Tree structure of Slovak sentences

Michaela Linková[1] and Stanislav Krajči

Institute of Computer Science, Pavol Jozef Šafárik University in Košice, Slovakia,
michaela.linkova@student.upjs.sk

Abstract: In this work-in-progress paper, we propose the algorithm for creation of a tree structure of the Slovak sentence. The tree structure of a sentence represents the relationships and dependencies between words in a sentence. The root of the tree is predicate. Finding the right sentence structure helps to understand its meaning better. In Slovak language, words have different forms, and there are various ways how to compose sentences. We found that the algorithm properly works for simple sentences with one predicate and subject and sentences connected with one conjunction.

1 Introduction and related works

The natural language processing in the Slovak language is a very actual and fruitful area of research interest. It is the most challenging issue to arise in recent years. The Slovak language belongs to a group of a flexible language and has complex rules for word inflection as there are many possible word forms, classification of contexts. One part of natural language processing is understanding the structure of sentences. This work-in-progress paper proposes the algorithm for creation of a tree structure of the Slovak sentence. Algorithm is not base on statistical data from the corpus, but takes raw data from Tvaroslovník. It is a database of all forms of Slovak words. The tree structure of a sentence can represent the relationships and dependencies between words in a sentence. The root of the tree is a predicate. The tree structure for Slovak sentence: "Lucia číta veľmi peknú knihu." [1] is shown in Figure 1.

[1] Lucy is reading very beautiful book.

The natural language processing in the Slovak language is a very actual and fruitful area of research interest. It is the most challenging issue to arise in recent years. However, we can find several tools, dictionaries and conferences in this area of research. For example, Paper Online Natural Language Processing of the Slovak Language presents the web site for NLP tools such as lemmatization, correction, finding part-of-speech of words in the sentence and others [1]. Paper Morphological analysis of Slovak language introduces a statistic algorithm of segmenting words by identification of a suffix. This ability to identify suffix helps to classify even unseen words in the training corpus [2]. It is necessary to have more information about words to work with them. Language Institute of Ľudovit Štúr offers a wide selection of dictionaries. These include a Dictionary of the Slovak language, Slovak spelling rules dictionary, Dictionary of foreign words, Synonymous dictionary and much more [3]. It also provides the Slovak National Corpus. It is an electronic database, mainly containing Slovak texts from 1955 from different styles, genres, thematic areas, region and other. Every two years, the institute organizes a conference SLOVKO on natural language processing [4].In 2017, Slovak Dependency Treebank in Universal Dependencies was created form Slovak National Corpus.[5] Database of Slovak words and their forms Tvaroslovník was created at Pavol Jozef Šafárik University at Košice [6], [7]. Master thesis Syntaktická analýza slovenskej vety pomocou Tvaroslovníka deals with the creation of an algorithm for finding the structure of the sentence [8]. Statistical and machine learning approaches have been developed in recent years. For example two-stage multilingual dependency parser, which was evaluate on 13 diverse languages including Czech language [9] or a neural network classifier for use in a greedy, transition-based dependency

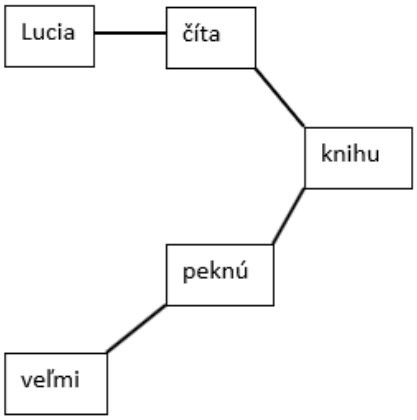


Figure 1: Example of sentence tree structure sentences "Lucia číta veľ mi peknú knihu."[2]

parser [10].

Slovak and Czech language have similar grammatical rules and structure of sentences. Therefore tools and algorithms for natural language processing of the Czech language can also be inspiration for algorithms for the Slovak Language. There are three main universities in the Czech republic that deals with natural language processing. One is Masaryk University in Brno and the other is Charles University in Prague. Masaryk university created tools for adding diacritics into texts, topics detection, named entity recognition, morphological analyzer "Majka" and other [11]. Institute of Formal and Applied Linguistic at Charles University developed tools for annotation, tagging, correction and also morphological analyzer and other [12]. This institute develops s a trainable pipeline "UDPipe" which performs sentence segmentation, tokenization, lemmatization and dependency parsing[13]. The third is Institute of Theoretical and Computational Linguistics at Charles University. This institute develops computational tools for automatic language processing, for example syntactic annotation of Czech corpora or grammar-based treebank of Czech [14].

