Multiword Expression Translation Using Generative Dependency Grammar

Stefan Diaconescu

SOFTWIN Str. Fabrica de Glucoza Nr. 5, Sect.2, 020331 Bucharest, ROMANIA
sdiaconescu@softwin.ro

Abstract. The Multi-word Expressions (MWE) treatment is a very difficult problem for the Natural Language Processing in general and for Machine Translation in particular. This is true because each word of a MWE can have a specific meaning but the expression can have a totally different meaning both in source and in target language of a translation. The things are complicated also by the fact that the source expression can appear in the source text under a very different form from its form in a bilingual MWE dictionary (it can have some inflections) and, most of all, it can have some extensions (some MWE words can have associated new words that do not belong to the MWE). The paper show how this kind of problems can be treated and solved using Generative Dependency Grammar with Features.

1 Introduction

The translation problem of Multiword Expressions (MWE) is one of the most difficult problems of machine translation and even human translation. As [15] says, the MWE are "a pain in the neck of natural language processing (NLP)". There are many types of MWE classified on different criteria [15] [5]. MWE appear under different names and interpretations: collocations, compound words, co-occurrences, idioms, fixed syntagmas, lexical solidarities, phraseologisms, phraseolexemes, polylexical expressions [5]. Not all these names are fully equivalent but they have something in common: the fact that they are concerned with groups of words that must always be considered as a whole. The words that compose a MWE must have a representation that can indicate the fact that in an MWE instance into a text they can have eventually different forms (can be inflected) or even they can be replaced by other words that belong to some lexical classes. The MWE representation is specific to each grammar type: HPSG [5], Lexicon Grammar [7], Tree Adjoining Grammar [18] etc.

The most important problems of the MWE are followings:

a) The means of the MWE can be very different from the sense of each word of the MWE (the so called compositionality [1] or idiomaticity problem [16]). *Example:* John *kicked the proverbial bucket.*

b) MWE can appear with its word on different inflected forms (morphological variation [16] [17]). *Example:* John *turned on* the light. John *will turn on* the light.

c) MWE appears under different form by passivization, topicalization, scrambling, etc. (syntactic modification [1] or structural variation [17] [18]). *Example:* He *turned the tables on* me, and then I *turned them on* him. [13]

d) MWE appears with different extensions attached to some words of the expression (modifications [18]). Exemple: Il *est à bout des nerfs.*(He is exhausted.)) Il *est à bout maladif des nerfs.* (He is illy exhausted.)

e) MWE appears with different inserted word that some times are not directly linked with the expression words. *Example:* He *moved* John *to tears.* He *moved* a lot of people *to tears.*

f) MWE appears with different words that have similar sense (lexical variation [17] [18]). *Example:* John *sweep the dust under the carpet (rug, mat)* [13].

Solutions for these problems are not easy to find and perhaps, the great number of names of MWE are due (partially) to the fact that each sort of representation solves some aspects of the problems.

In this paper we will show a very general method to MWE representation based on Generation Dependency Grammar with Feature (GDGF) [2] [3] that solves many of these problems and can be used in machine translation (MT).

2 Representation of MWE in GDGF

We will show in an informal manner how MWE can be represented in GDGF. A more formal definition of GDGF can be found in [2] [3]. Let us have E_s a source expression from a text (phrase) T_s in the language with a lexicon L_s and E_t a target expression (equivalent with E_s) in the translation T_t (of the text T_s) in the language with the lexicon L_t. E_s and E_t can have a different number of words and sometimes with different senses (if we take the words one by one). In a bilingual phraseological dictionary (or MWE dictionary) that contains the correspondences between the expressions belonging to two languages L_s and L_t, ED_s and ED_t are the expressions in a basic form ("lemmatized form") corresponding to E_s and E_t. E_s is different from ED_s (it can be inflected or can have some insertions of other words, linked or not with the words of E_s by dependency relations. We will note E_x the extension of the ED_s in E_s. E_x will contain all the words that have dependency relations with the word from E_s. In the translation process E_s must be searched in the MWE dictionary (though it is not identical with ED_s) and the extension E_x must also be considered in order to create a similar extension for E_t.

