ADX — Agent for Morphologic Analysis of Lexical Entries in a Dictionary

Bogdan Pătruț
Department of Mathematics and Computer Science,
Faculty of Sciences, "Vasile Alecsandri" University of Bacău
Calea Mărăşeşti, 157, 600115, Bacău, Romania
bogdan@edusoft.ro

Abstract

This paper refers to the morphological analysis of words, as an important process in the domain of natural language processing. We will present the classical solution, based on the use of inflection paradigms and of an extended database, containing all roots of the every words, and then there are emphasized some of the disadvantages of this method. Then we will present an original method, which dynamically generates the roots of words, using phonological alternances in the context of inflection rules. There are also presented some optimizations of the morphologic analysis algorithm.

Keywords: natural language processing, morphologic analysis, inflection type, phonological alternance, pattern matching, searching tree

1. Introduction

Text processing is a very important domain of the computer science, even if we see it only commercially. So, one of the best sold soft packs is the Microsoft Office, that contains the professional text editor Microsoft Word. We say "professional" having in mind the fact that Word has not only the usual text and graphic processing, but, also, powerful tools for analysing the text from a grammatical point of view and we refer here to the possibility of spelling checker and hyphenation; thus, Word can discover if the words from a phrase come from a certain language, previously selected (in the boundaries of an available dictionary). But the next step is to find if those words form any syntactically and semantically correct phrase, in the selected language. So, for example, the phrase: "*Copilul mâncat bătaie a*" ("*The child eaten beat has*") contains only Romanian words, but, of course, the phrase topic is completely wrong, so a syntactic analysis could show us that the correct phrase would be "*Copilul a mâncat bătaie*" ("*The child has eaten beat*") (N.B. The correct translation of the "*Copilul a mâncat bătaie*" is "*The child has been beaten.*").

Yet, this sentence can be easily rejected by a semantic analyser that wouldn't know that the "*a mânca bătaie*" ("*to eat beat*") syntagm should not be understood literally, being a phrasal verb.

In the course of a text processing, there are two main phases, strictly linked between them. The first phase (*the establishing of a standard text*) it is the one that has as an input the text in the form of either a sound wave, or of a written text, usually handwritten. The output data of the first phase are constituted of a recognizable text for any editor, preferably in the form of a text file.

- If in the first phase the analysis of a sound signal (the speech itself) is made, then this fact will be the subject of the electronic study, of the sound processing and hardware engineers, but of the linguists also, who would trace the phonetic aspects of the text.
- If recognition of the handwriting is made, then the problem is a software one, being linked with the recognition of the forms' domain.

Once we have accomplished this first phase, the text must be *grammatically analysed* and this analysis consists in: lexical analysis, morphologic analysis, syntactic analysis, semantic analysis, stylistic analysis and, according to some authors, pragmatic analysis, too.

The first three analyses constitute an important phase, having as a result the obtaining of a phrase that we cannot say is incorrect in that language, e. g. "*Copilul a mâncat bătaie*". It is hard to suppose, of course, that a child can have a "*bataie*" ("*beating*") for his meal, but the problem is solvable from a semantic point of view, because "*a mânca bătaie*" is a phrasal verb having the meaning of "*a fi bătut*" ("*to be beaten*"). (The correct translation of the "*Copilul a mâncat bătaie*" is "*The child has been beaten.*").

As a result, we can present the general scheme of a linguistic analysis of a phrase (figure 1):