2 Slovak syntax

Grammatical rules define the structure of a sentence. A sentence member is a basic unit of a sentence. It is part of a sentence, which is in some relation to the other parts. Main sentence members are subject and predicate.

Subject is part of the sentence, which describes who or what is doing something. It could be noun, adjective, pronoun or numeral. Subject should be in the first case. In Slovak sentences, subject could be expressed or unexpressed. Unexpressed subject means that subject is not in this sentence. For example, in the sentence "Nakupujeme", which means We are shopping, is only verb and no subject. Verb "Nakupujeme" is in plural and in first person form, it is noticeable in Slovak language that subject is "we". Predicate expresses an action or situation of the subject. It could be verb or auxiliary verb plus noun, adjective or numeral. In Figure 1., subject is "Lucia" and predicate is "číta".

Other sentence members are an object, attribute and adverbial. Object specifies predicate and shows object. It could be noun or pronoun and it should not be in the first case. Adverbial gives more information about time, place, manner or cause. Adverbial could be adverb or noun. Attribute modifies subject and it is adjective, pronoun or numeral. There are two types of

attribute. One type has to have the same grammatical categories as subject and the other type has at least one of the grammatical categories different from subject. In Figure 1., object is "knihu" and attribute is "peknú". In the sentence "Deti prišli večer."[3], adverbial is "večer". Every sentence member except for predicate can be in a sentence more than one time. Conjunction or commas connect same sentence members. These sentence members create a relation between them. The word which depends on the other word in relation is called dependent and another word is superior.

The Slovak language distinguishes three types of sentence: simple, compound and fundament sentence. Simple sentence has only one predicate. Compound sentence has two or more predicates. Part of compound sentence are connected by conjunction or commas. The third type has no subject, only verb or noun. Verb in these sentences usually describes general action, for example "Prší" [4] [15].

3 Finding structure of sentence

Tvaroslovník is a database of all forms of Slovak words. It was created at the University of Pavol Jozef Šafárik. The database contains a lot of Slovak words, their form and other information about these words. Every row contains information about form of the word, its part-of-speech and grammatical categories of the word. Data in Tvaroslovník was collected from the dictionary of Slovak language. All data and information are saved in one table. There is a list of columns:

- idWord: unique identification number for word,
- idForm: unique identification number of word's form,
- form: a form of a word,
- part-of-speech,
- categories: grammatical categories, there are different for every part-of-speech.

Table 1. shows an example of records for the word "kniha" [5].

The algorithm for finding a tree structure of a sentence has Slovak text as input. In the first step, an input is split into sentences by using a dot as a separator. Sentences are put into list of sentences. Then algorithm iterates over list of sentences. It finds all forms and characteristics for each word in a sentence by using Tvaroslovník. In the next step, the algorithm finds out how many of a predicates sentence has. If there is more than one predicate, the algorithm separates sentence into smaller parts. Separation is done according to conjunction or coma. The next steps are same as steps for sentences with one predicate. However, the algorithm uses these smaller parts instead of a whole sentence. After identifying the type of sentence, algorithms checks how many of same sentence members are in a sentence. If there is more than one, the algorithm takes this part of a sentence and creates a relation between each member and comma or conjunction, which is connecting these members. These relations are added to the list of possible relations of sentence. After that, the algorithm takes tuples of words, which are standing next to each other and tries to choose the possible relationship between those words. The algorithm has defined 29 types of relations between words. Except for the definition of part-of-speech and grammatical categories, every relation has its priority defined by empirical experience. All types of relations and their priorities are presented in Table 2.

