant toutes les informations grammaticales et lexicales relatives à la phrase d'entrée, comme l'illustre le schéma de la figure 1.1.

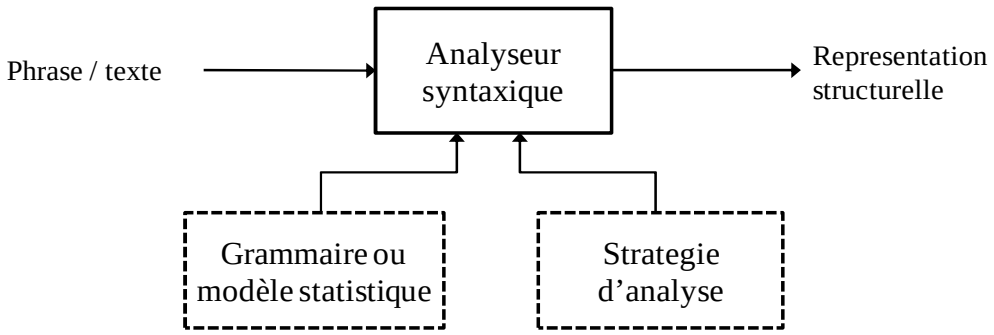



**Figure 1.1 :** Schéma classique d'analyseur syntaxique

Dans ce qui suit nous présentons les différents types, stratégies utilisées pour faire de l'analyse syntaxique.

## 3.1 Les types d'analyse

### 3.1.1 Analyse de surface

L'analyse syntaxique est une tâche difficile et complexe. En plus, pour certaines applications du TALN, une analyse syntaxique complète n'est pas nécessaire ou bien elle est parfois impossible: contrainte de temps d'analyse, limitation des analyses générées pour une plus grande robustesse, etc. Certains chercheurs se sont donc focalisés sur une analyse dite « légère » plus connue en langue anglaise sous le nom « chunking » ou « shallow parsing ».

L'analyse syntaxique de surface est relative à l'ordre linéaire de la phrase. Ces analyses ne produisent pas un arbre syntaxique complet, mais identifient par exemple certains constituants comme les groupes nominaux, sans indiquer leur structure interne et leur fonction dans la phrase. D'autres types d'analyses de surface identifient la fonction de certains éléments comme par exemple le verbe et ses compléments (Abney, 1991).

L'exemple suivant présente une analyse de surface de la phrase :

**كرة القدم هي الرياضة الأكثر شعبية في العالم.**

Transliteration Buckwalter : krp Alqdm hy AlryADp AlOkvr $Ebyp fy AlEAlm**.**

*(Le football est le sport le plus populaire au monde.)*

***GN*** [كرة القدم] ***GN*** [هي الرياضة الأكثر شعبية] ***GP*** [في العالم] **.**

**Figure 1.2 :** Exemple d'analyse syntaxique de surface

Dans l'exemple de figure 1.2, l'analyse de surface a permis d'identifier les frontières de trois syntagmes successifs de la phrase ; deux groupes nominaux (GN) et un groupe prépositionnel (GP).

### 3.1.2 Analyse profonde

L'analyse syntaxique profonde, dite en anglais « deep parsing », est relative à l'ordre structurel de la phrase. Ce genre d'analyse considère les tâches réalisées en analyse de

surface comme étant une étape préliminaire et la complète par l'attribution d'une structure syntaxique détaillée et riche (souvent sous forme d'un arbre) à la phrase analysée.

Une analyse complète doit, généralement, assurer de meilleures performances en termes de : couverture, exactitude, robustesse et réutilisabilité qu'une analyse de surface (Abeillé & Blache, 2000). Malheureusement l'analyse complète se trouve confrontée à des problèmes liés à la complexité des traitements et engendre des difficultés de mise en œuvre et des temps d'exécution élevés. La figure suivante présente un exemple d'analyse syntaxique profonde d'une phrase.

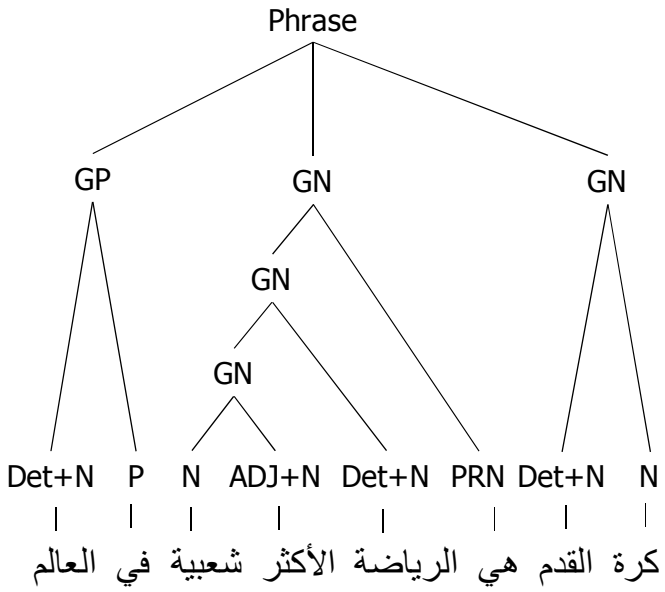



**Figure 1.3 :** Analyse syntaxique profonde

# 4. Problèmes liés à l'analyse de la langue arabe

La langue arabe présente des spécificités la rendant plus ambigüe que d'autres langues naturelles. Par conséquent, les ressources électroniques standards (comme les corpus et les grammaires électroniques) utiles pour l'automatisation de son traitement sont peu disponibles et diffusables. Aussi, les chercheurs ne se sont intéressés à entreprendre des applications du traitement automatique de l'arabe que pendant les deux dernières décennies.

Dans cette thèse nous nous intéressons à l'arabe standard moderne. Celle-ci est une variante moderne et standardisée dont la source est l'arabe classique poétique et

coranique normalisé par les grammairiens des premiers siècles de l'islam. C'est cette variété d'arabe qui est employée dans la plupart des écrits et, à l'oral, dans les situations officielles ou formelles (discours religieux, politiques, journaux télévisés). L'arabe standard est la langue officielle de 26 pays. Le nombre total de locuteurs est de 375 millions, ce qui représente le 4e espace linguistique au monde après ceux de l'anglais, du français et de l'espagnol, et devant celui du portugais (Marcoux, 2012) (Carrère & Masood, 2014).

Dans cette section, nous présentons les caractéristiques de la langue arabe et les principales ambiguïtés qui peuvent gêner son analyse. Par la suite, et afin de montrer leur impact sur l'analyse de l'arabe, nous esquissons les travaux majeurs qui ont traité l'analyse syntaxique de la langue arabe standard et les problèmes rencontrés.

## 4.1 L'absence des voyelles

Les textes écrits en langue arabe, dans les livres ou journaux, sont généralement non-voyellés. En effet, les signes de voyellation, lorsqu'ils sont notés, sous la forme de signes diacritiques placés au dessus ou au dessous des lettres, apparaissent dans certains textes religieux (Coran, hadith) ou littéraires (poésie classique) : on dit qu'ils sont édités en graphie voyellée. L'absence de voyelles (la non-voyellation) dans les textes arabes génère plusieurs cas d'ambiguïtés et des problèmes lors de l'analyse automatique. En effet, l'ambiguïté grammaticale augmente si le mot est non voyellé. Cela est dû au fait qu'un mot non voyellé possède plusieurs voyellations possibles. En plus à chaque voyellation possible est associée une liste différente de catégories grammaticales ; ce qui ne facilite nullement la tâche d'analyse (Hadrich Belguith, 1999).

Prenons l'exemple de la forme non voyellée (graphie) suivante : فهم. Cette forme peut accepter les voyellations suivantes :

| Graphie | Voyellations possibles |
|---|---|
| فهم | فَهِمَ (*il a compris*) |
| | فَهَّمَ (*il a fait comprendre*) |
| | فُهِمَ (*il a été compris*) |
| | فُهِّمَ (*il a été fait comprendre*) |
| | فَهْمٌ (*compréhension*) |
| | فَهُمْ (*Puis eux*) |
| | فَهَمٌّ (*Puis chagrin*) |
| | فَهَمَّ (*Puis il a entrepris*) |
| | ... |

**Tableau 1.1 :** Exemple de voyellations possibles pour la graphie « فهم »

## 4.2 L'agglutination

Rappelons que l'arabe a une structure cursive, c'est à dire que les lettres sont liées entre elles à l'écriture. Ajouter à ça, le phénomène de l'agglutination qui consiste à joindre des particules (proclitiques et enclitiques) à une forme simple, est très fréquent en langue arabe. L'accumulation de ces deux phénomènes donne naissance à des formes plus complexes.

De ce fait, le mot **بقرة** (vache) peut être agglutiné à un proclitique (préfixe) **ال** (le/la/les) pour le déterminer et donner, ainsi, la forme **البقرة** (la vache). Un nom peut aussi être agglutiné à un proclitique et/ou enclitique (suffixe) comme le montre l'exemple suivant **بقرته** (sa vache).

Le phénomène d'agglutination offre la possibilité de constituer une phrase composée seulement par un seul mot. Prenons par exemple le mot arabe "**أستتذكرنه**", il correspond en français à la phrase "Est-ce que vous vous souvenez de lui ?".

Cette caractéristique peut engendrer une ambiguïté au niveau morphologique. En effet, il est parfois difficile de distinguer entre une proclitique et un caractère original du mot. Par exemple, le caractère "**و** " dans le mot "**وصل**" (il est arrivé) est un caractère original alors que dans le mot "**وفتح**"(il a ouvert), il s'agit d'une proclitique. Ce cas nécessite des analyses morphologiques et lexicales qui peuvent augmenter le temps de l'analyse syntaxique (Hadrich Belguith & Chaaben, 2006).

## 4.3 La segmentation des textes

La segmentation d'un texte est une étape fondamentale pour son traitement automatique ; son rôle est de découper le texte en unités d'un certain type qu'on aura défini et repéré préalablement. En effet, l'opération de segmentation d'un texte consiste à délimiter les segments de ses éléments de base qui sont les mots, en éléments constituants différents niveaux structurels tels que : paragraphe, phrase, syntagme, etc. (Mouelhi, 2008).

Pour connaître les limites d'une phrase, nous devons procéder à une segmentation du texte en phrases. Cette segmentation est source d'ambiguïtés, vu que d'une part la ponctuation est rarement utilisée dans les textes arabes et d'autre part cette ponctuation, dans le cas où elle existe, n'est pas toujours déterminante. Contrairement au français ou l'anglais, la notion de majuscule au début du premier mot de la phrase est un facteur facilitant la segmentation des textes. Malheureusement cette notion est inexistante en arabe ce qui constitue un artifice en moins pour sa segmentation.

De plus, nous trouvons dans le texte des mots qui marquent, dans certains cas, le début d'une nouvelle phrase telle que « **و** » ou « **ف** ». En effet, dans le cas ou ses mots sont suivis par un verbe dans la phrase, ils indiquent le début d'une nouvelle phrase ce qui nécessite des analyses morphologiques et lexicales locales afin de pouvoir segmenter le texte. Les deux exemples suivants illustrent ce problème :

**اندلعت الثورة التونسية يوم 17 ديسمبر 2010 و اطاحت النظام الدكتاتوري يوم 14 جانفي 2011.**

Transliteration Buckwalter : AndlEt Alvwrp Altwnsyp ywm 17 dysmbr 2010 w ATAHt AlnZAm AldktAtwry ywm 14 jAnfy 2011.

*(La révolution tunisienne a commencée le 17 décembre 2010 et a écroulé le régime dictateur le 14 janvier 2014.)*

Dans cet exemple, la conjonction (**و**) est suivie par un verbe et elle représente donc un séparateur réel de phrases, ce qui donne :

- Phrase1 : **اندلعت الثورة التونسية يوم 17 ديسمبر 2010** *(La révolution tunisienne a commencée le 17 décembre 2010)*
- Phrase2 : **اطاحت النظام الدكتاتوري يوم 14 جانفي 2011** *(a écroulé le régime dictateur le 14 janvier 2014.)*

Par contre, pour la phrase suivante :

**دشّن رئيس الجمهورية قسم العيادات الخارجية و وحدة تصفية الدم.**

Transliteration Buckwalter : d$~n r}ys Aljmhwryp qsm AlEyAdAt AlxArjyp w wHdp tSfyp Aldm

*(Le président de la république a inauguré le service des consultations externes et l'unité de dialyse.)*

La conjonction (**و**) est suivie par un nom et elle joue le rôle de conjonction de coordination seulement, non celui de séparateur de phrases.

## 4.4 Ordre des constituants dans une phrase

L'ordre des mots en arabe est relativement libre. Blachère (Blachère & Gaudefroy-Demombynes, 1975) affirme que « *l'ordre des mots, en Arabe classique, n'est ni « libre », ni absolument « fixe ». L'Arabe possède un certain nombre de combinaisons rigides à l'intérieur desquelles peuvent s'introduire des variantes*».

D'une manière générale, on met au début de la phrase le mot sur lequel on veut attirer l'attention et l'on termine sur le terme le plus long ou le plus riche en sens ou en sonorité (Blachère & Gaudefroy-Demombynes, 1975). En Arabe, on a toujours le libre choix du terme qu'on veut mettre en valeur, en tête de la phrase. Cet ordre, relativement libre des mots, provoque des ambiguïtés syntaxiques artificielles dans la mesure où il faut prévoir dans la grammaire toutes les règles de combinaisons possibles d'inversion de l'ordre des mots dans la phrase (Aloulou, 2005). Dans certains cas, ces règles peuvent autoriser des combinaisons de mots qui n'existent pas dans la langue arabe.

A titre d'exemple, nous pouvons donner une phrase simple dont la structure est composée de verbe+sujet+complément :

**إشترى الطالب حاسوباً.**

Transliteration Buckwalter : I$trY AlTAlb HAswbAF.

*(A acheté l'étudiant un ordinateur.)*

Pour cette même phrase, nous pouvons changer l'ordre des mots et avoir les deux structures suivantes :

Sujet+Verbe+complément :

**الطالب إشترى حاسوباً.**

Transliteration Buckwalter : AlTAlb I$trY HAswbAF.

*(L'étudiant a acheté un ordinateur.)*

Complément+Verbe+Sujet :

**حاسوباً إشترى الطالب.**

Transliteration Buckwalter : HAswbAF I$trY AlTAlb.

*(Un ordinateur a acheté l'étudiant)*

## 4.5 Multiplicité des interprétations syntaxiques pour une même phrase

Une phrase peut avoir plus d'une interprétation syntaxique possible. Ceci est dû au fait qu'une forme textuelle mise dans un contexte bien spécifique peut être interprétée syntaxiquement de différentes manières. De plus, les problèmes recensés dans les paragraphes précédents, touchant la voyellation, la segmentation, l'agglutination, etc., peuvent mener à des interprétations syntaxiques distinctes d'une même phrase ou d'un même extrait de texte. En plus, la langue arabe est tellement riche que même l'être humain à du mal à déduire l'analyse syntaxique adéquate et par conséquent, le sens souhaité par l'auteur de la phrase. Considérons par exemple la phrase suivante :

**زرت ساحة المدينة الجميلة.**

Transliteration Buckwalter : zrt sAHp Almdynp Aljmylp**.**

Cette phrase peut être interprétée de deux manières suivantes et qui sont toutes deux syntaxiquement correctes :

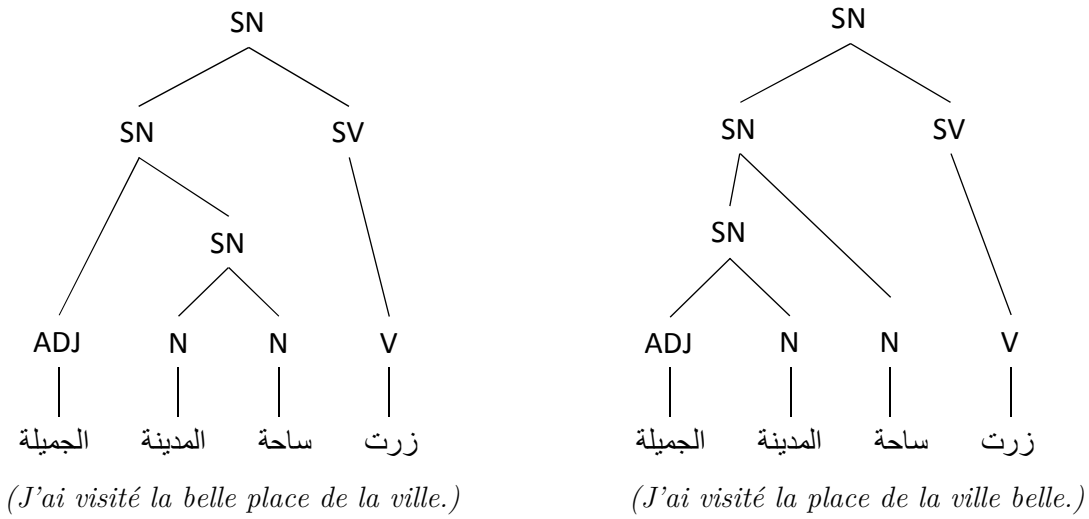



**Figure 1.4 :** Deux arbres syntaxiques pour une même phrase

# 5. Les analyseurs syntaxiques de la langue arabe

Pendant les deux dernières décennies, nous avons remarqué l'intérêt accru donné à la tâche d'analyse syntaxique de la langue arabe. Différents travaux ont été menés dont la plupart sont basés sur une approche symbolique (Aloulou, 2005) (Bataineh & Bataineh, 2009) (Ben Mohamed, Zrigui, Zouaghi, & Zrigui, 2015) (Ouersighni, 2014) (Hammo, Moubaiddin, Obeid, & Tuffaha, 2014) (Marton, Habash, Rambow, & Alkuhlani, 2013) (Al-Taani, Msallam, & Wedian, 2012). Toutefois, nous ne pouvons pas négliger quelques travaux qui sont basés sur l'approche numérique (Tounsi & Van Genabith, 2010) (Ben Fraj F. , 2010).

Nous présentons dans ce qui suit les travaux selon l'ordre chronologique de publication.

## 5.1 L'analyseur de Aloulou

Cet analyseur a été conçu par Aloulou dans le cadre de ses travaux de thèse le au sein du laboratoire MIRACL. Aloulou a choisi de développer son système MASPAR (*Multi-Agent System for Parsing ARabic*) selon une approche multi-agent. En effet, l'auteur à justifié son choix par le fait qu'un traitement séquentiel des différentes phases d'analyse linguistique présente des inconvénients ; à savoir l'absence d'interaction entre les niveaux de représentation, l'absence de distribution du contrôle et des connaissances, la rigidité de l'architecture, la lourdeur du traitement et aussi la

difficulté de l'évolution du système de traitement. Le formalisme grammatical choisi est le HPSG (*Head-Driven Phrase Structure Grammar*). C'est une représentation qui lui a permis de réduire au minimum le nombre de règles syntaxiques et de proposer des représentations lexicales riches et structurées (Aloulou, 2005). Néanmoins, l'application de HPSG pour la langue arabe a nécessité une modification des schémas standards pour pouvoir y intégrer les particularités de la langue Arabe.

Le système MASPAR traite un ensemble de tâches d'une manière parallèle. Il est ainsi composé de six agents ; à savoir : l'agent 'Segmenteur', l'agent 'Lexical', l'agent 'Morphologie', l'agent 'Anaphore', l'agent 'Ellipse'.

Tous ces agents travaillent en collaboration afin de donner l'analyse syntaxique des textes électroniques en entrée. Ils communiquent entre eux par envoi de messages variant selon le type des textes et des phrases en entrée.

L'auteur a évalué l'agent « Syntax » sur un corpus de 3871 phrases dont la longueur maximale ne dépasse pas 10 mots. Ils ont obtenu 61 % d'analyses correctes, 16.5 % d'analyses partiellement correctes et 22.5 % d'analyses incorrectes (Hadrich Belguith, Aloulou, & Ben Hamadou, 2007).

## 5.2 L'analyseur de Bataineh et al.

Les concepteurs de cet analyseur ne donnent pas d'indications sur le contexte de leur travail. Leur analyseur se base sur les réseaux à transitions récursives. Le principe de ces réseaux a été la base des grammaires à transitions récursives. Ces dernières considèrent les règles syntaxiques comme étant des graphes à arcs et à labels. Ce sont des automates à états finis représentant des transcriptions de règles hors contexte. Les auteurs ont alors créé une grammaire à règles hors contexte décrivant les structures les plus courantes des phrases en langue arabe. (Bataineh & Bataineh, 2009)

Les chercheurs se sont aperçus que les structures des phrases varient selon le domaine et les circonstances de l'écrit. Ainsi, la structure d'une phrase en poésie diffère énormément de celles des phrases du discours ordinaire. Pour représenter le maximum de structures, un ensemble de patrons de phrases a été dérivé à partir de textes scolaires. Ces patrons ont été convertis eux aussi en règles hors contexte, en faisant intervenir des linguistes arabes.

Dans ce travail, l'entrée de l'analyseur est une phrase composée de mots étiquetés par leurs caractéristiques correspondantes extraites à partir d'un dictionnaire. L'analyse se fait selon la stratégie descendante en utilisant un algorithme de réseau de transitions récursives.

L'analyseur met en jeu à la fois les règles des patrons et celles hors contexte. Il considère que les deux catégories de règles (simples ou extraites à partir de textes) sont complémentaires. Une phrase est acceptée par cette grammaire si elle est générée par un parcours complet (sans interruption) de ces réseaux de transitions.

L'analyseur a été évalué sur un ensemble de 90 phrases. Parmi ces phrases 85,6% ont été analysées correctement, 2,2% n'ont pas été analysées correctement et 14,4% n'ont pas pu être analysées pour diverses raisons (problèmes de reconnaissance lexicale, phrases incorrectes ou bien phrases non couvertes par la grammaire).

## 5.3 L'analyseur de Tounsi et al.

Cet analyseur a été développé par l'équipe de recherche GramLab du centre NCLT (National Centre for Language Technology) a l'université de Dublin. L'analyseur utilise un modèle appris à partir du PATB afin d'affecter aux différentes phrases leurs structures syntaxiques respectives suivant le formalisme LFG (*Lexical Functional Grammar*). Pour ce faire, les auteurs ont été amenés à convertir le corpus au formalisme LFG. (Tounsi & Van Genabith, 2010)

Cet analyseur utilise l'algorithme A3 (*Arabic Annotation Algorithm*). Ce dernier algorithme a été révisé et adapté à partir d'une méthode antérieure construite pour l'anglais. Les auteurs ont aussi adapté l'analyseur à base d'apprentissage de (Bikel, 2002) conçu originalement pour analyser les textes en langue anglaise. Ils utilisent les deux méthodes adaptées afin d'affecter les bonnes structures LFG aux différentes composantes des phrases. Les dépendances lointaines (anaphores ou ellipses) entre les composantes d'une même phrase sont exploitées à travers la richesse des informations lexicales contenues dans le corpus d'apprentissage.

Un test a été effectué sur 250 phrases choisies à partir du PATB qui compte 23611 phrases annotées. Le f-score représentant les dépendances trouvées a été estimé à 77%.

## 5.4 L'analyseur de Ben Fraj

Ben Fraj a proposé un analyseur syntaxique dans le cadre de ses travaux de thèse au sein du laboratoire RIADI. L'auteur a conçu un analyseur pour les textes en langue arabe qui attribue l'arbre syntaxique le plus approprié pour une phrase à partir de patrons d'arbres syntaxiques. Pour réaliser son système, l'auteur a d'abord construit des ressources pour l'apprentissage (corpus arboré) avant de passer à la procédure d'analyse syntaxique proprement dite.

L'auteur a commencé par le développement d'une grammaire générique partiellement lexicalisée baptisée ArabTAG sous le formalisme d'arbres adjoints (Tree Adjoining Grammar). Celle-ci joue le rôle de référence lors de l'étiquetage syntaxique du corpus arboré. La construction du corpus de textes annotés et arborés a suivi quatre étapes successives : une analyse morphosyntaxique, un étiquetage grammatical, une segmentation en phrases fines et une annotation syntaxique. Un corpus comptant 5000 mots (composant 950 phrases) a été obtenu.

L'auteur a aussi défini des patrons comme étant des modèles de phrases représentés sous forme arborescente à composants stratifiés. Ces patrons sont extraits à partir du corpus arboré. Une Algèbre de composition a été définie pour expliciter les combinaisons licites de patrons de celles qui ne les sont pas.

La procédure d'analyse est non déterministe, progressive et opère selon une stratégie ascendante. A chaque étape, une seule unité de traitement (forme textuelle) est manipulée. Cette unité subit un filtrage qui consiste en la superposition des informations morphosyntaxiques et contextuelles de l'unité en cours de traitement. Un ensemble de patrons est alors choisi. Après chaque filtrage, il y a composition ou union des patrons choisis avec les portions de patrons déjà construits dans les étapes précédentes de l'analyse. Un ensemble de cas de blocage d'analyse est aussi étudié avec la présentation de résolutions possibles.

L'auteur a testé les performances de son analyseur syntaxique sur seulement 50 phrases d'une longueur moyenne de 3,95 mots et a obtenu une précision de 89.85%. (Ben Fraj F. , 2010)

## 5.5 L'analyseur de Al-Taani et al.

Cet analyseur a été réalisé en collaboration entre le département informatique de l'université de Yarmouk en Jordanie et le département informatique de l'université Al-Aqsa en Palestine. Les auteurs présentent un analyseur syntaxique à base de grammaire avec une stratégie descendante. Il est composé des trois modules suivants :

- Classifieur de mots : ce module permet de classifier les mots soit comme verbe soit comme nom. Cette classification utilise les préfixes et les suffixes du mot ainsi que des patrons de règles.
- La grammaire : Al-Taani à construit une grammaire CFG qui permet selon l'auteur de décrire la plus part des cas d'ambigüités de la langue arabe. Les règles de la grammaire ont été regroupées dans deux ensembles en fonction du type de la phrase à analyser (nominale ou verbale).
- L'analyseur : Ce module utilise la grammaire CFG, les mots annotés avec leur catégorie grammaticale et le classifieur de mots pour générer l'analyse syntaxique complète de la phrase.

Les auteurs ont testé leur analyseur sur un petit corpus de 70 phrases de longueur moyenne égale à 3.98 mots (2-6 mots) extraites à partir de documents arabes. Ils ont obtenu une exactitude moyenne de 94,3% (Al-Taani, Msallam, & Wedian, 2012). La valeur, anormalement élevée, de l'exactitude obtenue par cet analyseur est dûe aux types des phrases du corpus de test utilisées. En effet ses phrases sont de petite taille et de forme classique n'exprimant ni la richesse de la langue arabe en tenant ni ses formes exceptionnelles.

## 5.6 L'analyseur de Alqrainy et al.

L'analyseur de Alqrainy et al. a été développé au département informatique et ingénierie de l'université appliquée de Al-Balqa en Jordanie. Celui-ci est basé sur une grammaire hors contexte récursive simple et minimale qui permet de vérifier si une phrase donnée appartient au langage. Pour ce faire, il suit une stratégie descendante selon les étapes suivantes.

A l'axiome de départ S, l'analyseur sélectionne la règle de production avec le symbole S sur sa partie gauche et pour chaque symbole de son coté droit, il construit le sous

arbre approprié. Quand un symbole terminal est ajouté à l'analyse et qui ne correspond pas à la phrase en entrée, alors il revient en arrière pour trouver la règle qui permet de résoudre cet échec. L'analyseur procède de cette manière pour tout symbole non terminal de la partie droite de la règle de départ.

Les auteurs ont évalué leur système sur un corpus de 150 phrases de quatre mots (103 verbale, 47 nominales) préalablement annotées morphologiquement. Leur système a atteint une exactitude de 95% avec 141 phrases correctement analysées (Alqrainy, Muaidi, & Alkoffash, 2012).

## 5.7 L'analyseur de Marton et al.

Cet analyseur a été présenté pendant la campagne d'évaluation SPRML (the Fourth Workshop on Statistical Parsing of Morphologically Rich Languages) par le groupe CADIM (Columbia Arabic & Dialect Modeling group) (Marton, Habash, Rambow, & Alkuhlani, 2013). Cet analyseur est réalisé dans le cadre du projet sur la modélisation des dialectes arabes pour le traitement de la langue et la parole à l'université de Columbia. Ce système est une adaptation pour l'arabe de l'analyseur « Easy first Parser » (Goldberg & Elhadad, 2010) développé pour l'anglais. C'est un analyseur en dépendance basé sur l'apprentissage automatique. Les critères utilisés pour apprendre le model d'analyse sont:

- Des critères morphologiques basés sur les catégories grammaticales des mots.
- Des critères flexionnels qui utilisent la présence ou non du déterminant, la personne, le genre, le nombre…
- Des critères lexicaux qui définissent le concept de rationalité (rationnel/irrationnel/ambigus/…).

Les auteurs ont évalué leur système en suivant les recommandations du workshop, c'est à dire sur une partie du corpus d'apprentissage (10 % de l'ATB[1]). Ils ont calculé les deux valeurs suivantes (LAS/UAS) spécifiques pour les analyseurs en dépendance et définies par le workshop SPRML :

- Label Attachment Score (LAS) : c'est le pourcentage des mots dont le système a prédit la tête et la relation de dépendance correcte, avec une valeur de 80,5%.

[1] Arabic Tree-Bank

– Unlabel Attachment Score (UAS) : C'est le pourcentage de têtes correctement identifiés, avec une valeur de 83,5%.

## 5.8 L'analyseur de Ouersighni

Ouersighni propose une architecture modulaire avec stratégie d'analyse ascendante pour le développement d'un analyseur syntaxique. D'abord, la phrase à analyser passe à travers un module d'analyse morphologique. Celui-ci segmente la phrase en mots et associe un ensemble d'étiquettes morphologiques à chaque mot reconnu comme unité lexicale minimale. La sortie de ce module est placée comme entrée pour le module d'analyse syntaxique. Ce module utilise une grammaire formelle robuste basée sur le formalisme AFGL (Affix Grammars over Finite Lattice) et composée de 850 règles. L'auteur a créé des règles supplémentaires (70 règles) qui permettent d'améliorer la robustesse de l'analyse.

D'après l'auteur, Cette grammaire couvre les phénomènes syntaxiques les plus fréquents incluant des structures syntaxiques de phrases simples et aussi des structures de certains types de phrases complexes telles que les formes négatives, les formes elliptiques, plusieurs formes interrogatives, quelques types de coordination et de déterminants complexes.

L'auteur a testé son analyseur sur 200 phrases extraites d'un corpus nommé Arcolex. Ce corpus comprend des phrases de longueur moyenne 10.52 mots (6-20 mots). Cette évaluation vise à tester la robustesse de l'analyseur. 141 phrases ont été totalement analysées, 49 phrases ont été analysées en utilisant les règles de robustesse et 10 phrases n'ont pas été analysées. L'auteur ne donne pas d'information concernant l'exactitude ou la précision de son système (Ouersighni, 2014).

## 5.9 L'analyseur de Hammo et al.

Cet analyseur a été conçu par le groupe de recherche ANLP du département informatique de l'université de Jordanie.

Cet analyseur propose une architecture en modules pour analyser syntaxiquement des phrases arabes. L'analyseur commence par annoter morphologiquement la phrase en entrée. Une fois la phrase annotée, elle est placée en entrée pour l'analyseur syntaxique qui utilise une grammaire hors contexte basée sur la théorie « Government and

Binding Theory (GB) » proposée par Chomsky. Cette grammaire est constituée de deux ensembles de règles : un premier ne tient pas compte du cas grammatical, alors que le deuxième ensemble inclue les cas grammaticaux des mots. Le cas est au sens large un trait grammatical principalement associé au nom, au pronom, à l'adjectif et au déterminant, et exprimant la fonction syntaxique de ceux-ci dans la proposition, où leur rôle sémantique est en rapport avec l'action exprimée par le verbe.

Le processus d'analyse suit une stratégie descendante selon l'enchaînement suivant. Il commence par générer les structures syntaxiques candidates qui vérifient les contraintes de la théorie GB. Ensuite il procède à un filtrage des candidats qui violent l'une des contraintes. Les structures qui passent le processus de filtrage avec succès sont considérées comme des arbres syntaxiques possibles pour la phrase en entrée.

Les auteurs ont testé leur système sur corpus de 500 phrases voyellées et non voyellées. Les auteurs n'ont présenté aucune valeur permettant de mesurer la performance de leur analyseur et se sont contentés de donner quelques exemples de phrases analysées avec leur système. (Hammo, Moubaiddin, Obeid, & Tuffaha, 2014)

## 5.10 L'analyseur de Ben Mohamed et al.

Ben Mohamed a réalisé un analyseur syntaxique pour l'arabe dans le cadre de ses travaux de thèse au sein du laboratoire LaTICE. L'auteur (Ben Mohamed, Zrigui, Zouaghi, & Zrigui, 2015) part de l'hypothèse que l'utilisation d'une grammaire basée sur l'utilisation des schèmes au lieu des mots aurait un plus grand taux de couverture des spécificités de la langue arabe (Ben Mohamed, Mallat, Nahdi, & Zrigui, 2015). Pour cela ils ont construit une grammaire hors contexte probabiliste basée sur les schèmes à partir de 100000 mots arabes.

Le processus d'analyse basée sur les schèmes se déroule en deux phases. La première phase est une phase de conversion qui vise à convertir le texte en entrée en schèmes. Elle commence par reconnaître les particules ensuite elle convertit les mots en schèmes en utilisant une base constituée de 3027 schèmes. Cette phase produira pour chaque schème l'expression régulière correspondante qui identifie le schème correspondant au mot traité. Enfin, une comparaison avec la liste des racines des mots arabes afin d'éviter une certaine ambiguïté de conversion.

Une fois la conversion terminée, on passe à la phase d'analyse qui génère l'analyse syntaxique la plus probable, avec une stratégie ascendante, en utilisant la grammaire PCFG générée et l'algorithme d'analyse CYK.

Les auteurs ont testés leur système sur un ensemble de 836 phrases extraites de sources en ligne (blogs, forums,…) et avec une longueur moyennes de 5,08 mots. Ils ont obtenu une exactitude de 63.35% (Ben Mohamed, Mallat, Nahdi, & Zrigui, 2015).

## 5.11 L'analyseur de Barhoumi et al.

Cet analyseur a été développé au sein du laboratoire MIRACL par (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015). Ce système est basé sur l'apprentissage automatique supervisé en utilisant le corpus PATB.

L'idée de base de cet analyseur numérique est que l'analyse syntaxique profonde peut être réalisée en effectuant une succession d'analyses de surface ou chunking. Les auteurs ont étudié les arbres syntaxiques du corpus d'apprentissage PATB pour déterminer le nombre d'analyse successive nécessaire afin d'analyser syntaxiquement une phrase. Ils ont remarqué que l'arbre le plus profond du corpus à 8 niveaux (1 niveau morphologique et 7 niveaux syntaxiques). Ils ont donc opté pour une analyse syntaxique à 8 niveaux successifs. Cette analyse multi-niveaux nécessite un modèle statistique pour chaque niveau.

Le processus d'analyse syntaxique se déroule comme suit : l'analyseur effectue, en premier lieu, l'étiquetage morphosyntaxique pour chaque mot de la phrase. Par la suite, l'analyseur effectue une cascade d'analyse de surface ; à chaque niveau, il détermine, s'il existe des constituants intégrés à ceux du niveau inférieur dans l'arabe syntaxique. Et en dernière étape, l'analyseur précise le type de la phrase analysée.

Les modèles statistiques ont été appris sur une partie du PATB constitué de 10103 phrases (environ 80% du PATB) et l'évaluation a été faite sur une deuxième partie qui englobe 2525 phrases (environ 20% du PATB). Les auteurs ont calculé le rappel, la précision et la f-mesure pour chaque niveau d'analyse. Le tableau 1.2 présente les valeurs obtenues.

| | **Précision** | **Rappel** | **F-mesure** |
|---|---|---|---|
| **Niveau 7** | 92.08% | 92.08% | 92.08% |
| **Niveau 6** | 90.96% | 89.53% | 90.24% |
| **Niveau 5** | 69.86% | 64.74% | 67.20% |
| **Niveau 4** | 66.68% | 61.45% | 63.96% |
| **Niveau 3** | 68.42% | 63.86% | 66.06% |
| **Niveau 2** | 69.77% | 66.64% | 68.17% |
| **Niveau 1** | 67.40% | 63.98% | 65.65% |
| **Niveau 0** | 63.76% | 49.22% | 55.56% |
| **Moyenne** | 73,94% | 68,94% | 71.12% |

**Tableau 1.2 :** Résultats de l'évaluation de l'analyseur de Barhoumi et al.

## 5.12 L'analyseur de Jaf et al.

L'analyseur de Jaf a été développé par l'équipe « Innovative Computing Research Group » de l'école d'ingénierie et d'informatique de l'université de Manchester. Cet analyseur en dépendance repose sur une approche hybride qui propose de profiter des avantages de l'approche dirigée par les données (numérique) et qui soit guidée par des règles symboliques pour améliorer les analyses produites. Afin de valider leur approche, les auteurs ont développés, dans un premier temps, un analyseur à base d'apprentissage automatique et un analyseur à base de règles. Ensuite, Ils ont intégré des règles symboliques à l'analyseur numérique pour guider le processus d'analyse. Les auteurs ne mentionnent pas la manière avec laquelle cette hybridation a été faite.

