Towards a Treebank of Abkhaz
The AbNC, Analysing Abkhaz, and the Importance of Good Tools

Paul Meurer, Bergen

1. Introduction

In this article, I present my efforts to build an Abkhaz treebank within the Universal Dependencies (UD) framework, based on texts from the Abkhaz National Corpus (AbNC) and a morphological analyzer developed in the AbNC project.

A treebank is a digital corpus of syntactically annotated sentences, where the syntactic annotations are built according to a syntactic theory or framework, or practical annotation guidelines. Depending on the formalism, the syntactic annotations can be formally characterised as tree structures, or as more general directed graphs. Among the more widely used syntactical grammar frameworks for building treebanks we find:

- Phrase structure and other constituency grammars. Phrase structure trees are ordered trees (possibly with crossing edges and discontinuous nodes), in which the inner nodes are labelled with syntactic categories and span over syntactic constituents, and the leaf nodes consist of the words of the sentence. In some theories, empty surface nodes are allowed.
- Dependency grammars. The associated structures are unordered trees (possibly with secondary edges); the tree nodes consist of the words of the sentence, and edges encode syntactic functional dependencies between words.
- Other grammar formalisms, among them Lexical Functional Grammar and HPSG, whose structures can be characterised as a combination of phrase structure trees and dependency structures.

Whereas phrase structure grammars (including LFG) often are inspired by Chomsky’s Xbar theory or are otherwise quite theory-laden, dependency grammars, building on Tesnière’s pioneering work,[1] are more intuitive and therefore more suitable for a general linguistic audience. Dependency analyses are well-suited for free-order languages, and Abkhaz in particular, where the subject-predicate dichotomy central to phrase structure frameworks is problematic,[2] and subject and direct, indirect, local, and relational objects are linked to the verb in similar ways.

Universal dependencies[3] (UD) has in recent years become the most well-known dependency grammar framework. It has gained wide acceptance in the computational linguistics community and is being used to build treebanks for a large number of languages. Currently, there are UD treebanks of widely varying sizes available for 141 languages. Strikingly, Abkhaz is not among them. This project is intended to fill that gap.

According to the UD web site, “Universal Dependencies (UD) is a framework for consistent annotation of grammar (parts of speech, morphological features, and syntactic dependencies) across different human languages.” An important aim of the UD project was thus to devise a

[1] Tesnière 1959.
[2] One could even argue that the notion of subject is not fully well-defined in Abkhaz.
[3] https://universaldependencies.org.

formalism that would be applicable to any language, and that would make it easy to compare analyses for different languages.

The UD annotation guidelines lay out in detail the adopted principles concerning tokenisation and word segmentation, morphological annotation, and coding of specific syntactic constructions. They also document the tags to be used for parts of speech, morphological features, and syntactic relations, and explain how language-specific features that are not explicitly addressed in the guidelines should be coded.

The requirement of cross-linguistic comparability results in the principle of lexicocentrism: lexical/functional heads should always be lexical words, never function words; copula verbs are dependents of the predicate noun or adjective, and auxiliary verbs are dependents of the main verb. To avoid having a conjunction as the head word of a conjoined phrase and to handle asyndetic coordination, coordination is coded asymmetrically: the first conjunct is formally the head of all the remaining conjuncts.

I find some of these principles somewhat unfortunate. For Abkhaz, it would feel more natural to treat asyndetic coordination as right-headed, in particular in the case of chains of verbs in the past indefinite linked to an aorist to the right. Here, the final aorist can naturally be viewed as the head of the conjoined phrases, but the UD guidelines do not allow this, and it is even not possible to submit analyses to the UD project repository that violate these principles. In cleft constructions, I would prefer to treat the copula verb as the head of the phrase, which is also not allowed by the guidelines. The solution here is not to formally register the verb as a copula verb but treat it as a regular verb.

To illustrate a UD dependency analysis, let us consider the following sentence:

Бнарак	аӄны	амҩахәаста	ианыланы	инеиуан	Абгахәычы.
bnará-ḳ	a-q̇nə́	a-my°a.x°ásta	i-a-nə́.la-nə	i-néi-ua.n	a-bga.x°əç̣ə́
forest-a	it-to	the-way.bend	it-it-enter-Gnd	it-go-Impf	the-wolf.small

‘Walking along a bend in the path, the Fox was going to a forest.’

We can identify the following relations in this sentence:

- The main verb *i-néi-ua.n* ‘was going’ is the head of the sentence (it will be attached to an auxiliary ROOT node via the *root* relation);
- *a-bga.x°ə́ç̣ə* ‘the fox’ is the subject of the main verb (*nsubj* relation);
- *a-q̇nə́* ‘to it’ is a case marker dependent of *bnará-ḳ* ‘a forest’ (*case* relation);
- *bnará-ḳ* ‘a forest’ is an adverbial modifier of the main verb (together with its case marker), which is treated in UD as an oblique argument of the verb (*obl* relation);
- *a-my°a.x°ásta* ‘the way-bend’ is a local object dependent of *i-**a**-nə́.la-nə* ‘entering’ (*obj:lo* relation);
- the converb (absolutive) *i-a-nə́.la-nə* ‘entering’ attaches to the main verb as an adverbial clause (*advcl:conv* relation);
- the full stop is attached to the sentence head via the *punct* relation.