For each tuple algorithm iterates over all form of words of tuple and try to find suitable word forms according to possible types of relations. After choosing the type of relation for each tuple, relations are put in the list of possible relations and sorted according to priority from the highest to the smallest. If there is more than one possible relation, algorithm put into list of possible relations all relations. Next, the algorithm selects the relation with the highest

[3] Children came in the evening.
[4] It is raining.
[5] book

idWord	idForm	form	part-of-speech	categories
1	0	kniha	noun	gender: feminine; number: singular; case: nominative
1	1	knihy	noun	gender: feminine; number: singular; case: genitive
1	2	knihe	noun	gender: feminine; number: singular; case: dative
1	3	knihu	noun	gender: feminine; number: singular; case: accusative
1	4	knihe	noun	gender: feminine; number: singular; case: locative
1	5	knihou	noun	gender: feminine; number: singular; case: instrumental
1	6	knihy	noun	gender: feminine; number: plural; case: nominative
1	7	kníh	noun	gender: feminine; number: plural; case: genitive
1	8	knihám	noun	gender: feminine; number: plural; case: dative
1	9	knihy	noun	gender: feminine; number: plural; case: accusative
1	10	knihách	noun	gender: feminine; number: plural; case: locative
1	11	knihami	noun	gender: feminine; number: plural; case: instrumental

Table 1: Tvaroslovnik

priority at the list. Selected relation is added to the list of final relations. This list helps built the tree. After that, the tuples are removed from the list of possible relations, and the dependent word is removed from a sentence. Every relation, which has the same dependent word and other superior word as selected relation, is removed from the list of possibilities. Then words next to the removed word are candidates for a new relation, therefore algorithm checks if they can form relation. If there is some type of relation, this new relation is added to the list of possible relations, and then the list is sorted according to priority again. If there are two relation with same dependent and superior word form, but they have different type of relation as selected relation, algorithm creates copy of list of possible relation. Algorithm creates from copied list other possible tree structure with different types of relations. The process is repeated again until there is only one word in a sentence.The last step is to build a tree structure from the list of final relations. Algorithm for finding tree structure is implemented in Java programming language and data are stored in Java Collections classes List and Map. If there is some ambiguity in relation, algorithm finds all possible tree structure of sentences

The example below illustrates the work of the algorithm. Input is the sentence "Lucia číta veľmi peknú knihu." [6] First, the algorithm creates a list of words from a sentence and finds all possible forms from Tvaroslovník and puts it to the map. In our example, the algorithm generates the following map:

- Lucia: [idWord: 128848, idForm: 1, form: Lucia, part-of-speech: noun, categories: gender: feminine; number: singular; case: nominative]
- číta: [idWord: 8679, idForm: 6, form:číta, part-of-speech: verb, categories: person: third; number: singular; time: present]
- veľmi: [idWord: 102690, idForm: 0,form: veľmi, part-of-speech: adverb, categories: None]
- peknú: [idWord: 56578, idForm: 17, form: peknú, part-of-speech: adjective, categories: gender: feminine; number: singular; case: accusative]
- knihu: [idWord: 27834, idForm: 4, form: knihu, part-of-speech: noun, categories: gender: feminine; number: singular; case: accusative]

After that, the map of possible relations is created with their priorities and it is sorted according to priority:

- dependent: veľmi and superior: peknú, priority: 9

[6] Lucy is reading very beautiful book.

Dependent	Superior	Priority	Required grammatical categories	Type of relation
Noun	Conjunction	10	None	Multiple sentence member with noun
Adjective	Conjunction	10	None	Multiple sentence member with adjective
Pronoun	Conjunction	10	None	Multiple sentence member with pronoun
Numeral	Conjunction	10	None	Multiple sentence member with numeral
Adverb	Adverb	9	None	Complex adverbial
Adverb	Adjective	9	None	Complex attribute
Pronoun	Pronoun	9	None	Two pronoun
Pronoun sa, si	Verb	8	None	Reflexive verb
Preposition	Adjective	7	Same case	Preposition with adjective
Preposition	Pronoun	7	Same case	Preposition with pronoun
Preposition	Noun	6	Same case	Preposition with noun
Auxiliary verb	Verb	6	None	Complex predicate
Auxiliary verb	Noun	6	None	Complex predicate
Auxiliary verb	Adjective	6	None	Complex predicate
Auxiliary verb	Pronoun	6	None	Complex predicate
Pronoun	Adjective	5	Same gender, number and case	Complex attribute
Pronoun	Noun	4	Same gender, number and case	Pronoun attribute
Adjective	Noun	4	Same gender, number and case	Adjective attribute
Noun	Noun	3	Nouns shouldn't have same gender, number or case	Attribute
Preposition	Verb	3	None	Preposition with verb
Pronoun	Verb	2	Pronoun shouldn't be in the nominative case	Pronoun object
Noun	Verb	2	Noun shouldn't be in the nominative case	Noun object
Adjective	Verb	2	Noun shouldn't be in the nominative case	Adjective object
Numeral	Verb	2	Noun shouldn't be nominative case	Numeral object
Adverb	Verb	2	None	Adverbial
Noun	Verb	1	Noun should be in the first case	Noun subject
Adjective	Verb	1	Adjective should be in the first case	Adjective subject
Pronoun	Verb	1	Pronoun should be in the first case	Pronoun subject
Numeral	Noun	1	Numeral should be in the first case	Numeral subject

