On the Role of Syntactic Analysis of Natural Languages

Vladislav Kuboň

ÚFAL MFF UK
Malostranské nám. 25, Praha
vk@ufal.mff.cuni.cz

Abstract. The article tries to advocate the fact that although the field of rule-based syntactic analysis of natural languages has recently been practically taken over by data-driven (mostly stochastic) methods. it is still important both for the theory of formal description of natural languages as well as for certain types of applications, especially those dealing with an ill-formed input. The arguments are based upon the experience gained in the process of development a pilot implementation of a grammar checker of Czech. Although the methods described in the paper did not lead directly to a commercial application, they definitely increased the level of understanding of certain complicated linguistic phenomena, namely the phenomenon of grammaticality and non-projectivity of Czech sentences.

Keywords: Syntactic analysis, parsing, natural languages

1 Introduction

The syntactic analysis of natural languages had been in the centre of attention of computational linguists for decades. This is not surprising given the fact that an automatic syntactic analyzer constitutes a key element of a wide range of applications involving in some way the automatic processing of natural languages. The quality of various types of applications directly depends on the quality of modules of syntactic analysis they contain.

Syntactic analysis of natural languages is a very interesting research field. On the one hand, it is relatively close to the problem of parsing formal languages, which has been studied by computer scientists for a very long time and which has therefore been investigated to a great depth (at least for regular and context-free languages). The mathematical background developed for formal languages provides a good ground for mathematically sound description of differences between formal and natural languages. It also allows an implementation of methods which are able to guarantee certain desired properties - probably the best example is the area of grammar checking of natural languages where it is extremely important to use adequate framework allowing for both a sufficient expressive power and an effective implementation.

On the other hand, the syntactic analysis of natural languages does not allow for a direct application of the results of the theory of formal languages and thus provides more than a sufficient challenge for those wishing to develop new methods and approaches both in theory and in implementation.

2 A Shift of the Meaning

The area of syntactic analysis has also undergone a major change in the course of the last two decades. The most important change had been the shift of the mainstream research from the traditional rule-based approaches to data-driven ones, among which the stochastic processing of natural languages plays the most prominent role. This shift was allowed not only by a huge increase of computational power available (even if we consider a milder version of Moore's law [15], we may roughly estimate that the processing power doubled ten times during the period of 20 years since 1990, which means that nowadays we have computers which are 1024 times more powerful than they were back then), but also as a result of very intensive work being performed in the field of corpus linguistics, which provided huge quantities of reliably annotated data for many languages including Czech. These factors have drastically changed not only the field of syntactic analysis of natural languages, but also the whole field of computational linguistics.

At this point it is necessary to mention certain terminological issues. The fact that data driven methods, especially probabilistic and stochastic ones, became a prevailing research direction in computational linguistics of the 21st century also influenced the common understanding of the term "parsing." While more traditional definitions of parsing do not make a distinction between "parsing" and "syntactic analysis," in the contemporary literature the term "parsing" became almost synonymous to the term "stochastic parsing." For example in the book [5] (published for the first time in 1990) the first sentence of the preface says: *"In computer science and linguistics,* ***parsing****, or, more formally,* ***syntactic analysis****, is the process of analyzing a sequence of tokens to determine their grammatical structure with respect to a given (more or less) formal grammar."* This understanding of the term "parsing" is natural, given the fact that until the data driven methods became the mainstream research direction, the analysis of formal and natural languages strongly influenced each other.

On the other hand, the process of stochastic parsing typically does not require any explicit grammar, neither formal nor informal. It tries to determine the mutual relationship of tokens (words, punctuation marks etc.) on the basis of an annotated corpus of a given natural language. The knowledge of the language is not contained in any grammar, it is contained implicitly in the annotation of individual sentences in the corpus.

The fact that high quality manual annotation typically involves much more than syntax of a given language makes it difficult to use the term syntactic analysis as a synonym for stochastic parsing. The syntactically annotated corpora (PennTreebank, PDT, Negra, Tiger, Szeged Treebank) usually contain only one representation (tree) for each sentence, even in case that the sentence is ambiguous. For example, the annotation guidelines of PennTreebank [3] mention specific rules for bracketing the so-called Financialspeak: *"Annotators determine intuitively whether a particular set of* *tokens is Financialspeak."* The guidelines rely on the annotator's understanding of the meaning: *"it's talking about financial stuff, but you don't really know what's going on" or "there's no sensible way to interpret the sentence other than as Financialspeak."*

If we look at the sentences annotated at the analytical level of the Prague Dependency Treebank, we may find several examples where the sentence is ambiguous from the strictly syntactic point of view. Let us take for example the following sentence:

Na poštách v Praze dnes končí omezený prázdninový provoz, fronty u poštovních přepážek, na které si dost našich čtenářů stěžovalo, by se proto měly zmenšit.