Les auteurs ont utilisé le corpus PATB pour le développement de leur parseur. Ils ont du le convertir vers le format en dépendance. Cet analyseur a été entrainé sur un corpus de 100 000 mots et son évaluation a été réalisée sur un corpus de 10 000 mots. L'exactitude obtenue par cet analyseur hybride est supérieure que celle de l'analyseur numérique seul, mais elle est inferieur à l'exactitude de l'analyseur à base de règle. Concernant le temps d'analyse, l'analyseur hybride est plus rapide que l'analyseur symbolique mais moins rapide que l'analyseur numérique. Les résultats sont exposés dans le tableau 1.3.

| Approche utilisée | Exactitude | Durée (sec) |
|---|---|---|
| Numérique | 63% | 235 |
| Symbolique | 90% | 878 |
| Hybride | 88% | 459 |

**Tableau 1.3 :** Evaluation de l'analyseur Hybride de Jaf et al.

# 6. Synthèse sur les analyseurs syntaxiques existants pour l'arabe

Quelques soient les approches, méthodes, techniques ou même les ressources, les résultats des travaux qui traitent de l'analyse syntaxique sont généralement jugés satisfaisants par leurs concepteurs. Mais, à quel point ces résultats ont-ils répondu aux besoins des concepteurs ? A quel point les tests réalisés peuvent-ils renseigner sur la robustesse, la précision et la couverture des différents analyseurs traitant différentes langues ? Et comment peut-on évaluer et comparer ces différentes approches d'analyse syntaxique ?

Toutes ses questions soulignent que la comparaison des analyseurs syntaxiques reste une tâche difficile. Cela est dû au type de l'analyse produite (en dépendance ou en constituants), à la différence des données d'évaluation et à la non disponibilité des analyseurs en ligne. Et même s'ils le sont, adapter tous les analyseurs sur une seule structure de données de test est une tâche fastidieuse et ardue.

Pour avoir une vue d'ensemble sur les analyseurs que nous venons de citer pour l'arabe, nous les avons comparés selon les critères suivants :

- L'approche,
- La stratégie,
- L'architecture,
- Les données de l'évaluation,
- Et les résultats obtenus.

Le tableau 1.4 expose les résultats de notre étude comparative entre les analyseurs syntaxiques cités pour l'arabe.

| Auteurs | Approche | | Stratégie | Architecture | Evaluation | Résultats |
|---|---|---|---|---|---|---|
| | Linguistique | Numérique | | | | |
| (Aloulou, 2005) | ✘ (Grammaire HPSG) | | Ascendante | Multi-agent | 3871 phrases<br>Agent « syntax »<br>(<11 mots) | Analyses correctes 61 %<br>Analyses partiellement correctes16.5 %<br>Analyses incorrectes 22.5 % |
| (Bataineh & Bataineh, 2009) | ✘ (Grammaire CFG) | | Descendante | Modulaire | 90 phrases<br>(6 mots) | Analyse correctes 85.4 %<br>Analyses incorrectes 2.2 %<br>Analyses rejetées 12.4 % |
| (Ben Fraj F. , 2010) | ✘ (Grammaire ARAB Tag) | | Ascendante | Modulaire | 50 phrases<br>(3,95 mots) | Précision 84.78%<br>Rappel 71.54%<br>F-mesure 77.52% |
| (Tounsi & Van Genabith, 2010) | | ✘ (A.P[2]) | - | - | 250 phrases | F-mesure 77% |
| (Al-Taani, Msallam, & Wedian, 2012) | ✘ (Grammaire CFG) | | Descendante | Modulaire | 70 phrases<br>(2-6 mots) | Précision 94% |
| (Alqrainy, Muaidi, & Alkoffash, 2012) | ✘ (Grammaire CFG) | | Descendante | Modulaire | 105 phrases | Exactitude 95% |
| (Marton, Habash, Rambow, & Alkuhlani, 2013) | | ✘ (A.P) | - | - | 10% de l'ATB<br>(~1800 phrases) | LAS 80.5%<br>UAS 83.5% |
| (Hammo, Moubaiddin, Obeid, & Tuffaha, 2014) | ✘ (Grammaire CFG) | | Descendante | Modulaire | 500 phrases | - |
| (Ouersighni, 2014) | ✘ (Grammaire AFGL) | | Ascendante | Modulaire | 200 phrases<br>(6-20 mots) | 141 phrases totalement analysées,<br>49 avec les règles de robustesse,<br>et 10 non analysées. |

[2] Apprentissage Automatique

| | | | | | | |
|---|---|---|---|---|---|---|
| (Ben Mohamed, Mallat, Nahdi, & Zrigui, 2015) | ✘ (Grammaire PCFG) | | Ascendante | Modulaire | 836 phrases (une moyenne de 10,08 mots) | Exactitude de 63.35%. |
| (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015) | | ✘ (A.P) | Ascendante | Modulaire | 2525 phrases | Précision : 73,94%<br>Rappel : 68,94%<br>F-mesure : 71.12% |
| (Jaf & Ramsay, 2016) | ✘ | ✘ | Descendante | Modulaire | 10 000 mots | Exactitude de 88%. |

**Tableau 1.4 :** Tableau comparatif des analyseurs syntaxiques arabes

En fait, nous avons remarqué un gain d'intérêt pour l'analyse syntaxique de l'arabe durant ces deux dernières décennies. Certains ont utilisé l'approche numérique et d'autres l'approche linguistique.

Les travaux de Tounsi et al. (Tounsi & Van Genabith, 2010) et Marton et al. (Marton, Habash, Rambow, & Alkuhlani, 2013) utilisent une approche numérique basée sur l'apprentissage automatique. Tounsi et al. ont utilisé l'analyseur syntaxique commercialisé de Bikel (Bikel, 2002) conçu pour la langue anglaise. Ils l'ont adapté pour la langue arabe après avoir effectué des transformations pour qu'il respecte au mieux la grammaire de la langue traitée. Quant à Matron et al., ils ont adapté l'analyseur « Easy first Parser » de (Goldberg & Elhadad, 2010) initialement créé pour la langue anglaise afin de pouvoir l'utiliser aussi pour la langue arabe.

Nous pensons que des analyseurs dont la conception initiale est orientée vers l'analyse de la langue anglaise ne peuvent rendre compte de façon précise des différentes structures syntaxique de l'arabe et cela malgré les adaptations que les auteurs ont bien dit avoir réalisées.

Les autres travaux ont utilisé une approche linguistique basée sur les grammaires symboliques. Les différents auteurs ont créé leurs grammaires suivant différents formalismes grammaticaux afin de guider au mieux le processus d'analyse.

Cependant, les spécificités de la langue arabe, déjà citées dans la section 4, rendent difficile la construction d'une grammaire exhaustive et à large couverture qui comprend à la fois les régularités, les exceptions ainsi que les structures mal formées mais fréquemment utilisées dans notre langue. En plus, la majorité des corpus de tests utilisés sont de taille insuffisante pour donner une idée objective sur les performances des analyseurs. Par exemple, (Ben Fraj F. , 2010) a utilisé 50 phrases de tests. Al-Taani et ses alliés (Al-Taani, Msallam, & Wedian, 2012) ont utilisé 70 phrases. (Alqrainy, Muaidi, & Alkoffash, 2012) ont évalué leur analyseur sur 105 phrases.

Aussi, nous avons remarqué que la taille des phrases de test dans les cas où elle est indiquée par les auteurs, ne reflète pas la taille des phrases arabes utilisées dans des cas de données réelles. Par exemple, la taille maximale des phrases de test de (Aloulou, 2005) est de 10 mots. (Bataineh & Bataineh, 2009) ont utilisé des phrases de 6 mots maximum. La moyenne des phrases utilisés par (Ben Mohamed, Mallat, Nahdi, & Zrigui, 2015) est de 10,08 mots.

En effet, Vu la richesse de la langue arabe et ses différentes caractéristiques, la longueur des phrases dépassent largement les 10 mots et peuvent atteindre facilement 20 mots voir même 40 mots et plus. Et si on prend en considération le phénomène d'imbrication des phrases en langues arabe, la longueur peut même dépasser les 60 mots.

D'autant plus, les résultats obtenus par les analyseurs cités dans cet état de l'art sont intéressants mais restent en dessous des analyseurs syntaxiques pour l'anglais ou le français ce qui nous encourage à continuer le travail sur ce domaine.

Pour résumer cette discussion, nous pouvons dire que les travaux, ainsi décrits, présentent un ensemble de limites se résumant dans les points suivants :

- L'inadaptation des analyseurs syntaxiques numériques, conçus et développés spécialement pour la langue anglaise, à l'analyse syntaxique de la langue arabe.
- Les grammaires utilisées et développées manuellement sont insuffisantes pour couvrir la plupart des phénomènes syntaxiques de la langue arabe et nécessitent la validation des experts linguistes.
- Les corpus de test sont de taille insuffisante et ne reflètent pas la réalité des phrases en arabes surtout en termes de longueur.
- Les résultats obtenus ne sont pas au niveau des analyseurs d'autres langues comme l'anglais ou le français.

# 7. Conclusion

Nous avons commencé ce chapitre par présenter les notions de base de la syntaxe et de l'analyse syntaxique. Ensuite, nous avons étudié les caractéristiques de la langue arabes telles que la non voyellation, l'agglutination, l'irrégularité de l'ordre des mots, etc., qui rendent son traitement difficile et plus ambigu que celui des autres langues naturelles. Nous avons clôturé ce chapitre par la présentation et la discussion des analyseurs syntaxiques existants pour la langue arabe.

Le chapitre suivant sera consacré aux différentes approches proposées dans la littérature pour la mise en œuvre d'analyseurs syntaxiques. Nous attacherons une attention particulière à l'aspect hybride de ces approches qui constituent un élément important de cette thèse.

# Chapitre 2

Etat de l'art sur les approches d'analyse syntaxique

# 1. Introduction

L'analyse syntaxique représente une étape fondamentale dans le processus d'analyse automatique des langues, puisque c'est à elle que revient la tâche cruciale de déterminer les structures syntaxiques des phrases d'un texte. Plusieurs travaux ont été effectués pour résoudre les problèmes de l'analyse syntaxique. Ces travaux peuvent être classés dans trois approches principales : approche symbolique (linguistique), approche numérique (statistique), et approche mixte ou hybride (Aloulou, 2005). L'approche symbolique est basée sur la donnée explicite de connaissances lexicales et de règles linguistiques pour lever les ambigüités des mots de la phrase et valider la structure de la phrase. Les approches statistiques se basent essentiellement sur des modèles statistiques ou probabilistes ; l'aspect lexical et la structure grammatical de la phrase sont ignorés. L'approche hybride est un mélange des deux approches précédentes. Elle intègre une analyse linguistique ainsi qu'une analyse statistique.

Ce chapitre présente le résultat d'une recherche bibliographique concernant notre domaine d'étude, à savoir l'analyse syntaxique. Dans un premier temps, nous présentons l'approche symbolique. Nous présentons, dans un deuxième temps l'approche numérique. Ensuite nous présentons l'approche hybride. Nous achèverons ce chapitre par une discussion sur les trois approches.

# 2. L'approche symbolique

L'approche symbolique a été la première utilisée pour effectuer de l'analyse syntaxique. Les travaux basés sur cette approche reposent essentiellement sur l'utilisation d'une grammaire et d'une stratégie d'analyse pour pourvoir analyser des phrases brutes. La grammaire permet de capturer les différentes combinaisons possibles entre les syntagmes de la langue et de les représenter à travers des règles, tandis que l'algorithme d'analyse  permet de vérifier si la phrase en entrée fait partie de la langue décrite par la grammaire. La figure suivante présente l'architecture globale d'un analyseur symbolique.

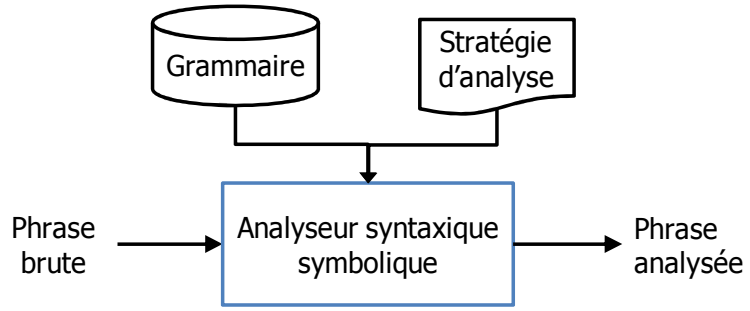



**Figure 2.1 :** Architecture globale d'un analyseur syntaxique symbolique

La principale caractéristique des approches symboliques est qu'elles offrent la possibilité de retracer les étapes qui auront permis d'associer entre eux les symboles de la grammaire. Cette trace consiste en la liste des symboles et des règles de la grammaire, qui rend lisible et reproductible le passage de la phrase analysée à l'ensemble des descriptions résultant de l'analyse (la succession des règles appliquées pour effectuer l'analyse). En TALN, cette trace est cruciale pour la compréhension des phénomènes analysés (Abeillé & Blache, 2000).

Plusieurs grammaires ont vu le jour et plusieurs stratégies d'analyse ont été proposées. Dans ce qui suit nous présentons un aperçu global des grammaires syntaxiques, des stratégies d'analyse syntaxique.

## 2.1 Les grammaires syntaxiques

### 2.1.1 Les grammaires génératives

La grammaire générative est une théorie syntaxique s'inscrivant dans le courant de la linguistique générative. Elle s'est développée depuis 1957 sous l'impulsion de Noam Chomsky. Cette théorie tente de caractériser la connaissance de la langue qui permet l'acte effectif du locuteur-auditeur. Chomsky soutient donc la thèse selon laquelle il existe une grammaire universelle innée, qui serait le domaine de compétences spécifiques à l'espèce humaine (Boltanski, 2002).

Selon Noam Chomsky, *« la grammaire d'une langue propose d'être une description de la compétence intrinsèque du locuteur-auditeur idéal. Si la grammaire est, de plus, parfaitement explicite (en d'autres termes, si elle ne fait pas simplement confiance à la compréhension du lecteur intelligent, mais fournit une analyse explicite de l'activité*

*qu'il déploie), nous pouvons, non sans redondance, l'appeler grammaire générative. »* (Chomsky, 1965).

Pour soutenir sa thèse, Chomsky a défini un ensemble de règles permettant de générer des syntagmes ou des phrases. Ces règles sont dites « règles de réécriture » et s'illustrent sous la forme :

A→BC ou A se réécrit B,C

L'application de ces règles permet de faire des prédictions quant aux types des phrases rencontrées dans une même langue. Les dites règles constituent la base des dérivations de la grammaire générative. Et à partir de ces règles, il devient possible de générer/valider toutes les phrases possibles de langue.

Après avoir dégagé cette notion de grammaire générative, Chomsky a défini une hiérarchie de grammaires. Elle définit quatre types de grammaires selon des contraintes que l'on impose (ou pas) sur la forme des règles. La figure suivante schématise cette hiérarchie (Chomsky, 1997).

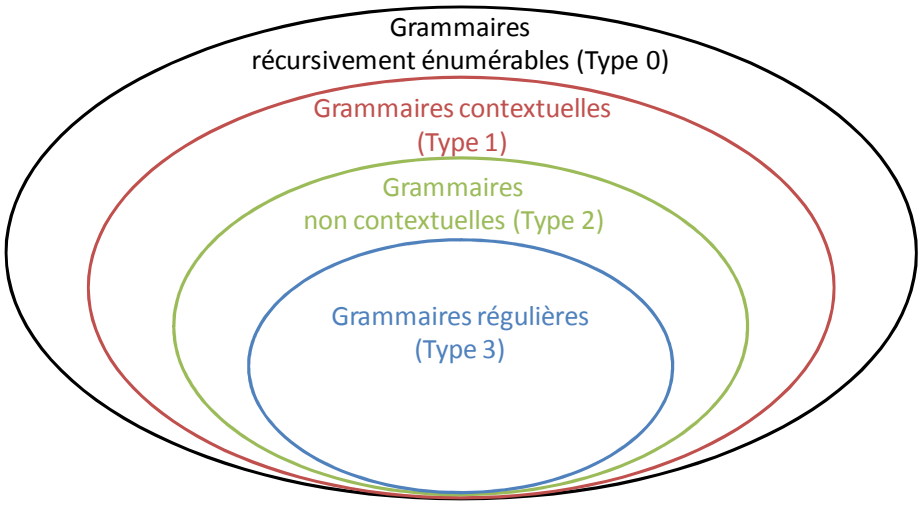



**Figure 2.2 :** Hiérarchie des grammaires de Chomsky (Chomsky, 1997)

Chomsky a défini quatre classes de grammaires hiérarchiquement imbriquées, nommées de type 0 à type 3. Les langages de type 0 sont les plus généraux: ce sont les langages récursivement énumérables qui n'imposent aucune restriction sur les règles de la grammaire. Ils contiennent les langages de type 1, les langages contextuels, en anglais « context-sensitive ». Les langages de type 2 sont appelés langage algébriques ou «

hors contexte », en anglais « context-free ». Ils contiennent eux-mêmes les langages de type 3, les langages « réguliers » ou langages rationnels.

### 2.1.2 Les grammaires d'unifications

Les formalismes d'unification sont apparus à partir du milieu des années soixante-dix. Ces formalismes ont tous en commun l'usage d'un mécanisme particulier qui est l'unification des structures de traits, ce qui représente un changement total par rapport aux grammaires génératives de Chomsky (Aloulou, 2005).

Les structures ou matrices de traits sont un ensemble de couples attributs-valeurs notés entre crochets. Les attributs peuvent être à valeur atomique (c'est-à-dire valués avec un terme simple) ou complexe (c'est-à-dire valués avec une structure de traits emboîtés).

L'amélioration de la couverture de ces formalismes passe en effet par une multiplication des règles et des traits utilisés, et par conséquent, cette amélioration a donné naissance à des modèles de grammaires d'unification développées comme des théories linguistiques, dont nous donnons un bref aperçu:

- Les Grammaires Lexicales Fonctionnelles, LFG, sont issues des travaux de Joan Bresnan et Ronald Kaplan (Kaplan & Bresnan, 1982). Dans ces modèles, la phrase est décrite non seulement par des représentations arborescentes mais aussi par des structures de traits qui codent directement les différentes fonctions grammaticales.

- Les Grammaires Syntagmatiques Généralisées, GPSG (Generalized Phrase Structure Grammar), proposées par (Gazdar, 1985). Il s'agit d'un formalisme basé sur des contraintes très générales, mais dont l'objectif est de traiter une grande variété de phénomènes qui posent des problèmes d'analyse. GPSG se positionne comme un héritier de la grammaire générative.

- Les Grammaires Syntagmatiques guidées par les Têtes, HPSG (Head driven Phrase Structure Grammar), enrichit et systématise l'utilisation de structures de traits de GPSG. Ses concepteurs C. Pollard et I. Sag, ont également repris des notions présentes dans d'autres formalismes comme LFG et la grammaire catégorielle (Aloulou, 2005).

- Les Grammaires d'Arbres Adjoints, TAG (Tree Adjoinning Grammar), 1988, mettent en œuvre des unités linguistiques qui sont des arbres pouvant se combiner par adjonction ou substitution. Les grammaires d'arbres Adjoints correspondent à un modèle mathématique défini en 1975 par A. Joshi (Joshi & Bangalore, 1994) dans le cadre d'une extension des grammaires en chaînes (Harris, 1962).

## 2.2 Les stratégies d'analyse

Dans ce qui suit, nous abordons les stratégies existantes dans la littérature, concernant la manière ou la démarche dont est appliquée l'analyse. Différentes stratégies d'analyse ont été proposées et utilisées pour guider le processus d'analyse syntaxique. Ces stratégies peuvent être classées en se basant sur la manière selon laquelle est construit l'arbre syntaxique (ascendante, descendante ou mixte).

Nous présentons dans ce qui suit une description de ses stratégies d'analyse.

### 2.2.1 Analyse ascendante

L'analyse ascendante est également appelée « analyse dirigée par les données » car elle part des données que constituent les mots de la phrase à analyser. L'analyse ascendante essaie de combiner les éléments dans une phrase en entrée de différentes manières jusqu'à aboutir à un arbre qui couvre tous les mots de la phrase. Elle commence par les mots d'une phrase et procède par regroupement des mots en des composants de plus en plus larges jusqu'à ce que le tout soit regroupé sous la même racine (O'Donnell, 1994).

L'analyse ascendante est plus flexible puisqu'elle permet de construire des solutions partielles (il est possible de reconnaître un syntagme nominal dans une proposition même si l'ensemble de la proposition n'est pas reconnue). Toutefois, il est à noter que cette analyse est plus difficile à mettre en œuvre (elle nécessite notamment le recours à la méta-programmation); ce qui explique qu'elle est moins utilisée (Aloulou, 2005).

### 2.2.2 Analyse descendante

Une analyse descendante (dite dirigée par les hypothèses) commence par la racine de l'arbre, puis construit les différents branchements jusqu'à atteindre les feuilles de

l'arbre (généralement des mots). Théoriquement, une telle stratégie se base sur l'idée de produire, en utilisant une grammaire générative, toutes les structures syntaxiques possibles (O'Donnell, 1994).

L'analyse descendante permet d'éviter de construire des constituants qui, en fin d'analyse, se révéleraient impossibles à rattacher. Par contre, elle ne permet pas des analyses partielles; toute la phrase doit être reconnue (Aloulou, 2005).

### 2.2.3 Analyse mixte

Bien que les deux types d'analyse soient formellement équivalents, on constate que les analyses ascendante et descendante se heurtent à des difficultés de nature différente. L'analyse mixte propose d'appliquer une analyse ascendante et une analyse descendante dans un même processus d'analyse syntaxique. La stratégie mixte doit pouvoir améliorer les résultats de l'analyse en tirant profit des avantages des deux approches précédente et en essayant de pallier leurs inconvénients (Wehrli, 1997).

## 2.3 Analyseurs basés sur l'approche symbolique

Dans cette section nous allons présenter deux travaux qui ont utilisé l'approche symbolique au cours du processus d'analyse syntaxique. Nous avons choisi de présenter des analyseurs multilingues et qui sont largement cités dans les travaux de recherche et mettant en œuvre deux types de grammaire symbolique majeur, à savoir grammaire en constituant et grammaire en dépendance.

### 2.3.1 L'analyseur FIPS

FIPS est un analyseur syntaxique multilingue (français, anglais, espagnol, italien, et allemand) basé sur une grammaire inspirée du modèle de Chomsky et des grammaires lexicales-fonctionnelles(LFG).

Cet analyseur balaye une chaine d'entrée de gauche à droite, sans retour en arrière, ensuite il lit un mot de catégorie X et projette un syntagme de catégorie XP, dont le mot constitue la tête lexicale, puis combine le constituant XP avec l'analyse en cour en fonction des règles de la grammaire. L'analyseur calcule toutes les solutions possibles et les ordonne selon des critères psycholinguistiques. A la fin du processus, si l'analyseur a une ou plusieurs structures couvrant toute la séquence, il les affiche ; sinon il retourne

un message d'échec et affiche une structure formée par la concaténation d'analyses partielles.

FIPS utilise sa propre base lexicale qui est constituée des concepts suivants :

- les mots, représentant toutes les formes morphologiques des mots d'une langue, regroupés en paradigmes flexionnels,
- les lexèmes, décrivant des formes lexicales plus abstraites qui correspondent aux lectures syntaxiques et sémantiques d'un mot,
- les collocations, qui regroupent les expressions multi-mots,
- et les variantes orthographiques, qui dressent l'ensemble des graphies alternatives autorisées pour un mot.

Le temps de traitement dépend de la complexité des corpus, il est généralement compris entre 100 et 250 symboles à la seconde. FIPS n'a pas été évalué de façon précise dans une campagne d'évaluation. Les seules données statistiques à grande échelle disponible sont le nombre d'analyses complètes (plus de 80% pour le français et l'anglais) et le nombre de mots inconnus (0,2% pour l'anglais et 0,1% pour le français) sans compter les noms propres (Wehrli & Nerima, 2015).

### 2.3.2 L'analyseur LIMA

Cet analyseur syntaxique est multilingue (français, anglais, espagnol, arabe, japonais, chinois, allemand) et se base sur les grammaires de dépendances.

LIMA (Besançon, et al., 2010) est implémenté comme un pipeline de modules indépendants appliqués successivement sur un texte. Il met en œuvre une grammaire de dépendance, en ce sens que les analyses produites sont exclusivement représentées comme des relations de dépendance binaire entre les mots. L'analyseur comprend, entre autres modules, un segmenteur en tokens reposant sur les signes de ponctuation, un étiqueteur morphosyntaxique, des extracteurs de dépendances à courte et longue portée fondés sur des automates à états finis, définis par des règles contextuelles.

Appliqué à  la langue arabe, le système LIMA est assez lent ; la vitesse de traitement est de 146 mots/sec contre 4332 mots/sec pour l'anglais ou 6580 mots/sec pour le français, ce qui fait une grande différence de vitesse. Cela est du a la complexité de la langue arabe qui nécessite une analyse morphologique performante et une grammaire

très riche. Le tableau suivant indique les performances de l'analyseur syntaxique du système LIMA pour la langue française dans les campagnes d'évaluation Passage(2009) et Easy(2005).

| Campagne | LIMA | Meilleur système |
|---|---|---|
| Easy Groupes | 0.82% | 0.89% |
| Passage Groupes | 0.85% | 0.94% |
| Easy Relations | 0.50% | 0.59% |
| Passage Relations | 0.60% | 0.68% |

**Tableau 2.1 :** Evaluation de LIMA pendant les campagnes Easy et Passage.

# 3. L'approche numérique

Cette approche a fait son apparition dans les années 90 avec l'avènement de modèles statistiques innovants et faciles à mettre en œuvre. Elle se base principalement sur des modèles statistiques qui sont automatiquement dérivées des corpus d'apprentissage. La structure globale de la phrase ainsi que l'aspect grammatical sont ignorés. La figure suivante présente l'architecture globale d'un analyseur numérique.

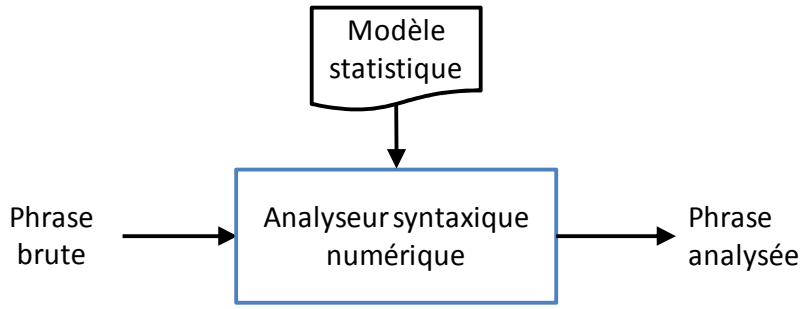



**Figure 2.3 :** Architecture globale d'un analyseur syntaxique numérique

Nous commençons cette section par la présentation des modèles utilisés dans cette approche ensuite nous présentons quelques analyseurs syntaxiques basés sur l'approche numérique.

## 3.1 Le modèle génératif

Le modèle génératif dans l'approche numérique de l'analyse syntaxique se fonde sur la tradition des langages formels et des systèmes de réécriture. L'analyse syntaxique est

envisagée ici comme un processus permettant de passer d'une structure initiale (une chaîne d'entrée) à une structure finale (un arbre ou une forêt d'analyse). On utilise le plus couramment les grammaires algébriques ou les automates qui peuvent s'analyser en temps polynomial. Ces grammaires et automates sont généralement extraits d'un corpus annoté qui servira en même temps à la phase de probabilisation. En effet, une distribution de probabilités conditionnelles est construite sur les règles de la grammaire ou les états des automates à partir du corpus. Ainsi une probabilité est attribuée à chaque règle et a chaque passage d'un état à un autre (Urieli, 2013). Dans ce qui suit, nous présentons deux exemples de modèles génératifs.

### 3.1.1 Le modèle de Markov caché

Le modèle de Markov caché (en anglais, Hidden Markov Model, HMM) est l'un des modèles les plus connu au sein des modèles génératifs. Les HMM sont largement utilisés pour l'étiquetage morphosyntaxique, la désambiguïsation syntaxique, la reconnaissance des entités nommées, etc. (Manning & Schütze, 1999). En fait, Ils utilisent des automates finis avec des transitions d'états stochastiques et des observations (Baum & Petrie, 1966). La figure suivante montre une vue graphique d'un modèle de Markov caché.

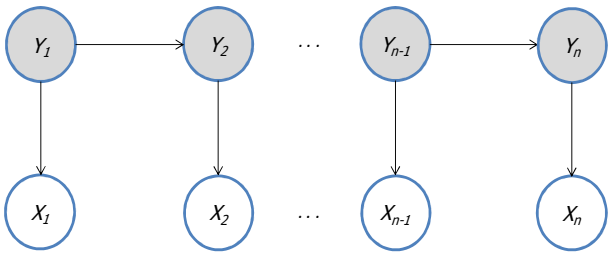



**Figure 2.4 :** Modélisation graphique d'un modèle de caché de Markov

Le modèle de Markov caché traite le texte comme une séquence d'états cachés (la séquence des catégories), chaque état produisant des émissions (les mots observables de la phrase). Dans le modèle de Markov, on considère que le choix d'une étiquette pour un mot doit dépendre des « n » étiquettes précédentes, et du mot lui même. Le passage d'un état à l'autre, c'est-à-dire d'une étiquette de catégorie à la suivante, est appelé une transition (Graja, 2016).

Un modèle de Markov caché est défini par :

- Un ensemble fini d'états $Y$ comportant un état initial, un état final et des états émettant des symboles.
- Un ensemble fini d'observation $X$.
- Une matrice de transition définissant les probabilités de transition d'un état $y$ à un autre état $y'$ : $\{P(y'/y),\ y \in Y,\ y' \in Y\}$, sachant qu'il n'existe pas de transition vers l'état initial et une transition vers l'état final.
- Une matrice d'émission définissant les probabilités d'une observation dans un état : $\{P(x/y),\ y \in Y\}$.
- Une loi de probabilité définissant la probabilité des états initiaux $\{P(y_0)\}$.

La probabilité conjointe d'une séquence d'observation $X$ et d'une séquence $Y$ est donnée par l'équation suivante :

$$\mathrm{P}(X|Y) = P(y_0)\, P(x_0|y_0) \prod_{t=1}^{T-1} P(y_t|y_{t-1})\, P(x_t|y_t)$$

### 3.1.2 Le modèle maximum d'entropie

Le modèle Maximum d'Entropie (ME) (Jaynes, 1957) (Berger, Della Pietra, & Della Pietra, 1996) consiste à utiliser un ensemble de données d'apprentissage afin de construire un modèle probabiliste qui s'adapte à ces données. Par exemple, les traders de la finance construisent des modèles statistiques qui prédisent le cours de la bourse des prochains jours à partir des prix du marché des jours précédents. De la même façon, un modèle ME va tenir compte uniquement des informations disponibles afin de déterminer une loi de probabilité (Sigogne, 2012).

On va donc chercher un modèle qui maximise l'entropie, le maximum d'informations issues du corpus d'apprentissage, tout en tenant compte de contraintes sur ces informations.

L'algorithme de modélisation ME peut être vu comme deux étapes distinctes. La première étape consiste à tirer une série d'observations du corpus, que l'on appelle traits. En deuxième étape, ces traits sont intégrés au modèle statistique. Le but est d'obtenir un modèle uniforme sous contraintes de ces traits afin qu'il puisse refléter la distribution initiale du corpus.

On suppose qu'un processus aléatoire génère une variable y, élément d'un ensemble fini Y, conditionné par un éventuel contexte x, élément d'un ensemble fini X. Dans le cadre de l'étiquetage morphosyntaxique, y est une étiquette morphosyntaxique et x un historique (ou contexte) restant à définir. Le but est de construire un modèle statistique qui reflète le comportement de ce processus aléatoire. Il s'agit donc d'une méthode d'estimation de la probabilité conditionnelle p(y|x) que le processus génère y étant donné un contexte x. Afin de déterminer le comportement de ce processus, la première étape consiste à tirer des observations d'un corpus C constitué de paires d'exemples (x, y) tel que C = $(x_1, y_1)$, $(x_2, y_2)$, …, $(x_N, y_N)$. Le corpus d'apprentissage peut être représenté par la distribution de probabilités p, définie par :

$$P(x,y) = \frac{1}{N} \times nb_{\text{occ}}(x,y)$$

Où $nb_{occ}(x, y)$ est le nombre d'occurrence de la paire (x; y) dans le corpus, normalisée par la taille N du corpus.

Cette probabilité est tout simplement la distribution des paires (x; y) distinctes dans le corpus. Le but est de calculer une loi de probabilité permettant de modéliser le processus générant le corpus d'apprentissage. Comme énoncé précédemment, on souhaite extraire du corpus un ensemble d'événements statistiquement pertinents à inclure dans le modèle (Sigogne, 2012).

## 3.2 Le modèle discriminant

Au cours des dernières années, les méthodes d'analyse discriminantes ont gagné en popularité. En fait, les systèmes utilisant cette approche se fondent sur la « syntaxe comme théorie des modèles » (en anglais model-theoretic syntax) (Pullum & Scholz, 2001) et ont connu un regain d'intérêt grâce aux progrès réalisés dans le domaine de l'apprentissage automatique, plus précisément en classification automatique. Dans cette approche, la grammaire est vue comme un système de contraintes sur les structures syntaxiques correctes. Les mots de la phrase d'entrée sont eux-mêmes vus comme des contraintes sur les positions qu'ils occupent et l'analyse syntaxique revient à résoudre ces contraintes. Le problème majeur de cette seconde approche tient à sa complexité. (Urieli, 2013)

L'efficacité de cette approche dans l'analyse syntaxique repose sur le choix de critères ou traits (en anglais *features*) significatifs qui permettent de caractériser le

comportement syntaxique d'une phrase ou d'un groupe de mots. Ces critères peuvent être statistiques en considérant la fréquence des occurrences ou non statistiques en considérant uniquement les informations linguistiques comme les informations morphologiques.

Comme nous l'avons dit au début, Les méthodes basées sur cette approche utilisent des techniques d'apprentissage automatique, donc nous proposons d'abord quelques définitions concernant l'apprentissage automatique. Ces définitions vont nous permettre d'identifier la fonction, l'objectif, et le résultat attendus de cette approche. Nous présentons ensuite les principaux types d'apprentissage automatique. Enfin nous présentons quelques analyseurs syntaxiques basés sur l'approche numérique.

### 3.2.1 L'apprentissage automatique

Selon Mitchell (Mitchell, 1997) « L'apprentissage automatique est l'étude des algorithmes informatiques qui s'améliorent automatiquement à travers les expériences ».

D'après Sewell, « L'apprentissage automatique est un domaine de l'intelligence artificielle qui s'occupe de l'étude des algorithmes capables de s'améliorer automatiquement à travers l'expérience. En pratique cette idée nécessite la création de programmes qui optimisent un critère de performance à travers l'analyse des données » (Sewell, 2005).

Simon (Simon, 1983) déclare que « L'apprentissage dans un système est indiqué par les changements qu'il subit. Ces changements sont adaptatifs dans le sens où ils rendent possible, au système, de réaliser une même tâche, ou des tâches tirées d'une même population, d'une façon plus efficace et plus efficiente la prochaine fois qu'elle sera réalisée ».

Pour Osório (Osório, 1998) « L'apprentissage automatique se fait par des outils qui permettent d'acquérir, d'élargir et d'améliorer les connaissances disponibles au système ». En général, l'apprentissage implique des processus d'adaptation et de modification des structures de contrôle et/ou de représentation des connaissances du système en question.

À partir de ces définitions, on peut dire que l'apprentissage automatique présente une tentative de comprendre et de reproduire la faculté d'apprentissage relative aux êtres

humains (apprendre, à partir des expériences passées, des choses plus ou moins fondamentales comme la reconnaissance d'une voix, d'un visage familier, apprendre à marcher, à parler, etc.) dans des systèmes formels artificiels. Plus précisément, on cherche à acquérir des règles générales qui représentent les connaissances obtenues à partir d'exemples. En effet, le principe de l'apprentissage automatique consiste à générer dans un premier temps des modèles qui servent ensuite à classer les objets (mots, textes, phrases, formes graphiques, etc.). Contrairement aux autres méthodes sans apprentissage, les méthodes d'apprentissage nécessitent systématiquement un corpus d'entraînement permettant une classification ultérieure sur les données à classer.

