

# PARSING ARABIC TEXTS USING REAL PATTERNS OF SYNTACTIC TREES

F. Ben Fraj\*, C. Ben Othmane Zribi, and M. Ben Ahmed

RIADI Laboratory, National School of Computer Sciences  
University of Manouba – 2010 - Tunisia

## الخلاصة:

يقدم هذا البحث طريقة جديدة للتحليل الإلكتروني للنصوص العربية تحليلاً نحويًا عميقًا. وقد اخترنا في هذا البحث أن نعتمد على التعلم الإلكتروني. ويتم التعلم من بنك للمعلومات يحتوي على جمل محللة نحويًا. ففي خطوة أولى، نقوم باستخلاص نماذج التحليل النحوية من البنك. وتمثل هذه النماذج عدة تراكيب نحوية مكونة من مجموعة من الطبقات كما أنها غنية بثتى أنواع المعلومات: النحوية، التركيبية، الاشتقاقية والسياقية. نستعمل هذه النماذج لتوجيه التعامل مع الجمل. فالمحلل النحوي الإلكتروني يعمل بتعاقب لأنه يحل الجمل على مراحل مكملة بعضها البعض. وعدد هذه المراحل يساوي عدد الكلمات المكونة للجمل. في كل مرحلة، يتم البحث عن النماذج المحتملة التي يمكنها تمثيل كلمة معينة في السياق الذي تنتمي إليه. ثم يتم تركيب النماذج المختارة مع النماذج التي تم الحصول عليها في المراحل السابقة. في نهاية التحليل يمكننا تكوين مجموعة من التراكيب النحوية الكاملة المحتملة للجمل ككل. عندئذ يتم ترتيب التراكيب بالاعتماد على تواتر تواجدها في بنك الجمل المحللة. مكنتنا النتائج الأولية للاختبارات من الحصول على معدلات مرضية (دقة في التحليل تساوي 84,8 % وأف-سكور مساو لـ77,5%).

---

\*Corresponding Author :

E-mail: [Feriel.BenFraj@riadi.rnu.tn](mailto:Feriel.BenFraj@riadi.rnu.tn)

## ABSTRACT

In order to parse Arabic texts, we have chosen to use a machine learning approach. It learns from an Arabic Treebank. The knowledge enclosed in this Treebank is structured as patterns of syntactic trees. These patterns are representative models of the Arabic syntactic components. They are both layered and rich structurally and contextually. They serve as an informational source for guiding the parsing process. Our parser is progressive since it proceeded by treating a sentence into a number of stages equal to the number of its words. At every step, the parser affects the target word with the most likely patterns that represent it in the context where it is put. Then, it joins the selected patterns with those collected in the previous parsing steps in order to construct the representative syntactic tree(s) of the whole sentence. If more than one tree is proposed, all the analysis trees are sorted according to their appearance frequencies in the Treebank. The preliminary tests have yielded accuracy and f-score equal to 84.8% and 77.5%, respectively.

**Key words:** patterns of syntactic trees, parsing, combination of patterns, Arabic language

## PARSING ARABIC TEXTS USING REAL PATTERNS OF SYNTACTIC TREES

### 1. INTRODUCTION

Parsing Arabic texts is not an easy task to perform because of two fundamental phenomena. The first phenomenon is related to the particularities of that language that make it more ambiguous than other natural languages (especially the Indo-European ones)<sup>1</sup>. These characteristics influence the different levels of Arabic language processing: morphological, syntactic, and semantic. We identify the following characteristics [1] that influence the parsing task 0:

- the diacritization phenomenon and the grammatical ambiguities;
- the agglutination of enclitics with the simple forms;
- the flexible order of the components within the same sentence;
- the multiplicity of the syntactic interpretations for the same sentence;
- the abundance of the embedded structures;
- and the problems related to the segmentation of paragraphs into sentences.

The second phenomenon concerns the significant scarcity of the available digital resources for the Arabic language, especially the grammars and corpora.

Therefore, there have not been many research studies that have focused on parsing the Arabic texts. Until now, it has been difficult to say that there are parsers for the Arabic language that are efficient and/or marketed, while there has been much research on parsing other natural languages.

In this paper, we address the task of parsing Arabic texts using an approach that is based on the machine learning paradigm. Indeed, the parser works in two phases. The first phase consists on learning the real knowledge from an Arabic Treebank. Therefore, we propose a new concept for modeling the training data. The new models are called “patterns of syntactic trees” [2]. As a result of the preliminary stage, we get a base of patterns that includes different models of the Arabic syntactic components. The second phase corresponds to the parsing process. When we parse a new sentence, the previously mentioned patterns include information that helps to indicate how to build the whole syntactic tree(s) for this target processed sentence.

This paper is structured as follows. In Section 2, we describe the syntactic specificities of the Arabic language that influence its parsing process. We provide some motivations for the construction of our Treebank and we describe its creation process in Section 3. In Section 4, we present the concept of the patterns of syntactic trees and discuss the differences between this new concept and the standard elementary trees of the TAG formalism (Tree Adjoining Grammar) [3]. We present the learning stage in Section 5. Section 6 concerns the parsing approach presentation [4]. It is a progressive process and not a greedy process. It takes advantages of the learning paradigm. In the same section, we note some deadlock cases that can block the generic parsing process. After, we propose the means for managing them. In Section 7, we present our main results. We present the related works in Section 8. Therefore, we discuss the Arabic parsers on the one hand, and the works based on the supertagging task on the other hand. In Section 9, we conclude and present our expected outlooks.

### 2. SYNTACTIC SPECIFICITIES OF THE ARABIC LANGUAGE

Syntactically, we have noticed a number of specificities that make the Arabic parsing a very hard task. These characteristics can correspond to the words and the phrases, as well as the sentences.

The first one is the diacritization phenomenon that gives rise to the grammatical ambiguities. Indeed, generally, the Arabic texts are not affected by the diacritics. However, the graphic representations of the words without diacritics are not satisfactory for disambiguating the grammatical interpretations and the semantic meanings. Thus, when the lector reads a sentence, he “virtually” adds these marks in order to specify its grammatical structures and semantic meaning. Consequently, the unvocalized texts are more ambiguous than the vocalized ones. According to Debili’s statistics [5], 74% of the Arabic words accept more than one vocalization.

The second specificity is the grammatical ambiguity which is influenced by the diacritization. In fact, a word can have more than one grammatical interpretation. The Debili’s statistics affirm this fact [5]. The grammatical ambiguity rate<sup>2</sup> reaches 5.6 on average for the vocalized words and 8.7 on average for the unvocalized ones.

The third characteristic is the agglutination of clitics to the simple forms in order to construct more complex ones. An agglutinative form can constitute a whole sentence, as for instance *واستقبلهم* (*Then he welcomed them*).

<sup>1</sup> The references of Debili [5] and Ben Othmane [11] give comparison studies between the Arabic language and English and French.

<sup>2</sup> The grammatical ambiguity rate is the average number of the parts-of-speech that can have a word.

This phenomenon increases the syntactic difficulties since it leads to exceptional structures. Therefore, it requires some specific treatments to search their correct syntactic structure.

The fourth specificity is the abundant use of recursive structures in the Arabic texts. In fact, the embedded structures are common in the Arabic texts as well as in other natural languages. However, it is more frequent for Arabic language since the propositions can play roles in other propositions. Let us consider the following example: الكلاب هي التي أيقظته بنباحها الذي لم ينقطع. (*The dogs have wakened him by their continuous bark*). It is a nominal sentence, while the proposition (خبر) is also a nominal sentence: هي التي أيقظته بنباحها الذي لم ينقطع. (*have wakened him by their continuous bark*). In this same example, even the segmentation into sentences is not possible since there are four propositions that are not independent and do not belong in the same syntactic level. However, they constitute a hierarchical structure of three levels. As a result of this phenomenon, the lengths of the Arabic sentences are not limited.

Furthermore, the order of the sentences' components is flexible. The writer is free to put any component at the beginning of a sentence. This component specifies the order of the constituents that should follow. Nevertheless, the decomposition of a paragraph into sentences also constitutes a problem. Indeed, in the Arabic language, the punctuation marks are not very important for specifying the sentences' delimiters, whereas some specific words and rules are used to decompose the texts into sentences such as the particle لكن (*Therefore*) or the conjunction و (*and*) followed by a verb [6].