[Lit.: On post-offices in Prague today ends limited holiday operation, queues by post counters, on which $\langle Refl. \rangle$ number of our readers complained, $\langle Cond. \rangle \langle Refl. \rangle$ consequently should shrink.]

(*A limited holiday regime ends in the post offices in Prague today, the queues at the post office counters, which were criticized by a number of our readers, should therefore shrink.*)

The analytical tree of this sentence looks as follows:

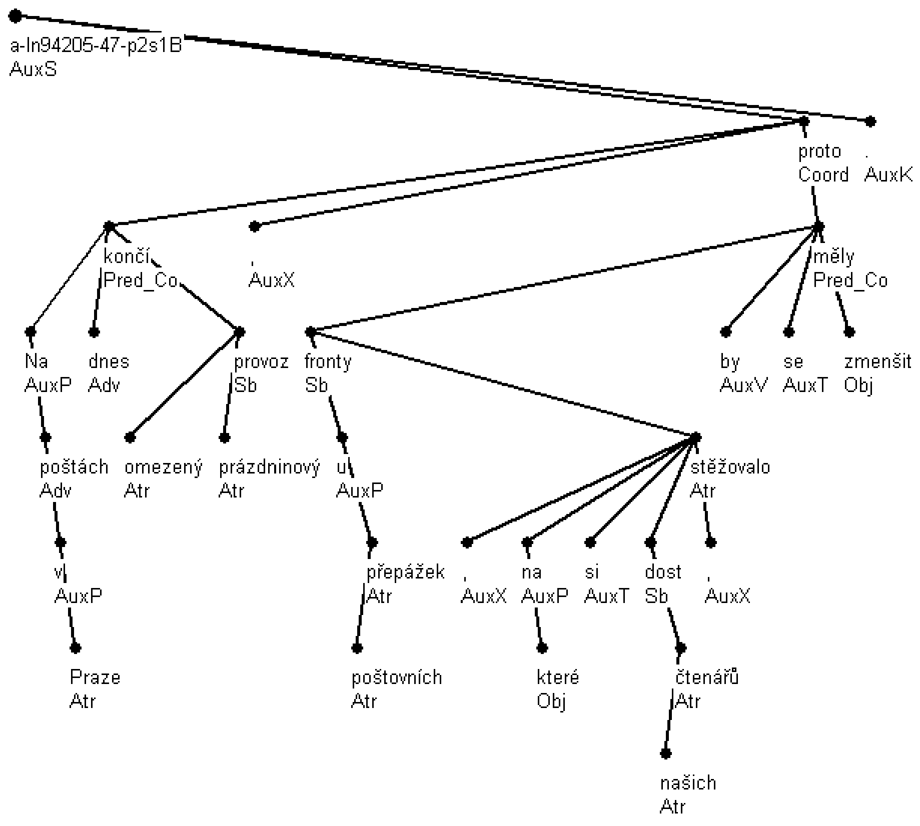

Figure 1: The analytical tree of the sample sentence

There are several syntactic ambiguities in this sentence, they are caused mainly by a typical ambiguity of an attribute expressed by a prepositional group (e.g. "v Praze"). Apart from them, there is also an ambiguity of the attachment of the subordinated clause "*na které si dost našich čtenářů stěžovalo,*" [which were criticized by a number of our readers] which may be attached both to the noun "*fronty*" [queue] and the noun "*přepážek*" [counters]. In order to resolve this ambiguity, the human annotator had to decide which of the two readings describes more adequately a real situation - were people complaining about queues or about the counters (which might not have been working properly)? This decision has nothing to do with syntax, it is based not only on the annotator's knowledge of the grammar, it is very often also based on the context, semantics, pragmatics or real world knowledge. The annotators very often did not even notice the syntactic ambiguity hidden in a sentence because the additional (syntactically plausible) readings did not make any sense to them. All the knowledge about the acceptable and unacceptable readings is implicitly contained in the annotated data. The data-driven parsers naturally exploit it, they can not make any distinction between syntactic or semantic clues for particular dependencies contained in the treebank. For this reason it is more adequate to use the term "syntactic analysis" for such analysis of natural languages which is based exclusively on the syntax of a given language. This is also the meaning we are going to use in the subsequent text.

