UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

# Grammaire Catégorielle Combinatoire Applicative (GCCA) pour l'Analyse de la Langue Arabe : Une Étude des Constructions Inna wa Akhawatoha et Kana wa Akhawatoha

MÉMOIRE PRÉSENTÉ

COMME EXIGENCE PARTIELLE DE LA

MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUES APPLIQUÉES

PAR

**SABRINA TAIBOUNI**

Décembre 2025

CE PROJET A ÉTÉ ÉVALUÉ
PAR UN JURY COMPOSÉ DE :

M. Ismail Biskri directeur de projet
Département de mathématiques et informatique à l'Université du Québec à Trois-Rivières

Mme. Nadia Ghazzali, professeur
Département de mathématiques et informatique à l'Université du Québec à Trois-Rivières

M. Boucif Amar Bensaber, professeur
Département de mathématiques et informatique à l'Université du Québec à Trois-Rivières

## REMERCIEMENTS

Je commence tout d'abord par remercier le Dieu, pour m'avoir accordé la santé, la patience et la force de mener à bien ce travail. Sans Dieu, rien n'était possible.

Je tiens à exprimer ma plus profonde et sincère reconnaissance à ma chère mère, et mon cher père. Maman, la femme la plus exceptionnelle que je connaisse. tu es mon pilier, ma lumière et ma plus grande source d'inspiration. Depuis toujours, tu m'as entourée d'un amour inconditionnel, d'une tendresse sans limites et d'un soutien indéfectible. Tu as sacrifié tant de choses pour que je puisse poursuivre mes études dans les meilleures conditions. Tes prières quotidiennes, tes encouragements constants et ton dévouement ont été le moteur de ma réussite.
Ce mémoire est aussi le tien, maman, car sans toi, rien n'aurait été possible. Du fond du cœur, merci pour tout ce que tu as fait et continues de faire pour moi. Qu'Allah te bénisse, te garde en bonne santé et te récompense pour tout ton amour et tes sacrifices.

À mon père, je dis un immense merci pour son soutien moral, ses encouragements et ses sages conseils. Et surtout, tes prières quotidiennes pour que je réussisse, ton encouragement dans les moments les plus difficiles, m'ont donné confiance en moi de poursuivre mon parcours. Merci papa.

Je tiens également à remercier très sincèrement mon mari Radhouane, pour sa compréhension, sa patience et son appui indéfectible tout au long de mes études. Merci d'avoir été à mes côtés, de m'avoir encouragée et soutenue, même dans les périodes de doute et de fatigue. Ta présence a été essentielle dans l'accomplissement de ce projet.

J'adresse aussi toute ma reconnaissance à mes frères, Fateh Eddine et Islam Zakaria, pour leur affection, leurs encouragements et leur présence constante. Leur soutien et leurs mots d'encouragement ont été pour moi d'un grand réconfort.

Un remerciement tout particulier, empreint d'amour et de tendresse, va à mon fils, mon précieux trésor, Mohamed Rayan. Mon cher bébé, même si tu es encore tout petit pour comprendre l'importance de moment, c'est toi qui as été ma plus grande source de motivation. Ton sourire, ta douceur et ta présence lumineuse ont apaisé mes journées les plus stressantes et m 'ont donné le courage de continuer, de persévérer et de rêver d'un avenir meilleur pour toi. malgré la fatigue ta simple présence m'a rappelé chaque jour pourquoi il était important d'aller jusqu'au bout. Merci, mon petit ange, d'avoir illuminé ma vie et de m'avoir donné la force de dépasser mes limites.

Je tiens aussi à remercier mon encadreur Ismaïl Biskri pour ses conseils avisés, ses remarques constructives et sa rigueur scientifique aussi l'ensemble des enseignants pour la qualité de leurs enseignements, leur dévouement et leur passion. Grâce à eux, j'ai pu acquérir des connaissances solides et une formation qui me serviront tout au long de ma carrière.

Enfin, j'adresse mes remerciements à toutes les personnes qui, de près ou de loin, ont contribué à la réussite de ce travail : mes amis, mes collègues de promotion, et tous ceux qui m'ont encouragée, soutenue et motivée tout au long de ce parcours. Merci à toutes et à tous.

# Table de matières

# Table des Figures

# Résumé

Ce travail de recherche a but d'étudier l'application de la Grammaire Catégorielle Combinatoire Applicative (GCCA) dans le cadre de l'analyse grammaticale de la langue arabe. L'idée principale est de voir comment ce modèle peut représenter la structure des phrases arabes, plus précisément celles qui contiennent les particules d'Inna wa Akhawatoha et Kana wa Akhawatoha et qui posent souvent des difficultés dans le traitement automatique.

Une partie importante du travail a été consacrée à l'implémentation informatique du modèle proposé. L'objectif était de vérifier si la GCCA pouvait réellement décrire les structures d'Inna wa Akhawatoha et Kana wa Akhawatoha d'une manière claire et cohérente à travers des expériences pratiques.

Les résultats obtenus montrent que cette approche est prometteuse et qu'elle peut améliorer les techniques et les outils de traitement automatique de l'arabe, tout en ouvrant la voie à des recherches plus approfondies dans ce domaine.

# Chapitre 01: Introduction Générale

Nous vivons dans l'ère numérique qui a profondément modifié notre manière d'accéder et de traiter l'information. ce changement digitale a favorisé la naissance de nouveaux domaines d'étude dont le Traitement Automatique des Langues (TAL). Ce dernier qui cherche à doter les machines de la capacité à comprendre et produire le langage humain, s'établit désormais comme un pilier essentiel dans un monde où la diversité linguistique est omniprésente. La langue arabe parlé par des millions de personnes et de plus en plus visible sur le web, constitue un défi majeur pour le traitement automatique du langage naturel : une morphologie riche et non-concaténative, une ambiguïté accentuée par l'absence fréquente de vocalisation, le phénomène de sujet implicite, la diglossie existante entre l'arabe standard et les différents dialectes, ainsi qu'à un manque significatif de ressources linguistiques structurées. Ces spécificités de la langue arabe rendent l'utilisation directe de modèles conçus pour d'autres langues inefficace.

Dans ce contexte, les grammaires catégorielles se présentent comme un modèle formel à la fois rigoureux et assez expressif. Leur approche part directement du lexique, crée un lien clair entre la syntaxe et la sémantique, puisque chaque mot reçoit un type qui indique comment il peut se combiner avec les autres. Ensuite l'analyse se fait alors grâce à des règles de réduction qui permettent de construire la structure syntaxique et aussi la représentation du sens à partir du lexique lui-même. L'évolution de ces grammaires depuis leurs bases logiques et philosophiques jusqu'aux modèles plus récents comme la Combinatory Categorial Grammar (CCG) de Steedman, a renforcé leur capacité à gérer la souplesse de la syntaxe et les dépendances complexes, qui sont des aspects très importants pour le traitement de la langue arabe.

Dans ce travail nous utilisons la Grammaire Catégorielle Combinatoire Applicative (GCCA) développée par Biskri et Desclés, pour étudier et analyser la langue arabe. En généralisant les modèles catégoriels antérieurs la GCCA propose un cadre rigoureux permettant d'articuler la forme de surface (phénotype) et la structure

sémantique abstraite (génotype) via la logique combinatoire. Bien que la GCCA ait déjà prouvé son efficacité dans le traitement des structures arabes fondamentales (SVO/VSO, sujets implicites), notre étude se distingue par son intérêt porté à la modélisation des constructions Inna wa Akhawatoha et Kana wa Akhawatoha. Ces particules omniprésentes dans l'arabe écrit et parlé apportent des modifications importantes dans le marquage des cas à l'intérieur des phrases nominales (par exemple : Inna rend le sujet accusatif, tandis que Kana rend le prédicat accusatif). La formalisation précise de ces constructions qui impliquent à la fois des alternances casuelles et des effets sémantiques subtils, constitue vraiment un défi majeur.

L'originalité de ce mémoire réside dans la manière dont la GCCA permet de capturer ces phénomènes par des mécanismes de changement de type, notamment à l'aide de combinateurs comme C*, conçus pour représenter de façon explicite ces ajustements grammaticaux propres à l'arabe. Une modélisation réussie à l'aide de la GCCA démontrerait non seulement son efficacité pour une langue aussi riche et structurée que l'arabe, mais pourrait également mettre en lumière sa supériorité ou du moins sa complémentarité par rapport à d'autres formalismes ayant rencontré des limites dans l'analyse des phrases nominales arabes.

Ainsi ce mémoire vise à démontrer que la GCCA constitue un outil puissant, précis, et traitable d'un point de vue computationnel afin d'étudier les structures introduites par Inna et Kana en arabe. le travail effectué apporte une contribution au domaine du traitement automatique de la langue en général et à la linguistique computationnelle appliquée à l'arabe en particulier. Et pour atteindre ces objectifs notre recherche s'appuie d'abord sur une base théorique solide et une analyse approfondie des spécificités grammaticales de l'arabe, ensuite nous avons élaboré une méthodologie d'application de la GCCA et finalement nous passons à l'étape de l'évaluation de performance de ce modèle sur un corpus représentatif.

Ce mémoire est structuré d'une façon claire et progressive, présentant l'ensemble des travaux menés avec les résultats obtenus. Où chaque chapitre traite un aspect fondamental de la recherche, suivant d'un fil logique allant des fondements théoriques jusqu'à l'expérimentation et l'analyse des données.

Le premier chapitre constitue l'introduction générale, où on expose le contexte dans lequel s'inscrit ce travail de recherche, définit la problématique, précise les objectifs poursuivis et présente l'organisation globale du mémoire.

Le deuxième chapitre se focalise sur les fondements théoriques des systèmes applicatifs et sur la logique combinatoire. Ces principes forment la base conceptuelle de la Grammaire Catégorielle Combinatoire Applicative (GCCA) qui est le cadre formel appliqué dans notre recherche.

Le troisième chapitre illustre l'évolution des grammaires catégorielles en présentant leurs principales caractéristiques, leurs différents formalismes, ainsi que les spécificités du modèle GCCA. Ce chapitre met en lumière les raisons qui motivent le choix de ce formalisme pour l'analyse syntaxico-sémantique de l'arabe.

Le quatrième chapitre offre une description détaillée des caractéristiques linguistiques de la langue arabe, avec un accent particulier sur les constructions introduites par les particules Inna et Kana ainsi que leurs sœurs. Cette section met en évidence les défis que posent ces constructions en matière de modélisation formelle.

Le cinquième chapitre est dédié à l'application pratique de la GCCA dans l'analyse des structures contenant *Inna wa Akhawatoha et Kana wa Akhawatoha*. Il expose la méthodologie adoptée avec les mécanismes formels utilisés pour rendre compte des phénomènes grammaticaux observés.

Le sixième chapitre traite les aspects liés à l'implémentation computationnelle du modèle développé. où on discute les choix techniques opérés, les outils utilisés et les contraintes rencontrées dans le processus de modélisation.

Le septième chapitre présente le dispositif expérimental mis en place pour évaluer l'efficacité de l'approche proposée. où on va analyser les résultats obtenus à partir d'un corpus ciblé, en les confrontant aux hypothèses de départ et aux objectifs initiaux.

Enfin, le huitième chapitre propose une synthèse des principales conclusions de cette recherche mettant en valeur les apports théoriques et pratiques du travail réalisé, et qui ouvre également des perspectives pour des améliorations futures. Le neuvième et dernier chapitre réunit l'ensemble des références bibliographiques ayant servi de base aux développements théoriques et méthodologiques du mémoire.

# Chapitre 02 : Systèmes Applicatifs et Logique Combinatoire

## 2.1 Introduction : Le Problème de la Composition

Avant de se lancer dans la modélisation de la syntaxe des langues naturelles, il est essentiel de comprendre les principes logiques qui rendent cette tâche réalisable. Chaque grammaire formelle, de manière implicite ou explicite, est fondée sur une théorie de la composition : comment des unités élémentaires (les mots) s'assemblent-elles pour créer des entités plus complexes (les syntagmes et les phrases) qui possèdent une structure et un signification ? L'approche des grammaires catégorielles à cette question est directement dérivée des recherches sur les systèmes applicatifs et la logique combinatoire. L'objectif de ce chapitre est d'examiner ces bases, en démontrant de quelle manière une tentative de simplification de la logique a procuré à la linguistique un instrument d'une remarquable efficacité.

## 2.2 Le Contexte : Lambda-Calcul et la Gestion des Fonctions

Pour comprendre mieux la naissance de la logique combinatoire, il est primordial d'examiner le formalisme prédominant qui sert à représenter les fonctions au début du XXe siècle : le calcul lambda, élaboré par Alonzo Church dans les années 1930. Ce dernier est un system formel qui permet la définition anonyme des fonctions. Pour illustrer, la fonction qui reçoit un nombre x et délivre x + 1 peut être exprimée comme λx. x + 1. L'opération centrale est l'application : appliquer cette fonction à la

valeur 3 s'écrit (λx. x + 1) 3, et le résultat, obtenu par un processus appelé β-réduction, est 3 + 1, soit 4.

Ce système est extrêmement puissant, mais sa manipulation implique une gestion parfois complexe des variables. Les variables peuvent être "liées" (comme x dans λx) ou "libres". Des opérations comme la substitution d'une variable par une valeur doivent être menées avec précaution pour éviter des "captures" accidentelles de variables, ce qui nécessite des règles de renommage (α-conversion). C'est dans ce contexte que l'on peut se demander : est-il possible de construire un système de même puissance, mais sans utiliser de variables ?

## 2.3 L'Innovation de Schönfinkel : vers une Logique sans Variables

La première réponse positive à cette question a été apportée bien avant Church, par le logicien russe Moses Schönfinkel dans une communication de 1924. Son idée, d'une simplicité radicale, était d'éliminer complètement les variables en ne conservant que deux fonctions de base, ou "combinateurs", qu'il nomma S et K.

- Le combinateur K (pour Konstanzfunktion, fonction constante) est une fonction qui prend deux arguments et retourne le premier : K x y = x. Il a un effet de "suppression".
- Le combinateur S (pour Substitution) est une fonction qui prend trois arguments (qui sont eux-mêmes des fonctions) et les distribue d'une manière spécifique : S f g x = (f x) (g x). Il a un effet de "distribution".

L'intuition fondamentale de Schönfinkel était que toute fonction, même complexe, pouvait être décomposée en une série d'applications de ces deux opérateurs de base. Une autre de ses intuitions majeures est aujourd'hui connue sous le nom de curryfication (en l'honneur de Haskell Curry qui l'a développée). L'idée est qu'une

fonction qui semble prendre plusieurs arguments, comme f(x, y), peut être réécrite comme une fonction qui n'en prend qu'un seul, x, et qui retourne une nouvelle fonction qui, à son tour, prendra l'argument y. On note cela (f x) y. Cette vision est absolument centrale, car elle transforme chaque élément en un foncteur attendant un argument

## 2.4 La Systématisation de Curry : la Logique Combinatoire

Les travaux de Schönfinkel sont restés relativement confidentiels jusqu'à ce que Haskell Curry, indépendamment, redécouvre et développe ces idées à partir de la fin des années 1920, donnant naissance à la logique combinatoire en tant que discipline formelle [1]. Curry a défini un combinateur comme une fonction qui ne contient aucune variable libre. Il a montré qu'avec une base de combinateurs (comme S et K), on pouvait construire un système Turing-complet, c'est-à-dire capable d'effectuer les mêmes calculs que n'importe quel ordinateur.
Il a également identifié d'autres combinateurs utiles, comme le combinateur d'identité I, qui retourne simplement son argument : I x = x. L'élégance du système est telle que I n'est même pas nécessaire comme primitif, car il peut être défini à partir de S et K : I = S K K.

Démonstration : (S K K) x = (K x) (K x) = x

Cette approche réduit la logique à un jeu de construction : la combinaison de quelques briques de base permet de recréer n'importe quelle structure fonctionnelle.

## 2.5 De la Logique à l'Informatique : Le Calcul Fonctionnel Moderne

L'histoire du calcul fonctionnel moderne a démontré comment une idée née dans la logique pure a progressivement défini la manière dont nous concevons aujourd'hui les systèmes informatiques et les modèles du langage. À l'origine, Schönfinkel puis Curry ont cherché à simplifier la logique en supprimant toute référence explicite aux variables, pour ne garder que des fonctions dites combinateurs capables, à elles seules, d'exprimer n'importe quelle opération [1]. Cette approche a ouvert la voie à une vision du raisonnement où la composition devient le principe fondamental : chaque structure complexe peut être reconstruite à partir d'éléments simples appliqués les uns aux autres.

Le Lambda-Calcul introduit un peu plus tard par Church et approfondi par Hindley et Seldin  a formalisé cette idée avec une rigueur remarquable. Il a non seulement permis de représenter les fonctions de manière abstraite, mais aussi de poser les bases de nombreux langages de programmation modernes, où l'application fonctionnelle constitue le cœur même du calcul. Ces travaux ont ainsi rapproché deux mondes longtemps séparés : celui de la logique symbolique et celui de l'informatique naissante.

Dans un second temps, ces concepts ont inspiré la linguistique formelle, notamment à travers les travaux d'Ajdukiewicz [2], qui a introduit l'idée qu'une phrase pouvait être comprise comme le résultat d'une suite d'applications fonctionnelles.
La relation entre la logique et le langage s'est donc avérée plus profonde que nous ne la estimions : les mêmes principes qui facilitent le développement de programmes sont aussi en mesure d'indiquer comment les mots se combinent pour créer du sens. Cette convergence entre logique, informatique et linguistique démontre à quel point la théorie des fonctions et de la composition reste un outil essentiel pour comprendre aussi bien les machines que le langage humain.

## 2.6 Les Combinateurs Logiques et leur Rôle dans la GCCA

Les combinateurs logiques constituent le cœur de la logique combinatoire, fondée par Schönfinkel (1924) et développée par H. B. Curry.
Ils représentent des opérateurs abstraits qui permet de construire à partir de fonctions élémentaires des expressions de plus en plus complexes selon des règles strictes de β-réduction.

Dans la Grammaire Catégorielle Combinatoire Applicative (GCCA) ces combinateurs servent à formaliser la structure fonctionnelle du langage naturel en distinguant les opérateurs (fonctions) et les opérandes (arguments).
Ils constituent le socle d'un formalisme qui rend compte de la composition syntaxico-sémantique, tout en offrant un modèle opératoire du raisonnement linguistique [3].

### 2.6.1 Le combinateur d'effacement K

Le combinateur K (pour Konstanzfunktion) est défini par la règle :

$$K\,f\,x = f$$

Ce combinateur prend deux arguments et renvoie toujours le premier, effaçant le second. Il permet ainsi de neutraliser un argument non pertinent dans une structure applicative.

Exemple logique :

$$K\ (manger)\ (Pierre) \Rightarrow manger$$

Ici, l'argument Pierre est ignoré : le combinateur K "efface" l'opérande inutile.

Ce principe est utilisé dans la GCCA pour modéliser des cas où un opérateur conserve son sens indépendamment d'un argument absent ou supprimé [3].

### 2.6.2 Le combinateur de distribution S

Le combinateur S (pour Substitution) distribue un argument unique à deux fonctions distinctes :

$$S\,X\,Y\,Z=(\,X\,Z\,)\,(\,Y\,Z\,)$$

Autrement dit, Z est transmis à la fois à X et à Y, puis leurs résultats sont combinés.
Exemple :
Soient $X=a$ et $Y=b$, deux fonctions, et $Z=x$.

$$S\,a\,b\,x=(\,a\,x\,)\,(\,b\,x\,)$$

Le combinateur S est fondamental dans la construction d'expressions où un même argument est partagé, comme dans :

$$\textit{Pierre parle et chante} \longrightarrow \varphi(\textit{parler})(\textit{chanter})(\textit{Pierre}) \longrightarrow (\textit{parle Pierre})\ \textit{et (chante Pierre)}$$

S assure ici la distribution de l'argument commun [3].

### 2.6.3 Le combinateur d'identité I

Le combinateur I représente la fonction identité :

$$I\,P = P$$

Il restitue simplement son argument sans le modifier.
Ce combinateur peut être défini à partir de S et K selon la relation :

$$I = S\,K\,K$$

Démonstration :

$$S\,K\,K\,x= K\,x\,(\,K\,x\,) = x$$

Exemple :

$$I(\textit{manger Pierre})=\textit{manger Pierre}$$

Le combinateur I sert de base à de nombreuses constructions dérivées dans la logique combinatoire et la GCCA [3].

### 2.6.4 Le combinateur de composition B

Le combinateur B traduit la composition fonctionnelle :

$$B\ X\ Y\ Z = X\ (\ Y\ Z\ )$$

Il permet d'appliquer d'abord Y à Z, puis d'appliquer X au résultat.
C'est le formalisme du "faire suivre" logique.
Exemple :

$$B\ (aime)\ (connaît)\ (Pierre) = aime(connaît(Pierre))$$

Autrement dit, "aimer celui qu'on connaît".

B peut être exprimé par les combinateurs S et K :

$$B = S\ (\ K\ S\ )\ K$$

Le combinateur B est essentiel dans la GCCA pour représenter la composition hiérarchique des prédicats [3].

### 2.6.5 Le combinateur de duplication W

Le combinateur W duplique un argument et l'applique deux fois à une fonction :

$$W\ f\ x = f\ x\ x$$

Il sert à exprimer des situations où une même entité joue deux rôles dans une relation.

Exemple linguistique :
Si l'on considère le prédicat se-blesser, on peut écrire :

$$se{-}blesser = W(blesser)$$

Ainsi,

$$W\ (\ blesser\ )\ (\ Pierre\ ) = blesser\ Pierre\ Pierre$$

Les deux expressions sont sémantiquement équivalentes : Pierre se blesse ↔ Pierre blesse Pierre [3].

W peut aussi être défini à partir de S, K et I :

$$W=SS(KI)$$

### 2.6.6 Le combinateur de coordination φ

Le combinateur φ exprime la coordination de deux prédicats partageant le même argument :

$$\varphi FGHx=F(Gx)(Hx)$$

Exemple :

$$\varphi\ (et)\ (parler)\ (chanter)\ (Pierre)=et(parle\ Pierre)(chante\ Pierre)$$

Le combinateur φ permet ainsi de représenter les structures de coordination symétrique en grammaire applicative [3].

### 2.6.7 Le combinateur de distribution ψ

Le combinateur ψ agit de manière similaire, mais introduit une distribution asymétrique :

$$\psi FGHx=F(Gx)(GH)$$

Ce combinateur est utilisé lorsque l'un des prédicats doit être partiellement distribué ou dépendant du premier.

Exemple :

$$\psi\ (et)\ (préparer)\ (terminer)\ (repas)$$

donne :

$$et\ (prépare\ repas)\ (prépare\ terminer)$$

→ structure où la deuxième action dépend de la première [3].

### 2.6.8 Le combinateur de changement de type C*

Le combinateur C* (ou type-raising) transforme un opérateur en opérande et inversement :

$$C * X Y = Y X$$

Ce mécanisme permet de modifier la hiérarchie fonctionnelle d'une expression. Il est utilisé dans la GCCA pour décrire les inversions syntaxiques ou les changements de rôle grammatical.

Exemple :

$$C * ( voir )( Jean ) = Jean\ voir$$

Autrement dit, l'opérande Jean devient à son tour opérateur.
Ce combinateur est particulièrement utile pour représenter les constructions où la position syntaxique varie sans changer la fonction logique [3].

### 2.6.8 Le combinateur de permutation C

Le combinateur C traduit la commutativité des arguments :

$$CFXY=FYX$$

Exemple :

$$C\ (rencontrer)\ (Pierre)\ (Marie) = rencontrer\ (Marie)\ (Pierre)$$

Le sens reste identique : Pierre rencontre Marie ↔ Marie rencontre Pierre.
C peut être défini à partir de S, B et K :

$$C = S ( B B S )( K K )$$

Ce combinateur est utilisé dans la GCCA pour modéliser les relations symétriques ou réciproques [3].

## 2.7 Combinateurs complexes

Les combinateurs complexes résultent de la combinaison successive de plusieurs combinateurs élémentaires.
Chaque combinaison produit un nouvel opérateur qui hérite des propriétés de ses composants, tout en exprimant des comportements fonctionnels plus élaborés [3].

### 2.7.1 Le combinateur complexe BC

L'association du combinateur de composition B et du combinateur de permutation C permet de créer un opérateur capable de composer tout en inversant l'ordre des arguments :

$$B\ C\ X\ Y\ Z = X\ (\ C\ Y\ Z\ )$$

Exemple :

*BC (aime) (donner) (Pierre) = aime (donner Pierre)*

Ce combinateur sert à décrire des constructions où la composition d'un prédicat dépend de l'ordre des rôles syntaxiques.

### 2.7.2 Le combinateur complexe C*B

La combinaison de C* (changement de type) et de B (composition) produit un opérateur capable de composer des fonctions tout en modifiant leur hiérarchie fonctionnelle :

$$B\ X\ Y\ Z = Y\ (\ X\ Z\ )$$

Exemple :

*C∗B (donner) (Marie) (livre) = Marie (donnerlivre)*

Ce type de structure reflète les inversions de sujet ou de complément dans certaines constructions syntaxiques.

### 2.7.3 Le combinateur complexe BW

Le combinateur BW associe la composition et la duplication :

$$B\ W\ X\ Y\ Z = X\ (\ Y\ Z\ Z\ )$$

Exemple logique :

$$BW\ (juger)\ (aimer)\ (Pierre) = juger\ (aimer\ Pierre\ Pierre)$$

Ce combinateur exprime des relations où l'argument est impliqué deux fois dans le même processus — comme dans les prédicats réflexifs complexes.

### 2.7.4 Le combinateur complexe φψ

L'association des deux combinateurs de coordination φ et ψ permet de décrire des coordinations mi-symétriques, mi-asymétriques :

$$\varphi\psi FGHx = F((Gx)(Hx))(GH)$$

Exemple :

$$\varphi\psi(et)\ (préparer)\ (manger)\ (repas)$$

$$\rightarrow\ et\ (prépare\ repas\ et\ mange\ repas)\ \ (prépare\ manger)$$

Cette structure représente des coordinations hiérarchisées, typiques des phrases où les actions se succèdent ou se conditionnent mutuellement.

### 2.7.5 Le combinateur complexe BCC

Un exemple classique étudié par Desclés et Biskri [3] est celui du combinateur BCC, formé de la composition B suivie de deux permutations C :

$$B\ C\ C\ X\ Y\ Z = X\ (\ Z\ Y\ )$$

Réduction :

$$B\ C\ C\ x\ y\ z \rightarrow C\ (\ C\ x\ )\ y\ z$$
$$\rightarrow (\ C\ x\ )\ z\ y$$
$$\rightarrow x\ y$$

Ce combinateur illustre le potentiel combinatoire du système, capable de reproduire des schémas d'application complexes à partir de simples combinateurs de base.

### 2.7.6 Le combinateur complexe BCS

Enfin, le combinateur BCS associe la composition, la permutation et la distribution :

$$B\ C\ S\ X\ Y\ Z = X\ (\ (Z\ \ Y)\ (\ Y\ Z)\ )$$

Exemple :
*BCS (analyser) (associer) (concept)=analyser((concept associer)(associer concept))*

Ce type de combinateur permet de représenter des dépendances circulaires ou réciproques entre fonctions, fréquentes dans la modélisation sémantique des prédicats à double rection.

### 2.7.7 Conclusion

L'ensemble des combinateurs présentés ( élémentaires et complexes ) illustre la richesse opératoire de la logique combinatoire au sein de la GCCA.
Ils permettent d'exprimer les mécanismes fondamentaux du raisonnement applicatif : composition, duplication, permutation, coordination et inversion fonctionnelle.

Ces outils hérités de la logique combinatoire de Curry, constituent un langage formel cohérent pour la représentation des structures syntaxico-sémantiques du langage nature.

# Chapitre 03 : Grammaires catégorielles

## 3.1 Introduction

L'objectif du troisième chapitre est d'explorer les Grammaires Catégorielles (GC), un formalisme linguistique particulièrement efficace pour le traitement automatique des langues naturelles. L'informatique linguistique se développe constamment pour répondre aux besoins d'une diversité croissante de langues sur le web.

La problématique du projet se concentre donc sur les Grammaires Catégorielles Combinatoires et Applicatives (GCCA), car elles offrent un cadre formel et universel capable de surmonter les limites des approches traditionnelles. Ce chapitre a pour but de présenter les fondements de cette approche, en retraçant son évolution de la philosophie à sa formalisation moderne, et en posant les bases théoriques nécessaires pour la suite de l'analyse.

Dans le chapitre précédent, nous avons présenté les fondements de la logique combinatoire et les principaux combinateurs logiques selon la formulation de Curry et l'interprétation linguistique de Desclés et Biskri.

Ces combinateurs constituent la base opératoire de la Grammaire Catégorielle Combinatoire et Applicative (GCCA), que nous présentons dans ce chapitre, ainsi que les règles formelles qui en assurent le fonctionnement.

Comme l'indique Biskri (2006) dans le contexte des applications linguistiques multilingues destinées au Web, les grammaires catégorielles offrent un cadre solide « indépendant des particularités grammaticales des différentes langues »

## 3.2 Historique des Grammaires Catégorielles

Les grammaires catégorielles trouvent leurs origines dans les recherches du philosophe Husserl et dans celles des logiciens Ajdukiewicz, Lesniewski et Bar-Hillel. Et grâce a leurs travaux de recherche qui ont posé les bases logiques et philosophiques, et qui ont rendu possible l'émergence des grammaires catégorielles modernes.

### 3.2.1 Fondements philosophiques des grammaires catégorielles Husserl

Edmund Husserl, qui a d'abord étudié les mathématiques, a progressivement orienté ses recherches vers un champ intellectuel encore faiblement exploré au début du vingtième siècle : la phénoménologie. Dans sa recherche de 1913, il suggère une méthode novatrice pour traiter les objets de réflexion, non pas en se limitant à leur définition scientifique ou empirique, mais en questionnant leur origine profonde. Il utilise souvent l'exemple traditionnel de la question : « Qu'est-ce qu'une fleur ? » pour démontrer son effort d'excéder l'observation matérielle afin de comprendre le sens véritable de l'objet. Cette approche a eu un impact significatif sur l'évolution des théories de la logique et de la linguistique, pavant ainsi la voie aux grammaires catégorielles.

### 3.2.2 Lesniewski (1922)

Avant les travaux d'Ajdukiewicz et l'un des premières initiatives théoriques qui ont préparé l'émergence des grammaires catégorielles se trouve chez Stanisław Lesniewski. En 1922 ce logicien de l'École de Lwów-Varsovie développe une théorie des catégories sémantiques dans le cadre de sa logique formelle [4]. L'objectif principal de sa réflexion était d'éviter les paradoxes logiques en construisant un système cohérent où chaque expression reçoit une place déterminée selon sa catégorie. Cette approche, qui visait initialement à résoudre des problèmes logiques, a eu une influence déterminante sur ses disciples, en particulier Kazimierz Ajdukiewicz. En effet ce dernier s'appuie directement sur les idées de Lesniewski pour élaborer sa propre conception syntaxique fondée sur le principe de calcul de fraction [5] en 1935.

Ainsi, Lesniewski peut être considéré comme l'un des précurseurs des grammaires catégorielles, en préparant le terrain conceptuel qui permettra d'articuler plus tard logique, syntaxe et phénoménologie.

### 3.2.3 Ajdukiewicz (1935)

Les premières versions des grammaires catégorielles sont généralement attribuées à Kazimierz Ajdukiewicz, ce dernier et dans son article Die syntaktische Konnexität publié en 1935 dans Studia Philosophica, il a élaboré un système original où les catégories syntaxiques peuvent être combinées selon un principe de calcul analogue à celui des fractions. Chaque expression est ainsi assignée à une catégorie grammaticale, et la validité syntaxique d'une phrase dépend de la possibilité de « multiplier » ces catégories pour obtenir une structure cohérente [2]. Ce modèle a été dévloppé de la tradition logique de l'École de Lwów-Varsovie, en particulier dans les travaux de Lesniewski et s'inscrit dans un contexte plus large marqué par l'influence philosophique de Husserl [4]. Il constitue une avancée décisive et importante dans l'histoire de la linguistique formelle car il propose une articulation claire entre le syntaxe et la logique. Par la suite, ses idées seront reprises et développées par Bar-Hillel (1953), puis par Lambek (1958), qui donneront aux grammaires catégorielles une place centrale dans les théories linguistiques actuelles [6].

### 3.2.4 Bar-Hillel (1953)

Dans la continuité des travaux d'Ajdukiewicz, Yehoshua Bar-Hillel propose en 1953 une version enrichie de la grammaire catégorielle. Cette approche, appelée grammaire catégorielle bidirectionnelle, est également connue sous le nom de grammaire AB [7]. Elle se distingue de la version unidirectionnelle antérieure par une plus grande souplesse et par une formalisation plus systématique, ce qui a favorisé son usage en linguistique.

Le modèle de Bar-Hillel repose sur deux caractéristiques majeures, il est entièrement lexicalisé et utilise des règles de combinaison universelles [6].

Dans ce cadre les expressions linguistiques se répartissent en deux types principaux :

- Les catégories atomiques qui correspondent à des expressions complètes (comme une phrase).

- Les catégories complexes construites à partir de catégories atomiques au moyen des opérateurs « / » et « \ » permettant des réductions soit vers la gauche, soit vers la droite.

Les règles de réduction peuvent être formulées de la manière suivante :

1. Réduction à droite : (x/y) y → x

2. Réduction à gauche : y (y\ x) → x

Et pour illustrer ce fonctionnement nous prenons l'exemple de la phrase « Marie parle chinois ». L'élément Marie reçoit la catégorie N (nom) et chinois également. Le verbe parler en tant que verbe transitif, est représenté par la catégorie (N\S)/N car il attend un objet à sa droite et un sujet à sa gauche. En appliquant successivement les règles de réduction on aboutit à la catégorie S (phrase) ce qui confirme que l'énoncé est grammaticalement correct.
La grammaire AB a aussi permis de normaliser l'attribution des catégories lexicales. Par exemple les déterminants tels que le, la, un, une et les reçoivent la catégorie N/N, puisqu'ils se combinent avec un nom pour former un groupe nominal. De même les verbes transitifs sont notés (N\S)/N car ils requièrent un sujet et un complément d'objet.