[كَبُرَ الشاب] [ولكنه ظلَّ على مجونه] [وأصبح أكثر استهتارا] (*The boy grew up but he is still corrupted and has become more giddy.*)

We should notice that this kind of segmentation leads to many elliptic and anaphoric structures that accentuate the parsing difficulties.

Another syntactic characteristic can be mentioned. It corresponds to the possible multiplicity of the syntactic interpretations for a same target sentence. Indeed, sometimes a sentence can have more than one grammatical interpretation, as for instance the following example: لعب الطفل مع ابن جارهم المؤدب. This sentence can have two meanings according to its two syntactic interpretations:

- *The boy plays with the son of their polite neighbor.*
- *The boy plays with the polite son of their neighbor.*

The first interpretation attaches the adjective المؤدب (*the polite*) to the name جارهم (*their neighbor*). The second one attaches the same adjective to the nominal phrase of annexation ابن جارهم (*the son of their neighbor*). This phenomenon can be expected in other natural languages. However, it is more frequent in the Arabic language than in other languages. We can illustrate this reality by the following example: شاهدت لبوة مع ليث, which can have two different semantic interpretations that lead to two syntactic trees:

- *I perceived a lioness with a lion.*
- *I perceived a lioness with Layth.*

For the first interpretation, we consider that the word ليث means 'lion'. In this case, the expression مع ليث (*with a lion*) is associated with the word لبوة (*a lioness*) to constitute the object of the verb شاهدت (*I perceived*), whereas the same word ليث, in the second interpretation is considered the first name of a boy. Thus, the expression مع ليث (*with Layth*) should be associated with the hidden subject (*I*) to construct a complex subject.

In order to deal with these specificities that constitute problems for Arabic processing, we choose to use all kinds of real information in the parsing task. Thus, we have constructed a new Treebank.

### 3. CONSTRUCTION OF THE TREEBANK

The construction of the Treebank does not represent the research reported in this paper, and we describe it only for the sake of clarity. In spite of the Arabic Treebanks that already exist— Penn Arabic Treebank (PATB) [7], Prague Arabic Dependency Treebank (PADT) [8], the Quranic Arabic Dependency Treebank (QADT) [9], and the Columbia Arabic Treebank (CATiB) [10]—we chose to create a new Treebank. Its construction was undertaken for the following reasons.

Firstly, the present work constitutes a node within a set of other Arabic processing applications. All of them are based on the morphological analyzer of Ben Othmane [11]. This analyzer affects every target word with a rich set of morphological information. Also, it makes use of "finer parts-of-speech". This kind of POS is positional. It includes a set of information that helps to specify the surrounding context of each target item. We mention the following examples of our POS [11]:

- 2 - فعل مضارع مجهول (a verb in the passive voice that is conjugated with a dual pronoun, its case is nominative)
- مضاف إليه معرف بالـ (a complement of noun that is attached to the proclitic "the", its case is genitive)

In contrast, the existent Treebanks manipulate macro-POS. In addition, even if we change the macro-POS into finer ones, we will confront two other phenomena.

Secondly, most of the existent Treebanks contain journalistic texts (PATB, PADT, and CATiB). This writing style has a set of frequent errors in the written Arabic. We can list the following as examples:

- اجتمع زيد بعمر instead of the correct expression اجتمع زيد إلى عمر (Zayd met Amr). We may change the preposition ب (with) to إلى (to).
- تحرى الأمر instead of the correct sentence تحرى الأمر (He investigated the matter). We should omit the preposition عن (about) since the verb تحرى (He investigated) is directly transitive.

Consequently, we suggest that the literary writing style provides correct texts. Thus, it describes the Arabic syntax better than the other writing styles (especially the journalistic ones).

Thirdly, previous research [12] has demonstrated that each portion of information gives additional knowledge to the parsing task. Therefore, we need a Treebank rich in different kinds of information (structural, compositional, and functional) in order to use them in the parsing process. The sentences in a Treebank should be well presented with their correct syntactic analysis. These analyses may present their compositions in more developed elements, as the internal structures of the subject and/or the object in a verbal sentence. Every element should be associated with its correspondent role in the whole parse-tree. On the contrary, the previous Treebanks describe the logical structures<sup>3</sup> of the texts (PATB and CATiB) or the dependencies between the components of these texts (PADT and QADT), whereas, our Treebank consists of a set of sentences that are annotated with

- the morpho-syntactic information of their words,
- their parse-trees,
- the dependencies between their components,
- and the correspondent roles of these components.

Figure 1 presents a portion of our Treebank. This extract is the result of the four following annotation steps. For the sake of legibility, the figure describes the morpho-syntactic information of only the first word, as well as the derivation tree<sup>4</sup> of the whole sentence.

Therefore, we have taken care to build our Arabic Treebank in order to have a Treebank that follows the previous mentioned chain and that contains different types of information (compositions, dependencies, and functions). This Treebank is based on a logical annotation that uses the grammar ArabTAG [13]. It is a grammar that has been inspired from the Tree Adjoining Grammar (TAG) formalism [3,14]. We have implemented it to represent the Arabic syntactic structures as elementary trees. The choice of TAG has not been made randomly. However, a preliminary study of the different grammatical formalisms leads us to choose it [13]. Its conception was based on Kouloughli's book [15].

The Treebank construction follows four steps. The first step is the morphological analysis [11]. In this phase, every word in the texts is affected by sets of morpho-syntactic information:

- all the possible decompositions in the 3-uplets (proclitic/stem/enclitic),
- the possible POS of the clitics and all the possible lemmas of every stem;
  - every lemma can be affected by different sets of fine POS, where every POS is affected by:
    - the gender and number if it is a noun,
    - the transitivity and pronoun if it is a verb.

This first step is followed by a grammatical tagging stage [16]. It associates each word to its most suitable information set according to its surrounding context.

After that, the texts are segmented into sentences. The segmentation [17] is qualified to be fine since we do not use only the punctuation marks but also some specific words or expressions that point out the delimiters of sentences. Two examples of these delimiters were mentioned in the previous section.

<sup>3</sup> The logical annotation of a Treebank consists in affecting the syntactic components by different grammatical phenomena as the elliptical and anaphoric forms. It is used in the English Penn Treebank [18].

<sup>4</sup> The derivation tree is a concept that we take from the TAG formalism. It illustrates the syntactic analysis with a tree where the nodes are the elementary trees (of the TAG formalism) and the links are the operations used for gluing these elementary trees. For more details about this concept and others, consult the reference [14].

```

- <Sentence value = "أدورُ في الدَّارِ الصَّغِيرَةِ الْمُتَوَاضِعَةِ." >
+ <Word num="1" value="أدورُ">
+ <Word num="2" value="في">
+ <Word num="3" value="الدَّارِ">
+ <Word num="4" value="الصَّغِيرَةِ">
- <Word num="5" value="الْمُتَوَاضِعَةِ">
  <Segmentation value="ال/ مُتَوَاضِعَةٍ/">
  <Proclitic value="ال"/>
  <POS-Proclitic>1< POS-Proclitic >
</Proclitic>
  <Stem value="مُتَوَاضِعَةٍ">
  <Lemma value="مُتَوَاضِع">
  <POS-Stem value="203">
  <Gender>F</Gender>
  <Number>S</Number>
  <Pronoun></Pronoun>
  <Transitivity></Transitivity>
  </POS-Stem>
  </Lemma>
  </Stem>
  <Enclitic/>
  </Segmentation>
</Word>
<Punctuation>.</Punctuation>
- <Derivation tree>
  <Tree ref="VS91" />
  <Operation type="instantiation" position="0" />
  <Tree ref="PP1" />
  <Operation type="substitution" position="PP! (n1)" />
  <Tree ref="NP-NA8" />
  <Operation type="substitution" position="NP! (n2)" />
  <Tree ref="NP-Conj2" />
  <Operation type="substitution" position="NP! (n3)" />
  <Tree ref="NP-A" />
  <Operation type="substitution" position="NP! (n4)" />
</Derivation tree>
</Sentence>

```

Figure 1: An extract of an annotated text (The symbol +/- represents the expansion/contraction in XML element.)

In the final step, we tag the sentences by their respective syntactic trees. For this purpose, we have implemented a tagging tool. It operated by classification [19–21] to facilitate the attribution of the most suitable syntactic trees to the different sentences.

The result of all these different steps is a Treebank that consists of real literary texts written by different authors. The texts are taken from basic education reading books of Tunisia. The texts are both vocalized and unvocalized. As a starting point, we began by the syntactic annotation of a part of our texts. Therefore, we constructed a small Treebank of 5000 words that compose 950 sentences. These are of different lengths ranging between one word and 21 words. Every sentence has been assigned by a single syntactic tree.

Certainly, the Treebank contains redundancies. They may overburden the learning stage in a linguistic application. Thus, we choose to model the data into schemes that we call “patterns of syntactic trees”. These patterns are factorizations of the most important information of the Treebank. At this stage, our goal is to collect real syntactic structures from the Treebank. These structures should be well organized, rich, and without redundancies.

## 4. DESCRIPTION OF THE PATTERNS OF SYNTACTIC TREES

### 4.1. What are the Patterns?

We define the patterns as mildly generic models for the representation of the Arabic syntactic components. They have tree structures that contain layered compositions. They include information of different forms: morphological, contextual, and compositional, which are useful for the parsing task to undertake. These patterns are extracted from the Treebank [2].

A pattern is a tree structure that necessarily possesses a root and a signature, and may contain a host of other components. The root specifies the type of the syntactic component that the pattern represents (VS (verbal sentence), NP (nominal phrase), *etc.*). The signature is the basic element of the pattern, as the noun is for a nominal phrase or the verb is for a verbal sentence. It is, generally, placed at the beginning of the syntactic component. It is affected by

contextual and morpho-syntactic information that can help to specify the arrangement of all the following components. The morpho-syntactic criteria constitute the following set:

- the transitivity, voice, and pronoun for a verbal signature;
- the gender, number, and case for a nominal signature;
- and the POS for a particle type signature.

Each signature is also affected by enclitics (proclitic and/or enclitic). All signatures must also admit roles which they play in the pattern structure.

Furthermore, a pattern includes a set of other internal components. The number of these components varies between zero and  $n$  (where  $n > 0$ ). These components are organized according to a lawful syntactic arrangement. Their organization is in concordance with the morpho-syntactic information of the signature.

Every internal component includes a set of contextual criteria within the pattern:

- a category that specifies if the component is a phrase or a proposition;
- a role that the component plays within the pattern;
- a type specifying whether the component is mandatory or optional;
- and a set of possible derivations.

These derivations are represented as a matrix which indicates for every extensible component all lawful patterns that can derive it. The derivations are based on the context in which the component is placed [2]. In reality, a pattern can contain

- The extensible nodes, which are those that can be derived in further processing steps. Each one admits a column of possible derivations in the derivational matrix.
- The non-extensible nodes are those that cannot be derived. We present the two following examples to illustrate this type of nodes:

-أدارت  $\emptyset$  مغزلها- (She turns her spindle) (❶).  
 -ضرب  $\emptyset$  به بعنف- (He beats him violently) (❷).

The two sentences are verbal. In the first sentence, the subject is hidden (فاعل ضمير مستتر) so as to avoid redundancy. We present the absence of the subject with the symbol ( $\emptyset$ ). For the second sentence, the subject is also hidden. In addition, we notice the presence of the enclitic  $\text{ـه}$  (*him*) that plays the role of the direct complement of the verb ضرب (*to beat*). The following figure illustrates the patterns that present these two sentences. We should notice that the patterns must be read from right to left.

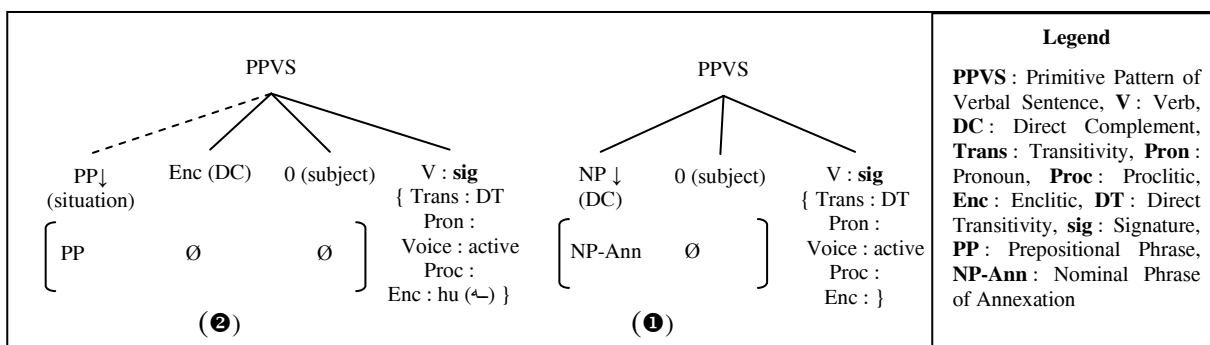


Figure 2: Representative patterns of non-derived nodes

#### 4.2. Different Categories of Patterns

According to the composition of the Treebank, we have divided the patterns into two main classes. The first class corresponds to the primitive patterns that represent high-level components (sentences a priori). The second one is relative to the internal patterns which correspond to the components of intermediate levels (generally the phrases). For each class, we have defined two families of patterns. On the one hand, there are primitive patterns for nominal sentences and others for verbal sentences. On the other hand, there are internal patterns for nominal phrases and others for prepositional ones. The internal patterns can be interior in a primitive pattern or interior in each other. Hence, this emphasizes the possibility of representing the recursive structures. All these kinds of patterns are well

described in [2]. We present in Figure 3<sup>5</sup>, as examples, a primitive pattern of a simple verbal sentence (❶) (Verb, Subject (NP), Direct Object (NP)) and one of its internal patterns. This one is a nominal phrase of annexation (❷) (مركب إضافي). It consists of a noun (N) that should be followed by a complement and an additional nominal phrase (NP) that will play the role of the complement.

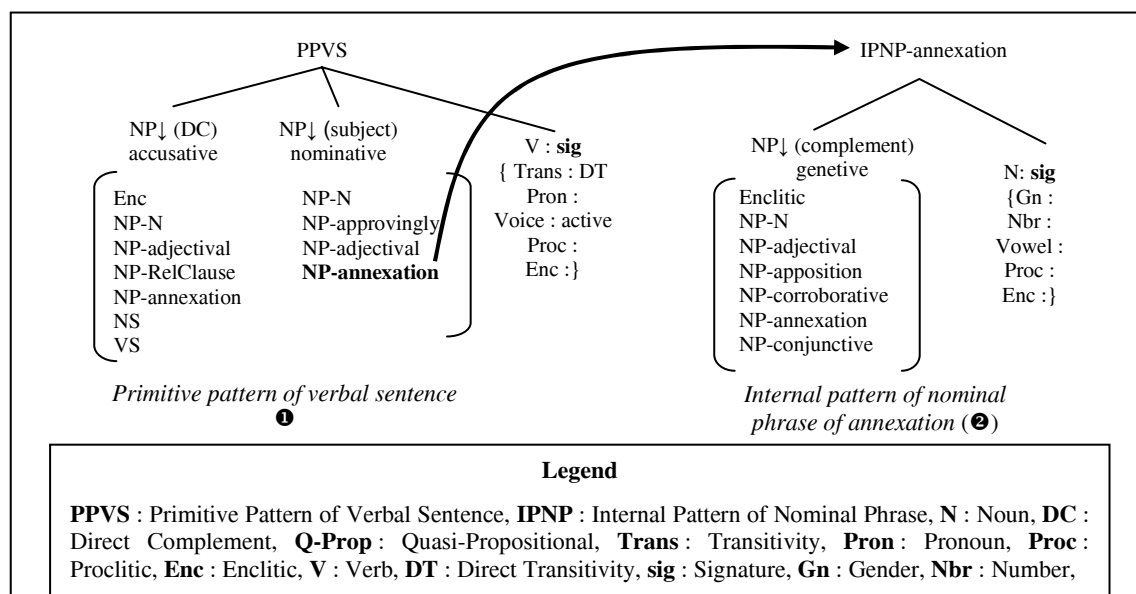


Figure 3: Examples of representative patterns of two Arabic syntactic components

It is noted that to represent an extensible component in the pattern, we lend the substitution symbol ( $\downarrow$ ) to the TAG formalism. Each element of the matrix is a correct derivation for the target component. We note also that the two components in the pattern (❶) do not have exactly the same possible derivations. Also, they have two different ultimate diacritics and two different roles to play in the pattern.

In addition, the derivational matrix allows the restriction of the derivations of the different components in the pattern. Indeed, for example, the NP interior component of the pattern that represents the NP of annexation (❷) should not be extended by a pattern that illustrates the NP of a relative clause (مركب موصولي).