The texts T_s (and E_s) and the entry ED_s (with its equivalence ED_t) in MWE dictionary will be represented in GDGF.

A GDGF is a rule collection. Each rule has in the left hand a non terminal and in right hand a syntactic part and a dependency part. The syntactic part is a sequence of non-terminals/terminals/pseudo-terminals/procedural-actions (*ntpa*). Each *ntpa* from the syntactic part have associated an Attribute Value Tree (AVT) that describes the syntactical/lexical categories and their

values. The dependency part is a dependency tree (dependencies between the *ntpa*-s from the syntactic part). Terminals are words that belongs to the lexicon (L_s or L_t in our case). Non-terminals are labels (symbols) for grammar categories (and are considered to not belonging to the lexicons L_s or L_t). Each non-terminal appears at least one time in the left side of a rule (with the exception of one symbol named root symbol). A pseudo-terminal is a sort of non-terminal that does not appear in the left side of a rule but it can be replaced directly by a word from the lexicon. A procedural-action (or action) is a routine that must be executed in order to identify the symbols from the text (for example a routine that identify a number).

In the MWE dictionary, an E_s (and the corresponding E_t) is represented only by one rule so it will not contain *ntpa* but only terminals, pseudo-terminals, actions (*tpa*). The left part of the right side of this rule will contain only *tpa*-s and the right part of the right side (the dependency tree) will be a final tree.

Until now all the GDGF description is conform to [2] [3]. We will introduce now something that is specific to the MWE treatment.

Each *tpa* from ED_s will have associated a "type of variability". There are three types of variability: invariable, partial variable and total variable.

a) Invariable *tpa*. A *tpa* is invariable, if it appears always in any context in the same form (so it will have associated always the same AVT or an AVT that subsumes the corresponding AVT from the MWE dictionary).

b) Partial variable *tpa*. A *tpa* is partial variable, if it has the same lemma (and lexical class) in all the apparitions of the expression in different contexts, and it has different lexical categories (it can be inflected).

c) Total variable *tpa*. A *tpa* is total variable, if it is any word (accepted by the grammar in the corresponding position), in all the apparitions of the expression in different contexts. More of this, in the source text, this *tpa* can be also a coordinate relation.

Example

Let us have the Romanian expression "*a abate pe cineva de la calea cea dreapta*" and an English equivalent "*to lead somebody astray*". The grammar rule for the Romanian expression is the following (we will do not indicate completely the AVTs):

<ED_s> -> ("*a abate*" [class = verb] "*pe*" [class = preposition] "*cineva*" [class = pronoun] [pronoun type = indefinite] "*de la*" [class = preposition] "*calea*" [class = noun] [gender = feminine] "*cel*" [class = pronoun] [pronoun type = demonstrative] [gender = feminine] [gender = feminine] "*dreapta*" [class = adjective] [gender = feminine], "*a abate*" (@r_1@("*pe*"(@r_2@("*cineva*"())))), @r_4@("*de la*"(@r_5@("*calea*"(@r_6@("*dreapta*")), @r_7@("*cel*")))))))

(Observations: In the rule, the terminals are lemmas. The inflected form is indicated by AVT. That is why the word "*cea*" from expression appears as "*cel*".)

The relations can have more intuitive names like "relations between noun and attribute" but we used only generic names like r_1, r_2, etc.

The grammar rule for the corresponding English expression is:

<ED_t> -> ("*to lead*" [class = verb] "*somebody*" [class = pronoun] [pronoun type = indefinite] "*astray*" [class = adverb], "*to lead*"(@r_8@("*somebody*"())), @r_9@("*astray*" ())))

We will define a correspondence between *cineva*() and *somebody*(), both being of total variable type:

Correspondence *cineva*() / *somebody*()

The Romanian expression can appear in a context like: "*... au abatut pe Ion si Maria de la calea cea dreapta ...*". The corresponding final rule will be:

<T_s> = (..."*a abate*" [class = verb] [mode = indicative] [person = III] [number = present] [time = perfect] "*pe*" [class = preposition] "*Ion*" [class = noun] [noun type = proper] "*si*" [class = conjunction] "*Maria*" [class = noun] [noun type = proper] "*de la*" [class = preposition] "*calea*" [class = noun] [number = singular] "*cel*" [class = pronoun] [pronoun type = demonstrative] [gender = feminine] [number = singular] "*dreapta*" [class = adjective] [number = singular] [gender = feminine] ..., ..."*a abate*"(@r_1@("*pe*"(@r_2@(@r_3@("*Ion*"(), "*Maria*"() /)))), @r_4@("*de la*"(@r_5@("*calea*"(@r_6@("*dreapta*")), @r_7@("*cel*")))))))...)

In order to find the expression "*a abate pe cineva de la calea cea dreapta*" in the source text we must make an equivalence between "*cineva*"() and @r_3@("*Ion*"(), "*Maria*"() /) that is a conjunctive relation represented as a coordinate relation. This is possible if we declare "*cineva*"() as "total variable". In the target expression, the terminal somebody() will be substituted by the coordinate relation @r_3@("*Ion*" (),"*Maria*" () /).

3 Transformation rules

Between the nodes (of type terminal / pseudo-terminal / action / relation - *tpar*) from the two expressions ED_s and ED_t we will indicate some transformation rules. There are two types of transformation rules (see figure 1):

a) Transformation rule of type "correspondence" between two nodes indicates that the two *tpar*-s are relatively equivalent in the two expressions. This means that *tpar* from ED_t, during the translation process, will take over all the links (with their descendants) corresponding to *tpar* from Es (that contains the links from ED_s but also the links from the extension E_x of ED_s in T_s). We will consider the following convention: if between two *tpar*-s from the two expressions ED_s and ED_t exists a correspondence, then they are of the same invariable, partial variable or total variable type. A source *tpar* or a target *tpar* can appear at most in a transformation of type correspondence.

b) Transformation rule of type "transfer" between tpa_1 from ED_s and tpa_2 from ED_t indicates the fact that the two *tpa*-s are not equivalent in the two expressions but, if tpa_1 from ED_s has some links that goes to the extension

E_x of the expression E_s (that corresponds to ED_s), then all these links with their descendents will be taken over by tpa_2 from ED_t (in order to obtain E_t). A transfer can not be defined between two relations.

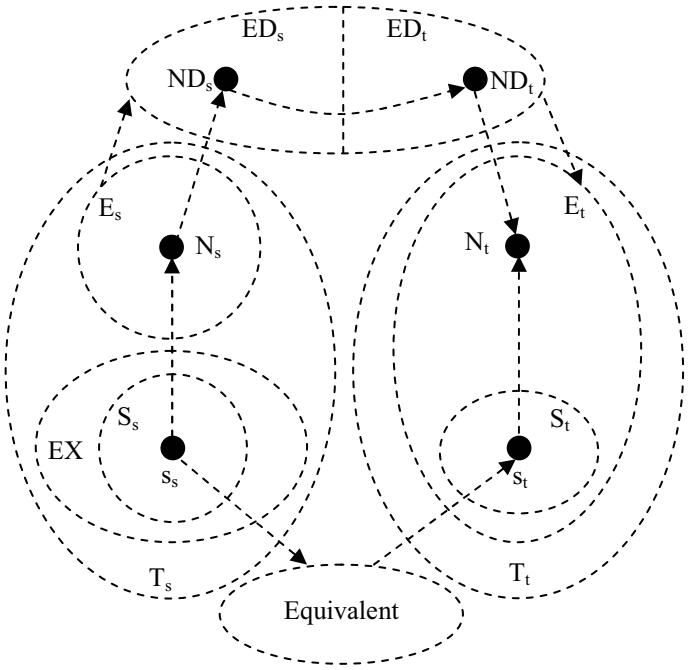

Fig. 1. Correspondences and transfers treatment

A source *tpa* that appears in correspondences or transfers is said to have nonexclusive relationships, i.e. it can have in T_s some links that go to the extension E_x of E_s.

A source *tpa* that do not appear in correspondences or transfers is said to have exclusive relationships, i.e. it can not have links in T_s that go to the extension E_x. These names are explained by the followings:

- If a source *tpa* is nonexclusive then it can have some descendents in the extension of the expression and these descendents must be attached (by correspondence or transfer) to a target *tpa* from the target expression, so, in this case, *tpa* must have a correspondence or a transfer.

- If a source *tpa* is exclusive, then it can not have descendents in the extension of the expression and it is not useful to put it in a correspondence or a transfer.

Not always is possible to transfer the elements from an extension of the source expression to others nodes of the target expression. An important feature that we can associate to the *tpa* that is the target of a correspondence

or transfer is the *combination feature*. Let us have a coordinate relation r_x = @relation name@ (a conjunction relation "and" for example) and a *tpa* s_t from target expression subordinated to another *tpa* t_t by the relation r_t. We consider now the *tpa* s_s from the extension of the source expression subordinate by the relation r_s so that $eq(r_s) = r_t$ (the equivalent of r_s from the source language is r_t in the target language) to another *tpa* t_s from source expression. We suppose that between t_s and t_t was defined a "correspondence" or a "transfer". We suppose also that the equivalent of t_s from the source expression is $eq(t_s)$ in the target expression. In this case the translation of the source expression together with its extension using the MWE dictionary will be made as follows: r_x will be subordinated to t_t, $eq(s_s)$ will be subordinated by $eq(r_s)$, s_t will be subordinated by r_t and $eq(r_s)$ and r_t will be coordinated by r_x. We will give later some examples to make the things more clear.

We make also the following important observation: the description in the MWE must be done so that the heads of the source and target expressions are always of the same type (both are relations or both are *tpa*-s).

4 Transformation cases

Besides the above notations we will use also the following notations too (see figure 1):

N_s = A source node from E_s (it is of type *tpa*).

S_s = The set of nodes (of type *tpa* or coordinated relations *cr*) linked to N_s by subordinate relations belonging to the extension E_x of E_s.

n_s = The number of nodes from S_s.

N_t = A target node from (it is of type *tpa*).

S_t = The set of nodes linked with N_t by subordinate relations and formed by the nodes that come from the MWE dictionary and by the nodes obtained by previous transfer or correspondence operations.

n_t = The current number of nodes from S_t.

ND_s = The source node from MWE dictionary corresponding to N_s.

ND_t = The target node associated by correspondence or transfer to ND_t.

r_x = A relation defined for a combination.

type(s) = The type of s (where s is a *tpa* node) i. e. the tuple formed by lexical class of s and the relation by which it is subordinated to another node.

sr = Subordinate relation.

cr = Coordinate relation.

We will have $type(s_s) = type(s_t)$ if:

- $eq(s_s)$ has the same lexical class with s_t;
- if s_s is subordinated by r_s and s_t is subordinated by r_t then $eq(r_s) = r_t$.

We will have $type(s_s)$ # $type(s_t)$ (# means not equal) if:

- $eq(s_s)$ has the same lexical class with s_t;
- if s_s is subordinated by r_s and s_t is subordinated by r_t then $eq(r_s)$ # r_t.

We will have s_s ~ s_t if:

- $eq(s_s) = s_t$

- if s_s is subordinated by r_s and s_t is subordinated by r_t , then $eq(r_s) = r_t$.

Using these notations we will present some transformation cases.

The transformations by correspondences or transfers using eventually the combination feature are used in order to recognize in the source text not only the MWE but also the MWE that have different extensions and in order to translate these extended MWE more appropriately. For example the expression "*în toiul noptii*" in Romanian language that has the English equivalent "*in the dead night*" will be recognized also under the form "*în toiul noptii negre*" and translated under the form "*in the dead black night*" if between "*noapte*" and "*night*" will be indicated a correspondence.

The correspondence, transfers and combinations can be defined only for *tpa* and not for relations.

A source element of a correspondence or a transfer can be:

- *tpa* with *sr* descendents;
- *tpa* with *cr* descendents;
- *tpa* without descendents.

A subordinate relation *sr* can be:

- *sr* with *tpa* descendents;
- *sr* with *cr* descendents.

A coordinate relation *cr* can be:

- *cr* with only *cr* descendents;
- *cr* with only *tpa* descendents;
- *cr* with *tpa* and *cr* descendents.

By combining these possibilities for the source and for the target element we will have a lot of situations. Not all of these situations are useful in practical cases.

When an element A of a source expression is in a correspondence or a transfer relation with an element B of an equivalent target expression, the descendents of A that go to the extension of the expression in which A appears must be attached as descendents of B (by its equivalents). But B can have also some descendents. In this case we must establish a method to put together the descendents of A and the descendents of B.

We will consider the next transformation cases:

a) Both A and B have descendents. There are many situations:

Transformation case a1: The node s_s is of type *tpa* and is linked with the node N_s by subordinate relation r_s. For each s_s belonging to S_s that have an s_t belonging to S_t so $type(s_s) = type(s_t)$ but $eq(s_s)$ # s_t we will do the followings:

- Create a new coordinate relation r_x with two entries r_x (1) and r_x (2).
- Mark that the equivalent of s_s is coordinated by r_x (1).
- Mark that s_t is coordinated by r_x (2).
- Mark that r_x is linked with N_t.

This type of treatment is named *attachment by coordination.*

The equivalences of the node s_s and of its eventual descendents will be transferred in the target language one by one.

Transformation case a2: The node s_s is of type *tpa* and is linked with the node N_s by subordinate relation r_s. For each s_s belonging to S_s so there is not an s_t belonging to S_t and type(s_s) = type(s_t) we will mark the fact that eq(s_s) is subordinated by eq(r_s) to N_t.

This type of treatment is named *attachment without coordination.*

Transformation case a3: The node s_s is of type *tpa* and is linked with the node N_s by subordinate relation r_s. For each s_s belonging to S_s so there is not an s_t belonging to S_t and s_s ~ s_t we will do the followings:

- s_s is subsumed by s_t and it will be ignored.
- r_s is subsumed by r_s and it will be ignored.

This type of treatment is named *attachment by subsumption.*

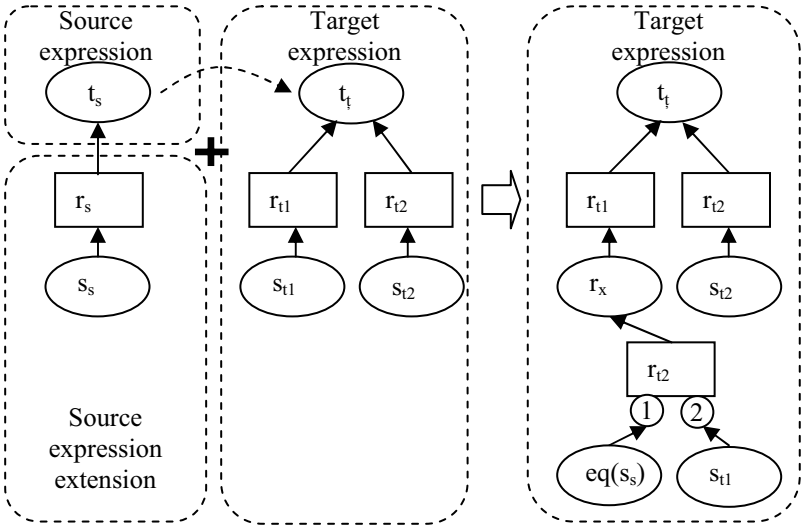

Fig. 2. Example of MWE dictionary entry

Transformation case a4: The node s_s is of type coordinate relation and is linked with the node N_s by a subordinate relation. For each s_s belonging to S_s we will mark that eq(s_s) is subordinated by eq(r_s) to N_t. It is as in case (a2) an *attachment without coordination.*

b) A have descendents but B has not descendents. In this case the A descendents will be passed to B without conflicts.

c) A has not descendents but B has descendents. In this case the problem disappears because we have nothing to pass from A to B. B will remain with its descendents.

d) Neither A nor B have descendents. In this case the problem disappears.

This cases cover if not all the situations but a very large number of possible situations.

Example

Let us have the expression "*a fi în gratiile cuiva*" in Romanian language and the equivalent expression "*être dans des bonnes grâces de qn.*" in French language ("to be liked by someone" or literally "to be in the graces of someone").

The grammar rule for the Romanian expression is (we will do not indicate completely the AVTs) (remember that the terminals are lemmas):

<ED$_s$> -> ("*a fi*" [class = verb] "*în*" [class = preposition] "*gratie*" [class = nom] [gender = feminine] [article = articled] [article type = definite] "*cineva*" [class = pronoun] [pronoun type = indefinite] [case = genitive], "*a fi*"(@r$_1$@("*în*"(@r$_2$@("*gratie*"(@$_3$@("*cineva*"()))))))))

The grammar rule for the corresponding French expression is:

<ED$_t$> -> ("*être*" [class = verb] "*dans*" [class = preposition] "*le*" [class = article] [article type = definite] [gender = feminine] [number = plural] "*bon*" [class = adjective] [gender = feminine] [number = plural] "*grâce*" [class = nom] [gender = feminine] "*de*" [class = preposition] "*qn.*" [class = pronoun] [pronoun type = indefinite], "*être*"(@r$_4$@("*dans*"(@r$_5$@(("grâce"(@r$_7$@("*le*"()), @r$_6$@("*bon*"()), @r$_8$@("*de*" (@r$_9$@("*qn.*"( ))))))))))

In the MWE dictionary something like in the figure 2 will appear. A correspondence will be defined in MWE dictionary between "*a fi*" (to be) and "*être*"and between "*gratiile*" (grace) and "*grâce*". A combination with the relation "*et*" (and) will be associated with the target "*grâce*".

Let us have now an expression with an extension "*a fi în gratiile languroase ale cuiva*" (literally "to be in the languishing graces of someone").The grammar rule for this expression is the following:

<Exp> -> ("*a fi*" [class = verb] "*în*" [class = preposition] "*gratie*" [class = nom] [gender = feminine] [number = plural] [article = articled] [article type = definite] [number = plural] "*languros*" [class = adjective] [gender feminine] [number = plural] "cineva" [class = pronoun] [pronoun type = indefinite] [case = genitive], "*a fi*"(@r$_1$@("*în*"(@r$_2$@("*gratie*"(@r$_0$@("*languros*"()), @r$_3$@("*cineva*"()))))))))

The general transformation case (of type (a1)) can be represented as in the figure 3. By applying this transformation case we will obtain the final translated rule (see figure 2):

<Exp.'> -> ("*être*" [class = verb] "dans" [class = preposition] "*le*" [class = article] [article type = definite] [gender = feminine] "*bon*" [class adjective] [gender = feminine] [number = plural] "*et*" [class = conjunction] "*langoureux*" [class = adjective] [gender = feminine] "*grâce*" [class = nom] [gender = feminine] "*de*" [preposition] "*qn.*" [class = pronoun] [pronoun type = indefinite], "*être*"(@r$_4$@("*dans*"(@r$_5$@("grâce"(@r$_7$@("*le*"()), @r$_6$@(@rx@("bon"(), "*langoureux*"())), @r$_8$@("*de*" (@r$_9$@("*qn.*"()))))))))))

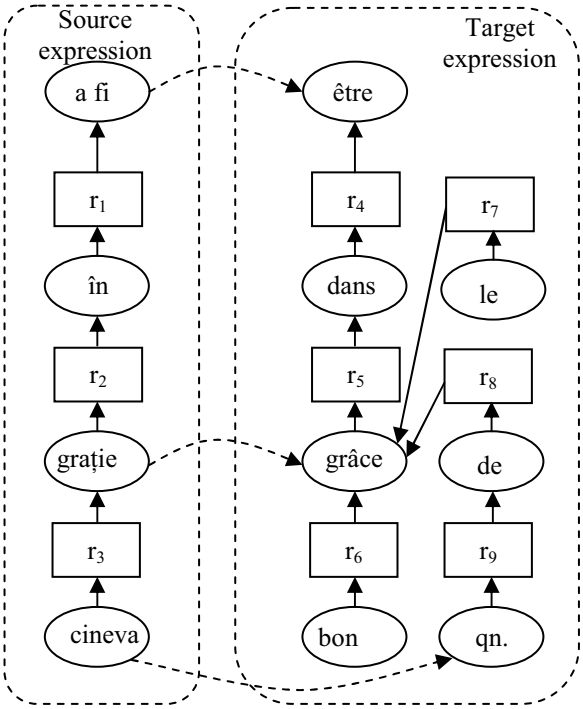

Fig. 3. Example - principle

The reconstruction of the surface text "*être dans les bonnes et langoureuses grâce de qn.*" will be done using a GDGF grammar (that is a reversible grammar [2]) of the target language and a mechanism to force the agreement between a subordinate and a terminal to which the subordination is realized [4]. This mechanism says the followings: if the node B is subordinated by a relation to the node A, then the attribute values of B will be forced accordingly with the attribute values of A as described in the agreement sections of the grammar rules.

5 Conclusions

We presented how GDGF solve some problems of MWE treatment. We added to the GDGF formalism the notions of correspondence, transfer, combination and variability of MWE components (invariable, partial variable, total variable) that allow the MWE description and the MWE dictionary construction. Using partial variability and the total variability, the method is important not only for MWE but also it can be used to create a sort of dictionary of grammatical structures. So, we can put in correspondence some specific structures of the source language and the target language. The ideas pre-

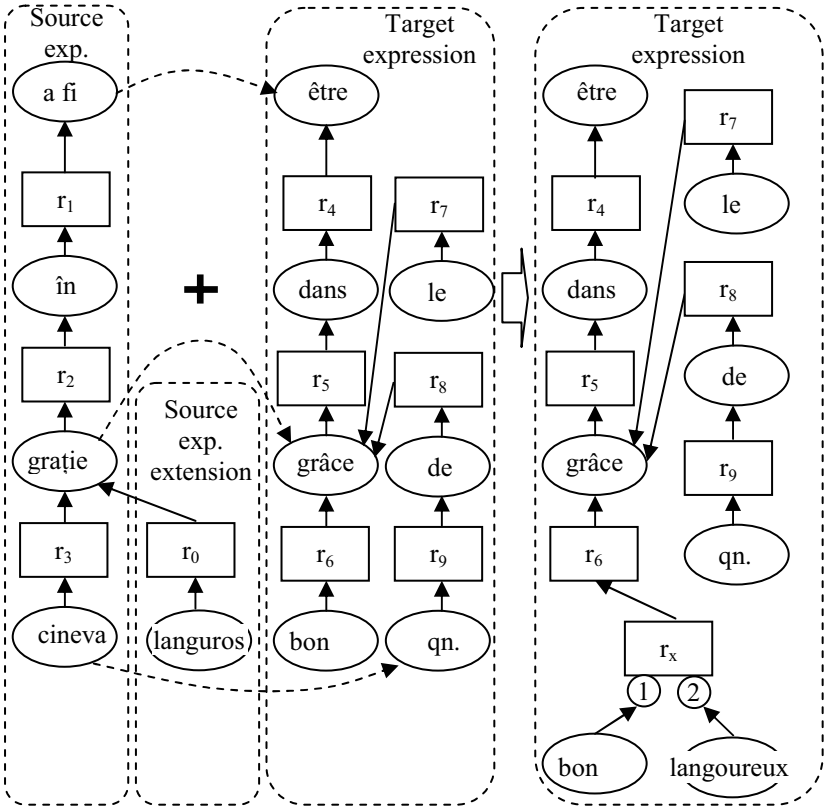

Fig. 4. Example - result

sented in the paper were introduced in a language GRAALAN (Grammar Abstract Language) that contains many other features, all of them based on GDGF. This language is designated to describe pair of languages in order to make possible the machine translation between them. GRAALAN is now used to describe Romanian language and the correspondences with French and English language.

References

1. Chang, N., Fischer, I. (2000) Understanding Idioms, in KONVENS 2000 / SPRACHKOMMUNIKATION, Konferenz zur Verarbeitung natrlicher Sprache - ITG-Fachtagung "SPRACHKOMMUNIKATION", Technische Universitt Ilmenau.
2. Diaconescu, Stefan (2003) Morphological Categorization Attribute Value Tree and XML, in Mieczyslav A. Klopotek, Slawomir T. Wierzchon, Krzysztof Trojanowski (Eds), Intelligent Infrmation Processing and Web Mining, Proceedings

of the International IIS: IIPWM'03 Conference, Zakopane, Poland, June 2-5, Springer 131-138.
3. Diaconescu, Stefan (2002) Natural Language Understanding Using Generative Dependency Grammar, in Max Bramer, Alun Preece and Frans Coenen (Eds), in Proceedings of ES2002, the 21-nd SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence, Cambridge UK, Springer, 439-452
4. Diaconescu, Stefan (2003) Natural Language Agreement Description for Reversible Grammars, in Proceedings of 16th Australian Joint Conference on Artificial Intelligence University of Western Australia, Perth 2-5 December , in print.
5. Erbach, G. (1992) Head-Driven Lexical representation of Idioms in HPSG, in Proceedings of the International Conference on Idioms,Tilburg (NL)
6. Fellbaum, C. (1998) Towards a Representation of Idioms in WordNet, in Proceedings of the Workshop on Usage of WordNet in Natural Language Processing Systems, COLING/ACL. Montreal, Canada, 52-57
7. Gross, M. (1986) Lexicon-Grammar - The Representation of Compound words, in Proceedings of COOLING
8. Jurafsky, D. (1933) A Cognitive Model of Sentence Interpretation: the Construction Grammar approach, University of California at Berkley, TR-93-077
9. Matiasek, J. (1996) The Generation of Idiomatic and Collocational Expressions, in Trappl R. (ed), Cybernetics and Systems '96, "Osterreichische Studiengesellschaft fur Kybernetik, Wien 1201-1205
10. McKeown K.R., Radev D.R. (1998) Collocations, in Robert Dale, Hermann Moisl, and Harold Somers (eds), A Handbook of Natural Language Processing Marcel Dekker
11. Pedrazzini S. (1999) Treating Terms As Idioms, in Proceedings of the Sixth nternational Symposium on Communication and Applied Linguistics, Santiago de Cuba, Editorial Oriente, Santiago de Cuba
12. Poibeau, T. (1999) Parsing Natural Language Idioms with Bidirectional Finite-State Machines, in Jean-Marc Champarnaud, Denis Maurel, Djelloul Ziadi (eds.) Automata Implementation, Third International Workshop on Implementing Automata, WIA'98, Rouen, France, Lecture Notes in Computer Science 1660 Springer
13. Pulman, S.G. (1993) The recognition and interpretation of idioms, in C. Cacciari and P. Tabossi (eds.) 'Idioms: Processing, Structure, and Interpretation', New Jersey: Laurence Earlbaum Cognitive Science Monographs, 249-270.
14. Riehemann, S. (1997) Idiomatic Construction in HPSG, Stanford University
15. Riehemann, S., Bender, E. (1999) Absolute Constructions: On the Distribution of Predicative Idioms, in Sonya Bird, Andrew Carnie, Jason D. Haugen, and Peter Norquest (eds.) WCCFL 18: Proceedings of the 18th West Coast Conference on Formal Linguistics
16. Sag, I., Baldwin, T., Bond, F., Copestake, A., Flickinger, D. (2002) Multiword Expressions: Pain in the Neck for NLP, in Alexander Gelbukh, ed., Proceedings of CICLING-2002, Springer
17. Segond, F., Tapanainen P. (1995) Using a finite-state based formalism to identify and generate multiword expressions, Rank Xerox Research Center Technical Report, MLTT, 19
18. Segond, F., Valetto, G., Breidt, E. (1995) IDAREX: Formal Description of Multi-Word Lexemes with Regular Expressions, Meylan Tubingen