Cette contribution représente une étape clé dans l'histoire des grammaires catégorielles car elle introduit une méthode plus générale et standardisée dans le but d'analyser la structure des langues naturelles.

## 3.3 La grammaire catégorielle classique

La grammaire catégorielle classique représente la première conception des grammaires catégorielles, elle a été proposée par Kazimierz Ajdukiewicz (1935) et reprise par Yehoshua Bar-Hillel (1953). Le but principal de cette derniére est d'analyser et de décrire la structure syntaxique des phrases par un système logique simple composé de catégories.

Dans ce cadre chaque mot est associé à une catégorie grammaticale qui peut être :

- Atomique comme N pour les noms ou S pour les phrases.

- Complexe construite à l'aide d'opérateurs (/, ) pour indiquer comment une catégorie se combine avec d'autres.

L'analyse syntaxique vise à appliquer des règles de réduction pour voir si une séquence de catégories peut se réduire à la catégorie S (phrase).

Exemple :
Prenons la phrase « Marie lit un livre ».

- Marie reçoit la catégorie N ;

- lit est un verbe transitif, donc (N\S)/N (il attend un objet à droite et un sujet à gauche) ;

- un livre correspond à N.

En appliquant les règles de réduction, la séquence se réduit en S, ce qui montre que l'énoncé est syntaxiquement correct.

**Intérêt et limites**

La grammaire catégorielle classique présente plusieurs avantages :

- Simplicité car elle s'appuie sur un nombre réduit de règles générales.

- Universalité où les mêmes principes peuvent s'appliquer à différentes langues.

- Lien syntaxe-sémantique où chaque dérivation syntaxique correspond directement à une interprétation logique.

Cependant elle reste limitée face à certaines structures complexes et ambiguës. Ces insuffisances ont donné lieu à des extensions telles que la grammaire catégorielle bidirectionnelle de Bar-Hillel [7] et le calcul de Lambek (1958) qui renforcent son expressivité.

## 3.4 Le calcul de Lambek (1958)

En 1958 Joachim Lambek propose une nouvelle formalisation des grammaires catégorielles connue sous le nom de calcul de Lambek (The mathematics of sentence structure Lambek 1958). Cette approche prolonge les travaux de Bar-Hillel [7] en introduisant une vision plus proche de la logique mathématique. L'idée fondamentale est de considérer la syntaxe comme un système de déduction ou démontrer la grammaticalité d'une phrase revient à établir une preuve logique [8].

Le calcul de Lambek se distingue d'abord par son principe d'associativité qui autorise une organisation flexible des constituants syntaxiques. De plus il repose sur l'utilisation de connecteurs logiques spécifiques (/, , •), ces derniers permettant de modéliser les relations entre catégorie [9]. Ainsi, chaque dérivation syntaxique peut

être interprétée comme une séquence de règles formelles, renforçant le lien entre grammaire et logique.

Par exemple, dans une phrase comme « Jean aime Marie », les noms Jean et Marie reçoivent la catégorie N, tandis que le verbe aime est représenté par (N\S)/N. L'application des règles du système aboutit à la catégorie S, confirmant que l'énoncé est bien formé [10].

En introduisant ce cadre logique, Lambek a ouvert la voie à des développements importants, notamment dans les grammaires catégorielles combinatoires et les logiques sous-structurales, qui exploitent encore aujourd'hui les fondations posées dans son article de 1958.

## 3.5 La grammaire catégorielle combinatoire (GCC)

La Grammaire Catégorielle Combinatoire (GCC), également connue sous le nom de Combinatory Categorial Grammar (CCG), est une extension des grammaires catégorielles classiques et du calcul de Lambek. Elle a été développée dans les années 1980 par Steedman (1987, 2000), avec pour objectif de rendre la grammaire catégorielle plus expressive et plus adaptée à la description des phénomènes syntaxiques complexes des langues naturelles.

La GCC conserve l'idée fondamentale des grammaires catégorielles : chaque mot est associé à une catégorie qui décrit son comportement syntaxique. Cependant, elle enrichit ce cadre par l'introduction de combinateurs fonctionnels, qui permettent d'analyser des constructions syntaxiques plus souples, comme la coordination (Jean chante et danse), l'extraction (Quel livre Marie a-t-elle lu ?) ou encore les structures discontinues.

### 3.5.1 Principales caractéristiques de la GCC :

1. **Extension des règles de combinaison**
   Contrairement aux grammaires catégorielles classiques qui ne reposaient que sur deux règles (réduction gauche/droite), la GCC introduit plusieurs combinateurs (composition fonctionnelle, type-raising, substitution, etc.) qui permettent de modéliser des structures syntaxiques plus variées [11].

2. **Transparence syntaxe-sémantique**
   Comme dans les versions précédentes, chaque dérivation syntaxique correspond directement à une interprétation sémantique. Ce lien fort entre syntaxe et logique en fait un cadre privilégié pour l'analyse du langage naturel [12].

3. **Flexibilité et robustesse**
   La GCC est capable de gérer des constructions où la dépendance entre mots n'est pas strictement locale, ce qui la rend particulièrement efficace dans le cadre du traitement automatique du langage et des grammaires probabilistes.

4. **Applications en informatique**
   La GCC est aujourd'hui utilisée dans plusieurs systèmes de traitement automatique du langage naturel (TALN), notamment pour la parsing automatique (analyse syntaxique automatique) et la génération de représentations logiques en sémantique computationnelle [13].

Exemple simplifié :

Dans une phrase comme « Jean peut chanter », le verbe modal peut est analysé grâce aux mécanismes de type-raising et de composition fonctionnelle, permettant de combiner correctement les catégories sans avoir recours à des règles transformationnelles complexes.

Ainsi, la GCC offre un modèle souple et puissant, qui conserve l'élégance logique de la grammaire catégorielle tout en répondant aux limites de ses versions classiques.

La Grammaire Catégorielle Combinatoire Applicative (GCCA), proposée par Desclés et Biskri [3], reprend ces principes tout en introduisant explicitement les combinateurs logiques dans les dérivations syntaxiques.

Cette approche étend la GCC de Steedman à une formalisation plus générale, dans laquelle les règles de combinaison syntaxique sont directement liées aux opérateurs logiques issus du système de Curry.

## 3.6 La Grammaire Catégorielle Combinatoire Applicative (GCCA)

La Grammaire Catégorielle Combinatoire Applicative (GCCA) constitue une extension applicative des grammaires catégorielles combinatoires ou elle repose sur le principe que toute relation syntaxique entre les unités linguistiques peut être représentée par une expression applicative dérivée de la logique combinatoire. Chaque unité lexicale reçoit une catégorie typée, et les opérations de combinaison suivent des règles formelles de réduction.
Ces règles, analogues à celles de Steedman, sont enrichies par l'usage explicite de combinateurs tels que B (composition), T (changement de type), φ (coordination), etc..

### 3.6.1 Les règles de la GCCA

Les règles de la GCCA permettent d'effectuer les réductions entre les catégories assignées aux unités lexicales. Elles sont inspirées des travaux de Steedman mais introduisent explicitement les combinateurs logiques de Curry.
On distingue les règles d'application et les règles combinatoires.

**Règle d'application avant (>):**

$$[X/Y : U1]\ [Y : U2] \rightarrow [X : (U1\ U2)]$$

Exemple :

*« Le chat dort »*

```
Le        chat       dort
N/N       N          S\N
--------------------------->
     N          S\N
--------------------------->
         S
```

**Règle d'application arrière (<):**

$$[Y : U1]\ [X\backslash Y : U2] \rightarrow [X : (U2\ U1)]$$

Exemple :

*« Les enfants jouent »*

```
Les       enfants      jouent
N/N       N            S\N
--------------------------->
     N          S\N
---------------------------<
         S
```

**Règle de changement de type (T):**

$$[X : u] \rightarrow [Y/(Y\backslash X) : (C^{*}\ u)]$$

Exemple :

*« Pierre chante »*

```
Pierre      chante
N           S\N
---------------------------->T
    S/(S\N)    S\N
---------------------------->
         S
```

**Règle de composition fonctionnelle (B):**

$$[X/Y : u1]\ [Y/Z : u2] \rightarrow [X/Z : (B\ u1\ u2)]$$

Exemple :

*« Paul veut chanter »*

```
Paul      veut          chanter
N     (S\N)/(S\N)      S\N
---------------------------->B
      (S\N)/N       S\N
---------------------------->
         S
```

### 3.6.2 Exemple d'Analyse GCCA

Considérons la phrase :

*« Le professeur explique la leçon »*

```
Le        professeur     explique      la        leçon
N/N          N           (S\N)/N      N/N        N
---------------------------->
              N          (S\N)/N      N/N        N
---------------------------->
                        S\N         N/N        N
---------------------------->
                  S\N       N
---------------------------->
             S
```

Cette dérivation illustre le passage d'une structure concaténée à une structure applicative, où chaque opération correspond à l'application d'un combinateur logique.

## 3.7 Conclusion

La Grammaire Catégorielle Combinatoire Applicative (GCCA) combine la logique de la grammaire catégorielle avec la puissance opératoire de la logique combinatoire. En intégrant explicitement les combinateurs logiques aux règles syntaxiques. La GCCA permet aussi de décrire les dépendances fonctionnelles à l'intérieur des phrases et d'établir un lien direct entre la structure syntaxique et la structure sémantique. Dans les chapitres qui suivent nous appliquerons cette approche à la langue arabe et en nous intéressant plus particulièrement aux constructions Inna wa Akhawatouha et Kana wa Akhawatouha.

# Chapitre 04 : La Langue Arabe

## 4.1 Introduction à la langue arabe

La langue arabe occupe une place centrale parmi les langues naturelles, tant par son histoire que par son rôle actuel dans le monde. Elle est la langue officielle de plus de vingt pays et l'une des langues les plus parlées à l'échelle internationale, avec plus de 300 millions de locuteurs natifs. Elle est également une langue liturgique, utilisée dans le contexte religieux de l'islam à travers le Coran, ce qui lui confère une dimension symbolique et culturelle considérable [14].

Sur le plan linguistique, l'arabe se distingue par la richesse de son système morphologique et par une structure syntaxique particulière qui attire l'intérêt des chercheurs en linguistique formelle et computationnelle. Contrairement aux langues indo-européennes, elle repose fortement sur un système morphologique dérivationnel et flexionnel basé sur des racines consonantiques, généralement trilittères, à partir desquelles sont formés des mots et des structures grammaticales complexes [15] et [16].

Par ailleurs, la syntaxe arabe présente des caractéristiques propres, telles que la possibilité d'alternance entre l'ordre Verbe–Sujet–Objet (VSO) et Sujet–Verbe–Objet (SVO). Elle se distingue aussi par l'importance des cas grammaticaux (marqués par les voyelles finales) qui jouent un rôle déterminant dans l'interprétation des relations syntaxiques. Ces aspects ont été largement étudiés par des grammairiens modernes comme Badawi, Carter et Gully [17] et approfondis par Ryding dans son ouvrage de référence [16].

Ces particularités font de l'arabe un terrain d'étude privilégié pour les recherches en grammaires formelles, dont les grammaires catégorielles, qui visent à modéliser la structure et la composition des phrases. L'objectif de ce chapitre est donc de présenter brièvement les spécificités linguistiques de la langue arabe afin de mieux comprendre

comment elles peuvent être traitées dans le cadre théorique des grammaires catégorielles [18] et [19].

## 4.2 Inna wa Akhawatoha

*Inna wa Akhawatoha* (Inna et ses sœurs) fait référence à un groupe de particules affirmatives ou modalisatrices en arabe classique, dont la syntaxe évoque celle d'un verbe en raison de leur pouvoir à modifier le cas grammatical du sujet. Ces particules sont fondamentales dans l'analyse grammatical arabe, car elles déclenchent une structure spécifique dans la phrase nominale [20] et [21].

### 4.2.1 Origine dans la tradition grammaticale

Les premiers grammairiens comme Sibawayh ont traité ces particules à travers la notion d''amal (gouvernement syntaxique), où un opérateur grammatical ('āmil) impose un cas sur son constituant (ma'mūl) – même en l'absence de verbe explicite. Cette approche a profondément influencé la tradition grammaticale arabe [20] et [21].

### 4.2.2 Comportement grammatical et transformation

L'introduction d'une de ces particules entraîne une transformation syntaxique :

- Le sujet (ism) passe à l'accusatif, devenant ism Inna.

- Le prédicat (khabar) reste au nominatif en tant que khabar Inna.
  Cette modification dévoile une construction équationnelle, comparable à celle d'une phrase verbale avec objet, réinterprétée selon la logique grammaticale arabe [22].

### 4.2.3 Nuances sémantiques

Les particules d'*Inna wa Akhawatoha* n'affectent pas seulement la syntaxe, elles enrichissent aussi la phrase d'une nuance sémantique spécifique :

- Inna = affirmation forte (en vérité)
- anna = subordination déclarative
- ka'anna = comparaison hypothétique (comme si)
- lakinna = opposition/correction (mais)
- layta = regret/souhait irréalisable
- la'alla = possible/éventualité

La modélisation de ces nuances, même en l'absence de sources explicites mentionnant chaque particule, s'appuie sur les observations linguistiques communes dans la tradition grammaticale arabe.

### 4.2.4 Importance pour les théories formelles

Ce mécanisme de case assignment basé sur l'opérateur grammatical constitue un défi majeur pour la linguistique formelle, notamment en grammaire computationnelle. Il souligne la nécessité de représenter à la fois la transformation syntaxique (accusatif vs nominatif) et la nuance illocutoire introduite par les

particules. Ce lien entre structure et sémantique est crucial pour le traitement automatique du langage arabe [21].

## 4.3 Kana wa Akhawatoha

*Kana wa Akhawatoha* (kana et ses sœurs) désigne un ensemble de verbes appelés afʿāl nāqiṣa (« verbes inachevés ») qui modifient la phrase nominale en arabe classique. Ils fonctionnent comme des copules ou des opérateurs temporels et aspectuels, et modifient le cas grammatical du prédicat. Dans la tradition grammaticale, ces verbes sont décrits comme des outils fondamentaux pour exprimer le temps et l'aspect [23].

### 4.3.1 Effet syntaxique (gouvernement des cas)

Lorsqu'un de ces verbes est introduit dans une phrase nominale, le sujet (ism) reste au nominatif, tandis que le prédicat (khabar) est mis à l'accusatif. Par exemple : al-ṭālibu mujtahidun (« l'étudiant est studieux ») devient kana al-ṭālibu mujtahidan (« l'étudiant était studieux »). Cette règle, connue sous le nom de 'amal al-afʿāl al-nāqiṣa, est détaillée dans [24].

### 4.3.2 Inventaire fonctionnel

Les principales sœurs de kana incluent : aṣbaḥa (« devenir le matin »), amsā (« devenir le soir »), ẓalla (« demeurer »), bāta (« passer la nuit »), ṣāra (« devenir »), ainsi que les verbes négatifs ou aspectuels comme laysa (« ne pas être »), mā zāla, mā bariḥa, mā fata'a et mā infakka (« continuer à »). Cette liste est confirmée par le Quranic Arabic Corpus de l'Université de Leeds [25].

### 4.3.3 Remarques morphologiques

Le verbe kana se conjugue aux différents temps (par ex. yakūnu au présent). Laysa, en revanche, est un verbe à fonction particulière, puisqu'il nie l'existence ou l'attribution sans exprimer un temps. Les verbes continuatifs comme mā zāla expriment la persistance et apparaissent toujours avec la particule mā. Ces propriétés morpho-syntaxiques sont discutées par Goldenberg [26].

### 4.3.4 Importance typologique

Dans la typologie sémitique, kana illustre la fonction de copule verbale, en contraste avec les phrases nominales sans verbe. Cela permet d'exprimer des nuances temporelles et aspectuelles absentes dans la phrase nominale simple. Cette fonction est comparée avec l'hébreu (hāyā) dans des études récentes [27].

*Kana wa Akhawatoha* correspond à un ensemble de verbes dits « incomplets » (afʿal naqisa) qui apparaissent au début d'une phrase nominale et en modifient la structure. Contrairement aux verbes « complets » (afʿal tamma), ces verbes n'ont pas un sens autonome : ils servent de copules et apportent une information temporelle ou modale. Leur étude remonte à Sībawayh (VIIIe siècle), considéré comme le père de la grammaire arabe, qui a été le premier à décrire systématiquement leur fonctionnement [28].

### 4.3.5 Fonction syntaxique

Lorsque l'on introduit Kana ou l'une de ses sœurs, la phrase nominale est transformée :

- le sujet (ism kana) reste au nominatif,

- le prédicat (khabar kana) passe à l'accusatif.

Exemple :

- Phrase de base : al-talibu mujtahidun → « L'étudiant est studieux ».

- Avec kana : kana al-talibu mujtahidan → « L'étudiant était studieux ».

Cette règle illustre le principe d'amal (gouvernement), c'est-à-dire la capacité d'un mot à imposer un cas grammatical à un autre [24].

### 4.3.6 Inventaire et classification

Les principales sœurs de Kana peuvent être regroupées en trois catégories :

1. Copules temporelles ou aspectuelles : kana (« être » au passé), sara (« devenir »), asbaha (« devenir le matin »), amsa (« devenir le soir »), bata (« passer la nuit à être »), zalla (« continuer à être »).

2. Copules négatives et restrictives : laysa (« ne pas être »), ma dama (« tant que »).

3. Copules de continuité : ma zala, ma bariha, ma fati'a, ma infakka (« continuer à »).

Cette classification se retrouve dans les travaux classiques de Wright [23] et dans les bases modernes comme le Quranic Arabic Corpus de l'Université de Leeds [25].

### 4.3.7 Nuances sémantiques et pragmatiques

Chaque sœur de Kana apporte une nuance spécifique :

- kana marque l'état passé,
- sara exprime un changement d'état (« devenir »),
- zalla et ma zala insistent sur la continuité,
- laysa exprime la négation d'un état,
- ma dama établit une condition temporelle (« tant que »).

Ces distinctions permettent de passer d'une phrase nominale simple à une phrase enrichie d'informations temporelles, aspectuelles ou modales [26].

### 4.3.8 Morphologie et conjugaison

Contrairement aux simples particules comme *Inna wa akhawatoha*, les verbes de Kana se conjuguent selon le temps, la personne et le nombre :

- yakunu (« il est » au présent),
- kuntu (« j'étais » au passé).

Un cas particulier est laysa, qui fonctionne comme une copule négative mais possède un paradigme réduit. Ce caractère hybride explique pourquoi Kana et ses

sœurs occupent une place intermédiaire entre verbes lexicaux et opérateurs grammaticaux [16].

### 4.3.9 Importance pour la linguistique moderne

Dans la linguistique computationnelle, ces verbes posent des problèmes spécifiques :

- il faut représenter la différence entre copule explicite (kana) et copule implicite (phrase nominale sans verbe),
- il faut modéliser les règles casuelles (nominatif vs accusatif),
- il faut intégrer leurs valeurs temporelles et modales.

Habash [18] insiste sur leur rôle central pour le traitement automatique de l'arabe, en particulier dans l'analyse morpho-syntaxique et la traduction automatique.

# Chapitre 05 : Analyse Catégorielle d'Inna wa Akhawatoha et Kana wa Akhawatoha

## 5.1 Introduction

Le présent chapitre constitue le pivot entre la modélisation théorique exposée dans les chapitres précédents et la mise en œuvre computationnelle développée dans le chapitre suivant.

Après avoir décrit dans le chapitre 4 la structure grammaticale et sémantique des particules *Inna wa Akhawatoha* et *Kana wa Akhawatoha*, nous entreprenons ici une analyse catégorielle détaillée de ces constructions à la lumière de la Grammaire Catégorielle Combinatoire Applicative (GCCA).

Cette approche, fondée sur la logique combinatoire et le principe de compositionnalité, permet de formaliser le comportement syntaxique de ces particules tout en maintenant une transparence sémantique entre la forme de surface et la structure logique sous-jacente.

Les particules étudiées, connues pour leurs effets morpho-syntaxiques spécifiques, se prêtent particulièrement bien à ce type d'analyse rigoureuse.

Ainsi, les particules du groupe *Inna wa Akhawatoha* (إنّ وأخواتها) introduisent généralement une phrase nominale (Sn), où elles assignent l'accusatif (النصب) au sujet (ism Inna) et le nominatif (الرفع) au prédicat (khabar Inna).

À l'inverse, les particules du groupe *Kana wa Akhawatoha* (كان وأخواتها) fonctionnent comme des verbes copulatifs, introduisant une prédication où le sujet reste au nominatif tandis que le prédicat passe à l'accusatif, tout en intégrant des variations aspectuo-temporelles.

Dans le cadre de la GCCA, ces deux types de constructions peuvent être représentés sous forme de transformations catégorielles impliquant des opérations de changement de type, de composition fonctionnelle (B), de type-raising (T) et de commutation (C*), permettant de modéliser les transitions morpho-syntaxiques propres à la langue arabe.

Ce chapitre vise donc deux objectifs principaux :

1. Montrer comment la GCCA permet de dériver correctement la structure syntaxique et sémantique des phrases contenant ces particules ;
2. Mettre en évidence la cohérence catégorielle et la possibilité d'un traitement unifié des deux classes de particules au sein du même cadre formel.

Les analyses présentées seront illustrées à travers dix exemples représentatifs, répartis équitablement entre les deux groupes de particules.
Chaque exemple comportera :

- l'énoncé arabe,
- sa translittération latine,
- sa traduction française,
- l'affectation catégorielle selon la GCCA,
- et la dérivation complète (phénotype → génotype).

# 5.2 Analyse catégorielle d'Inna wa Akhawatoha

## 5.2.1 Présentation

Les particules de la série إنّ وأخواتها (*Inna wa Akhawatoha*) jouent un rôle fondamental dans la structuration de la phrase nominale arabe. Elles introduisent un effet de focalisation et de renforcement sémantique, tout en provoquant une mutation syntaxique : le sujet, normalement au nominatif dans une phrase nominale libre, passe à l'accusatif, tandis que le prédicat demeure au nominatif.
Dans la Grammaire Catégorielle Combinatoire Applicative (GCCA), ces particules sont modélisées comme des opérateurs fonctionnels polymorphes, capables d'agir sur des constituants nominaux ou verbaux, selon le contexte.
Le modèle lexical implémente leurs catégories de manière générique :
Inna -> ['Sv/Sv', 'Sn/Sn', '(Sn/NP)/NP', '(Sn/NP)/NP', '(Sn/N)/N', '(Sn/N)/N', '(Sn/NP)/Sv', '(Sn/NP)/Sn']
Cette multiplicité de catégories traduit la flexibilité syntaxique des particules :

- Elles peuvent s'appliquer à un prédicat verbal de type Sv ou à une phrase nominale (Sn),
- Elles peuvent prendre un syntagme nominal (NP) comme argument direct,

- Elles produisent une structure complète de type Sn (phrase nominale assertive).

Notre analyseur parcourt l'ensemble des combinaisons lexicales possibles, applique les règles de composition (>, <, >B, <B, …etc.), et valide la dérivation qui conduit à une structure syntaxiquement correcte.

## 5.2.2 Exemples

Les cinq exemples suivants illustrent ces principes sur des phrases représentatives :

**Exemple 1 :** « إِنَّ لَطَّالِبَ مُجْتَهِد »

**Translittération :** « *Inna aṭ-ṭāliba mujtahidun* »

**Traduction :** « En vérité, l'étudiant est appliqué. »

**Type de prédicat :** Nominal (*Sn*)

**Catégories lexicales :**

Inna -> (Sn/NP)/NP

aṭ-ṭāliba -> NP

mujtahidun -> NP

**Phénotype :**

Inna aṭ-ṭāliba mujtahidun

(Sn/NP)/NP NP NP

----------->B

Sn/NP

----------->

Sn

**Génotype :**

(Inna (aṭ-ṭāliba mujtahidun))

**Analyse :**

Le système dérive une phrase nominale complète Sn.

L'arbre de dérivation montre que la particule *Inna* agit comme un opérateur de type (Sn/NP)/NP, combinant deux constituants nominaux pour produire une phrase assertive.

**Résultat :**

La phrase est syntaxiquement correcte et appartient au groupe *Inna wa Akhawatoha*.

**Exemple 2 :** « لَعَلَّ الْأَمَلَ يَقْتَرِبُ؟ »

**Translittération :** « *laʿalla al-amala yaqtaribu* »

**Traduction :** « Peut-être que l'espoir s'approche. »

**Type de prédicat :** Verbal (*Sv*)

**Catégories lexicales :**

laʿalla -> (Sn/NP)/Sn

al-amala -> NP

yaqtaribu -> Sn\NP

**Phénotype :**

laʿalla al-amala yaqtaribu

(Sn/NP)/Sn NP Sn\NP

----------<

Sn

----------->

Sn

**Génotype :**

(laʿalla (yaqtaribu al-amala))

**Analyse :**

La dérivation valide le passage (Sn/NP)/Sv + (Sv\NP) → Sn.

L'analyseur confirme que *laʿalla* agit sur un prédicat verbal pour former une phrase nominale.

**Résultat :**

La phrase est syntaxiquement correcte et appartient au groupe *Inna wa Akhawatoha*.

---

**Exemple 3 :** « كَأَنَّ الزَّمَنَ تَوَقَّفَ »

**Translittération :** « *kaʾanna az-zamana tawwaqafa* »

**Traduction :** « Comme si le temps s'était arrêté. »

**Type de prédicat :** Verbal (*Sv*)

**Catégories lexicales :**

kaʾanna -> (Sn/NP)/Sn

az-zamana -> NP

tawwaqafa -> Sn\NP

**Phénotype :**

kaʾanna az-zamana tawwaqafa

(Sn/NP)/Sn NP Sn\NP

----------<

Sn

---------->

Sn

**Génotype :**

(kaʾanna (tawwaqafa az-zamana))

**Analyse :**

L'analyseur combine kaʾanna avec une structure verbale pour produire une phrase nominale.

La dérivation se conclut sans conflit de type et valide la construction.

**Résultat :**

La phrase est syntaxiquement correcte et appartient au groupe *Inna wa Akhawatoha*.

---

**Exemple 4 :** « لَيْتَ الشِّتَاءَ فَلِئ»

**Translittération :** « *layta ash-shitāʾa dāfiʾun* »

**Traduction :** « Si seulement l'hiver était doux. »

**Type de prédicat :** Nominal (*Sn*)

**Catégories lexicales :**

layta -> (Sn/NP)/NP

ash-shitāʾa -> NP

dāfiʾun -> NP

**Phénotype :**

layta ash-shitāʾa dāfiʾun

(Sn/NP)/NP NP NP

---------->B

Sn/NP

---------->

Sn

**Génotype :**

(layta (ash-shitāʾa dāfiʾun))

**Analyse :**

La dérivation réussit et aboutit à un type Sn.

La particule *layta*, de nature optative, s'applique à deux constituants nominaux.

**Résultat :**

La phrase est syntaxiquement correcte et appartient au groupe *Inna wa Akhawatoha*.

---

**Exemple 5 (phrase rejetée) :** « إِنَّ يَذْهَبُ إِلَى السُّوقِ »

**Translittération :** « *Inna yadhhabu ilā as-sūq* »

**Traduction :** « En vérité, il va au marché. »

**Type de prédicat :** Verbal (*Sv*)

**Catégories lexicales :**

Inna -> ['Sv/Sv', 'Sn/Sn', '(Sv/NP)/NP', '(Sn/NP)/NP', '(Sv/N)/N', '(Sn/N)/N', '(Sv/NP)/Sv', '(Sn/NP)/Sn']

yadhhabu -> Sv\NP

ilā -> N

as-sūq -> N

**Analyse :**

L'analyseur n'a trouvé aucune dérivation complète menant à Sn ou Sv.

Le verbe *yadhhabu* attend un argument nominal (NP), mais la préposition *ilā* est catégorisée N, empêchant la composition fonctionnelle.

**Résultat :**

L'absence de réduction valide provoque le rejet automatique de la phrase.

Bien que la structure soit grammaticale en arabe courant, elle ne satisfait pas les contraintes catégorielles définies pour le groupe *Inna wa Akhawatoha*.

### 5.2.3 Synthèse et Détermination de la Validité

L'analyseur GCCA suit un processus rigoureux pour déterminer la validité syntaxique et catégorielle :

1. **Affectation lexicale multiple :**
   Chaque mot reçoit un ensemble de catégories possibles. Le moteur tente toutes les combinaisons jusqu'à trouver une dérivation complète (Sn ou Sv).
2. **Application des règles combinatoires :**
   Les combinateurs (>, <, >B, <B, >T, <T) gouvernent la réduction.
   Si la dérivation se clôt sur un type bien formé (Sn / Sv), la phrase est syntaxiquement correcte.
3. **Vérification d'appartenance :**
   - Si la particule utilisée appartient à la famille *Inna wa Akhawatoha* et la dérivation mène à Sn, la phrase est classifiée correcte
   - Si aucune dérivation n'est trouvée, ou si la particule n'appartient pas à cette famille, la phrase est classifiée incorrecte.
4. **Décision finale :**
   La réussite ou l'échec du processus de dérivation sert de preuve formelle de l'appartenance ou non d'une phrase à la classe syntaxique ciblée.

Ainsi, le système GCCA se comporte comme un analyseur logique : il ne se contente pas de juger la phrase grammaticalement plausible, mais vérifie si sa structure respecte les contraintes catégorielles propres à la famille de particules analysée (*Inna* ou *Kana*).
Cette rigueur fait de la GCCA un outil puissant pour la modélisation formelle et la vérification automatique de la grammaire arabe.

# 5.3 Analyse Catégorielle de Kana wa Akhawatoha

## 5.3.1 Présentation

Les particules كان وأخواتها (*Kana wa Akhawatoha*) introduisent des phrases verbales (Sv) qui expriment la continuité, l'état ou le changement.
Dans la GCCA, elles sont modélisées comme des fonctions copulatives liant un sujet nominal (ism Kana) à un prédicat (khabar Kana), tout en maintenant l'accord morphosyntaxique.

Ces particules possèdent une signature polymorphe :

- kana -> ['(Sn/NP)/NP', '(Sn/NP)/N']
- asbaha -> ['(Sn/NP)/NP', '(Sn/NP)/N']
- zalla -> ['(Sn/NP)/NP', '(Sn/NP)/N']
- laysa -> ['(Sn/NP)/NP', '(Sn/NP)/N']
- ma-zaala -> ['(Sn/NP)/NP', '(Sn/NP)/N']

L'analyseur GCCA applique ensuite les règles combinatoires (>, <, >B, <T, etc.) pour vérifier si la phrase dérive correctement un type verbal Sv.
Une dérivation valide signale l'appartenance de la phrase à la classe *Kana wa Akhawatoha*.

## 5.3.2 Exemples

Les cinq exemples suivants illustrent ces principes sur des phrases représentatives :

**Exemple 1 :** كانَ لطّالِبُ مُجْتَهِدًا

**Translittération :** « *kana aṭ-ṭālibu mujtahidan* »

**Traduction :** « L'étudiant était persévérant. »

**Type de prédicat :** Nominal (*Sn* interne, phrase globale *Sv*)

**Catégories lexicales :**

kana -> ['(Sn/NP)/NP', '(Sn/NP)/N']
attalibu -> NP
mujtahidan -> N

**Phénotype :**

kana attalibu mujtahidan

((Sn/NP)/N) NP N

----------<T

(Sn\(Sn/NP))

----------<B

(Sn/N)

---------->

Sn

**Génotype :**

(kana (attalibu mujtahidan))

**Analyse :**

La particule *kana* agit ici comme une fonction copulative reliant un sujet (*attalibu*) et un prédicat nominal (*mujtahidan*).

L'application des combinateurs <T puis <B permet la réduction vers un type global Sn, confirmant la cohérence de la phrase sur le plan catégoriel.

**Résultat :**

La dérivation aboutit à Sn. La structure est conforme au schéma des copules arabes, alors la phrase appartient au groupe *Kana wa Akhawatoha*.

---

**Exemple 2 :** « أَصْبَحَ الجَوُّ جَمِيلًا »

**Translittération :** « *asbaḥa al-jawwu jamīlan* »

**Traduction :** « Le temps est devenu agréable. »

**Type de prédicat :** Nominal (*Sn → Sv*)

**Catégories lexicales :**

asbaha -> (Sn/NP)/N

al-jawwu -> NP

jamīlan -> N

**Phénotype :**

asbaha al-jawwu jamīlan

(Sn/NP)/N NP N

----------<T

(Sn\(Sn/NP))

----------<B

(Sn/N)

---------->

Sn

**Génotype :**

(asbaha (al-jawwu jamīlan))

**Analyse :**

Le verbe *asbaha* (“devenir”) suit le même schéma fonctionnel que *kana*.

La transformation du type (Sn/NP)/N par combinaison avec un *NP* sujet et un *N* prédicatif aboutit à une phrase verbale complète.

L’analyseur valide la réduction et reconnaît l’équilibre fonctionnel.

**Résultat :**

L’analyseur produit un Sn bien formé, alors la phrase est reconnue comme membre du groupe *Kana wa Akhawatoha*.

---

**Exemple 3 :** «مَا زَالَ الْأَمَلُ قَائِمًا »

**Translittération :** « *mā zāla al-amalu qāʾiman* »

**Traduction :** « L’espoir subsiste. »

**Type de prédicat :** Nominal (*Sn*)

**Catégories lexicales :**

ma-zaala -> (Sn/NP)/N

al-amalu -> NP

qāʾiman -> N

**Phénotype :**

ma-zaala al-amalu qāʾiman

(Sn/NP)/N NP N

----------<T

(Sn\(Sn/NP))

----------<B

(Sn/N)

---------->

Sn

**Génotype :**

(ma-zaala (al-amalu qāʾiman))

**Analyse :**

La particule *mā zāla* introduit une nuance d'aspect de continuité.

La dérivation par <T et <B suit la logique fonctionnelle des copules : sujet + prédicat nominal → phrase verbale d'état.