From the two classes of patterns (primitive and internal), we defined a third category. It is the class of the derived patterns. Such a pattern is generated by combining two or more patterns (primitive and/or internal). It may contain extensible components and, therefore, accept other subsequent derivations. The number of layers in its hierarchy is variable, depending on the complexity of its correspondent syntactic structure. The stratification of a derived pattern ends with patterns that we called the unary patterns. They are particular types of (primitive or internal) patterns. They include only the signature. Then, their derivational matrix  $M$  is empty.

#### 4.3. Patterns vs Elementary Trees

Two questions arise:

Why do we use the patterns of syntactic trees instead of the elementary trees of ArabTAG?

And more precisely, what are the differences between our patterns and the TAG formalism?

On the one hand, let us justify why we do not use ArabTAG. Firstly, we should notice that ArabTAG has been used for structuring the Treebank. It was constructed theoretically. In contrast, the patterns are extracted from the described Treebank which is composed of real texts. Thus, the knowledge modelled into the patterns is more representative than the theoretically constructed elementary trees. Secondly, ArabTAG encloses only the standard Arabic structures, *i.e.*, those that consist of the mandatory components [13]. In the patterns, we handle all kinds of components: mandatory as well as optional.

On the other hand, in spite of having the same tree structure, the elementary trees of the TAG formalism and our patterns are different. Firstly, the types of nodes are different from those of the TAG since we distinguish, in each

<sup>5</sup> In this figure, the acronyms correspond to different categories of nominal phrases of the Arabic language which are: NP-annexation (مركب إضافي), NP-adjectival (مركب نعتي), NP-RelClause (مركب موصولي), NP-confirmatory (مركب توكيدي), NP-approvingly (مركب بدلي), NP-Quasi-Propositional (مركب شبه إسنادي), NP-conjunctive (مركب عطفی).

pattern, three main classes of nodes: the signature, the derived nodes, and the non-derived ones. We introduce the non-derived nodes to describe the items that should not be extended in further steps.

Secondly, the derived nodes can be extended by two different operations (composition and union). The operations that we use for the patterns' combination are not exactly the same as those of TAG. They are introduced according to the compositions of the derivational matrices and the necessities of the parsing process. These operations will be well described later in this paper.

Thirdly, among the three types of nodes, only the signature includes a set of information that helps the parsing process, which will be discussed in a following section. The other nodes admit a minimum of information (only the roles and the ultimate diacritics of the components) and the, already described derivational matrices. Thus, the internal attachment of the patterns to each other is not made randomly, thanks to these matrices and the signature information. Indeed, an adjectival nominal phrase (مركب نعني), for example, should not accept an approvingly nominal phrase (مركب بدلي) as an adjective. In contrary, the composition of the elementary trees of the TAG is made according to the unification of the feature structures.

Fourthly, the patterns are made up of both mandatory and optional components [2]. The mandatory components are those that should be present to specify the meaning of the sentence. The optional ones are present to give additional information to the reader as, for instance, the circumstantial elements. In Figure 2, the situation (الحال), which is an optional component, is presented by a dotted line. This characteristic allows us to present rich patterns and will be useful for the parsing process.

## 5. THE LEARNING STAGE: EXTRACTION OF THE PATTERNS OF SYNTACTIC TREES

The extraction process of the patterns from the Treebank is considered preliminary and an important stage, since it constitutes the knowledge base that is useful to manage the parsing process. Figure 4 shows this process.

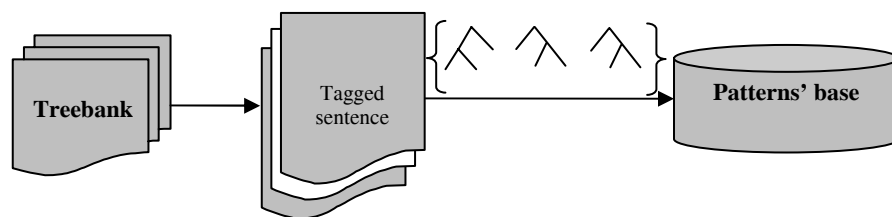


Figure 4: The patterns' extraction process

When we extract these patterns, the parse-trees of the sentences are browsed in depth and width. Thus, we extract the primitive and internal patterns. Each pattern is affected by its signature as well as by all its components that can be extensible and/or non-extensible.

A pattern should exist only once in the generic base of patterns. Thus, each new pattern extracted (primitive or internal) is compared with those that were already placed in the base. The comparison of two patterns is accomplished with the following criteria [2]:

- their respective signatures;
- the numbers of their respective components;
- the categories of their respective components and their roles in the pattern;
- and, their derivational matrices.

Initially, the first three criteria are compared. If the pattern does not exist in the base, it will be added. In contrast, if it has a homologous pattern in the base, we check whether the derivations of the different components of the new pattern exist already among the components of the homologous pattern. If at least one of these derivations does not exist, it will be added to the appropriate place in this homologous pattern.

The following table gives us a small set of examples of sentences that can be represented by the primitive pattern (1) shown in the figure 3.

**Table 1. Examples of Sentences that can be Represented by the Primitive Pattern (❶) of the Figure 3**

Sentence	Translation	Decomposition
أكل الولد تفاحة	<i>The boy ate an apple.</i>	أكل (V) - الولد (subject- NP-N) - تفاحة (Direct Object -NP-N)
أكل الولد تفاحة شهية	<i>The boy ate a delicious apple.</i>	أكل (V) - الولد (subject- NP-N) - تفاحة شهية (Direct Object- NP-adjectival)
أكل الولد الصغير تفاحة شهية	<i>The little boy ate a delicious apple.</i>	أكل (V) - الولد الصغير (subject- NP-adjectival) - تفاحة شهية (Direct Object- NP-adjectival)
أكل ابن الجيران التفاحة كلها	<i>The son of the neighbours ate a whole apple.</i>	أكل (V) - ابن الجيران (subject- NP-annexation) - التفاحة كلها (Direct Object- NP-confirmatory)
أكل الولد ما اشتراه والده	<i>The boy ate what his father bought.</i>	أكل (V) - الولد (subject- NP-N) - ما اشتراه والده (Direct Object- NP-RelativeClause)
بدأ الرجل يشعر بخطورة الموقف.	<i>The man began to feel the gravity of the situation.</i>	بدأ (V) - الرجل (subject- NP-N) - يشعر بخطورة الموقف (Direct Object- VS)

The patterns are encoded in XML format as in the example of Figure 5. This figure illustrates a primitive pattern of a nominal sentence that has a noun (theme) as signature and possesses only one component: a nominal phrase (NP). Consequently, the derivational matrix is composed of only one column that illustrates the possible derivations of this component.

```

- <Meta-pattern title ="PPVS" reference="PP127" frequency="6">
  -<Signature value="N", role="مبتدا">
    <Gender>M/</Gender>
    <Number>S/</Number>
    <Vowel>N/</Vowel>
    <Pronoun/>
    <Transitivity/>
    <Voice/>
    <Proclitic/>
    <Enclitic>39/</Enclitic>
    <GVStem>194/</GVStem>
  </Signature>
  -<Component value="NP" role="خير" type="extensible" mandatory="+ ">
    <Derivation> NP-QPropositional </Derivation>
    <Derivation> NP-Adjectival </Derivation>
    <Derivation> NP-Approvingly </Derivation>
  </Component>
</Meta-pattern>

```

Figure 5: Example of a primitive pattern for a nominal sentence

## 6. DESCRIPTION OF THE PARSING APPROACH

First, we should note that the parsing procedure does not parse raw texts. Nevertheless, the texts must undergo some pre-treatments that are, successively, a morphological analysis [11] followed by a grammatical tagging [16] and segmentation into sentences [17]. Consequently, we suggest that the processed sentences are syntactically correct.

Our approach is both gradual and not greedy. It is gradual since it ensures the sequential processing of all the words in the target sentence. Words are treated from the right to the left in the order of their appearance in the sentence. Thus, the number of steps during the parsing process is equal to the number of words that compose this sentence. At each step  $i$ , the parser chooses the most plausible patterns (primitive or internal) to represent the target word in its surrounding context. After that, the chosen patterns are joined to the derived patterns built until the step  $(i-1)$  of the same process. At the end of the processing, we get its possible representative syntactic trees.

The parsing approach is also not greedy because we suggest that an Arabic sentence can easily admit more than one syntactic interpretation giving the structural richness of that language. Therefore, at a parsing stage, all the selected patterns that represent the target word are considered. Each pattern is used as a basis for subsequent stages of the parsing process. Thus, the procedure conducts the treatment of all the selected patterns jointly. We keep the ambiguity throughout the parsing process until it is lifted in some stages or at the end. Therefore, we might have more than one candidate for the syntactic interpretation of a given sentence.

