Mathematical Problems of Computer Science 35. 5 13, 2011.

Development and Application of Interactive Algorithms for Natural Language to UNL and UNL to Natural Language Transformations

Aram A. Avetisyan

Institute for Informatics and Automation Problems of NAS of RA
e-mail: a.avetisyan@undlfoundation.org

Abstract

The present article describes certain developments worked out for the creation of NL-UNL transformation framework. In particular, new interactive algorithms aimed at analyzing and converting sentences of natural languages (NLs) into UNL and converting UNL structures back into texts of natural languages, are described. In this article, we describe the research on the work done for the creation of self-learning conversion system, based on the analysis of the interactive user-defined intervention. We also analyze the existing resources and describe certain algorithmic and technical solutions.

1 Introduction

UNL (Universal Networking Language) [1] is an artificial meta-language, designed for semantic information presentation. The aim of UNL is the storage of the meaning of text resources in the language-independent form. UNL can be used for various purposes, such as:

- Translation of sentences from one natural language into another.
- Semantic search of words and phrases.
- Gathering of semantic information.
- Collection of statistical data.

UNL stores the information in a form of semantic networks with hyper-nodes. Each sentence in UNL is presented as a directed coherent graph. In the UNL graph, nodes present concepts, and arcs present relations between concepts. The given concepts are called Universal Words (UW). The relations between UW are illustrated as directed arcs (arrows), defining the role of each word in a sentence.

By the end of 1990s UNL center in Tokyo published the first version of UNL specification and presented the programs for converting sentences of natural languages into UNL and vice versa, which were called correspondingly EnConverter and DeConverter.

1.1 Shortcomings and Difficulties of EnConverter and DeConverter Workflow

We should not underestimate the significance of the work done for the creation of EnConverter and DeConverter. however. with the lapse of time certain difficulties emerge while working with these programs. and one of the primary goals of the given article is to overcome these difficulties. The main difficulties [2] of work with the programs are listed below:

- Both programs are single-threaded. Operating system can launch only one instance of the program. activated by a certain process. This brings to difficulties in attempt to develop web applications and web services. that use EnConverter or DeConverter.
- EnConverter and DeConverter are applications of win32 type, which means that they are designed to work only in the environment of operating system Microsoft Windows. Whereas the majority of professional network servers use such operating systems as Linux, UNIX, Solaris etc.
- In the time of development of EnConverter and DeConverter machine resources were very limited and their economy was one of the primary tasks during program development. This is the main reason of some resource limitations.
- As input text resources the programs support only files in ASCI encoding, which brings to great difficulties, considering the fact that today the majority of available resources use UNICODE or its other variations.
- Not the least of the problems is a strict limitedness of grammar rules. A grammar developer cannot generalize certain rules in order to cover more cases, instead of it he/she has to describe each case separately, which in its turn increases the number of rules, and decreases the efficiency.
- The program source codes are not available for editing and developing, which again contradicts the UNL concept of openness.

Our goal is to create new more efficient algorithms of sentence conversion from UNL into natural languages and vice versa. bringing flexibility for rules syntaxes. The programs should allow the user to interfere in the process of transformation and direct it to a preferable path.We also try to develop a system. that allows to analyze the user revisions train.

2 LILY (Language-to-Interlanguage-to-Language System)

Despite all the improvements made in jDeCo [2]. it was decided to continue the work and make cardinal changes in the whole system of the UNL project. The UNDL Foundation launched the project LILY. where the Armenian side took the responsibility of transformation software development. The project includes the development of software for automatic and semi-automatic (interactive) transformation of natural language sentences into UNL (IAN) and transformation of UNL sentences into a natural language (EUGINE). Later on. UNL sentence transformer into NL sentences will be called EUGENE or Generator (the process of Generation), and the transformer of NL sentences into UNL sentences - IAN or Analyzer (the process of Analysis).