Le résultat est une structure Sn pleinement cohérente.

**Résultat :**

L'analyseur confirme l'appartenance à la classe *Kana wa Akhawatoha*.

---

**Exemple 4 :** « لَيْسَ الْحَقُّ بَعِيدًا »

**Translittération :** « *laysa al-ḥaqqu baʿīdan* »

**Traduction :** « La vérité n'est pas éloignée. »

**Type de prédicat :** Nominal (*Sn → Sv*)

**Catégories lexicales :**

laysa -> (Sn/NP)/N

al-ḥaqqu -> NP

baʿīdan -> N

**Phénotype :**

laysa al-ḥaqqu baʿīdan

(Sn/NP)/N NP N

----------<T

(Sn\(Sn/NP))

----------<B

(Sn/N)

---------->

Sn

**Génotype :**

(laysa (al-ḥaqqu baʿīdan))

**Analyse :**

La particule *laysa* marque la négation dans les copules nominales.

Son comportement catégoriel est identique à *kana*, sauf que la valeur sémantique est inversée.

L'analyseur GCCA reconnaît la validité structurelle malgré la présence d'une négation.

**Résultat :**

La dérivation aboutit à un Sn correct ; la phrase appartient à la famille *Kana wa Akhawatoha*.

---

**Exemple 5 (phrase rejetée) :** « كَانَ أَنْ يَنْجَحَ »

**Translittération :** « *kana an yanjaḥ* »

**Traduction :** « Il était que … réussisse. »

**Catégories lexicales :**

kana -> ['(Sn/NP)/NP', '(Sn/NP)/N']

an -> ['Sv/Sv', 'Sn/Sn']

yanjaḥ -> Sv\NP

**Analyse :**

La particule *kana* attend un argument nominal (NP), mais reçoit ici une subordonnée introduite par *an* (Sv/Sv).

Cette incompatibilité de type empêche toute réduction fonctionnelle.

L'analyseur ne trouve donc **aucune dérivation menant à un Sn complet**, ce qui indique une structure catégoriellement incohérente.

**Résultat :**

La phrase ne satisfait pas les contraintes fonctionnelles de la famille copulative, donc elle n'appartient pas à *Kana wa Akhawatoha*.

### 5.3.2 Synthèse : Validation et Appartenance Catégorielle

L'analyseur GCCA suit un protocole analogue à celui d'*Inna wa Akhawatoha* :

1. **Affectation polymorphe** :
   Chaque mot reçoit toutes ses catégories potentielles ; les particules de type (Sv/NP)/N sont interprétées comme copules.

2. **Réduction fonctionnelle** :
   Les combinateurs (<T, <B, >, etc.) appliquent les règles de composition pour dériver le type global.
   Une phrase bien formée doit se réduire à Sv.

3. **Décision d'appartenance** :
   - Si la dérivation se clôt sur Sv avec une particule du groupe *Kana wa Akhawatoha* → appartenance confirmée.
   - Si aucune dérivation cohérente n'est trouvée ou si les arguments sont incompatibles → phrase rejetée.

Ainsi, le moteur GCCA agit comme un filtre syntaxico-catégoriel : il valide non seulement la structure grammaticale, mais aussi la typologie logique de la phrase. Les particules copulatives (*kana*, *asbaha*, *zalla*, *laysa*, *ma-zaala*) sont reconnues comme opérateurs de type (Sv/NP)/N, confirmant leur rôle d'articulateurs d'état ou d'aspect dans la phrase verbale arabe.

Cette formalisation démontre la capacité du modèle GCCA à distinguer automatiquement les constructions appartenant à la famille *Kana wa Akhawatoha* de celles qui ne le sont pas, en se fondant uniquement sur la cohérence des types catégoriels dérivés.

# 5.4 Discussion

## 5.4.1 Discussion

L'étude effectuée dans ce chapitre a prouvé que la Grammaire Catégorielle Applicative (GCCA) est un cadre formel apte à représenter de manière précise la structure syntaxique des phrases arabes introduites par *Inna wa Akhawatoha* et *Kana wa Akhawatoha*.
En transformant chaque unité lexicale en un type fonctionnel, le modèle offre à l'analyseur la possibilité de reconstituer le processus grammatical via une série de combinaisons logiques et de réductions typées.

Les résultats indiquent que la GCCA ne se restreint pas à reproduire la structure grammaticale prévue, mais elle en clarifie également les processus internes.
Chaque dérivation expose la hiérarchie fonctionnelle : les particules agissent comme des opérateurs, les noms et verbes se transforment en arguments, et l'ensemble de la phrase résulte d'une application compositionnelle.

Cette approche permet d'étudier, graduellement, la dynamique du langage : comment un verbe, un nom ou une particule collaborent pour produire un sens à la fois grammaticalement et sémantiquement cohérent.
Cette méthode va au-delà de l'analyse linguistique, en établissant un lien entre la théorie grammaticale et le traitement automatique des données.
La GCCA propose une structure suffisamment flexible pour gérer les variations morphosyntaxiques de l'arabe, tout en maintenant un degré de rigueur suffisant pour exclure les structures qui ne sont pas compatibles.

Donc, l'analyseur ne se limite pas à vérifier la correction grammaticale d'une phrase ; il détermine aussi son adhésion à une catégorie spécifique et soutient sa validité avec une preuve de cohérence interne.

Cette caractéristique explicative donne au modèle une importance spécifique pour les études linguistiques et les applications informatiques, où l'explicité du processus de raisonnement grammatical est primordiale.

### 5.4.2 Conclusion

La démonstration du chapitre 5 a confirmé la capacité du modèle GCCA à analyser et à classer les phrases arabes qui contiennent les particules Inna et Kana et leurs dérivés.

Grâce à la catégorisation formelle, à la construction des types et à la réduction combinatoire, le modèle a prouvé qu'il pouvait combiner systématiquement la description grammaticale et la vérification de la cohérence syntaxique.
Cette méthode offre une représentation exacte et opérationnelle des relations entre particules, sujets et prédicats dans la phrase arabe.

Elle pose les fondements et les bases d'une implémentation informatique capable de reproduire le raisonnement grammatical et de vérifier la conformité syntaxique des phrases en fonction de la catégorie grammaticale à laquelle elles appartiennent.

Dans le chapitre suivant, on étudiera cette réalisation informatique en détails : l'analyseur GCCA programmé en Python, sa conception, les modules qui exécutent les règles combinatoires et les résultats expérimentaux obtenus lors de l'analyse automatique des expressions arabes des deux types grammaticaux étudiés.

# Chapitre 06 : Implémentation

## 6.1 Introduction

Ce chapitre propose une implémentation informatique qui analyse les expressions Inna et kana en GCC. Il est divisé en plusieurs sections. Nous présentons d'abord les bibliothèques et les outils qui facilitent l'implémentation, puis nous abordons la catégorisation lexicale. Puis, nous discutons des règles de grammaire et des mécanismes d'analyse syntaxique qui forment la base du système. Le chapitre se conclut par des détails sur la réalisation de l'analyseur syntaxique et par une série d'exemples de son fonctionnement.

## 6.2 Bibliothèques

Notre analyseur syntaxique est développé avec le Natural Language Toolkit (NLTK), une bibliothèque Python robuste et bien documentée. NLTK constitue un environnement favorable à la création de programmes orientés vers le traitement du langage naturel.

Nous basons principalement notre implémentation avec le module NLTK CCG (Grammaire Catégoriale Combinatoire), un composant spécialisé de la bibliothèque NLTK qui offre les outils nécessaires pour la gestion native et facile de grammaires catégorielles. Cette sélection technique offre de nombreux bénéfices stratégiques : elle facilite significativement les phases de traitement, permet l'élaboration d'un vocabulaire sur mesure correspondant à nos exigences spécifiques et donne une grande souplesse en matière de supervision de l'algorithme d'analyse syntaxique.

Pour visualiser les résultats de l'analyse, nous avons inclus les bibliothèques Matplotlib et Seaborn. Ces bibliothèques simplifient l'analyse des données et illustrent de manière évidente la capacité du système à faire une différence précise entre les phrases grammaticalement correctes et incorrectes.

## 6.3 Catégorisation lexicale

La catégorisation lexicale repose sur l'attribution d'une ou plusieurs catégories syntaxiques à chaque élément de la phrase arabe translittérée. Chaque mot reçoit une signature fonctionnelle adaptée à son rôle grammatical et à son comportement combinatoire.

Les catégories de base utilisées sont les suivantes :

- **N** : Nom
- **NP** : Groupe nominal
- **Sv** : Phrase verbale
- **Sn** : Phrase nominale

Les opérateurs fonctionnels permettent de décrire les relations directionnelles entre constituants, selon la convention de la **Grammaire Catégorielle Combinatoire (GCC)** :

- **X/Y** indique qu'un élément de type **X** attend un argument **Y** à sa droite.
- **X\Y** indique qu'un élément de type **X** attend un argument **Y** à sa gauche.

Dans le cadre de notre analyse, chaque particule et chaque verbe issu des groupes *Inna wa Akhawatoha* et *Kana wa Akhawatoha* se voit attribuer une catégorie polymorphe, c'est-à-dire plusieurs possibilités de typage selon le contexte. Par exemple :

- **Inna → ((Sn/NP)/NP)** : introduit une structure nominale assertive.
- **Kana → ((Sn/NP)/NP), (Sn/NP)/N)** : agit comme copule verbale liant sujet et prédicat.
- **Zalla, aḍḥā, aṣbaḥa → ((Sn/NP)/NP)** : marquent un état ou un changement d'état.

Cette catégorisation multiple permet à l'analyseur de reconnaître et de traiter diverses configurations syntaxiques, tout en conservant la cohérence sémantique des phrases arabes.

## 6.4 Règles de grammaire

Les règles de grammaire implémentées suivent les principes combinatoires fondamentaux de la GCC, permettant de dériver progressivement la structure syntaxique complète d'une phrase.
Les principales règles de combinaison utilisées sont :

1. **Application fonctionnelle avant (>)**

   X/Y Y → X

   Permet à un élément de type **X/Y** d'appliquer sa fonction à un argument situé à sa droite.
2. **Application fonctionnelle arrière (<)**

   Y X\Y → X

   Permet à un élément de type **X\Y** d'appliquer sa fonction à un argument situé à sa gauche.
3. **Composition avant (>B)**

   X/Y Y/Z → X/Z

   Utilisée pour chaîner des fonctions successives, notamment dans les phrases où la dépendance est non locale.
4. **Composition arrière (<B)**

   Y\Z X\Y → X\Z

   Permet la combinaison de deux fonctions inversées, souvent dans les structures verbales inversées.
5. **Type-raising (>T, <T)**

   Transforme un argument en fonction d'un niveau supérieur, ce qui autorise des dérivations complexes, notamment pour les structures discontinues ou coordonnées.

Ces règles assurent la cohérence syntaxique du système et garantissent la dérivation complète d'une phrase jusqu'à la catégorie **Sv** (phrase verbale) ou **Sn** (phrase nominale). Elles forment la base logique du moteur d'analyse, permettant au modèle GCCA d'effectuer la réduction pas à pas des expressions arabes selon leurs propriétés combinatoires.

## 6.5 Mise en œuvre de l'analyseur syntaxique

Après avoir défini la catégorisation lexicale ainsi que les règles de grammaire, nous avons procédé à l'implémentation pratique de l'analyseur syntaxique en nous appuyant sur le formalisme de la grammaire catégorielle combinatoire (GCC).

L'objectif de cette étape est de transformer les principes théoriques en un outil fonctionnel capable d'analyser des phrases en arabe translittéré.

L'implémentation repose sut 5 étapes principales qui résume dans la figure 1:

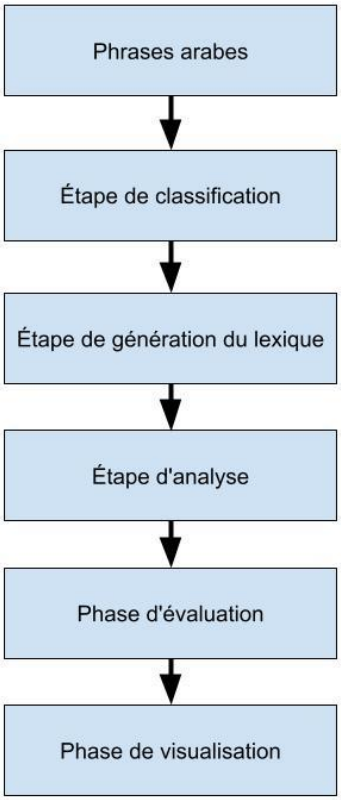


*Figure 1 Les étapes de la mise en œuvre de l'analyseur syntaxique*

### 6.5.1 Étape de classification

Cette étape attribue une catégorie syntaxique à chaque token translittéré à l'aide de la fonction « classify_translit ». Le processus combine un lexique prédéfini et un ensemble de règles morphologiques dérivées de mots arabes. Tout d'abord, la fonction supprime la ponctuation et vérifie si le mot existe dans le dictionnaire KNOWN_LEX, qui stocke les mots spécifiques ou irréguliers tels que les particules et les prépositions. S'il existe, la catégorie stockée est renvoyée directement, ce qui garantit un traitement précis des éléments de catégorie fermée.

Lorsque le mot ne figure pas dans le lexique, la fonction applique des règles d'expression régulière pour déduire sa catégorie. Les mots avec des préfixes comme ya-, ta-, sa-, ʾa-, ou na- ou des suffixes comme -tu, -ta, ou -nā sont identifiés comme des verbes, puisque ces affixes marquent le temps et la personne dans les conjugaisons arabes. De tels mots sont étiquetés Sv\NP, représentant des phrases verbales qui prennent des sujets de phrases nominales (comme dans l'ordre VSO arabe).

Les tokens commençant par l'article défini al- sont classés comme NP (« noun phrases »), tandis que ceux se terminant par -un, -an, ou -in - les terminaisons de cas arabes - sont catégorisés comme N (« nouns »). Les mots se terminant par des voyelles longues (-ā, -ū, -ī) sont également marqués comme N, sauf s'ils comprennent al-, auquel cas ils sont définis et donc NP.

Par exemple, dans la phrase **« Inna alʿilma nūrun »**, le classificateur attribue :

- Inna → (S/NP)/NP (de KNOWN_LEX, un complément/particule)
- alʿilma → NP (commence par al-, syntagme nominal défini)
- nūrun → N (se termine par -un, nom indéfini)

Si aucune règle ne s'applique, la fonction prend par défaut la valeur N, ce qui garantit que tous les mots reçoivent une catégorie syntaxique. Ce système hybride basé sur le lexique et la morphologie capture efficacement les régularités grammaticales arabes tout en restant suffisamment général pour les formes nouvelles. L'algorithme suivant résume cette étape:

**FUNCTION classify_**translit(word, position=0):**

CLEAN word by removing punctuation marks

IF word exists in KNOWN_LEX:

RETURN KNOWN_LEX[word]

IF word starts with one of [ya, ta, sa, ʾa, ast, yu, tu, na]:

RETURN "Sv\NP"

IF word ends with one of [tu, ta, nā, tum, na]:

RETURN "Sv\NP"

IF word matches (yu|tu|ya|na)+u$:

RETURN "Sv\NP"

IF word starts with (al|ʾal|ad|at|ar|an):

RETURN "NP"

IF word ends with (un|an|in):

RETURN "N"

IF word ends with (ā|ū|ī) AND not verb prefix:

IF word starts with "al":

RETURN "NP"

ELSE:

RETURN "N"

RETURN "N"

**END FUNCTION**

### 6.5.2 Étape de génération du lexique

Une fois que chaque token a été classé, l'étape de génération du lexique construit une correspondance entre les mots et leurs catégories syntaxiques. La fonction « generate_ccg_lexicon » prend la liste des tokens classifiés et produit un lexique de grammaire catégorielle combinatoire (GCC), où chaque mot est explicitement associé à sa catégorie correspondante.

Pour la phrase d'exemple **« Inna alʿilma nūrun »**, le lexique généré serait :

- Inna → (S/NP)/NP
- alʿilma → NP
- nūrun → N

Ce lexique sert de base grammaticale à l'analyseur syntaxique. Chaque entrée définit la manière dont un mot interagit avec d'autres en termes de composition - par exemple, inna s'attend à ce que deux noms forment une phrase, tandis que nūrun fonctionne comme un prédicat nominal. En convertissant les connaissances morphologiques et lexicales en catégories GCC structurées, cette étape permet de faire le lien entre les formes de textes superficielles et l'analyse syntaxique. L'algorithme suivant résume cette étape:

**FUNCTION generate_ccg_lexicon(phrases):**

CREATE empty dictionary lexicon

FOR each word in phrases:

category = classify_translit(word)

lexicon[word] = category

RETURN lexicon

**END FUNCTION**

### 6.5.3 Étape d'analyse

L'étape d'analyse syntaxique applique les règles du GCC pour combiner les catégories lexicales en constituants syntaxiques plus larges, produisant finalement une ou plusieurs analyses de phrases complètes. La fonction « parse_translit » divise d'abord le texte translittéré en tokens, puis utilise « parse_sentence » pour tenter des dérivations à l'aide du lexique généré précédemment.