### 6.1. General Treatment

The treatment of the first unit is an essential step when we parse an Arabic sentence. Indeed, for the Arabic language, the basic element which we emphasize to specify the type of the whole syntactic component is generally set at its beginning [23].

Thus, when processing the first word  $w_1$ , the primitive patterns in the base of the patterns will be filtered in order to choose the most likely ones to represent this target word. We proceed with a filtering phase. This phase is rule-based. It consists of a correspondence between the information of the different patterns, and those of the word  $w_1$ . This first phase provides an initial set of patterns  $SP_1$ . Each element of  $SP_1$ , if it is not unary, can be the basis for possible future extensions.

For the treatment of an intermediate word  $w_i$ , we proceed, as for the first word, by filtering all the patterns in the patterns' base. A set  $SP_i$  of representative patterns of  $w_i$  is generated. Each pattern  $p'$  from  $SP_i$  will undergo a combination with each pattern that belongs to the set  $(SD_{i-1})$  of the derived patterns that were built till the step  $(i-1)$  of the parsing process. Thus, if the number of the elements of  $SD_{i-1}$  is equal to  $n$  and the number of those of  $SP_i$  is equal to  $m$ , the maximum number of possible combinations is estimated to be  $n \times m$ . This process ends with a new set  $SD_i$  of derived patterns that are built until the current step.

Consequently, at every step, we construct a forest of derived trees that has the chance to be extended in further stages of the parsing process. The density of the patterns that compose the forests increases at each step. In a given forest, the number of patterns can be

- increased by lawful combinations,
- and decreased by eliminating the patterns that can not participate in other ulterior combinations.

This same principle is repeated until the last word of the processed sentence is reached.

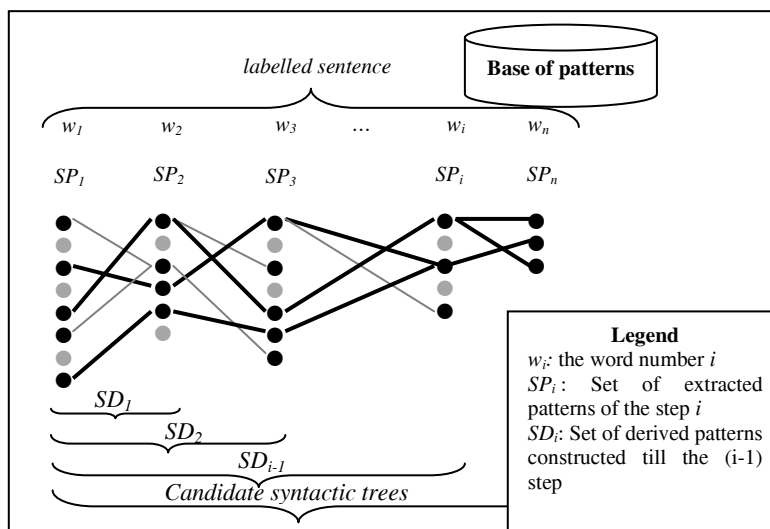


Figure 6: General treatment of the parser

The figure below illustrates the general treatment process that the parser follows. During this process, there are alternations between the filtering phase and the combination one. The symbol (●) represents a pattern. The filtered patterns at any step may participate in further combinations, or they can be neglected in the following parsing stages. This second case is shown in the figure by the gray patterns (●). The combinations at any step can strengthen or weaken previous constructed blocs. The paths in thick lines represent the lawful established combinations. Consequently, they illustrate the most suitable syntactic tree(s) for interpreting the processed sentence.

Ultimately, we get different analysis trees as results. If the obtained tree is completely derived, *i.e.*, it does not contain extensible nodes, it will be considered as an analysis result. However, if the obtained tree admits further derived nodes, two cases are distinguished:

- if, at least, one of the derived nodes presents a mandatory component, then the tree will be excluded from the list of candidates;
- if all the derived nodes present optional components, the tree will be considered as a parse-tree for the sentence.

## 6.2. The Patterns' Association

At each parsing step, the filtering process leads to a new set of patterns that can represent the current processed word. These patterns can be joined to the derived patterns which were constructed until the previous step. The algorithm of Figure 7 describes the composition of the patterns.

```

S = {w1, ..., wi, ..., wn}
{SD1} ← Filtring_Process (PPs, w1)
{SP1} ← null
Historic_1 ← {PDs}
i ← 2
while i ≤ n do
    {SPi} ← Filtring_Process (Patterns, wi)
    Historici ← {SDi-1}
    Buffer ← null
    for each pattern pc from {SDi-1}
        for each pattern pi from {SPi}
            P ← pc ★ pi /* The operator ★ is used for composing the patterns */
            if P ≠ ∅ then
                Add P to Buffer
            end if
        end for
    end for
    {SDi} ← Buffer
    i ← i + 1
end while
Eliminate_redundancy ({SDn}) /*The result is the set {SDn}*/
/* The silences of the composing phase lead us to use the derived patterns stocked in the Historici */

```

Figure 7: Algorithm of associating the patterns

The construction of the derived patterns is based on a compositional algebra [22]. This algebra manipulates the patterns as operands with two main operators: composition and union. These operators are presented in the algorithm by the same symbol ★.

- *Composition* if the extension of the host pattern  $pc$  with the pattern  $p_i$ , exactly at the current component to extend  $c$ , is possible. The composition information is included in the patterns and their derivational matrices.
- *Union* if the pattern  $p_i$  does not exist in the derivational matrix of the host pattern  $pc$  and even the verification of its composition does not allow the composition of  $c$  with  $p_i$ . In this case, two types of unions are defined:

- Union by coordination if the two patterns in the union belong to the same syntactic level. In this case, the two patterns  $pc$  and  $p_i$  will be the internal components of a newly formed pattern  $P$ . For example, this operation is absolutely useful for attaching different adjectives that qualifies the same item, as in the expression المرأة حسنة الخلق المضحية التي تشقى من أجل أطفالها (*the polite and poor woman that curses for her children*). The three qualifications should be joined in the same coordination structure.
- Union by subordination if the two patterns belong to two different syntactic levels. In this case,  $pc$  will be enriched by a new component that will receive the pattern  $p_i$ . The new component is called  $nc$  in Figure 8, which illustrates abstractions of the two union types. For instance, the union by subordination allows attaching a new object to the primitive pattern of a verbal sentence if its verb can admit more than one object, as in the sentence رأيت الصدقَ خيرَ وسيلة للنجاح في الحياة (*I saw the truth the best way for success in life*). In this sentence, the first two words constitute a correct sentence (verb/hidden subject (I)/object) syntactically, since the verb is directly transitive. However, the portion خيرَ وسيلة للنجاح في الحياة (*the best way for success in life*) is also an object that can be added to the sentence without causing syntactic problems. In this case, the verb is considered double transitive (متعدي بمفعولين). The union operation can deal with this case.

The composition and union are restricted by a set of syntactic constraints. Therefore, we defined, for our algebra, a third operator: the restriction [22]. This operation plays an important role when gluing the patterns. It is helped by the information organized in the patterns (in the signatures and the derivational matrices) in order to allow the construction of lawful syntactic structures.

Furthermore, in the previously presented algorithm, at every stage, the buffer contains a new set of newly constructed derived patterns. At the end of the actual word processing, the buffer gives its content to the actual *SD*. In addition, at each step, the filtered patterns are stocked in a set that we call Historic. This set can be useful if the composition of the patterns fails to give us, at least, one derived pattern. This case is considered as a deadlock case that we explore in the following paragraph.