2.1 Grammar Rules

Grammar rules allow IAN and EUGINE to translate NL sentences into UNL (NL-UNL transformation rules) and vice versa (UNL-NL transformation rules). There are two types of grammar rules: *transformation rules and disambiguation rules* [3]. Transformation rules are used for transformation of UNL sentences into NL and vice versa, whereas disambiguation rules are used as meta-rules to improve the results of transformation rules application. Disambiguation rules define which rules should be applied and which rules should not, depending on the specificity of the current state. Both types of rules operate with the elements of NL sentence structure or to the elements of UNL graph. Transformation rules are defined as follows: $\alpha := \beta;$, where the left side is the condition and the right side is the action. *Disambiguation rules are* defined as follows: $\alpha = P :$, where the left side is the condition and the right side is an integer $0 < P < 225$, defining the possibility of a structure described in α (where $P = 0$- is the impossible case, and $P = 225$- the situation of highest possibility).

2.2 Dictionaries

The UNL-NL or NL-UNL Dictionary is a bilingual dictionary linking entries of the UNL Dictionary to entries of the NL Dictionary [4]. UNL-NL dictionaries are used for Generation process, while NL-UNL dictionaries are for Analysis. Those are lists of lexical items with their corresponding features. The structure of a dictionary entry is presented below, along with several examples:

[NLW] ID " UW (ATTR , ...) < FLG , FRE , PRI >;

Where:

NLW (Natural Language Word)

The lexical item of the natural language. Its format is decided by the dictionary builder. It can be:

- a multiword expression: [United States of America]
- a compound: [hot-dog]
- a simple word: [happiness]
- a simple morpheme: [happ]
- a non-motivated linguistic entity: [g]
- a complex structure: [[bring] [back]]
- a regular expression: [/colou0.1r/]

ID

The unique identifier (primary-key) of the entry.

UW (Universal Word)

The Universal Word of UNL. This field can be empty if a word does not need a UW. It can also be a regular expression.

ATTR

The list of attributes of the NLW (separated by ,). extracted out of the UNDL Foundation tagset.

FLG

The three-character language code according to ISO 639-3.

FRE

The frequency of NLW in natural texts. Used for natural language analysis (NL-UNL). It can range from 0 (less frequent) to 255 (most frequent).

PRI

The priority of the NLW. Used for natural language generation (UNL-NL). Ranges from 0 to 255.

3 The Development of Interactive Converter for IAN and EUGENE

The programs developed for sentence transforming from NL into UNL and vice versa, are based on the same algorithm, which will be described in this chapter. The programs were developed using Java2 programming language, which gives great versatility for their development and applications.

As it has already been mentioned, there are three types of resources necessary for the work of the transformers: a dictionary, a list of transformation rules, and input sentences (document). It is obvious that unlike the case of dictionaries and transformation rules, where the syntax does not depend on the fact whether they are designed for the Analysis or Generation, in case of input sentences this condition cannot be met. The Analyzer receives NL text as an input, whereas the Generator receives UNL sentences, and even though UNL sentence may contain its initial NL form, this information does not affect the generation process itself. Later we will see that at some stage of the Analysis, the sentence is reformatted into UNL, after which the main transformation process starts.

3.1 The Program Interface of IAN. The first Step of the Interactive Transformation

For a better understanding of the application workflow, we will start by examining the user interface of IAN. This is the program, which initiates the generation process itself, considering the user presets.

We can see the WEB interface of IAN program on Img. 1. The parameters indicating which resources are used by the program for transformation are shown on the left panel (a - d), and the instruments for editing the resources and for initializing of the process itself are shown on the right panel (e - k).

The first section is "Dictionaries, which is the section for a dictionary selection (Img. 1a). There are two options of dictionary input. The default option is the requesting dictionary entries from the database (in "Img. 1?, shown as "Dictionary Editor). If the user chooses this option, he/she will see the subsidiary application Dictionary Editor on the right side of the screen, where he/she can open the necessary dictionaries, edit, delete and add new ones. The converter, by checking the source of dictionary, will use only the dictionaries chosen by user. In the Dictionary Editor the user can also compile the necessary dictionaries from the database into a file and use this compiled dictionary for the transformation. This type of dictionary is shown in " Img. 1?, above the Dictionary Editor option.