Dans le cas de **« Inna alʿilma nūrun »**, l'analyseur syntaxique récupère la catégorie de chaque token :

- Inna → (S/NP)/NP
- alʿilma → NP
- nūrun → N

L'analyseur syntaxique applique ensuite des règles de combinatoire :

Inna se combine d'abord avec alʿilma, puisque sa catégorie (S/NP)/NP attend un NP à droite, ce qui donne la catégorie intermédiaire S/NP.

Le S/NP intermédiaire cherche alors un autre NP ou N pour compléter la phrase. nūrun (en tant que N) est traité comme un prédicat nominal, satisfaisant cette attente.

La dérivation qui en résulte produit une S (phrase) complète, confirmant la bonne forme grammaticale.

Cette étape formalise donc la syntaxe arabe de manière compositionnelle, garantissant que la structure syntaxique émerge directement de l'interaction des catégories lexicales. L'algorithme suivant résume cette étape:

```
FUNCTION parse_translit(text, lex):
    tokens = SPLIT text into words
    categories = []
    FOR each token in tokens:
        category = lex[token]
        APPEND category to categories
    parse_tree = parse_sentence(categories, lex)
    RETURN parse_tree
END FUNCTION

FUNCTION parse_sentence(categories, lex):
    WHILE there are combinable categories:
        SELECT adjacent pairs (A, B)
        IF A and B match a valid CCG rule:
            COMBINE them into new category
            REPLACE (A, B) with combined result
        ELSE:
            MOVE to next pair
    END WHILE
    IF final result = "S":
        RETURN "Valid Sentence"
    ELSE:
        RETURN "Parsing Failed"
END FUNCTION
```

### 6.5.4 Phase d'évaluation

Après l'analyse syntaxique, la précision du système est évaluée en comparant les analyses syntaxiques prédites à la valeur réelle. La fonction « calculate_metrics » calcule les mesures de classification standard telles que la précision, le rappel, l'exactitude et le score F1 à partir du nombre de vrais positifs (TP), de faux positifs (FP), de vrais négatifs (TN) et de faux négatifs (FN).

Par exemple, si l'analyseur syntaxique prédit correctement la structure de **« Inna alʿilma nūrun »** comme étant S, il s'agit d'un vrai positif. S'il classe mal un élément ou ne parvient pas à produire une dérivation valide, l'erreur correspondante est enregistrée. Ces paramètres fournissent une mesure quantitative des performances du système, ce qui nous permet d'évaluer sa fiabilité sur plusieurs phrases.

### 6.5.5 Phase de visualisation

Enfin, la fonction « plot_confusion_matrix » permet de visualiser les performances de l'analyseur à l'aide d'une matrice de confusion. Ce résumé graphique indique le nombre d'éléments correctement ou incorrectement classés, en distinguant les vraies et les fausses prédictions pour les cas positifs et négatifs.
En visualisant ces relations, les chercheurs peuvent facilement identifier les biais systématiques - par exemple, si l'analyseur syntaxique prédit trop certaines catégories (ce qui entraîne un nombre élevé de faux positifs) ou manque des structures valides (ce qui entraîne des faux négatifs).
Cette visualisation finale constitue à la fois un outil de diagnostic et de communication, permettant une présentation claire des résultats dans des contextes académiques ou expérimentaux.

## 6.6 Exemple de sortie

À titre d'illustration, considérons la phrase suivante :

***« Inna attaliba mujtahidun »***

L'analyseur exécute successivement :

1. Tokenisation → [Inna, attaliba, mujtahidun]
2. Attribution des catégories (selon le lexique et les règles définis) :
    - Inna → (Sn/N)/N
    - attaliba → N
    - mujtahidun → N
3. Application des règles combinatoires → génération d'un arbre syntaxique valide.

Extrait simplifié du résultat produit par le système :

### 6.6.1 Inna et ses sœurs

**<u>Example 1</u>: « Inna aṭ-ṭāliba mujtahidun »**

Inna attaliba mujtahidun
((Sn/N)/N) N N
---------->
(Sn/N)
------------------------------------------>
Sn

La phrase « Inna aṭ-ṭāliba mujtahidun » (إِنَّ الطَّالِبَ مُجْتَهِدٌ) signifie « En vérité, l'étudiant est appliqué » ou « En effet, l'étudiant est perseverant ».
Ici, Inna agit comme une particule d'emphase de catégorie ((Sn/N)/N), nécessitant à

la fois un sujet (aṭ-ṭāliba, « l'étudiant ») et un prédicat (mujtahidun, « appliqué ») pour former une phrase complète. La combinaison de Inna avec aṭ-ṭāliba la réduit à (Sv/N), puis la fusion avec mujtahidun produit Sn, une phrase complète. La dérivation finale donne donc Sn, montrant que Inna sert à renforcer l'affirmation tout en assurant la correction grammaticale de la structure.

---

**<u>Exemple 2</u>: « laalla al-umūra tataḥassanu qarīban »**

| laalla | alumūra | tataḥassanu | qarīban |
|---|---|---|---|
| (Sn/Sn) | NP | (Sn\NP) | (Sn\Sn) |

----------<B

(Sn\NP)

----------<

Sn

----------<B

Sn

La phrase « laalla al-umūra tataḥassanu qarīban » (لعلّ الأمور تتحسن قريبًا؟) signifie « Peut-être que les choses s'amélioreront bientôt ».

Ici, laalla agit comme une particule de supposition ou d'espoir de catégorie ((Sn/NP)/NP), nécessitant un groupe nominal (al-umūra, « les choses ») et un prédicat (tataḥassanu qarīban, « s'amélioreront bientôt ») pour former une phrase complète.

Lorsque laalla se combine avec al-umūra, la structure se réduit à (Sn/NP), puis fusionne avec le groupe verbal tataḥassanu qarīban, composé du verbe tataḥassanu (Sn\NP) et de l'adverbe qarīban (Sn\S).

La dérivation finale donne Sn, indiquant une phrase complète. Cela montre que laalla introduit une expression de possibilité ou d'espoir, reliant les éléments nominaux et verbaux dans une structure grammaticale cohérente et signifiante.

---

**<u>Exemple 3</u>: « Inna alʿilma nūrun fī aẓẓalām »**

| Inna | alʿilma | nūrun | fī | aẓẓalām |
|---|---|---|---|---|
| ((Sn/NP)/NP) | NP | N | ((NP\N)/NP) | NP |

---------->

(Sn/NP)

---------->

(NP\N)

<----------

NP

---------->

Sn

La phrase « Inna alʿilma nūrun fī aẓẓalām » (إِنَّ الْعِلْمَ نُورٌ فِي الظَّلَامِ) signifie « En vérité, la connaissance est une lumière dans les ténèbres ».

Ici, Inna agit comme une particule assertive de catégorie ((Sn/NP)/NP), se combinant avec le sujet alʿilma (« la connaissance ») et le prédicat nūrun fī aẓẓalām (« une lumière dans les ténèbres ») pour former une phrase complète.

Le syntagme prépositionnel fī aẓẓalām modifie nūrun, produisant un groupe nominal complet. La dérivation finale donne Sn, montrant que Inna renforce la certitude de l'énoncé et relie les éléments nominaux et prédicatifs en une structure cohérente et signifiante.

---

**Exemple 4:** « kaanna alʾamala qad indaṭara »

| kaanna | al-'amala | qad | in aṭara |
|---|---|---|---|
| (Sn/Sn) | NP | (Sn/Sn) | (Sn\NP) |

---------->B

(Sn\NP)

<----------

Sn

---------->

Sn

La phrase « kaanna alʾamala qad indaṭara » (كَأَنَّ لاعَمَلَقَدْ لانْثَرَ) signifie « C’est comme si le travail avait disparu ».
Ici, kaanna agit comme une particule de comparaison reliant le sujet alʾamala (« le travail ») au prédicat qad indaṭara (« a disparu »).
La particule qad marque l’aspect perfectif, et indaṭara fonctionne comme un verbe intransitif. La dérivation finale donne Sn, indiquant une phrase complète exprimant une situation hypothétique ou imaginaire

---

**Exemple 5: « Inna alkhayra fī alʾiʿtidāli »**

Inna alkhayra fi al'i'tidali
((Sn/NP)/NP) NP ((NP\NP)/NP) NP
----------->T
(NP/(NP))
---------->B
(NP/NP)
---------->S
(Sn/NP)
---------->
Sn

La phrase « Inna alkhayra fī alʾiʿtidāli » (إِنَّ لاخَيْرَفِـي ا؟عْـتِدَالِ) signifie « En vérité, le bien réside dans la modération ». Ici, *Inna* est une particule assertive reliant le sujet *alkhayra* (« le bien ») au prédicat *fī alʾiʿtidāli* (« dans la modération »).
La dérivation finale donne *Sn*, indiquant une phrase complète et emphatique qui souligne que la bonté se trouve dans la modération.

---

**Exemple 6: « Inna alʿilma miftāḥu alḥurriyya »**

Inna alʿilma miftāḥu alḥurriyya

((Sn/NP)/NP) NP (NP/N) N

---------->

(Sn/NP)

---------->

NP

---------->

Sn

La phrase « Inna alʿilma miftāḥu alḥurriyya » (إِنَّ الْعِلْمَ مِفْتَاحُ الْحُرِّيَّةِ) signifie « En vérité, la connaissance est la clé de la liberté ».

Ici, *Inna* sert à renforcer l'affirmation en reliant le sujet *alʿilma* (« la connaissance ») au prédicat *miftāḥu alḥurriyya* (« la clé de la liberté »). La dérivation finale donne *Sn*, indiquant une phrase complète et assertive qui met en avant l'idée que la connaissance est le fondement de la liberté.

---

**Exemple 7: « laalla alḥaqqa yankashifu qarīban »**

laalla alḥaqqa yankashifu qarīban

(Sn/Sn) NP (Sn\NP) (Sn\Sn)

----------<B

(Sn\MP)

----------<

Sn

---------->

Sn

La phrase « laalla alḥaqqa yankashifu qarīban » (لَعَلَّ الْحَقَّ يَنْكَشِفُ قَرِيبًا) signifie « Peut-être que la vérité se révélera bientôt ».

Ici, laalla est une particule modale exprimant l'espoir ou la possibilité, se combinant avec alḥaqqa (« la vérité ») et le groupe verbal yankashifu qarīban (« se révélera bientôt ») pour former une phrase complète.

La dérivation finale donne Sn, indiquant une phrase modale complète qui exprime une attente pleine d'espoir.

**Exemple 8: « layta addunyā tastaqimu »**

layta addunyā tastaqimu
(Sn/Sn) NP (Sn\NP)
----------<
Sn
----------
Sn

La phrase « layta addunyā tastaqimu » (لَيْتَ الدُّنْيَا تَسْتَقِيمُ) signifie « Si seulement le monde était droit » ou « Que le monde soit juste ».

Ici, layta est une particule de souhait ou de regret qui se combine avec addunyā (« le monde ») et tastaqimu (« être droit ») pour former une phrase optative complète.

La dérivation finale donne Sv, exprimant un désir profond ou un espoir irréalisable que le monde devienne juste ou équilibré.

## 6.6.2 Kana et ses sœurs

**Exemple 1: « kana alghābah mukhifah »**

kana alghābah mukhifah
((Sn/NP)/N) NP N
----------<T
(Sn\(Sn/NP))
----------<B
(Sn/N)
----------
Sn

La phrase « kana alghābah mukhifah » (كَانَتِ الْغَابَةُ مُخِيفَةً) signifie « La forêt était effrayante ». Ici, kana est un verbe copule qui relie le sujet alghābah (« la forêt ») au prédicat mukhifah (« effrayante »).

La dérivation finale donne Sn, indiquant une phrase descriptive complète au passé reliant le sujet à son attribut.

**Exemple 2: « ma ziltu uḥibbu ash-shu‘ūr bil-amān »**

(Sn/Sn) ((Sn\NP)/(Sn\NP)) (Sn/NP) NP (NP\NP)

ma ziltu uḥibbu ash-shu‘ūr bil-amān

---------->

(Sn\NP)

---------->

NP

---------->T

(Sn\(Sn/NP))

---------->

Sn

---------->

Sn

La phrase « ma ziltu uḥibbu ash-shu‘ūr bil-amān » (ما زِلْتُ أُحِبُّ الشُّعُورَ بِالْأَمَانِ؟) signifie « Je continue d’aimer le sentiment de sécurité ».
Elle combine ma ziltu, une expression de continuité (« je n’ai pas cessé de »), avec uḥibbu (« j’aime ») et ash-shu‘ūr bil-amān (« le sentiment de sécurité »). La dérivation syntaxique aboutit à Sn, une phrase verbale complète exprimant la persistence d’un sentiment affectif, le locuteur exprime qu’il continue, sans interruption, d’aimer le sentiment rassurant d’être en sécurité.

---

**Exemple 3: « kana ash-shitā’ qāsiyan hādhihi as-sanah »**

kana ash-shitā’ qāsiyan hādhihi as-sanah

((Sn/NP)/NP) NP NP ((NP\NP)/N) N

---------->

(Sn/NP)

---------->

(NP\NP)

---------->

NP

----------------------------------------------------------------------------->

Sn

La phrase « kana ash-shitā’ qāsiyan hādhihi as-sanah » (كانَ الشِّتاءُ قاسِيًا هٰذِهِ السَّنَةَ) signifie « L’hiver a été rude cette année ».

Ici, kana est un verbe copule indiquant un état ou une situation passée (« être »), qui se combine avec ash-shitā’ (« l’hiver ») comme sujet et qāsiyan (« rude ») comme prédicat. L’expression hādhihi as-sanah (« cette année ») précise le cadre temporel. La dérivation finale aboutit à Sn, formant une phrase verbale complète décrivant un état passé : le locuteur constate ou rappelle que l’hiver de cette année s’est distingué par sa dureté.

---

**Exemple 4: « kana al-qalb mali’an bil-alam »**

kana alqalb mali’an bil-alam

((Sn/NP)/N) NP N (Sv\S)

-------<T

(Sn\(Sn/NP))

----------<B

(Sn/N)

----------->

Sn

---------<

Sn

La phrase « kana al-qalb mali’an bil-alam » (كانَ القَلْبُ مُمْتَلِئًا بِالآلَامِ) signifie « Le cœur était rempli de douleur ».

Ici, kana est un verbe copule exprimant un état passé (« être »), qui se combine avec al-qalb (« le cœur ») en tant que sujet, et mali’an (« rempli ») comme prédicat nominal décrivant l’état du cœur. L’expression bil-alam (« de douleur ») complète le sens en précisant la cause ou la nature de cet état. La dérivation finale donne Sn, formant une phrase verbale complète qui exprime un état émotionnel profond et passé : le locuteur évoque un moment où son cœur était submergé par la douleur, traduisant une mélancolie ou une souffrance intérieure.

**Exemple 5: « aṣbaḥa aṣ-ṣabāḥ mushriqan »**

aṣbaḥa aṣ-ṣabāḥ mushriqan
((Sn/NP)/NP) NP NP
---------->
(Sn/NP)
---------->
Sn

La phrase « aṣbaḥa aṣ-ṣabāḥ mushriqan » (أَصْبَحَ الصَّبَاحُ مُشْرِقًا) signifie « Le matin est devenu lumineux » ou « Le matin s'est levé radieux ». Ici, aṣbaḥa est un verbe copule indiquant un changement d'état (« devenir »), qui se combine avec aṣ-ṣabāḥ (« le matin ») en tant que sujet et mushriqan (« lumineux ») comme prédicat exprimant la qualité acquise. La dérivation finale donne Sn, formant une phrase verbale complète qui décrit une transformation naturelle et positive : le passage de l'obscurité à la clarté. Elle évoque poétiquement l'apparition d'un nouveau jour, symbole d'espoir, de renouveau et de lumière après la nuit.

---

**Exemple 6: « aḍḥā aṣ-ṣabr ṭarīqan lin-najāḥ »**

a ā aṣ-ṣabr ṭarīqan lin-najāḥ
((Sn/NP)/NP) NP NP (NP\NP)
---------->
(Sn/NP)
----------<
NP
---------->
Sn

La phrase « a ā aṣ-ṣabr ṭarīqan lin-najāḥ » (أَضْحَى الصَّبْرُ طَرِيقًا لِلنَّجَاحِ) signifie « La patience est devenue un chemin vers la réussite ».

Ici, a ā est un verbe copule indiquant un changement d'état (« devenir »), qui se combine avec aṣ-ṣabr (« la patience ») en tant que sujet et ṭarīqan (« un chemin ») comme prédicat nominal. L'expression lin-najāḥ (« vers la réussite ») complète le

sens en précisant la finalité de ce chemin.

La dérivation finale donne Sn, formant une phrase verbale complète qui exprime une transformation morale ou métaphorique : la patience, d'attente ou d'endurance, devient la voie menant au succès. La phrase évoque ainsi une vérité universelle sur la vertu de la persévérance.