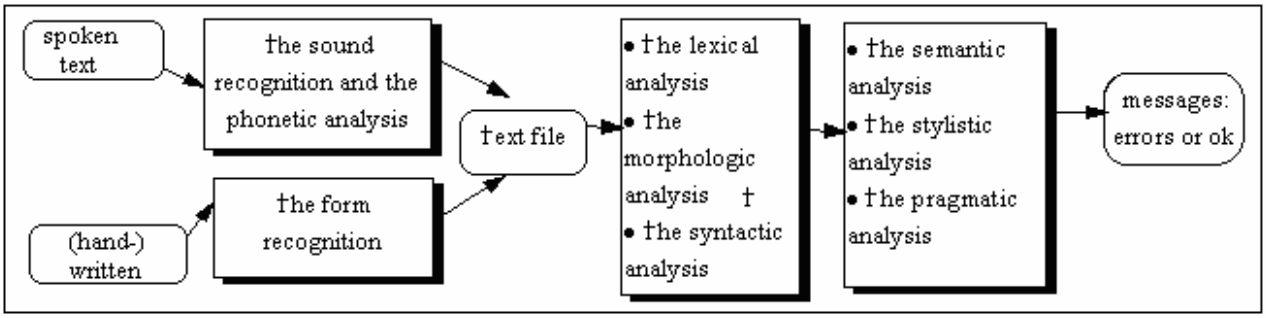

Figure 1. Linguistic analysis

This analysis may be continued as a reverse text generating process (written or spoken) so the final result is a translation of a text from one language into another: for example, the Romanian text "*Plouă cu găleata*" would become "*It rains cats and dogs*" in English, owing to the semantic analyses.

But, before applying semantic analysis, the syntactic analysis has a very important role, and an efficient morphologic analysis can simplify the work of the syntactic analyser.

2. Objectives

We want to develop a software agent which will automatically detect the current word edited in Microsoft Word and for this word the agent will do some tasks:

- it morphologically analyses the actual word typed in Microsoft Word, and obtains its standard form;
- it obtains the definition in Romanian for the standard form (of the current word from MS Word);
- it finds synonyms and antonyms for the given word;
- it generates semantic rules for the synonymic and antonymic relations for the ASR agent.

Dex Online is a project initiated and coordinated by Catalin Francu [3]. He intended to realise an online database for all the words in the Romanian language, using the main explanatory dictionaries, dictionaries of synonyms, neologisms, published by the Romanian Academy and other scientific forums.

The database was completed by volunteers, similarly to the Wikipedia system. They actually transcribed the information from different important dictionaries, but many words have been electronically entered by two companies (Siveco and Litera International Publishing House).

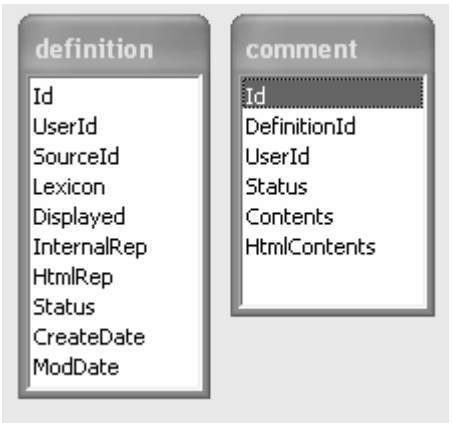

Figure 2. The structures of the two tables from the Dex Online database

Being a dictionary, and not a WordNet, Dex Online has a simpler organization, where the entries are words, and not meanings. Because there is no finished project like WordNet, we considered useful to implement an agent that uses this lexical resource to the maximum.

We can download the complete database of the dictionary at www.dexonline.ro [3], in MySQL format, which can be easily converted in MDB (Microsoft Access) format.

There are two tables in the Dex Online database (figure 2).

The most important table is *definition*, which offers for the word into the Lexicon field its definition in a specific form (InternalRep) or in HTML format (HTMLRep). The source of the word is stored in *SourceId* field. For example, if *SourceId* is 1, then the source is "Explanatory Dictionary of the Romanian Language" („Dicționarul Explicativ al Limbii Române", edited by the Romanian Academy, if *SourceId* is 6, then the source is a dictionary of synonyms, and if *SourceId* is 7, then we can obtain the antonyms of a given word.