#### 3.2.1.1 L'apprentissage supervisé

L'apprentissage est de type *supervisé* si les classes sont prédéterminées et les exemples sont connus et étiquetés. Le système va donc apprendre à classer des données selon un modèle de classement. La formulation de ce type d'apprentissage nécessite l'acquis préalable d'un ensemble fini d'exemples sous forme de paires d'objets : entrée/sortie désirée (typiquement des vecteurs), puis de tenter d'obtenir automatiquement un système capable de trouver de façon relativement fiable la sortie correspondante à toute nouvelle entrée pouvant lui être présentée. Ainsi, l'algorithme apprenant doit être capable de généraliser les données présentes à des exemples non déjà traités. Sa sortie peut être une valeur continue (elle est alors dite fonction de *régression*), ou bien l'étiquette d'une classe prédite de l'objet en entrée (elle sera ainsi dite fonction de *classification*) (Ben Fraj F. , 2010).

#### 3.2.1.2 L'apprentissage semi supervisé

L'apprentissage *semi-supervisé* est une classe de techniques d'apprentissage automatique qui utilise un ensemble de données étiquetées et non-étiquetées. En générale, on a recourt à cette technique lorsqu'on dispose d'un petit ensemble de données étiquetées et d'un grand ensemble de données non étiquetées. Il se situe ainsi entre l'apprentissage supervisé qui n'utilise que des données étiquetées et l'apprentissage non-supervisé qui n'utilise que des données non-étiquetées. Il a été démontré que l'utilisation de données non-étiquetées, en combinaison avec des données étiquetées, permet d'améliorer significativement la qualité de l'apprentissage. Un autre intérêt provient du fait que l'étiquetage des données nécessite l'intervention d'un

utilisateur humain. Lorsque les jeux de données deviennent très grands, cette opération peut s'avérer fastidieuse. Dans ce cas, l'apprentissage semi-supervisé, qui ne nécessite que quelques étiquettes, revêt un intérêt pratique évident (Blum & Mitchell, 1998).

#### 3.2.1.3 L'apprentissage non supervisé

Contrairement à l'apprentissage supervisé, la méthode d'apprentissage non supervisé utilise des données brutes c'est-à-dire, non étiquetées préalablement. Dans ce genre d'apprentissage, il n'y a pas de notion de sortie désirée. Il existe, néanmoins, un ensemble fini de données d'apprentissage, formé seulement d'entrées, sans qu'aucun label ne leur soit rattaché (Russell & Norvig, 2003). L'apprentissage non-supervisé est souvent traité comme un problème d'estimation de densité. Cela a amené les chercheurs à reformuler le problème sous la forme d'un problème de partitionnement (ou *clustering*) des données en sous-groupes (ou *clusters*), qui est plus simple (Amini, 2001).

Dans le domaine du TALN, les chercheurs ont plus tendances à utiliser la technique d'apprentissage supervisé. Cette tendance s'explique par la profusion d'outils et d'algorithme d'apprentissage permettant de produire des modèles statistiques plus performants en des temps toujours plus courts. Malheureusement, cette technique se heurte à la rareté de corpus étiqueté de bonne qualité et de grande taille. Ce fait est beaucoup plus important pour la langue l'arabe.

## 3.3 Analyseurs basés sur l'approche numérique

### 3.3.1 L'analyseur MALT

MALT est un analyseur syntaxique indépendant de la langue qui utilise une approche dirigée par les données. La principale caractéristique de ce système c'est qu'il peut être utilisé pour induire un modèle d'analyse à partir d'un corpus arboré et analyser syntaxiquement des données en utilisant les modèles induits. Il donne en sortie des phrases analysées sous forme de graphe de dépendance.

La méthode d'analyse utilisée se base sur trois composants essentiels:

- Des algorithmes d'analyses déterministes pour la construction d'arbres en dépendance annotés.

- Des modèles basés historique pour prédire les actions non déterministes du parseur.
- Apprentissage discriminatif pour mapper l'historique aux actions du parseur.

Le système n'utilise pas de grammaire, mais repose entièrement sur l'apprentissage inductif depuis les corpus arborés pour l'analyse de nouvelles phrases et sur l'analyse déterministe pour la désambigüisation. Les auteurs affirment que cette combinaison de méthodes garantit que l'analyseur est à la fois robuste et efficace (Nivre, Hall, & Nilsson, 2006).

L'analyseur MALT a été entrainé et testé sur de la version en dépendance du WSJ PennTreebank. Il a été testé de deux façons : une première fois en utilisant un étiqueteur morphologique (Pred) et la deuxième fois avec un corpus préalablement étiqueté (Gold). Les résultats sont présentés dans le tableau suivant.

| | **Pred** | **Gold** |
|---|---|---|
| **Etiquetés** | 86.9% | 87.4% |
| **Non étiquetés** | 88.9% | 89.3% |

**Tableau 2.2 :** Résultats de l'évaluation de l'analyseur MALT

### 3.3.2 L'analyseur syntaxique MICA

MICA est un analyseur syntaxique probabiliste en dépendances pour la langue anglaise (Bangalore, Boulllier, Nasr, Rambow, & Sagot, 2009). Il est basé sur la grammaire d'arbre inséré (Tree Insertion Grammar, TIG) et retourne une analyse profonde en dépendances. Le principe d'analyse de MICA a été conçu selon les trois étapes suivantes :

- Dérivation des Supertag : les supertags sont des arbres élémentaires lexicalisés qui ressemblent aux arbres de la grammaire d'arbres adjoints. Ces Supertags sont dérivés à partir du Penn TreeBank (4727 supertags extraits) et des probabilités leur sont associées directement.
- Génération de la grammaire probabiliste : En premier lieu une grammaire d'arbres insérés(TIG) est extraite depuis le TreeBank avec une table de comptage de fréquences. Ensuite cette grammaire TIG est transformée en une grammaire PCFG lexicalisée qui génère des chaines de Supertags. (Les

informations lexicales utilisées sont comprises dans les Supertags précédemment extraits)

- Dérivation des structures syntaxiques les plus probables à partir des n-meilleurs supertags en utilisant la grammaire PCFG et un analyseur syntaxique utilisant l'algorithme de Earley.

L'analyseur MICA a été entrainé et testé en utilisant la version en dépendance du WSJ PennTreebank. Il a été évalué de deux façons : une première fois avec leur propre étiqueteur morphologique (Pred) et la deuxième fois avec un corpus préalablement étiqueté (Gold) (Bangalore, Boulllier, Nasr, Rambow, & Sagot, 2009). Les résultats obtenus sont présentés dans le tableau suivant :

| | **Pred** | **Gold** |
|---|---|---|
| **Etiquetés** | 85.8% | 97.3% |
| **Non étiquetés** | 87.6% | 97.6% |

**Tableau 2.3 :** Résultats de l'évaluation de l'analyseur MALT

# 4. L'approche hybride

L'approche hybride dans l'analyse syntaxique n'est pas seulement liée aux deux approches symbolique et numérique. En effet, Le concept d'hybridation est beaucoup plus globale et peut être relatif à d'autres aspects de l'analyse syntaxique telle que :

- Le formalisme des grammaires (LFG, HPSG, TAG,…) : Dans une approche hybride, on peut utiliser deux formalismes de grammaire qui se complètent pour avoir une analyse bien détaillée.

- Les techniques d'acquisition des connaissances : Les analyseurs syntaxiques hybrides peuvent aussi combiner l'utilisation des différentes techniques d'apprentissage automatique tel que l'apprentissage semi supervisé et supervisé.

- Les types d'analyse : L'hybridation peut être aussi la combinaison des deux types d'analyse, profonde et de surface, selon les besoins de l'utilisateur.

- Les stratégies d'analyses : Une analyse syntaxique hybride peut être une mixture des deux stratégies ascendante et descendante.

Dans cette thèse, nous nous focalisons sur l'aspect d'hybridation numérique/symbolique. En effet, ce type d'hybridation intègre à la fois, une analyse linguistique à base de règles symboliques ainsi qu'une analyse numérique utilisant des calculs statistiques ou des modèles statistiques. Elle vise à tirer profit des avantages fournis par l'approche symbolique et l'approche numérique. Les stratégies d'hybridation peuvent êtres regroupées en deux catégories ou architectures : hybridation forte et hybridation faible (Foth, 2006). Dans ce qui suit nous allons présenter ces deux stratégies en mettant l'accent sur leurs avantages et leurs inconvénients

## 4.1 Hybridation faible

Cette stratégie correspond à une hybridation dite en série qui suit une approche en pipeline : le système correspond à une chaine de processus faisant intervenir successivement un module numérique puis un module symbolique (ou inversement). Le second module travaille sur les sorties du premier. Le schéma suivant illustre les différents modules d'une hybridation faible (Foth, 2006)

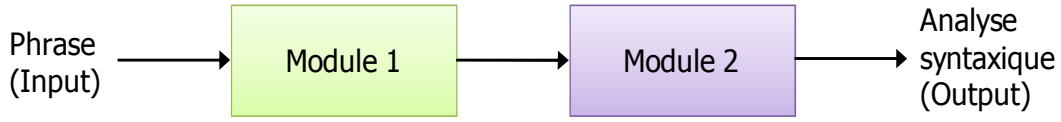



**Figure 2.5 :** Architecture globale du modèle d'hybridation faible

Dans le cas d'une hybridation faible, plusieurs cas de collaboration entre les deux modules peuvent être envisagés et ceci selon les besoins de l'utilisateur dont voici les plus communs :

- Le deuxième module essaie de corriger les erreurs possibles dans l'analyse fournie par le premier module. Par exemple si le premier module est basé sur une approche statistique et que le deuxième module est inspiré de l'approche symbolique, le module symbolique se compose alors généralement d'un nombre assez restreint de règles rendant compte des irrégularités les plus fréquentes de la langue.

- La tâche d'analyse syntaxique est répartie entre les deux modules. Un premier module détermine une partie des informations syntaxiques et le deuxième

complète l'arbre syntaxique dans sa totalité. Par exemple, le premier module peut définir le type de la phrase (phrase nominale ou phrase verbale) et ainsi guider le deuxième module pour générer l'analyse syntaxique complète de la phrase. Ou encore le premier module peut déterminer la structure syntaxique de surface de la phrase tandis que le deuxième détermine les structures profondes.

## 4.2 Hybridation forte

Cette stratégie constitue la contrepartie de la solution précédente, elle consiste à :

- Intégrer un ensemble de contraintes symboliques dans l'analyse statistique.
- Ou inversement, intégrer des données statistiques au sein d'une analyse symbolique.

Cependant, la première stratégie est la plus utilisée dans ce type d'hybridation (Foth, 2006).La figure suivante présente l'architecture d'une hybridation forte

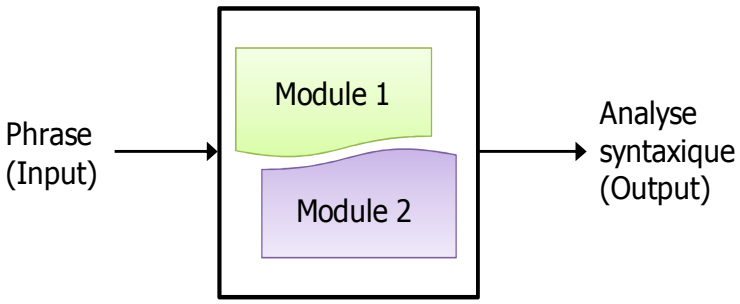



**Figure 2.6 :** Architecture de l'hybridation forte

Cette stratégie est rarement utilisée par rapport à l'hybridation faible. Ceci est entre autres dû à la difficulté de mise en œuvre d'une telle approche.

## 4.3 Analyseurs basés sur l'approche hybride

### 4.3.1 L'analyseur de Sennrich et al.

Sennrich (Sennrich, Schneider, Volk, & Martin, 2009) a conçu un analyseur syntaxique hybride en dépendance et bi-lexicalisé pour l'allemand. Il combine l'utilisation d'une Grammaire Fonctionnelle en Dépendance(GFD) construite manuellement avec un

lexique statistique obtenu à partir du corpus allemand TuBa-D/Z. Les auteurs utilisent le lexique statistique pendant l'analyse syntaxique basée sur la grammaire GFD pour réduire les possibilités d'analyse et réordonnancer les analyses qui satisfassent les contraintes de la grammaire selon leur probabilités.

Les auteurs ont utilisé environ 32000 phrases pour la phase d'apprentissage et ils ont consacrés 3000 phrases annotées morphologiquement pour l'évaluation de leur analyseur hybride. Ils ont obtenu une précision de 88,6%, un rappel de 82,6% et un F-mesure de 78,4%.

### 4.3.2 L'analyseur de Sigone et al.

Sigone (Sigogne, Constant, & Laporte, 2014) présent les résultats d'une évaluation sur l'intégration des données issues d'un lexique syntaxique, le Lexique Grammaire, dans un analyseur syntaxique. Les chercheurs montrent qu'en modifiant le jeu d'étiquettes des verbes et des noms prédicatifs, un analyseur syntaxique probabiliste non lexicalisé obtient des performances accrues sur le français. Ces performances sont obtenues, principalement, grâce à la hiérarchie des tables des verbes qui permet de limiter l'ambiguïté en termes de nombre de classes associées à un verbe. Ceci a pour effet d'augmenter la couverture des verbes annotés selon le niveau de granularité utilisé.

Pour entrainer et évaluer leur analyseur, les auteurs ont utilisé le corpus arboré du français, le French Treebank (FTB) (Abeillé, Clément, & Toussenel, 2003), contenant 20860 phrases et 540648 mots issus du journal Le Monde (version de 2004). Ils ont effectué l'évaluation selon la méthode de validation croisée sur 10% du FTB. Ils ont obtenu un F-mesure de 85.32%.

### 4.3.3 L'analyseur de Francopoulo

Gil Francopoulo a développé un analyseur syntaxique suivant une hybridation faible nommé TagParser (Francopoulo, 2009) est un analyseur syntaxique pour le français et l'anglais. Il a été conçu de manière à permettre un développement incrémental de la grammaire. Il enchaine les principaux modules suivants sous forme d'un pipeline : un ségmenteur, un analyseur morphologique, un chunker (TagChunker) et un calculateur de relation syntaxique. Le chunker appartient à la famille des analyseurs syntaxiques hybrides qui combinent des informations symboliques (via un dictionnaire et un automate) et un modèle probabiliste (via une matrice de pondération).

Tagparser commence par construire une grammaire à partir d'un corpus annoté manuellement ainsi que des matrices statistiques destinées à l'algorithme d'élagage. Un automate de reconnaissance, utilisant la grammaire, est appliqué sur la phrase. Si un seul résultat est produit, il est alors considéré comme étant le bon résultat. Si plus d'un résultat est trouvé, alors un algorithme d'élagage utilisant les matrices statistiques est appliqué pour ne retenir qu'un seul résultat. Si l'analyse statistique n'aboutit pas, alors des grammaires sont combinées afin de produire des analyses candidates. Si un seul résultat est produit, alors, il est considéré comme étant le bon résultat et si plus d'un résultat est produit alors, l'algorithme d'élagage est appliqué de nouveau afin de ne retenir qu'un seul résultat ; Sinon, c'est un échec.

La sortie d'analyse comporte trois types de résultat : les constituants sans enchâssement, les relations syntaxiques et les entités nommées.

### 4.3.4 L'analyseur de Charniak et Johnson

Charniak et Johnson (Charniak & Johnson, 2005) proposent le reclassement discriminatif des 50 meilleures analyses générées par un analyseur à base d'une grammaire. Cet analyseur génère les 50 meilleures analyses qui sont ensuite placées comme entrée pour un ré-ordonnanceur. Ce dernier se base l'algorithme maximum d'entropie pour sélectionner la meilleure analyse possible. L'auteur a défini 13 attributs pour décrire les données nécessaires pour l'apprentissage.

Les concepteurs ont évalué la performance de leur système selon la métrique standard PARSEVAL. Ils ont entrainé le système de génération des n-meilleurs arbres avec les sections 2-21 du Penn Treebank et ont utilisé la section 24 en tant que données de développement pour régler les paramètres de mixages du modèle de lissage. De la même manière, le ré-ordonnanceur a été entrainé sur la section 2-21 du Penn Treebank et la constante C a été ajusté pour maximiser le F-score en utilisant la section 24 du corpus (Charniak & Johnson, 2005).

# 5. Discussion sur les approches d'analyses syntaxiques

Nous avons commencé ce chapitre par la présentation de l'approche symbolique qui se base sur les grammaires pour décrire le comportement syntaxique d'une langue. Toutefois cette approche présente des limites dont voici les plus importantes :

- L'élaboration manuelle d'une grammaire à base de connaissance représente une opération complexe et fastidieuse. En plus, les concepteurs sont confrontés à des problèmes sévères de maintenance et d'extension de la grammaire à partir d'une certaine taille critique. Les grammaires symboliques ont ainsi du mal à tenir le pari contradictoire d'une large couverture associée à une grande précision : tout enrichissement de la grammaire se traduit en effet généralement par une perte de robustesse due au risque de contradictions entre la grammaire initiale et les nouvelles connaissances introduites.

- De plus, cette large couverture aura tendance à alourdir le système d'analyse et pénaliser sa vitesse d'exécution. En effet les analyseurs devront faire face à une explosion exponentielle des possibilités d'analyse due au grand nombre de règles exigées par la large couverture.

- Même si l'introduction des grammaires lexicalisées représente une avancée de ce point de vue, les approches symboliques sont les plus adaptées à la modélisation des irrégularités de la langue qu'à celle de ses exceptions. Considérées collectivement, ces dernières ont pourtant une fréquence d'occurrence non négligeable.

Face à ce constat, il serait erroné de conclure à l'inadéquation des approches symboliques. En fait, plus que la limite d'un paradigme donné, c'est la complexité de la modélisation symbolique qui est révélée par l'insuffisance des résultats obtenus. La structure linguistique d'une phrase recouvre en effet deux dimensions que tentent de rendre compte les grammaires symboliques, à la différence des modèles statistiques actuels. En plus, selon la littérature, l'efficacité de l'approche symbolique dans le traitement des phrases relativement petites n'est plus à prouver ni en terme de qualité, ni en termes de temps d'exécution.

Nous nous sommes intéressés en deuxième lieu à l'approche numérique basée essentiellement sur le développement de modèles statistiques pour les langues

naturelles. Ces modèles ont permis des avancés spectaculaires des applications du TALN au cours des deux dernières décennies.

Tout le monde s'accorde à dire que les modèles numériques utilisant l'apprentissage supervisé, qui est la technique la plus utilisée dans le TALN, exigent l'utilisation de grand corpus d'apprentissage annotés pour tester l'efficacité des modèles générés. Cette ressource est très rare et n'est pas disponible librement sur internet, et surtout pour les langues sémitiques comme l'arabe. La construction de telle ressource est une tache de longue haleine qui demande de grands moyens à la fois humains et matériels. De plus la définition des critères d'apprentissage efficaces, et décrivant de la façon la plus fidèle le comportement syntaxique de la langue est une tâche difficile et complexe.

Prenant en considération toutes les difficultés de mise en œuvre de l'approche numérique, on peut se demander aussi si ces techniques numériques n'ont pas atteint leurs limites sur la modélisation de la dimension linéaire des langues. On sait en effet que les modèles statistiques actuels, basés sur la seule observation de séquences restreinte d'entités linguistiques (mots, catégories syntaxiques, etc.), ne peuvent rendre compte de la structure profonde des énoncés. Ce problème est largement caractérisé par la communauté scientifique comme l'insuffisance principale des modèles numériques. Sans oublier pour autant qu'une compréhension fine des phrases nécessite une caractérisation précise de leurs structures linguistiques et ne peut donc reposer seulement sur une modélisation statistique.

Après étude des différentes approches pour le domaine de l'analyse syntaxique, ainsi que les différentes méthodes utilisées, à notre connaissance, il n'existe pas des travaux de recherche qui ont résolut le problème d'analyse syntaxique de la langue arabe avec une performance digne de l'état de l'art d'autre langue comme l'anglais et le français. Ceci est surement dû d'une part aux particularités de la langue arabe et d'autre part à la conception des approches d'analyse syntaxique orientée vers les langues indo-européennes.

Il faut, cependant, noter que le traitement de la tâche d'analyse syntaxique de l'arabe standard en combinant les méthodes numériques et symboliques, pourrait permettre de franchir un palier et de s'approcher un peu plus des résultats d'analyseurs des langues indo-européennes. Le paradigme d'analyse des phrases en se basant sur une approche hybride qui privilégiera l'utilisation des techniques numériques et symboliques peu servir, à notre point de vu, à être un pas en avant vers la génération d'une analyse syntaxique de meilleure qualité. Le choix de la stratégie d'hybridation est déterminant

pour pouvoir améliorer les performances d'une analyse syntaxique numérique ou symbolique. Pour ce faire, nous pensons que la meilleure façon de faire ce choix est de partir d'expériences d'analyses réelles et d'ensuite, essayer de l'optimiser avec une stratégie hybride. C'est ce que nous allons faire dans le chapitre suivant.

# 6. Conclusion

Dans ce chapitre, nous avons exploré quelques méthodes qui ont été proposées pour résoudre la problématique de génération d'analyse syntaxique d'une phrase. Il semble bien que les méthodes purement statistiques, simplement implantées et rapidement adaptables à d'autres domaines soient limitées en ce sens qu'elles n'ont pas une vision détaillée des phrases analysées. Les méthodes fondées sur la linguistique nécessitent des grammaires symboliques et des ressources linguistiques avancées et ne peuvent être donc appliquées qu'à des domaines restreints. En revanche, les méthodes hybrides qui tiennent compte à la fois des techniques symboliques et numériques sont plus prometteuses. Malgré les avancées enregistrées par ces méthodes, la qualité des analyses produites reste toujours à améliorer. Dans le chapitre suivant, nous donnons notre proposition d'hybridation pour une meilleure analyse syntaxique de l'arabe.

# Chapitre 3

## Analyse syntaxique hybride pour l'arabe standard

# 1. Introduction

Suite à notre étude des approches d'analyse syntaxique, nous estimons que l'approche hybride peut être la plus efficace et la plus performante car elle vise à profiter des avantages des deux approches numérique et symbolique.

Néanmoins, il existe beaucoup de manières pour concevoir une stratégie d'hybridation. Dès lors, pour choisir la bonne stratégie d'hybridation, nous avons choisi de commencer par la proposition d'une méthode d'analyse syntaxique pour l'arabe standard basé sur l'approche symbolique. Ensuite, en se basant sur les résultats obtenus par cette méthode et ceux obtenus par la méthode d'analyse syntaxique numérique développé dans notre équipe (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015), nous proposerons une nouvelle méthode basée sur l'approche hybride que nous pensons capable d'améliorer les résultats obtenus par la méthode symbolique.

Dans la suite de ce chapitre, nous allons détailler le travail réalisé en commençant par la présentation de la méthode symbolique. Puis nous discutons les résultats obtenus. Enfin, nous proposons une méthode hybride et nous détaillons ses principales étapes.

# 2. Méthode proposée pour l'analyse syntaxique symbolique

Comme nous l'avons expliqué dans le chapitre précédent, d'une manière générale, une méthode d'analyse symbolique se base principalement sur l'utilisation d'un algorithme d'analyse et d'une grammaire de la langue. Les algorithmes d'analyse sont nombreux mais, il faut savoir faire le choix adéquat selon les besoins de l'analyse. Cependant, à notre connaissance, il n'existe pas de grammaire arabe, de bonne qualité et disponible librement qu'on puisse utiliser dans le cadre de ce travail. Nous avons donc commencé par la construction d'une grammaire syntaxique de type hors contexte probabiliste (Probabilisitc Context Free Grammar, PCFG). Ensuite, nous utilisons cette grammaire avec l'aide d'un algorithme d'analyse pour analyser syntaxiquement des phrases en arabe standard. La figure 3.1 expose l'architecture générale de notre méthode en montrant le workflow entre les différents processus.

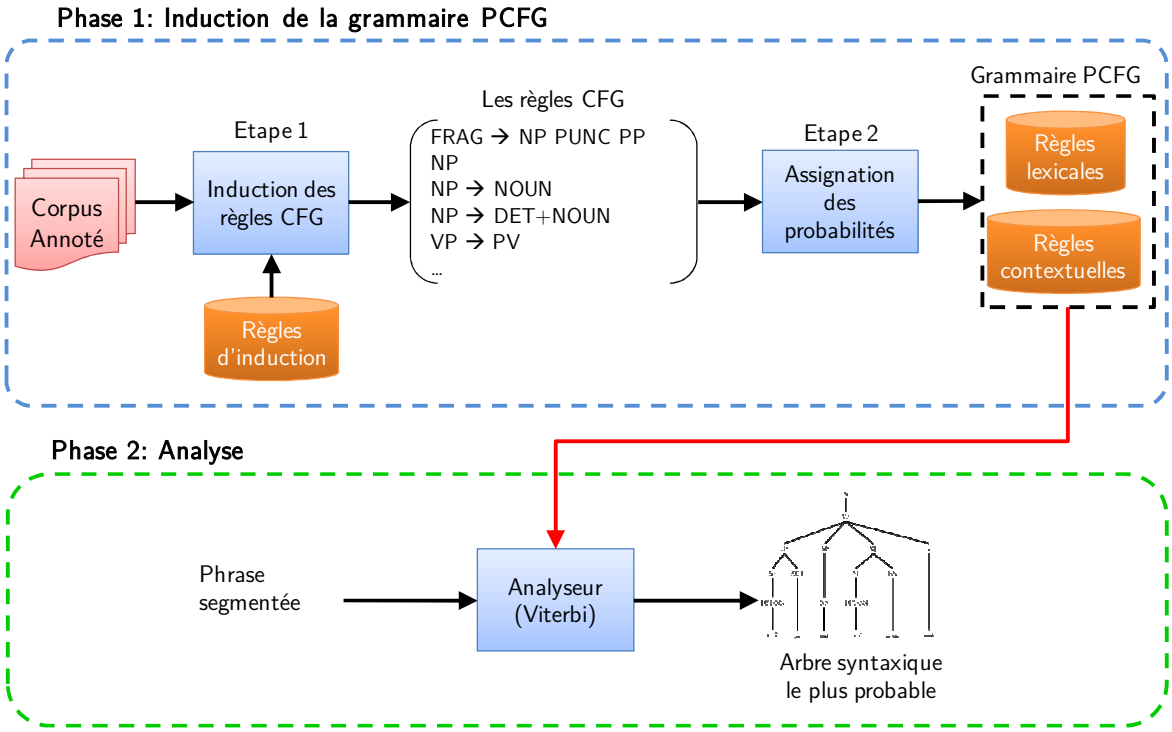



**Figure 3.1 :** Architecture globale de notre analyseur symbolique

La méthode d'analyse symbolique, que nous proposons, repose sur deux phases principales à savoir la phase d'induction de la grammaire PCFG et la phase d'analyse (voir la figure 3.1). La première phase consiste en la génération d'une grammaire PCFG arabe à partir d'un corpus annoté. La deuxième phase exploite la grammaire générée pour analyser des phrases syntaxiquement.

Nous donnons, dans ce qui suit, une description détaillée de ces deux phases.

## 2.1 Induction de la grammaire hors contexte probabiliste

L'objectif de cette phase est d'induire automatiquement une grammaire PCFG à partir d'un corpus annoté morpho-syntaxiquement (Khoufi, Aloulou, & Hadrich Belguith, 2015). Cette phase d'induction se déroule en deux étapes: la première se charge d'induire les règles hors contexte classiques (CFG) depuis les arbres morphosyntaxiques du corpus annoté. Ensuite, nous assignons les probabilités à chaque règle en tenant compte des occurrences d'apparition des deux parties de la règle CFG dans le corpus annoté. L'application de ces deux étapes nous a permit d'obtenir une grammaire PCFG. La figure 3.2 montre l'enchainement de ces deux sous étapes.

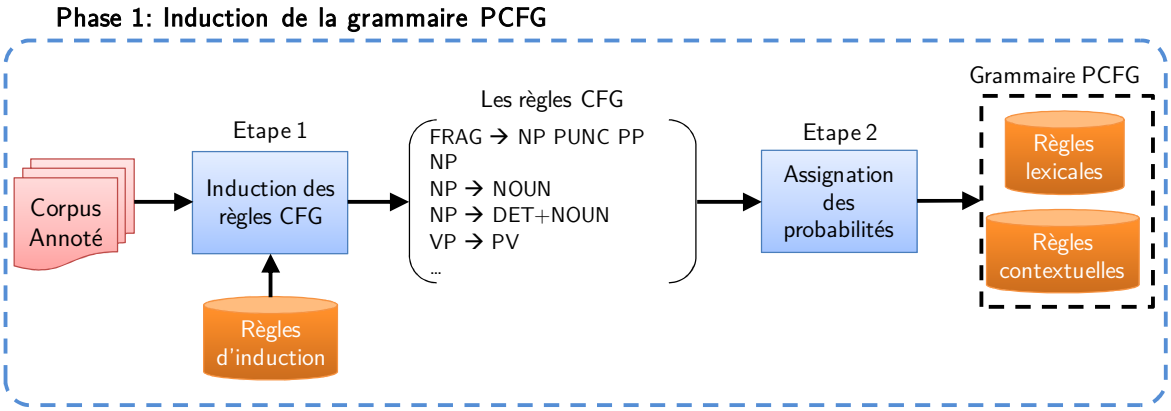



**Figure 3.2 :** Les étapes de la phase d'induction de la grammaire PCFG

Dans ce qui suit, nous commençons par donner une brève description formelle de la grammaire PCFG, ensuite nous détaillons le processus de construction de cette grammaire.

### 2.1.1 Grammaire PCFG: définition formelle

Une grammaire probabiliste hors-contexte est une extension naturelle de la grammaire hors contexte (CFG) et se définit par le 5-tuplet < N, T, R, S, P> où:

- N est l'ensemble fini des symboles non terminaux.
- T est l'ensemble fini des symboles terminaux.
- R est l'ensemble fini des règles *ri* de la forme: $A \rightarrow \alpha$, $A \in N$, $\alpha \in (N \cup T)$
- S est l'axiome de départ
- P est l'ensemble des probabilités *pi* associées aux règles *ri* telles que : $\sum P(A \rightarrow \alpha)=1$, $\forall A \in N$ et $\alpha \in (N \cup T)$

### 2.1.2 Induction des règles hors contexte

Une étude approfondie du corpus annoté PATB nous a permis de dégager des principes d'induction formalisés, sous forme de règles, pour obtenir des règles hors contexte (CFG). Nous nous sommes focalisés sur les arbres morphosyntaxiques et nous avons identifié les règles d'induction suivantes :

- R1: Racine de l'arbre → Symbole initial (S)

- R2: Nœud interne de l'arbre → Symbole non terminal (N)
- R3: Feuille de l'arbre → Symbole terminal (T)
- R4: Fragment de l'arbre → règle hors contexte (R)

Nous avons remarqué que chaque arbre syntaxique peut être considéré comme une séquence de règles hors contexte. Tous les arbres syntaxiques du corpus ont un symbole racine qui marque le début de la phrase. Ce symbole est le même pour tous les arbres. Nous avons donc choisi ce symbole comme symbole initial de notre grammaire (S). Les nœuds internes de l'arbre sont considérés comme symboles non terminaux (N). L'ensemble des feuilles de l'arbre (les mots) représente l'ensemble des symboles terminaux (T). Les arcs entre les nœuds de l'arbre représentent les relations hiérarchiques entre les symboles non terminaux et aussi entre les symboles non terminaux et les symboles terminaux. Cette représentation est utilisée pour l'induction des règles de production hors contextes (R). Pour mieux comprendre ces règles d'induction que nous venons de décrire, nous présentons dans la figure 3.3 un exemple d'application sur un arbre morphosyntaxique d'une phrase en arabe standard.

Soit la phrase arabe suivante :

**عَادَت عَقَارِبُ الزَمَنِ فَجْأَةً إِلَى الوَرَاءِ.**

Translitération de Buckwalter : EaAdat EaqaAribu Alzamani fajoOapF IilaY AlwaraA'i.

*(Les aiguilles du temps sont soudainement revenues en arrière)*

Cette phrase a la représentation arborescente syntaxique suivante :

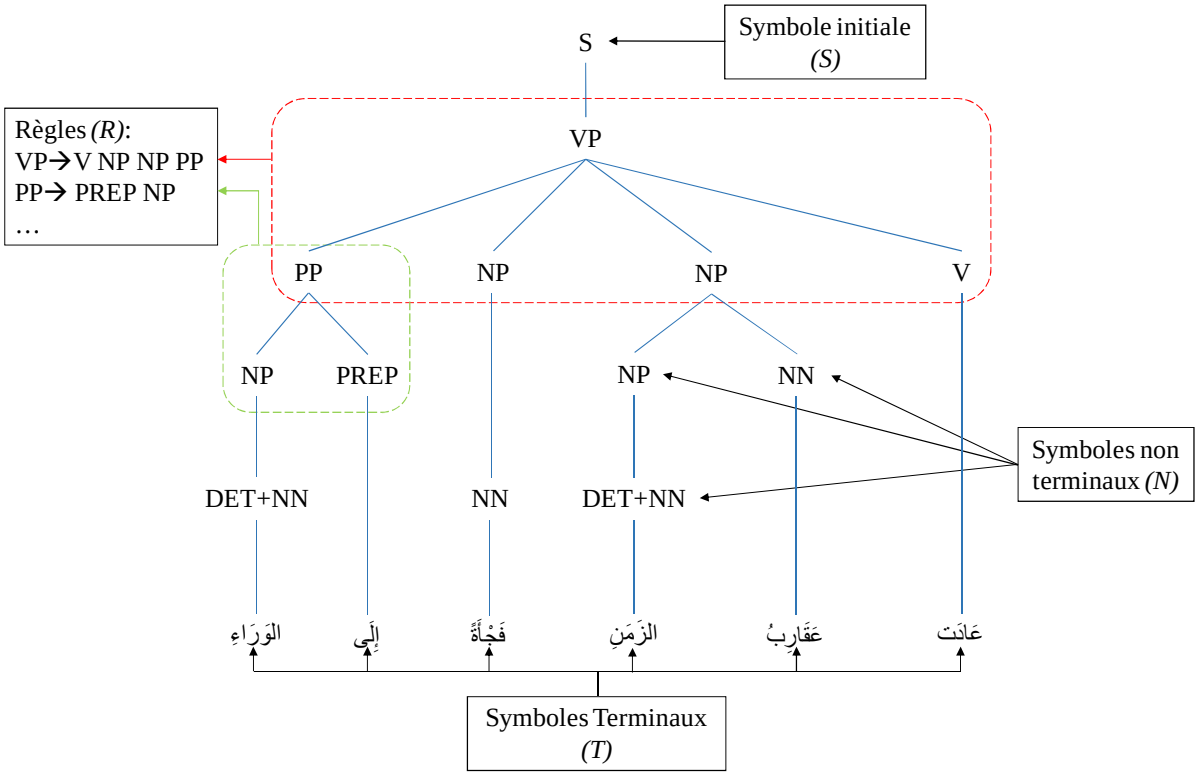



**Figure 3.3 :** Induction des éléments de la grammaire à partir d'un arbre morphosyntaxique

Le corpus PATB est très riche en informations. Il contient plusieurs annotations comme le genre, le nombre, l'humeur, etc. Le PATB est annoté en utilisant un large ensemble d'annotations ce qui lui confère un grand niveau de granularité. En effet, le corpus utilise un ensemble de 498 annotations pour décrire tous les traits morphologiques. Il utilise aussi 22 annotations pour décrire les catégories syntaxiques et 20 autres annotations qui décrivent les relations sémantiques entres les mots. En plus, les mots vides, qui sont très nombreux dans l'arabe standard, sont aussi annotés avec des annotations spécifiques. L'étude du PATB nous a montré l'existence de plusieurs informations utilisées dans l'annotation du treebank qui sont :

- Le cas du mot
- Le genre
- Le nombre
- Le mode de conjugaison
- Le déterminant

- La catégorie lexicale
- La catégorisation syntaxique
- La numérotation

L'incorporation de toutes ces informations dans notre grammaire augmente d'une part sa complexité et d'autre part sa taille. En effet, la taille de la grammaire dépend essentiellement du niveau de granularité des catégories qu'elle décrit : plus le niveau de granularité est haut, plus la grammaire est large et complexe, et plus elle respecte les spécificités de la langue. Considérons par exemple cet ensemble d'annotation :

- ADJ+CASE_DEF_ACC : adjective défini accusatif
- ADJ+CASE_DEF_GEN : adjectif définie génitif
- ADJ+CASE_DEF_NOM : adjective défini nominatif
- ADJ+CASE_INDEF_ACC : adjective indéfini accusatif
- ADJ+CASE_INDEF_GEN : adjectif indéfinie génitif
- ADJ+CASE_INDEF_NOM : adjective indéfini nominatif

Cet ensemble décrit la catégorie « adjectif » avec différents niveaux de granularité. Si le niveau de granularité est réduit à son minimum, cet ensemble sera réduit à la seule annotation « ADJ ». Cette réduction influence le nombre de règles générées de la grammaire. Pour éviter l'explosion combinatoire des possibilités d'analyse tout en gardant une bonne consistance de la grammaire, il est nécessaire de procéder à une réduction systématique du nombre des étiquettes, plus précisément le nombre des étiquettes morphologiques dont le nombre est très important. Par conséquent, nous avons opté pour une réduction de l'ensemble des annotations morphologiques et n'avons gardé que les plus importantes, et ceci en se référant à l'avis d'un expert linguiste, pour faciliter l'utilisation de la grammaire dans les applications de TALN. Cette réduction ne se réduit pas à une simple sélection des étiquettes du plus haut niveau mais plutôt à la sélection des étiquettes dont nous avons jugées qu'elles ont de la valeur et du sens pour la description syntaxique de la langue arabe. Par exemple, la catégorie « adjectif » est décrite par 75 annotations différentes en tenant compte du plus haut niveau de granularité. Apres réduction, nous avons choisi de garder quatre annotations qui sont :

- ADJ : Adjectif.
- ADJ.VN : Adjectif qui joue le rôle d'un verbe dans la phrase.
- ADJ_COMP : Adjectif qui joue le rôle d'un complément dans la phrase.
- ADJ_NUM : Adjectif numérique.