There is at least one more reason why it is reasonable to distinguish between a syntactic analysis and a data-driven natural language parsing. Both methods aim at a slightly different goal. The syntactic analysis tries to provide a set of all syntactically correct readings of a given input sentence, while the most frequent output of a natural language parser is a single structure representing the most probable reading of the input. Many stochastic parsers are in principle able to provide a ranked list of an arbitrary number of sentence structures. However, this feature is usually not used, especially not in the case when the success of a particular parser is being measured by standard metrics, where only the best candidate structure is being compared with its manually annotated counterpart in the evaluation part of the corpus.

On the other hand, giving all syntactically plausible readings is a natural result of the (grammar-based) syntactic analysis. These results may be useful for further (semantic) analysis which may be able to choose the most suitable result for a given context or situation. If the syntactic analyzer aims at a single structure only, we talk about syntactic recognition, not about analysis. The syntactic structure provided by a recognizer is typically the first (in the course of computation - it is not necessarily the most likely or most probable one) structure covering the complete input sentence. The syntactic recognizer thus only answers the question whether a particular input sentence is a syntactically correct sentence according to a given grammar (and provides one of the possible trees for this sentence), nothing more.

3 The importance of syntactic analysis

The overwhelming success of stochastic methods in natural language processing naturally invokes a very serious question - is it reasonable to stick to the traditional rule-based approach to syntactic analysis in the moment when there are huge amounts of annotated data available for many languages, which may serve for the training of good data-driven parsers? The answer to this question is not simple; let us summarize some arguments in favor of the rule-based syntactic analysis in the following paragraphs.

The close relatedness of syntactic analysis of natural languages to the methods of syntactic analysis of formal languages allows the exploitation of results achieved in this deeply and thoroughly investigated field for the application in natural language analysis. The theorems describing the properties of individual classes of languages in the Chomsky hierarchy quite clearly indicate the required level of tools we need to exploit if we want to be able to develop grammar rules describing a particular (natural) language. For example, the syntactic theory tells us that the non-projective constructions (which are quite typical and also relatively frequent in languages with a higher degree of word-order freedom) cannot be handled by context-free grammars and that if we want to describe the grammar of those languages in its full complexity, we must apply a more complex mechanism which would be able to deal with this linguistic phenomenon. On the other hand, the formal language theory also helps to estimate what will happen if for some reason (simplicity, efficiency etc.) we decide to stick to a certain type of a grammar (regular, context free etc.). It is able to provide reliable information about the possible success of a certain approach to the problem of the analysis, it can guarantee whether the chosen approach is adequate or not. And, it is able to provide this guarantee in advance, unlike the data-driven methods where the adequacy of the methods applied can be judged only ex post, after the evaluation of each experiment.

This distinction is very important also for practical applications - the theory behind the traditional rule-based approach helps to estimate the complexity of each particular application and it is also able to provide a sort of qualified estimation of the achievable quality of the output given the basic parameters for the application (e.g. the timeframe and budget available etc.). This estimation is very difficult for data-driven methods, where the success depends on the type of training data available (the performance of parsers may drop significantly if they are used for parsing texts out of the training data domain), on a type of natural language they are used for (for example, the results of well-known parsers of Collins[2], Charniak[1] and McDonald[14] present significant drop of quality of parsing Czech when we compare it to the results of parsing English (see, e.g., [20]).

The data-driven parsers are also not so well suited for the applications which require a sort of "negative" information – e.g., the information about grammatical errors people might make when they are writing. All existing data-driven parsers work with positive examples only – they try to follow the rules about how a sentence structure should look like for a particular input sentence on the basis of the evidence found in the training corpus. If a particular construction is not found in the training corpus, it does not mean that it is incorrect, due to a data sparseness there are many constructions which do not appear in a particular training corpus and still constitute a correct expression. In order to overcome this problem, the data-driven parsers exploit a method called smoothing – the combinations of word forms which are not encountered in the whole training corpus are assigned a very small non-zero probability which helps to obtain a non-zero probability for the whole input sentence containing some of those word combinations. Although this method helps a lot in the analysis of correct constructions, its main disadvantage is that it actually wipes out the difference between the incorrect constructions and between the correct ones which are not represented in the training data. This makes it virtually impossible to include any reliable information about grammatically incorrect constructions into the language model.

A corpus specialized on grammatically incorrect constructions might constitute a solution of this problem, but although the idea sounds simple, its practical realization would be more difficult. The main problem is the fact that there is practically infinite number of grammatically incorrect variants of a particular grammatically correct sentence. Some of these incorrect variants might even be perfectly understandable for a human reader. We might of course concentrate on collecting the most frequent examples of errors people really do, but then the question will be how to obtain reliable data of this kind given the fact that different people make different errors (native speakers will tend to make a different kind of errors than non-native speakers and the set of most typical errors made by non-native speakers will probably vary according to their native language, etc.). On top of that, different errors can be encountered in different situations (for example, using a text editing software invokes certain typical errors - unwillingly deleted or added words or lines, errors inserted by the software itself through its auto-correction features, etc.). Such a collection of errors will definitely be very subjective and biased.

Last but not least - the development of syntactic analyzer by means of hand-written grammars may help to discover new facts about the syntax of the language, about the role of individual language phenomena; it may help to push forward the syntactic theory. This is something very difficult to achieve if we perform the research only as a series of experiments, by a trial and error method which so frequently accompanies the data-driven research. Let us now present an overview of the development of one specific type of application, the grammar checker for Czech in the following sections. It provides a good illustration of the importance of syntactic analysis for studying the properties of a particular natural language.

4 Grammar Checking of Czech

The second half of 80's and the first half of 90's witnessed a major breakthrough in the computer technology. A new category, personal computers, was created with the aim to provide individual users with their own computers, a truly personal device whose main purpose was to make the professional life of all kinds of office and knowledge workers more efficient. The gradual improvement of the hardware also inspired new kinds of applications which were sought for by the users especially for the work in the office. The typewriters and the clumsy dedicated text editing machines were being gradually replaced by relatively simple software offering more and more useful text editing functions. Apart from the ability to help to create, store, update and print all kinds of documents, the modern text editors also aimed at providing new additional functionalities. The tools for checking the correctness of the typed text were among the first and most important ones.

At that moment it became clear that the actual error checking of written text consists of at least two more or less independent applications – the checking of individual word forms (a spelling checking) and the checking of grammatical correctness of the text (a grammar checking). The first application, spelling checker, was implemented very swiftly - the dictionary of all possible word forms for languages with a limited inflection or a dictionary of basic word forms accompanied by a clever inflectional algorithm for languages with richer inflection constituted the proper answer to the challenge. The development of spelling checkers for such highly inflected languages as Czech followed very quickly after the spelling checker for English became a standard part of any text editing software.

The grammar checking posed a significantly bigger challenge. It is relatively simple for languages with a more strict word order – grammar checkers for such language type are usually based on error patterns, short combinations of words which by any means cannot constitute any part of a grammatically correct sentence of a given language. The strict word order helps to identify those combinations – if the noun immediately preceding a finite verb in an English sentence does not agree with the following verb in number and gender, it is very likely that the subject-verb agreement is violated and the grammar checker may issue an error message.

The free word order makes the identification of grammatical errors more difficult. There are, of course, some errors patterns which may be exploited similarly as in the languages with a strict word order, but the really simply identifiable ones are only a few – for example, in Czech every subordinating conjunction has to be immediately preceded by a comma. There are certain exceptions when the precedence may not be immediate (e.g. certain adverbs may appear between the comma and the conjunction) or when there is no comma at all (a coordination of subordinated clauses, each of which starts with a subordinating conjunction "*že*" , e.g. "*Slíbil, že přestane pít a že se zřekne žen.*" – [He promised to stop drinking and to give up women.]), but in principle such (extended) error pattern might reliably detect an error in punctuation, one of the most frequent grammatical error types in Czech. The same method will probably fail for the subject/verb agreement errors, a second very frequent type of grammatical errors in Czech. The error patterns won't be able to capture the possible ill-formed constructions if they cannot rely on the position of words in the sentence. A more powerful mechanism seemed to be required for agreement errors in languages with a higher degree of word order freedom - using syntactic analysis based on hand crafted rules constituted a viable alternative for them. It is therefore no wonder that one of the first European projects in the field of computational linguistics including Central and East European countries aimed at the problem of grammar checking for Slavic languages, namely Czech and Bulgarian. It started in 1993 and it involved four partners – University of Saarlandes in Saarbruecken, University of Barcelona, University of Sofia and Charles University in Prague. The main goal of the project was to develop a pivot implementation of a grammar-based grammar checker for both participating Slavic languages.

4.1 Initial considerations

The strategy of our approach to the problem has been for the first time described in the paper [11] It exploits the results previously achieved in the automata theory and in the theory of error recovery in syntactic analysis of formal languages. More precisely, it subscribes to the idea of deleting automata described in [8] and it applies certain methods of error recovery developed for formal languages (described e.g. in [18] to a new field - grammar checking of natural languages. Although the paper is relatively short, it contains a couple of ideas which proved to be very important for the success of the whole project.

The first idea is the separation of the grammar rules into two parts - one describing the syntactically correct constructions (the automaton P) and the other describing the incorrect ones (the automaton N). This basic idea - having one grammar with two sets of rules and dealing with the correct and incorrect constructions in a uniform way- was developed further in the subsequent stages of the project. The second idea is the idea of a stepwise simplification of input sentences by removing the ”less important” words (in fact dependent ones) while preserving the correctness or incorrectness of the original input sentence. It had been later developed further in the form of the analysis by reduction, a powerful tool for syntactic analysis of languages with a free word-order. A very detailed description of the latest version of the framework can be found in [13].

4.2 Grammaticality and Non-projectivity

The automata were later [7] replaced by a special type of grammar designed in the frame of the project, the RFODG, Robust Free-Order Dependency Grammar. The grammar (and the parser based on this grammar) proved to be more flexible and effective for the actual implementation of the grammar checker. The idea of the pair of automata P and N had been in fact transformed and enriched by taking into account an additional phenomenon - a non-projectivity. This language phenomenon is very typical for languages with a higher degree of word-order freedom. It is at the same time a very complex phenomenon, which cannot be fully handled by context-free grammars and thus it constitutes a substantial challenge for any syntactic analyzer aiming at a full coverage of a particular natural language. On top of that, our research carried in the project has revealed that non-projectivity also plays a very substantial role in the grammar checking. Let us demonstrate this fact on the following example:

Let us take a Czech sentence *Které děvčata chtěla dostat ovoce?* [Lit.: Which(acc.sg) girls wanted to_get fruit(acc.sg)? – Which fruit wanted the girls to get?]. The dependency structure of this (syntactically correct) sentence is described in Fig. 2.

The problem is that although the sentence is syntactically correct, the reading preferred by most readers is not the one described by the Fig.2. The more obvious reading contains an error in the ending of the pronoun *které* [which] – the proper form která would agree with the governing noun *děvčata* in gender, number and case. In other words, without having a clear clue which noun should the pronoun depend upon, we can hardly distinguish between a non-projective and syntactically incorrect reading of this sentence. This observation actually influenced the design of the experimental grammar checker for Czech – non-projective and syntactically incorrect constructions were handled in a uniform way. The uniform handling of those constructions was achieved by a set of parameters applied to a grammar – the same (meta)rule of the grammar can be interpreted in a different way according to a global (applied to the whole grammar at once) or a local (applied to a particular grammar rule)

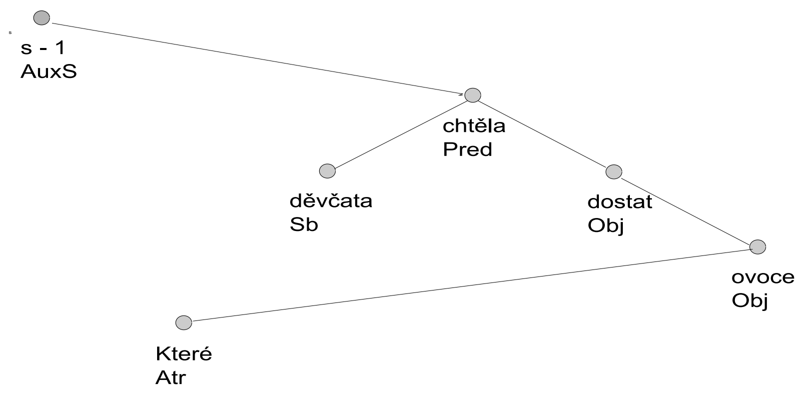

Figure 2: An analytical tree of the sentence *Které děvčata chtěla dostat ovoce?*

parameter. The design described in this paper was cited later in several papers and journal articles, for example in [17], [9], [4] etc. Although the results of the project were positive, none of the implementations developed in the project (the remaining two were described in [16] and [10] reached the stage of a working prototype. It took several years before the experience gained in the Lateslav project helped to implement a real grammar checker for Czech, which became a standard part of the Microsoft Office package. The theoretical research inspired by the project had more direct continuation. It lead to a series of articles on topics concerning especially two interesting topics - non-projectivity and robustnes.

4.3 Theoretical Results Concerning Non-projectivity

The series of papers on the non-projectivity and its influence at the complexity of syntactic analysis had been started by the paper [6]. The research described in this paper was in fact stimulated by the discovery that a single Czech clause may contain more than one non-projective construction. Our initial assumption, published for example in [7], was different - only a single non-projective construction per Czech clause. This assumption was even reflected in some versions of the parser implemented by T.Holan - whenever a number of non-projective constructions in a particular syntactic tree exceeded one, the derivation of the tree stopped and the analysis continued in a different way (if possible). Our working hypothesis supposed that clauses with more than a single non-projectivity are difficult to understand even for a native speaker. This hypothesis may be valid for languages with stricter word order, but it is definitely not true for Czech. The decisive argument against the original hypothesis was provided by the sentence *Tuto knihu jsem se mu rozhodl dát k narozeninám.* [Lit.: This book I myself him decided to_give to birthday. - I have decided to give him this book to his birthday.] Fig. 3 shows the syntactic tree of this sentence.

The nodes labeled by the word forms *knihu* and *mu* in Fig.3 mark two non-projective constructions present in the sentence. The sentence is not only syntactically correct, it is also natural and human readers have no difficulties to understand it. This sentence gave us a hint that the number of non-projective constructions in a simple

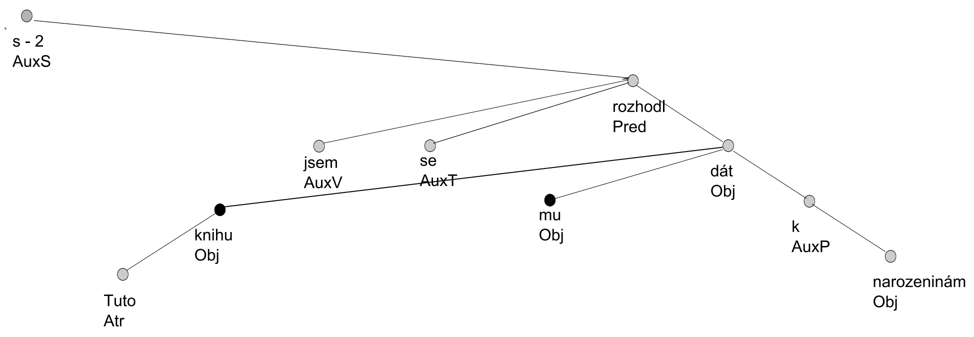

Figure 3: An analytical tree of the sentence *Tuto knihu jsem se mu rozhodl dát k narozeninám.*

Czech sentence (clause) not only exceeds one, but it may theoretically not have any upper limit. Of course, the sentences having more than four non-projectivities usually are slightly more difficult to understand, but it would be very difficult to claim that that they are not grammatical.

The sentence from Fig.3 also represents a very productive pattern for Czech non-projective constructions – a main verb with a dependent verb in infinitive, both richly modified, with interlocking modifiers (actants, free modifiers). This is definitely not the only type of non-projectivity in Czech (cf. e.g. [19]), but it is probably the most productive one with regard to the number of non-projective constructions present in a single Czech clause. Apart from this basic statement the article [6] also introduces very natural measures of non-projectivity, namely the magnitude of non-projectivity of a sentence and of a language. The fact that these measures are very closely and naturally related to the complexity of syntactic analysis of non-projective sentences is also one of the very important results presented in this article.

4.4 Grammar Checking and Robust Syntactic Analysis

One of the topics which arised during the research on grammar-based grammar checking concerned the difference between grammar checking and robust syntactic analysis. Both activities have a number of common features, therefore they used to be considered by many linguists as almost identical. This opinion does not stand the test of a real application - when developing a grammar checker, the author has to solve a number of issues which are specific for this task. These issues are closely related to the fact that the target audience of a grammar checker is an ordinary user of a text editor. Such user has a very limited understanding of what is going on. The grammar checker thus not only must be able to process the ill-formed input, it must also be able to identify the errors, describe them to the user and to offer a correction which has to be in accordance with syntactic rules of the language. It is also quite natural that the system has to concentrate on precision rather than recall, mainly because nothing can annoy a typical user more than a false error message. The errors which are not discovered in most cases won't be noticed by the user either, so they are not so much important. When the system reports an error, it must really be an error;

otherwise the user gets an impression that the grammar checker does not perform its task properly.