In figure 3 we present a capture screen of using the ADX[1] agent in editing the text in Microsoft Word. It displays the definition of the current word, but it also generates the synonyms, used in the application of some web searching rules, used by another intelligent agent, called ASR. The ASR agent will automatically compose some search strings to use for a web search engine, like Google, Yahoo, or other. For example, if the user will search the word *zăpadă* (snow) on Yahoo search engine, then one can also generate a search for the word *nea* (synonym of *zăpadă*) by using the following ASR rule:

 # Yahoo search
IF
http://search.yahoo.com/search?p=^X^&fr=yfp-t-309&toggle=1&cop=mss&ei=UTF-8
THEN
http://search.yahoo.com/search?p=^Clasa(X)^&fr=yfp-t-309&toggle=1&cop=mss&ei=UTF-8

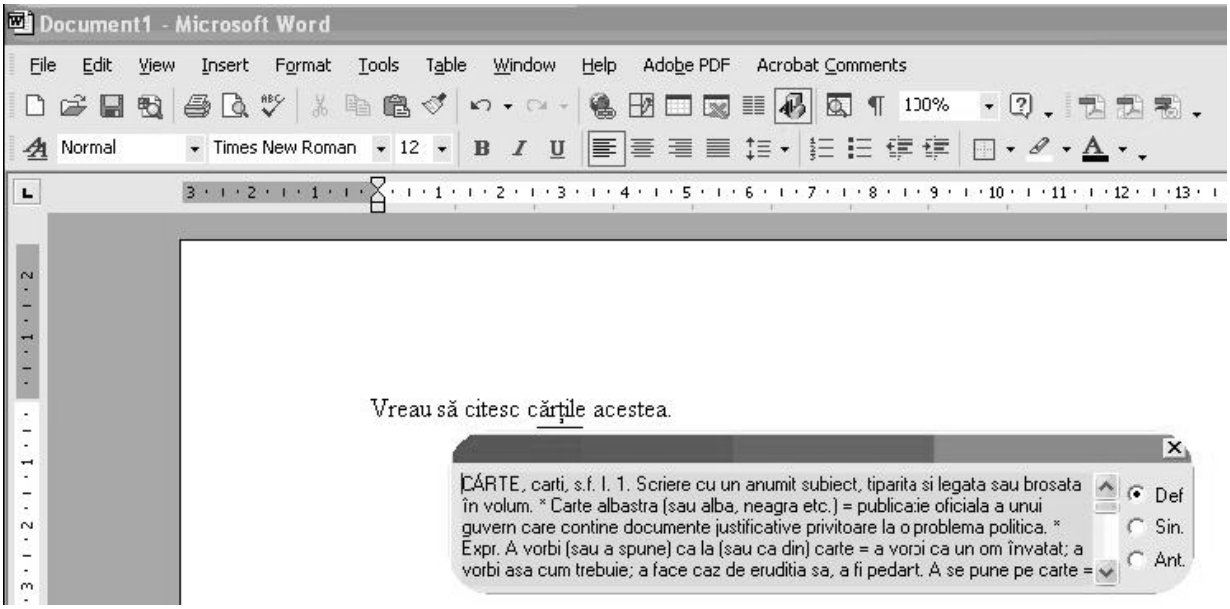

Figure 3. Using the ADX agent for obtaining definitions, synonyms and antonyms, for a given word, during MS Word editing

[1] ADX comes from "Agent of DEX Online" (DEX = usual abbreviation for Explanatory Romanian Dictionary)

The ASR[2] and ADX agents are parts of our multi-agent system that we develop for monitoring and suggesting purposes regarding editing in Microsoft Office. This does not make the subject of our discussion, so, for now, it is important to emphasize the ADX agent..

The important fact about the ADX agent is that it can work not only with standard words (the dictionary entry), but with different inflected forms of the given word (for example, a declined noun, a conjugated verb etc.).