La liste complète des annotations utilisées au cours du processus de création de notre grammaire est présentée dans l'annexe.

L'application de notre méthode d'induction sur l'exemple de la figure 3.3, nous donne la grammaire CFG suivante composée de règles contextuelles et de règles lexicales :

| **Règles contextuelles** | **Règles lexicales** |
|---|---|
| S→VP | V→عَادَت |
| VP→V NP NP PP | NN→عَقَارِبُ |
| NP→NN NP | DET+NN→الزَمَنِ |
| NP→DET+NN | NN→فَجْأَةً |
| NP→NN | PREP→إِلَى |
| PP→PREP NP | DET+NN→الوَرَاءِ |

**Tableau 3.1 :** Grammaire CFG obtenue par induction à partir de l'exemple de la figure 3.3

Une fois les règles CFG induites, nous passons à l'étape suivante d'assignation des probabilités que nous détaillons dans la section ultérieure.

### 2.1.3 Assignation des probabilités

À l'issue de l'étape précédente, nous obtenons un ensemble de règles hors contexte. La grammaire que nous venons de générer est certes ambigus puisque elle donne plus d'un arbre syntaxique à une même chaine en entrée. Cette étape de probabilisation permet de pallier à ce problème en affectant une probabilité à chaque arbre syntaxique candidat et ainsi permettre le choix de l'arbre le plus probable.

Cette grammaire contient plusieurs règles qui partagent la même partie gauche comme par exemple la règle NP → NN NP et la règle NP→NN. Nous avons recensé 1611 règles avec NP comme partie gauche. Le formalisme PCFG propose d'assigner une probabilité (P) à chaque règle qui permet de donner une importance à cette règle pendant la phase d'analyse, selon sa fréquence d'apparition dans le corpus. Pour cela,

nous utilisons la formule suivante qui tient compte des fréquences d'apparition des règles dans le corpus :

$$P(X \rightarrow Y) = \frac{Fréquence\ (X \rightarrow Y)}{Fréquence\ (X)} \qquad (3.1)$$

Avec :

- P(X→Y) : la probabilité assignée à la règle X→Y
- Fréquence (X→Y) : le nombre d'apparition de la règle X→Y dans le corpus.
- Fréquence (X) : le nombre de règles dans le corpus qui ont la même partie gauche X.

Appliquons ce principe sur un exemple de règle du tableau 3.1. La règle VP→V NP PP est présente dans le corpus 109 fois, et nous avons compté 1311 règles qui ont la même partie gauche VP. Nous obtenons donc une probabilité de :

$$P(VP \rightarrow V\ NP\ NP\ PP) = \frac{109}{1311} = 0{,}08314264$$

Le tableau 3.2 présente les règles PCFG induites à partir de l'exemple de la figure 3.3. Les probabilités indiquées dans ce tableau sont calculées à partir de l'exemple de la figure 3.3 seulement.

| **Règles contextuelles** | **Probabilité** | **Règles lexicales** | **Probabilité** |
|---|---|---|---|
| S→VP | [1.0] | V→عَادَت | [1.0] |
| VP→V NP NP PP | [1.0] | NN→عَقَارِبُ | [0.5] |
| NP→NN NP | [0.33] | DET+NN→الزَمَنِ | [0.5] |
| NP→DET+NN | [0.33] | NN→فَجْأَةً | [0.5] |
| NP→NN | [0.33] | PREP→إِلَى | [1.0] |
| PP→PREP NP | [1.0] | DET+NN→الوَرَاءِ | [0.5] |

**Tableau 3.2 :** Grammaire PCFG obtenue par induction depuis l'exemple de la figure 3.2

L'application de cette étape permet d'obtenir une grammaire PCFG arabe. Notons que la somme des probabilités des règles qui ont la même partie gauche doit être exactement égale à 1.

### 2.1.4 La grammaire obtenue

Suite à la phase d'induction de la grammaire nous obtenons une grammaire PCFG arabe qui se compose de règles lexicales et de règles contextuelles. Les règles lexicales décrivent le lexique de la grammaire et les règles contextuelles décrivent les relations entre les symboles non terminaux. Le tableau 3.3 présente le nombre de règles de notre grammaire.

| | |
|---|---|
| **Règles contextuelles** | 4661 |
| **Règles lexicales** | 38901 |
| **Total** | 43562 |

**Tableau 3.3 :** Nombre de règles de la grammaire PCFG arabe obtenue

Le tableau 3.4 présente une répartition des règles les plus fréquentes selon l'étiquette de sa partie gauche.

| **Symbole de la partie gauche** | NP | VP | S | ADJP | PP |
|---|---|---|---|---|---|
| **Nombre de règles** | 1611 | 1133 | 1013 | 243 | 118 |

**Tableau 3.4 :** Nombre de règles les plus fréquentes dans le corpus

Vu que la grammaire a une taille conséquente, nous avons essayé de donner un échantillon représentatif de la grammaire en indiquant un extrait de 20 règles pour chaque étiquette de la partie gauche de la règle. Cet extrait est représenté dans l'annexe A.

Au terme de cette section, nous avons crée les ressources nécessaires pour notre méthode symbolique. Dans ce qui suit, nous détaillons le processus d'analyse syntaxique.

## 2.2 Analyse syntaxique

Rappelons que l'analyse syntaxique symbolique repose sur deux éléments essentiels : la grammaire de la langue et l'algorithme d'analyse. À ce niveau, nous avons créé une grammaire de la langue arabe, il reste alors à choisir l'algorithme d'analyse adéquat. Dans ce qui suit, nous commençons par une justification du choix de l'algorithme d'analyse. Ensuite, nous présentons le processus d'analyse en soit. Enfin nous exposons les résultats de l'évaluation de la méthode d'analyse symbolique.

### 2.2.1 Choix de l'algorithme d'analyse

Il existe plusieurs algorithmes possibles pour effectuer la tâche d'analyse syntaxique en utilisant la grammaire induite PCFG, dont les plus connus sont CYK (Younger & Kasami, 1978), Earley (Earley, 1970) et Viterbi (Viterbi, 1967). L'efficacité de ces algorithmes a été déjà démontrée depuis longtemps, cependant, chacun d'eux a des avantages et des inconvénients. En nous basant sur étude comparatives de ses caractéristiques, nous avons opté pour l'utilisation de l'algorithme d'analyse de Viterbi pour les raisons suivantes :

L'algorithme de Viterbi peut être utilisé avec n'importe quelle grammaire PCFG (Viterbi, 1967). Cet algorithme n'impose aucune contrainte, ni sur la forme, ni sur le contenu de la grammaire. Par contre, l'algorithme CYK exige que la grammaire soit sous la forme normale de Chomsky. Cette transformation aura comme conséquence l'augmentation de la taille de la grammaire et par conséquent l'augmentation du temps d'analyse.

Le temps d'exécution est un élément primordial pour un algorithme d'analyse syntaxique. Ce temps est fondamentalement lié à leur complexité algorithmique. L'algorithme de Viterbi a une complexité linéaire (*o(n)*) (Petit & Christiansen, 2009) (Pradet, 2011) qui est inferieure à la complexité des deux autres algorithmes (complexité cubique (*o(n3)*)) (Clément, Gerdes, & Marlet, 2009) (Kallmeyer, Maier, & Parmentier, 2009). Cette différence de complexité offre à l'algorithme de Viterbi un temps d'exécution plus rapide que les deux autres algorithmes ce qui lui confère un net avantage.

La figure 3.4 présente le pseudo code que nous avons utilisé pour l'implémentation de l'algorithme de Viterbi.

```
| Create an empty most likely constituent table, *MLC*.
| For width in 1...len(text):
|   For start in 1...len(text)-width:
|     For prod in grammar.productions:
|       For each sequence of subtrees [t[1], t[2], ..., t[n]] in MLC,
|         where t[i].label()==prod.rhs[i],
|         and the sequence covers [start:start+width]:
```

```
|          old_p = MLC[start, start+width, prod.lhs]
|          new_p = P(t[1])P(t[1])...P(t[n])P(prod)
|          if new_p > old_p:
|            new_tree = Tree(prod.lhs, t[1], t[2], ..., t[n])
|            MLC[start, start+width, prod.lhs] = new_tree
| Return MLC[0, len(text), start_symbol]
```

Ou :

- MLC : Table des constituants les plus probables.
- Text : La chaine en entrée.
- Prod : Règle sélectionnée.
- Grammar.production : Les règles de la grammaire.

**Figure 3.4 :** Pseudo code de l'algorithme de Viterbi

L'analyse de Viterbi est une analyse ascendante qui utilise les principes de la programmation dynamique pour trouver l'analyse la plus probable. La programmation dynamique s'appuie sur un principe simple, appelé le principe d'optimalité de Bellman : toute solution optimale s'appuie elle-même sur des sous-problèmes résolus localement de façon optimale. Concrètement, cela signifie que l'on peut déduire une ou la solution optimale d'un problème, en combinant des solutions optimales d'une série de sous-problèmes. Les solutions des problèmes sont calculées de manière ascendante, c'est-à-dire qu'on débute par les solutions des sous-problèmes les plus petits pour ensuite déduire progressivement les solutions de l'ensemble.

Dans le cas de l'analyse syntaxique, l'analyse de Viterbi procède par remplissage de la table des constituants les plus probables. Cette table enregistre les fragments de l'arbre les plus probables pour chaque nœud de celui-ci.

### 2.2.2 Processus d'analyse

Cette phase reçoit, en entrée, une phrase segmentée écrite en arabe standard. Elle utilise la grammaire PCFG et l'algorithme d'analyse pour analyser syntaxiquement la

phrase. Nous obtenons à la suite de cette phase l'arbre syntaxique le plus probable comme le montre la figure 3.5.

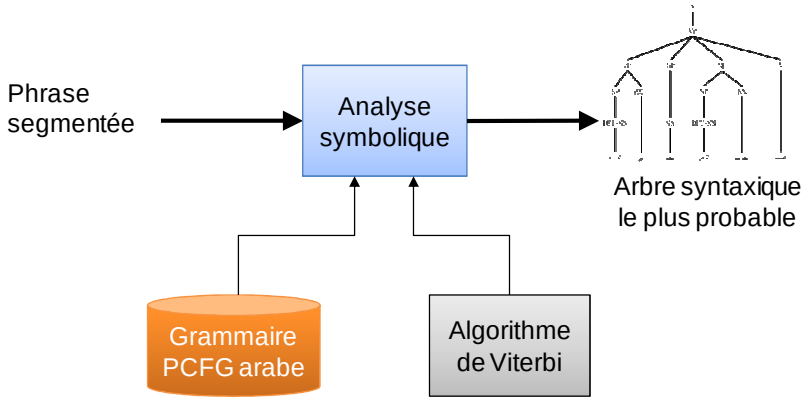



**Figure 3.5 :** Architecture de l'analyse symbolique

### 2.2.3 Evaluation et résultats obtenus

Nous avons développé un prototype logiciel pour pouvoir évaluer notre méthode d'analyse syntaxique symbolique. En effet ce prototype permet d'induire la grammaire PCFG arabe à partir du corpus PATB et d'analyser des phrases. Ce prototype sera présenté dans le chapitre 4 et nous nous contentons dans cette section de présenter le protocole expérimental et les résultats obtenus.

#### 2.2.3.1 Protocole expérimental

Pour tester et évaluer la performance de notre méthode d'analyse syntaxique symbolique, nous avons utilisé le corpus PATB en suivant les recommandations de l'atelier "Johns Hopkins 2005 Workshop"[3] qui ont été utilisées dans le développement de l'analyseur « Stanford Parser ». Nous avons ainsi utilisé une partie du corpus pour l'induction de la grammaire (90 %) et l'autre partie (10 %) pour l'évaluation de la méthode. Le corpus de test est composé de 1650 phrases extraites du corpus PATB. La longueur moyenne des phrases de test est d'environ 21 mots (longueur entre 3 et 40 mots).

[3] http://nlp.stanford.edu/software/parser-arabic-data-splits.shtml

### 2.2.3.2 Métriques utilisées

Afin d'évaluer notre approche, nous utilisons les métriques standards de l'évaluation des analyseurs syntaxiques ; à savoir la Précision, le Rappel, et la F-mesure dont les interprétations varient légèrement selon le domaine d'application.

- La précision : La précision représente le taux des constituants corrects générés par l'analyse par rapport au nombre de constituants générés par l'analyse de la phrase de test.

$$Précision = \frac{cardinalité\ (Référence\ \cap\ Test)}{cardinalité\ (Test)} \qquad (3.2)$$

- Le rappel : Cette mesure représente le taux des constituants corrects générés par l'analyse par rapport aux nombre de constituants générés par l'analyse de la phrase de référence.

$$Rappel = \frac{cardinalité\ (Référence\ \cap\ Test)}{cardinalité\ (Référence)} \qquad (3.3)$$

- La F-mesure : C'est une mesure qui combine la précision et le rappel. Elle constitue leur pondération et elle est nommée F-mesure (F-measure en anglais) ou F-score :

$$F\text{–}mesure = \frac{2\ \times\ (Précision\ \times\ Rappel)}{(Précision\ +\ Rappel)} \qquad (3.4)$$

Nous avons calculé les trois mesures pour chaque phrase du corpus de test. Nous avons ensuite calculé la moyenne des valeurs obtenues (Khoufi, Aloulou, & Hadrich Belguith, 2016). Les résultats obtenus sont présentées dans le tableau 3.5.

| **Précision** | **Rappel** | **F-mesure** |
|---|---|---|
| 83.59 % | 82.98% | 83.23% |

**Tableau 3.5 :** Résultats de l'évaluation

Dans la section suivante nous allons discuter les résultats obtenus par les deux méthodes d'analyse et nous présenterons ensuite la méthode d'analyse hybride.

# 3. Discussion des résultats des deux méthodes d'analyse

La démarche que nous suivons pour choisir une stratégie d'hybridation (numérique /symbolique) performante consiste à partir des résultats d'expérimentations concrètes d'une méthode d'analyse numérique et d'une méthode d'analyse symbolique. Pour cela nous avons étudié de plus près les résultats obtenus par notre méthode symbolique et les ceux obtenus par la méthode numérique de (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015) (développée au sein de notre équipe) afin d'identifier les faiblesses de l'analyse symbolique et voir comment ont peut les surmontées par l'application d'une analyse numérique.

L'étude des résultats obtenus par ces deux méthodes d'analyse syntaxique nous a permis de voir leurs points forts et leurs points faibles sur le plan pratique. Nous avons remarqué que leurs performances varient selon plusieurs critères tels que le niveau de profondeur de l'analyse ou la longueur des phrases analysées.

L'expérimentation et l'évaluation des deux méthodes nous a permis de faire les constatations suivantes :

Les résultats obtenus par l'analyse syntaxique numérique sont très encourageants par rapport aux résultats d'autres analyseurs de la langue arabes. Cependant, ces résultats sont en de ça de ceux obtenus par l'analyseur symbolique décri dans la section précédente sachant qu'ils ont été évalués sur le même corpus PATB comme le montre le tableau 3.6 :

| **Analyse** | **Précision** | **Rappel** | **F-mesure** |
|---|---|---|---|
| Symbolique (Khoufi, Aloulou, & Hadrich Belguith, 2016) | 83.59% | 82.98% | 83.23% |
| Numérique (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015) | 73,62% | 68,94% | 71,12% |

**Tableau 3.6 :** Tableau comparatif des résultats obtenus par notre analyser symbolique et l'analyse numérique

Nous avons aussi remarqué que l'évolution des résultats de l'analyse syntaxique numérique en fonction de la profondeur d'analyse indique une baisse significative de la performance entre les résultats du niveau 6 et ceux du niveau 5. Le niveau 6 étant le

premier niveau de l'arbre syntaxique qui indique les constituants principaux ou globaux de la phrase. Le niveau 5 étant le niveau supérieur au niveau 6 dans l'arbre et indique les constituants inclues dans les constituants du niveau 6. Le niveau 7 représente le niveau le plus bas de l'arbre (niveau morphologique) alors que le niveau 0 est le niveau le plus haut (niveau phrase). La figure 3.6 présente les résultats obtenus par la méthode d'analyse syntaxique numérique en fonction du niveau de profondeur de l'analyse.

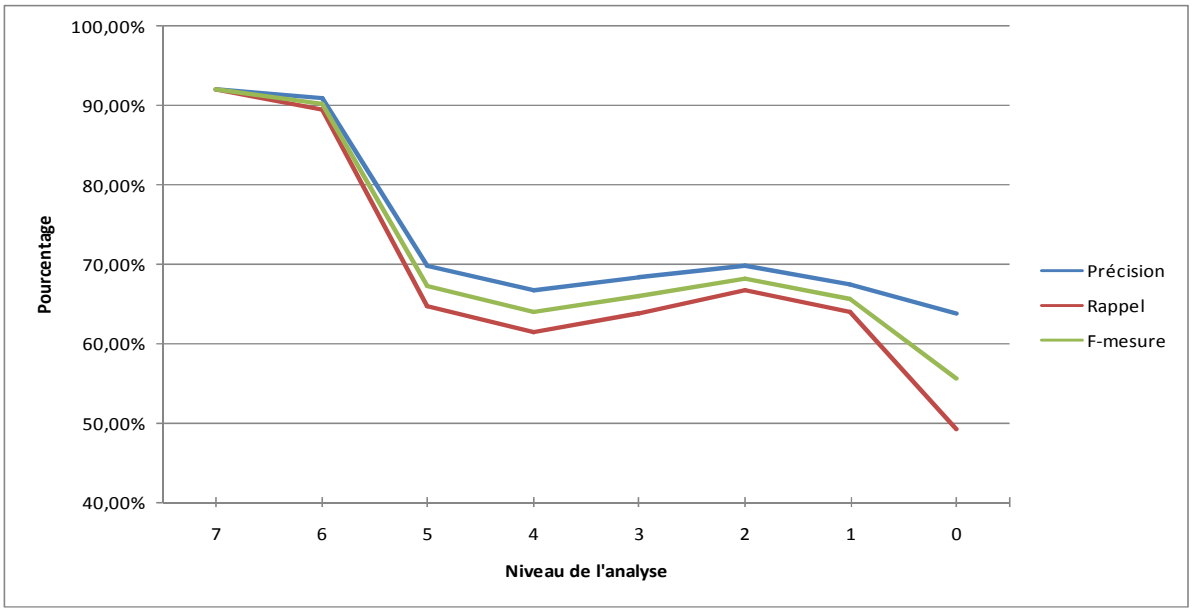



**Figure 3.6 :** Evolution des valeurs du rappel, précision et f-mesure en fonction du niveau d'analyse syntaxique de l'analyse numérique

En effet la valeur du f-mesure passe de 90,24% à 67,20 % soit une diminution de plus de 33% ce qui est conséquent. Ceci nous porte à croire que l'approche numérique donne de meilleurs résultats pour l'analyse syntaxique de surface plutôt que pour l'analyse syntaxique profonde.

De l'autre coté, notre méthode d'analyse symbolique donne des résultats intéressants mais nous remarquons une baisse des performances au fur et à mesure que la phrase augmente en longueur comme le montre la figure 3.7.

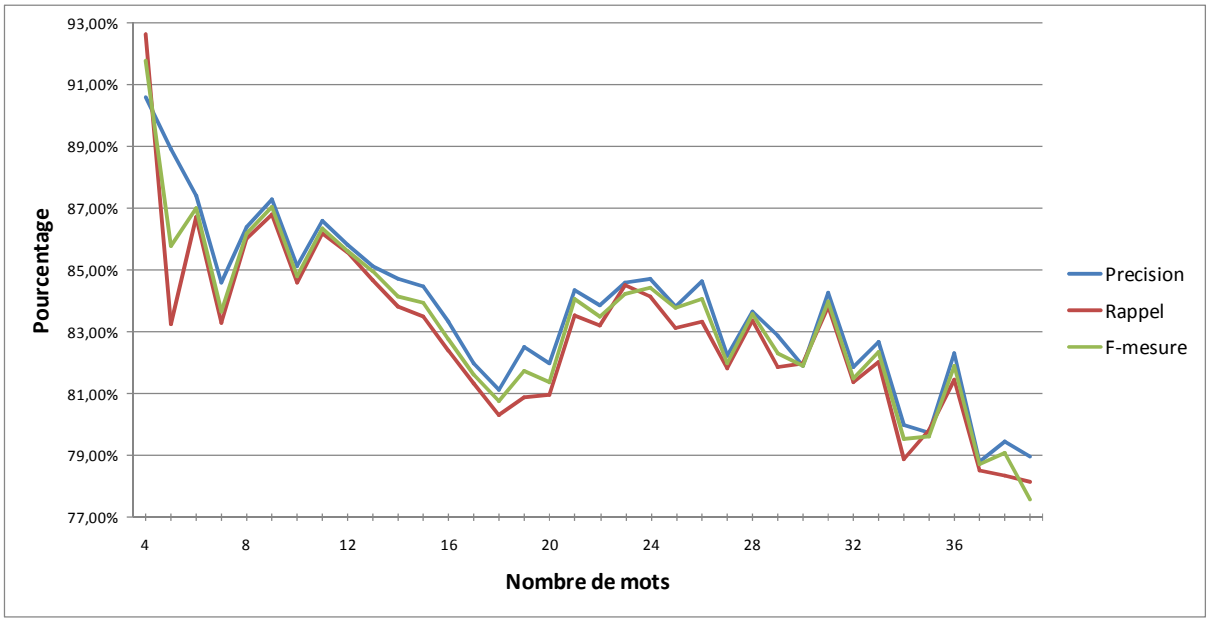



**Figure 3.7 :** Evolution des valeurs du rappel, précision et du f-mesure en fonction du nombre de mots dans les phrases de tests

En effet, nous remarquons que le f-score passe de 86,36 % pour les phrases de longueur 11 mots à 80,73% pour les phrases de longueur 18 mots ; soit une baisse d'environs 6%.

Nous avons remarqué que l'analyseur symbolique a du mal à analyser les phrases imbriquées. Ce genre de phrase génère un temps d'analyse trop long et peut même bloquer complètement l'analyseur. L'imbrication de phrases est un phénomène très présent dans l'arabe standard et génère des phrases de longueur pouvant aller jusqu'a 60 mots, voire même plus.

En plus, nous avons remarqué que plus la phrase est longue, plus le temps d'analyse augmente de façon exponentielle à cause de l'explosion combinatoire des possibilités d'analyse. En effet, nous avons calculé la durée moyenne d'analyse sur des phrases dont la longueur varie entre 4 et 39 mots. La figure 3.8 présente la courbe d'évolution du temps d'analyse moyen en fonction de la longueur de la phrase.

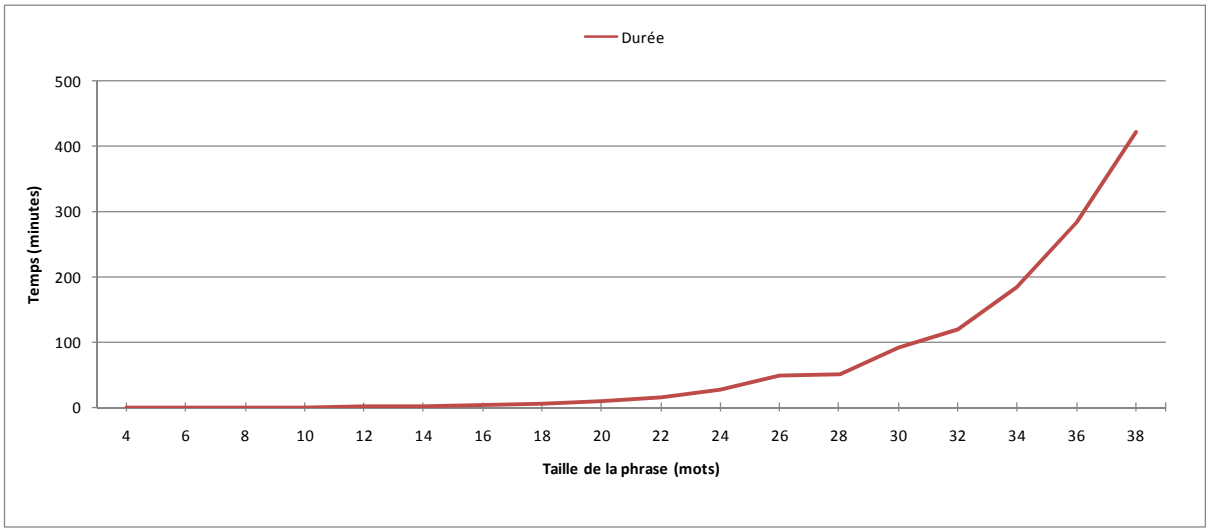



**Figure 3.8 :** Evolution du temps d'analyse en fonction de la longueur des phrases

Suite à ces différentes constatations, nous proposons dans ce qui suit notre méthode hybride pour l'analyse syntaxique de l'arabe par laquelle nous essayons d'améliorer les performances de notre méthode d'analyse syntaxique symbolique.

## 4. Méthode hybride proposée

D'après l'étude des résultats obtenus des deux méthodes d'analyse syntaxique, symbolique et numérique, nous proposons une méthode d'hybridation qui permet de palier certaines insuffisances de chaque analyseur et de profiter des avantages de chacun d'eux (Khoufi, Aloulou, & Hadrich Belguith, 2016).

Dans notre proposition, l'analyse syntaxique d'une phrase donnée est générée par une approche basée sur l'utilisation conjointe d'une analyse numérique et d'une analyse symbolique. L'analyseur symbolique a un rôle principal dans notre proposition. Le rôle de l'analyseur numérique est d'aider l'analyseur symbolique selon un scenario bien défini, déduit à partir des constatations faites dans la section précédente.

L'information clé pour cette hybridation est la longueur de la phrase en mots. En effet, selon cette information, nous allons choisir soit un traitement purement symbolique en utilisant la méthode symbolique, soit un traitement hybride qui partage la tâche d'analyse syntaxique entre les deux processus (numérique et symbolique). La figure 3.9 illustre l'enchaînement de notre méthode hybride.

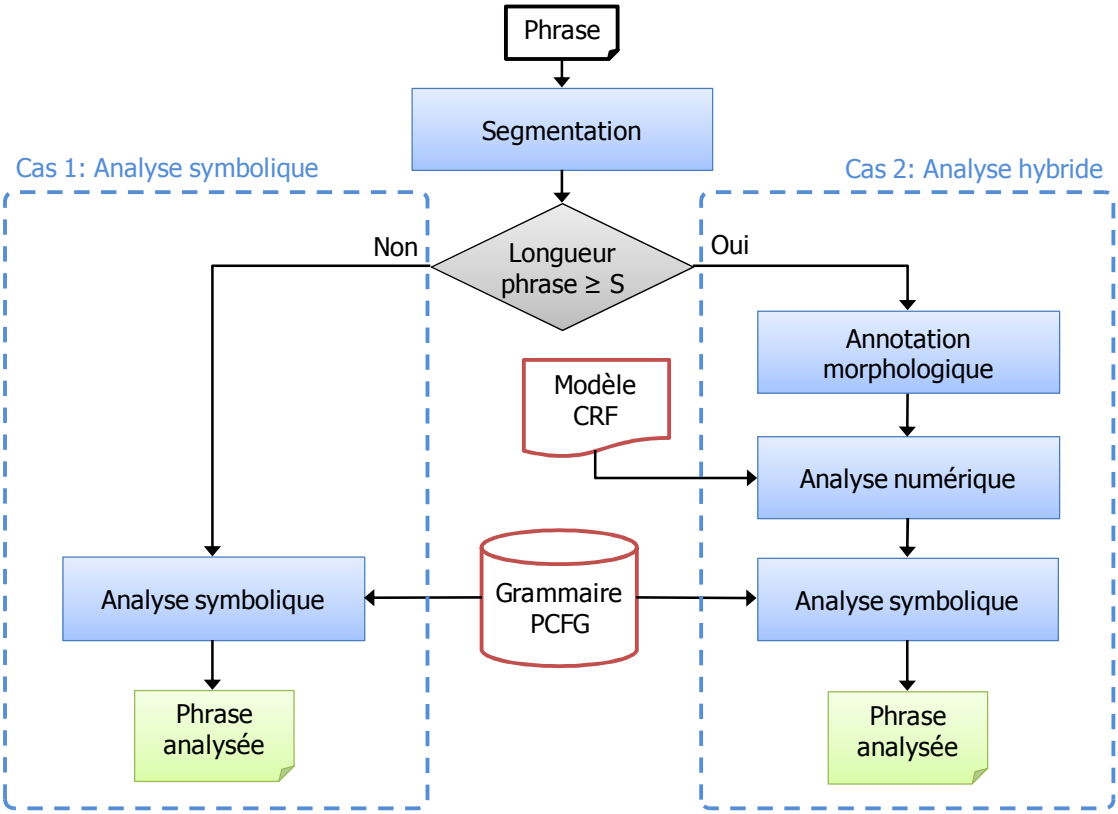



**Figure 3.9 :** Étapes de la méthode hybride proposée

## 4.1 La segmentation

Lors du traitement automatique de l'arabe, la segmentation du texte en unités lexicales parait un processus nécessaire. Cette phase est considérée comme une étape de préparation ou prétraitement pour plusieurs tâches du TALN.

Le principale but de la segmentation consiste à préciser les frontières des mots c'est-à-dire de localiser les unités lexicales qui vont être analysées dans la phase suivante tel que les noms, les verbes, les adverbes, les adjectifs, les déterminants, etc. Ainsi, la segmentation sert à détacher du mot les unités minimales qui ont une forme libre tel que les clitiques ainsi que les préfixes et les suffixes.

Ces unités, qui paraissent attachées aux mots, sont difficiles à identifier et à écarter du mot agglutiné à cause de leurs formes graphiques changeables. L'exemple suivant montre le résultat de la segmentation du mot « أسيعطونه ؟ » qui peut être traduite en français par la phrase « Est-ce qu'ils vont le donner ? »

| Enclitique | Suffixe | Corps schématique | Préfixe | Proclitique |
|---|---|---|---|---|
| ه | ون | يعط | س | أ |
| Pronom suffixe | Suffixe verbale exprimant le pluriel | Dérivé de la graphie (ي ع ط) | Préfixe verbale expriment l'aspect inaccompli | Conjonction d'interrogation |

**Tableau 3.7 :** Exemple illustratif de la structure segmentée du mot « أسيعطونه »

## 4.2 Cas 1 : Analyse symbolique

L'analyse syntaxique linguistique basée sur l'utilisation de grammaire symbolique est connue pour donner des structures syntaxiques détaillées de bonne qualité pour des petites phrases et cela est valable aussi pour l'arabe standard (Aloulou, 2005) (Al-Taani, Msallam, & Wedian, 2012). L'analyseur que nous avons développé ne déroge pas à cette règle en offrant des résultats intéressants.

Le choix de ce premier cas de figure a été fait à partir de l'étude des résultats de l'analyseur symbolique. Nous avons remarqué que celui ci donne des résultats d'analyse syntaxique que nous jugeons satisfaisant pour des phrases dont la longueur ne dépasse pas un seuil « S » que nous définirons durant nos expérimentations.

Dans ce cas de figure, nous avons choisi de faire une analyse syntaxique purement linguistique basée sur une grammaire de la langue. Nous utilisons pour cela la grammaire PCFG arabe qui a été induite à partir du corpus PATB ainsi que l'algorithme d'analyse de Viterbi. L'analyse se déroule alors selon les étapes décrites dans la section 2.2 de ce chapitre.

## 4.3 Cas 2 : Analyse hybride

Le deuxième cas de figure met en relief notre méthode d'hybridation symbolique/numérique pour analyser syntaxiquement une phrase. En effet au delà d'une certaine taille de la phrase, nous mettons en œuvre un traitement basé sur l'apprentissage automatique suivi d'un traitement basé sur une grammaire symbolique. Cet aspect d'hybridation vise à profiter des points forts de ces deux traitements.

Avant de commencer le processus d'analyse syntaxique, la phrase doit être étiquetée morphologiquement car cette information constitue la base des traitements suivants.

Ensuite, un premier module effectue une analyse de surface en utilisant un modèle statistique obtenu par apprentissage automatique afin d'identifier les frontières des constituants syntaxiques principaux de la phrase (Khoufi, Aloulou, & Hadrich Belguith, 2014). Ce traitement a pour but de segmenter la phrase en constituants plus facile et plus rapide à traiter.

Par la suite le module symbolique prend le relai pour continuer le processus d'analyse. Ce module reçoit comme input les constituants syntaxiques identifiés par le premier module. Ensuite il va appliquer une analyse linguistique avec la grammaire PCFG arabe et l'algorithme de Viterbi pour compléter l'analyse syntaxique de la phrase. La structure syntaxique globale de la phrase est formée par le regroupement de toutes les structures obtenues.

Dans ce qui suit, nous commençons par la description du processus d'apprentissage automatique nécessaire pour l'obtention du modèle statistique ensuite nous présentons le processus d'analyse.

### 4.3.1 Apprentissage du modèle CRF

Au cours de cette étape, nous avons eu recours aux techniques d'apprentissage automatique supervisé en utilisant l'algorithme des champs conditionnels aléatoires connu sous l'acronyme CRF (Conditional Random Fields). Cet algorithme a largement fait ses preuves dans plusieurs domaines du TALN telles que la reconnaissance d'entité nommée (McCallum & Li, 2003), l'analyse syntaxique de surface pour l'anglais (Sha & Pereira, 2003) et l'analyse syntaxique profonde (Tsuruoka, Tsujii, & Ananiadou, 2009). CRF a été aussi testé pour le traitement automatique de l'arabe et a donné de bon résultats pour l'annotation de dialogue (Ben Dbabis, Reguii, Ghorbel, & Hadrich Belguith, 2016) et même pour l'annotation de dialogue en dialecte tunisien (Graja, 2016).

L'objectif à ce niveau est d'apprendre automatiquement un modèle statistique qui sera utilisé pour segmenter la phrase en constituants syntaxiques principaux. La figure 3.10 présente le processus d'apprentissage pour la création et la validation du modèle CRF.

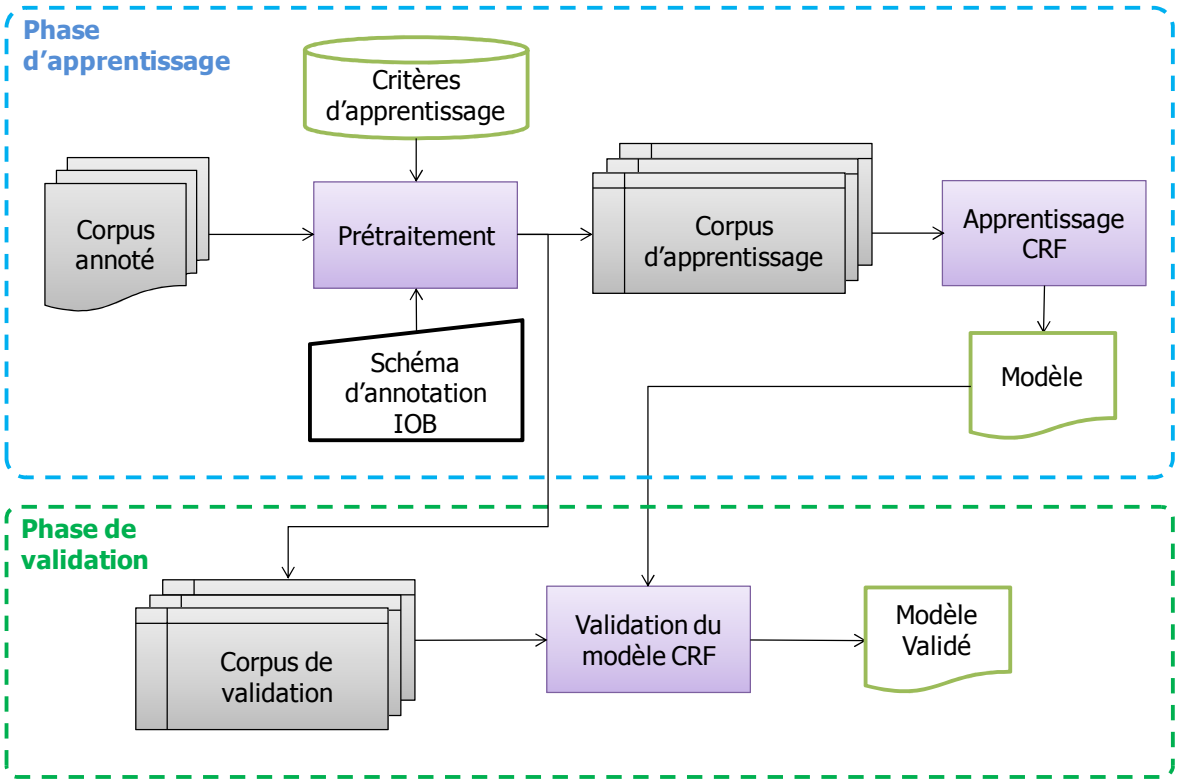



**Figure 3.10:** Principe de l'apprentissage

Dans ce qui suit nous commençons par la définition des modèles CRFs ensuite nous présentons les détails de notre processus d'apprentissage.

#### 4.3.1.1 Définition des CRFs

Les champs conditionnels aléatoires (Conditional Random Fields ou CRFs) (Lafferty, McCallum, & Pereira, 2001) sont des modèles graphiques non dirigés ayant pour objectif de définir une distribution de probabilités sur les annotations $y$ étant donné une observation $x$. Ils sont définis comme suit.

Soit G = (V,E) un graphe non dirigé (appelé graphe d'indépendances) ou V est l'ensemble des nœuds (vertices), E est l'ensemble des arcs (edges), et X et Y deux champs aléatoires décrivant respectivement l'observation et son annotation, de sorte que pour chaque nœud v ∈ V il existe une variable aléatoire $Y_v$ dans Y. On dit que (X, Y) est un champ conditionnel aléatoire si chaque variable aléatoire $Y_v$ respecte la propriété de Markov suivante :

$$\forall v, p(Y_v|X, \{Y_w, w \neq v\}) = p(Y_v|X, \{Y_w, (V, W) \in E\}) \qquad (3.5)$$

C'est-à-dire chaque variable aléatoire $Y_v$ dépend uniquement de X et de ses voisins dans le graphe d'indépendances.

Pour les CRFs, Lafferty, McCallum et Pereira (Lafferty, McCallum, & Pereira, 2001) ont proposé de définir la forme de ces fonctions de potentiel comme l'exponentielle d'une somme pondérée de fonctions $f_k$ appelées features, les $\lambda_k$ étant les poids associés à chacune de ces features :

$$\psi_c(y_c, x) = \exp\Big(\sum_k \lambda_k f_k(y_c, x, c)\Big) \tag{3.6}$$

Les features sont des fonctions à valeurs réelles. C'est à travers elles que toutes les connaissances du domaine sont intégrées dans le modèle.

Dans la plupart des cas, les features sont des fonctions binaires valant 1 si un phénomène est observé, 0 sinon. Ces features prennent en paramètres les valeurs prises par les variables aléatoires de la séquence sur laquelle elles s'appliquent ($y_c$), ainsi que l'ensemble de l'observation x. Par conséquent, la valeur prise par une variable aléatoire peut dépendre de toute l'observation x.

Par exemple, dans le cas de l'annotation d'une séquence, le choix de l'étiquette associée au dernier élément de la séquence peut être lié à la valeur du premier élément de cette séquence. A ces features sont associés des poids $\Lambda=\{\lambda_k\}$. Ces poids sont les paramètres du modèle. Ils permettent d'attacher plus ou moins d'importance à certaines features, ou même d'indiquer que le phénomène caractérisé par une feature ne doit pas se produire (si le poids est négatif).

Un CRF est donc défini par un graphe d'indépendances G et un ensemble de features $f_k$, auxquelles sont associés des poids $\lambda_k$. La probabilité conditionnelle d'une annotation connaissant une observation, comme défini par un CRF, s'exprime alors par :

$$p(y|x) = \frac{1}{Z(X=x)} \exp\Big(\sum_{c\in\mathcal{C}} \sum_k \lambda_k f_k(y_c, x, c)\Big) \tag{3.7}$$

Ou Z(x) se réécrit :

$$Z(x) = \sum_y \exp\Big(\sum_{c\in\mathcal{C}} \sum_k \lambda_k f_k(y_c, x, c)\Big) \tag{3.8}$$

#### 4.3.1.2 Prétraitement du corpus d'apprentissage

Cette étape vise à préparer le corpus annoté pour la phase d'apprentissage en indiquant l'information à prendre en compte lors de ce processus.

Le corpus PATB est très riche en informations. Il utilise un jeu d'étiquettes très détaillé composé entre autres de 498 étiquettes morphologiques, 22 catégories syntaxiques et 20 autres annotations qui décrivent des relations sémantiques de la phrase.

Pour notre processus d'apprentissage, nous n'avons besoin que des étiquettes syntaxiques ainsi que de l'étiquette du début qui représente le début de la phrase « S » se qui ramène le nombre d'étiquettes utilisées à 23 étiquettes.

En plus des annotations du PATB, nous utilisons le schéma d'annotation IOB (Inside, Outside, Begin), introduit par Ramshow et Marcus (Ramshow and Marcus, 1995), qui permet de définir les limites des constituants de la phrase pour l'apprentissage automatique. Ce schéma propose que chaque mot de la phrase soit annoté avec une des annotations suivantes :

- B (Begin) : Premier mot d'un constituant.
- I (Inside): Mot d'un constituant différent du premier.
- O (Outside) : Mot n'appartenant à aucun constituant.

Chaque annotation du jeu d'annotation utilisé devra être décorée par une annotation du schéma IOB. Prenons par exemple l'annotation NP, en appliquant le schéma IOB, cette annotation devient deux : B-NP et I-NP. Le tableau 3.8 présente une phrase annotée en suivant le schéma IOB.

| **Phrase** | . | 2016 | العربية | الثقافة | عاصمة | صفاقس |
|---|---|---|---|---|---|---|
| **Translitération** | . | 2016 | AlErbyp | AlvqAfp | EASmp | SfAqs |
| **Traduction** | . | 2016 | Arabe | Culture | Capitale | Sfax |
| **Annotation IOB** | O | I-NP | I-NP | I-NP | B-NP | B-NP |

**Tableau 3.8 :** Exemple de phrase annotée suivant le schéma IOB.

#### 4.3.1.3 Les critères utilisés

La phase d'apprentissage supervisé nécessite l'utilisation de critères d'apprentissage pour spécifier les informations à utiliser dans le corpus d'apprentissage ce qu'on appelle critères ou « features » en anglais. Ces derniers reflètent les connaissances du domaine d'application nécessaires pour créer un modèle d'apprentissage automatique. Dans le cadre de ce travail, elles reflètent les connaissances syntaxiques de la langue arabe. Nous avons utilisé les mots eux même et leurs annotations morphologiques. Ce choix est le fruit d'une recherche bibliographique sur les travaux utilisant l'apprentissage automatique supervisée pour l'étiquetage morphosyntaxique. Nous avons remarqué que plusieurs travaux tel que (Taku & Matsumoto, 2001) pour l'anglais ou (Ben Fraj, Ben Othmane Zribi, & Ben ahmed, 2009) pour l'arabe ont utilisé une fenêtre de (-2+2) pour la préparation du corpus d'apprentissage et ont donnée de bons résultats. Nous avons donc adopté cette fenêtre avec les traits morphologiques et les mots comme critères pour l'apprentissage. Le tableau 3.9 présente la liste des critères utilisés.

| **Critère** | **Signification** |
|---|---|
| w[t-2] | Mot au voisinage droit à la position t-2 |
| w[t-1] | Mot au voisinage droit à la position t-1 |
| w[t] | Mot en cour de traitement |
| w[t+1] | Mot au voisinage gauche à la position t+1 |
| w[t+2] | Mot au voisinage gauche à la position t+2 |
| pos[t-2] | Catégorie grammaticale du mot au voisinage droit à la position t-2 |
| pos[t-1] | Catégorie grammaticale du mot au voisinage droit à la position t-1 |
| pos[t] | Catégorie grammaticale du mot en cour de traitement |
| pos[t+1] | Catégorie grammaticale du mot au voisinage gauche à la position t+1 |
| pos[t+2] | Catégorie grammaticale du mot au voisinage gauche à la position t+2 |

**Tableau 3.9 :** Liste des critères utilisés dans la phase d'apprentissage

En plus des critères d'apprentissage nous avons utilisé le modèle N-grammes. En effet nous avons construit des bigrammes et des trigrammes associés aux mots et aux catégories grammaticales. Cette information sur le contexte contribue à la précision du

modèle appris à partir du corpus d'apprentissage. Le tableau 13 expose les bigrammes et trigrammes utilisés.

| | |
|---|---|
| **Bigrammes sur les mots** | w[t-1]\|w[t]<br>w[t]\|w[t+1] |
| **Bigrammes sur les catégories grammaticales** | pos[t-2]\|pos[t-1]<br>pos[t-1]\|pos[t]<br>pos[t]\|pos[t+1]<br>pos[t+1]\|pos[t+2] |
| **Trigrammes sur les catégories grammaticales** | pos[t-2]\|pos[t-1]\|pos[t]<br>pos[t-1]\|pos[t]\|pos[t+1]<br>pos[t]\|pos[t+1]\|pos[t+2] |

**Tableau 3.10 :** liste des bigrammes et trigrammes utilisés

Les critères et les N-grammes vont être utilisés pour préparer le corpus à la phase d'apprentissage. Une fois l'apprentissage réalisé, nous obtenons le modèle CRF qui nous permet d'identifier les constituants syntaxiques principaux de la phrase. Il est à noter que la phase d'apprentissage n'est réalisée qu'une seule fois.

### 4.3.2 Analyse numérique

C'est au cours de cette étape qu'on va mettre en œuvre le modèle statistique CRF appris. En effet celui-ci est utilisé pour réaliser l'analyse numérique. Cette étape reçoit comme entrée la phrase préalablement segmentée et étiquetée morphologiquement ensuite applique le modèle sur cette phrase pour identifier les frontières des constituants syntaxiques principaux. La figure suivante illustre le processus d'analyse de surface.

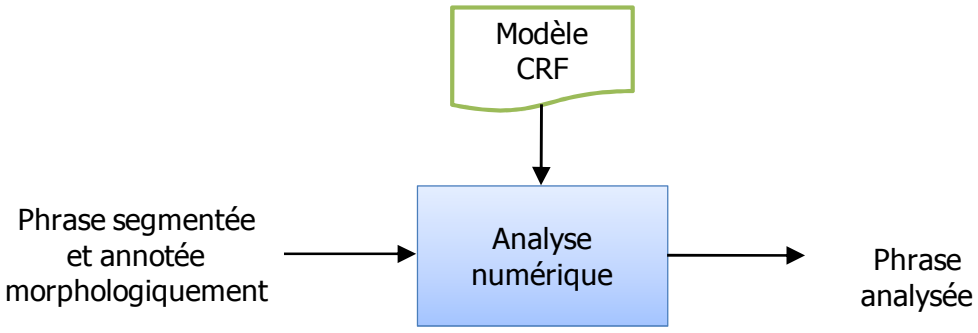



**Figure 3.11 :** Schéma de l'analyse syntaxique numérique

Nous obtenons à la fin de cette analyse numérique une phrase dont les constituants syntaxiques principaux de la phrase ont été identifiés.

Afin de percevoir l'objectif de cette phase, prenons l'exemple de la phrase suivante :

**أكثر من خمسة ملايين سائح أجنبي زاروا البلاد التونسية في السنة الماضية.**

Translitération Buckwalter : Okvr mn xmsp mlAyyn sA}H Ojnby zArwA AlblAd Altwnsyp fy Alsnp AlmADyp**.**

*(Plus de cinq millions de touristes étrangers ont visité la Tunisie l'année dernière.)*

Après l'application de notre analyse numérique, le modèle va annoter chaque mot de la phrase avec l'étiquetage IOB en utilisant le modèle CRF. Ainsi nous aurons les frontières des constituants syntaxiques principaux :

[أكثر من خمسة ملايين سائح أجنبي] [زاروا البلاد التونسية في السنة الماضية]

Constituant 1 : NP Constituant 2 : VP

*(Plus de cinq millions de touristes étrangers) (ont visité la Tunisie l'année dernière)*

### 4.3.3 Analyse symbolique

A ce stade de l'analyse, nous avons identifié les constituants syntaxiques principaux de la phrase. L'objectif de cette étape est de continuer le processus d'analyse en générant les structures syntaxiques internes de chaque constituant principal. Pour ce faire, nous utilisons notre méthode à base de règles symboliques. Cette méthode reçoit donc en entrée les constituants syntaxiques identifiés et génère les structures syntaxiques internes de chaque constituant. Par la suite nous procédons au regroupement de toutes les informations syntaxiques produites par les deux analyses afin d'obtenir la structure syntaxique complète de la phrase en sortie. La figure 3.12 illustre le processus d'analyse symbolique.

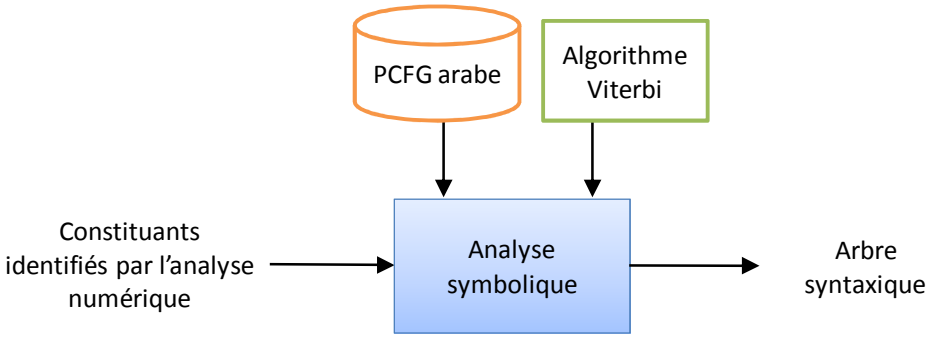



**Figure 3.12 :** Schéma de l'analyse syntaxique symbolique

Continuons avec l'exemple que nous avons pris pendant l'étape d'analyse numérique pour mieux appréhender cette étape. Nous analysons donc chaque constituant identifié précédemment séparément afin d'obtenir sa structure syntaxique. Par la suite nous regroupons les deux analyses afin d'obtenir la structure syntaxique complète de la phrase. La figure 3.13 présente le résultat final de l'analyse selon la méthode hybride.

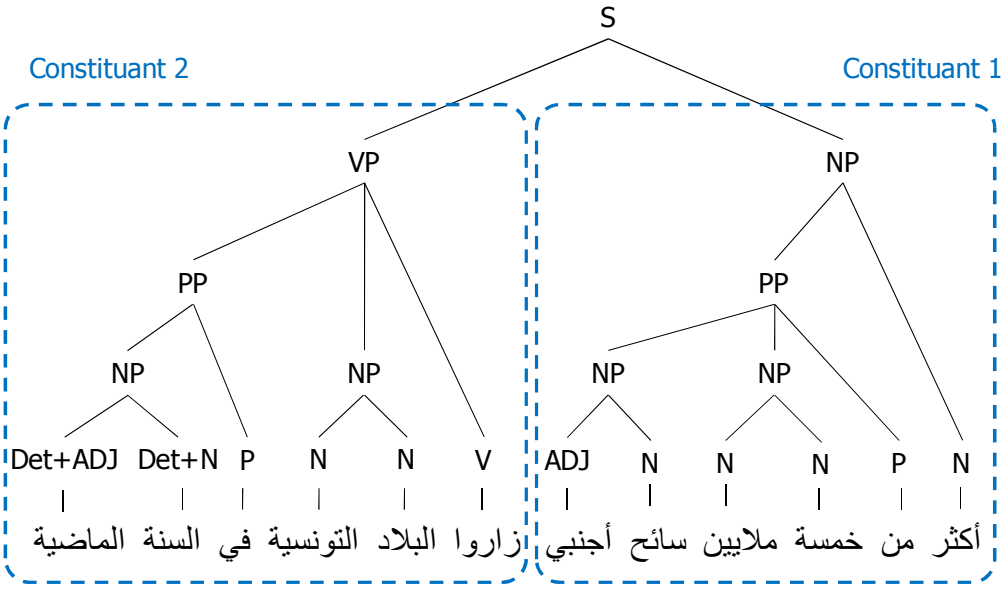



**Figure 3.13 :** Résultat final après l'application de l'analyse syntaxique hybride

## 5. Conclusion

Nous avons commencé ce chapitre par la présentation de notre méthode d'analyse syntaxique symbolique basé sur une grammaire PCFG arabe. Cette grammaire a été construite par induction à partir du corpus annoté PATB. Le processus d'induction

que nous avons proposé permet de construire rapidement une grammaire arabe sans avoir des connaissances approfondies de la langue.

Ensuite, nous avons présenté les différentes étapes de notre méthode hybride d'analyse syntaxique de l'arabe. Nous avons ainsi proposé deux cas de figure qui proposent soit une analyse purement symbolique soit une analyse numérique/symbolique.

Au niveau du premier cas de figure, nous avons utilisé notre méthode symbolique basée sur la grammaire PCFG arabe présenté dans la section 2. Au niveau du deuxième cas de figure, nous avons proposé trois étapes successives à savoir, une étape d'annotation morphologique, une étape d'analyse syntaxique numérique et une étape d'analyse syntaxique symbolique.

Le chapitre suivant détaille l'évaluation des différentes méthodes proposées, à savoir la méthode numérique, la méthode symbolique et la méthode hybride Nous allons présenter les mesures d'évaluation utilisées, les prototypes logiciels développés et les résultats obtenus. Nous mettrons l'accent sur l'apport de notre méthode hybride.

# Chapitre 4

## Réalisation et évaluation des méthodes proposées

# 1. Introduction

Dans ce chapitre, nous allons présenter l'évaluation des trois méthodes proposées dans le cadre cette thèse, à savoir la méthode d'analyse numérique, la méthode d'analyse symbolique et la méthode hybride. Pour juger de la qualité des analyses produites, nous utilisons des métriques standards que nous venons de présenter dans le chapitre 3 à savoir, le rappel, la précision et la f-mesure.

Nous commençons ce chapitre par la présentation du corpus PATB utilisé au cours de la conception et de l'évaluation de notre proposition. Nous présentons aussi les métriques d'évaluation utilisés. Par la suite, nous présentons le processus et les résultats de l'évaluation de la méthode numérique. Nous enchainons avec la présentation du prototype développé pour l'évaluation de la méthode symbolique. Ensuite nous présentons les détails de l'évaluation de notre méthode hybride. Nous terminons par une discussion des résultats obtenus par la méthode hybride.

# 2. Présentation du corpus PATB

Notre approche hybride utilise un corpus annoté tout au long de ses différentes étapes. Cette ressource représente un pilier sur lequel se base tout notre travail. Nous avons donc choisi de commencer par la présentation du corpus utilisé, le Penn Arabic Treebank (PATB).

Ce corpus a été conçu par LDC (Linguistic Data Consortium) qui est basé à l'université de Pennsylvanie aux USA (Maamouri M. , Bies, Buckwalter, & Mekki, 2004). LDC est un consortium ouvert des universités, des entreprises et des laboratoires de recherche gouvernementaux. Il crée, collecte et distribue les bases de données de textes et de discours, lexiques et autres ressources à des fins de recherche et de développement linguistique.

Le corpus PATB consiste en des données extraites de sources linguistiques écrites en langue arabe standard et moderne. Il se compose de 599 textes extraits du journal libanais « An Nahar ». Les textes sont non voyellés ou partiellement voyellés et segmentés. Chaque mot est annoté avec plusieurs informations comme le trait morphologique, la partie du discours, la traduction en anglais et la translitération Buckwalter. Il contient aussi les arbres syntaxiques de chaque phrase. L'annotation

morphologique du PATB a été faite à l'aide de l'analyseur morphologique SAMA. En effet, l'annotation la plus adéquate est sélectionnée parmi toutes les annotations possibles générées par SAMA (Maamouri, Bies, & Kulick, 2008).

Dans nos travaux, nous utilisons la version 3.2. Cette version est composée de 402,291 mots après séparation des clitiques. Cette version est fournie sous différents formats pour étendre son utilisation à différents besoins de recherche. Ces formats que nous citons sont au nombre de cinq :

- Le format « SGM »
- Le format « POS »
- Le format « XML »
- Le format « penntree »

Dans ce qui suit, nous présentons en détail ces différents formats.

## 2.1 Le format SGM

Le format SGM représente les documents sources utilisée pour construire le PATB. Le fichier contient seulement le nom de fichier et le texte en format UTF-8 sans aucune modification du contenu. Le texte a été segmenté en paragraphes puis en phrases en utilisant des balises. Un identifiant a été assigné à chaque phrase. La figure 4.1 présente un extrait d'un fichier au format SGM.

```
<DOC docid="ANN20020115.0003">
<p>
<seg id=2> الوقت الذي تركز" افغانستان أمس ضمن الحملة التي تشنها على مقاتلي تنظيم قصفت طائرات امبركية مجمعات كهوف في شرق
<seg id=3> ار على مسافة 30 تتخذ اسلام اباد مقرا لها انه بعد قصف دون توقف الاحد وافادت "وكالة الانباء الاسلامية" الافغانية التي
<seg id=4> وقالت: "لم يهدأ القصف طوال الساعات الـ 48 الاخيرة ". </seg>
</p>
<p>
<seg id=5> ته امس وأكد أنه في أحسن حال بعدما أغمي عليه فترة وجيزة مساء الاحد نتيجة سد قطعة من الكعك المملح "بريتزل" حلقه
<seg id=6> ررة على ثلاث ولايات في الغرب الاوسط الاميركي، بينما أكد طبيبه ان هذا العارض ليس خطيراً وأنه لم يصف للرئيس أي دواء
<seg id=7> وطمأن بوش (55 سنة) الصحافيين قبيل مغادرته البيت الابيض إلى انه يشعر بانه في حال "رائعة" وصحة "جيدة جداً ". </seg
<seg id=8> وأضاف مبتسماً: </seg>
</p>
<p>
<seg id=9> د والصين على كفاح مشترك ضد الارهاب، وأكدتا ان العلاقات بينهما لا تعتمد على طريقة نظر كل من الدولتين الى باكستان
<seg id=10> ووقعتا عدداً من الاتفاقات في مجال التعاون السياحي. </seg>
<seg id=11> لمسؤول صيني خلال مأدبة تكريم لرئيس الوزراء الصيني زو رونجي الذي وقال رئيس الوزراء الهندي اتال بيهاري فاجبابي
</p>
<p>
<seg id=12> ي في "كتائب شهداء الاقصى" التابعة لحركة "فتح" محمد رائد الكرمي (28 سنة) بتفجير عبوة ناسفة لدى مروره في طولكرم
<seg id=13> وردت هذه المنظمة فوراً باطلاق النار على جنود اسرائيليين في نابلس فقتل احدهم واصيب اثنان. </seg>
<seg id=14> القدس الشرقية لم تثنها الاحتجاجات الدولية عن مواصلة هدم المنازل ومضت اسرائيل في سياسة الاغتيالات في الوقت الذي
<seg id=15> وافاد مسؤولون </seg>
</p>
```

**Figure 4.1 :** Extrait d'un fichier au format « SGM »

## 2.2 Le format POS

Ce format permet l'affichage de plusieurs informations, comme la translitération, la voyéllation et la traduction, décrivant chaque mot source sous forme de champs avant et après la séparation des mots agglutinés. En effet, chaque mot source du texte est définit par un ensemble d'attributs décrivant ses diverses propriétés morphologiques que nous présentons ci-dessous :

- INPUT STRING : Le mot en entrée qui appartient au corpus PATB.
- IS_TRANS : La translitération du mot selon l'analyseur morphologique SAMA 3.1.
- COMMENT : Commentaire concernant le mot en entrée
- INDEX : Numéro du paragraphe et numéro du mot dans ce même paragraphe.
- OFFSETS : l'index de début et de fin du mot indiquée comme INPUT STRING.
- TOKENS : les index des mots selon la repartirions « before » et « after ».
- STATUS : Un statu indiqué par l'analyseur SAMA.

- LEMMA : Le Lemme associé au mot utilisé par SAMA.
- UNSPLITVOC : La translitération voyellé du mot en entrée.
- POS : Catégorie grammaticale du mot en entrée.
- VOC : La voyellation du mot.
- GLOSS : La traduction anglaise du mot.

Le format POS a deux sous catégories qui sont « before » et « after » qui comprennent presque les mêmes informations mais ont une utilisation différente. Le format « before » (figure 4.2) est utilise avec l'analyseur morphologique SAMA 3.1 tandis que le format « after »(figure 4.3) est utilise pour l'annotation graphique dans le format XML et Tree.

```
INPUT STRING: الحكومة
    IS_TRANS: AlHkwmp
       INDEX: P1W1
     OFFSETS: 0-8
      TOKENS: P1W1-P1W1
      STATUS: 1
       LEMMA: [Hukuwmap_1]
  UNSPLITVOC: (AlHukuwmapu)
         POS: DET+NOUN+NSUFF_FEM_SG+CASE_DEF_NOM
         VOC: Al+Hukuwm+ap+u
       GLOSS: the + government/administration + [fem.sg.] + [def.nom.]

INPUT STRING: الاردنية
    IS_TRANS: AlArdnyp
       INDEX: P1W2
     OFFSETS: 8-17
      TOKENS: P1W2-P1W2
      STATUS: 1
       LEMMA: [>urodun~iy~_1]
  UNSPLITVOC: (Al>urodun~iy~apu)
         POS: DET+ADJ+NSUFF_FEM_SG+CASE_DEF_NOM
         VOC: Al+>urodun~iy~+ap+u
       GLOSS: the + Jordanian + [fem.sg.] + [def.nom.]
```

**Figure 4.2 :** Extrait d'un fichier au format « before »

```
INPUT STRING: الحكومة
    IS_TRANS: AlHkwmp
     COMMENT: []
       INDEX: P1W1
     OFFSETS: 0,8
 UNVOCALIZED: AlHkwmp
   VOCALIZED: Al+Hukuwm+ap+u
         POS: DET+NOUN+NSUFF_FEM_SG+CASE_DEF_NOM
       GLOSS: the + government/administration + [fem.sg.] + [def.nom.]

INPUT STRING: الاردنية
    IS_TRANS: AlArdnyp
     COMMENT: []
       INDEX: P1W2
     OFFSETS: 8,17
 UNVOCALIZED: Al>rdnyp
   VOCALIZED: Al+>urodun~iy~+ap+u
         POS: DET+ADJ+NSUFF_FEM_SG+CASE_DEF_NOM
       GLOSS: the + Jordanian + [fem.sg.] + [def.nom.]

INPUT STRING: الجديدة
    IS_TRANS: Aljdydp
     COMMENT: []
       INDEX: P1W3
```

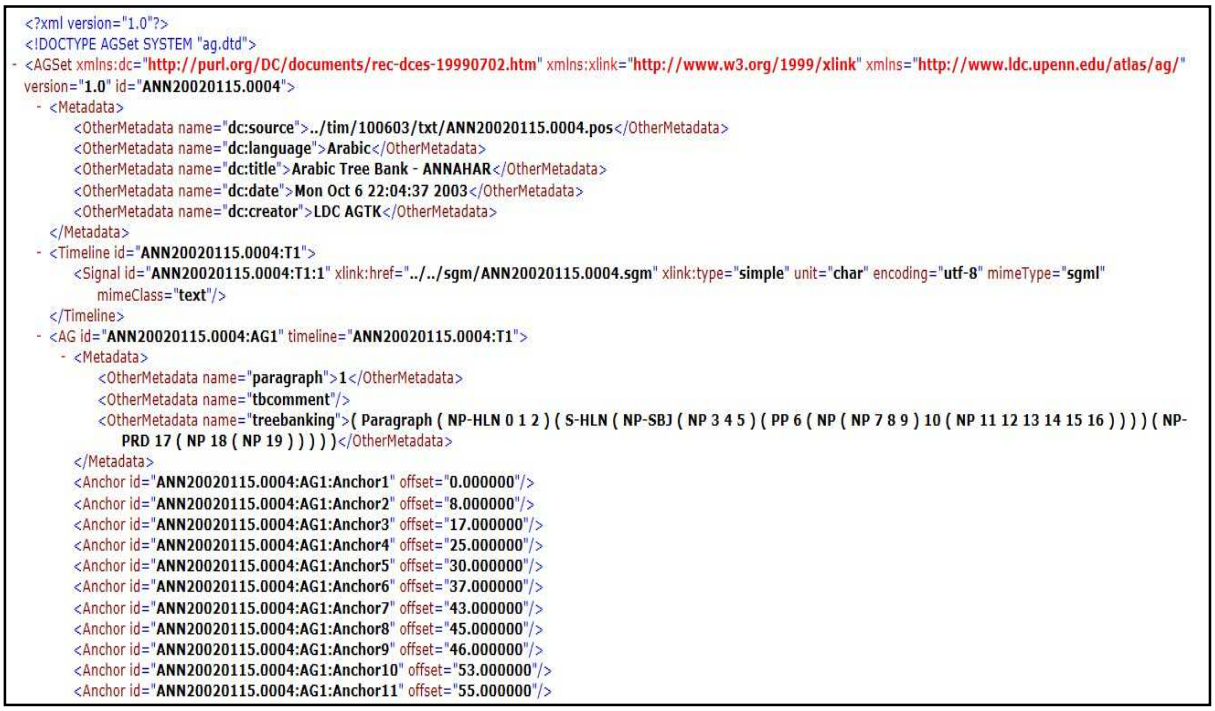
**Figure 4.3 :** Extrait d'un fichier au format « after »

## 2.3 Le format XML

Les fichiers XML contiennent une annotation graphique des arbres syntaxique complets de chaque paragraphe. Ce format suit une DTD nommée « ag.dtd » qui définit un élément racine AGSet et des éléments que nous présentons ci-dessous :

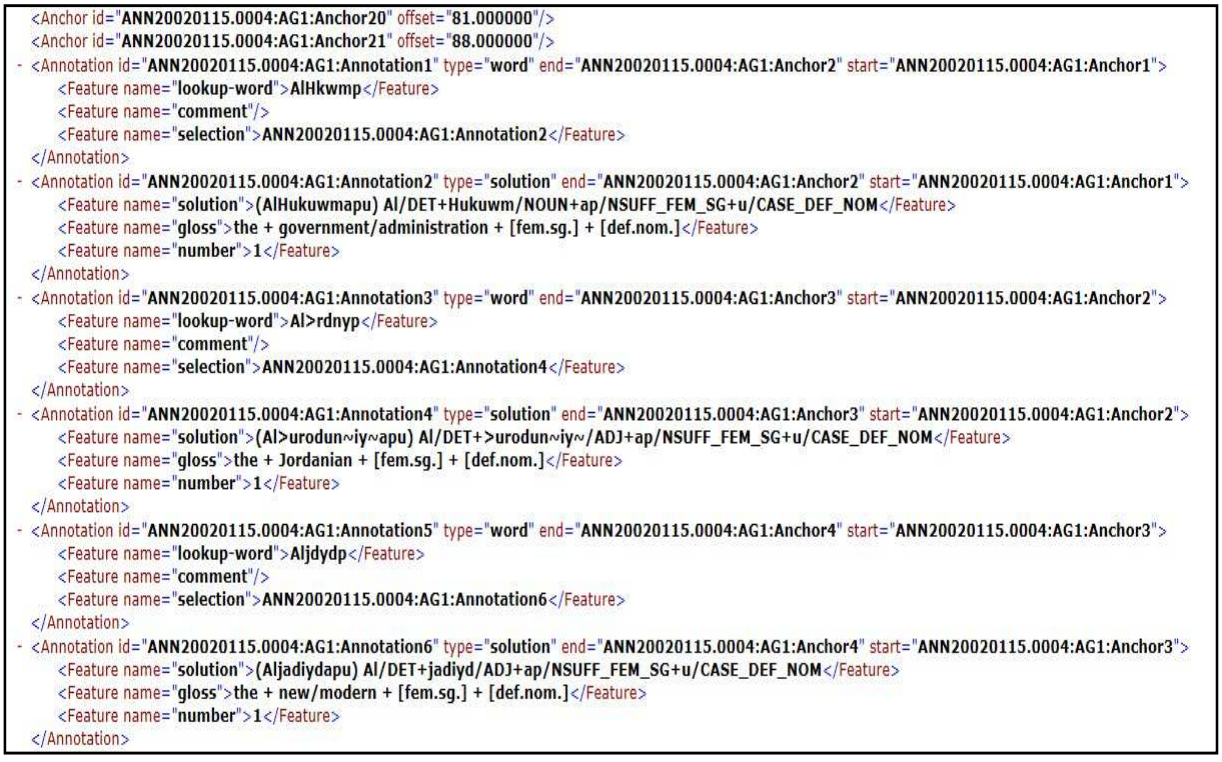
- Metadata : indique les métadonnées du fichier XML
- Timeline : donne un identifiant pour chaque arbre décrit dans le fichier
- AG : comprend l'annotation détailles de l'arbre analysé a travers les trois éléments suivants :
  - Metadata : cette élément donne le numéro du paragraphe, le commentaire, s'il existe et l'arbre syntaxique du paragraphe concerné.
  - Anchor : donne la positon du début de chaque mot du paragraphe
  - Annotation: cet élément associe à chaque mot sa translitération, son commentaire, son annotation morphologique et sa traduction anglaise.

Les figures 4.4 et 4.5 présentent un exemple d'un fichier XML montrant les différents éléments que nous venons de présenter.



```
<?xml version="1.0"?>
<!DOCTYPE AGSet SYSTEM "ag.dtd">
- <AGSet xmlns:dc="http://purl.org/DC/documents/rec-dces-19990702.htm" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.ldc.upenn.edu/atlas/ag/"
version="1.0" id="ANN20020115.0004">
  - <Metadata>
      <OtherMetadata name="dc:source">../tim/100603/txt/ANN20020115.0004.pos</OtherMetadata>
      <OtherMetadata name="dc:language">Arabic</OtherMetadata>
      <OtherMetadata name="dc:title">Arabic Tree Bank - ANNAHAR</OtherMetadata>
      <OtherMetadata name="dc:date">Mon Oct 6 22:04:37 2003</OtherMetadata>
      <OtherMetadata name="dc:creator">LDC AGTK</OtherMetadata>
    </Metadata>
  - <Timeline id="ANN20020115.0004:T1">
      <Signal id="ANN20020115.0004:T1:1" xlink:href="../../sgm/ANN20020115.0004.sgm" xlink:type="simple" unit="char" encoding="utf-8" mimeType="sgml"
        mimeClass="text"/>
    </Timeline>
  - <AG id="ANN20020115.0004:AG1" timeline="ANN20020115.0004:T1">
    - <Metadata>
        <OtherMetadata name="paragraph">1</OtherMetadata>
        <OtherMetadata name="tbcomment"/>
        <OtherMetadata name="treebanking">( Paragraph ( NP-HLN 0 1 2 ) ( S-HLN ( NP-SBJ ( NP 3 4 5 ) ( PP 6 ( NP ( NP 7 8 9 ) 10 ( NP 11 12 13 14 15 16 ) ) ) ) ( NP-
          PRD 17 ( NP 18 ( NP 19 ) ) ) ) )</OtherMetadata>
      </Metadata>
      <Anchor id="ANN20020115.0004:AG1:Anchor1" offset="0.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor2" offset="8.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor3" offset="17.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor4" offset="25.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor5" offset="30.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor6" offset="37.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor7" offset="43.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor8" offset="45.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor9" offset="46.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor10" offset="53.000000"/>
      <Anchor id="ANN20020115.0004:AG1:Anchor11" offset="55.000000"/>
```

**Figure 4.4 :** Exemple de fichier XML (partie 1)



```
<Anchor id="ANN20020115.0004:AG1:Anchor20" offset="81.000000"/>
<Anchor id="ANN20020115.0004:AG1:Anchor21" offset="88.000000"/>
- <Annotation id="ANN20020115.0004:AG1:Annotation1" type="word" end="ANN20020115.0004:AG1:Anchor2" start="ANN20020115.0004:AG1:Anchor1">
    <Feature name="lookup-word">AlHkwmp</Feature>
    <Feature name="comment"/>
    <Feature name="selection">ANN20020115.0004:AG1:Annotation2</Feature>
  </Annotation>
- <Annotation id="ANN20020115.0004:AG1:Annotation2" type="solution" end="ANN20020115.0004:AG1:Anchor2" start="ANN20020115.0004:AG1:Anchor1">
    <Feature name="solution">(AlHukuwmapu) Al/DET+Hukuwm/NOUN+ap/NSUFF_FEM_SG+u/CASE_DEF_NOM</Feature>
    <Feature name="gloss">the + government/administration + [fem.sg.] + [def.nom.]</Feature>
    <Feature name="number">1</Feature>
  </Annotation>
- <Annotation id="ANN20020115.0004:AG1:Annotation3" type="word" end="ANN20020115.0004:AG1:Anchor3" start="ANN20020115.0004:AG1:Anchor2">
    <Feature name="lookup-word">Al>rdnyp</Feature>
    <Feature name="comment"/>
    <Feature name="selection">ANN20020115.0004:AG1:Annotation4</Feature>
  </Annotation>
- <Annotation id="ANN20020115.0004:AG1:Annotation4" type="solution" end="ANN20020115.0004:AG1:Anchor3" start="ANN20020115.0004:AG1:Anchor2">
    <Feature name="solution">(Al>urodun~iy~apu) Al/DET+>urodun~iy~/ADJ+ap/NSUFF_FEM_SG+u/CASE_DEF_NOM</Feature>
    <Feature name="gloss">the + Jordanian + [fem.sg.] + [def.nom.]</Feature>
    <Feature name="number">1</Feature>
  </Annotation>
- <Annotation id="ANN20020115.0004:AG1:Annotation5" type="word" end="ANN20020115.0004:AG1:Anchor4" start="ANN20020115.0004:AG1:Anchor3">
    <Feature name="lookup-word">Aljdydp</Feature>
    <Feature name="comment"/>
    <Feature name="selection">ANN20020115.0004:AG1:Annotation6</Feature>
  </Annotation>
- <Annotation id="ANN20020115.0004:AG1:Annotation6" type="solution" end="ANN20020115.0004:AG1:Anchor4" start="ANN20020115.0004:AG1:Anchor3">
    <Feature name="solution">(Aljadiydapu) Al/DET+jadiyd/ADJ+ap/NSUFF_FEM_SG+u/CASE_DEF_NOM</Feature>
    <Feature name="gloss">the + new/modern + [fem.sg.] + [def.nom.]</Feature>
    <Feature name="number">1</Feature>
  </Annotation>
```

**Figure 4.5 :** Exemple de fichier XML (partie 2)

## 2.4 Le format Tree

Le format « tree » représente le corpus en deux versions (voyellée ou non voyellée) sous forme d'arborescence affichant chaque mot translitéré dans sa structure hiérarchique syntaxique et devant son étiquette POS. La figure 4.6 montre un exemple d'un fichier au format « tree » non voyelle, par contre la figure 4.7 présente la version voyelle du même fichier.

