Some Approaches to the Automation of the Correction Algorithm of the NL–UNL Transformation Errors*

Aram A. Avetisyan
Institute for Informatics and Automation Problems of NAS RA
Yerevan, Armenia
e-mail:
a.avetisyan@undlfoundation.org

Levon R. Hakobyan
Institute for Informatics and Automation Problems of NAS RA
Yerevan, Armenia
e-mail:
levon.r.hakobyan@gmail.am

ABSTRACT
An approach to automated correction of UNL grammar rules for NL-UNL transformations is described. We claim that the development process of the rule sets could be made in a more effective way if some tools are used which can find the rules responsible for the possible errors. In previous articles we already presented such a method for UNL-NL transformations. In the article we present a classification of possible errors applicable for a result of the transformation process from NL to UNL and some modifications of the correction algorithm described in previous publications. An example of the error correction process is also considered.

Keywords
UNL, computational linguistics, machine translations, transformation rules, grammar

1. INTRODUCTION
UNL (Universal Networking Language) is an artificial meta-language used to describe the semantic information usually stored using the natural languages as detailed as possible [1,2].

Any sentence in UNL is an oriented linked graph also called a semantic graph and is believed to be logically precise, humanly readable and computationally tractable. In the UNL approach, information conveyed by natural language is represented sentence by sentence, as a graph composed of a set of oriented binary labeled links (referred to as "relations") between nodes (the "Universal Words", or simply "UW"), which stand for concepts. UWs can also be annotated with "attributes" representing modality [1,2,3].

Translation process from a natural language to UNL and from UNL into a natural language is called transformation. Corresponding applications use dictionaries and grammar rule sets for a specific natural language to provide the transformation. One of the main goals of the UNL project is the development of grammar rule sets.

In previous articles [3, 4] we described an approach to the automated correction of the transformation rules of UNL grammar for the UNL-NL direction of the transformation. Here we present an approach to applying these methods for the NL-UNL direction.

We introduce a classification of the errors which is applicable for the NL-UNL direction of the transformation process. This classification is different than the one was presented in [4] because we get different outputs as a results of the NL-UNL and the UNL-NL transformations.

----

* This work was supported by State Committee of Science, MES RA, in frame of the research project SCS 13-B321

We use already existing definitions to describe the transformation process because despite of different outputs which mainly rely on grammar, the transformation algorithm is the same for the both directions.
In the paper some modifications for the correction algorithm presented in [3] are also described.

2. A CLASSIFICATION OF NL-UNL TRANSFORMATION ERRORS
The result of NL-UNL transformation is presented in an oriented linked hyper-graph. Formally, hyper-graph is a pair $H = (X, E)$ where X is a set of nodes and E is a set of edges.

The obtained result in such structure may contain the following error types:

1. Absence of a specific subgraph: A situation, where in order to obtain a correct result, additional nodes $\{x\}$ need to be added to X and\or an additional relations $\{e\}$ need to be added to E and obtain the following hypergraph:
$H = (X \cup \{x\},\ E \cup \{e\})$

2. Excessive subgraph: A situation, where in order to obtain a correct result, excessive nodes $\{x\}$ need to be removed from X and\or an excessive relations $\{e\}$ needs to be removed from E and obtain the following hypergraph: $H = (X / \{x\},\ E / \{e\})$

3. Error inside a subgraph: In this case, we have a subgraph which may contain two types of errors:

3.a. Error in a relation
3.b. Error in a node

Each of this types can also be divided in two types.
For 3.a case we have following options:

3.a.1 Wrong type of a relation.
Such an error occurs when a wrong type of relation was chosen during transformation process. For example, a relation "obj(%x, %y)" should be replaced by the following relation "agt(%x, %y)".

3.a.2 Wrong orientation of a relation.
As we mentioned, the result of the transformation is an oriented hypergraph, so it is important to set the right orientation to a relation. For example, if a relation "obj(%x, %y)" has the wrong orientation it should be replaced by the following relation "obj(%y, %x)".

For 3.b. case we also have two types of possible errors:

3.b.1 Wrong UW

We have this situation when wrong UW was chosen for current node during the transformation process. For example, the node "boy(icl>male, icl>person).@def" should contain not "boy(icl>male, icl>person)" UW but the following UW "man(icl>male, icl>person)".

3.b.2 Wrong attributes

This kind of errors occur when a set of the attributes of the corresponding node is wrong. For example, if the node "knife(icl>instrument).@indef.@with" should contain the following set of the attributes "@def.@with" it should be replaced by the following "knife(icl>instrument).@def.@with".

4. In this paper we do not consider any other type of errors.

3. CORRECTION ALGORITHM

3.1. Basic definitions

All definitions used to describe the UNL-NL transformation process which are presented in [3] (Def. 1-8) including the pivot definitions of *responsible* and *indirectly responsible rules* are also applicable in NL-UNL case.

Rule is called affected on current node or relation if the current node or relation has been created or modified by applying the rule, or this rule's application has triggered the creation or modification of the mentioned node or mentioned relation in future steps.
Rule is called responsible for some sub-graph if its application has resulted from the corresponding sub-graph appearance in the transformation process.
Rule is called indirectly responsible for some sub-graph if its application has triggered a rule responsible for this sub-graph in future steps.
Of course, several rules can be responsible for a single sub-graph.
Taking these definitions into consideration, we can rephrase our goal as: functionality for searching all rules responsible for current nodes. [3]

3.2. Correction algorithm

The correction algorithm for NL- UNL direction is very close to the one was presented in [3] for UNL-NL direction. It also uses *affected rules* and *responsible rules* concepts to describe the roles of the rules during transformation process.
The algorithm is looking through all *affected steps* (steps on which affected rules were applied) and trying to find a responsible rule and the indirectly responsible rules.

The node or relation id should be used as an input parameter for the algorithm. Also a type of the error should be mentioned.

The algorithm returns a set of the directly and indirectly responsible rules.

As we mentioned in previous article [3], our approach consists of two parts: basic algorithm and the algorithms for the corresponding error type correction (modules). Basic part is an algorithm that reviews the steps of the transformation process and marks affected steps. Basic algorithm also consists of two parts: first part is used to process information about transformation steps during transformation process; second part reviews the steps to find affected steps.

The algorithms for the corresponding error type correction are responsible for correction of errors of types described above.

Basic algorithm reviews all transformation steps, marks some of them which have affected on current nodes, calls the corresponding module and passes data to that module depending on user input.
A module analyzes all steps marked as affected on current node by basic algorithm and marks the rules responsible for current nodes.

4. AN EXAMPLE

Let's consider a simple sentence "He killed her". The right translation of the current sentence into UNL is the following graph:

```
[S:1]
    {org}
       He killed her
    {/org}
    {unl}
       agt(kill:03.@past, 00:01.@3.@male)
       obj(kill:03.@past, 00:05.@3.@female)
    {/unl}
[/S]
```

Let's assume the transformation process gave a wrong result:

```
[S:1]
    {org}
       He killed her
    {/org}
    {unl}
       agt(kill:03, 00:01.@3.@male)
       obj(kill:03, 00:05.@3.@female)
    {/unl}
[/S]
```

It is easy to notice that the *.@past* attribute is missing from node *kill:03* in the given result
In this case we have to deal with a case 3.b.2 (see chapter 2)

When the user indicates the error, the correction algorithm tracks back the trace of processing, searching for a rule that contains the mentioned node.

```
(^AUX,^BLK,^[not],^[n't])(V,{^AUX|COP},PAS):=(
)(-PAS)
```

After finding the misleading rule, the user applies the needed improvements:

```
(^AUX,^BLK,^[not],^[n't])(V,{^AUX|COP},PAS,^@p
ast):=()(+att=@past)
```

After this modification the transformation process returns the correct result.

5. ACKNOWLEDGEMENT

The authors would like to thank Igor Zaslavsky, Yuri Shoukouryan, Vladimir Sahakyan and the State Committee of Science of MES RA.

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] Aram A. Avetisyan, Levon R. Hakobyan "An Automated System for Correction of Rules of Transformation from Universal Networking Language into Natural Language", Mathematical Problems of Computer Science Volume 41, IIAP NAS RA, Yerevan, pp. 114-121, 2014.
[4] Aram A. Avetisyan, Levon R. Hakobyan "A Classification of UNL –NL Transformation Errors and Some Approaches to the Automation of their Correction Algorithm", Preprint № 14-09/1, 04.09.2014, IIAP NAS RA, Yerevan, pp. 1-11, 2014.