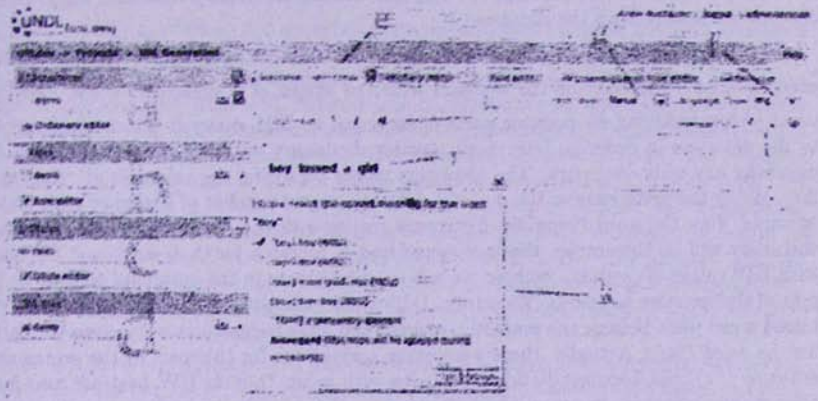

Fig. 1 IAN Application.

The "Rules section is placed under the Dictionary section. This section indicates the transformation rules to be used (Img. 1b). As in case with the dictionaries. the user can open the transformation rules editor and mark the necessary sets of rules from the database. He can also compile these sets from the database into a file and give it to the application input.

The next section is the section of disambiguation rules "DRules (Disambiguation Rules, Img. 1c), the analogous section of transformation rules. IAN does not require disambiguation rules for the process.

The last section of resources is the "NL Files section (Img. 1d), where the user selects the text document of NL sentences for the input.

By pressing the Process button the user receives access to the section, where he can initiate the transformation. First the program checks whether all the necessary resources are given and then, depending on the chosen language (Img. 1f), splits the text document into sentences. Then. according to the number of sentences. buttons. indicating each sentence. are created (Img. 1h). The selected sentence is converted into UNL and the trace. together with the final result. appears on the screen. How thorough the transformation is described in trace, depends on the value of "*Trace Leve* l (Img. 1g) field, set by the user; "*None* (final result only), "*Minimal*. "*Average*. "*Detailed* (detailed description), and "*Manual* (detailed description with the possibility to interfere in the process of choosing the convenient rule at any stage of transformation). When you press the button indicating the sentence number. the program checks whether the "*manual WSD* (Img. 1g, **WSD** - *Word Sense Disambiguation*) option is selected. WSD is a process, during which the sentence is splitted into words. phrases or other elements. and then those elements are matched with universal words from the selected NL-UNL dictionary.

If "*manual WSD* is selected. then the user can also control both the process of sentence splitting and the process of the UW matching. In this case when the user presses the button indicating the sentence number. the process of manual WSD is launched first. Depending on the type of chosen dictionary (database or compiled). different algorithms will be used for dictionary entry retrieving and matching. The matching algorithm for compiled dictionary

is described in the article [3]. here we will describe the dictionary matching algorithm for dictionaries loaded from the database.

In order to find matches from the dictionary. it is necessary to split the sentence in all possible ways. Thereby. for the sentence with the length of n symbols, we can have $\sum_{k=1}^{N} k$ parts. After receiving all possible parts of sentence, an SQL query is constructed and sent to the database in order to receive all possible dictionary entries with values in HW field matching any indicated part. The algorithm works regarding the principle of "the longest first. giving the preference to the matches with the largest number of letters or symbols. For example. if for the word "republic, dictionary entries with the identical HW field are found. then they will be the entries, that are considered as a match for that word, and the entries with HW values of "public, "pub or "re will not participate in the matching process for that part of the sentence anymore. The "Img. 1j illustrates the situation where the sentence "boy kissed a girl went though the matching process and. four suggestions were given as matches for the word "boy. Actually, there were other suggestions for this part of the sentence (for example- "b), but because dictionary entries with value "boy in HW field are also found. then shorter matches are not considered anymore. Consider the case when the principle "the longest first prevents from choosing the correct entry. Let's take, for example, a sentence including the following phrase: " this vitamin actively. In this case the WSD process can choose the following division "this, " . "vitamin a. "ctively, even though it will not find any matches for the part "ctively. In that case the user can change the automatically chosen division. The double click on the sentence line will show him/her the text division of the sentence in the text form of "this--- ---vitamin a—ctively, where the chosen parts of the sentence are separated by "— symbol. By adding. deleting or changing the position of these symbols, the user makes his/her own division. thus, improving the result of the matching process. This opportunity is the very first milestone for realization the idea of the interactive process of NL-UNL transformation.