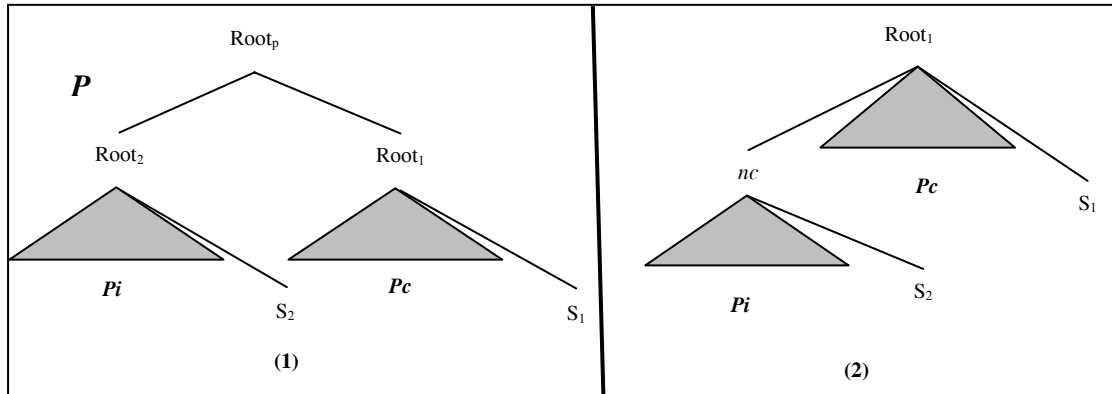


Figure 8: Abstractions of the two union types: (1) coordination and (2) subordination

### 6.3. Identification of the Deadlock Cases

Each pattern  $p$  that belongs to  $SD_{i-1}$  is skimmed in depth and width in order to make the combinations during the parsing process. The idea consists of searching the rightmost extensible component to derive it with the different elements of the set  $SP_i$  of patterns. The component to expand can be so far in the hierarchy of the pattern  $p$ . If we consider  $c$  this component, the composition of  $c$  with the pattern  $p'$  ( $p' \in SP_i$ ) in the derived pattern  $p$  ( $p \in SD_i$ ) may lead to two possible situations. The first situation is when  $c$  is already ready to receive the pattern  $p'$ , *i.e.*, it contains the pattern  $p'$  among its possible derivations. In this case,  $p'$  will be considered as a composition of  $c$  in the derived pattern  $p$ . In contrast, the second situation occurs when the pattern  $p'$  does not belong to the set of derivations of the component  $c$  in the pattern  $p$ . This situation generates two different cases:

- the element  $p'$  can extend the component  $c$  but the derivational matrix of the pattern  $p$  is incomplete;
- the element  $p'$  can not be an extension of the component  $c$ .

For the two cases, we proceed with an attempt to compose the two patterns based on a set of syntactic constraints. These constraints are correspondences between the component  $c$  and the root of the pattern  $p'$  according to their contextual and morpho-syntactic respective criteria.

If the composition is lawful (1<sup>st</sup> case), we attach  $p'$  (the pattern) at the appropriate place in  $p$ . Then, we update the original base of patterns by adding  $p'$  amongst the possible derivations of  $c$  in  $p$ . Thus, our parser has the ability to achieve an incremental learning, *i.e.*, that the base of patterns is improving gradually as we are currently parsing. In contrast, if the composition is not possible (2<sup>nd</sup> case), we try to extend  $c$  with each of the other patterns of  $SP_{i-1}$ .

In some cases, no composition is allowed between the two sets of patterns. Therefore, we planned to construct fragments of patterns that represent different portions of the target sentence. These fragments are the results of the well done combinations in different parsing steps. Subsequently, we make use of the union operations to attach the fragments based on a set of syntactic constraints that allow or prohibit their links. The lawful unions allow the building of newly derived patterns not present in the base. The newly built patterns can broaden the coverage of the whole base.

In some other cases, the filtering procedure fails to allocate representative patterns to some units that belong to the sentence. In addition, the union operation may fail to build new patterns. For these two situations, the labelled sentence will be presented to an expert linguist who will affect it with its correct syntactic tree. Thus, it supplies the Treebank and improves the coverage of the base of patterns.

### 6.4. How to Sort the Resulting Analysis Trees

Ultimately, the parsing process can lead to an important number of analysis trees which can even contain some erroneous ones. The errors vary. They can concern the labels' nodes in the resulting parse-tree and/or the dependencies in the structure of this tree. The results may also contain the correct analysis trees. We choose to use a statistic model, in order to sort the plausible trees. If the resulting syntactic tree  $P$  (derived pattern) is the junction of  $i$  patterns (primitive and/or internal), the probability  $Prb_p$  of  $P$  is estimated using the following formula:

$$\Pr b_p = \prod_{i=1}^n pb_{p_i}$$

The probability  $\Pr b_p$  of the syntactic tree  $P$  is the product of the elementary probabilities of the different patterns  $p_i$ . These elementary probabilities are extracted from the Treebank. The  $pb_{p_i}$  is the co-occurrence frequency of the pattern  $p_i$  in the Treebank with its surrounding patterns in  $P$ , *i.e.*, the patterns  $p_{i-1}$  and  $p_{i+1}$  if they exist. The most plausible parse-tree should be set at the top of the sorted list of candidates.

## 7. EVALUATION OF THE PARSER

Our evaluation process checks the ability of our parser to associate the correct patterns to the different words in the sentences as well as its capacity to affect all the suitable parse-trees of the Arabic sentences. We have performed five cross-validation tests to undergo this evaluation. Thus, we get a portion of our Treebank. This portion consists of 1134 words that compose 250 sentences. The sentences' lengths vary between 1 and 12 words.

Then, we proceeded by doing five different tests. In each test, we consider the Treebank as follows: 80% for the learning stage and 20% for the test. Thus, in each test, 200 sentences were used to extract the patterns, whereas the remaining 50 sentences are used for the test.

Three different measures are used. The following formulas respectively express the filtering phase and the parsing one:

- Precision =  $\frac{\text{number of the correct filterings (resp. analysis)}}{\text{number of the generated filterings (resp. analysis)}}$
- Recall =  $\frac{\text{number of the correct filterings (resp. analysis)}}{\text{number of all the items to be filtered (resp. sentences to be parsed)}}$
- F - score =  $\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

We mention, in the following paragraphs, the results that we got. However, we start by presenting the bases of patterns that we have extracted.

### 7.1. Learning Stage: Extraction of the Base of Patterns

Before parsing, we have extracted five sets of patterns, as shown in Table 2. From every set of 200 sentences used for a learning stage, we have extracted a base of patterns. These patterns are divided into two families: the primitive patterns and the internal ones. The extraction reduces the sizes of training sets according to two dimensions: the phrases and the words.

**Table 2. The Numbers of (Primitive and Internal) Patterns Extracted by Test**

Tests	Primitive patterns	Internal patterns
1	103	179
2	99	164
3	99	176
4	93	160
5	80	142
<b>Average</b>	<b>≈ 95</b>	<b>≈ 164</b>

We have extracted 95 primitive patterns, on average, from the sentences. Thus, we decrease the size of the Treebank of more than half of the sentences (52%). This reduction does not generate a loss of information because the primitive patterns possess all information contained in the sentences. Apart from the linguistic units, all the other knowledge is reserved.

Furthermore, the average of the numbers of words in the training sets is 906. In addition, the internal patterns represent the interior composition of the sentences. On average, the words are reduced to more than 80%. As for the primitive patterns, the most generic information is reserved in the internal patterns.

These reductions allow us to reduce the searching space in the Treebank when parsing.

### 7.2. Evaluation of the Filtering Process

When evaluating, we are interested in the filtering phase in the first step. The results are shown in Table 3.

**Table 3. Evaluation Results of Cross-Validation of the Filtering Phase**

Tests	Precision (%)	Recall (%)	F-score (%)
1	89.85	86.71	88.25
2	80.59	79.41	80.00
3	93.82	91.56	92.68
4	89.32	86.79	88.03
5	86.02	84.21	85.10
<b>Average</b>	<b>87.92</b>	<b>85.73</b>	<b>86.81</b>

The filtering procedure gives us results that we can judge satisfactory (86.81%: average of the F-score). In the five tests, our parser had filtered the base of patterns for 227 words per test, on average. It succeeded in attributing patterns to 85.73% of the words with a precision estimated at 87.92%.

However, the failures (estimated at 14.26%) are mainly due to deficiencies in the bases of patterns. Indeed, the patterns which were constructed in the preliminary learning phase (about 259 patterns on average) may not be able to cover all syntactic representations that are enclosed in the test sentences. Furthermore, the selective power of the filtering procedure may also cause cases of silence and/or confusion. It is silent when it does not give any pattern to represent the target word. Also, it creates confusion when it confuses the target word with other words and affects it by erroneous patterns. As examples of the confusion, we note that some modifiers of nominal sentences such as *كان* (*He was*), which is a temporalization modifier, admits the same morpho-syntactic markers and contextual attributes as those of verbs, *i.e.*, the transitivity and the pronoun. Thus, the modified nominal sentences can be confused with the verbal ones.

### 7.3. Evaluation of the Parsing Phase

Table 4 describes the results of the cross validation for the patterns' combination. The compositions are achieved simultaneously with the different filtering steps. In this evaluation, we include the handling of the deadlock cases, mentioned before.

**Table 4. Evaluation Results of the Parsing Phase**

Tests	Precision (%)	Recall (%)	F-score (%)
1	95.45	84.00	89.35
2	82.22	72.54	77.07
3	87.75	82.69	85.69
4	76.92	60.00	67.41
5	81.57	58.49	68.12
<b>Average</b>	<b>84.78</b>	<b>71.54</b>	<b>77.52</b>

From this table, we note that the parser provides an F-score average that we consider satisfactory (77.52%), given the smallness of the Treebank that we used for these tests. Indeed, for 71.54% of the processed sentences, the parser is able to award, at least, a correct syntactic interpretation, with an average accuracy estimated at 84.78%. However, the parser fails to parse 28.46% of the sentences (on average). It can generate erroneous parse-trees for 12.24% of the failures cases. For the other 16.22% cases, the parser generates fragments of trees but not a whole tree for the processed sentence and fails to unite them. This high rate of silence is the result of the following:

- The failures during the filtering procedure which are caused by the inability of the bases of patterns to cover all the syntactic structures that the test files contain. Consequently, the low numbers of patterns that the learning stage generates in tests 4 and 5 justifies the low performances of these two tests.
- The failures when combining the patterns of the filtering stage, which are caused by the lack of information within their derivational matrices.
- The lengths of sentences: the longer the sentence (in terms of words), the more there are risks of errors in its parsing process. The test files for experiments 4 and 5 contain sentences of average lengths equal to 4.5 and 4.62, respectively. These two averages are higher than those of the first three experiments that are, respectively, 3.14, 3.92, and 3.60 (words/sentence). This justifies the performance decrease of the last two experiments compared with the others.

Nevertheless, our parser has properly dealt with the sentences that have complex structures, *i.e.*, the sentences which are composed of more than one proposition at a time. This is thanks to the richness of the patterns (morphological, contextual, and compositional). This knowledge is very useful for the parsing task. For instance, the parser was able to treat the two following sentences.

The first sentence is *وذلك يربت على أنفه* (*And this one pats his nose*). This is a nominal sentence where its first part *ذلك* (*this one*) is a demonstrative and its second part is a verbal proposition. The parsing result of this sentence is illustrated by the following figure:

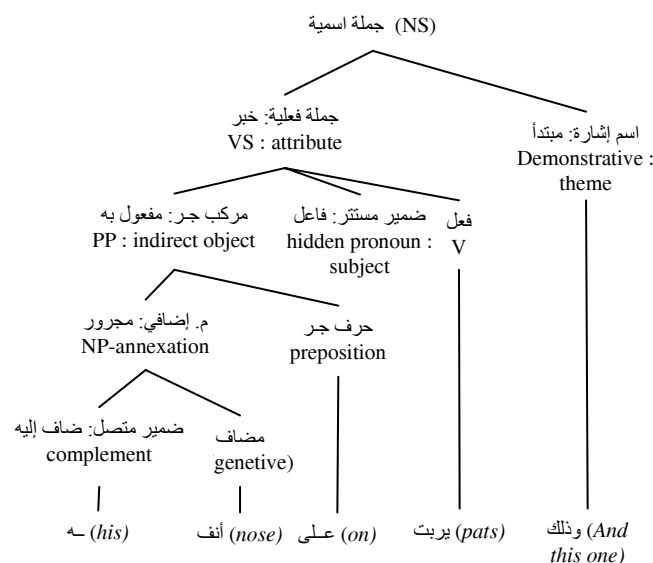


Figure 9: The resulting graph that represents the sentence *وذلك يربت على أنفه* (*and this one pats its nose*)

In this figure, every node has its correspondent label and role. Also, we should note that this graph is coded in an XML format. Each leaf node possesses all its morpho-syntactic information. The parser can give more than one syntactic tree. The resulting trees are sorted according to their respective probabilities.

The second sentence is *وبدأت أشعر بهول المهمة التي وضعتها لنفسي* (*And I began to feel the gravity of the mission I set myself*). This sentence is verbal but it contains three propositions. The third one consists of a relative subordinate.

## 8. RELATED WORKS

Over the last ten years, there has been a great increase in the construction of the Arabic parsers. The fundamental goal is to confront this language with the technological challenges.

Consequently, a set of parsers for the Arabic texts have been designed. Most of the researchers (Ouersighni [24], Othman *et al.* [25], Aloulou [26], Bahou *et al.* [27], and Bataineh [28]) have chosen rule-based approaches. Thus, they collect sets of grammatical rules in order to construct the syntactic tree that represents the analyzed sentence. However, the specificities of the Arabic language make complex the construction of a complete grammar with good coverage. In this same context, for a rule-based parser, if the rules are very specific and restrictive, the research area that they describe will be reduced and we may be faced with a coverage problem. In contrast, if the rules are not restrictive, the search space will be very large, but the disambiguation between syntactic structures becomes a challenge in itself [29]. Moreover, some of these works consider the parsing as a sub-task in a complete process of linguistic analysis [24,26].

A second direction of more recent research has investigated the use of a machine learning paradigm for the same task. We refer to the works of Mathkour *et al.* [30], Tounsi *et al.* [31], Green and Manning [32], and our present approach. The researchers in this perspective use real knowledge organized as Treebank. The majority of the previous researchers adapted some other research that was developed for English [33] in order to use them for parsing Arabic texts. Adapting English parsers for the Arabic language requires a lot of investigation because of the large differences between the Arabic and the other languages. It can be studied in further research.

However, we are not aware of any other work that chooses to model the learning knowledge as patterns to facilitate their manipulation in an Arabic parsing approach. The concept of the patterns has been used in three different parsing research studies [34–36]. In all these studies, the patterns are context free rules that are enriched by additional morphological and contextual information. Our work follows this direction. It gives a new way of modeling for the patterns which, largely, help the parsing process.

Furthermore, the present work seems to be similar to the works on supertagging for parsing of Bangalore and Joshi [37] or Nasr and Rambow [38]. The idea is the same. Indeed, these works manipulate elementary trees according to the grammatical formalism TAG [3]. For instance, Nasr and Rambow [38] convert the elementary trees

to a recursive transition network to make their exploitation easy when parsing. Also, they use a set of stochastic models to choose the most suitable trees for the target word in its surrounding context. However, our parser makes use of the patterns that have the same structures (trees), but are generated from real annotated texts and are rich in information. Their structure and richness help their conjunction to construct more complex structures. In addition, the compositional operations are not the same as those of TAG since we make use of the union operation that is different from the adjunction. We suggest that our approach is neither a simple imitation of the supertagging research nor an adaptation of the TAG formalism in a parsing approach. However, it takes care of the syntactic specificities of the Arabic language. Also, the pattern's formalization is the result of these characteristics.

## 9. CONCLUSION AND OUTLOOKS

In this work, we have built a parser for Arabic texts, which takes advantage of the machine learning paradigm. We have defined a new manner for modeling the knowledge that composes an Arabic Treebank. The new models are called the patterns of syntactic trees. The concept of patterns has allowed us to reduce the size of the Treebank in terms of sentences and words, which also permit the reduction of the search space during the parsing process.

The structural and informational richness of these patterns helped the management of the parsing process, which is progressive and not greedy. Our preliminary results are considered satisfactory given the smallness of the Treebank that we used for the evaluation. Also, we have noticed that the parser is able to parse complex sentences.

Actually, we are interested in proposing a pattern recognition model in order to improve the accuracy of the filtering procedure and to reduce the number of the filtered patterns at each step. In this model, the classes are the patterns and the items to classify are the words with their correspondent contexts.

In addition, we expect to test our parsing procedure on other test files. For this purpose, we are ameliorating our Treebank by adding other texts from different fields and styles (classical, poetic, quranic, *etc.*). Also, we plan to make other tests by using other Treebanks.

Furthermore, we intend to improve the modelling system by affecting the transitions, within and between patterns, by probabilities that are estimated from the Treebank. We expect also to refine the syntactic constraints of the patterns' combination to well manage their different junctions.

Finally, we think that the stochastic model used to sort the resulted analysis trees is not very reliable. Thus, we intend to define a more sophisticated statistical module for sorting such candidates.