Table 2: Relations and their priorities

- dependent: peknú and superior: knihu, priority: 4
- dependent: vel'mi and superior: číta, priority: 2
- dependent: Lucia and superior: číta, priority:1

First iteration Relation with priority 9 is chosen. This relation is added to the list of final relations and removed from the list of possible relations. The algorithm is going through the list of possible relations and removes every relation with dependent word vel'mi. Dependent word is removed from a sentence, and the algorithm checks if a new relation is created. After removal, there is a new possible relation between words číta and peknú with priority 2. This new relation is added to the list of possible relations, and the list is sorted again. After the first iteration, there is the sentence "Lucia číta peknú knihu."[7].The list of final relations is:

- dependent: vel'mi and superior: peknú, priority: 9

And the list of possible relations is:

- dependent: peknú and superior: knihu, priority: 4

[7]Lucy is reading beautiful book

- dependent: peknú and superior: číta, priority: 2
- dependent: Lucia and superior: číta, priority:1

Second iteration Relation with priority 4 is selected. This relation is added to the list of final relations and removed from the list of possible relations. the algorithm is going through the list of possible relations and removes every relation with dependent word peknú. Dependent word is removed from sentence, and the algorithm checks if a new relation is created. There is a new possible relation between words číta and knihu with priority 2. This new relation is added to the list of possible relations, and list is sorted again. After second iteration, the sentence is "Lucia číta knihu."[8]. The list of final relations is:

- dependent: veľmi and superior: peknú, priority: 9
- dependent: peknú and superior: knihu, priority: 4

And the list of possible relations is:

- dependent: knihu and superior: číta, priority: 2
- dependent: Lucia and superior: číta, priority:1

Third iteration Relation with priority 2 is selected. This relation is added to the list of final relation and removed from the list of possible relations. The algorithm is going through the list of possible relations and removed every relation with dependent word knihu. Dependent word is removed from sentence and it checks if a new relation is created. However, there is no possibility to create a new relation. After third iteration, the sentence is "Lucia číta.".[9] The list of final relations is:

- dependent: veľmi and superior: peknú, priority: 9
- dependent: peknú and superior: knihu, priority: 4
- dependent: knihu and superior: číta, priority: 2

And the list of possible relations is:

- dependent: Lucia and superior: číta, priority:1

Fourth iteration Relation with priority 1 is chosen. This relation is added to the list of final relations and removed from the list of possible relations. Dependent word is removed from the sentence. Only last word remains in the sentence and the list of possible relations is empty, therefore the fourth iteration is the last one and all needed relations are in the list of final relations. The list of final relations is:

- dependent: veľmi and superior: peknú, priority: 9
- dependent: peknú and superior: knihu, priority: 4
- dependent: knihu and superior: číta, priority: 2
- dependent: Lucia and superior: číta, priority:1

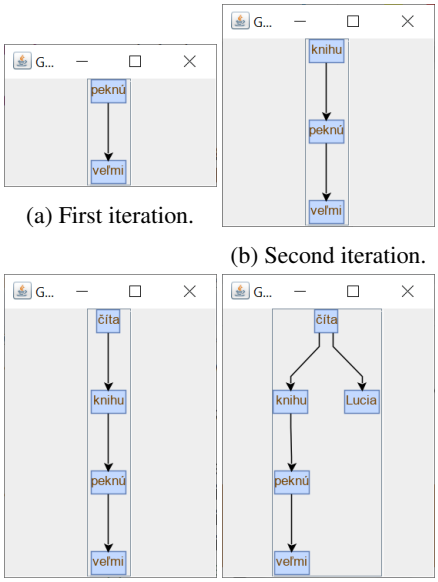
In the last step, a tree structure is built from the list of final relations. Figure 2. illustrates the gradual construction of a tree structure. (a),(b),(c) show a partial tree of a sentence after iterations, and (d) gives an output of a whole sentence after the fourth iteration.