Together, these relations give rise to a complete (universal) dependency analysis. The dependency tree can be displayed in two different ways. If we wish to represent the linear order in the display, we can arrange the words in the order in which they occur in the sentence at the bottom and draw labelled edges from heads to dependents. This display is called ‘linear display’ (cf. Fig. 1).

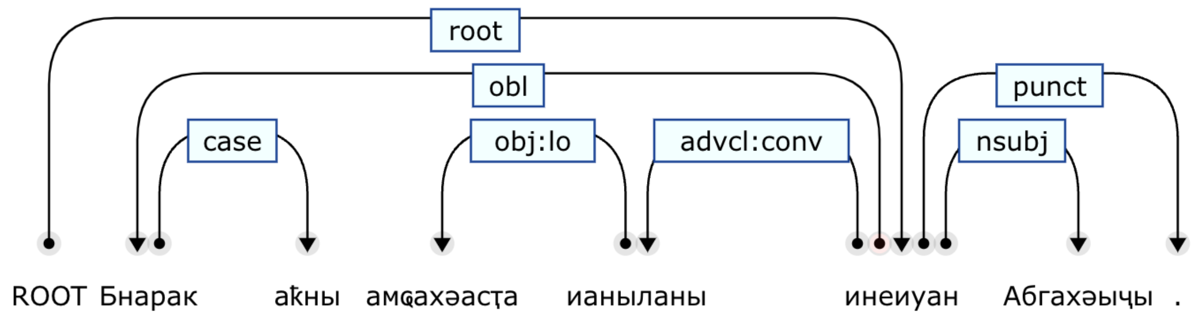


Fig. 1: Linear Display

Alternatively, we can draw a more traditional (top-down) tree which visualises the hierarchical structure of the analysis more directly, at the expense of the linear order, which cannot be deduced from the tree alone (cf. Fig. 2).

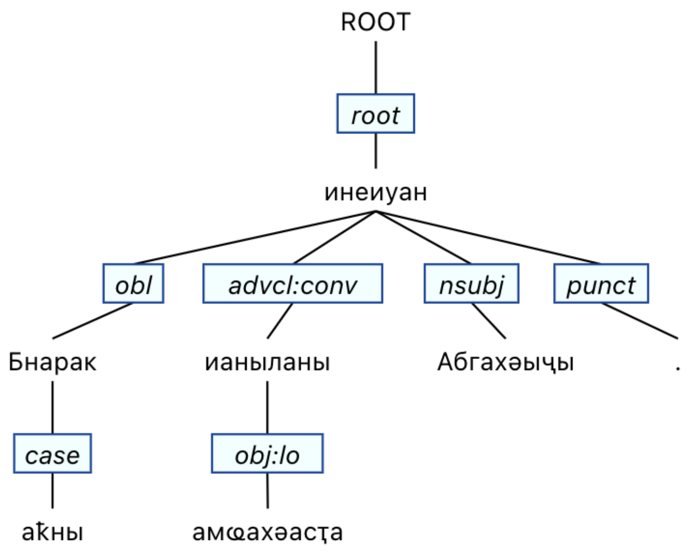


Fig. 2: Tree structure

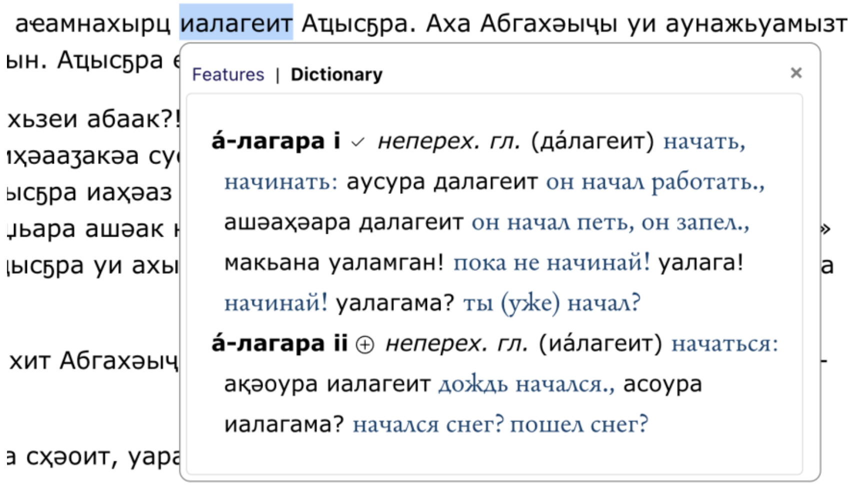
2. The Abkhaz National Corpus

The treebank I am developing is based on the Abkhaz National Corpus (AbNC). The sentences to be analyzed are taken from the texts in the corpus, and the treebanking tools are tightly integrated into the AbNC software.

The AbNC was developed in the years 2016–2018 in a project financed by USAID, with participants from Sukhumi, Tbilisi, Frankfurt, and Bergen.[4] It comprises more than 10 million tokens of texts from a variety of genres and is morphologically annotated. The corpus is hosted in the Corpuscle corpus management tool, which has advanced possibilities for searching and viewing the corpus texts.[5]

[4] Meurer 2018.
[5] https://clarino.uib.no/abnc

In addition to being a corpus-linguistic resource and tool, the corpus also serves as a digital library and as a pedagogical tool for language learning. The texts can be read in a neatly typeset, paginated presentation, and pages can be bookmarked for later reference. Most importantly, when the user clicks on a word in the text, grammatical information about that word is displayed, and in addition, the corresponding articles in the integrated Abkhaz-Russian Dictionary[6] are shown (see the screenshot in Fig. 3 below). For dictionary lookup to work, it is crucial that the texts are lemmatized and annotated by part of speech.


аеамнахырц иалагеит Аҵысҕра. Ахa Абгахәыҵы уи аунажьуамызт

Features | Dictionary

á-лагара i ✓ *неперех. гл.* (дáлагеит) начать, начинать: аусура далагеит он начал работать., ашәаҳәара далагеит он начал петь, он запел., макьана уаламган! пока не начинай! уалага! начинай! уалагама? ты (уже) начал?

á-лагара ii ⊕ *неперех. гл.* (иáлагеит) начаться: ақәоура иалагеит дождь начался., асоура иалагама? начался снег? пошел снег?

Fig. 3: Dictionary lookup of a word in the text

The corpus is a resource in the Norwegian CLARINO infrastructure, the texts and the annotations are freely available under a Creative Common license.

3. Morphosyntactic analysis of Abkhaz

Before a syntactic analysis of a sentence can be built, it has to be tokenised, i.e., split up into words and punctuation, and morphologically analysed. Tokenisation is a comparatively straight-forward procedure, which I will not go into here. Morphosyntactic analysis is the task of assigning to each word in a sentence its lemma or base form and its morphosyntactic features. It consists of looking up all readings of a word form in the lexicon underlying the analyser (lexicon lookup), and then choosing the correct reading according to the context (disambiguation).

What counts as the base form of a morphological paradigm in the context of morphological analysis is largely based on conventions and lexicographic traditions. For Abkhaz, I choose the citation form used in dictionaries (including the word accent), which is the masdar or verbal noun for inflected verb forms and the definite singular form for nouns and adjectives. Here, the classification of predicates derived from adjectives or nouns can be problematic. As a rule, the adjective or noun itself is taken as the base form of such a derived predicate, with *Noun* or *V* as part of speech, together with an additional morphological marker *Pred*. Other such predicates are lexicalised and found in dictionaries; those are treated as (static) verbs or are even analysed ambiguously. A uniform solution has still to be found.

The morphosyntactic features assigned to a reading of a word consist of a part of speech (POS) tag and an unordered set or bag of morphological and morphosyntactic features, including definiteness for nouns, and transitivity, finiteness, negation, person, gender, and relational

[6] Kaslandzia, 2005.

markers and much more for verbs. Enclitics are not treated as independent words; they are coded as additional morphological features. Altogether, there are more than 300 different features.

The lexicon lookup module of the analyser is implemented as a finite state transducer.[7]

Let us consider a typical Abkhaz word form:

> исзеилмыргацәқьакәа
> i-s-z-ei.l-mə-r-ga-ç°q̇’a-ḳ°a
> it-I-for-PV-Neg-Caus-ROOT-Really-Conv
> ‘me not at all being able to understand it’

The analyzer output for this form will be:

> а́и·лы·р:га-ра (*ái-lə·r:ga-ra*)
> V Dyn Intr Caus NonFin Conv Aor Neg S:3 PO:1Sg LO:Rec Reln:Pot Encl:Really

However, the given analysis is only one of 34 readings that are produced by the analyser. This points to one of the main challenges in the development of the analyser, namely dealing with the extraordinarily high degree of homonymy of many word forms, which is due to the polysynthetic nature of the Abkhaz verb and the absence of stress in written text. Little attention is paid to the problem of ambiguity in linguistic work on Abkhaz. Word forms are usually presented with one reading, the one that fits the context. But in parsing, it is obviously a major problem.

In the Abkhaz analyser, disambiguation is implemented in the Constraint Grammar (CG) formalism.[8] Constraint Grammar rules, which take syntactic context into account, can be used to a certain degree to disambiguate a given form, but often, semantic information is needed to fully disambiguate a word. Semantic information can be (and is being) partially integrated into the CG parser, but a principled approach would have to be based on a semantic word net, or on statistical information (e.g., word vectors) derived from a gold standard corpus. Such an approach will be pursued in future work.

When running the CG parser on a sentence, the rules are in essence considered one by one, and each rule is tried on each token in a sentence. This process is repeated until no rule has an effect anymore. Disambiguation rules are for the most part either SELECT or REMOVE rules: A SELECT rule selects a reading of a word if the rule’s matching and context conditions are met, and a REMOVE rule discards a matching reading in the same way. The last surviving reading of a word will never be discarded, this ensures that there is always at least one reading for every word in the output; in this sense, CG is a robust formalism.

The following examples illustrate how typical disambiguation rules are implemented.

Example: Select the verb а·у-ра́ *a·u-rá* (‘fall’, of rain, snow etc.), thereby discarding readings like а́у-ра *áu-ra* ‘receive’ and а-у-ра́ *a-u-rá* ‘work’, if preceded by а-қәа́ *a-k°á* ‘rain’ etc.

First, we define a list of words designing precipitation.

```
LIST Precipitation = "а-қәа́" "а-сы́" "а-кырцх" "а-қәаршаҩы́" ;
```

[7] Meurer 2011.
[8] https://en.wikipedia.org/wiki/Constraint_grammar

In the following SELECT rule, the reading а·у-ра́ *a·u-rá* is chosen if it is preceded by a word of precipitation.

```
SELECT ("а·у-ра́" Dyn) IF (-1 Precipitation) ;
```

Example: The verb а-ды́р-ра *a-də́r-ra* 'know' should contain the potential marker when negated (e.g., и-з-дыр-уа-м *i-z-dər-ua-m* 'I don't know').

The corresponding rule removes readings of 'а-ды́р-ра' *a-də́r-ra* containing the Neg feature, but missing the Reln:Pot feature.

```
REMOVE ("а-ды́р-ра") + Neg - Reln:Pot ;
```

These examples are rather basic, the last rule does not even need sentence context. Context conditions can however be quite complex and consider readings of words arbitrarily far to the left or right; context conditions can be linked, and they can require that all readings meet a given condition, or only one of them, and so on.

3. Building a treebank

Dependency treebanks (and treebanks in general) can be built in a variety of ways: they can be manually constructed, they can be built using either a statistical or a rule-based parser, or by a combination of those methods.

Manual construction is a rather error-prone and time-consuming process and is rarely carried out nowadays. It is primarily useful for lesser-resourced languages, where no automatic tools are available, and for small treebanks.

When constructing a treebank semi-automatically or automatically, rule-based or statistical parsing techniques can be used, or a combination of those. In statistical parsing, a parser has to be trained on a large gold standard, whereas in rule-based parsing, syntax rules have to be written manually by a linguist. Both approaches have their advantages and disadvantages: a gold standard for training might not be available or be too small, such that it first has to be constructed using other methods; writing grammar rules, on the other hand, can be tedious and time-consuming and requires a good knowledge of the syntax of the language as well as the intricacies of the formalism used, which is essentially a programming language.

4. Syntactic analysis

Since there is no training corpus available for Abkhaz, I am using the rule-based approach in the Abkhaz treebank project, combined with manual correction. The rules that build dependency relations are also written in the CG formalism and constitute an add-on to the disambiguation module.

Again, when running the parser, the syntax rules are consulted one by one, in grammar order; they look at one word at a time and try to find a suitable head to attach the word to.

As an example, consider the rule below. It matches a singular non-human noun (which is not a predicate) (line 1) and tries to find a verb with a 3rd singular non-human local object marker, *LO:3SgNH*, anywhere to the right (1*), but not going beyond a comma (line 2). If such a verb is found, a dependency relation is established between the noun and the verb (as head) (line 3), and the label *obj:lo* is attached.

```
WITH NOPARENT Noun + Sg + NH - Pred
   IF (1* V + LO:3SgNH BARRIER Comma) {
      SETPARENT (*) TO (jC1 (*)) ;
      MAP >obj:lo (*) ;
};
```

It is apparent that this rule, when applied to the morphosyntactically annotated and disambiguated introductory example sentence, will match the noun and establish an *obj:lo* dependency to the adjacent verb, as in the fragment shown here:

```
...
амҩахәасҭа    а-мҩахәа́сҭа   Noun NH Sg Det
ианыланы      а-ны́·ла-ра     V Dyn Intr NonFin Abs Aor S:3 LO:3SgNH
...
```

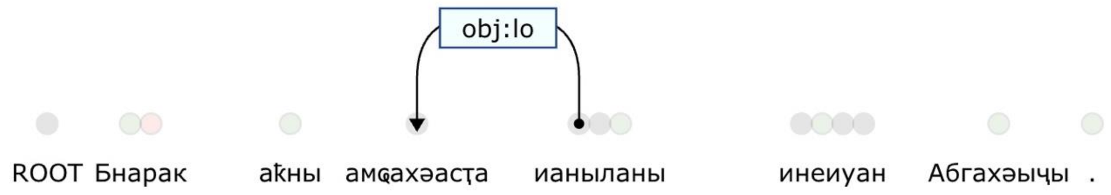


Fig. 4: Obj.lo dependency

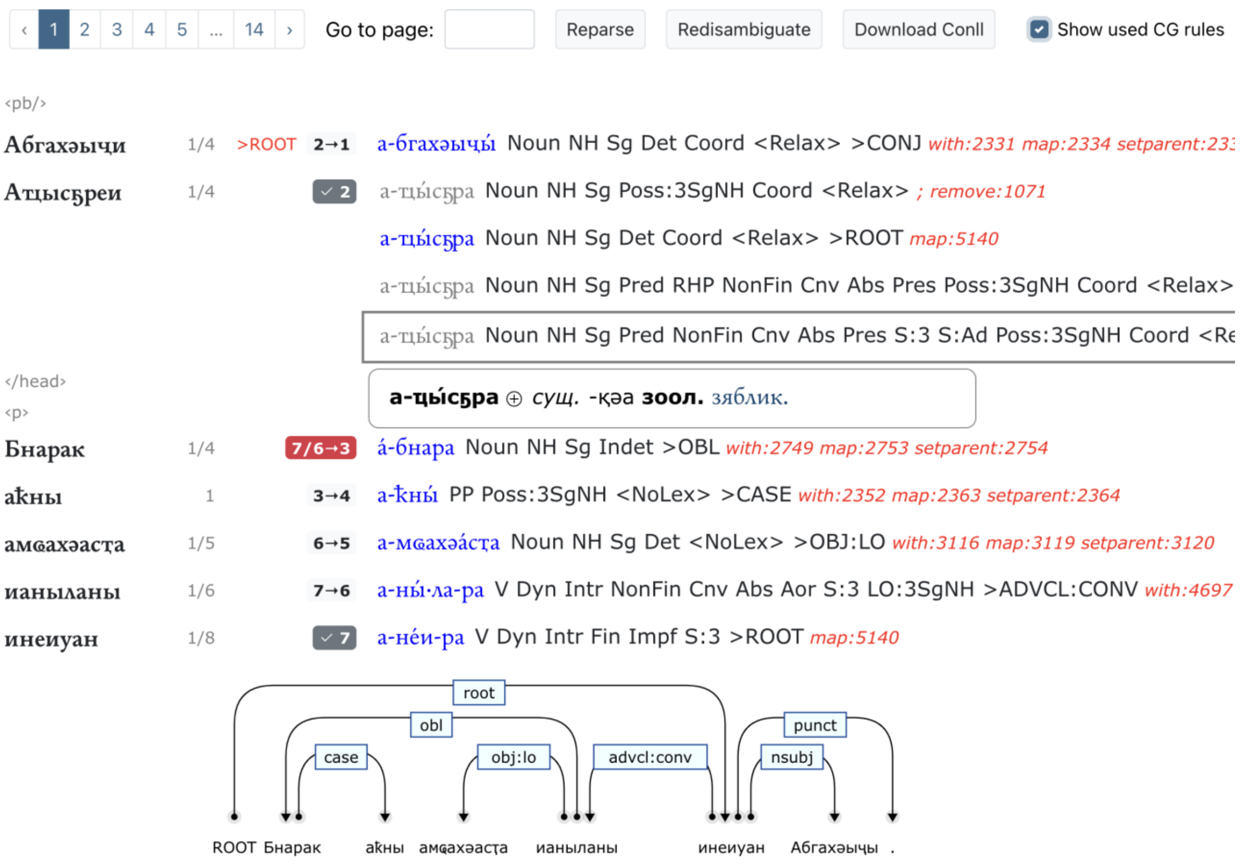
The rules rely heavily on the morphosyntactic features from the lexicon lookup. To obtain best results, it is therefore crucial that the input to the syntax rules be fully (and correctly) disambiguated. This requires manual control and correction of the disambiguation results. After parsing, the resulting dependency analyses must often again be manually corrected, or rules must be refined or added to handle constructions not yet covered by the grammar.

This workflow of disambiguating and correcting the morphosyntactic analyses, inspecting and correcting dependency trees, and correcting and amending the grammar requires a set of powerful, intuitive, and user-friendly tools that help streamlining these tasks. I have integrated several such tools into the AbNC corpus tool, including the following:

- On-the-fly parsing of short text passages
- Automatic morphosyntactic annotation and (partial) disambiguation using the CG rules after the grammar has been changed
- Dictionary lookup of lemma readings
- Manual disambiguation and correction of an analyzed corpus text page
- Automatic tree construction using the CG rules
- Manual correction of dependencies and labels.

The screenshot in Fig. 5 below illustrates some of these features. The sentence tokens are arranged vertically, with the readings to their right. The second line shows all readings, although only one reading, the second, which is not grayed out, has been chosen by the grammar. (The user may choose the display of all readings of the token in focus that are found during lexicon lookup, but normally the discarded readings are hidden.) The CG rules used in disambiguating the sentence are shown in red after the corresponding readings, as rule type and line number. For the last reading of the second token (which is in focus, as the gray box outline indicates), the corresponding lexicon entry is shown, and for the next sentence the analysis is displayed. Here, the user can edit the dependency structure by dragging the dot of an edge and dropping it onto a different word, or by clicking on a label and choosing a different label from a drop-down list. When a dependency is edited manually, the differences to the parsed structure are shown in red, either as a changed label (>ROOT in the screenshot), or as a changed attachment (7/6 → 3 below, which means that the head of token 3 has been changed from token 6 to token 7).[9]

All navigation and editing can be done using keyboard shortcuts alone. Using the arrow keys, the user can move from token to token, or to a specific reading, or to the next page. Pressing ENTER selects the reading in focus, and other keys operate on the token or reading in focus. Key shortcuts can be used to show and hide the lexicon entry or the dependency analysis. Only when editing dependency attachments, the mouse must be used.



```
<pb/>
Абгахəычи    1/4  >ROOT 2→1  а-бгахəычы́ Noun NH Sg Det Coord <Relax> >CONJ with:2331 map:2334 setparent:233
Атцысҕреи    1/4       ✓ 2   а-тцы́сҕра Noun NH Sg Poss:3SgNH Coord <Relax> ; remove:1071
                             а-тцы́сҕра Noun NH Sg Det Coord <Relax> >ROOT map:5140
                             а-тцы́сҕра Noun NH Sg Pred RHP NonFin Cnv Abs Pres Poss:3SgNH Coord <Relax>
                             а-тцы́сҕра Noun NH Sg Pred NonFin Cnv Abs Pres S:3 S:Ad Poss:3SgNH Coord <Re
</head>                      а-цы́сҕра ⊕ сущ. -қəа зоол. зяблик.
<p>
Бнарак       1/4    7/6→3    а́-бнара Noun NH Sg Indet >OBL with:2749 map:2753 setparent:2754
аҟны         1       3→4     а-ҟны́ PP Poss:3SgNH <NoLex> >CASE with:2352 map:2363 setparent:2364
амҩахəаста   1/5     6→5     а-мҩахəа́ста Noun NH Sg Det <NoLex> >OBJ:LO with:3116 map:3119 setparent:3120
ианыланы     1/6     7→6     а-ны́·ла-ра V Dyn Intr NonFin Cnv Abs Aor S:3 LO:3SgNH >ADVCL:CONV with:4697
инеиуан      1/8     ✓ 7     а-не́и-ра V Dyn Intr Fin Impf S:3 >ROOT map:5140
```

Fig. 5: Screenshot illustrating analyses

5. The treebank

For the first version of the Abkhaz UD treebank, I have analysed 231 sentences taken from a collection of fairy tales for children.[10] Those are relatively short and straightforward sentences, but they nevertheless display a lively language that is quite representative of syntactic constructions in Abkhaz.

The analyses are stored in the AbNC tool database, but they can be imported into the INESS platform.[11] INESS (Norwegian Infrastructure for the Exploration of Syntax and Semantics) is

[9] The displayed analysis is the unedited one, the edited one is also shown in the interface, but not in the screenshot.
[10] Аҧсуа лакəқəа – Ахəычқəа рзы, editor: А. Мықəба, 2013
[11] https://clarino.uib.no/iness

a treebanking infrastructure that was developed at the CLARINO Bergen Centre;[12] it hosts treebanks for many languages in several formalisms and provides tools for viewing and searching treebanks. The INESS query language, which is based on first order predicate logic, allows quite sophisticated linguistic searches in treebanks.[13]

To illustrate the search capabilities, we can have a look at cleft constructions, which feature prominently in Abkhaz. A (pseudo-)cleft sentence is a copula construction whose subject (as given topic) is a relative clause, and its predicative is in focus (as new information). If the relative phrase is to the right of the copula verb, the construction is called cleft construction, otherwise pseudo-cleft construction. This definition can be translated into the following INESS query expression:

```
#c:[features=("Cop")] >csubj:rel #rel
```

This query can be paraphrased as: Find a word *#c* that has the feature *Cop*, and which has a dependent word *#rel* with the dependency label *csubj:rel*, i.e., that is the subject of the copula and is at the same time a relative clause. This query returns several matching sentences, among them the following:

шәызҿу	закәызеи	шәара,	абзамыҟәа?
š°ə-z-č̣-ú	z-á-ḳ°ə-zei	š°ará,	á-bzaməq̇°-k°a?
you-what.Rel-be.in-Pres	what-it-is-Q	you.Pl	the-fool-Pl

‘What is it you are doing, you fools?’

In the display of the corresponding dependency structure (Fig. 6), the matching nodes *#c* and *#rel* are highlighted in red.

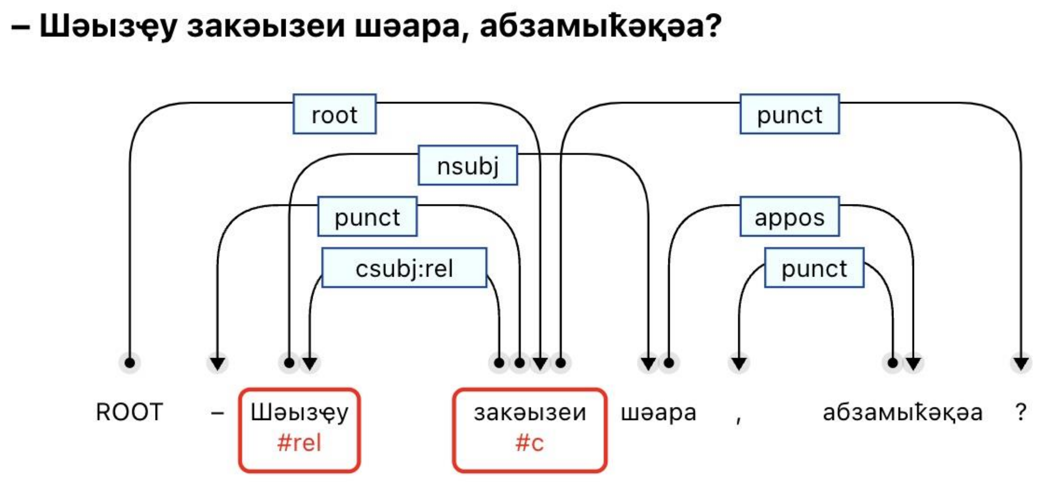


Fig. 6: Dependecy structure

6. Conclusion and future work

The small treebank I have built so far is just a start, and I will expand it in the future with more complex sentences drawn from other texts and text categories. In this process, I will need to improve the dependency grammar to cover syntactic constructions that I have not encountered so far. The tools I have developed are providing me with the best possible support to annotate new material rapidly and consistently.

[12] https://clarino.uib.no
[13] Meurer 2012.

I am working on contributing the treebank I am building to the Universal Dependencies collection of treebanks. The analyses in the treebank will still have to be modified to qualify and be accepted as genuine UD analyses: the POS and morphological features that the parser uses internally (and that are used in the AbNC) must be translated into UD tags, which take the form of attribute-value-pairs. Where possible, features and values already documented in the guidelines should be reused, and new features and values necessary for Abkhaz must be registered and documented. The Abkhaz treebank will be part of the next UD release, v2.14, which is scheduled for release in May 2024.

References

Kaslandzia (2005): Владимир Арушанович Касландзия, Аԥсуа-аурыс жәар / Абхазско-русский словарь. В 2 томах. Аҟәа / Сухум. http://apsnyteka.org/271-kaslandzia_v_abkhazsko_russky_slovar.html.

Meurer (2011): Paul M., “A finite state approach to Abkhaz morphology and stress”. *Lecture Notes in Computer Science* 6618, 271–282. https://clarino.uib.no/abnc/doc/Tbilisi2009-LNAI.pdf.

— (2012): Paul M., “INESS-Search: A search system for LFG (and other) treebanks”, in Miriam Butt and Tracy Holloway King (eds.), *Proceedings of the LFG ’12 Conference,* Stanford, CA: CSLI Publications, 404–421. http://csli-publications.stanford.edu/LFG/17/lfg12.html; http://cslipublications.stanford.edu/LFG/17/papers/lfg12meurer.pdf.

— (2018): “The Abkhaz National Corpus”, in Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (eds.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation* (*LREC 2018 Miyazaki*), no. 548, 2456–2460. http://www.lrec-conf.org/proceedings/lrec2018/index.html; http://www.lrec-conf.org/proceedings/lrec2018/pdf/548.pdf.

Tesnière (1959): Lucien T., *Éléments de syntaxe structurale*. Paris: Klincksieck. https://archive.org/details/LucienTesniereElementsDeSyntaxeStructurale.

აფხაზური ენის ხეთა ბანკი აფხაზური ენის ეროვნული კორპუსი, აფხაზური ენის ანალიზი და შესაფერისი ინსტრუმენტები

პაულ მოირერი (ბერგენი)

სტატიაში წარმოდგენილია აფხაზური სინტაქსური ბანკის შექმნის მცდელობა უნივერსალური დამოკიდებულებების (UD) ჩარჩოს ფარგლებში.

UD არის თეორიული ჩარჩო, რომელიც გამოიყენება გრამატიკის – მეტყველების ნაწილების, მორფოლოგიური მახასიათებლებისა და სინტაქსური დამოკიდებულებების თანმიმდევრული ანოტირებისთვის. დღეისათვის შექმნილი და ხელმისაწვდომია სხვადასხვა მოცულობის UD ხეთა ბანკები 141 ენისთვის, რომელთა შორის აფხაზური არ არის. წარმოდგენილი პროექტი მიზნად ისახავს ამ ხარვეზის შევსებას.

აფხაზური ენისათვის ხეთა ბანკის შემუშავება ეფუძნება აფხაზური ენის ეროვნულ კორპუსს (AbNC), რომელიც წარმოადგენს აფხაზური წერილობითი ტექსტების

გრამატიკულად ანოტირებულ კრებულს. ის შეიცავს ათ მილიონზე მეტ სიტყვას. გარდა იმისა, რომ კორპუსი ლინგვისტური კვლევის რესურსი და ინსტრუმენტია, კორპუსი ასევე წარმოადგენს ციფრულ ბიბლიოთეკას და პედაგოგიურ ინსტრუმენტს ენის შესწავლისთვის. აფხაზური ენის კორპუსში შესაძლებელია ტექსტების წაკითხვა, რომელიც ბუნებრივი, გადანომრილი ფორმით არის მოცემული და გვერდების მითითების საშუალებას იძლევა შემდგომი რეფერენციისათვის. კორპუსის ერთ-ერთ მთავარ ღირსებას წარმოადგენს მორფოლოგიური ანალიზატორი – მომხმარებელს შეუძლია დააწკაპოს ტექსტში მოცემულ ნებისმიერ სიტყვას, რის შედეგადაც ეკრანზე გამოჩნდება გრამატიკული ინფორმაცია შერჩეული სიტყვის შესახებ. გარდა ამისა, მომხმარებელს დამატებით შეუძლია გამოიძახოს შერჩეული ერთეულის შესაბამისი სიტყვა-სტატია კორპუსში ინტეგრირებული აფხაზურ-რუსული ლექსიკონიდან (კასლანძია, 2005 წ.).

აფხაზურის მორფოსინტაქსური ანალიზი: ანალიზატორის ლექსიკური მოდული აგებულია როგორც სასრული ავტომატი (Meurer 2011), ხოლო დისამბიგვირება (ორაზროვნების მოხსნის მექანიზმი) ჩაშენებულია შემზღუდველების გრამატიკის (CG) ფორმალურ მოდელში. ანალიზატორის აგების დროს მთავარი გამოწვევა იყო სიტყვაფორმების უჩვეულოდ მაღალი დონის ომონიმიასთან გამკლავება, რომელიც განპირობებულია აფხაზური ზმნის პოლისინთეზური ბუნებით და წერილობით ტექსტში მახვილის არარსებობით. CG-ის წესები, რომლებიც ითვალისწინებენ სინტაქსურ კონტექსტს, გარკვეულწილად შეიძლება გამოყენებულ იქნეს ომონიმიის მოსახსნელად, მაგრამ ომონიმიის სრული მოხსნისთვის ხშირად სემანტიკური ინფორმაციაც არის საჭირო. სემანტიკური ინფორმაცია შეიძლება ნაწილობრივ ინტეგრირებული იყოს (და არის კიდეც) CG-ის პარსერში, მაგრამ პრინციპული მიდგომა დაფუძნებული უნდა იყოს სიტყვათა ქსელზე, ან ოქროს სტანდარტის კორპუსიდან მიღებულ სტატისტიკურ ინფორმაციაზე (მაგ. სიტყვების ვექტორებზე).

ხეთა ბანკების აგება: დამოკიდებულებების ხეთა ბანკები შეიძლება აიგოს სხვადასხვა გზით: მანუალურად (ხელით აშენების მეთოდის გამოყენებით), სტატისტიკური ან წესებზე დაფუძნებული პარსერის გამოყენებით, ან ამ მეთოდების კომბინაციით. ჩემს პროექტში ვიყენებ წესებზე დაფუძნებულ პარსერს, რასაც მოჰყვება მიღებული შედეგის ხელით კორექტირება. დამოკიდებულების წესები დაწერილია შემზღუდველების გრამატიკის ფორმალურ მოდელზე დაყრდნობით და წარმოადგენს მორფოლოგიური ანალიზის მოდულის დამატებას. წესები დიდწილად ეყრდნობა მორფოსინტაქსურ მახასიათებლებს. საუკეთესო შედეგების მისაღებად, გადამწყვეტი მნიშვნელობა ენიჭება სიტყვების ომონიმიის სრულ (და სწორ) მოხსნას. მანქანური ანალიზის შემდეგ, მიღებული სინტაქსური ხე ხშირად ხელით უნდა შესწორდეს, ან წესები უნდა დაიხვეწოს ან დამატებითი წესები უნდა დაიწეროს რთული შემთხვევების გასაანალიზებლად. ამ მიზნით მე შევქმენი რამდენიმე ინსტრუმენტი, რომელიც ხელს უწყობს ამ პროცესის გამარტივებას: თითოეული (არასწორად გაანალიზებული, ან ორაზროვანი) სიტყვის ანალიზი ადვილად შეიძლება შეიცვალოს და შედეგად მიღებული ხეები შეიძლება რეორგანიზებულ იქნეს გრაფიკულ ვებინსტრუმენტში.

მიღებული აფხაზური ხეთა ბანკი განთავსდა INESS-ში. INESS (ინფრასტრუქტურა სინტაქსისა და სემანტიკის შესასწავლად) არის ნორვეგიული ინფრასტრუქტურა და ვებგვერდი, რომელიც სხვადასხვა ტიპის ხეთა ბანკების დათვალიერებისა და ძიების ეფექტურ შესაძლებლობებს იძლევა. გარდა ამისა, ჩემ მიერ აფხაზური ენისათვის შემქნილი ხეთა ბანკი უნივერსალურ დამოკიდებულებათა ბანკების შემდეგი ოფიციალური გამოშვების ნაწილი იქნება.