The robust syntactic analyzer, on the other hand, has to concentrate on delivering a reasonable syntactic representation of every sentence, even the ill-formed one. The ability to construct a syntactic tree is much more important than the ability to identify and correct the grammatical errors. Robustness may not only mean the endeavor to create the syntactic tree as similar to the syntactic tree of a corresponding correct sentence, it may as well be implemented in a slightly primitive way, without the proper regard for the syntax, simply constructing artificial edges according to some heuristics with the aim to provide at least some complete syntactic tree for a particular ill-formed input sentence. If the robust parser is used for example as a part of a rule-based machine translation system, then the simple robustness of the source language analyzer may be more important than correctness or adequacy. Badly (but completely) analyzed sentence may turn out to be better than a sentence analyzed correctly, but incompletely. This observation is in fact an additional argument for the claim that grammar checking and robust syntactic analysis are different - a robust syntactic analyzer must definitely prefer recall over precision.

These considerations led to the introduction of a notion of an *accurate robustness*, which has been presented in [12]. It takes the considerations outlined in the previous paragraphs even further. It argues that it is important to have the robustness under control, that unlimited robust analysis providing syntactic structures even for sentences which are so much ill-formed that not even the humans are able to understand them, is in fact counter-productive. The paper introduces a way how to limit the robustness by means of two measures, a *node-gap complexity* and a *degree of robustness*. Both measures in fact reflect our earlier observation that both ill-formed and non-projective constructions have certain effect on syntactic analysis, that both types of constructions make syntactic analysis substantially more difficult and computationally more expensive. The node-gap complexity expressed the complexity of a sentence with regard to the degree of its non-projectivity. It is not a simple sum of all non-projective constructions in a sentence, the measure is able to express the fact that independent non-projectivities do not raise the complexity of the analysis in the same way as non-projectivities which are governed by a single node of a dependency tree. Due to this property the measure corresponds naturally to the complexity of the syntactic analysis of non-projective constructions.

The second measure, the degree of robustness, is more simple. It is in fact a sum of violations of syntactic rules describing syntactically correct constructions of a particular language. These violations are in our terminology called *syntactic inconsistencies*. This term was created due to the fact that a syntactic error may very often manifest itself on a different place in a sentence than a human reader would locate it. This behavior is partially caused by the direction of the analysis – the analysis by reduction is performed in a sort of "bottom-up manner," it builds the tree from the dependent to the governing nodes (the dependent words are reduced prior to the governing ones) and thus the error may be propagated upwards in the tree. This happens especially with agreement errors when a morphological ambiguity of word forms from the input sentence may provide agreements which are locally correct but globally incorrect.

The paper is interesting from one more point of view. Apart from the two measures which may be used for the parameterization of a robust syntactic analyzer so that it outputs only those results which are "reasonably robust," it also introduces two very important data structures – D-trees and DR-trees. The D-trees are in fact dependency trees, the DR-trees (Delete-Rewrite) represent a special kind of constituent trees. There are very good reasons for using both kinds of trees. The *D-trees* represent a very natural way of transparent representation of syntactic relations including the ill-formed or non-projective constructions. The *DR-trees* represent straightforwardly the history of analysis for each particular reading of an input sentence. Both data types have very interesting properties from the theoretical point of view. Each DR-tree can be easily transformed into a corresponding D-tree by a simple operation of contraction of vertical edges (paths) of the DR-tree and by the addition of the information about the length of the contracted paths. The degree of robustness is the same for both kinds of trees, although the number of individual non-projectivities for particular nodes may differ.

Using both kinds of trees, each of them for particular tasks they are best suited for, in fact gives an answer to a very old and persistent question in the field of computational linguistics, namely the question which data type is best suited for the representation of natural language syntax, the dependency trees or the constituent ones? From our point of view, the question does not make much sense, both data types are best for a particular (and slightly different) task. Using both gives us more powerful way of expressing the findings concerning the properties of natural languages.

5 Conclusion

The overview of the research in the field of rule-based syntactic analysis of Czech tries to advocate the fact that although this field has recently witnessed a strong domination of data-driven (mostly stochastic) methods. it is still important both for the theory of formal description of natural languages as well as for certain types of applications, especially those dealing with an ill-formed input. The arguments presented in the paper are based upon the experience gained in the process of development of a pilot implementation of a grammar checker of Czech. Although the methods described in the paper did not lead directly to a commercial application, they definitely increased the level of understanding of certain complicated linguistic phenomena, namely the phenomenon of grammaticality and non-projectivity of Czech sentences.

Acknowledgments

This paper was supported by the Grant Agency of the Czech Republic, grant No. 405/08/0681.

Bibliography

[1] Eugene Charniak. A Maximum-Entropy-Inspired Parser. In *Proceedings of NAACL*, Seattle, USA, 2000.

[2] Michael Collins, Jan Hajič, Lance Ramshaw, and Christopher Tillmann. A Statistical Parser for Czech. In *Proceedings of the ACL'99*, Maryland, USA, 1999.

[3] Bies et al. *Bracketing Guidelines for Treebank II Style Penn Treebank Project.* University of Pensylvania, Pensylvania, 1995.

[4] A.Vandeventer Faltin. *Syntactic Error Diagnosis in the context of Computer Assisted Language Learning.* PhD thesis, Universite de Geneve, 2003.

[5] Dick Grune and Ceriel Jacobs. *Parsing Techniques - A Practical Guide.* Ellis Horwood Limited, Chichester, England, 1994.

[6] Tomáš Holan, Vladislav Kuboň, Karel Oliva, and Martin Plátek. Two Useful Measures of Word Order Complexity. In *Proceedings of the Workshop Processing of Dependency-Based Grammars, COLING-ACL'98*, Montreal, Canada, 1998.

[7] Tomáš Holan, Vladislav Kuboň, and Martin Plátek. A Prototype of a Grammar Checker for Czech. In *Proceedings of the 5th ANLP'97*, Washington D.C., USA, 1997.

[8] Petr Jančar, František Mráz, Martin Plátek, Martin Procházka and Joerg Vogel. Deleting automata with a restart operation. In *Proceedings of DLT'97*, Thessaloniki, Greece, 1997.

[9] K.F.Shaalan. Arabic GramCheck: a grammar checker for Arabic. *Software: Practice and Experience, Volume 35, Issue 7*, 2004.

[10] Zdeněk Kirschner. CZECKER - a Maquette Grammar-Checker for Czech. *The Prague Bulletin of Mathematical Linguistics 62*, 1994.

[11] Vladislav Kuboň and Martin Plátek. A Grammar Based Approach to Grammar Checking of Free Word Order Languages. In *Proceedings of COLING 94*, Kyoto, Japan, 1994.

[12] Vladislav Kuboň and Martin Plátek. A Method of Accurate Robust Parsing of Czech. In *Proceedings of TSD 01*, Železná Ruda, Czach Republic, 2001.

[13] Markéta Lopatková, Martin Plátek, and Petr Sgall. Towards a Formal Model for Functional Generative Description: Analysis by Reduction and Restarting Automata. *The Prague Bulletin of Mathematical Linguistics 87*, 2007.

[14] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of the Human Language Technology / Empirical Methods in Natural Language Processing conference (HLT-EMNLP)*, Vancouver, Canada, 2005.

[15] Roger E. Moore. Cramming more components onto integrated circuits. *Electronics, Volume 38, Number 8*, 1965.

[16] Karel Oliva. Techniques for Accelerating a Grammar Checker. In *Proceedings of the 5th ANLP'97*, Washington D.C., USA, 1997.

[17] Patrizia Paggio. Spelling And Grammar Correction For Danish In SCARRIE. In *Proceedings of NAACL/ANLP*, Seattle, USA, 2000.

[18] Martin Plátek. Zotavení ze syntaktických chyb se zárukami. In *Sborník konference SOFSEM'89*, Ždiar, Czechoslovakia, 1989.

[19] Ludmila Uhlířová. On the Non-projective Constructions in Czech. *The Prague Studies in Mathematical Linguistics 3, Praha, Academia*, 1972.

[20] Daniel Zeman. *Parsing with a Statistical Dependency Model.* PhD thesis, Charles University, Prague, Czech Republic, 2004.