3. Using phonological alternances and inflexional types for morphological analysis

As in any dictionary, words appear in the standard form (nominative/accusative, singular/plural, no article for nouns, infinitive and indicative present, first person, singular for verbs, respectively), the ADX agent must morphologically analyze a given word, to obtain the dictionary form. Thus, if in figure 2, we had the phrase "Vreau să citesc cărțile acelea" ("I want to read those books"), we would obtain the same definition as for the word *cărțile*, with the note that the standard form is *carte*, a noun, in the nominative/accusative case, plural and articulated form.

We created and implemented an algorithm for morphological analysis of the words in the Romanian language (nouns, pronouns, verbs, adjectives) based on phonological alternances and inflection types, that we will further present. The main advantage of the method is that it dynamically generates the roots of the words, using the phonological alternances in the rules of declination and conjugation, the roots having not been previously stored.

Of course the solution is not always unique, as we can see from comparing the two phrases: "E o **luptă** cu morile de vânt" ("It is a fight against the windmills.") and "El **luptă** cu morile de vânt" ("He fights against the windmills."), where the word "luptă" ("fight") is either a noun or a verb.

Although ADX is not organized in parts of speech, like WordNet, it can offer different meanings of a word and its definitions.

4. The Morphologic Analysis

It studies the forms of words, in their different inflected forms, starting with the meanings that the morphemes have.

The word is the basic unit of a language—seen from its two essential types of existence: as *system* and as *speech*—through its central position of its functioning, that is in the activity of linguistic communication. The word tells us about the world.

The morpheme is the smallest linguistic unit with a bi-plan structure; it has a *signifier* (phonetically realised) and a *signified* (a meaning). The morpheme exists and realises its significant function inside the word, it is part of and only through an opposition inside the grammatical category that determines its meaning.

Having an inflected word (declined, in the case of the nouns, and of the adjectives a.o., or conjugated, as in the verbs' case), a linguistic engineer would look for obtaining the word in its standard form (the same with the one from the dictionary), together with a series of explanations referring to some of the grammatical categories: tense, genre, case etc.

But this is a syntactic analysis problem, that presents words in their context, and not singularly, as in the morphologic analysis case.

In order to understand the expected result from a morphologic analysis, let us take an example. For such a word as "cepelor" we have to get the following information: the standard word = "ceapă", part of speech: noun; case = genitive or dative; articled = yes; number = plural; genre = feminine.

[2] ASR comes from "Agent for Semantic Rules" (we develeped a complex system of software agents, where each agent has its own role)

5. The Classical Morphologic Analysis

Generally, the linguistic engineers ([1], [2]) who studied morphologic analysis (for Romanian) have solved such problems by starting from an extended database, to which they applied a relatively simple algorithm. So, each entry in the database consisted of a word and its inflected paradigm (another word that is equally declined or conjugated) and all the roots that a word may have when being inflected. So, for example, for the word "ceapă" you may use as an inflected paradigm the word "casă", and the algorithm would analyse "ceapă" by using the endings that derive from the word "casă" in its declination process. Of course, you may observe that "ceapă" and "casă" are declined in the same way, but in the articled genitive/dative plural case, whereas "casă" becomes simply "caselor", the word "ceapă" would use another root: "cep-" and not "ceap-".

Despite the advantage owed to the algorithm's simplicity, this method has some disadvantages:

- The creation of the database is a very hard work, because all the word roots in their different forms must be taken into consideration.
- The created database may be enormous and the searching process, necessary for the morphologic analysis, can be slowed down and only because of the huge data volume that must be explored.
- In some cases, there is also the risk that the selected inflected paradigm can not be used for some exceptional words.

6. The Idea of the Phonological Alternances

To remove the disadvantages of such a classical method, one can start from the idea of finding the standard form of any word *cuv*, one must apply to it a pattern having the form of $*s_1*s_2*...*s_n*$, in which the asterisk has the significance of any string existence (even in the case of the avoidance of those strings), and s_1, s_2, ... s_n are groups of standard letters, found in many words.

For example, for the word "cepelor"[3], we have n=2, with s_1 = "e" and s_2 = "elor".

If we have a pattern-matching upon a word *cuv*, then one may obtain the standard form of *cuv*, by applying a rule such as: $*s_1*s_2*...*s_n \rightarrow *d_1*d_2*...*d_n*$, in wich d_1, d_2, ..., d_n are also letter groups. For example, for "cepelor" we have d_1 = "ea", d_2 = "ă" and by applying the rule $*e*elor* \rightarrow *ea*ă*$ we can obtain the standard word "ceapă".

Generally, the number of such changes is small, varying from 1-2 for adjectives to 1-3 for nouns and verbs. That is why we can give up the idea of the pattern, by using what linguistics call *phonological alternances*. Any word that has been declined or conjugated got another ending and replaced the standard one, and the word root suffered changes because of the phonological alternance.

So, if $*a_1*t_1$ is an inflected word, its standard form would be $*a_2*t_2$, that is the application of a rule having the form $*a_1*t_1 \rightarrow *a_2*t_2$. In fact, this is the reverse of the inflectional process, in which the t_2 ending is replaced by the t_1 ending, and the root of the word suffers a phonological alternance $a_2 \rightarrow a_1$.

It should be noted that these phonological alternances are not of any importance, because they form themselves only at the level of some letter groups (especially vowels, diphthongs). Also, there are some phonetic regular consonant phonological alternances, that overlap the endings, or are the result of their application, like: *-c + i → -ci; -g + i → -gi; -t + i →ţi; -d + i → -zi; -s + i → -şi; -sc + i → -şti; -sc + e → -şte; -str + i → -ştri*.

Another advantage of the morphologic analysis usage based on the phonological alternance consists in the fact that these rules are applied randomly in the inflection of words, but under certain circumstances. For example, "ceapă"[4] becomes "cepelor" when using the articled declination at genitive/dative cases and plural number. This information can be used in the synthetic analysis, too.

---

[3] *of the onions / to the onions* (genitive/dative, plural for *onion*)
[4] *onion* (singular, nominative/acusative)

But, the application of the declination in the backward meaning cannot always have a correct result, because there are also too many conditions that have to be fulfilled:

- the obtained standard word, has to be in the dictionary (its lexical analysis has to be done);
- the obtained standard word must be of the same *inflected type* [4] as the inflected word from which we started.

The inflected type is a grammatical category that gives us exact information concerning all the forms that a word can take when being inflected. So, the inflected type contains not only the inflection paradigm, but also the given information for applying either this or that phonological alternance.

Coming back at the "*cepelor*" word case, as a result of the backward flexion rules applications, only from the feminine nouns declinations we can get three solutions: "capă", "cepelă" and "ceapă". Each solution corresponds to the backward sense application of a single declination rule that corresponds to its inflected type.

Of course, only the last solution is valid, and this fact can be seen only by consulting the dictionary:

- "capă"[5] can be found in the dictionary, but it belongs to another inflected type;
- "cepelă" does not appear in the dictionary;
- "ceapă" appears in the dictionary and belongs to the inflected type requested by the application of the rules *ea* → *e* and *a* → *elor*.

However, there are other cases in which we can obtain two or more valid solutions, like: "luptă", "duce".

The dictionary articles have the structure of two fields: the standard word, accompanied by its inflected type. It can also be used in many ways: the tree (B type), stored on a magnetic support, eventually combined with an internal hash table, for an easier finding of the articles.

7. Reducing the Searching Space

It is necessary to check all the endings and all their corresponding alternances, viewing the rules of application. But we may notice that for each word it is necessary to verify the rules that contain only some endings, (for example, "cepelor" can have the ending/suffix: zero (-), "lor" or, finally, "elor", that contains the other two and that gives us the solution).

The tracking down of the possible endings of a word can be done in the following way:

- the word *cuv* is mirrored, so that we may get *cuvi*;
- a searching binary tree is formed; it contains in its nodes all the reverses of the endings that some cells' lists are associated with, each cell memorizing two values (*l*,*c*); these represent the finding rows or columns of the given ending in the flexion table (memorized in an array, having reduced dimensions and being easily accessible).
- *cuvi* is looked for in the endings' tree as follows:
  - you overlap *cuvi* with the information from the current node (which is an ending);
  - if it matches, that ending is memorised, but you must also pass to the subtree, from the right side which could contain other possible endings;
  - in case of failure, the searching ends.

At the very beginning, the tree will be built like a searching tree, additionally having a certain characteristic:

If N is the information from a node and L and R are the bits of information from the left subtree root, respectively the right one, then it is necessary that:

- $L<N<R$, so as the tree could be a searching one;
- in addition: if $Reverse(N)=t_1$ is the ending for a *cuv* word, then any other ending t_2, with $t_2=*t_1$ it would be only in the root of the right subtree, i. e. in R ($Reverse(R)=t_2$) (*Reverse(s)* is a function that

[5] *pelerine* (singular, nominative/acusative)

returns the reversal of an *s* string). (E.g.: *Reverse("role") = "*elor*"* and *Reverse("*rol*") = "*lor*"*. It is worth noticing that "role" < "rol" (having in view the alphabetic order) and we could have *N="*rol*", R="*role*"*, because if "elor" is an ending for a word, "lor" is also the ending for the same word (i.e. t_1 = *Reverse(N)* = "*lor*", t_2 = *Reverse(R)* = "elor" and t_2=*t_1, * = "*e*") (e.g. *cuv* = "cepelor", "ferestrelor", "caselor")).

The zero ending (-) is a possible ending for any word, and that is why it will be the root of the whole searching tree, that corresponds to any table.

An example of such searching trees for nouns of all genders (masculine, feminine, neuter) together with the entire analysis made by the ADX agent for the word "cepelor" is presented in figure 2. The pairs of numbers in brackets form lists of lines and columns from the inflections' charts where endings from the top of the list are found. We have noted the zero ending with – (that is always the root of the tree) and the void list with * (Null/Nil).

```
-> -: 7 (18,3)(19,3)(20,3)(32,2)(33,2)(34,2)(34,3)
  ├──*
  └── i:20 (6,3)(7,3)(8,3)(9,3)(10,3)(11,3)(12,3)(13,3)(14,3)(15,3)(16,3)(17,3)(18,8)
  (19,8)(20,8)(21,3)(22,3)(34,8)(35,2)(35,3)
  ├─── a:43 (1,2)(1,6)(2,2)(2,6)(3,2)(3,6)(4,2)(4,6)(5,2)(5,6)(6,2)(6,6)(7,2)(7,6)(8,2)(8,6)
  (9,2)(9,6)(10,2)(10,6)(11,2)(11,6)(12,2)(12,6)(13,2)(13,6)(14,6)(18,6)
  (19,6)(20,6)(26,2)(26,6)(27,2)(27,6)(28,2)(28,6)(29,2)(29,6) (30,2)(30,6)(31,2)
  (31,6)(34,6)
  │ ├──*
  │ └┬─ ae: 8 (15,6)(16,6)(17,6)(23,6)(24,6)(25,6)(35,6)(36,6)
  │ ├──*
  │ └┬─ ael: 2 (21,6)(22,6)
  │ ├──*
  │ └── e:17 (1,3)(2,3)(3,3)(4,3)(5,3)(14,2)(15,2)(16,2)(17,2)(18,2)(19,2)(20,2)(23,2)(24,2)(25,2)
(36,2)(36,3)
  │ ├─── au: 2 (32,6)(33,6)
  │ └┬─ el: 8 (18,7)(19,7)(20,7)(21,2)(22,2)(32,3)(33,3)(34,7)
  │ ├──*
  │ └┬─ ele: 6 (1,7)(2,7)(3,7)(4,7)(5,7)(36,7)
  │ ├──*
  │ └┬─ elel: 2 (32,7)(33,7)
  │ ├──*
  │ └── eli:15 (6,7)(7,7)(8,7)(9,7)(10,7)(11,7)(12,7)(13,7)(14,7)(15,7)(16,7)(17,7)(21,7)
  (22,7)(35,7)
  │ ├──*
  │ └── eliru: 9 (23,7)(24,7)(25,7)(26,7)(27,7)(28,7)(29,7)(30,7)(31,7)
  └── ii:21 (6,8)(7,8)(8,8)(9,8)(10,8)(11,8)(12,8)(13,8)(15,8)(16,8)(17,8)(21,8)
  (22,8)(23,8)(24,8)(25,8)(26,8)(27,8)(28,8)(29,8)(35,8)
  ├──┬ ie: 9 (1,8)(2,8)(3,8)(4,8)(5,8)(14,8)(30,8)(31,8)(36,8)
  │  ├──*
  │  └── iel: 2 (32,8)(33,8)
  ├┬─ iru: 9 (23,3)(24,3)(25,3)(26,3)(27,3)(28,3)(29,3)(30,3)(31,3)
  ├──*
  ├┬─ rol: 4 (18,9)(19,9)(20,9)(34,9)
  ├──*
  ├┬─ role: 6 (1,9)(2,9)(3,9)(4,9)(5,9)(36,9)
  ├──*
  ├┬─ rolel: 2 (32,9)(33,9)
  ├──*
  └── roli:15 (6,9)(7,9)(8,9)(9,9)(10,9)(11,9)(12,9)(13,9)(14,9)(15,9)(16,9)(17,9)(21,9)(22,9)(35,9)
  ├──*
  └── roliru: 9 (23,9)(24,9)(25,9)(26,9)(27,9)(28,9)(29,9)(30,9)(31,9)
-> rol: 4 (18,9)(19,9)(20,9)(34,9)
  ├──*
  ├┬─ role: 6 (1,9)(2,9)(3,9)(4,9)(5,9)(36,9)
  ├──*
  ├┬─ rolel: 2 (32,9)(33,9)
  ├──*
  ├┬─ roli:15 (6,9)(7,9)(8,9)(9,9)(10,9)(11,9)(12,9)(13,9)(14,9)(15,9)(16,9)(17,9)(21,9)(22,9)(35,9)
  ├──*
  └── roliru: 9 (23,9)(24,9)(25,9)(26,9)(27,9)(28,9)(29,9)(30,9)(31,9)
-> role: 6 (1,9)(2,9)(3,9)(4,9)(5,9)(36,9)
  ├──*
  ├┬─ rolel: 2 (32,9)(33,9)
  ├──*
  ├┬─ roli:15 (6,9)(7,9)(8,9)(9,9)(10,9)(11,9)(12,9)(13,9)(14,9)(15,9)(16,9)(17,9)(21,9)(22,9)(35,9)
  ├──*
  └── roliru: 9 (23,9)(24,9)(25,9)(26,9)(27,9)(28,9)(29,9)(30,9)(31,9)
```

```
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: cepelore–F4
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: #–F4a
Found root: cepelor Corresp. ending: – Components: cepel.r Looking in dict: cepeloare–F4b
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: cepelor–F9
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: cepelor–F9a
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: cepelor–F10
Found root: cepelor Corresp. ending: – Components: cepelor. Looking in dict: cepelor–F10
Found root: cepe Corresp. ending: lor Components: cepe. Looking in dict: cepee–F4
Found root: cepe Corresp. ending: lor Components: cepe. Looking in dict: cepee–F4a
Found root: cepe Corresp. ending: lor Components: cepe. Looking in dict: cepee–F4b
Found root: cepe Corresp. ending: lor Components: cepe. Looking in dict: cepe–F10
Found root: cep Corresp. ending: elor Components: cep. Looking in dict: cepa–F1
Found root: cep Corresp. ending: elor Components: c.p Looking in dict: ceapa–F1a
Found root: cep Corresp. ending: elor Components: c.p Looking in dict: capa–F1b
Found root: cep Corresp. ending: elor Components: c.p Looking in dict: capa–F1c
Found root: cep Corresp. ending: elor Components: c.p Looking in dict: cepa–F1d
Found root: cep Corresp. ending: elor Components: cep. Looking in dict: cepe–F10b
```

Figure 2. The searching tree for Romanian feminine nouns

8. Inflection Tables

After we have found the possible endings of a word, we begin to partially explore the tables, dynamically generating the roots of the words, by using the rules with the order numbers given by the cell lists associated with the possible endings.

Generally, these tables are realised by the linguistic specialists according to the parts of speech. The columns are realised having in mind the different inflection situations and the rows are realised having in mind the inflection types. The cells in the table may contain letter groups either from the phonological alternances range, or from the endings range.

Presented below (table 1) is a fragment from the table associated with the feminine nouns declinations. The last column has the possible endings of "cepelor" written in bold letters. The looking up in the dictionary will find only the shaded solution.

Table 1. The declination of feminine nouns (after [4])

inflected type	declination without an article				declination with an article			
	endings		alternances		n/a endings		g/d endings	
	sg. ending	pl. ending	sg.	pl.	na sg	na pl	gd sg	gd pl
1	ă	e	-	-	a	ele	ei	**elor**
1a	ă	e	ea	e	a	ele	ei	**elor**
1b	ă	e	a	e	a	ele	ei	**elor**
1c	ă	e	ă	e	a	ele	ei	**elor**
1d	ă	e	ă/`	e/i	a	ele	ei	**elor**
....								
4	e	-	-	-	a	le	i	**lor**
4a	e	-	a	ă	a	le	i	**lor**
4b	e	-	oa	o	a	le	i	**lor**
....								
9	-	le	-	-	ua	lele	lei	lelor
9a	-	le	ea	e	ua	lele	lei	lelor
10	-	-	-	-	a	le	i	**lor**

The morphologic analysis also depends upon the given part of speech. The algorithm (implemented in ADX) uses the data structures that we have talked about.

9. Conclusions

Natural language processing is a very current domain, its techniques being indispensable to producing professional text editors. In natural language processing, the morphologic analysis has a very remarkable importance, its results having an overwhelming importance for the syntactic analysis of a phrase. If the classical approach (based on the idea of using a database with all the possible roots of the words and their inflected paradigms) has some disadvantages related to space

and speed, although the used algorithm is relatively simple, in this paper we emphasized a more complex algorithm that dynamically generates the roots of the words, starting from the phonological alternances. This has the advantage that a more reduced amount of information is required, it is more flexible and may be optimized by using adequately data structures.

In conclusion, the ADX agent will take each word edited in Word and will morphologically analyse it before searching for it in the Dex Online database. If the classical approach (elaborated on the idea of using a database containing all the possible roots of words and their inflected paradigms) has some disadvantages regarding space and speed, although the algorithm used is relatively a simple one, in this paragraph we have emphasized a rather more complex algorithm that dynamically generates the roots of words starting with the phonological alternances. The advantage in this case is that it requires a smaller amount of information, it is more flexible and can be improved by using structures of adequate data.

10. References

[1] Cristea, D. (1999). Let's Play With Words. Projects in Natural Language Processing, TEMPUS JEP 11168/97, Hamburg.

[2] Cosman C., Cristea D. Para-morph: a tool for paradigmatic morphology. Retrieved in January 1, 2009 from http://consilr.info.uaic./ro/ro/index.php?showpage=0603&showid=0 (at Romanian Consortium for the Computerization of the Romanian Language).

[3] Francu, C., Online Romanian Explanatory Dictionary. Retrieved in January 1, 2009 from http://www.dexonline.ro.

[4] Iliescu, M., Neagu, V., Nedelcu, C., Scurtu G. (1982). *The minimal Romanian language vocabulary for foreign students*, Bucharest: Editura Didactică şi Pedagogică.