Mathematical Problems of Computer Science 41, 131---134, 2014.

A Method of Constructing and Using the Tree of Possible Transformations from UNL to the Natural Language

Levon R. Hakobyan

Institute for Informatics and Automation Problems of NAS RA
e-mail: levon.r.hakobyan@gmail.com

Abstract

An approach to improving the transformation process from UNL to natural language is described. We introduce the tree of possible transformations, which lets us generate all the possible transformation paths for the current sentence and rule set. The implementation of the algorithm in UNDL Platform is described.

Keywords: UNL, Machine translations, Transformation rules, Grammar.

1. Introduction

Some new methods for improving the transformation rules from UNL to natural language are described.

The sentences in Universal Networking Language (UNL) contain some semantical information which is usually represented in different natural languages. Such semantical information is formalized in UNL by a structure of an oriented linked graph [1]. The information usually contained in a sentence on a natural language is transformed in UNL using oriented binary labeled links (which are treated as "relations' between nodes or nodes). Nodes are represented by "Universal Words", or simply "UW"; they express the corresponding concepts. The semantics of the sentence in UNL is expressed also by the so-called "attributes" which contain some information about modality of the used concepts.

We will consider one of the main systems for the UNL processing, namely, the UNDL Platform. It has been developed by UNDL Foundation in recent years [2].

The terms "analysis" and "generation" we will use as the names of the transformation process, respectively, from a natural language to UNL, and from UNL to the natural language. These processes are implemented in UNDL Platform respectively by IAN and EUGEN projects [2].

The current transformation algorithm applies a sequence of the rules to the current sentence. This sequence is created by using some kind of priorities. But other rules could be used for the current sequence.

So, we need some tools which let us find the most appropriate sequence of the rules to be applied to the sentence. In the existing transformation algorithms it is not easy to understand which rules sequence should be used.

To solve this problem we suggest extending the existing algorithms by adding functionality for the construction of the tree of possible transformations, also known as the tree of possible paths of transformation.

In this article we describe the tree of possible paths of transformation and its implementation in UNDL Platform system.

In the first part we give a short description of the transformation process done by the generator, and the description of the tree of possible paths of the transformation.

In the second part we give algorithms for the construction of the tree and information about classes and methods, used in the implementation in the UNDL Platform environment.

Our approach is developed for transformations from UNL to the natural language. But it could be used also for transformations from natural language to UNL, with some modifications.

2. The Tree of Possible Transformations

Transformations in the UNDL Platform system are implemented by consequently applying transformation rules on the UNL sentence. In case of the analyzer, in the result of this process the linear structure is transformed into a semantic graph. In case of the generator, the semantic graph is transformed into a linear structure.

In the process of the generation when a given UNL sentence is transformed to a sentence in the natural language the given dictionary and the set of transformation rules are used.

Any transformation rule has the form (α, β), where α is a predicate (i.e. condition which can be true or false for a given current sentence), and β is an algorithm defining the transformation of the considered current sentence. We assume that some number P such that $0 \leq P \leq 255$ is attached to any transformation rule; this number is called “priority coefficient”. If there are several rules which can be applied to a considered sentence then the rule having the greatest priority coefficient is chosen.

The process of transformation includes a finite number of steps. On any step only one transformation rule can be applied. The process is assumed to be completed when there is no rule which can be applied to the considered sentence; this sentence is admitted as the result of the process.

By changing the priorities, we can obtain more than one sequence of applicable rules. Using this we can construct the tree of possible paths of transformation. Node in this tree is a structure which contains, besides other additional data, a UNL sentence and a rule, by applying which the sentence was obtained.

In the tree every sub node contains, besides other additional data, a corresponding rule and a UNL sentence, which was obtained by applying the rule to the UNL sentence of the parent node.

In root node the rule should be null, and the sentence should be original to be transformed. The leaves obtained after the construction of the tree, will store all possible transformation results.

3. The Construction of the Tree

As we know, UNDL Platform is a web application. Computational time is very important for us, because it is bounded by server response time.

Vertices of the tree are implemented by the objects of the *TreeNode* class. The objects of this class contain, besides some basic and additional data about the vertex, links to a parent and the children of the vertex. Every object of the *TreeNode* class has its unique ID which lets us operate with such kind of objects; even then they are identical by other parameters.

The tree is stored in the *Tree* class. Static object of that class is created for every web request. The *Tree* class contains, besides some other data and methods, a *Layers* object which is implemented by the hash-table. The key in that hash-table is an Integer which shows the number of the layer in the tree. And its value is a linked list which stores links to vertices of the current layer, which are implemented by the objects of the *TreeNode* class.

This solution lets us have an access to vertices in the tree in two ways: first by the original way by recursive searching from root vertex, and second by accessing it by using the index and other parameters.

The second method lets us access to vertices faster.

In this article we describe two methods for the implementation of the tree in the UNDL Platform environment.

The first method is a depth-first search.

In the UNDL Platform a functionality was implemented, which lets us apply rules with the corresponding indices by passing a sequence of that indices. In the transformation process on each step, other rules which could be applied on this step, besides the applied rule, are registered by activating the corresponding feature.

This lets us get the first transformation path, and some free vertices after the first call of the transformation function. A free vertex is a vertex from which the construction of the tree could be continued.

After that, a loop is used to get the next free vertex. For that vertex the corresponding sequence of indices are constructed. This sequence is a path from the root to the current free vertex. After that, a function of transformation with this sequence is called. The loop continues as long as there are free vertices in the tree.

The second method for the construction of the tree of possible transformation is based on the breadth-first search.

There is a great probability of occurring of equal sentences in different vertices of the same layer, because on every transformation path the same subset of the rules could be applied in another sequence.

This lets us group equal vertices and continue the construction of the tree only for one vertex from a group. In this way, we can avoid analyzing some paths in the tree which leads to the same results and reduces the computational time.

In the current implementation, the breadth-first search is used up to some predefined layer in the tree. After that, the depth-first search is used for the corresponding vertices.

4. Conclusion

In this article we introduce the tree of possible transformations and describe its implementation in the UNDL Platform system. The tree of possible transformations lets us obtain all possible transformation results for the current sentence and rule set as the leaves of the tree. This lets the user compare these results with reference translation. Algorithms for the evaluation of translations like BLEU and METEOR can be used to compare transformation results. The most appropriate transformation path can be found by using these tools. This feature can be used as a tool for automated editing.

The experiments showed that the tree of possible transformations of the comparatively simple sentence (containing no more than 5 UWs) can be constructed by the corresponding programs using Test Drive Corpus in reasonable time and corresponding results set will contain no more than 130 possible transformations.

The main problem of using the methods described above is a computational complexity because of a huge amount of vertices on testing on real examples. This also leads to the problem of great amount of information, which makes the working process uncomfortable for the user.

Therefore, further researches should be done to make the algorithm work faster and to create an effective method for the users to deal with data which are generated in the result of the algorithm.

References

[1] H. Uchida, M. Zhu, and T. Della Senta, *A gift for a millennium*, IAS/UNU, 1999.
[2] UNDL Foundation web site, [Online]. Available: http://www.undlfoundation.org/.
[3] DeConverter Specification Version 2.6, UNL Center /UNDL Foundation 2002
UNL Center, Enconverter Specifications, Version 3.3 Tokyo, 2001

Submitted 24.12.2013, accepted 02. 03. 2014.

UNL լեզվից դեպի բնական լեզու տրանսֆորմացման հնարավոր ուղիների ծառի կառուցման մեթոդը

Լ. Հակոբյան

Ամփոփում

Հոդվածում նկարագրված է UNL լեզվից դեպի բնական լեզու տրանսֆորմացման գործընթացի բարելավման մեթոդը: Ներկայացված է տրանսֆորմացման հնարավոր ուղիների ծառը, որը թույլ է տալիս ստանալ բոլոր հնարավոր տրանսֆորմացման ուղիները տվյալ նախադասության և կանոնների խմբի համար: Հոդվածում նկարագրված է նաև համապատասխան ալգորիթմների իրականացումը UNDL Platform համակարգում:

Метод построения и использования дерева возможных направлений трансформации с языка UNL на естественный язык

Л. Акопян

Аннотация

Описывается подход к улучшению процесса трансформации с языка UNL на естественный язык. Дается метод построения дерева возможных путей трансформации, который позволяет генерировать всевозможные пути трансформации для данного предложения и набора правил. Описана также реализация соответствующих алгоритмов в системе UNDL Platform.