4 Conclusions and further research

In this paper, we have proposed the algorithm for creating a tree structure of the Slovak sentence. We have presented the running of our algorithm on an example of Slovak sentence "Lucia číta veľmi peknú knihu". However, the Slovak language has various orders of words and combinations of sentence members. The algorithm also finds a tree structure of sentences:

- simple sentence, for example "Lenka nakupuje oblečenie."[10]

[8] Lucy is reading book.
[9] Lucy is reading.
[10] Lenka is buying clothes.



(a) First iteration. (b) Second iteration.

(c) Third iteration. (d) Fourth iteration.

Figure 2: Tree structure after each iteration.

- simple sentence without subject, for example "Kráčame do školy."[11]
- simple sentence with same sentence member, for example "Milý a pekný Martin kupuje kvety Lucke."[12]
- compound sentence with two predicates, for example "Deti sa hrajú na ihrisku a rodičia sa rozprávajú."[13]
- compound sentence with two predicates and same sentence member, for example "Pekný a upravený dom stojí na kraji ulice a bývajú v ňom dvaja ľudia."[14]

We aim to improve the presented algorithm in our future research. Particularly, our objective is to analyze the following special types of sentences:

- fundament sentence, for example "Prší." [15]
- simple sentence with a complex predicate, for example "Mal by som už ísť." [16]
- compound sentence with more predicates, for example "Na záhrade máme červené tulipány, ktoré nám darovala stará mama a vedľa tulipánov je záhon ruží." [17]

The testing of the algorithm for the various types of Slovak sentences and the visualization of the outputs in the user friendly environment are the main objectives of our future research.

[11] We are going to school.
[12] Nice and handsome Martin buys flowers for Lucy.
[13] Children are playing on a playground and parents talk.
[14] A beautiful and tidy house stands on the side of the street and two people are living in it.
[15] It is raining.
[16] I should go now.
[17] We have red tulips in the garden that our grandmother gave us, and next to the tulips is a bed of roses.

References

[1] D. Hladek, S. Ondáš, and J. Staš, "Online natural language processing of the slovak language," 11 2014.

[2] D. Hladek, J. Stas, and J. Juhar, "Morphological analysis of the slovak language," *Advances in Electrical and Electronic Engineering*, vol. 13, no. 4, pp. 289–294, 2015.

[3] "Slovenské slovníky." `https://slovnik.juls.savba.sk/?d=pskcs&d=psken&d=locutio&d=ma`. (Accessed on 06/09/2020).

[4] R. Garabík, "Slovenský národný korpus," 2020.

[5] D. Zeman, "Slovak dependency treebank in universal dependencies," *Journal of Linguistics/Jazykovedný casopis*, vol. 68, no. 2, pp. 385–395, 2017.

[6] N. R. Krajči S., "Tvaroslovník – databáza tvarov slov slovenského jazyka," in *zborník príspevkov z pracovného seminára ITAT*, pp. 57–61, 2012.

[7] N. R. Krajči S., "Projekt tvaroslovník – slovník všetkých tvarov všetkých slovenských slov," in *Znalosti 2012*, pp. 109–112, Vydavatelství MFF UK, 2012.

[8] J. Hiľovská, *Syntaktická analýza slovenskej vety pomocou Tvaroslovníka*. PhD thesis, 2017.

[9] R. McDonald, K. Lerman, and F. Pereira, "Multilingual dependency analysis with a two-stage discriminative parser," in *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pp. 216–220, 2006.

[10] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *Proceedings of the 2014 conference on*

empirical methods in natural language processing (EMNLP), pp. 740–750, 2014.

[11] "Natural language processing centre." `https://nlp.fi.muni.cz/en/NLPCentre`. (Accessed on 06/09/2020).

[12] "Tools | Úfal." `https://ufal.mff.cuni.cz/tools`. (Accessed on 06/09/2020).

[13] M. Straka, "Udpipe 2.0 prototype at conll 2018 ud shared task," in *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pp. 197–207, 2018.

[14] "Utkl." `http://utkl.ff.cuni.cz/en/utkl.html`. (Accessed on 06/09/2020).

[15] J. Pavlovič, *Syntax slovenského jazyka I*. 2012.