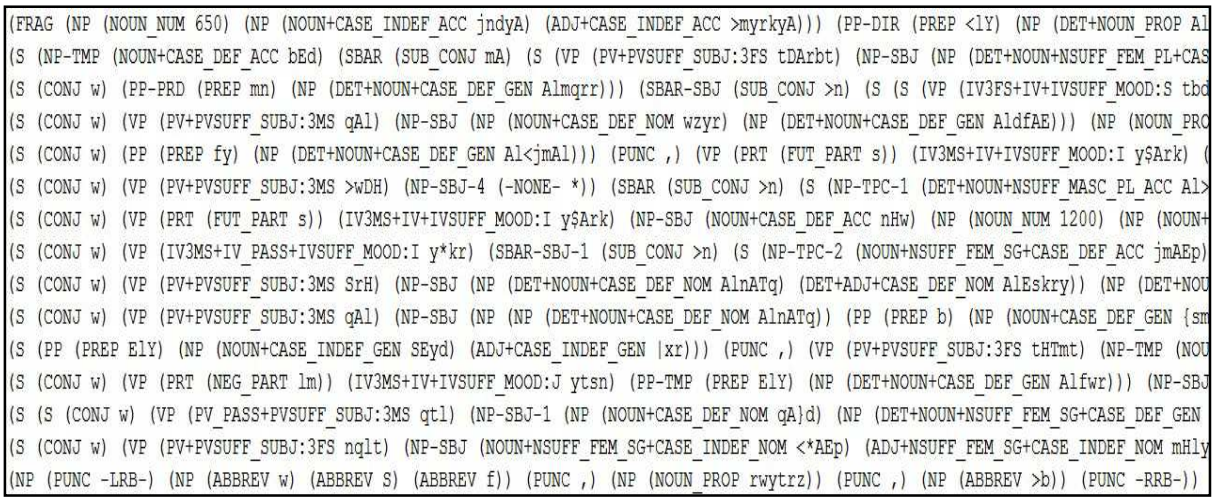



**Figure 4.6 :** Exemple de fichier au format « tree » non voyellé

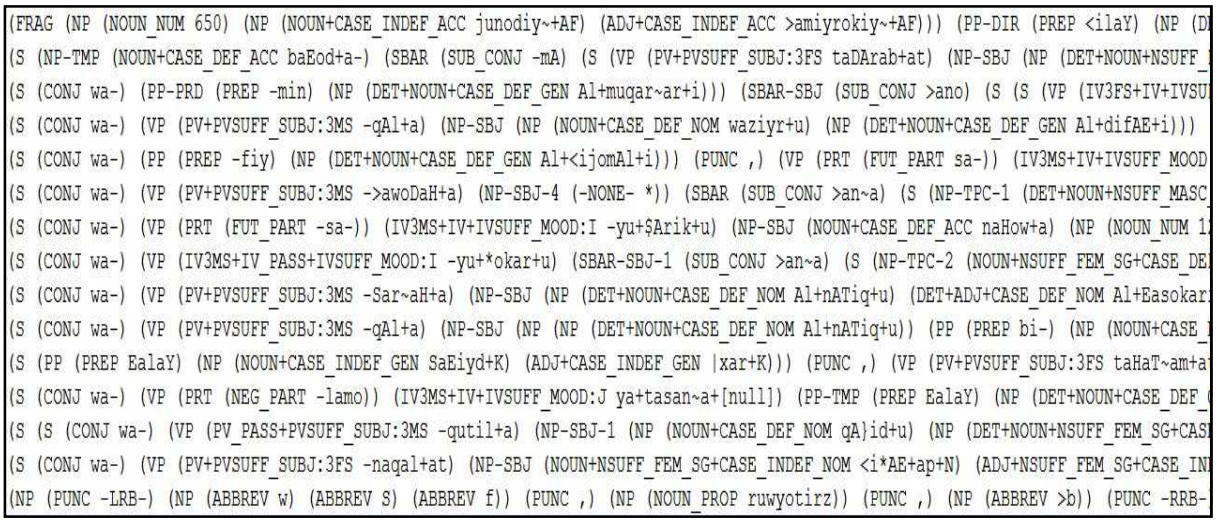



**Figure 4.7 :** Exemple de fichier au format « tree » voyellé

## 2.5 Le format Integrated

Finalement, le format « integrated » affiche des informations aussi bien sur la structure arborescente que sur chaque mot source avant et après la séparation des agglutinations. Ce format réunie dans un même fichier les informations sur les mots du corpus depuis le format « POS » avec ses deux sous catégorie « before » et « after » ainsi que les structures arborescentes du fichier XML. La figure 4.8 et 4.9 représentent un extrait d'un fichier au format « integrated ».

```
FILEPREFIX:
;; <?xml version="1.0"?>
;; <!DOCTYPE AGSet SYSTEM "ag.dtd">
;; <AGSet id="ANN20020115.0001" version="1.0" xmlns="http://www.ldc.upenn.edu/atlas/ag/" xmlns:xlink="http://www.w3.org/1999/xl
;; <Metadata>
;; <MetadataElement name="dc:source">../tim/100603/txt/ANN20020115.0001.pos</MetadataElement>
;; <MetadataElement name="dc:language">Arabic</MetadataElement>
;; <MetadataElement name="dc:title">Arabic Tree Bank - ANNAHAR</MetadataElement>
;; <MetadataElement name="dc:date">Mon Oct  6 22:01:53 2003</MetadataElement>
;; <MetadataElement name="dc:creator">LDC AGTK</MetadataElement>
;; </Metadata>
;; <Timeline id="DOC_ANN20020115.0001:Timeline1">
;; <Signal id="DOC_ANN20020115.0001:Timeline1:Signal1" mimeClass="text" encoding="utf-8" unit="char" xlink:type="simple" xlink:
;; </Timeline>

CHUNK:ANN20020115.0001:1
;; <MetadataElement name="paragraph">1</MetadataElement>
;; <MetadataElement name="tbcomment"></MetadataElement>
#source tokens:16
#tree tokens:17
#trees:1
s:0  ·650·650·0·4·0·0·4·[DEFAULT]·(650)·OK
s:1  ·جندياً·jndyAF·4·11·1·1·1·[junodiy~_1]·(junodiy~AF)·OK
```

**Figure 4.8 :** Première partie d'un exemple de fichier au format « Integrated »

```
s:11 ·على·ElY·66·70·12·12·1·[EalaY_1]·(EalaY)·OK
s:12 ·"·"·70·71·13·13·4·[DEFAULT]·()·OK
s:13 ·أبو·>bw·71·75·14·14·1·[>abuw_1]·(>abuw)·OK
s:14 ·سياف·syAf·75·79·15·15·1·[say~Af_1]·(say~Af)·OK
s:15 ·"·"·79·80·16·16·4·[DEFAULT]·()·OK
t:0  ·NOUN_NUM·f·f·650·nogloss·0·4·650·650·[]
t:1  ·NOUN+CASE_INDEF_ACC·f·f·junodiy~+AF·soldier + [acc.indef.]·4·11·جندياً·jndyA·[]
t:2  ·ADJ+CASE_INDEF_ACC·f·f·>amiyrokiy~+AF·American + [acc.indef.]·11·20·أميركياً·>myrkyA·[]
t:3  ·PREP·f·f·<ilaY·to/towards·20·24·إلى·<lY·[]
t:4  ·DET+NOUN_PROP·f·f·Al+fiyliyb~iyn·the + Philippines·24·34·الفيليبين·Alfylybyn·[]
t:5  ·PREP·f·f·min·from·34·37·من·mn·[]
t:6  ·DET+NOUN+CASE_DEF_GEN·f·f·Al+yawom+i·the + today + [def.gen.]·37·43·اليوم·Alywm·[]
t:7  ·PREP·f·f·fiy·in·43·46·في·fy·[]
t:8  ·NOUN+NSUFF_FEM_SG+CASE_INDEF_GEN·f·f·biEov+ap+K·delegation/mission + [fem.sg.] + [indef.gen.]·46·51·بعثة·bEvp·[]
t:9  ·ADJ+NSUFF_FEM_SG+CASE_INDEF_GEN·f·f·tadoriybiy~+ap+K·training/coaching/practice + [fem.sg.] + [indef.gen.]·51·59·تدريبية·tc
t:10 ·PREP·f·t·li·for/to·59·60·ل·l·[Separated]
t:11 ·DET+NOUN+CASE_DEF_GEN·t·f·Al+qaDA'+i·the + extermination/annihilation + [def.gen.]·60·66·القضاء·AlqDA'·[]
t:12 ·PREP·f·f·EalaY·on/above·66·70·على·ElY·[]
t:13 ·PUNC·f·f·"·nogloss·70·71·"·"·[]
t:14 ·NOUN_PROP·f·f·>abuw·Abu·71·75·أبو·>bw·[]
t:15 ·NOUN_PROP·f·f·say~Af·Sayyaf·75·79·سياف·syAf·[]
t:16 ·PUNC·f·f·"·nogloss·79·80·"·"·[]

TREE:ANN20020115.0001:1:1:17
(TOP (FRAG (NP W0 (NP W1 W2)) (PP-DIR W3 (NP W4)) (PP-TMP W5 (NP W6)) (PP W7 (NP (NP W8 W9) (PP-PRP W10 (NP (NP W11) (PP W12 (NP
```

**Figure 4.9 :** Deuxième partie d'un exemple de fichier au format « Integrated »

Dans nos travaux nous avons eu recours la plupart des formats de fichier existant dans le PATB en nous focalisant spécialement sur les formats « XML » et « Tree » car ils contiennent les informations nécessaires à nos expérimentations et avec une représentation hiérarchique et claire.

# 3. Mesures d'évaluation

Afin d'évaluer notre proposition, nous utilisons les métriques standards de l'évaluation des analyseurs syntaxiques (Thompson, 1992) ; à savoir la précision, le rappel, la F-mesure. Les définitions de ces métriques ont été présentées dans la section 1 du chapitre 3. Nous avons aussi utilisé une autre mesure employée pour l'évaluation des analyseurs syntaxiques qui est l'exactitude ou « accuracy » en anglais qui traduit le pourcentage des étiquettes correctement associées par rapport à l'ensemble des mots testés. La formule qui permet de calculer l'exactitude est la suivante :

$$Exactitude = \frac{\text{Etiquettes correctement associées}}{\text{Ensemble des mots testés}}$$

# 4. Evaluation de la méthode d'analyse numérique

## 4.1 Présentation de l'outil CRFsuite

Afin de pouvoir réaliser l'apprentissage automatique de notre modèle CRF, nous avons eu recours à l'outil CRFsuite (Okazaki, 2007). Celui-ci est une implémentation des champs conditionnels aléatoire pour l'étiquetage séquentiel des données. Cet outil offre plusieurs caractéristiques avantageuses comme :

- Un processus d'apprentissage et d'étiquetage rapide.
- Un format simple des données pour l'apprentissage et l'étiquetage.
- Des modèles d'apprentissage récents
- Possibilité de calculer les valeurs de rappel, précision, f-mesure et exactitude sur les données de test.
- Un format efficient pour le stockage du modèle généré pour une meilleure accessibilité.

La figure suivante présente l'outil CRFsuite durant le processus d'apprentissage sur notre corpus.

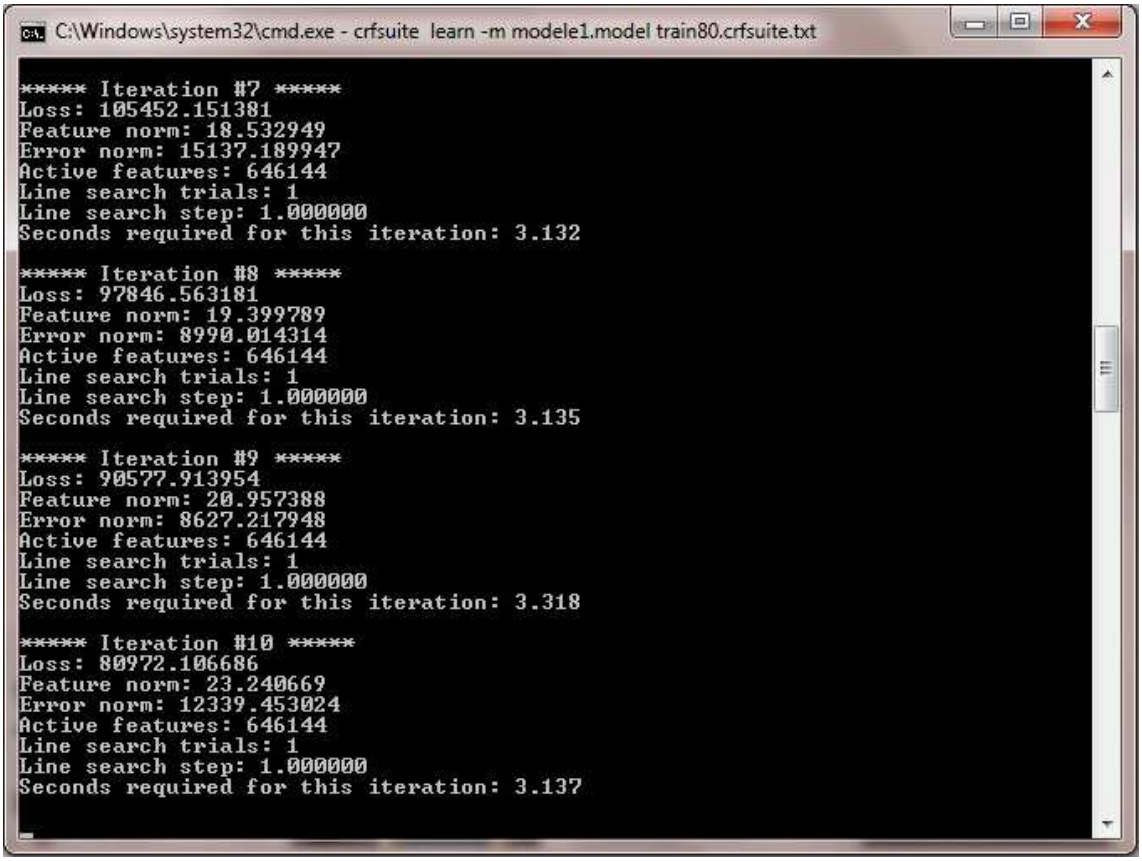



**Figure 4.10 :** Apprentissage avec l'outil CRFsuite

Cet outil permet aussi d'effectuer l'apprentissage automatique tout en observant l'évolution des performances à chaque itération d'apprentissage. Les performances sont calculer sur un ensemble de données défini des le départ par l'utilisateur. Cette commande permet d'observer l'amélioration du modèle au fur et à mesure que la phase d'apprentissage progresse. La figure 4.11 et 4.12 présentent un exemple d'apprentissage utilisant ce mode d'apprentissage.

```
L-BFGS optimization
c1: 0.000000
c2: 1.000000
num_memories: 6
max_iterations: 2147483647
epsilon: 0.000010
stop: 10
delta: 0.000010
linesearch: MoreThuente
linesearch.max_iterations: 20

***** Iteration #1 *****
Log-likelihood: -279935.059188
Feature norm: 5.000000
Error norm: 45136.783202
Active features: 452755
Line search trials: 2
Line search step: 0.000050
Seconds required for this iteration: 5.230
Performance by label (#match, #model, #ref) (precision, recall, F1):
    B-NP: (8312, 10265, 12422) (0.8097, 0.6691, 0.7328)
    B-PP: (3986, 6030, 4811) (0.6610, 0.8285, 0.7354)
    I-NP: (14116, 27744, 14376) (0.5088, 0.9819, 0.6703)
    B-VP: (0, 0, 4658) (0.0000, 0.0000, 0.0000)
    I-VP: (0, 0, 2646) (0.0000, 0.0000, 0.0000)
    B-SBAR: (0, 0, 535) (0.0000, 0.0000, 0.0000)
    O: (3298, 3338, 6180) (0.9880, 0.5337, 0.6930)
    B-ADJP: (0, 0, 438) (0.0000, 0.0000, 0.0000)
    B-ADVP: (0, 0, 866) (0.0000, 0.0000, 0.0000)
    I-ADVP: (0, 0, 89) (0.0000, 0.0000, 0.0000)
    I-ADJP: (0, 0, 167) (0.0000, 0.0000, 0.0000)
    I-SBAR: (0, 0, 4) (0.0000, 0.0000, 0.0000)
    I-PP: (0, 0, 48) (0.0000, 0.0000, 0.0000)
    B-PRT: (0, 0, 106) (0.0000, 0.0000, 0.0000)
    B-LST: (0, 0, 5) (0.0000, 0.0000, 0.0000)
    B-INTJ: (0, 0, 2) (0.0000, 0.0000, 0.0000)
    I-INTJ: (0, 0, 0) (******, ******, ******)
    B-CONJP: (0, 0, 9) (0.0000, 0.0000, 0.0000)
```

**Figure 4.11 :** Apprentissage avec évaluation à chaque itération (partie 1)

La figure 4.11 indique les performances de l'itération 1 en calculant les valeurs de précision, rappel et f-mesure pour chaque étiquette syntaxique décorée par le schéma IOB. Ainsi par exemple pour l'étiquette B-NP, 8313 est le nombre d'étiquettes communes entre la référence et le résultat des tests, 10265 est le nombre d'étiquettes généré par le modèle et le nombre d étiquettes de référence est de 12422. La figure 4.12 présente l'itération d'apprentissage numéro 162. A partir des deux figures 4.11 et 4.12, nous pouvons remarquer que pour l'étiquette B-NP, la précision a augmenté passant de 0.8097 a 0.9708.

```
***** Iteration #162 *****
Log-likelihood: -13143.933308
Feature norm: 81.103204
Error norm: 2.207139
Active features: 452755
Line search trials: 1
Line search step: 1.000000
Seconds required for this iteration: 1.980
Performance by label (#match, #model, #ref) (precision, recall, F1):
    B-NP: (12013, 12374, 12422) (0.9708, 0.9671, 0.9689)
    B-PP: (4713, 4878, 4811) (0.9662, 0.9796, 0.9729)
    I-NP: (13998, 14497, 14376) (0.9656, 0.9737, 0.9696)
    B-VP: (4470, 4668, 4658) (0.9576, 0.9596, 0.9586)
    I-VP: (2551, 2700, 2646) (0.9448, 0.9641, 0.9544)
    B-SBAR: (449, 499, 535) (0.8998, 0.8393, 0.8685)
    O: (5945, 6122, 6180) (0.9711, 0.9620, 0.9665)
    B-ADJP: (322, 403, 438) (0.7990, 0.7352, 0.7658)
    B-ADVP: (711, 836, 866) (0.8505, 0.8210, 0.8355)
    I-ADVP: (54, 82, 89) (0.6585, 0.6067, 0.6316)
    I-ADJP: (110, 137, 167) (0.8029, 0.6587, 0.7237)
    I-SBAR: (2, 15, 4) (0.1333, 0.5000, 0.2105)
    I-PP: (34, 42, 48) (0.8095, 0.7083, 0.7556)
    B-PRT: (80, 102, 106) (0.7843, 0.7547, 0.7692)
    B-LST: (0, 0, 5) (0.0000, 0.0000, 0.0000)
    B-INTJ: (1, 1, 2) (1.0000, 0.5000, 0.6667)
    I-INTJ: (0, 0, 0) (******, ******, ******)
    B-CONJP: (5, 8, 9) (0.6250, 0.5556, 0.5882)
```

**Figure 4.12 :** Apprentissage avec évaluation à chaque itération (partie 2)

L'outil CRFsuite permet aussi d'évaluer le modèle appris sur un ensemble de données de tests qui sont correctement étiquetées. Il calcul alors le rappel, la précision et la f-mesure pour chaque étiquettes et donne aussi la moyenne obtenus sur toutes les étiquettes utilisées. Cette commande est très intéressante car elle permet de faire une validation rapide du modèle générée. La figure 4.13 montre le résultat de la phase de test de l'outil CRFsuite.

```
$ crfsuite tag -qt -m CoNLL2000.model test.crfsuite.txt
Performance by label (#match, #model, #ref) (precision, recall, F1):
    B-NP: (12000, 12358, 12407) (0.9710, 0.9672, 0.9691)
    B-PP: (4707, 4872, 4805) (0.9661, 0.9796, 0.9728)
    I-NP: (13984, 14484, 14359) (0.9655, 0.9739, 0.9697)
    B-VP: (4466, 4662, 4653) (0.9580, 0.9598, 0.9589)
    I-VP: (2549, 2698, 2643) (0.9448, 0.9644, 0.9545)
    B-SBAR: (448, 498, 534) (0.8996, 0.8390, 0.8682)
    O: (5939, 6113, 6174) (0.9715, 0.9619, 0.9667)
    B-ADJP: (322, 403, 438) (0.7990, 0.7352, 0.7658)
    B-ADVP: (711, 835, 866) (0.8515, 0.8210, 0.8360)
    I-ADVP: (54, 82, 89) (0.6585, 0.6067, 0.6316)
    I-ADJP: (110, 137, 167) (0.8029, 0.6587, 0.7237)
    I-SBAR: (2, 15, 4) (0.1333, 0.5000, 0.2105)
    I-PP: (34, 42, 48) (0.8095, 0.7083, 0.7556)
    B-PRT: (80, 102, 106) (0.7843, 0.7547, 0.7692)
    B-LST: (0, 0, 4) (0.0000, 0.0000, 0.0000)
    B-INTJ: (1, 1, 2) (1.0000, 0.5000, 0.6667)
    I-INTJ: (0, 0, 0) (******, ******, ******)
    B-CONJP: (5, 7, 9) (0.7143, 0.5556, 0.6250)
    I-CONJP: (10, 12, 13) (0.8333, 0.7692, 0.8000)
    I-PRT: (0, 0, 0) (******, ******, ******)
    B-UCP: (0, 0, 0) (******, ******, ******)
    I-UCP: (0, 0, 0) (******, ******, ******)
Macro-average precision, recall, F1: (0.639239, 0.602512, 0.611086)
Item accuracy: 45422 / 47321 (0.9599)
Instance accuracy: 1176 / 2011 (0.5848)
Elapsed time: 0.940000 [sec] (2140.4 [instance/sec])
```

**Figure 4.13 :** Evaluation du modèle généré avec CRFsuite

Cette commande permet aussi de calculer l'exactitude niveau étiquette ainsi que l'exactitude niveau phrase du modèle par rapport un corpus de référence.

## 4.2 Corpus d'apprentissage

Pour pouvoir réaliser la phase d'apprentissage automatique, Le corpus PATB ne peut pas être directement exploité à travers l'un de ses formats existants que nous venons de présenter dans la section 2. En effet il doit être converti en un nouveau fichier selon une représentation standard adoptée par les modèles CRF, le formatage IOB (présenté dans le chapitre 3). Considérons la phrase suivante extraite du corpus PATB:

**ستة مسؤولين أميركيين وصلوا إلى البلاد ل البحث في التدابير اللوجستية**

*(Six responsables américains sont arrivés au pays pour trouver des solutions logistique.)*

Après conversion au format IOB, cette phrase aura une forme tabulaire comme le montre la figure 4.2.

| | | |
|---|---|---|
| ستة | NOUN_NUM | B-NP |
| مسؤولين | NOUN | I-NP |
| أميركيين | ADJ | I-NP |
| وصلوا | PV | B-VP |
| إلى | PREP | I-VP |
| البلاد | NOUN | I-VP |
| ل | PREP | I-VP |
| البحث | NOUN | I-VP |
| في | PREP | I-VP |
| التدابير | NOUN | I-VP |
| اللوجستية | ADJ | I-VP |
| . | . | O |

**Figure 4.14 :** Extrait de corpus après conversion au format IOB

Ensuite nous intégrons les critères d'apprentissage, que nous avons définis, dans le corpus d'apprentissage à travers des bigrammes et des trigrammes. Ces critères

d'apprentissage sont définis par les mots eux mêmes et leurs catégories grammaticale. La figure suivante présente un extrait du corpus d'apprentissage après intégration des critères.

```
B-NP    w[0]=ستة    w[1]=مسؤولين    w[2]=أميركيين    w[0]|w[1]=ستة|مسؤولين
I-NP    w[-1]=ستة    w[0]=مسؤولين    w[1]=أميركيين    w[2]=وصلوا    w[-1]|w[0]
I-NP    w[-2]=ستة    w[-1]=مسؤولين    w[0]=أميركيين    w[1]=وصلوا    w[2]=إلى
B-VP    w[-2]=مسؤولين    w[-1]=أميركيين    w[0]=وصلوا    w[1]=إلى    w[2]=البلاد
I-VP    w[-2]=أميركيين    w[-1]=وصلوا    w[0]=إلى    w[1]=البلاد    w[2]=ل    w[-1]|
I-VP    w[-2]=وصلوا    w[-1]=إلى    w[0]=البلاد    w[1]=ل    w[2]=البحث    w[-1]|w[0]
I-VP    w[-2]=إلى    w[-1]=البلاد    w[0]=ل    w[1]=البحث    w[2]=في    w[-1]|w[0]=ال
I-VP    w[-2]=البلاد    w[-1]=ل    w[0]=البحث    w[1]=في    w[2]=التدابير    w[-1]|w[0]
I-VP    w[-2]=ل    w[-1]=البحث    w[0]=في    w[1]=التدابير    w[2]=اللوجستية    w[-1]|
I-VP    w[-2]=البحث    w[-1]=في    w[0]=التدابير    w[1]=اللوجستية    w[2]=,    w[
I-VP    w[-2]=في    w[-1]=التدابير    w[0]=اللوجستية    w[1]=,    w[2]=على    w[
O    w[-2]=التدابير    w[-1]=اللوجستية    w[0]=,    w[1]=على    w[2]=أن    w[-1]|w[0]
```

**Figure 4.15 :** Extrait du corpus d'apprentissage après intégration des critères

Cette intégration consiste en le remplissage des valeurs de chaque critère pour chaque mot du corpus d'apprentissage. La figure suivante présente un exemple d'application des critères pour le mot « **أميركيين** ».

| |
|---|
| w[-2]= ستة<br>w[-1]=مسؤولين<br>w[0]=أميركيين<br>w[1]=وصلوا<br>w[2]=إلى |
| w[-1]\|w[0]=مسؤولين\|أميركيين<br>w[0]\|w[1]=أميركيين\|وصلوا |
| pos[-2]=NOUN_NUM<br>pos[-1]=NOUN<br>pos[0]=ADJ<br>pos[1]=PV<br>pos[2]=PREP |
| pos[-2]\|pos[-1]=NOUN_NUM\|NOUN<br>pos[-1]\|pos[0]=NOUN\|ADJ<br>pos[0]\|pos[1]=ADJ\|PV<br>pos[1]\|pos[2]=PV\|PREP |
| pos[-2]\|pos[-1]\|pos[0]=NOUN_NUM\|NOUN\|ADJ<br>pos[-1]\|pos[0]\|pos[1]=NOUN\|ADJ\|PV |

| pos[0]\|pos[1]\|pos[2]=ADJ\|PV\|PREP |
|---|

**Figure 4.16 :** Les valeurs des critères utilisés pour le mot « أميركيين »

A l'issue de la phase d'apprentissage, nous obtenons le modèle CRF. Dans ce qui suit, nous présentons la méthode d'évaluation que nous avons suivit.

## 4.3 Evaluation par validation croisée

La validation croisée, cross-validation en anglais, est une méthode d'estimation de fiabilité d'un modèle fondé sur une technique d'échantillonnage. Concrètement, Cette méthode d'évaluation n'est pas à proprement parler un algorithme établissant la significativité des résultats. Elle permet de calculer un score moyen sur un ensemble de données d'évaluation de taille importante.

En fait, il y a au moins trois techniques de validation croisée : « test-set validation » ou « holdout method », « k-fold cross-validation » et « leave-one-out cross validation » (Kohavi, 1995):

- La première méthode est très simple, il suffit de diviser l'échantillon de taille n en échantillon d'apprentissage (> 60 % de l'échantillon) et échantillon de test. Le modèle est bâti sur l'échantillon d'apprentissage et validé sur l'échantillon de test. L'erreur est estimée en calculant un test, une mesure ou un score de performance du modèle sur l'échantillon de test, par exemple l'exactitude.

- Dans la seconde, on divise k fois l'échantillon, puis on sélectionne un des k échantillons comme ensemble de validation et les (k-1) autres échantillons constitueront l'ensemble d'apprentissage. On calcule comme dans la première méthode l'erreur quadratique moyenne. Puis on répète l'opération en sélectionnant un autre échantillon de validation parmi les (k-1) échantillons qui n'ont pas encore été utilisés pour la validation du modèle. L'opération se répète ainsi k fois pour qu'en fin de compte chaque sous-échantillon ait été utilisé exactement une fois comme ensemble de validation. La moyenne des k erreurs quadratiques moyennes est enfin calculée pour estimer l'erreur de prédiction.

– La troisième méthode est un cas particulier de la deuxième méthode où k=n, c'est-à-dire que l'on apprend sur (n-1) observations puis on valide le modèle sur la énième observation et l'on répète cette opération n fois.

Nous avons suivie la méthode de validation croisée « test-set validation » pour réaliser l'évaluation.

## 4.4 Protocole expérimental et résultats obtenus

Ayant fixé la méthode d'évaluation, nous sommes passé à la répartition des données. Nous avons donc divisé notre corpus PATB en deux parties. Une première partie (80%) composée de 10.100 phrases est utilisée pour la phase d'apprentissage et une deuxième partie (20%) composée de 2.560 est utilisée pour la phase de validation. A partir de ces deux jeux de données, nous avons calculé dans un premier temps l'exactitude nous avons obtenu une valeur de 97,92% avec 81133 mots correctement étiquetés sur 82860 mots testés. Nous avons aussi calculé l'exactitude au niveau des phrases nous avons obtenu la valeur de 64,57% ce qui représente 1653 phrases correctement étiquetées sur 2560 phrases testées. Le reste des phrases testées ont été également étiquetées mais contiennent quelques erreurs d'étiquetage.

Nous avons ensuite calculé les mesures du rappel, précision et f-mesure sur les étiquettes syntaxiques majeures pour mesurer la performance du modèle par catégorie syntaxique (Khoufi, Aloulou, & Hadrich Belguith, 2015). Le tableau suivant expose les valeurs obtenues.

| **Constituants syntaxiques** | **Précision** | **Rappel** | **F-mesure** |
|---|---|---|---|
| NP | 93,32% | 95,55% | 94,41% |
| VP | 92,02% | 92,10% | 92,06% |
| PP | 49,80% | 49,94% | 49,87% |
| ADJP | 90,25% | 78,80% | 83,65% |
| ADVP | 48,74% | 48,41% | 48,58% |
| CONJP | 100,00% | 97,83% | 98,90% |
| O | 97,67% | 97,59% | 97,63% |
| S | 95,88% | 97,97% | 96,92% |

**Tableau 4.1 :** Résultats de l'évaluation de l'analyse numérique selon les étiquettes syntaxiques utilisées

En examinant les résultats affichés par le tableau 4.1, nous remarquons que notre modèle réussi à identifier les constituants syntaxiques avec une performance honorable. Le modèle atteint même une exactitude de 97,92%. Ces performances confirment les résultats obtenus pas l'analyseur SPA (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015) pour l'analyse du premier niveau syntaxique.

# 5. Evaluation de la méthode d'analyse symbolique

Les résultats de l'évaluation de cette méthode ont été présentés et discutés dans la section 2.2 et la section 3 du troisième chapitre. Dans cette section, nous présentons le prototype logiciel que nous avons développé pour tester cette méthode. Ce prototype permet de construire la grammaire PCFG arabe et permet aussi d'analyser des phrases en arabe standards (Khoufi, Aloulou, & Hadrich Belguith, 2016).

## 5.1 Présentation de l'outil

Afin de réaliser notre outil, nous avons utilisé le langage de programmation Java via l'environnement Eclipse pour la conception graphique des interfaces et pour l'induction de la grammaire. Nous avons aussi utilisé le langage de programmation Python version 3.4 pour l'implémentation de l'algorithme d'analyse Viterbi.

### 5.1.1 Interface d'induction de la grammaire PCFG

Notre outil est composé d'un ensemble d'interfaces. L'une d'entres elles permet de préparer la grammaire en autorisant d'abord l'utilisateur à indiquer les fichiers du corpus PATB qu'il veut utiliser pour l'induction de la grammaire. Ensuite une fois les fichiers choisis, notre outil généré des règles hors contexte (CFG) et les affiche à l'utilisateur. Ces règles sont affichées avec les doublons et les mots sont écrits avec la translitération Buckwalter.

Puis en continuant le processus, l'utilisateur raffine la grammaire en filtrant les annotations et en convertissant les mots depuis la notation Buckwalter vers la notation

arabe standard. A la fin, l'utilisateur va pouvoir assigner la probabilité de chaque règle générant ainsi la grammaire PCFG arabe finale.

Ce double affichage (CFG/PCFG) permet à l'utilisateur de constater le travail réalisé au niveau de la factorisation des annotations morphologiques et du calcul des probabilités des règles de la grammaire. La figure 4.5 présente l'interface d'induction de notre outil.

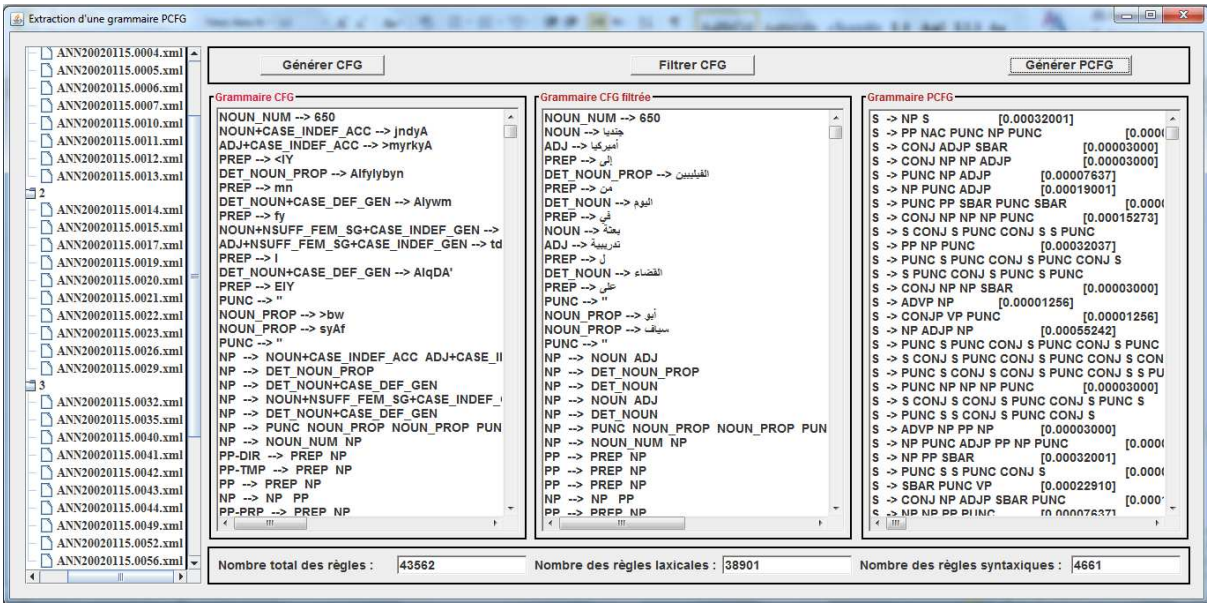



**Figure 4.17 :** Interface d'induction de la grammaire PCFG

Comme le montre la figure ci-dessus, le bouton « Générer CFG » permet d'induire les règles hors contexte sous leur forme brute. Le bouton « Filtrer CFG » permet de raffiner les règles et de convertir les mots vers l'arabe. Enfin le bouton « Générer PCFG » permet la constitution de la grammaire PCFG arabe finale.

Notre outil permet aussi d'afficher non seulement la grammaire induite, mais en plus des informations supplémentaires concernant le nombre de règles induites (contextuelles et lexicales).En effet, notre outil permet de trier les règles par sa partie gauche (NP, VP, ADJP,…) et affiche pour une règle sélectionnée le nombre d'occurrences, le nombre de règles avec la même partie gauche, sa probabilité et le nombre d'éléments dans la partie droite. La figure 4.6 expose l'interface de présentation de la grammaire PCFG. L'interface a deux grandes zones d'affichage. La zone de droite affiche les règles de la grammaire tandis que la zone de gauche est consacrée à l'affichage des informations supplémentaires.

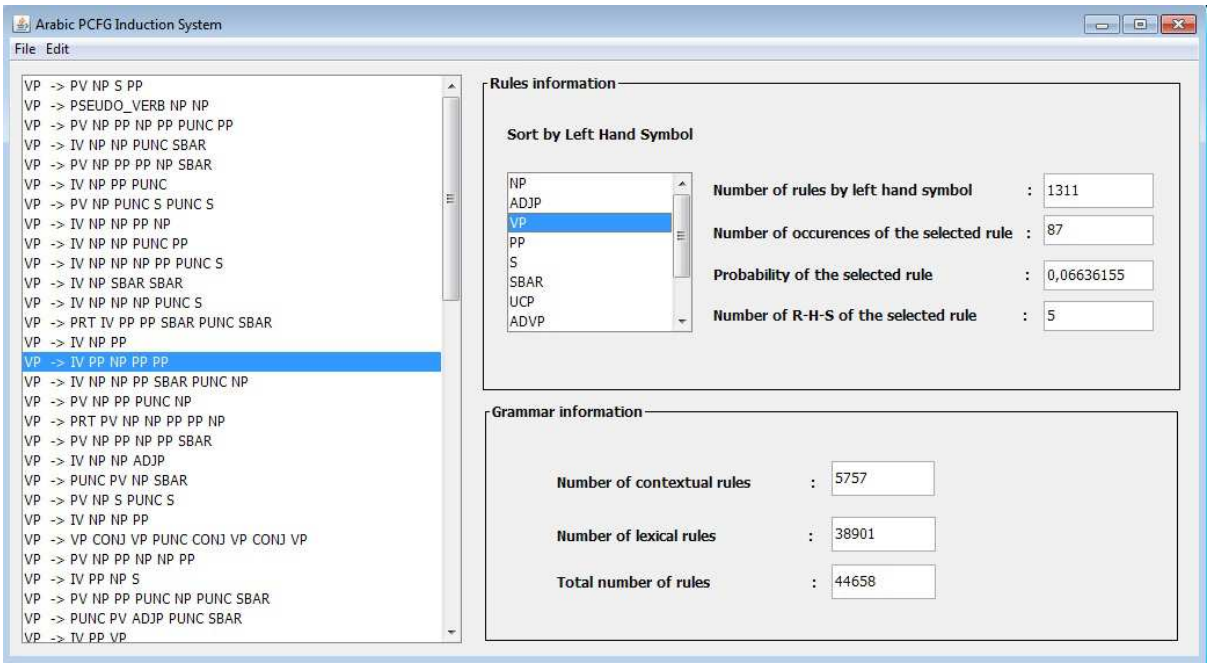



**Figure 4.18 :** Interface de présentation de la grammaire PCFG

## 5.1.2 Interface d'analyse

Une fois la grammaire induite par l'utilisateur, celui-ci peut passer à la phase d'analyse. Pour ce faire, l'utilisateur doit introduire une phrase segmentée. Le système va alors analyser cette chaine en utilisant l'algorithme de Viterbi intégré depuis une plateforme de TALN nommée NLTK.

Dans ce qui suit nous présentons d'abord la plate-forme NLTK puis nous présentons l'interface d'analyse syntaxique.

### 5.1.2.1 La plateforme NLTK

NLTK[4] (Natural Language Toolkit) est une plate-forme pour la construction de programmes Python pour le traitement automatique des langues naturelles. Elle fournit des interfaces faciles d'utilisation pour plus de 50 corpus (tel que le corpus WSJ) et ressources lexicales (telle que WordNet), avec un ensemble de bibliothèques de traitement de textes pour la classification, la segmentation, la lemmatisation, l'étiquetage, l'analyse syntaxique et le raisonnement sémantique.

[4] http://www.nltk.org/index.html

NLTK est disponible pour la plus part des systèmes d'exploitation tel que Windows, Mac OS et Linux. Cette plateforme est disponible gratuitement et en open-source.

Elle offre des solutions rapides pour l'implémentation de l'algorithme d'analyse syntaxique en Python tel que Viterbi, Recursive Descent Parsing, The Left-Corner Parser et bien d'autres.

### 5.1.2.2 Interface d'analyse

Cette interface permet à l'utilisateur d'exécuter le processus d'analyse syntaxique complet de la phrase. En effet, l'utilisateur saisi la phrase à analyser segmentée sous forme segmenté et ensuite en appuyant sur le bouton « analyse » le système utilise la grammaire PCFG avec l'algorithme Viterbi pour analyser la phrase et affiche le résultat obtenu sous forme parenthèsée avec calcul de la probabilité de l'arbre syntaxique. La figure 4.7 présente l'interface d'analyse de notre système.

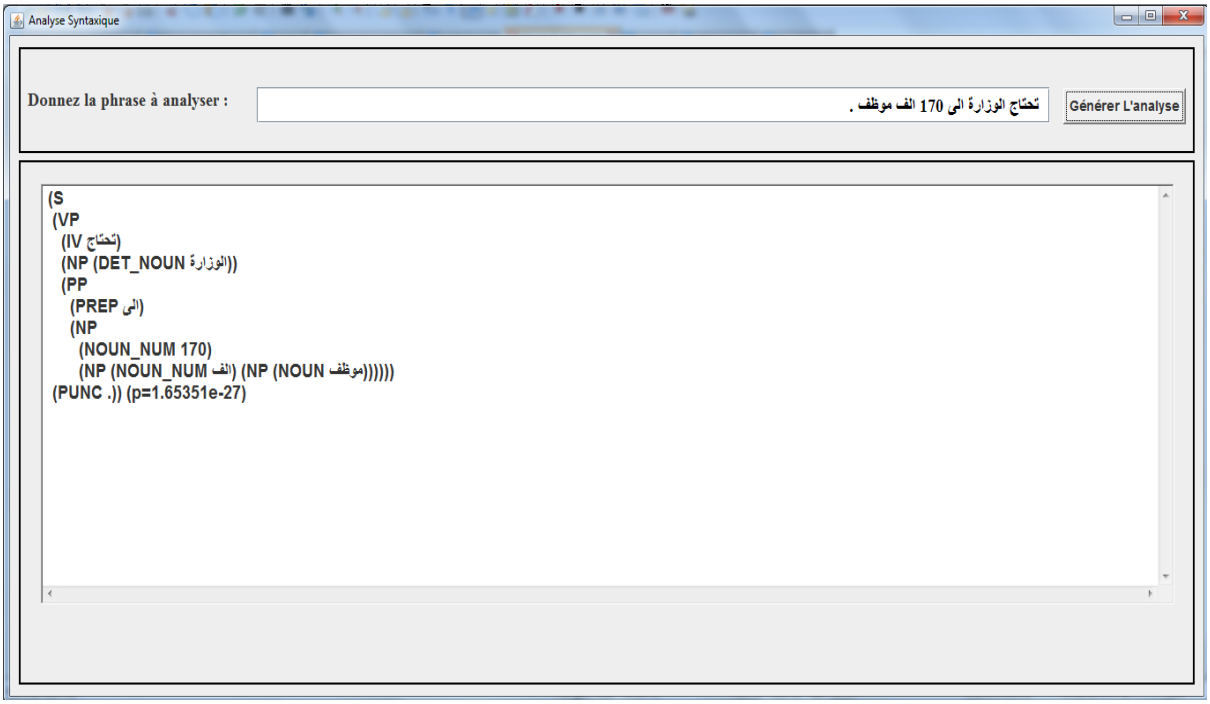



**Figure 4.19 :** Interface d'analyse

## 6. Evaluation de la méthode d'analyse hybride

Dans cette section nous présentons l'évaluation de la contribution principale de cette thèse, à savoir la méthode hybride pour l'analyse syntaxique. L'objectif de cette

évaluation est de mesurer l'apport de la méthode hybride par rapport à méthode purement symbolique déjà présenté dans le chapitre 3. Pour ce cela nous avons eu recours aux mesures standards qui sont la précision, le rappel et f-mesure.

Pour pouvoir choisir le seuil de notre méthode hybride, nous nous sommes basé sur les résultats obtenus par notre méthode symbolique indiquée dans la figure 3.7 du chapitre précédant. En effet, nous remarquons que le f-mesure passe de 86,36 % pour les phrases de longueur 11 mots à 80,73% pour les phrases de longueur 18 mots ; soit une baisse d'environs 6%. Cette chute brutale des résultats nous incite à choisir un seuil dans cet intervalle 11-17 mots pour espérer réduire l'importance de cette dégradation des performances. Nous donc avons choisi de fixer le seuil à une valeur médiane de 14 mots durant nos expérimentations.

Le corpus de test utilisé est composé de 72 phrases annotées morphologiquement extraites du corpus PATB. Ces phrases ont été choisi aléatoirement en suivant la repartions du PATB recommandée par de l'atelier "Johns Hopkins 2005 Workshop"[5]. Cette répartition a déjà été utilisée dans l'évaluation de notre méthode symbolique.

Les tailles des phrases varient de 4 à 38 mots avec une longueur moyenne de 21 mots. Pour chaque taille de phrase, nous avons analysé quatre phrases. Vingt phrases ont une taille inferieur à 14 mots et 52 phrases ont une taille supérieure ou égale à 14. Le tableau 4.2 présente les tailles des phrases du corpus de test.

| **Nombre de phrase** | **Taille (mots)** |
|---|---|
| 4 | 4 |
| 4 | 6 |
| 4 | 8 |
| 4 | 10 |
| 4 | 12 |
| 4 | 14 |
| 4 | 16 |
| 4 | 18 |
| 4 | 20 |
| 4 | 22 |
| 4 | 24 |

[5] http://nlp.stanford.edu/software/parser-arabic-data-splits.shtml

| | |
|---|---|
| 4 | 26 |
| 4 | 28 |
| 4 | 30 |
| 4 | 32 |
| 4 | 34 |
| 4 | 36 |
| 4 | 38 |
| **Total** = 72 phrases | **Moyenne** = 21 mots |

**Tableau 4.2 :** Taille et nombre de phrases du corpus de test

Ces phrases ont été analysée sur une machine équipé d'un système d'exploitation Windows 7 32 bits, un processeur Intel core 2 duos cadencé à 2.00 GHz et 3 giga de RAM.

Ces phrases ont été d'abord analysées avec la méthode symbolique afin constituer une base de référence à la quelle ont peut mesurer l'apport de hybridation à la procédure d'analyse. Ces mêmes phrases ont été ensuite analysées selon la méthode hybride. La figure suivante présente l'interface qui permet d'analyser des phrases selon la méthode hybride.

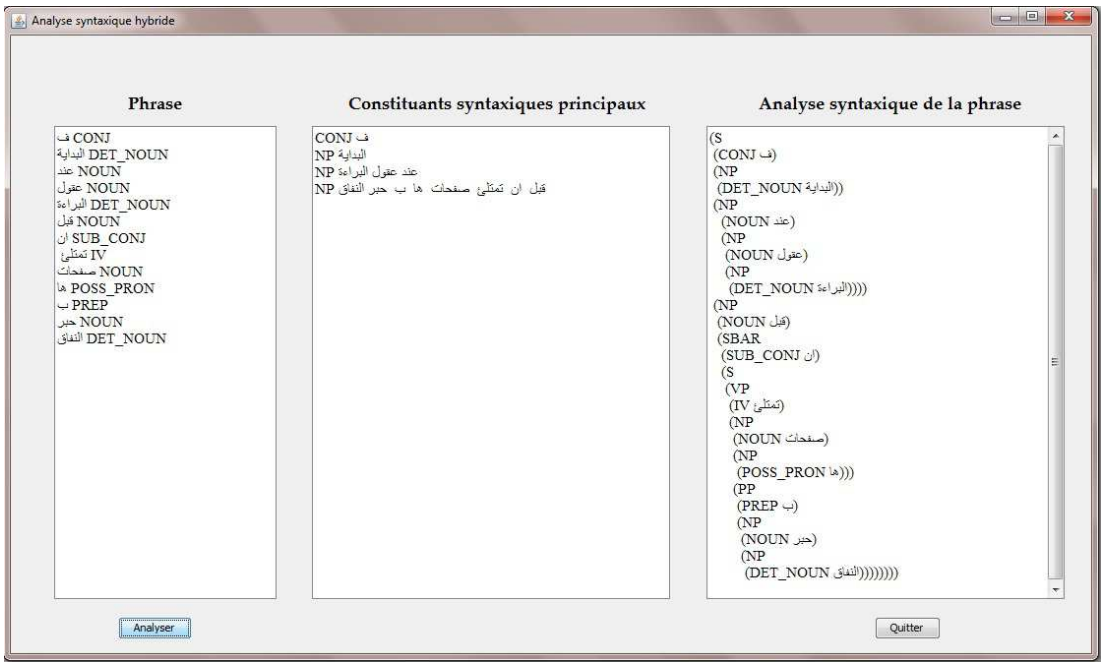



**Figure 4.20 :** Interface de l'analyse hybride

Nous avons calculé les mesures du rappel, précision et f-mesure pour chaque phrase et on a ensuite calculé la moyenne. Le tableau suivant explicite les moyennes obtenues au cours de cette évaluation

| **Taille des phrases** | **Méthode symbolique** | | | **Méthode hybride** | | |
|---|---|---|---|---|---|---|
| | **Rappel** | **Précision** | **F-mesure** | **Rappel** | **Précision** | **F-mesure** |
| 4 | 92,75% | 92,75% | 92,75% | 92,75% | 92,75% | 92,75% |
| 6 | 78,00% | 78,00% | 78,00% | 78,00% | 78,00% | 78,00% |
| 8 | 84,00% | 85,25% | 84,75% | 84,00% | 85,25% | 84,75% |
| 10 | 86,25% | 85,25% | 85,75% | 86,25% | 85,25% | 85,75% |
| 12 | 77,50% | 77,25% | 77,25% | 80,25% | 81,25% | 80,50% |
| 14 | 80,25% | 80,75% | 80,50% | 82,00% | 80,75% | 81,50% |
| 16 | 78,00% | 78,00% | 77,75% | 79,00% | 78,00% | 78,25% |
| 18 | 80,75% | 80,25% | 80,25% | 83,00% | 81,75% | 81,50% |
| 20 | 83,75% | 84,00% | 83,75% | 84,25% | 84,25% | 83,75% |
| 22 | 81,75% | 83,75% | 82,50% | 83,25% | 84,50% | 83,50% |
| 24 | 83,00% | 83,50% | 83,25% | 83,00% | 83,50% | 83,25% |
| 26 | 83,00% | 82,00% | 82,50% | 83,25% | 82,75% | 83,00% |
| 28 | 79,00% | 79,00% | 79,00% | 80,00% | 79,50% | 79,75% |
| 30 | 83,00% | 83,00% | 83,00% | 83,25% | 83,50% | 83,25% |
| 32 | 83,75% | 83,00% | 83,00% | 83,75% | 83,00% | 83,00% |
| 34 | 78,75% | 79,00% | 79,25% | 82,75% | 84,75% | 83,75% |
| 36 | 83,00% | 82,50% | 83,00% | 83,50% | 83,00% | 83,50% |
| 38 | 81,00% | 81,25% | 81,00% | 82,00% | 82,00% | 81,75% |
| **Moyenne** | 82,08% | 82,14% | 82,07% | 83,01% | 82,99% | 82,86% |

**Tableau 4.3 :** Moyennes des mesures de Précision, Rappel et F-mesure

Le tableau 4.3 permet de constater que l'analyse hybride a permis d'améliorer les résultats de précision, rappel et f-mesure de l'analyse symbolique de respectivement 0,85%, 0,93% et 0,79% . Ces valeurs sont encourageantes et montrent que les techniques d'hybridation s'adaptent bien pour l'analyse syntaxique de l'arabe non voyellé ce qui confirme les hypothèses que nous avons fixé au chapitre deux.

En nous intéressant plus en détails aux résultats de l'évaluation, nous avons remarqué que la méthode hybride permet d'améliorer les résultats de l'analyse des phrases longues comme nous le montre le tableau 4.4.

| | **Analyse symbolique** | | | **Analyse hybride** | | |
|---|---|---|---|---|---|---|
| | **Rappel** | **Précision** | **F-mesure** | **Rappel** | **Précision** | **F-mesure** |
| **Evaluation phrases ≥ S** | 81,46% | 81,54% | 81,44% | 82,54% | 82,40% | 82,29% |
| **Evaluation phrases avec des sous phrases** | 78,60% | 78,53% | 78,60% | 81,07% | 80,93% | 80,73% |

**Tableau 4.4 :** Résultats obtenus pour les phrases longues et contenants des sous phrases

En effet, les valeurs moyennes de rappel, précision et f-mesure pour les phrases contenants des sous phrases imbriquées passent de (R : 78,60%, P : 78,53%, F : 78,60%) avec la méthode symbolique à (P : 81 ,07%, R : 80,93%, F : 80,73%) avec la méthode hybride soit une augmentation moyenne de 2,33%. Cette amélioration est aussi remarquable pour les phrases de longueurs supérieur ou égale à 14 mots avec une augmentation moyenne de 0,93%.

Ces résultats montrent que plus la phrase est longue, plus elle a tendance à contenir des phrases imbriqués. Ce phénomène d'imbrication des phrases augmente la complexité de l'analyse et la méthode symbolique seule à du mal à donner une analyse fiable. Par contre en utilisant la méthode hybride, l'analyse numérique appliquée au départ facilite le travail de l'analyse symbolique en donnant les frontières des phrases imbriquées. Celles-ci étant identifiées, l'analyse symbolique peut alors donner une structure syntaxique de ces phrases et par conséquent de la phrase globale. Ce cas est illustré par l'exemple de la phrase suivante :

**و انعكس ذلك على أسعار الكاكاو عالميا إذ بلغ سعر الطن تسليم آذار المقبل 2340 أورو.**

Translitération Buckwalter: w AnEks *lk ElY OsEAr AlkAkAw EAlmyA I* blg sEr AlTn tslym |*Ar Almqbl 2340 Owrw.

*(Et cela s'est répercuté sur le prix du cacao mondialement atteignant un prix par tonne pour Mars prochain de 2340 euros.)*

- Sous phrase 1 :

**و انعكس ذلك على أسعار الكاكاو عالميا**

Translitération Buckwalter: w AnEks *lk ElY OsEAr AlkAkAw EAlmyA

*(Et cela c'est répercuté sur le prix du cacao mondialement)*

- Sous phrase 2 :

**إذ بلغ سعر الطن تسليم آذار المقبل 2340 أورو**

Translitération Buckwalter: I* blg sEr AlTn tslym |*Ar Almqbl 2340 Owrw.

*(Atteignant un prix par tonne pour prochain Mars de 2340 euros.)*

Dans cet exemple, la sous phrase 2 est considérée par l'analyse symbolique comme un syntagme faisant parti de la sous phrase 1 tandis que l'analyse hybride identifie les deux sous phrases et les analyse séparément.

Nous avons aussi remarqué que la méthode hybride donne de meilleurs résultats avec les phrases nominales qu'avec les phrases verbales. Les phrases nominales ont généralement une structure plus détaillée au premier niveau de l'analyse. Ce qui permet au module numérique de définir les structures syntaxiques principales de la phrase et les passe au module symbolique pour déterminer leurs structures syntaxiques internes. La tâche d'analyse syntaxique est ainsi mieux repartie entre les deux modules. Par contre, les phrases verbales sont généralement formées par un syntagme verbal qui englobe la structure syntaxique globale de la phrase ce qui diminue l'effet du traitement numérique dans notre processus d'analyse hybride.

Nous nous sommes aussi intéressés à la performance de nos deux méthodes au niveau du temps d'analyse pour avoir une idée sur le gain de temps acquis par l'utilisation de la méthode hybride. Ce paramètre de temps d'exécution est important dans les applications de TALN qui visent à donner des résultats dans les plus brefs délais. Nous avons donc calculé le temps d'analyse pour chaque phrase testée en utilisant d'abord l'analyse symbolique puis l'analyse hybride et ensuite, nous avons calculé la moyenne de ces temps selon la taille de la phrase. Les temps d'exécution que nous avons obtenus sont présentés dans le tableau 4.5.

| **Nombre de mots de la phrase analysée** | **Analyse symbolique** | **Analyse hybride** |
|---|---|---|
| 4 | 0,15 | 0,15 |
| 6 | 0,31 | 0,31 |
| 8 | 0,53 | 0,53 |
| 10 | 0,93 | 0,93 |
| 12 | 1,45 | 1,45 |
| 14 | 2,25 | 1,18 |
| 16 | 3,37 | 1,59 |
| 18 | 5,39 | 2,19 |
| 20 | 10,00 | 3,38 |
| 22 | 15,87 | 11,13 |
| 24 | 28,07 | 12,25 |
| 26 | 49,58 | 18,13 |
| 28 | 51,63 | 32,79 |
| 30 | 91,77 | 44,50 |
| 32 | 120,50 | 42,46 |
| 34 | 184,88 | 44,93 |
| 36 | 285,21 | 98,02 |
| 38 | 421,92 | 158,82 |

**Tableau 4.5 :** Temps moyen d'exécution en minutes

Le tableau 4.5 montre une diminution globale des temps d'analyses. Dans des cas, la diminution est plus importante que dans d'autres. Par exemple pour les phrases de taille 20 mots, le temps d'analyse passe de 10 à 3,38 (diminution de 66,2%) minutes. Pour les phrase de taille 22 mots, ce temps passe de 15,87 à 11,38 minutes (diminution de 28,29%). Cette différence de performance est due à la structure syntaxique de la phrase analysée. Les phrases de longueur 20 mots ont été découpé en plusieurs constituants syntaxiques par le modèle numérique, faite qui facilite le travail du module symbolique qui va donc analyser des constituants plus petits en taille que la phrase initiale. Et par conséquent, l'analyse prend beaucoup moins de temps. En revanche, les phrases de longueurs 22 mots sont constituées d'un seul constituant au premier niveau syntaxique (sous la racine). Dans ce cas, la méthode hybride analyse un seul constituant regroupant la phrase entière et nous obtenons des temps d'exécution similaire avec la méthode symbolique seule.

La figure 4.21 nous montre l'évolution des temps d'exécution des analyses symbolique et hybride.

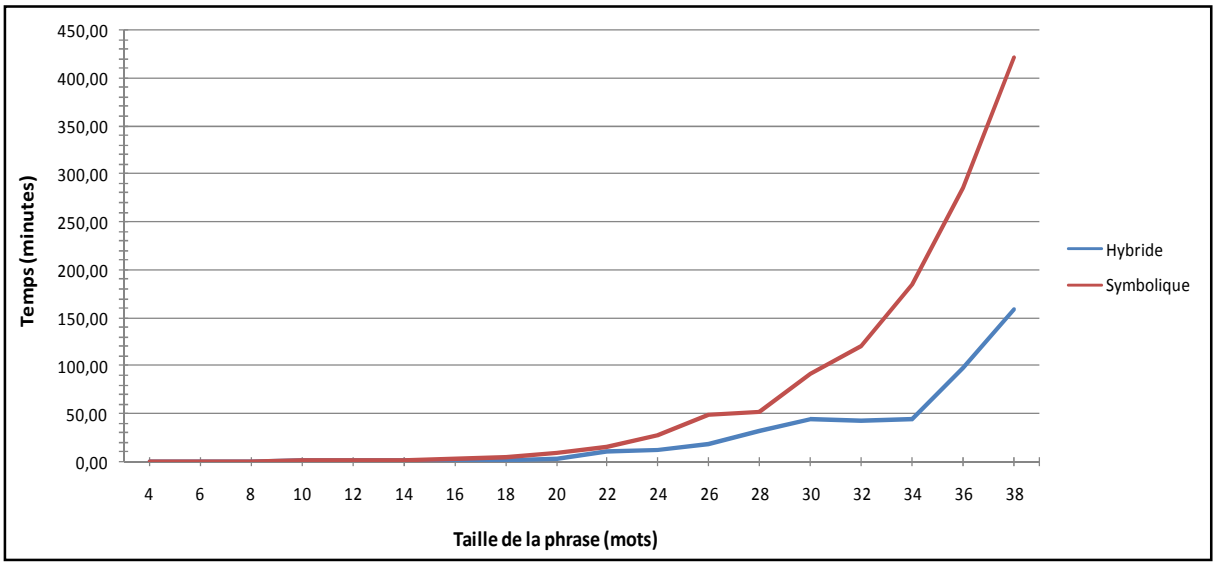



**Figure 4.21 :** Coubes d'évolution des temps d'analyses en fonction de la longueur des phrases de test

L'allure des courbes de la figure 4.6 permet de visualiser la différence de temps d'analyse entre les deux méthodes. En effet, Nous remarquons que les courbes sont presque superposées jusqu'a environ une longueur de 16 mots indiquant un temps d'analyse similaire entre les deux méthodes. Cette analogie est logique vu que le seuil que nous avons choisi dans notre méthode hybride est de 14 mots et au dessous de ce seuil, la méthode hybride et la méthode symbolique réalisent le même traitement d'analyse. En revanche, le temps d'analyse commence à diminuer significativement avec l'utilisation de la méthode hybride une fois qu'on a dépassé une taille de 17 mots. En effet nous avons mesuré une diminution moyenne du temps d'analyse des phrases dépassant 14 mots de 62,90% et une diminution sur l'ensemble des phrases de tests de 62,73%. Notre méthode hybride permet donc un gain important de temps d'analyse des phrases longues et imbriquées.

# 7. Conclusion

Ce chapitre a été consacré d'une part à la présentation des outils qui ont été développés et d'autre part à l'exposition des résultats de l'évaluation des différentes méthodes proposées.

Dans un premier temps nous avons présentés le corpus d'apprentissage utilisé dans le développement et l'évaluation des méthodes proposées à savoir, le PATB. Nous avons

ensuite présenté l'évaluation de la méthode d'analyse numérique qui permet d'identifier les constituants syntaxiques principaux de la phrase. Dans un deuxième temps, nous avons présenté l'outil que nous avons développé pour tester et évaluer notre méthode d'analyse syntaxique à base d'une grammaire symbolique. Enfin nous avons présenté l'évaluation de notre méthode hybride et nous avons montré l'apport de l'hybridation au processus d'analyse syntaxique par rapport à une analyse syntaxique symbolique. Les résultats obtenus sont encourageants et nous poussent à poursuivre l'amélioration de notre approche hybride.

# Conclusion et perspectives

L'importance de l'analyse syntaxique dans le processus de traitement automatique des langues naturelles, n'est plus à démontrer. En effet, cette analyse qui permet de déterminer les structures syntaxiques des phrases d'un texte, a été approchée de manières différentes selon les finalités qui lui sont assignées. Ainsi, plusieurs travaux de recherche ont été proposés pour une multitude de langues naturelles, cherchant d'une part, à améliorer la qualité des résultats attendus et d'autre part à optimiser le temps de calcul ainsi que les ressources utilisées.

L'étude des travaux antérieurs correspondant à cette tâche nous a permis de déceler qu'une approche à base de règles symboliques n'est pas à elle seule suffisante pour traiter une langue dotée d'une grande richesse structurale et ce avec des temps d'exécution raisonnables. De même une approche à base de modèles statistiques ne donne pas une description précise des structures profondes de la phrase.

La présentation de l'état de l'art et l'analyse critique des approches existantes nous ont permis de prendre conscience de la difficulté de la tâche traitée et de sa complexité. Outre ces difficultés, viennent s'ajouter d'autres contraintes liées aux spécificités de la langue arabe telle que l'agglutination des mots, l'ordre libre des constituants de la phrase, la non voyellation des mots, etc.

A l'issue de cette étude, nous avons opté  pour la réalisation d'un système d'analyse syntaxique de la langue arabe bénéficiant des avantages qu'offre une approche hybride numérique/symbolique. Néanmoins, il existe plusieurs stratégies d'hybridation possibles et le choix de l'une d'entre elles s'avère difficile. Dès lors, nous avons choisi de développer un analyseur symbolique et de se baser sur une étude critique de ses résultats pour pouvoir proposer une stratégie d'hybridation.

C'est dans l'objectif de préparer les processus (modules/composants) nécessaires à l'aboutissement de notre méthode hybride que nous avons donc proposé dans un premier temps, une méthode d'analyse syntaxique basée sur l'approche symbolique qui permet de donner une structure syntaxique profonde pour la phrase analysée. Cette méthode exploite une grammaire PCFG induite à partir d'un corpus annoté, le PATB. Le processus d'induction que nous avons proposé a le mérite d'être rapide et facile pour un non expert de la langue arabe. Cette méthode a été testée et validée sur un corpus de 1650 phrases et les résultats obtenus sont très encourageants (Précision : 83.59%, Rappel : 82.98%, F-score : 83.23%).

La confrontation des résultats obtenus par notre analyseur symbolique à ceux par l'analyseur numérique SPA (Barhoumi, Aloulou, Hadrich Belguith, & Zitouni, 2015) a montré que l'analyseur numérique donne de bons résultats dans d'analyse du premier niveau de l'arbre syntaxique. Elle a aussi révélé que l'analyseur symbolique donne de bon résultats avec des phrases relativement petites et qu'a partir d'un certains seuil, les valeurs de l'évaluation diminuent fortement et le temps d'analyse augmente sensiblement.

Suite à ces différentes constatations, nous avons proposé une méthode hybride qui repose d'une part sur une méthode à base d'apprentissage automatique pour l'identification des constituants syntaxiques principaux de la phrase, et d'autre par, sur une méthode à base de grammaire symbolique pour définir les structures syntaxiques détaillées des constituants syntaxiques principaux. L'originalité de notre méthode réside dans le fait qu'elle propose de faire, soit un traitement purement symbolique utilisant une grammaire arabe, soit de faire un traitement hybride alliant un processus numérique et un processus symbolique selon un seuil S représentant la taille de la phrase analysée.

La stratégie d'hybridation que nous avons proposée permet de faciliter la tâche à l'analyseur symbolique. En effet, celui-ci n'analyse plus toute la chaine en entrée mais analyse séparément les syntagmes identifiés par l'approche numérique. De cette manière nous profitons des performances accrues de l'approche symbolique dans l'analyse profonde des énoncés tout en gagnant en temps d'analyse.

La méthode d'analyse syntaxique basée sur l'approche numérique que nous avons proposée identifie les structures syntaxiques principales de la phrase. Concrètement, cette méthode utilise un modèle statistique appris en utilisant le paradigme des champs conditionnels aléatoires (les CRFs). Ce modèle se base sur les étiquettes

morphologiques et la corrélation entre les mots de la phrase pour prédire le plus fidèlement possible les frontières des constituants syntaxiques principaux de la phrase. Ce modèle a été testé selon la méthode de validation croisée et a donné des résultats probants (Exactitude : 97 %).

La méthodologie que nous avons suivie dans le développement de notre méthode hybride nous a permis de mettre en relief l'impact de l'hybridation sur le processus de l'analyse syntaxique. En effet l'hybridation, comparée à la méthode symbolique seule, permet d'améliorer non seulement les résultats en termes de précision, rappel et f-mesure mais aussi et surtout de diminuer significativement le temps d'analyse des phrases longues. Les résultats de l'évaluation des différentes méthodes proposées a mis en relief les performances de la méthode hybride et a montré expérimentalement l'intérêt de la collaboration entre des connaissances linguistiques et des modèles statistiques afin d'améliorer l'analyse syntaxique de l'arabe standard.

## Perspectives

Les travaux réalisés jusqu'à présent ont donné des résultats encourageants. Toutefois, ces résultats peuvent être améliorés et ouvrent la voix à des perspectives de recherche que nous esquissons dans les points suivants.

- Pour pouvoir traiter des phrases brutes en arabe, il est nécessaire d'intégrer un segmenteur de phrases tel que le segmenteur Star.

- Nous comptons aussi intégrer un analyseur morphologique qui aura une double fonction dans la méthode que nous proposons. La première fonction est d'annoter la phrase avant l'analyse numérique de celle-ci au deuxième cas de figure de notre proposition hybride. La deuxième fonction sera d'analyser les mots non reconnus par le lexique de la grammaire PCFG pour éviter les cas de blocage dû à la non reconnaissance des mots et permettre d'enrichir en même temps les règles lexicales par les nouvelles informations morphologiques fourni par l'analyseur morphologique.

- Les différentes étapes d'analyse de notre méthode hybride sont réalisées de façon séquentielle. Nous pensons en fait qu'une parallélisation des traitements peut réduire significativement le temps d'exécution.

- Nous envisageons aussi d'ajouter un outil de détection du type de la phrase qui va permettre d'orienter le processus d'analyse et améliorer par conséquent le temps d'analyse.
- L'intégration de modules supplémentaires permettant le traitement des phénomènes linguistiques comme l'anaphore ou l'ellipse.

# Bibliographie

Abeillé, A. (1993). *Les nouvelles syntaxes (grammaires d'unification et analyse du français.* Paris, France: Armand Collin.

Abeillé, A., & Blache, P. (2000). *Grammaires et analyseurs syntaxiques, Ingénierie des Langues* (éd. 1ère édition). (J.-M. Pierrel, Éd.) Paris: Hermes.

Abeillé, A., Clément, L., & Toussenel, F. (2003). Building a treebank for French. Dans A. Abeillé, L. Clément, & F. Toussenel, *Treebanks* (pp. 165-187). Springer Netherlands.

Abney, S. (1991). Parsing by chunks. Dans R. C. Berwick, S. Abney, & C. Tenny, *Principle-Based Parsing* (pp. 257-278). Kluwer Academic Publisher.

Aloulou, C. (2005). *Une approche multi-agent pour l'analyse de l'arabe : Modélisation de la syntaxe.* Thèse de doctorat, Ecole Nationale des Sciences de l'Informatique, Université de la Manouba, Tunisie.

Alqrainy, S., Muaidi, H., & Alkoffash, M. S. (2012). Context-Free Grammar Analysis for Arabic Sentences. *International Journal of Computer Applications , 53* (3), 7-11.

Al-Taani, A., Msallam, M., & Wedian, S. (2012). A Top-Down Chart Parser for Analyzing Arabic Sentences. *The International Arab Journal of Information Technology , 9* (3), 109-116.

Amini, M.-R. (2001). *Apprentissage Automatique et Recherche de l'Information : application à l'Extraction d'Information de surface et au Résumé de texte.* Thèse de doctorat, Université de Paris 06, Paris, France.

Attia, M. A. (2008). *Handling Arabic Morphological and Syntactic Ambiguity within the LFG Framework with a View to Machine Translation.* Thèse de doctorat, University of Manchester, Manchester, UK.

Aubin, S. (2002). *Grammaire de constituants ou grammaires de dépendances ? Quel type d'analyseur choisir pour un système d'extraction d'information.* Mémoire de DESS, INALCO-INRA.

Bangalore, S., Boulllier, P., Nasr, A., Rambow, O., & Sagot, B. (2009). MICA: a probabilistic dependency parser based on tree insertion grammars application note. *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, *Companion Volume: Short Papers*, pp. 185-188. Colorado, USA.

Barhoumi, A., Aloulou, C., Hadrich Belguith, L., & Zitouni, I. (2015). Analyse syntaxique statistique de la langue arabe. *Le deuxième colloque pour les Étudiants Chercheurs en Traitement Automatique du Langage Naturel et ses applications*, (pp. 31-39). Sousse, Tunisie.

Barzilay, R. (2010). Probabilistic approaches for modeling text structure and their application to text-to-text generation. *Empirical methods in natural language generation* (pp. 1-12). Springer Berlin Heidelberg.

Bataineh, B. M., & Bataineh, E. A. (2009). An efficient recursive transition network parser for Arabic language. *The World Congress on Engineering*, (pp. 1307- 1311). London, UK.

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics , 37* (6), 1554-1563.

Ben Fraj, F. (2010). *Un analyseur syntaxique pour les textes en langue Arabe à base d'un apprentissage à partir des patrons d'arbres syntaxiques.* Thèse de doctorat, Ecole Nationale des Sciences de l'Informatique, Université de la Manouba, Tunis.

Ben Fraj, F., Ben Othmane Zribi, C., & Ben ahmed, M. (2009). Quels attributs discriminants pour une analyse syntaxique par classification de textes en langue arabe ? *Conférence sur le Traitement Automatique des Langues Naturelles*, (pp. 1-8).

Ben Mohamed, M. A., Mallat, S., Nahdi, M. A., & Zrigui, M. (2015). Exploring the Potential of Schemes in Building NLP Tools for Arabic Language. *The International Arab Journal of Information Technology , 12* (6), 566-573.

Ben Mohamed, M. A., Mallat, S., Nahdi, M. A., & Zrigui, M. (2015). Exploring the Potential of Schemes in Building NLP Tools for Arabic Language. *The International Arab Journal of Information Technology , 12* (6), 566-573.

Ben Mohamed, M. A., Zrigui, S., Zouaghi, A., & Zrigui, M. (2015). N-scheme model: An approach towards reducing Arabic language sparseness. *5th International Conference on Information & Communication Technology and Accessibility* (pp. 1-5). Marrakech, Maroc: IEEE.

Ben Dbabis, S., Reguii, B., Ghorbel, H., & Hadrich Belguith, L. (2016). Utterance segmentation using Conditional Random Fields . *The 27th IBIMA conference on Innovation Management and Education Excellence Vision 2020.* Milan, Italie.

Bensalem Bahloul, R., Elkarwi, M., Haddar, K., & Blache, P. (2014). Building an Arabic Linguistic Resource from a Treebank: The Case of Property Grammar. *Text, Speech and Dialogue* (pp. pp 240-246). Brno, Czech Republic: Springer.

Berger, A. L., Della Pietra, V. J., & Della Pietra, S. A. (1996). A maximum entropy approach to natural language processing. *Computational linguistics , 22* (1), 39-71.

Besançon, R., De Chalendar, G., Ferret, O., Gara, F., Mesnard, O., Laïb, M., et al. (2010). LIMA : A Multilingual Framework for Linguistic Analysis and Linguistic Resources Development and Evaluation. *The seventh international conference on Language Resources and Evaluation*, (pp. 3697-3704). Malta.

Bikel, D. M. (2002). Design of a multi-lingual, parallel-processing statistical parsing engine. *The second international conference on Human Language Technology Research* (pp. 178-182). San Francisco, USA: Morgan Kaufmann Publishers Inc.

Blache, P. (2001). *Les Grammaires de Propriétés: des contraintes pour le traitement automatique des langues naturelles.* Hermes.

Blachère, R., & Gaudefroy-Demombynes, M. (1975). *Grammaire de l'arabe classique: morphologie et syntaxe.* Maisonneuve et Larose.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with cotraining. *The eleventh annual conference on Computational learning theory, COLT workshop* (pp. 92-100). Morgan Kaufmann.

Boltanski, J.-E. (2002). *La révolution chomskyenne et le langage.* L'Harmattan.

Boukedi, S., & Haddar, K. (2014). HPSG Grammar for Arabic Coordination Experimented with LKB System. *The Twenty-Seventh International Florida Artificial Intelligence Research Society Conference* (pp. 166-169). Pensacola, USA: Association for the Advancement of Artificial.

Boullier, P. (1999). *On TAG and Multicomponent TAG parsing.* Rapport de recherche, Unité de recherche INRIA Rocquencourt, Rocquencourt, France.

Candito, M., Benoît Crabbé, B., Denis, P., & Guérin, F. (2009). Analyse syntaxique du français : des constituants aux dépendances. *16e Conférence sur le Traitement Automatique des Langues Naturelles.* Senlis, France.

Carrère, C., & Masood, M. (2014). *Le poids économique des principaux espaces linguistiques dans le monde.* Global Studies Institute Université de Genève et Université de Genève. Geneve: fondation pour les études et recherches sur le développement international.

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. *The 43rd Annual Meeting of the ACL*, (pp. 173–180). Ann Arbor, USA.

Chomsky, N. (1965). *Aspects of the theory of syntax.* Cambridge: MIT Press.

Chomsky, N. (1957). *Syntactic Structures.* Mouton, la Hague.

Chomsky, N. (1997). *The Chomsky Hierarchy.* Cohen.

Clément, L., Gerdes, K., & Marlet, R. (2009). Grammaires d'erreur–correction grammaticale avec analyse profonde et proposition de corrections minimales. *16e conférence sur le Traitement Automatique des Langues Naturelles.*

*Dixel Dictionnaire.* (2010). Dictionnaires Le Robert.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM , 13* (2), 94-102.

Foth, K. A. (2006). *Hybrid Methods of Natural Language Analysis.* Thèse de doctorat, Université de Hambourg, Hambourg, Allemange.

Francopoulo, G. (2009). TagParser: combiner un corpus annoté avec un corpus brut. *The 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, (pp. 1-7). Suntec, Singapore.

Francopoulo, G. (2008). TagParser: well on the way to ISO-TC37 conformance. *The 1st First International Conference on Global Interoperability for Language Resources*, (pp. 22, 24). Hong-Kong, Chine.

Gazdar, G. (1985). *Generalized phrase structure grammar.* Harvard University Press.

Goldberg, Y., & Elhadad, M. (2010). An efficient algorithm for easy-first non-directional dependency parsing. *HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 742-750). Stroudsburg, PA, USA: ACL.

Goldman, J.-P., Laenzlinger, C., & Nebhi, K. (2010). Fipscolor: grammaire en couleur interactive pour l'apprentissage du français. *17ème conférence sur le Traitement Automatique des Langues Naturelles.* Montreal, Canada.

Graja, M. (2016). *Compréhension automatique de la parole en dialecte tunisien dans le cadre des systèmes de dialogue.* Rapport de Thèse, Faculté des Sciences Économiques et de Gestion de Sfax, Université de Sfax, Sfax, Tunisie.

Hadrich Belguith, L. (1999). *Traitement des erreurs d'accord de l'arabe basé sur une analyse syntagmatique étendue pour la vérification et une analyse multicritère pour la correction.* Thèse de doctorat, Faculté des Sciences de Tunis, Tunis.

Hadrich Belguith, L., & Chaaben, N. (2006). Analyse et désambiguïsation morphologiques de textes arabes non voyellés. *la 13ème conférence sur le Traitement Automatique des Langues Naturelles*, (pp. 493–501). Leuven, Belgique.

Hadrich Belguith, L., Aloulou, C., & Ben Hamadou, A. (2007). MASPAR : De la segmentation à l'analyse syntaxique de textes arabes. (CÉPADUÈS-Editions, Éd.) *Revue Information Interaction Intelligence* , 9-36.

Hammo, B., Moubaiddin, A., Obeid, N., & Tuffaha, A. (2014). Formal Description of Arabic Syntactic Structure in the Framework of the Government and Binding Theory. *Computacion y Sistemas , 18* (3), 611-625.

Harris, Z. S. (1962). *String analysis of sentence structure.* Mouton.

Huang, L., & Sagae, K. (2010). Dynamic programming for linear-time incremental parsing. *The 48th Annual Meeting of the Association for Computational Linguistics* (pp. 1077-1086). Uppsala, Sweden: Association for Computational Linguistics.

Jaf, S., & Ramsay, A. (2016). A Hybrid Approach to Parsing Natural Languages. Dans *Human Language Technology. Challenges for Computer Science and Linguistics* (pp. 136-145). Manchester, UK: Springer, Cham.

Jaynes, E. (1957). Information Theory and Statistical Mechanics. *Physical Review , 106* (2), 171–190.

Joshi, A. K., & Bangalore, S. (1994). Disambiguation of super parts of speech (or supertags): Almost parsing. *The 15th conference on Computational linguistics. 1*, pp. 154-160. Association for Computational Linguistics.

Kallmeyer, L., Maier, W., & Parmentier, Y. (2009). Un algorithme d'analyse de type earley pour grammaires à concaténation d'intervalles. *Conférence sur le Traitement Automatique des Langues Naturelles.*

Kaplan, R. M., & Bresnan, J. (1982). Lexical-functional grammar: A formal system for grammatical representation. Dans *Formal Issues in Lexical-Functional Grammar* (pp. 29-130). CSLI publications.

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2016). A Framework for Language Resource Construction and Syntactic Analysis: Case of Arabic. *17th International Conference on Intelligent Text Processing and Computational Linguistics (CICling 2016).* Konya, Turkie: Springer.

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2015). Arabic probabilistic context free induction from a Treebank. *Research in Computing Science , 90.*

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2014). Chunking Arabic Texts Using Conditional Random Fields. *The 11th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2014)* (pp. 428-432). Doha, Qatar: IEEE-CPS.

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2015). Enhacing CRF Model with N-grams for Arabic Texts Chunking. *The 25th International Business Information Management Conference (IBIMA 2015)*, (pp. 2877-2884). Amsterdam, Hollande.

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2016). Parsing Arabic using induced probabilistic context free grammar. *International Journal of Speech Technology , 19* (2), 313-323.

Khoufi, N., Aloulou, C., & Hadrich Belguith, L. (2016). Toward Hybrid Method for Parsing Modern Standard Arabic. *17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing.* Shanghai, Chine.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *The Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1137–1143). San Mateo, Californie, USA: Morgan Kaufmann.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *The eighteenth international conference on machine learning*, *1*, pp. 282-289.

*Le Trésor de la langue française informatisé.* (1971-1994). Consulté le Mai 30, 2016, sur Le Trésor de la langue française informatisé: http://atilf.atilf.fr/

Maamouri, M., Bies, A., & Kulick, S. (2008). Enhancing the Arabic Treebank: a Collaborative Effort toward New Annotation Guidelines. *The sixth international conference on Language Resources and Evaluation.* Marrakech, Maroc.

Maamouri, M., Bies, A., Buckwalter, T., & Mekki, W. (2004). The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. *NEMLAR International Conference on Arabic Language Resources and Tools* (pp. 102 – 109). Egypte: ELDA.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing* (Vol. 999). Cambridge, USA: MIT Press.

Marcoux, R. (2012). *Aperçu sur quelques espaces linguistiques dans le monde.* Consulté le 05 30, 2016, sur l'Observatoire démographique et statistique de l'espace francophone: file:///C:/Users/Nabil/Desktop/espaces_linguistiques.pdf

Marton, Y., Habash, N., Rambow, O., & Alkuhlani, S. (2013). SPMRL'13 Shared Task System: The CADIM Arabic Dependency Parser. *The Fourth Workshop on Statistical Parsing of Morphologically Rich Languages* (pp. 86–90). Seattle, Washington, USA: ACL.

McCallum, A., & Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. Dans A. f. Linguistics (Éd.), *The seventh conference on Natural language learning at HLT-NAACL 2003*, *4*, pp. 188-191.

Mel′čuk, I. A. (1988). *Dependency syntax: theory and practice.* SUNY press.

Mi, H., & Huang, L. (2015). Shift-reduce constituency parsing with dynamic programming and pos tag lattice. *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL* (pp. 1030–1035). Denver, Colorado, USA: Association for Computational Linguistics.

Mitchell, T. (1997). *Machine Learning.* McGraw-Hill.

Mouelhi, Z. (2008). AraSeg : un segmenteur semi-automatique des textes arabes. *9ème Journées internationales d'Analyse statistique des Données Textuelles.* Lyon, France.

Nivre, J., Hall, J., & Nilsson, J. (2006). MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. *The fifth international conference on Language Resources and Evaluation*, (pp. 2216-2219). Genoa, Italy.

O'Donnell, M. (1994). *Sentence analysis and generation: a systemic perspective.* Thèse de doctorat, University of Sydney, Sydney, Australie.

Okazaki, N. (2007). *CRFsuite: a fast implementation of Conditional Random (CRFs).* Récupéré sur Naoaki Okazaki's website: url=http://www.chokkan.org/software/crfsuite/

Osório, F. S. (1998). *Un système hybride neuro-symbolique pour l'apprentissage automatique constructif.* Thèse de doctorat, L'Institut National Polytechnique de Grenoble, Grenoble, France.

Ouersighni, R. (2014). Robust Rule-based Approach in Arabic Processing. *International Journal of Computer Applications , 93* (12), 31-37.

Petit, M., & Christiansen, H. (2009). Un calcul de Viterbi pour un Modele de Markov Caché Contraint. *5ème Journée Francophone de Programmation par Contraintes.*

Pollard, C., & Sag, I. A. (1988). *Information-based syntax and semantics: Vol. 1: fundamentals.* Stanford, USA: Center for the Study of Language and Information.

Pollard, C., & Sag, I. (1994). *Head-driven Phrase Structure Grammar.* Chicago: CLSI series.

Pradet, Q. (2011). *Analyse morphosyntaxique avec SVM dans une chaîne de traitement linguistique.* Rapport de stage, Laboratoire d'Ingénierie et de Vision des contenus.

Pullum, G. K., & Scholz, B. C. (2001). On the distinction between model-theoretic and generative-enumerative syntactic frameworks. Dans *Logical aspects of computational linguistics* (pp. 17-43). Springer Berlin Heidelberg.

Russell, S. J., & Norvig, P. (2003). *Artificial Intelligence: A Modern Approach* (éd. 2nd Edition). Prentice-Hall.

Schmid, H. (2004). Efficient parsing of highly ambiguous context-free grammars with bit vectors. *The 20th international conference on Computational Linguistics* (p. 162). Association for Computational Linguistics.

Sennrich, R., Schneider, G., Volk, M., & Martin, W. (2009). A New Hybrid Dependency Parser for German. *Proceedings of the German Society for Computational Linguistics and Language Technology* (pp. 115-124). Allemagne: GSCL.

Sewell, M. (2005). *Support Vector Machines*. Consulté le 2016, sur www.svms.orgs

Sha, F., & Pereira, F. (2003). Shallow parsing with conditional random fields. *the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, *1*, pp. 134-141.

Sigogne, A. (2012). *Integration de ressources lexicales riches dans un analyseur syntaxique probabiliste.* Rapport de thèse, Université Paris-Est, Laboratoire d'Informatique Gaspard Monge, Paris, France.

Sigogne, A., Constant, M., & Laporte, E. (2014). Intégration des données d'un lexique syntaxique dans un analyseur syntaxique probabiliste. Dans *Penser le Lexique-Grammaire. Perspectives actuelles* (pp. 505-516). Fryni Kakoyianni-Doa.

Simon, H. A. (1983). Why should machines learn? Dans H. A. Simon, *Machine learning* (pp. 25-37). Berlin Heidelberg: Springer.

Spriet, T., & El-Bèze, M. (1999). Introduction of rules into a stochastic approach for language modelling. Dans *Computational Models of Speech Pattern Processing* (pp. 350-355). Springer Berlin Heidelberg.

Taku, K., & Matsumoto, Y. (2001). Chunking with Support Vector Machines. *the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies* (pp. 1-8). Association for Computational Linguistics.

Tesnière, L. (1956). *Eléments de syntaxe structurale.* Librairie Klincksiek 1959.

Thompson, H. (1992). Parseval workshop. *ELSnews , 1 (2)* . (L. Cahill, Éd.) Brighton, UK, University of Sussex : School of Cognitive and Computing Sciences.

Tounsi, L., & Van Genabith, J. (2010). Arabic Parsing Using Grammar. *the International conference on Language Resources and Evaluation.* Valletta, Malta.

Tsuruoka, Y., Tsujii, J., & Ananiadou, S. (2009). Fast full parsing by linear-chain conditional random fields. Dans A. f. Linguistics (Éd.), *the 12th Conference of the European Chapter of the Association for Computational Linguistics*, (pp. 790-798).

Urieli, A. (2013). *Robust French syntax analysis: reconciling statistical methods and linguistic knowledge in the Talismane toolkit.* Thèse de doctorat, Université Toulouse 2 Le Mirail, Toulouse, France.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on 13.2* , 260-269.

Wehrli, E. (2007). Fips, a "Deep" Linguistic Multilingual Parser. *The Workshop on Deep Linguistic Processing* (pp. 120–127). Ann Arbor, USA: ACL.

Wehrli, E. (1997). *L'analyse syntaxique des langues naturelles, problèmes et méthodes.* Paris: Masson.

Wehrli, E., & Nerima, L. (2015). The Fips Multilingual Parser. Dans *Language Production, Cognition, and the Lexicon* (pp. 473-490). Springer International Publishing.

Younger, D. H., & Kasami, T. (1978). The Cocke-Kasami-Younger Algorithm. Dans M. A. Harrison, *Introduction to Formal Language Theory* (pp. 430-442). Reading, Massachusetts, USA: Addison-Wesley.

# Annexe A

**Extrait de la grammaire PCFG induite à partir du corpus PATB**

| Règles | Probabilités |
|---|---|
| ADJP -> DET_ADJ | [0.26570109] |
| ADJP -> DET_ADJ PP | [0.07643622] |
| ADJP -> ADJ | [0.12779942] |
| ADJP -> DET_ADJ NP | [0.01703992] |
| ADJP -> ADJ_COMP NP | [0.04771178] |
| ADJP -> PUNC ADJ NOUN PUNC | [0.00060857] |
| ADJP -> ADJ NP | [0.03347128] |
| ADJP -> DET_ADJ_NUM PP PP | [0.00170399] |
| ADJP -> ADJ CONJ ADJ | [0.03554041] |
| ADJP -> PUNC ADJ PUNC | [0.00401655] |
| ADJP -> DET_ADJ CONJ DET_ADJ CONJ DET_ADJ | [0.00827653] |
| ADJP -> DET_ADJ DET_ADJ DET_ADJ | [0.00048685] |
| ADJP -> ADJ PP | [0.09104187] |
| ADJP -> DET_ADJ DET_ADJ | [0.01643135] |
| ADJP -> ADJ_COMP PP | [0.01740506] |
| ADJP -> PUNC ADJ ADJ PUNC | [0.00036514] |
| ADJP -> ADJ PUNC ADJ | [0.00304284] |
| ADJP -> DET_ADJ CONJ DET_ADJ | [0.06292599] |
| ADJP -> DET_ADJ_NUM PP | [0.00413827] |
| ADJP -> DET_ADJ_COMP | [0.00389484] |
| ADVP -> PRT ADV | [0.05401929] |
| ADVP -> TYPO TYPO | [0.00064309] |
| ADVP -> NOUN_NUM | [0.21479100] |
| ADVP -> NOUN_NUM PUNC NOUN_NUM | [0.05337621] |

| | |
|---|---|
| ADVP -> ADV CONJ ADV | [0.00385852] |
| ADVP -> ADVP PUNC S | [0.00064309] |
| ADVP -> ADVP ADVP ADVP ADVP | [0.00064309] |
| ADVP -> ADVP ADVP ADVP | [0.00192926] |
| ADVP -> PUNC NOUN_NUM PUNC | [0.03665595] |
| ADVP -> NOUN_NUM PUNC PUNC NOUN_NUM | [0.00257235] |
| ADVP -> PUNC ADVP ADVP ADVP ADVP ADVP PUNC | [0.00064309] |
| ADVP -> TYPO ADV | [0.00064309] |
| ADVP -> NOUN_NUM NP | [0.00128617] |
| ADVP -> NOUN_NUM PUNC NOUN_NUM NOUN_NUM PUNC NOUN_NUM PUNC | [0.00128617] |
| ADVP -> NOUN_NUM PUNC | [0.00064309] |
| ADVP -> ADVP ADVP | [0.00064309] |
| ADVP -> ADVP CONJ ADVP | [0.00064309] |
| ADVP -> ADV | [0.61286175] |
| ADVP -> TYPO | [0.00064309] |
| ADVP -> NOUN_NUM PUNC PUNC NOUN_NUM NOUN_NUM PUNC NOUN_NUM | [0.00321543] |
| NP -> NOUN_PROP DET_ADJ | [0.00123347] |
| NP -> ADJ DET_NOUN | [0.00002535] |
| NP -> DET_NOUN_PROP | [0.00933549] |
| NP -> PRON NP | [0.00011405] |
| NP -> NP PUNC PUNC NP | [0.00002112] |
| NP -> NOUN NP PUNC PRN | [0.00002112] |
| NP -> NOUN_NUM PUNC NOUN_NUM | [0.00033794] |
| NP -> NOUN_QUANT NP ADJ | [0.00000422] |
| NP -> NP CONJ NP CONJ S | [0.00003802] |
| NP -> ABBREV PUNC ABBREV PUNC ABBREV | [0.00001690] |
| NP -> PRT PRON | [0.00019854] |
| NP -> NOUN NP PUNC | [0.00006336] |
| NP -> PRT NP ADJP | [0.00000422] |
| NP -> DET_ABBREV NOUN_NUM | [0.00000422] |
| NP -> NP CONJ NP CONJ NP CONJ NP CONJ NP CONJ NP | [0.00007604] |

| | |
|---|---|
| NP -> NOUN_PROP NOUN | [0.00004647] |
| NP -> DET_NOUN ADJ | [0.00008026] |
| NP -> NP NP NP | [0.00022811] |
| NP -> NP PUNC PP PP | [0.00001690] |
| NP -> PUNC NP PUNC CONJ PUNC NP PUNC | [0.00000845] |
| PP -> NP PP | [0.00009849] |
| PP -> PREP PP | [0.00007879] |
| PP -> PREP NP PRN | [0.00003940] |
| PP -> PP PUNC CONJ PP CONJ NP | [0.00003940] |
| PP -> PRT PREP S | [0.00003940] |
| PP -> PREP PP PUNC PP PUNC | [0.00001970] |
| PP -> PP PUNC PP PUNC | [0.00001970] |
| PP -> ADJP PREP NP | [0.00001970] |
| PP -> PUNC PP CONJ PP PUNC | [0.00001970] |
| PP -> PP PUNC CONJ PP PUNC | [0.00003940] |
| PP -> PP PUNC CONJ PP PUNC CONJ PP | [0.00009849] |
| PP -> PP CONJP PP CONJP PP | [0.00001970] |
| PP -> PREP NP CONJ PP | [0.00003940] |
| PP -> NP PUNC PREP NP | [0.00001970] |
| PP -> PP CONJ PP CONJ PP | [0.00035457] |
| PP -> PREP NP PUNC PP | [0.00001970] |
| PP -> PREP NP PUNC PUNC | [0.00001970] |
| PP -> PUNC PREP S PUNC | [0.00003940] |
| PP -> PP PUNC PP | [0.00045306] |
| PP -> PREP NP SBAR | [0.00009849] |
| PRN -> PUNC NP | [0.00595238] |
| PRN -> PUNC ABBREV PUNC | [0.00297619] |
| PRN -> PUNC ADJP | [0.00297619] |
| PRN -> PUNC CONJ ADVP NP PUNC | [0.00297619] |
| PRN -> PUNC VP PUNC PUNC | [0.00297619] |
| PRN -> CONJ NP | [0.01488095] |
| PRN -> CONJ PP PP | [0.00297619] |
| PRN -> PUNC PUNC DET_NOUN_PROP PUNC PUNC | [0.00297619] |
| PRN -> PRON | [0.00297619] |

| | |
|---|---|
| PRN -> CONJ PP | [0.02678571] |
| PRN -> CONJ FRAG | [0.00595238] |
| PRN -> PUNC NP PUNC PUNC | [0.00297619] |
| PRN -> CONJ VP | [0.00595238] |
| PRN -> PUNC PUNC | [0.00297619] |
| PRN -> NP | [0.05952381] |
| PRN -> NOUN | [0.00892857] |
| PRN -> PUNC NP NP S PUNC | [0.00297619] |
| PRN -> PUNC CONJ S | [0.00595238] |
| PRN -> PUNC PUNC PUNC PUNC PUNC | [0.00892857] |
| PRN -> CONJ SBAR | [0.01190476] |
| S -> NP NP PUNC VP | [0.00015542] |
| S -> NP PUNC NP PUNC NP VP PUNC | [0.00002220] |
| S -> NP PUNC SBAR PUNC PRT VP | [0.00004441] |
| S -> PP NP PUNC PP | [0.00006661] |
| S -> PP NP PUNC PP PUNC NP | [0.00004441] |
| S -> NP PP NP | [0.00108795] |
| S -> NP PUNC VP PUNC PUNC | [0.00004441] |
| S -> S CONJ S PUNC | [0.01412109] |
| S -> PUNC ADV NP PUNC S PUNC | [0.00002220] |
| S -> PP PUNC PRT VP PUNC | [0.00022203] |
| S -> CONJ NP VP PUNC | [0.00546193] |
| S -> CONJ NP NP | [0.00066609] |
| S -> PUNC S CONJP S CONJ S PUNC | [0.00002220] |
| S -> NP ADJP PP | [0.00146540] |
| S -> CONJ SBAR PUNC VP PUNC | [0.00062168] |
| S -> CONJ VP | [0.02031573] |
| S -> NP PP NP PUNC | [0.00002220] |
| S -> ADVP NP NP | [0.00013322] |
| S -> NP ADJP NP PP | [0.00008881] |
| S -> NP PP PP | [0.00088812] |
| VP -> IV PP NP NP | [0.00188573] |
| VP -> PV NP NP PP NP PP | [0.00051184] |
| VP -> PSEUDO_VERB NP PUNC SBAR | [0.00005388] |

| | |
|---|---|
| VP -> PUNC CV NP S PUNC | [0.00005388] |
| VP -> PV NP FRAG | [0.00013469] |
| VP -> PRT IV NP NP S | [0.00043102] |
| VP -> PV NP NP S PP PUNC S | [0.00002694] |
| VP -> PRT IV NP VP | [0.00016163] |
| VP -> PV NP PUNC PP PUNC PP | [0.00021551] |
| VP -> ADJ_VN NP NP NP NP | [0.00005388] |
| VP -> PRT IV NP NP PP UCP | [0.00002694] |
| VP -> IV NP NAC NP PUNC NAC PUNC SBAR | [0.00005388] |
| VP -> PV NP NP PP NP PP PUNC S | [0.00002694] |
| VP -> PRT PRT IV NP UCP | [0.00005388] |
| VP -> PRT IV NP NP PUNC SBARQ | [0.00002694] |
| VP -> PRT IV NP NP PP NP PP | [0.00010776] |
| VP -> PRT TYPO NP SBAR | [0.00002694] |
| VP -> PV NP ADVP NP NP PP | [0.00002694] |
| VP -> IV NP NP PP PP ADVP | [0.00002694] |
| VP -> NOUN_VN NP NP S PP | [0.00005388] |

# Annexe B

**Liste des annotations utilisées pour l'induction de la grammaire PCFG arabe**

| **Annotation** | **Explication** |
|---|---|
| ABBREV | Abréviation |
| ADJ | Adjectif |
| ADJ.VN | Adjectif Verbal |
| ADJ_COMP | Adjectif Comparatif |
| ADJ_NUM | Adjectif numérique |
| ADJP | Adjectif phrase |
| ADV | Adverbe |
| ADVP | Adverbe Phrase |
| CONJ | Conjonction |
| CONJP | Conjonction phrase |
| CONNEC_PART | Adjectif Comparatif |
| CV | Command Verbe |
| CVSUFF | suffixe du Command Verbe |
| DEM_PRON | Pronom Démonstratif |
| DET | Déterminant |
| DET+ABBREV | Déterminant + Abréviation |
| DET+ADJ | Déterminant + Adjectif |
| DET+ADJ.VN | Déterminant + Adjectif Verbal |
| DET+ADJ_COMP | Déterminant + Adjectif comparatif |
| DET+DIALECT | Déterminant + Dialecte |
| DET+FOREIGN | Déterminant + mot étranger |
| DET+NOUN | Déterminant + Nom |
| DET+NOUN.VN | Déterminant + Nom verbal |

| | |
|---|---|
| DET+NOUN_PROP | Déterminant + Nom Propre |
| DET+NOUN_QUANT | Déterminant + Noun Quantifier |
| DIALECT | Dialecte |
| EMPHATIC_PART | Particule emphatique |
| FOCUS_PART | Focus Particule |
| FOREIGN | Mot étrangère |
| FRAG | Fragment |
| FUT_PART | Particule de Futur |
| GRAMMAR_PROBLEM | Problème de grammaire |
| INTERJ | Interjection |
| INTERROG_ADV | Adverbe Interrogatif |
| INTERROG_PART | Particule Interrogatif |
| INTERROG_PRON | Pronom Interrogatif |
| INTJ | interjection |
| IV | Verbe au présent |
| IV_PASS | Forme passif du Verbe au présent |
| IVSUFF | Suffixe du Verbe au présent |
| JUS_PART | Particule jussif |
| LATIN | Mot Latin |
| LST | Liste de marqueur |
| NAC | N'est pas un Constituant |
| NEG_PART | Particule Négative |
| NOUN | Nom |
| NOUN.VN | Nom Verbal |
| NOUN_NUM | Nombre Cardinal |
| NOUN_PROP | Nom Propre |
| NOUN_QUANT | Nom Quantitatif |
| NP | Groupe nominal |
| Paragraph | Paragraphe |
| PART | Partie |
| POSS_PRON | Pronom Possessif |
| PP | Groupe Prépositionnel |
| PREP | Préposition |
| PRN | Parenthetical |

| | |
|---|---|
| PRON | Pronom Personnel |
| PRT | Particules |
| PSEUDO_VERB | Pseudo Verbe |
| PUNC | Ponctuation |
| PV | Verbe au passé |
| PV_PASS | Forme passif du Verbe au passé |
| PVSUFF | Suffixe du Verbe au passé |
| RC_PART | Particule Conditionnel de Réponse |
| REL_ADV | Adverbe Relatif |
| REL_PRON | Pronom Relatif |
| RESTRIC_PART | Particule Restrictif |
| S | Phrase |
| SBAR | Clauses relatives |
| SBARQ | Clauses relatives interrogatives |
| SQ | Phrase interrogative |
| SUB_CONJ | Conjonction de Subordination |
| TYPO | Typo |
| UCP | Unlike Coordinated Phrase |
| VERB | Verbe |
| VERB_PART | Particule d'un Verbe |
| VOC_PART | Particule Vocative |
| VP | Groupe Verbal |
| WHADVP | Wh- Adverbe Phrase |
| WHNP | Wh-non Phrase |
| WHPP | Wh-prépositionnel Phrase. |
| X | autre |

# Liste des publications

## Revues internationales avec comité de lecture

- **IJST 2016 (Indexée par Scopus et ACM DL et DBLP)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2016)
  Parsing Arabic using induced probabilistic context free grammar,
  *International Journal of Speech Technology-Springer, vol. 19( 2)* , *pp 313-323.*

- **RCS 2015 (Indexée par DBLP):**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2015)
  Arabic Probabilistic Context Free Grammar Induction from a Treebank
  *Research in Computing Science, vol 90, pp* 77-86.

## Conférences ou workshops internationaux avec comités de lecture

- **SNPD 2016 (Classe C, indexée par DBLP, ISI, EI, IEEE)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2015)
  Toward Hybrid Method for Parsing Modern Standard Arabic,
  *In 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, May 30 - June 1, 2016, Shanghai, China.

- **CICLing 2016 (Classe B, indexée par DBLP, ISI, EI)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2015)
  A Framework for Language Resource Construction and Syntactic Analysis: Case of Arabic,
  *In 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICling 2015)*, Springer LNCS, April, 2016, Konya, Turkey.

- **IBIMA 2015 (Classe B, indexée par Scopus et Web of Science)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2015)
  Enhacing CRF Model with N-grams for Arabic Texts Chunking,
  *In the Proceedings of 25th International Business Information Management Conference (IBIMA 2015)*, 7-8 May 2015, Amsterdam, Netherlands, pp2877-2884 .

- **CICLing 2015 (Classe B, indexée par DBLP, ISI, EI)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2015)
  Arabic Probabilistic Context Free Grammar Induction from a Treebank,
  *In 16th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2015*), selected for publication in the RCS journal, April 14–20, 2015, Cairo, Egypt.

- **AICCSA 2014 (Classe C, indexée par DBLP, IEEE Xplore DL et Web of Science)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2014)
  Chunking Arabic Texts Using Conditional Random Fields,
  *In the Proceedings of The 11th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2014)*, Doha, Qatar.

- **RANLP 2013 (Classe C, indexée par DBLP, ACL anthology)**
  Khoufi N., Boudokhane M. (2013)
  Statistical-based System for Morphological Annotation of Arabic Texts,
  *In Proceedings of the Recent Advances in Natural Language Processing (RANLP 2013)*, Hissar, Bulgaria, pp100-106.

- **NLPCS 2013 (Indexée par DBLP)**
  Khoufi N., Louati S., Aloulou C., Hadrich Belguith L.(2013)
  Supervised learning model for parsing Arabic language,
  *In Proceedings of the 10th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2013)*, Marseille, France, pp129-136.

- **ACIT 2013 (conférence internationale)**
  Khoufi N., Aloulou C., Hadrich Belguith L. (2013)
  ARSYPAR: A tool for parsing the Arabic language based on supervised learning,
  *In the 14th International Arabic Conference on Information Technology (ACIT 2013)*, Khartoum, Sudan.

**Résumé** : L'analyse syntaxique est une tâche importante pour les applications du TALN tel que les systèmes de questions réponses, l'analyse d'opinion, ou la traduction automatique. C'est une tâche complexe spécialement pour la langue arabe à cause de ses spécificités telle que l'agglutination et la non voyellation. Dans ce contexte, nous proposons une méthode hybride pour l'analyse syntaxique de phrases en langue arabe. L'objectif de cette méthode est de tirer profit des deux approches numérique et symbolique. Ainsi dans cette thèse nous proposons une méthode numérique, basée sur l'apprentissage automatique, qui permet d'identifier les constituants syntaxiques principaux de la phrase. Cette méthode s'enchaine avec une méthode symbolique basée sur une grammaire arabe pour produire des structures syntaxiques détaillées à chaque constituant précédemment reconnu. La structure complète de la phrase est composée de l'ensemble des structures des différents constituants.

Les méthodes proposées ont été évaluées et ont donnés des résultats encourageants en termes de performances et de temps d'exécution. L'évaluation a permit de mettre en relief l'apport de l'hybridation et de prouver ainsi son efficacité.

**Mots clés :** Traitement automatique de la langue arabe**,** Analyse syntaxique, approche symbolique, approche numérique, approche hybride, grammaire, apprentissage automatique

**Abstract:** Parsing is an important task for NLP applications such as question answering systems, opinion mining, or machine translation. This is a complex task especially for Arabic language because of its specific features such as agglutination and unvocalization. This thesis belongs to this context and proposes a hybrid method for parsing Arabic sentences. The goal of this method is to take advantage of both, numeric and symbolic approaches. So in this thesis, we propose a numerical method based on machine learning, which identifies the main syntactic constituents of the sentence. We also propose a symbolic method based on an Arabic grammar to produce detailed syntactic structures for each constituent already recognized. The final complete structure of the sentence is composed by the combination of all generated syntactic structures.

The proposed methods were evaluated and have shown encouraging results. The evaluation highlights the contribution of hybridisation and thus proves its effectiveness.

**Keywords:** Automatic processing of the Arabic language, parsing, symbolic approach, numerical approach, hybrid approach, grammar, machine learning

**التلخيص:** التحليل النحوي مرحلة هامة لتطبيقات البرمجة اللغوية مثل نظم طرح الأسئلة والإجابة، وتحليل الرأي، أو الترجمة الآلية. هذه المهمة معقدة بصفة خاصة للغة العربية لما لها من خصائص مثل تلاصق الحروف وعدم الشكل. في هذا السياق، نقترح في هذه الأطروحة طريقة هجينة لتحليل الجمل في اللغة العربية. الهدف من هذه الطريقة هو الاستفادة من كلا المنهجيتين الرقمية والرمزية. في هذه الأطروحة اقترحنا أولاً طريقة رقمية على أساس تقنية التعلم الآلي، والذي يحدد المكونات النحوية الرئيسية للجملة. ثم اقترحنا ثانياً طريقة رمزية تستعمل قواعد اللغة العربية لإنتاج الهياكل النحوية المفصلة لكل مكون حددته الطريقة الرقمية الأولى. يتم تشكيل الهيكل النحوي الكامل للجملة بتجميع كل الهياكل لكل التراكيب معاً.

تظهر نتائج تقييم الطرق المقترحة في هذه الأطروحة أن الطريقة الهجينة المقترحة للتحليل النحوي هي طريقة واعدة و فعالة.

**الكلمات الجوهرية :** المعالجة الآلية للغة العربية، التحليل النحوي، المنهج رمزي، المنهج الرقمي، المنهج الهجين، التعلم الآلي