Later on. when examining each part of a sentence, a window with a number of suggestions will open for that part, giving a short description of each suggestion (Img. 1k). The suggestions are sorted in descending order of the field "frequency (the matches are presented as dictionary entries. and NL-UNL dictionary entries use "frequency fields in order to determine the priority of the entry). The first suggestion is the chosen by default (with the highest value of the "frequency field). The user is given an opportunity to choose another suggestion as a match. The user can also define this part of the sentence as a "temporary one (Img. 1k: suggestion "[TEMP] a *temporary element*). Generally we define as "temporary the parts. which do not play any role by themselves, but can characterize another part of the sentence or even the whole sentence. For example. in the sentence "boy kissed a girl the letter "a is nothing more but an element. showing that the word "girl is indefinite. There is also a "Disregard option. By selecting this option the user removes the given part from the sentence (Img. 1k: suggestion "[Disregard] *this node will be ignored during processing*). The possibility to revise the choice of the dictionary entries for each part of the sentence is another precondition for the interactivity.

When all the parts of the sentence receive their matches the WSD process is finished. By pressing the "Finish button. the user orders to start the transformation process.

. In case the option "*manual WSD* is not selected the process is analogous to the previous one. the only difference is that the division and selection of suggestions is done automatically.

3.2 The Conversion Process

As it was already mentioned, the conversion process is almost identical for both directions (Generation and Analysis), and the only difference between them is the direction of the transformation rules. If the transformation rules have the RR, RL, LL direction, then there will be a transformation from UNL sentence into NL (Generation), and if the rules have the LL, LR, RR direction, then the NL sentence will be transformed into UNL (Analysis). When we want to make the algorithm to work in both directions, we need to satisfy the condition where input and output data is the same type (match the same syntax) in both cases. We already know that it is impossible to present UNL sentence as an ordered list, without any transformations, this means that in order to accomplish the condition mentioned above we will have to present the NL sentence in a way corresponding to the UNL syntax. The fact that after WSD process the sentence can be presented as an ordered list of Universal Words, which in its turn can be presented as a UNL sentence (where the relations are replaced by the list of Universal Words), will be of great help.

Thus, instead of an NL sentence, a UNL sentence will be given as an input to the Analyzer, and only after the process of transformation the UNL sentence will correctly reflect the given NL sentence.

Thus, a UNL sentence is always presented to the program, so the direction of the process depends only on the transformation rules direction. In our case the rules run the process in the direction of UNL relation creation (LL - LR - RR). The UNL sentence objects have two main arrays; one arrays contains all the relations, and the other one - all the nodes. The aim of NL-UNL transformation rules is to create UNL relations, based on the content of the node array, gradually emptying it and filling the array of relations.

3.2.1 The Process of Automatic or Interactive Application of Rules

Before starting the process of rule application, the program receives a document containing UNL sentences, a list of the dictionary entries, chosen during the WSD process (manual or automatic), lists of transformation and disambiguation rules, and the numbers of sentences, that should be used for transformation.

The rule application algorithm is cyclical, and in order to avoid loops, we use loop revealing algorithm and we also set the maximum number of rules application. When the maximum number of rule applications is reached, the transformation process stops.