## REFERENCES

- [1] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, "Grammaire TAG pour l'Analyse Syntaxique de Textes en Arabe comme un Problème de Classification", *9<sup>th</sup> IBIMA Conference, Special Session: Arabic Information Processing, Marrakech- Maroc*, 2008.
- [2] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, "Modeling the Data of an Arabic Treebank as Patterns of Syntactic Trees", *ACIT'2010, Benghazi, Libya*, 2010.
- [3] A. Joshi and Y. Schabes, *Handbook of Formal Languages, Volume 3*. 1997, pp. 69–124.
- [4] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, "Patterns of Syntactic Trees for Parsing Arabic Texts", *NLPKE 2010, Beijing, China*, 2010.
- [5] F. Debili, H. Achour, and E. Souissi, "La Langue Arabe et l'Ordinateur : De l'Etiquetage Grammatical à la Voyellation Automatique", *Correspondances* 71(2002), Lyon, p. 1.
- [6] G. Mourad, L. Hadrich Belguith, and L. Baccour, "Segmentation de Textes Arabes en Phrases Basée sur les Signes de Ponctuation et les Mots Connecteurs", *Troisièmes Journées Scientifiques des Jeunes Chercheurs en Génie Electrique et Informatique, Tunisie*, 2003, pp. 25–27.
- [7] M. Maamouri, A. Bies, T. Buckwalter, and W. Mekki, "The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus", in *NEMLAR International Conference on Arabic Language Resources and Tools, Cairo, Egypt*, 2004, pp. 102–109.
- [8] J. Hajič, O. Smrž, P. Zemánek, J. Šnaidauf, and E. Beška, "Prague Arabic Dependency Treebank: Development in Data and Tools", in *NEMLAR International Conference on Arabic Language Resources and Tools, Cairo, Egypt*, 2004, pp. 110–117.
- [9] K. Dukes and T. Buckwalter, "A Dependency Treebank of the Quran Using Traditional Arabic Grammar", in *The 7<sup>th</sup> International Conference of Informatics and Systems (INFOS 2010)*, 2010, pp. 1–18.
- [10] N. Habash and M. Roth Ryan, "CATiB: The Columbia Arabic Treebank", *ACL-IJCNLP 2009 Conference Short Papers, 2009 ACL and AFNLP, Suntec, Singapore*, 2009, pp. 221–224.

- [11] C. Ben Othmane Zribi, “De la Synthèse Lexicographique à la Détection et à la Correction des Graphies Fautives Arabes”, *Ph D thesis, Paris XI University, Orsay*, 1998.
- [12] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, “Quels Attributs Discriminants pour une Analyse Syntaxique par Classification de Textes en Langue Arabe ? ”, in *Traitement Automatique des Langues Naturelles (TALN 2009)*, Senlis, France, 2009.
- [13] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, “ArabTAG: A Tree Adjoining Grammar for Arabic Syntactic Structures”, in *Proceedings of International Arabic Conference of Information Technology (ACIT 2008)*, Hammamet, Tunisie, 2008.
- [14] A. K. Joshi, “Introduction to Tree Adjoining Grammar”, in *Mathematics of Language*. ed. A. Manaster-Ramer. Amsterdam: John Benjamins, 1987.
- [15] J. Kouloughli, *La grammaire Arabe pour Tous*. Edition Presses Pocket, 1992.
- [16] C. Ben Othmane Zribi, A. Torjmen, and M. Ben Ahmed, “An Efficient Multi-Agent System Combining POS-Taggers for Arabic Texts”, *CICLing 2006*, 2006, pp. 121–131.
- [17] F. Ben Fraj, “Un Système Multi-Agent pour la Détection et la Correction des Erreurs Cachées de la Langue Arabe” , *Master Memory, National School of Computer Sciences, Manouba University*, 2004.
- [18] M. Marcus, B. Santorini, and M. Marcinkiewicz, “Building a Large Annotated Corpus of English: The Penn Treebank”, *Journal of Computational Linguistics*, **19(2)**(1993), pp. 313–330.
- [19] F. Ben Fraj, C. Ben Othmane Zribi, and M. Ben Ahmed, “A Semi-Automatic TAG Syntactic Tagging Tool for Constructing an Arabic Treebank”, in *Computational Linguistics-Applications’09, Mragowo, Poland*, 2009, pp. 207–212.
- [20] C. Ben Othmane Zribi, F. Ben Fraj, and M. Ben Ahmed, “Combining Classifiers for Supertagging Arabic Texts”, in *NLPKE 2010, Beijing, China*, 2010.
- [21] C. Ben Othmane Zribi, F. Ben Fraj, and M. Ben Ahmed, “Decision Tree Classifier for Supertagging Arabic Texts”, in *ACIT’2010, Banghazi, Libya*, 2010.
- [22] F. Ben Fraj, “Un Analyseur Syntaxique pour les Textes en Langue Arabe à Base d’un Apprentissage à Partir de Patrons d’Arbres Syntaxiques”, *Ph D thesis, National School of Computer Sciences, Manouba University*, 2010.
- [23] R. Blachère and M. Gaudetroy-Demombynes, *Grammaire de la Langue Arabe*. Maisonneuve et Larose, 1975.
- [24] R. Ouersighni, “A Major Offshoot of the DIINAR-MBC Project: AraParse, a Morpho-Syntactic Analyzer of Unvowelled Arabic Texts”, in *ACL 39th Annual Meeting. Workshop on Arabic Language Processing: Status and Prospect, Toulouse*, 2001, pp. 66–72.
- [25] E. Othman, K. Shaalan, and A. Rafea, “A Chart Parser for Analyzing Modern Standard Arabic Sentence”, *The MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches, New Orleans, Louisiana, U.S.A.*, 2003.
- [26] C. Aloulou, “Une Approche Multi-Agent pour l’Analyse de l’Arabe : Modélisation de la Syntaxe”, *Ph D thesis, National School of Computer Sciences, Manouba University, Tunisia*, 2005.
- [27] Y. Bahou, L. Hadrich Belguith, C. Aloulou, and A. Ben Hamadou, “Adaptation et Implémentation des Grammaires HPSG pour l’Analyse de Textes Arabes non Voyellés”, *15<sup>ème</sup> Congrès Francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle RFIA’2006, Tours/France*, 2006.
- [28] M. Bataineh Bilal and A. Bataineh Emad, “An Efficient Recursive Transition Network Parser for Arabic Language”, *The World Congress on Engineering 2009, Vol. II, London, U.K.*, 2009.
- [29] K. Sagae and A. Lavie, “A Classifier-Based Parser With Linear Run-Time Complexity”, *The 9<sup>th</sup> International Workshop on Parsing Technologies, Vancouver, Canada*, 2005, pp. 316–323.
- [30] I. Mathkour Hassan, A. Tourir Ameur, and A. Al-Sanea Waleed, “Parsing Arabic Texts Using Rhetorical Structure Theory”, *Journal of Computer Science*, **4(9)**(2008), pp. 713–720.
- [31] L. Tounsi, M. Attia, and J. Van Genabith, *Parsing Arabic Using Treebank-Based LFG Resources*. Lexical Functional Grammar 2009, Cambridge, UK, 2009.

- [32] S. Green and C. D. Manning, “Better Arabic Parsing: Baselines, Evaluations, and Analysis”, *COLING 2010, Beijing*, 2010.
- [33] M. Bikel Daniel, “Design of a Multi-Lingual, Parallel-Processing Statistical Parsing Engine”, *The Human Language Technology Workshop*, 2002.
- [34] L. Wei-Chuan, P. Tzusheng, L. Bing-huang, and C. Chuei-Feng, “Parsing Long English Sentences With Pattern Rules”, *The 25th Conference of COLING*, 1990, pp. 410–412.
- [35] L. Hyeon-Yeong, H. Yi-Gyu, and L. Yong-Seok, “Parsing of Korean Based on CFG Using Sentence Pattern Information”, *IJCSNS*, **7(7)**(2007), pp. 57–62.
- [36] S. Rauzy and P. Blache, “*Un Point sur les Outils du LPL pour l’Analyse Syntaxique du Français*”, *ATALA2009*, 2009.
- [37] S. Bangalore and A. Joshi, “Supertagging: An Approach to Almost Parsing”, *Computational Linguistics*, **25(2)**, 1999, pp. 237–266.
- [38] A. Nasr and O. Rambow, “Supertagging and Full Parsing”, *The Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+7)*, 2004.
- [39] A. Abeille, “Parsing French With Tree Adjoining Grammar: Some Linguistic Accounts”, *The 12<sup>th</sup> Conference on Computational Linguistics, Budapest, Hungary*, 1988, pp. 7–12.