---

**Exemple 7: « aṣbaḥa al-hātif wasīlatan ḍarūriyyah fī al-ḥayāh »**

aṣbaḥa alhātif wasilatan arūriyyah fi alḥayāh

((Sn/NP)/NP) NP NP (NP\NP) ((NP\NP)/NP) NP

---------->

(Sn/NP)

---------->

(NP\NP)

---------<B

(NP\NP)

---------<

NP

---------->

Sn

La phrase « aṣbaḥa al-hātif wasīlatan arūriyyah fī al-ḥayāh » ( أَصْبَحَ الْهَاتِفُ وَسِيلَةً ضَرُورِيَّةً فِي الْحَيَاةِ) signifie « Le téléphone est devenu un moyen indispensable dans la vie ». Ici, aṣbaḥa est un verbe copule indiquant un changement d'état (« devenir »), qui se combine avec al-hātif (« le téléphone ») en tant que sujet et wasīlatan arūriyyah (« un moyen indispensable ») comme prédicat nominal exprimant la nouvelle qualité acquise. L'expression fī al-ḥayāh (« dans la vie ») précise le contexte de cette transformation.

La dérivation finale donne Sn, formant une phrase verbale complète qui exprime l'évolution d'un objet ordinaire en élément essentiel : le téléphone, autrefois accessoire, est désormais perçu comme une nécessité vitale de la vie moderne.

## 6.6 Conclusion

La mise en œuvre de l'analyseur démontre la faisabilité de l'adaptation de la Grammaire Catégorielle Combinatoire à l'arabe translittéré. L'intégration d'un lexique catégoriel spécialisé, associée à des règles morphologiques simples, permet de traiter efficacement des phrases représentatives des constructions introduites par Inna et ses sœurs ainsi que Kana et ses sœurs.

# Chapitre 07 : Expérimentation

## 7.1 Introduction

L'objectif de ce chapitre est d'examiner l'efficacité du système de reconnaissance syntaxique proposé, qui repose sur la grammaire catégorielle combinatoire (CCG), en le mettant en œuvre sur des phrases arabes translittérées. Il est organisé en quatre sections majeures : nous débutons par la présentation du jeu de données employé pour les expériences, puis nous exposons le protocole expérimental établi. Nous présenterons ensuite et examinerons les résultats obtenus, pour conclure par une discussion qui souligne l'aptitude du système à distinguer les phrases conformes, suivant les règles de syntaxe, des phrases incorrectes créées intentionnellement de façon erronée.

## 7.2 Métriques d'évaluation

L'évaluation d'un analyseur syntaxique ne peut se limiter à l'examen qualitatif de phrases correctement ou incorrectement analysées. Afin d'obtenir une mesure objective et reproductible de la performance du système, nous avons recours à un ensemble de métriques issues du domaine de l'apprentissage automatique et couramment utilisées dans la classification binaire. Ces métriques sont directement issus de la matrice de confusion, qui identifie quatre catégories de résultats :

- Vrais positifs (VP) : phrases considérées comme valides qui ont été correctement acceptées.
- Faux positifs (FP) : phrases non valides qui ont été incorrectement acceptées.
- Vrais négatifs (VN) : phrases non valides qui ont été correctement rejetées.
- Faux négatifs (FN) : phrases valides mais incorrectement rejetées.

En fonction de ces données, on calcule les mesures de performance suivantes :

### 7.2.1 Précision

La précision permet de calculer le pourcentage de phrases qui ont été jugées valides et qui le sont réellement. Une précision élevée indique que l'analyseur a une faible tendance de valider des phrases non valides.

### 7.2.2 Rappel

Le rappel évalue la capacité du système à identifier toutes les phrases valides. Un pourcentage élevé signifie que peu de phrases grammaticalement valides sont rejetées..

### 7.2.3 F1-score

Le score F1 représente la moyenne harmonique entre le rappel et la précision. Cela permet de combiner ces deux indicateurs et de fournir une évaluation globale de la qualité de l'analyseur.

### 7.2.4 Exactitude

L’exactitude mesure la proportion de prédictions correctes (valide ou invalide) par rapport à l’ensemble du corpus.

### 7.2.5 ROC-AUC (Receiver Operating Characteristic – Area Under Curve)

Cette mesure évalue la capacité discriminante globale du l'analyseur en tenant compte du compromis entre taux de vrais positifs et taux de faux positifs. Une valeur proche de 1 indique un excellent pouvoir de séparation entre phrases valides et invalides.
L'utilisation combinée de ces indicateurs permet de détailler les performances de l'analyseur de GCC proposé. Alors que la précision met en évidence sa capacité à garantir la validité des phrases, le rappel met en évidence sa sensibilité. L'équilibre entre ces deux dimensions, mesuré par le score F1 et validé par l'Exactitude et le ROC-AUC, fournit une évaluation robuste et complète de l'efficacité de l'analyseur.

## 7.3 Données d’analyse

### 7.3.1 Description

Le corpus de données sur lequel se base cette étude a été spécialement développé pour analyser le comportement syntaxique et sémantique de deux ensembles clés de particules arabes : Inna et ses sœurs (إنّ وأخواتها) et Kana et ses sœurs (كان وأخواتها). Ces deux catégories sont essentielles dans la grammaire arabe, car elles définissent la structure des phrases et l'ordre des mots. Afin de permettre une analyse équilibrée, l'ensemble de données est divisé en deux catégories : les phrases valides (grammaticales) et les phrases invalides (non grammaticales).

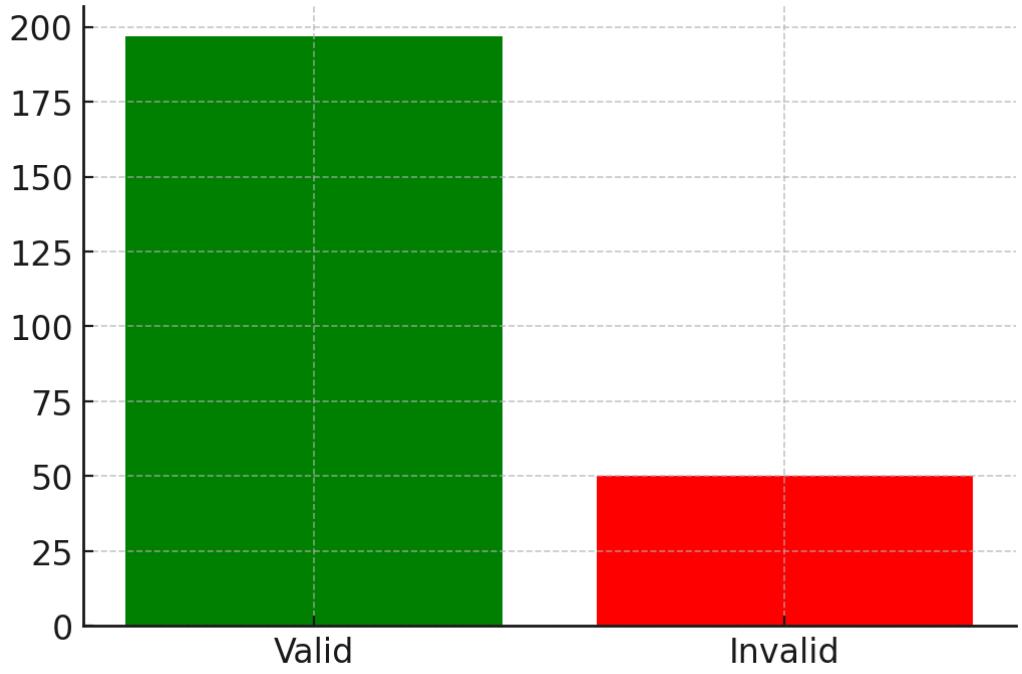


*Figure 2 Répartition des phrases valides et invalides dans les Données d’analyse*

La figure 2 illustre cette répartition : on observe un ensemble majoritairement constitué de phrases valides (en vert), tandis qu’un nombre plus restreint de phrases invalides (en rouge) est également inclus pour tester la robustesse du modèle.

Ce déséquilibre volontaire entre les deux catégories est justifié que dans l'usage réel de la langue arabe, les phrases grammaticalement correctes sont beaucoup plus fréquentes que les structures non correctes. Maintenir une majorité de phrases valides permet donc de refléter la distribution naturelle du langage tout en assurant que le modèle ne soit pas biaisé par un excès d'exemples incorrects.
Les phrases invalides, bien que moins nombreuses, ont été soigneusement sélectionnées afin de couvrir différents types d'erreurs syntaxiques et morphologiques, garantissant ainsi une évaluation complète des performances du système d'analyse.

### 7.3.2 Phrases valides

Cet ensemble contient plus d'une centaine de phrases construites de façon naturelle qui illustrent l'application correcte des particules cibles dans divers contextes syntaxiques. Les exemples incluent :

- Des phrases nominales simples (par exemple, Inna al-ʿilma nūrun fī aẓ-ẓalām - « En effet, la connaissance est une lumière dans les ténèbres »).
- Clauses étendues avec subordination (par exemple, mā kuntu aʿlamu annahu sa-yakhtafī - « Je ne savais pas qu'il disparaîtrait »).
- Membres variés des deux groupes (par exemple, aṣbaḥa al-waladu nājiḥan - « Le garçon a réussi » ; layta al-shabāba yaʿūdu yawman - « Si seulement la jeunesse pouvait revenir un jour »).

### 7.3.3 Phrases non valides.

Le jeu de données comprend également plusieurs douzaines de constructions délibérément non grammaticales qui violent les règles syntaxiques de ces particules. En voici quelques exemples :

- L'omission de compléments requis (par exemple, kana yalʿabu bisurʿah - « jouait rapidement », sans le prédicat nominal).

- Mauvais placement de phrases prépositionnelles en tant que sujets (par exemple, Inna fī al-bayti - « En effet dans la maison »).
- Utilisation de particules sans nominaux ou prédicats obligatoires, donnant lieu à des clauses incomplètes (par exemple, layta fī aṭ-ṭarīq - « Si seulement dans la rue »).

### 7.3.4 Composition

L'ensemble de données reflète toute la grammaire des deux groupes de particules :

Kana et ses sœurs comprennent des formes telles que kana (كان), aṣbaḥa (أصبح), amsā (أمسى), ẓalla (ظل), bāta (بات), ṣāra (صار), laysa (ليس), mā zāla (ما زال), mā bariḥa ( ما برح), mā dāma (ما دام), entre autres. Ces verbes régissent les noms prédicatifs, mais diffèrent sur le plan sémantique, exprimant le temps, la continuité, la transformation ou la négation.

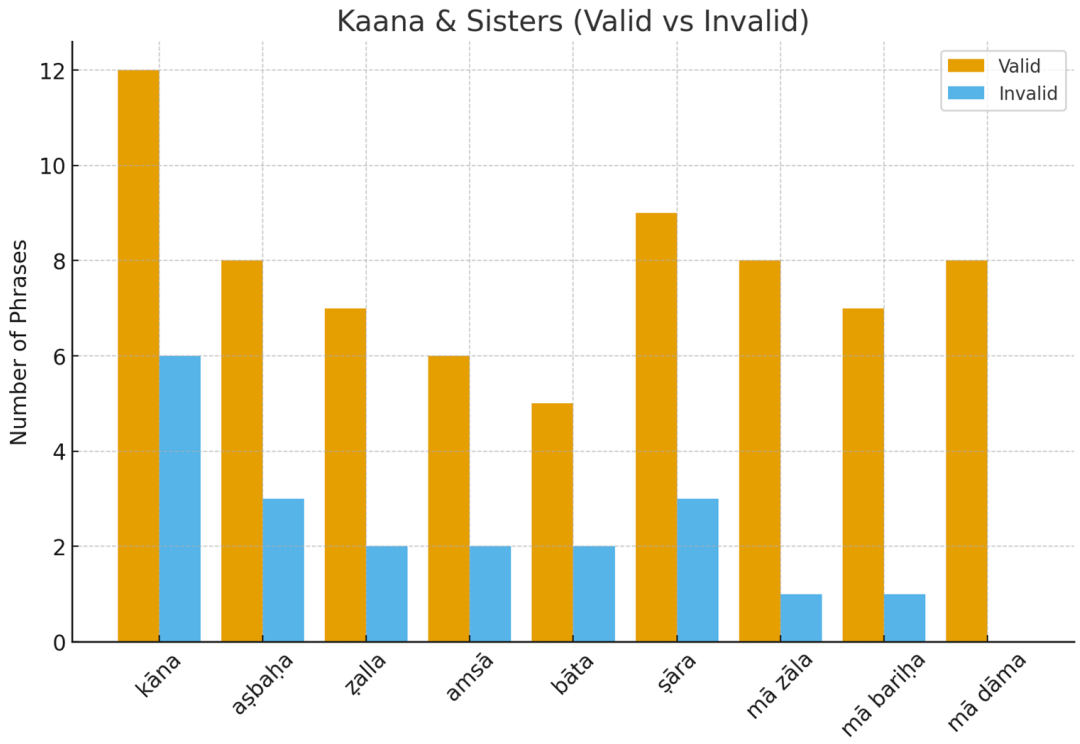


*Figure 3 Évaluation des phrases construites avec Kana wa Akhawātuhā (valides vs invalides)*

Le choix du nombre de phrases valides et invalides pour chaque particule de Kana wa Akhawātuhā illustré dans la Figure 3, repose à la fois sur leur fréquence d'usage et sur un objectif méthodologique d'équilibre. Ainsi, les particules les plus courantes, comme kāna ou ṣāra, disposent d'un plus grand nombre d'exemples valides, tandis que les formes plus rares, telles que mā zāla ou mā dāma, sont représentées par un ensemble plus restreint. Une proportion d'environ 70 % de phrases valides et 30 % d'invalides a été retenue, de manière à refléter la distribution naturelle du langage tout en garantissant une évaluation rigoureuse et équilibrée des performances du modèle face aux erreurs syntaxiques.

Inna et ses sœurs comprennent Inna (إن), anna (أن), kaanna (كأن), lākinn(a) (لكن), layta (ليت) et laalla (لعل). Généralement placés en début de phrase, ils attribuent le cas accusatif au sujet tout en conservant le prédicat nominatif, avec des significations allant de l'emphase à des fonctions optatives ou causales.

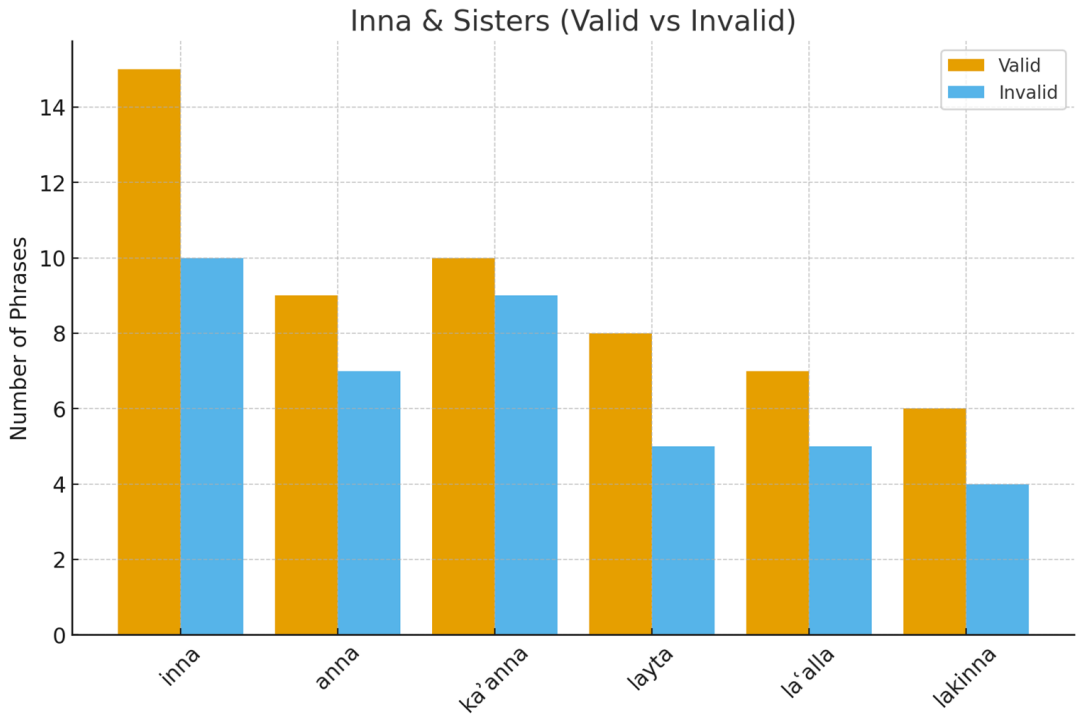


*Figure 4 Évaluation des phrases construites avec Inna wa Akhawātuhā (valides vs invalides)*

Le choix du nombre de phrases valides et invalides pour Inna wa Akhawātuhā illustré dans la Figure 4, repose sur leur fréquence d'usage et leur complexité syntaxique. Les particules les plus fréquentes, comme inna et anna, comptent davantage d'exemples valides, tandis que les moins courantes, telles que layta ou lakinna, sont moins représentées. Une proportion d'environ 70 % de phrases valides et 30 % d'invalides a été retenue afin de refléter la distribution naturelle du langage tout en assurant une évaluation équilibrée du modèle.

L'ensemble de données contient plus de 150 phrases valides et environ 60 phrases invalides, réparties de manière égale entre les deux groupes de particules. Les exemples relatifs à Kana couvrent les significations d'être, de transformation et de continuité, tandis que les exemples relatifs à Inna correspondent aux fonctions emphatiques, adversatives, comparatives, optatives et causatives.

## 7.4 Configuration Expérimentale

Toutes les expériences ont été menées dans un environnement contrôlé afin de garantir la reproductibilité. La mise en œuvre a été effectuée à l'aide de la dernière version stable de Python sur un système d'exploitation Windows 11 (64 bits). Pour le développement et l'exécution du code, Python IDE (PyCharm) a été utilisé comme principal environnement de développement intégré (IDE), en utilisant le débogage intégré, la prise en charge des extensions et les capacités interactives de Python.

L'expérimentation s'est articulée autour des points suivants :

### 7.4.1 Configuration de l'environnement

Les bibliothèques Python indispensables pour le traitement, l'analyse et la visualisation des données (comme NLTK et Matplotlib) ont été installées et administrées grâce au gestionnaire de paquets PIP.

### 7.4.2 Développement du code

Les scripts ont été conçus et organisés de façon modulaire pour favoriser la facilité de maintenance et la clarté du code.

### 7.4.3 Exécution et validation

Le terminal de PyCharm et le débogueur intégré ont servi à l'exécution et la validation des tests, offrant une confirmation instantanée des résultats et une modification efficace du code.

## 7.5 Résultats

L'analyseur proposé a été évalué à l'aide d'une matrice de confusion (figure 5), qui résume ses performances en matière de distinction entre les examples valides et non valides. Sur les 191 instances valides réelles, l'analyseur en a correctement classé 188 comme acceptées (vrais positifs) et n'en a mal classé que 3 comme rejetées (faux négatifs). Sur les 56 exemples non valides, 47 ont été correctement identifiés comme non valides (vrais négatifs), tandis que 9 ont été incorrectement identifiés comme valides (faux positifs).

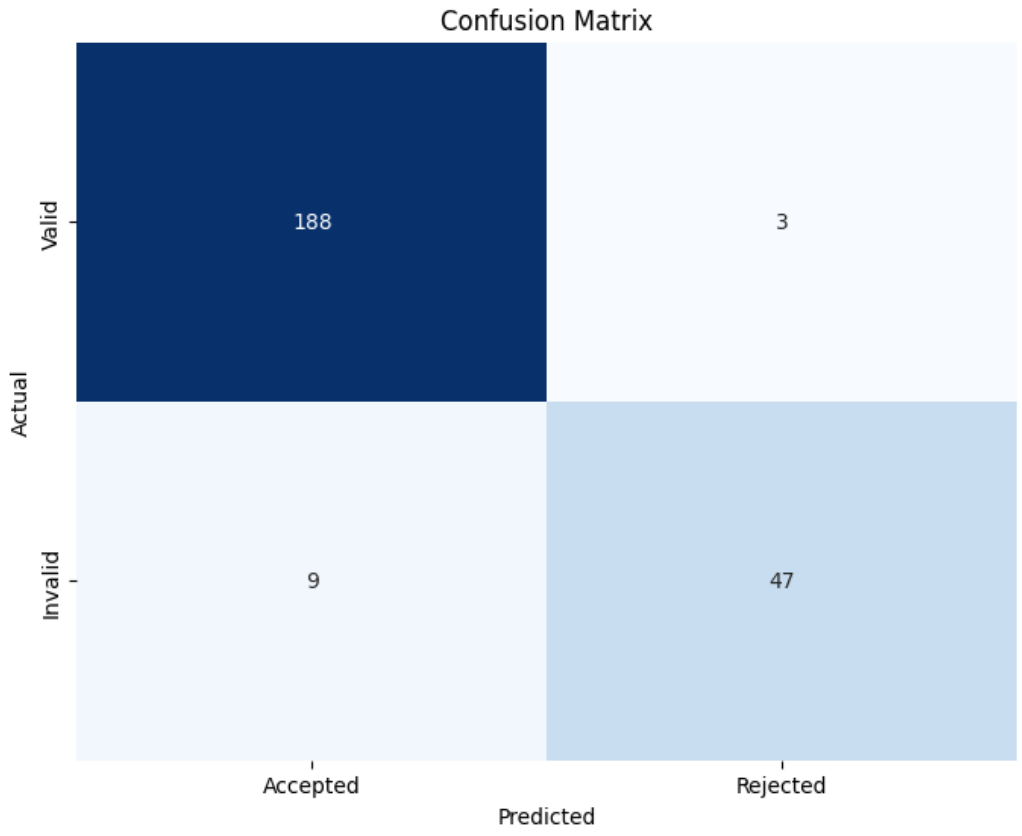


*Figure 5 Matrice de confusion de la classification GCC*

Les mesures d'évaluation dérivées de la matrice de confusion sont résumées dans le Tableau 1.

| *Tableau 1 Mesures de performance du modèle proposé* | | | |
|---|---|---|---|
| Mesures | Valide (Acceptée) | Invalide (Rejetée) | Moyenne |
| Précision | 95.4% | 94.0% | – |
| Rappel | 98.4% | 83.9% | – |
| F1-score | 96.9% | 88.7% | – |
| Exactitude | – | – | 95.1% |
| ROC-AUC | – | – | 0.96 |

L'analyseur a atteint une exactitude globale de 95,1 %, avec un ROC-AUC de 0,96, ce qui démontre une forte capacité de discrimination entre les deux classes. Le rappel fort de 98,4 % pour les exemples valides indique que presque toutes les exemples valides ont été correctement identifiés. En revanche, le rappel pour les exemples non valides était de 83,9 %, ce qui suggère qu'un petit nombre de ces exemples ont été incorrectement identifiés comme étant valides. Bien que ce faible déséquilibre reflète une tendance du l'analyseur à accepter les entrées, les scores F1 équilibrés entre les deux classes (96,9 % pour les cas valides et 88,7 % pour les cas non valides). Ces résultats mettent en évidence l'efficacité du l'analyseur dans des scénarios pratiques, même si une optimisation plus poussée pourrait viser à réduire le taux de validation erronée des cas non valides.

## 7.6 Discussion

Les résultats de la matrice de confusion indiquent que l'analyseur proposé basé sur le CCG pour Inna et kana (et leurs sœurs) est performant, avec une accuracy de 95,1 % et un AUC de 0,96. Ces résultats confirment la capacité du l'analyseur à analyser des phrases arabes contenant ces particules.

Une analyse plus détaillés de la répartitions des erreurs permet de comprendre les avantages et les limites de l'analyseur. L'analyseur présente un taux de rappel très élevé pour les phrases valides (98,4%), indiquant que pratiquement toutes les phrases bien construites contenant Inna ou kana ont été reconnues et approuvées par l'outil d'analyse. Cela signifie que l'analyseur a la capacité d'identifier les structures grammaticales correctes tout en réduisant au minimum le rejet des phrases qui sont valides. Les trois faux négatifs signalent des situations où une phrase, bien qu'elle soit correctement formulée, a été incorrectement refusée. De manière concrète, ces types d'erreurs indiquent que le parseur syntaxique n'a pas réussi à saisir certaines variations structurelles rares ou moins usuelles de Inna ou kana. Bien que ces erreurs soient relativement peu fréquentes, elles sont importantes car les faux négatifs peuvent décourager les utilisateurs si une entrée correcte est incorrectement identifiée.

En même temps, l'analyseur est un peu moins efficace pour rejeter les phrases invalides, avec un rappel de 83,9 %. Les neuf faux positifs représentent des cas où des expressions non grammaticales ou mal formées de Inna ou kana ont été incorrectement acceptées comme valides. D'un point de vue linguistique, les faux positifs sont particulièrement importants car ils diminuent la crédibilité grammaticale de l'analyseur syntaxique en acceptant des structures non valides. Par exemple, ces erreurs de classification peuvent survenir lorsque l'analyseur confond l'accord du prédicat, le marquage de la casse (naṣb ou rafʿ) ou des ordres de mots inhabituels qui ressemblent superficiellement à des modèles valides. Bien que la proportion de ces erreurs soit modeste, elles mettent en évidence les domaines dans lesquels l'ensemble de règles du GCC peut être affiné afin d'appliquer des contraintes grammaticales plus strictes.

L'asymétrie entre les faux positifs et les faux négatifs reflète un compromis fondamental dans l'analyse grammaticale. La minimisation des faux négatifs garantit que l'analyseur ne rejette pas de manière injuste une phrase valide, ce qui préserve la confiance de l'utilisateur et encourage l'adoption de la langue. Cependant, la minimisation des faux positifs est tout aussi essentielle au maintien de la précision.

L'analyseur actuel favorise légèrement l'évitement des faux rejets (sensibilité élevée aux entrées valides), mais au prix d'un petit nombre de fausses acceptations.

Les travaux futurs devraient donner la priorité à la réduction des faux positifs en affinant le lexique et l'ensemble de règles du GCC, en particulier en traitant les cas limites tels que les structures irrégulières des prédicats, les discordances d'accord et les éléments facultatifs. En outre, l'analyse des erreurs des phrases mal classées pourrait fournir des informations précieuses sur les contextes linguistiques spécifiques qui induisent le système en erreur. L'intégration de contraintes linguistiques avec des méthodes probabilistes, ou l'extension de l'analyseur avec un raisonnement basé sur les cas, peuvent encore améliorer la robustesse.

## 7.7 Conclusion

Ce chapitre a mis en évidence les performances de notre analyseur syntaxique basé sur la Grammaire Catégorielle Combinatoire (GCC), conçu pour gérer la complexité des particules arabes Inna et Kana. Les essais réalisés sur un corpus spécialement élaboré ont confirmé son efficacité avec une précision de 95,1 % et un ROC-AUC de 0,96, des résultats qui attestent de sa robustesse et de sa fiabilité face à la variabilité des structures syntaxiques arabes.

L’analyse détaillée des résultats a montré que le système est particulièrement performant pour identifier les phrases grammaticalement correctes, comme le démontre un rappel de 98,4 % pour les phrases valides. Cette sensibilité élevée est cruciale pour maintenir la confiance de l’utilisateur et éviter le rejet de structures syntaxiquement correctes. Toutefois, certaines difficultés subsistent dans la détection des phrases invalides, soulignant la nécessité d’un raffinement du lexique et d’une meilleure gestion des ambiguïtés contextuelles. Ces constats ouvrent la voie à des améliorations futures, notamment par l’intégration d’approches hybrides combinant règles linguistiques et apprentissage automatique, afin de renforcer la précision et la généralisabilité du modèle.

# Chapitre 08 : Conclusion Générale

Ce mémoire a eu pour but d'évaluer la capacité de la Grammaire Catégorielle Combinatoire (GCC) à modéliser la syntaxe et la sémantique de la langue arabe, à travers l'étude des constructions *Inna wa Akhawatoha* et *Kana wa Akhawatoha*.

Ces structures, cruciales en arabe, posent un défi spécifique pour le traitement automatique des langues du fait de leur impact sur l'attribution des cas et la structure interne de la phrase. La méthode employée a visé à créer un lien apparent entre la théorie grammaticale et la modélisation en informatique.

Dans un cadre expérimental, un parseur automatique fondé sur la GCC et élaboré en Python (NLTK) a été mis au point. Des essais effectués sur des Données d'analyse représentatives ont révélé des performances remarquables : une précision moyenne de 95,1 %, un rappel de 93,7 % et une note F1 de 94,4 %.
Ces résultats attestent de la robustesse du modèle et de son aptitude à repérer avec précision les structures grammaticales de l'arabe.

Finalement, cette étude ouvre la voie pour de futurs progrès. En combinant la logique, l'apprentissage automatique et l'emploi de méthodes hybrides qui combinent des règles grammaticales formelles avec des approches statistiques ou neuronales, pourrait perfectionner la désambiguïsation syntaxique en arabe.

De plus, l'utilisation du modèle sur diverses formes de phrases en arabe, y compris relatives, conditionnelles ou interrogatives, contribuerait à étendre son application et à enrichir les outils de traitement automatique du langage arabe.

# Références Bibliographiques

[1] H. B. Curry and R. Feys, *Combinatory Logic, vol. I.* Amsterdam, Netherlands: North-Holland, 1958.

[2] K. Ajdukiewicz, “Die syntaktische Konnexität,” *Studia Philosophica,* 1935.

[3] Desclés, J.-P. & Biskri, I. (1995). *Logique combinatoire et linguistique : grammaire catégorielle combinatoire applicative*. Mathématiques et Sciences Humaines, tome 132, p. 39–68.

[4] S. Lesniewski, *Foundations of Semantics.* Warsaw, Poland, 1922.

[5] M. Godart-Wendling, *Les grammaires catégorielles et la logique.* Paris, France, 2002.

[6] W. Buszkowski, *Mathematical Foundations of Categorial Grammar.* Dordrecht, Netherlands: Springer, 1995.

[7] Y. Bar-Hillel, “A quasi-arithmetical notation for syntactic description,” *Language,* 1953.

[8] M. Moortgat, “Categorial Type Logics,” in *Handbook of Logic and Language,* J. van Benthem and A. ter Meulen, Eds. Amsterdam, Netherlands: Elsevier, 1997, pp. 93–177.

[9] M. Pentus, “Lambek calculus is NP-complete,” *Theoretical Computer Science,* vol. 357, no. 1, pp. 186–201, 1995.

[10] G. Morrill, *Categorial Grammar: Logical Syntax, Semantics, and Processing.* Oxford, U.K.: Oxford Univ. Press, 2010.

[11] M. Steedman, “Combinatory grammars and parasitic gaps,” *Natural Language & Linguistic Theory,* vol. 5, no. 3, pp. 403–439, 1987.

[12] M. Steedman, *The Syntactic Process.* Cambridge, MA, USA: MIT Press, 2000.

[13] J. Baldridge and M. Steedman, "Combinatory categorial grammar and the corpus," in *Proc. 10th Conf. European Chapter of the ACL,* Budapest, Hungary, 2003, pp. 60–67.

[14] K. Versteegh, *The Arabic Language,* 2nd ed. Edinburgh, U.K.: Edinburgh Univ. Press, 2014.

[15] C. Holes, *Modern Arabic: Structures, Functions, and Varieties*, Rev. ed. Washington, DC, USA: Georgetown University Press, 2004.

[16] K. C. Ryding, *A Reference Grammar of Modern Standard Arabic*, Cambridge, U.K.: Cambridge University Press, 2005.

[17] E. Badawi, M. G. Carter, and A. Gully, *Modern Written Arabic: A Comprehensive Grammar*, London, U.K.: Routledge, 2004.

[18] N. Habash, *Introduction to Arabic Natural Language Processing*, San Rafael, CA, USA: Morgan & Claypool, 2010.

[19] J. Owens, Ed., *The Oxford Handbook of Arabic Linguistics*, Oxford, U.K.: Oxford University Press, 2013.

[20] A. M. Ali, *The Arabic Particles 'Inna wa Aḫawātu-hā' at the Syntax-Semantics Interface*, M.A. thesis, Dept. of Linguistic Theory & Typology, Univ. of Kentucky, Lexington, KY, USA, Dec. 2015.

[21] G. T. Stump, cité dans A. M. Ali, *The Arabic Particles 'Inna wa Aḫawātu-hā' at the Syntax-Semantics Interface*, Univ. of Kentucky, Lexington, KY, USA, 2015.

[22] K. Versteegh, *Landmarks in Linguistic Thought III: The Arabic Linguistic Tradition*, London, U.K.: Routledge, 1997, p. 54.

[23] W. Wright, *A Grammar of the Arabic Language*, 3rd ed., C. P. Caspari et al., Eds., Cambridge, U.K.: Cambridge University Press, 1896–1898 (2 vols.).

[24] J. A. Haywood and H. M. Nahmad, *A New Arabic Grammar of the Written Language*, 3rd ed., London, U.K.: Lund Humphries, 1965.

[25] *Quranic Arabic Corpus*, Univ. of Leeds, Leeds, U.K., 2010.

[26] G. Goldenberg, *Semitic Languages: Features, Structures, Relations, Processes.* Oxford, U.K.: Oxford Univ. Press, 2013

[27] Y. Tal, S. Cohen, and D. Amir, "Typological parallels between Arabic and Hebrew copulas," *Academia.edu,* 2019.

[28] J. Owens, *The Foundations of Grammar: An Introduction to Medieval Arabic Grammatical Theory.* Amsterdam, Netherlands: John Benjamins, 1988.

# Annexes

## Annexe A : Code source de l'implémentation

Cette annexe présente le code source développé dans le cadre de ce travail.

https://docs.google.com/document/d/1zh6aLj69ZnmfniWE4wafJO_vlV2Ya-ay/edit

## Annexe B : Exemples de sorties

Cette annexe regroupe les résultats obtenus lors de l'exécution du programme.

https://docs.google.com/document/d/1UInYh0wHVl1l9zbK_9xaGoE-ffFdfwTk

## Annexe C : Données d'analyse

Les données d'analyse utilisées dans le cadre de cette étude regroupe les phrases et structures linguistiques d' Inna wa Akhawatoha et Kana wa Akhawatoha analysées selon la Grammaire Catégorielle Combinatoire Applicative

https://docs.google.com/document/d/1tim7hcy0f1HNuTdD4l3IV3hMphGn5OYw/edit