The rule application process is the final stage of transformation. It is not difficult to notice that it is very important to provide the user with an interface of rule application control. In order to be able to interfere in the process of appropriate rule choosing, the user should choose the "*Manual* trace level option before beginning the transformation. In that case, the program processes the transformation and gives the result with a detailed description of the rule application process, giving a list of the applicable rules for each step and the applied rule itself. Here the user can change the choice of the program by selecting another applicable rule of his/her choice. After the choice is made, the process launches again, but this time considering the user choice for the given step. We can continue this way, by selecting the most appropriate rules for each step of conversion.

Thus, we can say that the mission of interactivity is complete, due to the possibility of user interference into any stage of transformation process.

4 Self-Learning

Despite the fact that the interactive transformation process helps us to improve the transformation results. it is obvious that it may be rather inconvenient to make the same corrections in many similar cases. and sometimes even impossible (for instance, processing documents containing thousands of sentences). A need for a new feature, which will allow us to avoid continuous corrections of common mistakes, emerges. In order to develop this feature. there are two steps that need to be completed:

- Development of a method of user interference analysis (learning).
- Application of the knowledge gained during the process of education.

It was decided. that the most convenient ways of user interference analysis and application of this experience is the automatic generation of the disambiguation rules. expressing user preferences, and adding them to the disambiguation rules list during further transformations.

5 Conclusion

We had a goal to develop new software for NL-UNL transformations, which would be able to eliminate the previous software shortcomings and offer new algorithmic and technical improvements. The newly developed system LILY resolves all the mentioned issues, offering new more flexible grammar rules, improved algorithms for fast dictionary search [5] and rules matching. The transformation process can be both entirely automatic and interactive. not only allowing the user to trace the transformation process but also easily control it. The interactivity of the process gave us the possibility to develop a mechanism. allowing the system to train and improve the quality of the transformation results. This facility becomes a breakthrough on the way of grammar rules development.

References

[1] H. Uchida. M. Zhu. "The Universal Networking Language(UNL) specifications". version 7, UNDL Foundation. June, 2005.

[2] A. Avetisyan. "Some approaches to the generation of sentences in natural language from UNL". *Proceedings of the conference CSIT*, p. 268,Yerevan . 2009.

[3] "Grammar Specifications", *UNDL Foundation online resources.* www.unlweb.net. 2011.

[4] "Dictionary Specifications". *UNDL Foundation online resources.* www.unlweb.net. 2011.

[5] I. Zaslawskiy. A. Avetisyan. V. Gevorgyan, "Implementation of Dictionary Lookup Automata for UNL Analysis and Generation", *International Journal of Information Theories and Applications.* vol. 17, n. 4. Sofia. Bulgaria. 2010.

Բնական լեզվով նախադասությունների համընդհանուր ցանցային լեզվի և հակառակը փոխակերպման համար երկխոսական ալգորիթմի մշակումը և կիրառումը

Ա. Ավետիսյան

Ամփոփում

Մեր առջև դրված էր խնդիր՝ մշակել նոր ծրագրային համակարգ NL-UNL երկկողմանի փոխակերպումների համար, որը կկարողանա լուծել նախորդ ծրագրային միջոցների կիրառման հետ կապված խնդիրները և կառաջարկի նոր ալգորիթմական և տեխնիկական լուծումներ: Նոր մշակված LILY համակարգը լուծում է բոլոր վերոհիշյալ խնդիրները, առաջարկելով կանոնների ներկայացման նոր ու ճկունացված տեսք, բառարանային փնտրման և կանոնների համապատասխանության համար մշակված ալգորիթմեր: Փոխակերպման ընթացքը այժմ կարող է լինել ամբողջովին ավտոմատացված կամ կիսաավտոմատացված (ինտերակտիվ), թույլ տալով օգտագործողին ոչ միայն հետևել փոխակերպման ընթացքին, այլ նաև հեշտությամբ կառավարել այն: Համակարգի երկխոսական լինելը, ընդգծում է նոր հնարավորություն մշակելու մեխանիզմ, որը թույլ կտա համակարգին սովորել և բարձրացնել փոխակերպման արդյունքների որակը: Այս հնարավորությունը կարևոր քայլ է բերականական կանոնների զարգացման համար: