Proceedings of the NoDaLiDa 2019 Workshop on Constraint Grammar - Methods, Tools and Applications

30 September 2019

Editors
Eckhard Bick and Trond Trosterud

Proceedings of the NoDaLiDa 2019 Workshop on Constraint Grammar - Methods, Tools and Applications
Edited by Eckhard Bick and Trond Trosterud

Linköping Electronic Conference Proceedings No. 168
ISSN: 1650-3686, eISSN: 1650-3740
Linköping, Sweden, 2019
NEALT Proceedings Series 43
ISBN: 978-91-7929-918-7
URL: http://www.ep.liu.se/ecp/contents.asp?issue=168

Preface

This volume contains the articles presented at the Constraint Grammar workshop on methods, tools and applications, co-located with the NoDaLiDa 2019 conference in Turku, held on 30 September 2019. This workshop series has been part of the NoDaLiDa conference since 2005, and is the eight in the row, thereby emphasizing the Nordic roots of Constraint Grammar.

True to its tradition, the workshop may be characterised along two thematic lines: The development of constraint grammars analysing individual languages or specific aspects of their grammar, and presentations and discussions on practical language technology tools where CG is the key component. Moreover the workshop also contained two more general papers, dealing with theoretical issues relevant to any language.

As for the first theme, these proceedings contain papers presenting CG grammars for Tibetan (Faggionato and Garret), Lithuanian (Jagiella), and Greenlandic (Molich). The one on Tibetan looks at verb valency for texts from different historical phases of the Tibetan literary language, whereas the one on Lithuanian presents the first version of a general disambiguator for the language. Molich's paper on Greenlandic looks at a particularly vexing problem of Greenlandic grammar: The disambiguation of conjunctional and adverbial functions of enclitical particles. In addition to being central in determining the overall structure of the Greenlandic sentence at hand, they also affect the performance of current work on Greenlandic to Danish machine translation. A paper with a similar scope is Schmirler and Arppe's presentation of a set of rules for dealing with negation in Plains Cree.

Relevant to the second theme are two papers on grammar checking: Aldezabal, Arriola and Estarrona present a grammar-helping tool for Basque, and Wiechetek, Moshagen, Gaup and Omma use the workshop to launch a brand-new grammar checker for North Saami. Grammar checking tools have a long tradition within CG, but these two presentations introduce grammar checking to languages with a much richer morphology than usual.

The workshop also contains some more general papers. Bick's "Tagging What Isn't There" discusses methods for annotating information not explicitly present in the language under analysis (here: Danish). Being set in the context of an MT project from Danish to Greenlandic the paper could be seen under the two previous categories as well, but the approach is kept at a general level, using Danish as an example language.

The two last papers look at interaction between CG and different machine learning approaches. The paper by Yli-Jyrä shows that both CG and recursive neural networks have finite-state properties, and discusses the theoretical implications of this observation. Finally, the paper by Muischnek, Müürisep and Särg describe how CG is used to build gold standards for machine learning, by tagging the Estonian Universal Dependency corpus with CG.

As can be seen from the presentation, CG holds its position as the dominant framework for morphology-rich languages. Greenlandic, Plains Cree and North Saami all belong in the morphology-rich corner of the typological spectre, and the rest of the languages under scrutiny also possess more inflectional categories than the mainstream languages. Another characteristics of CG, its ability of providing analyses good enough to be used for practical applications, is also evident from the list of contributions.

On behalf of the workshop organizers
Eckhard Bick & Trond Trosterud

Workshop organizers:

- Eckhard Bick, Research lector, Institute of Language and Communication, University of Southern Denmark
- Tino Didriksen, Developer, GrammarSoft ApS
- Kristin Hagen, Senior engineer, Tekstlaboratoriet, University of Oslo
- Inari Listenmaa, Ph.D. student, University of Gothenburg and Chalmers University of Technology
- Kaili Müürisep, Senior research fellow, Institute of Computer Science, University of Tartu
- Trond Trosterud, Assistant professor in Sámi computational linguistics, University of Tromsø

Program Committee:

- Eckhard Bick (Chair)
- Kristin Hagen
- Inari Listenmaa
- Kaili Müürisep
- Anders Nøklestad
- Trond Trosterud

Workshop website:

https://visl.sdu.dk/nodalida2019.html

Table of Contents

A modular grammar-helping tool for Basque: work in progress

Izaskun Aldezabal
IXA NLP group
University
of the Basque Country
izaskun.aldezabal@
@ehu.eus

Jose Mari Arriola
IXA NLP group
University
of the Basque Country
josemaria.arriola
@ehu.eus

Ainara Estarrona
IXA NLP group
University
of the Basque Country
ainara.estarrona
@ehu.eus

Abstract

In this article, we explain the first steps towards a grammar-helping tool for Basque from a ruled-based approach. Specifically, we show the first steps carried out for helping with verb agreement, some of the difficulties encountered, which linguistic issues arise when new rules are designed, and future perspectives.

1 Introduction

This article concerns the ongoing work of a Constraint Grammar (vislcg3) (Bick and Didriksen, 2015) based tool for helping with useful information for dealing with verb agreement in sentences. The evaluation report of the Basque Government (Government, 2017) about grammar competence at Primary School includes verb agreement and incorrect use of ergative as grave errors if they occur repeatedly. Based on this fact, the purpose of this work is twofold: a) detecting agreement errors and give help with that kind of grammatical information; b) helping to develop a system to certify the Basque level automatically, a similar approach to Hancke et al. (2012). For the first purpose, we follow similar steps proposed in DanProof (Bick and Didriksen, 2015; Antonsen et al., 2009). Concerning the second goal, the plan is to collaborate with HABE (Institute for Adult Literacy and Basque Learning) which certify Basque levels.

The underlying ideas for both goals are extending grammatical knowledge of the student and helping to certify the language level corresponding to each student. In this paper, we will focus on the detection of some agreement errors.

SAROI (Oronoz et al., 2010) is one of the first tools for detecting syntactic errors used in Basque based on the rule-based approach. Wiechetek (2017) gives an overview of Constraint Grammar-based grammar checkers for many languages.

In the preliminary study presented here, we have started using the information provided by the auxiliary verb in the sentence. In Basque, the auxiliary verb carries information, among others, about the arguments of the verb, including the subject, the object and the indirect object; whether they are first, second or third person, and whether they are singular or plural (Laka, 1996). The auxiliary verb must keep the agreement with such arguments so that the sentence is grammatical. However, it is known that errors that disturb the syntax and semantics of the whole sentence of running texts go beyond the morphological concordance between the auxiliary verb and the mentioned three arguments. For instance, for the verb erosi ('to buy') we find examples like:

(1) *Mikelek tomateak 5 eurogatik erosi ditu*
Mikel-Erg tomatoes-Abs 5 euro-Mot buy have
'Mikel has bought tomatoes for 5 euros'

In (1) where the argument *eurogatik* 'for 5 euros' expressing "asset" with the *-gatik* ('for') motivative case is not considered suitable with the verb *erosi* 'to buy' (surely used incorrectly by the interference of semantic equivalents of the Spanish preposition *por*). To deal with this type of errors, other kinds of linguistic resources are needed, such as verb lexicons containing information regarding valency and semantics of arguments, what we find in the Basque Verb Index (BVI) (Estarrona et al., 2018). Based on the information containing in this lexicon for the verb *erosi* 'to buy', we are able to determine that the third argument expressing "asset" is realized with the inessive case instead of the motivative one.

In this line, Wiechetek (2017) managed to detect valency errors based on a deep syntactic and semantic analysis using Constraint Grammar. For the future, we plan to reuse the BVI lexicon following the same idea.

In the current approach, we have implemented the first module of agreement rules using auxiliary information, and we have studied the frame and argument structure needed for a more global approach.

The paper is organized as follows. Section 2 deals with the adopted methodology and development phase (the corpus, grammar formalism and design principles), Section 3 describes the preliminary evaluation and, Section 4 explains the further steps and future work. Finally, Section 5 will present some conclusions.

2 Methodology

In this section, we present the initial steps of our methodology.

2.1 Compiling available corpora

For the construction of the grammar, we have used a fragment of annotated corpora with agreement error tags (Aldabe et al., 2007). The error corpus for developing the grammar contains 8.368 words and the corpus for testing contains 14.257 words. It is a heterogeneous corpus, containing different types of texts such as abstracts of final degree reports of university students, compositions of Basque learners of intermediate and high level, compositions of students of Basque for special purposes etc.

We have used the 8.368 word sample for developing the grammar and the other 14.257 word sample for testing and controlling false positives. For the later goal, we have also used a sample of EPEC, the Reference Corpus for the Processing of Basque (Aduriz et al., 2006), available in the Ixa group. The sample contains the 10 most frequent verbs in EPEC (covering the 85% of the corpus).

2.2 Analyzing available corpora

As starting point, we use the output of the morphological analyzer (naki Alegria et al., 1996) with all the analyses. We did not use the disambiguation module because it could eliminate correct information that might be needed later to find the error.

2.3 Designing initial grammar

The initial grammar covers maintaining agreement between finite verbs and subjects and objects. In addition, the tool provides possible correct alternatives for repairing those agreement errors. The system uses morphological information, and has a special focus on finite verbs, because we get basic information for checking the verb agreement with subject and object from them. For instance, in (2):

(2) *Diseinu inteligentearen bultzatzaileak beste bide batetik sartu nahi dute kreazionismo*
Design intelligent-Gen the prime movers-Erg another way from-Abl to lead wanted creationism-Abs
'The prime movers of the intelligent design wanted to lead creationism from another way'

Bultzatzaileak 'the prime movers' (with the *–ak* ergative third person singular or absolutive third person plural mark) is grammatically incorrect, because it does not agree with the auxiliary *dute* which demands ergative case, third person and plural.

This mistake is also common in native Basque speakers, specially writing. In these cases, we attach advice tags to finite verbs involved in the agreement error and the words with the incorrect morphological case for the agreement. For instance, we add to the word containing the error *bultzatzaileak* a helping message, such as "take care of the agreement for ergative plural" as shown in the next rule example:

(3) ADD (%Take_care_of_agreement_ERG_PL)
TARGET (ERG) IF (0 ERG-SING) (NOT *1 ERG-PL) (NOT *1 (NR_HAIEK)) (*1 (NK_HAIEK-K) BARRIER (NK-HARK));

The above rule attaches to the singular ergative *bultzatzaileak* the helping message, if there is an auxiliary verb that needs third person plural ergative (NK_HAIEK-K) and there is not an auxiliary verb that demands third person plural absolutive (NR_HAIEK) and the checking is delimited by an auxiliary verb that involves third person singular ergative (NK-HARK).

The current version of the grammar only handles agreement errors of sentences where finite verbs are involved.

3 Preliminary evaluation

The initial grammar rules to find errors describe the conditions for valid structures for sentences where finite verbs are involved, and if these conditions are not accomplished the error tags are added.

In order to evaluate the grammar, as mentioned we have used the hand-annotated corpus (14.257 words). We chose to evaluate the agreement of absolutive and ergative cases. In this section, we give a preliminary evaluation:

- **Annotated errors correctly detected**: for the absolutive case the 50% of the errors are detected correctly. Concerning ergatives, we are able to detect correctly the 28% of the annotated errors. We consider as erroneous annotations when the error tag is assigned to a correct auxiliary verb and to a correct word containing absolutive or ergative case.

Apart from uncorrect annotations there have been detected a big amount of false positives.

From a qualitative point of view the main difficulties encountered by our grammar are the following:

- **False positives**: most of the false positives encountered are due to the ellipsis of the grammatical objects or subjects. In these cases the helping messages are unnecesary because there is not an error. But the messages are just attached to the auxiliary.
- **Complex constructions**: dealing with some subordinating sentences is challenging in the case that the barriers are properly established. We need to improve barriers with a more systematic treatment.
- **Ambiguity of the input**: in the initial approach, we have used the output of the morphological analyzer with all the information, but the preliminary evaluation show us the need of an adaptation of the POS disambiguation module in order to discard verb/noun ambiguity, but maintaining cases.
- **Linguistic issues**: dealing with errors where –ak (absolutive plural / ergative singular) case is involved. For instance in (4):

(4) *Tabernak izugarrizko kutxak egiten dituzte*
Bars-Erg-S/Nom-Pl great takings obtain
'Bars obtain great takings'

This kind of errors would ideally be solved with more complex knowledge. Therefore, in these cases we just can give as advice that the ergative plural is missing according to the auxiliay verb.

- **Bad or incomplete rules**: in some cases we should refine our rules, because we have not taken into account some grammatical possibilities of the language.

In order to improve the ongoing grammar we need more corpora for a more exhaustive analysis.

4 Next steps and future workPreliminary evaluation

Considering the preliminary evaluation and the difficulties encountered, we have in mind the following steps:

- Try to find a solution to the phenomena explained in the previous section.
- Extend these small-scale studies on certain error types to a large-scale analysis of real word student's errors, compiling the learner's corpora for each level.
- Analyze if this kind of agreement errors appear in all levels
- Include the BVI information in the grammar and in the analyzed corpora, and see in which extend could improve the results.

5 Conclusions

This paper has presented a preliminary constraint grammar for helping Basque students with grammatical agreement. The preliminary evaluation indicates the main strategies to improve the results.

The grammar can be in principle reused for other applications that do not necessarily have anything to do with error detection, such as Intelligent Computer-Assisted Language Learning (ICALL) systems.

Acknowledgments

The research leading to these results has been carried out as part of the *DeepReading: Mining, Understanding, and Reasoning with Multilingual Content* project (RTI2018-096846-B-C21 (MCIU/AEI/FEDER, UE)).

References

Itziar Aduriz, Maria Jesus Aranzabe, Jose Maria Arriola, Aitziber Atutxa, Arantza Diaz de Ilarraza, Nerea Ezeiza, Koldo Gojenola, Maite Oronoz, Aitor Soroa, and Ruben Urizar. 2006. Methodology and steps towards the construction of epec, a corpus of written basque tagged at morphological and syntactic levels for the automatic processing. *Corpus Linguistics Around the World. Book series: Language and Computers*, 56:1–15.

Itziar Aldabe, Bertol Arrieta, Arantza Diaz de Ilarraza, Montse Maritxalar, Ianire Niebla, Maite Oronoz, and Larraitz Uria. 2007. Basque error corpora: a framework to classify and store it. In *Proceedings of the 4th Corpus Linguistic Conference*, Birmingham, UK.

Iñaki Alegria, Xabier Artola, and Kepa Sarasola. 1996. Automatic morphological analysis of basque. *Literary and Linguistic Computing*, 11(4):193–203.

Lene Antonsen, Saara Huhmarniemi, and Trond Trosterud. 2009. Constraint grammar in dialogue systems. In *Northern European Association for Language Technology, NEALT*, pages 13–21.

Eckhard Bick and Tino Didriksen. 2015. Cg-3 beyond classical constraint grammar. In *Proceedings of the 20th Nordic Conference of Computational Linguistics, NODALIDA 2015*, pages 31–39, Vilnius, Lithuania.

Ainara Estarrona, Izaskun Aldezabal, and Arantza Diaz de Ilarraza. 2018. https://doi.org/s10579-018-9440-0 How the corpus-based basque verb index lexicon was built. *Language Resources and Evaluation. First Online 05 December 2018*, pages 1–23.

Basque Government. 2017. *Idazmenaren ebaluazioa. Testuak zuzentzeko irizpideak*. ISEI-IVEI, Hezkuntza Saila, Eusko Jaurlaritza.

Julia Hancke, Detmar Meurers, and Sowmya Vajjala. 2012. Readability classification for german using lexical, syntactic, and morphological features. In *Proceedings of the 24th international conference on computational linguistics (COLING)*, pages 1063–1080, Mumbai, India.

Itziar Laka. 1996. *A Brief Grammar of Euskara, the Basque Language*. Office of the Vice-Rector for the Basque Language, UPV/EHU.

Maite Oronoz, Arantza Diaz de Ilarraza, and Koldo Gojenola. 2010. Design and evaluation of an agreement error detection system: Testing the effect of ambiguity, parser and corpus type. *Proceedings of the 7th international conference on Advances in natural language processing (IceTAL 2010)*, 6233:281–292.

Linda Wiechetek. 2017. *When grammar can't be trusted - Valency and semantic categories in North Sami syntactic analysis and error detection*. Ph.D. thesis, UiT The arctic university of Norway.

Tagging What Isn't There:

Enriching CG Annotation With Implicit Information

Eckhard Bick
University of Southern Denmark
eckhard.bick@mail.dk

Abstract

This paper examines ways to make existing Constraint Grammar (CG) annotation grammatically more explicit, allowing corpus users and application programs, such as machine translation (MT), to refer to context-implied grammatical features in a more direct fashion. Two types of categories are addressed. First, morphological categories are propagated to words that leave them under-specified (e.g. number and definiteness for Danish adjectives) or unexpressed (e.g. person-number for Danish verbs). Second, we also introduce new categories, such as aspect and future tense for Danish, that may be morphologically explicit in a given MT target language, but do not exist in the source language. In a pilot evaluation of four categories in the context of Danish-Greenlandic MT, the implemented enrichment grammar for Danish achieved F-scores of 97% for propagated categories and 85% for new categories. In addition to feature tagging, structural annotation is also made more explicit, adding secondary dependency links for e.g. the subjects of relative and infinitive clauses, or attribute links between subject complements and subjects.

1 Introduction

Arguably, every annotated corpus has specific uses and target groups in mind, and the choice of to-be-annotated categories, tag set and granularity may hamper research the corpus creators had not thought of. Missing information may well be present in implicit form, but difficult or impossible for the user to query. Based on feedback from users of a Portuguese treebank, Freitas et al. (2008) propose the introduction of so-called *searchables* - secondary tags that would allow the corpus equivalent of a 1-click order, subsuming in one new tag information that would otherwise be distributed across several tokens, e.g. complex tenses or np definiteness.

The same rationale can be extended beyond the corpus arena, to NLP pipelines where grammatical annotation supports applications such as proofing tools, computer-aided language learning or machine translation, all of which may be in need of specific information not explicitly provided by the underlying parser.

The add-on feature enrichment grammar presented here departs from a standard DanGram[1] CG annotation (Bick 2001) and systematically addresses under-specified and implicit information, progressing from simple morphological categories to more complex categories and dependency syntax. In sections 2 (morphology) and 4 (syntax), existing categories are treated. Section 3 is about adding new categories from distributed context clues, and section 5 addresses dependency issues. Rules and examples are for illustration purposes only. In the actual grammar, there are up to 20 rules, and/ or additional context restrictions, for the more difficult features.

[1] For an online demo and documentation, cf. http://visl.sdu.dk/da/

2 Inflection categories

2.1 Inflection morphemes

Traditional morphological analysis breaks down a word into morphemes and assigns feature-attribute pairs such as singular/plural for the number category of Danish nouns. In some cases, however, categories can be systematically underspecified and need to be disambiguated based on context. An example is the -e ending on Danish adjectives[2], wich is used independently of gender (nG=no gender), for either singular (S) definite (DEF) noun phrases (np's) or for plural (P) np's, in which case definiteness is left underspecified (nD):

store ('big')
"stor" ADJ nG P nD NOM ('[the] **big** cars')
"stor" ADJ nG S DEF NOM ('the **big** car')

One could say that the morphological categories of gender, number and definiteness come in two feature-bundles for the adjective ending -e. A tagger with a minimalist disambiguation approach would be content to choose one of the two reading lines in the cohort from context. However, a Danish np as a whole does have explicit gender (a lexical feature of the head noun, UTR[3]/NEU) and definiteness (marked either as an inflexion feature on single nouns, or by a lexically definite modifier in np's):

de P **DEF** *store* **nG** P **nD** *ulve* **UTR** P IDF
('the big wolfves)

In a first step, the new add-on grammar propagates the definiteness information from the article and the gender information from the noun, resolving the adjective's *nG* as *UTR*, and *nD* as *DEF*, making it possible to search for *common gender definite plural* adjectives in an annotated corpus.

2.2 Generalized inflection categories

However, it would still be difficult to search for *all definite/indefinite np's*, because there is no single np token that safely carries this information as an inflection morpheme. The logical candidate for a search target would be the head of the np, since it is the only obligatory part, but Danish nouns are only inflected for definiteness, if there are no (pre)modifiers. Therefore, in multi-word np's, an IDF tag on the noun head is merely a morphological zero-morpheme, while the definiteness information is distributed across other constituents of the np. This is a problem not only for corpus searches, but also for other tasks, such as syntactic tagging, topic/focus tagging, semantic role tagging and machine translation (MT). For instance, definiteness is one of several clues allowing a parser to distinguish between subjects and objects, or between agents and non-agents. In Danish-Greenlandic MT, the case of direct objects depends on definiteness (absolute changed into instrumental case for indefinite objects), and transitive Greenlandic verbs add a special *half-transitive* affix, if the direct object is indefinite.

In this case, because the noun already has a *morphological* tag for definiteness (here: IDF), our grammar adds a secondary <def> or <idf> tag referring to the definiteness of the entire np:

SUBSTITUTE (N) (<def>) TARGET (N IDF)
*-1 DEF-EDGE BARRIER NON-ATTR *(add <def> to indefinite nouns, if there is a definite np-edge word to the left with nothing but attributes in between, where DEF-EDGE is a set containing definite articles, demonstratives, possessives, genitive nouns etc.)*

2.3 Category propagation

In a further step, categories can be propagated to words that do not have them in Danish. A case in point are Danish verbs that only allow (1) tense and (2) participle morphemes, but completely lack person-number inflection common in many other European languages and Greenlandic. The example rule below harvests a person-number variable (e.g. 1S, 3P) from subject pronouns, unless they are conjuncts <cjt>, exploiting the c (child/daughter) dependency relation between subject and tense-carrying (finite) verb.

SUBSTITUTE (V) (V /$1/v)
TARGET V-TENSE
(c @SUBJ + (/\([123][SP]\)/r) - <cjt>)

Similar rules add *3S* and *3P* if the subject is a singular or plural noun, respectively. Co-

[2] Adjectives with a baseform ending in -e, and all comparative forms, do not inflect at all, and are thus ambiguous in all categories.

[3] Danish has a 2-gender system, NEU (neuter) and UTR ("utrum", common gender, historically a fusion of masculine/feminine genders)

ordinated subject trigger a plural marker, clausal subjects a singular.

A special case are the relative pronouns *som* and *der* that do not inflect in Danish, but may do so in an MT target language. Here, the dependency link from the relative clause to the antecedent can be used to recover number, gender or even semantic features from a noun, allowing MT transfer rules to "see" the necessary slot filler information in the relative clause itself.

2.4 Cross-level category transfer

Sometimes a morphological category in one language is entirely absent in another (or drastically under-specified), but still represented at the syntactic or semantic level. For instance, Danish cannot match the 6 cases used in Greenlandic, but case can still be assigned by identifying corresponding syntactic or semantic tags on the Danish side.

```
(a) SUBSTITUTE (NOM) (REL)
      TARGET N + @SUBJ
      (p VFIN LINK *1S <mv> LINK c @ACC)

(b) SUBSTITUTE (NOM) (LOK)
      TARGET §LOC
```

Thus, rule (a) assigns relative[4] case (REL) to subjects, if the parent (p) vp has a child daughter dependent (c - child) that is a direct object (@ACC). Rule (b) is an example of converting Danish semantic role tags[5] like §LOC (location) into Greenlandic case (LOK - locative).

3 Distributed information

Category mapping gets more complex, if the necessary information is distributed across several words. In Danish, this is the case for aspect, future tense and aktionsart, all of which are difficult to determine and have to be inferred from auxiliaries, framenet and semantic role tags (Bick 2011), adverbial particles and other clues. In our MT system, we introduced the secondary tags <fut> (future tense) and <iter> (iterative) in order to match special Greenlandic affixes, *SSA* and *TAR*, respectively.

```
(a) SUBSTITUTE (V) (<fut> V)
      TARGET ("ville" PR &AUX)
      (*1 @ICL-AUX< LINK 0 ("få") OR <ve>
OR V-NONCONTROL LINK *1 @<ACC
CBARRIER VV)
      (NEGATE *1 @ICL-AUX< LINK 0
("have") LINK *-1 @SUBJ> + HUM-person)

(b) SUBSTITUTE (V) (<fut> V)
      TARGET ("ville" PR &AUX)
      (c ROLE-NONCONTROL + @SUBJ) ;
```

The two rules above select the futures tense meaning of the Danish auxiliary *"ville"* over its other meaning 'want_to'. (a) looks for main verbs with frames that are -CONTROL (e.g. <fn:bodystate>, <fn:undergo>, <fn:worsen>)[6], with a safety condition of having a direct object, and an exception for *"vil have"* ('wants to have') with a human subject.

(b), on the other hand, looks for a -CONTROL subject, e.g. semantic roles like §TH (theme), §EXP (experiencer) or §STI (stimulus). At the time of writing, the add-on grammar contains about 20 rules about future tense, using hints like the following:

- always <fut> with <fn:become_be>, *blive* ('become'), *komme (komme til at* - 'shall')
- <fut> with future-triggering adverbs, dates, weekdays, months, unless the latter are modified by *hver* ('each') or the containing clause is headed by a preposition + *at* ('that')
- never <fut> with <fn:be_attr>, *omfatte* ('include'), *tilhøre* ('belong to'), *være* ('be'), *kunne* ('can'), *burde* ('should'), *måtte* ('must')
- never <fut> with generic present tense (e.g. substances or celestial bodies as subjects)

Another difficult category is aspect, since Danish does not explicitly mark any aspect categories. Telicity has a strong lexical bias and for many verbs it is possible to infer a default tag from a given verb frame. In our

[4] Greenlandic is an ergative language, and uses the neutral case (ABS) for subject of intransitive verbs and for objects of transitive verbs, changing subject case into (REL) in the latter scenario.

[5] "Adverbial" roles, e.g. time and space, are often realized by pp's in Danish. Greenlandic has no prepositions, but because DanGram tags roles on the semantic head of the pp rather than its syntactic head (the preposition), there is a simple one-on-one correspondence between Danish semantic role and Greenlandic case

[6] Some 30 frames in all

telicity scheme we use a 5-way distinction, with ±static, ±telic and ±time. In the table, ±control (±C) is added.

			-Time (0)	+Time (1)
-Static	Telic (t)	+C	goal-action	goal-activity
		-C	result-event	result-process
	Atelic (a)	+C	do-action	do-activity
		-C	pass-event	pass-process
+Static	state (s)			

Table 1: Telicity

The 5 lexical aspect categories are tagged as <aa:t0>, <aa:t1>, <aa:a0>, <aa:a1> and <aa:s>[7]. Because of the (partial) overlap between tense and aspect, and because Greenlandic verbs do not mark tense, these categories can be used to choose a Danish translation tense in the absence of more specific clues (such as time adverbs). For instance, <aa:t0> and <aa:a0> verbs like '*ramme' (hit)*, *eksplodere' (explode)* and *'opdage' (notice)* are much more likely to occur in the past tense than in the present tense.

An example of grammatical aspect is the Greenlandic morpheme category of iterative, which corresponds to the suffix *TAR* and the Danish support verb *'pleje at' (use to).* In most cases, however, the category is unmarked in Danish and has to be inferred from context:

```
(a) SUBSTITUTE (V) (<iter> V) TARGET V
      (c §LOC-TMP LINK c ("hver")) ;

(b) SUBSTITUTE (V) (<iter> V) TARGET V
      (c ("om" PRP) LINK c §LOC-TMP
      LINK 0 (<weekday>) OR (<season>)) ;
```

The first example marks a verb (V) as iterative (<iter>) if it has a dependent (c)[8] with a temporal semantic role (§LOC-TMP) modified by the determiner pronoun *'hver' (each).* Rule (b) asks for the preposition *'om' (at/on/about)* with a temporal argument (weekdays or seasons).

[7] 'aa' stands for aspect/aktionsart

[8] The CG3 implementation of Constraint Grammar uses 'c' (child) rather than the traditional 'd' (dependent/daughter).

4 Secondary syntactic tags

Pronouns are often subdivided into syntactic or semantic sub-classes such as relative, determiner, interrogative and quantifier. However, these are not necessarily lexeme classes. Thus, in Danish, the syntactic category of reflexive is only lexeme-bound in the 3rd person forms *'sig'* (accusative 'him-/herself') and *'sin'* (his/her own), and otherwise identical with ordinary personal object pronouns and possessives. For 1./2. person the <refl> mark can be safely added (a) and for 3. ps. plural it can be guessed (b):

```
SUBSTITUTE (<poss>) (<refl> <poss>)
      TARGET (<poss> @>N)
      (0 (<\([12][SP]\)>r))
      (p (*) LINK *p VFIN
        LINK c @SUBJ LINK 0 (VSTR:$1)) ;

SUBSTITUTE (<poss>) (<refl> <poss>)
      TARGET (<poss> 3P @>N)
      (*p VFIN LINK c @SUBJ
        LINK 0 (3P) OR (P) OR <cjt-head>) ;
```

In a feature propagation step, the <refl> tag can then be exploited to mark reflexivity on transitive verbs, in the presence of a <refl> @ACC tag. This mechanism is part of a more general method: valency instantiation. From its lexicon, the DanGram parser draws tags for valency potential, such as <vt> for monotransitive, <vdt> for ditransitive or <vr> for reflexive. The add-on grammar "instantiates" these tags by adding a '¤'-sign to it, e.g. <¤vt> if there, in fact, is a direct object corroborating the monotransitive tag. In the case of reflexive verbs (<¤vr>) this is useful in our MT setting, because Greenlandic verbs need to be inflected for reflexivity.

In Danish, with the exception of object-elliptic relative clauses and non-interrogative object clauses, all subclauses must begin with a subordinator. This is an obvious MT advantage with Danish as source language (SL), because the subordinator serves as a surface clue classifying the subclause and for choosing the right target language (TL) conjunction (e.g. English) or mood (e.g. Greenlandic). Thus, the conjunction *'hvis' (if)* translates into conditional mood inflection in Greenlandic. However, a little-known quirk in Danish syntax does allow conjunction-less conditional clauses, if they are fronted and SV is inverted to VS: *Kommer han ikke, må vi udskyde mødet.* ('If he doesn't come,

we will have to postpone the meeting.') Here, because word order is the only clue, a secondary marker tag is needed:

```
SUBSTITUTE (V) (<if> V) TARGET VFIN
1:  (0 @FS-ADVL>)
2:  (*-1 >>> BARRIER NON-KC)
3:  ((*1 KOMMA BARRIER CLB OR VV -
@ICL-AUX< LINK 1 VFIN)
4:  OR (*1 @<SUBJ-ALL BARRIER NON-
PRE-N/ADV LINK *1 @FMV BARRIER CLB-
ORD LINK *1 @<SUBJ-ALL BARRIER NON-
PRE-N/ADV)) ;
```

The rules asks for an adverbial subclause tag (1), beginning-of-sentence (2), and - to the right (*1) - either (3) a comma followed directly by a finite verb (VFIN) or (4) a left-pointing subject followed by a finite main verb (@FMV) and another left-pointing subject.

5 Secondary dependencies

Dependency relations (between content words) are the backbone of semantic disambiguation, be it frame annotation, semantic roles or the transfer stage of a rule-base MT system. Thus, choosing one translation of a verb over another often depends on the semantic class of its subject or object. For instance, *ride* needs 2-3 different translations in many languages, depending on its object dependent, i.e. whether you ride a horse, bicycle or train. But what do you do, if the necessary dependent either is not there, semantically empty or too far away in the syntactic tree? This is the case in Danish infinitive clauses (missing subject) and can be the case in relative clauses (missing object):

Den forsikring, han tegnede, var meget dyr. ('The insurance he took out, was very expensive')

Unlike in section 2.3, in this example there is no relative pronoun, a secondary tag could be added to. Therefore, a more structural solution is in order: Secondary dependencies. These can of course be added by the application program, MT or otherwise, that take the CG annotation as input - by following dependency paths and duplicating them where necessary. However, it is also possible to address the problem CG-internally, using the ADDRELATION(S) operator introduced in CG3 (Bick & Didriksen 2015). It allows a two-way relation, here named 'c-acc' (accusative object child), when seen from the dependent (*, the antecedent of the relative clause's "invisible" object), and 'p-acc' (parent-of-accusative), when seen from the relative clause verb (@FS-N<).

```
ADDRELATIONS (c-acc) (p-acc) TARGET (*)
    TO (c @FS-N<)
    (*-1 @SUBJ>
        BARRIER <rel> OR _TARGET_)
    (NEGATE *1S <mv>
        LINK c PRP LINK NONE c @P<);
```

In order to make sure that there is indeed an elliptic object, the rule asks for a surface subject in the relative clause and the absence of a stranded preposition, i.e. a preposition without it's argument child (c @P<), that could also be elliptic in Danish relative clauses.

Other candidates for secondary dependencies are

- subject relation between the object of a sensory or controlling verb and a dependent infinitive *(see/let **someone buy** a ticket)*
- attributive relation between subject complement and subject, or between object complement and object
- coordination, linking conjuncts both to each other and to their joint head

Expanding our example sentence to cover all these cases, automatic (DanGram) annotation[9] will look like this:

Konsulenten lod ham vide, at den forsikring, han havde tegnet, var både dyr og dårlig. ('The consultant let him know that the insurance he had taken out was both expensive and bad.)

```
Konsulenten [konsulent] <Hprof> N UTR S DEF
    NOM @SUBJ> #1->2 (The consultant)
lod  [lade-1] V IMPF AKT @FS-STA #2->0 (let)
ham [han] <aci-subj> PERS UTR 3S ACC @<ACC
    #3->2 R:c-subj:4 (him)
vide [vide] <mv> V INF AKT @ICL-<OA #4->2
    R:p-subj:3 (know)
$, [,] PU @PU #5->0
at [at] <clb> KS @SUB #6->14 (that)
den [den] <dem> DET UTR S @>N #7->8 (the)
forsikring [forsikring] <f-right> N UTR S IDF NOM
    @SUBJ> #8->14 R:c-acc:11 R:p-attr:16 R:p-
    attr:18 (insurance)
$, [,] PU @PU #9->0
```

[9] The annotation was somewhat simplified by omitting valency and certain other secondary tags.

han [han] PERS UTR 3S NOM @SUBJ> #10->11 *(he)*
havde [have] <aux> V IMPF AKT @FS-N< #11->8 **R:p-acc:8** *(had)*
tegnet tegne] V PCP2 AKT @ICL-AUX< #12->11
$, [,] PU @PU #13->0 *(taken out)*
var [være] <mv> V IMPF AKT @FS-<ACC #14->4 *(was)*
både [både] ADV @FOC> #15->17 *(both)*
dyr [dyr] <cjt-head> <jval> ADJ UTR S IDF NOM @<SC #16->14 **R:p-cjt:18** **R:c-attr:8** *(expensive)*
og [og] <co-sc> KC @CO #17->16 *(and)*
dårlig [dårlig] <cjt> <jqual> ADJ UTR S IDF NOM @<SC #18->14 **R:c-cjt:16** **R:c-attr:8** *(bad)*
$. [.] PU @PU #19->0

(wordform [lemma] <secondary tags> POS INFLECTION @SYNTACTIC_FUNTION #id[dep]->id[head])

Secondary relations are appended as R: tags on both tokens involved in a (binary) relation, and contain a relation name followed by the id of the other token. *'R:c-attr:8'*, for instance, means the child end of an attributive relation, where 8 is the id of the (attributed) parent token. The latter here gets the same[10] relation name, but with a 'p-' (parent) prefix. Similarly, conjunction is tagged as *c/p-cjt*, and the subject and object relations as *subj* and *acc*, respectively.

It should be noted that all of the above are meant as primarily *syntactic* dependencies and that they are *secondary* in the sense that the child tokens in question unorthodoxically are allowed *two* dependency heads, where an ordinary syntactic tree would allow them either one or the other, but never both.

This is different from systematically adding a completely new, non-syntactic layer of dependency, as is the case when DanGram assigns semantic dependencies for frame- and role-carrying tokens (Bick 2011). In this case, a second, semantic tree is constructed, and the individual relations may or may not coincide with primary or secondary syntactic relations:

Den[den] <dem> DET UTR S @>N #1->2
forsikring [forsikring] <f-right> N UTR S IDF NOM @SUBJ> **R:sd-TH:5** **R:sd-TH:7** **§TH** #2->7
$, [,] PU @PU #3->0
han [han] PERS UTR 3S NOM @SUBJ> **R:sd-AG:5** **§AG** #4->5
tegnede [tegne] <fn:buy> <mv> V IMPF AKT @FS-N< **R:sd-ATR:2** **§ATR** #5->2
$, [,] PU @PU #6->0
var [være] <fn:be_copula> <mv> V IMPF AKT @FS-STA #7->0
meget [meget] <aquant> ADV @>A #8->9
dyr [dyr] <jval> <Deco> ADJ UTR S IDF NOM @<SC **R:sd-ATR:7** **§ATR** #9->7
$. [.] PU @PU #10->0
(sd=semantic dependency, fn: = framenet class, §AG=agent, §TH=theme, §ATR=attribute

6 Evaluation and statistics

Some preliminary, inspection-based[11], evaluation was carried out for four categories: (1) number propagation and disambiguation, (2) person-number tagging for finite verbs (from scratch), (3) future tense marking (<fut>) and (4) iterative marking (<iter>). In order to provide well-mixed attributes for these features, we used a section from Korpus 2010[12] containing blog/internet data.

	R	P	F
number	95.8%	100%	97.9
v pers/num	97.1%	97.5%	97.3
<fut>	83.3%	88.2%	85.7
<iter>	92.9%	78.8%	85.3

Table 2: Category tagging accuracy

Results indicate that the propagation and specification of morphological features (such as number and person) works best (F scores above 97%), most likely because they are mostly already inflection-marked on some other word in the sentence tree. Truly implicit features, that are never marked morphologically in Danish, are much harder to determine (F scores around 85%). Interestingly, <fut> suffered more from false negatives (low recall), while <iter> had more problems with false positives (low precision).

[10] CG3 allows arbitrary relation names, for both ends of a relation, so using prefixes and a common relation name is just a convention chosen here.

[11] Inspection is a fairly safe method for morphological categories, because there are few clear categories and clear morphological clues elsewhere in the sentence. The <fut> and <iter> categories are more likely to cause controversy in a multi-annotator scenario. As a "hard" criterion we plan to use the Greenlandic translation, that must make these categories explicit.

[12] Korpus 2010 was compiled by the Danish Society of Language and Literature (DSL) as part of the DK-CLARIN project.

7 On-the-fly corpus search markers

The focus of this paper has been the enrichment of an ordinary annotation run, feeding into an application (like MT) that needs implicit information made explicit (as tags) for specific purposes, or simply providing complete annotation of a category that does exist in the language in question, but is often left under-specified.

However, the same method can be put to a rather different use - on-the-fly marking of "corpus searchables". In this scenario, it is up to the (expert) user of an annotated corpus to formulate a search as a CG mapping rule, rather than an ordinary tag field query. In other words, the search engine interprets a 1-rule mini-CG at run time. Obviously, such a rule can exploit any existing annotation in a fully context-capable way, handling complexities far beyond any ordinary search. The example below presupposes a dependency tree and semantic role annotation, and also exploits two of the secondary tags introduced above (<fut> and <if>), but in principle, rules could even be written for raw text, using CG3's regular expression format.

"Find verbs with an experiencer subject, and check which ones are modified by conditional clauses in the future tense."

MAP (£mark) TARGET <mv> (*-1S VFIN LINK c @SUBJ + §EXP) (c @FS-<ADVL + <fut> LINK (c ("fordi")) OR (0 <if>)) ;

For smaller corpora, this is possible in real time, but for larger corpora, live processing and the ensuing impossibility of an optimized search structure (such as a database) means that search results cannot be piped to a GUI, but need to be written to a file for later inspection.

8 Conclusions and outlook

We have shown, how an existing CG annotation can be enriched without changing the original grammar, in a modular and application-driven fashion. Obviously, for both scenarios discussed here, corpus linguistics and machine translation, the choice of categories is task-dependent. For instance, propbank-style ARG0, ARG1, ... annotation could be added for a corpus user, and a different target language would require different categories. Thus, "non-factuality" is an inflection category in Romance languages (subjunctive), but not explicitly marked in Danish.

Future work should explore and evaluate which categories can be inferred from a standard (Danish) CG annotation with a reasonable level of accuracy, and which would need alterations in the original grammar or lexica.

References

Bick, Eckhard & Tino Didriksen. 2015. CG-3 - Beyond Classical Constraint Grammar. In: Beáta Megyesi: Proceedings of NODALIDA 2015, May 11-13, 2015, Vilnius, Lithuania. pp. 31-39. Linköping: LiU Electronic Press. ISBN 978-91-7519-098-3

Bick, Eckhard (2011). A FrameNet for Danish. In: Proceedings of NODALIDA 2011, May 11-13, Riga, Latvia. NEALT Proceedings Series, Vol 11, pp.34-41. Tartu: Tartu University Library. (ISSN 1736-6305)

Bick, Eckhard. 2001. En Constraint Grammar Parser for Dansk, in Peter Widell & Mette Kunøe (eds.), 8. Møde om Udforskningen af Dansk Sprog, 12.-13. oktober 2000, pp. 40-50, Århus University

Freitas, Cláudia & Rocha, Paulo & Bick, Eckhard. 2008. "Floresta Sintá(c)tica: Bigger, Thicker and Easier", in: António Teixeira et al. (eds.) Computational Processing of the Portuguese Language (Proceedings of PROPOR 2008, Aveiro, Sept. 8th-10th, 2008), pp.216-219. Springer

Constraint Grammars for Tibetan Language Processing

Christian Faggionato
SOAS, University of London
10 Thornhaugh St, Bloomsbury
WC1H 0XG
cf36@soas.ac.uk

Edward Garrett
SOAS, University of London
10 Thornhaugh St, Bloomsbury
WC1H 0XG
eg15@soas.ac.uk

Abstract

This paper describes the diverse and distinctive ways that Constraint Grammar has been used within a Tibetan verb lexicon project. We present three CG3 grammars and how they fit into our workflow, along with the practical problems they were designed to solve.*

1 Introduction

The aim of our work is to develop a corpus-based verb lexicon of Tibetan covering the three major periods in the history of the language: Old, Classical and Modern Tibetan. The starting point for this work is a manually annotated corpus of Tibetan texts. This is obtained by importing part-of-speech tagged Tibetan texts into the BRAT annotation tool, where human annotators then draw labeled dependency arcs between verbs and their arguments. Here's an example:

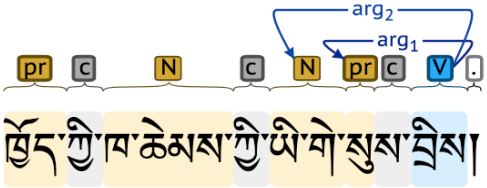

Figure 1. Verb-argument annotation.

In Figure 1, translated as "Who wrote the text of your testament?" (Wylie transliteration: *khod kyi kha chems kyi yi ge sus bris/*), the verb "write" is linked to its two arguments, the writer (*arg1*) and the thing written (*arg2*).

Although Tibetan sentences typically follow Subject-Predicate (Miller, 1970) or more specifically SOV word-order (DeLancey, 2003), deviation is permitted, as shown by Figure 1. Moreover, Tibetan case-marking does not provide a failsafe way of identifying verb arguments (Tournadre, 2010). Therefore, we hand-annotate these relations, but narrowly so, resulting in texts whose syntactic dependency structure is only partially annotated, as Figure 1 makes clear with its many unlinked words. This creates an opportunity for automated annotation methods to fill in the gaps. Section 2 of this paper describes a CG3 grammar that does just that.

Section 3 of the paper turns to the challenge of incorporating Old Tibetan materials into our workflow. Our Classical Tibetan annotators had the luxury of working with previously POS-tagged texts. However, no manually POS-tagged texts exist for Old Tibetan. We present some of the orthographic differences between Classical and Old Tibetan, and then describe the CG3 grammar we developed to normalize Old Tibetan texts into Classical Tibetan. By first applying this grammar, we can POS-tag our Old Tibetan texts using a tagger trained on Classical Tibetan materials.

In Section 4, we describe ongoing work on a third CG3 grammar, which has broader aims than the first two grammars. We wish to draw examples for our verb lexicon not just from manually annotated texts, but also from a wide

range of additional Tibetan texts. To do so, we must automatically annotate these further texts. The grammar described in Section 4 does just this, taking a POS-tagged text as input and outputting a text enhanced with the dependency relations we find essential.

We conclude the paper in Section 5. Because of the practical role these grammars currently play in our project, it would be both premature and improper to carry out a formal evaluation at this time. Instead, we make some concluding remarks and discuss the future direction of our work.

2 Other Dependencies

As illustrated in Figure 1, our project's manual annotations do not come close to providing full dependency parses for Tibetan sentences. In light of the project's primary goals, such complete parses may be unnecessary. Annotating certain relations, however, is essential. For example, in Figure 1, *arg1* is ཞ 'who'. The fact that it is followed by the ergative case marker suffix ས is important to us, because understanding how a Tibetan verb is used includes knowing how its arguments are case-marked.

We assert that it is possible to establish this particular case-marking relation, among other dependency relations, using automated methods. This is where Constraint Grammar 1 (G1) fits in. G1 consists of around a hundred hand-crafted rules which ensure that most words of a sentence have a non-root parent. G1 links modifiers such as adjectives, determiners and demonstratives to nouns, and converbs, punctuation and adverbs to verbs. We rely on Tibetan's relatively strict noun-phrase internal word-order (Garrett and Hill, 2015). Unsurprisingly, G1 consists largely of `SETPARENT` and `MAP` rules. Here is an example:

```
#genitive + pron:
SETPARENT (Case=Gen) (NONE p
ALLPOS) TO (-1 (PRON));
MAP (@case) TARGET (ADP) -
TAGS (p Head_NOUN OR (ADV));
```

In Tibetan, if a genitive case marker follows a pronoun, then it must depend on that pronoun via the `case` relation. The `SETPARENT` rule establishes this dependency, and the `MAP` rule assigns the tag `@case` to the genitive adposition.

Other examples are more complex, but in the end, the rules of G1 combine together to assign a near complete dependency parse for a Tibetan sentence, provided the starting point is text which has been manually annotated for verb-argument structure. Figure 2 shows the result of applying G1 to the sentence in Figure 1.

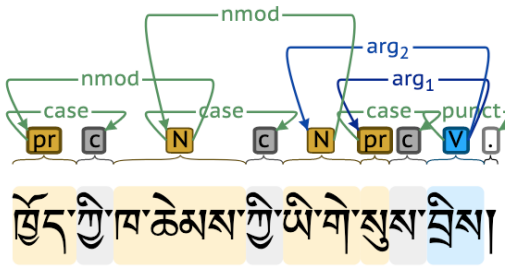

Figure 2. G1 applied to Figure 1.

The first two words of this sentence are the second person pronoun ཁྱོད་ 'you' and the genitive case marker ཀྱི. The dependency relations between them are established by the rules just mentioned.

Refinements and further additions to G1 may in time make it possible for us to offer a version of our hand-annotated texts that incorporates automatically inserted additional relations which fill in the gaps for a complete dependency parse. However, some relations are likely to require human adjudication, and so for now we content ourselves with lesser aspirations for G1.

3 Old Tibetan Normalization

Old Tibetan was initially introduced for administrative purposes and includes detailed historical accounts of the Tibetan Empire from the 7th to the 10th century (Hill, 2010). Although its vocabulary and grammar are strikingly similar to Classical Tibetan, it has many differences in spelling and orthography (Dotson & Helman-Ważny, 2016). For example, the Classical Tibetan genitive case marker ཀྱི may be written in Old Tibetan as ཀི. Instead of the standard *gigu* vowel we get a reverse *gigu*. In other cases, Old Tibetan words have characters that do not occur in their Classical Tibetan equivalents. Not only does the Old Tibetan form སྐྱེ 'person' have a

reverse *gigu*, but it also has a *ya-btags*: in Classical Tibetan the word would be much more simply མི་.

Our second CG3 grammar was developed to deal with these and other differences between Old and Classical Tibetan. We call it the Old Tibetan Normalization Grammar (G2), and its purpose is to make Old Tibetan look like Classical Tibetan. The differences just described can be characterized at the syllable level. It is possible in such cases to define simple regular-expression based SUBSTITUTE rules like the following rule. (Note that reverse *gigu* has been referenced using its Unicode escape value since superscript vowels display awkwardly when not attached to a base character.)

```
#Replace the reverse gigu
with gigu everywhere
SUBSTITUTE ("([^<]*)\\
u0F80(.*)"r) ("$1\u0F72$2"v)
TARGET (σ);
```

SUBSTITUTE rules do not always suffice to capture the differences between Old and Classical Tibetan. Traditionally in Classical Tibetan, syllables are separated by a *tsheg* (the dot seen in the above examples). In Old Tibetan texts, syllable margins are not so clear and often a syllable (verb, noun and so on) is merged together with the following case marker or converb: སྟེའི་ > སྟེབ་གཞི་, དུས་ > དུས་སུ་, བཀུག་ > བཀུག་མ་. To handle these cases, we came up with a cascading series of SPLITCOHORT rules, where initial rules split specific complex syllables into separate syllables, and later rules apply generically to syllables of a particular type. Here is an example of a specific SPLITCOHORT rule:

```
SPLITCOHORT (
  "<མཆི>" "མཆིས་" σ
  "<སྟ$1>"v "ན$1"v σ
) ("<མཆིསྟ(་?)>"r);
```

And here is an example of a more general and therefore less readable rule that applies to cases like གཅའོ་ > གཅལ་ཏོ་:

```
SPLITCOHORT (
  "<$1>"v "$1$3ད་"v σ
  "<$2>"v "ཏ$4"v σ
) ("<(.+)((.)\\u0F9F([\\
u0F7C\\u0F7A]་?))>"r);
```

The SPLITCOHORT rules reveal the form of the input that is passed to G2. Instead of passing word tokens, the grammar is passed syllable tokens. Any syllable which G2 normalizes is added in its original form as a new reading with the tag ↑OT by the following rule near the end of the grammar.

```
APPEND ("$1"v ↑OT)
("<(.*)>"r) (NOT 0 ("$1"v));
```

G2 concludes with a choice between two rules, depending on whether the user wants to select Old Tibetan "readings" (i.e. Old Tibetan syllables and syllables that didn't require normalization) or Classical Tibetan "readings" (i.e. Classical Tibetan normalizations as well as syllables not requiring normalization).

```
#Uncomment one rule:
#SELECT (↑OT);
#SELECT (σ);
```

Thus, each syllable of the input is treated as a CG3 cohort, whose different readings are the different syllable forms (Old or Classical) that it can take. Syllable readings are then joined together in their Classical Tibetan form in order to make a Classical Tibetan normalization.

The approach we are taking has three merits. First, we have carefully characterized the orthographic differences between Old Tibetan and Classical Tibetan, which is valuable in itself. Second, we can apply Meelen and Hill's (2017) tagger to the Classical Tibetan normalizations, rather than struggle with tagging Old Tibetan texts. And third, preserving a record of which syllables have been transformed enables us to reverse the process and denormalize back to Old Tibetan, after our Old Tibetan texts have been hand-annotated. After all, Tibetan scholars do not in general want Old Tibetan texts to look like Classical Tibetan.

4 Verb-Argument Annotation

So far we have described a workflow that consists of the following steps, which may not all be necessary for a given text:

```
text normalization → POS-
tagging → BRAT import →
manual verb-argument
annotation → automated other
dependency annotation
```

We would prefer to create a verb lexicon that is informed by and draws examples from Tibetan texts that have not been manually annotated, and not just those that have. To this end, we are pursuing various strategies for automatically annotating verb-argument structure.

The Verb Argument Dependencies grammar (G3) attempts to solve this problem with CG3. The input to G3 is a POS-tagged text without dependency annotations. G3 starts by inserting "helper" tags, such as tags which identify candidate constituent junctures. For example, the following rule tags those words which, if they occur to the left of a word, could not be part of a noun phrase with that word.

```
SET LEFT_NP_BOUNDARY =
(VERB) OR (ADP) OR (PUNCT)
OR (SCONJ) OR (PART);
```

The grammar then proceeds through dozens of `SETPARENT` and `MAP` rules, which set and label the verb-argument dependencies. These rules are rather intricate and will not be exemplified here.

G3 concludes with a series of "fixing rules" which `SUBSTITUTE` or remove mistaken tags. For example, if a noun preceding a genitive case marker has been marked as an argument, this cannot be correct, since a word to the right of the genitive would always be the argument.

```
SUBSTITUTE (@arg2) (*)
TARGET Head_NOUN + (@arg2)
(1 (Case=Gen)) (p (VERB));
```

In other cases, the fixing rules relate to specific verbs or verb classes that behave differently from the norm. For example, verbs of movement cannot take *arg2*:

```
SUBSTITUTE (@arg2) (@arg1)
TARGET Head_NOUN + (@arg2)
(p (VERB) + VMOVE - ("སྐྱེ"));
```

In general, G3 has worked very well with transitive verbs, where *arg1* is marked with ergative case. The main challenge has been to detect the argument structure of verbs with multiple arguments lacking case-marking.

5 Conclusion

In this paper we described three grammars that have proved helpful to our Tibetan verb lexicon project. By automating predictable dependency annotations, G1 has allowed our annotators to focus narrowly on verb-argument annotation. G2's treatment of Tibetan syllables as CG3 tokens has replaced haphazard search and replace with an accountable and reversible approach to text normalization. Finally, G3 is tackling the challenging task of automating verb-argument annotation. This remains a work in progress, subject to further improvement and comparison with alternative methods.

In future, we hope to address some missing elements of the work presented here. As regards G1, it will always be valuable to reduce the number of words outside the dependency structure. In addition, it may be worth evaluating the correctness of those dependencies which are not obvious against a reference set of hand-annotated examples. In terms of G2, the software processing pipeline including denormalization remains to be released. Finally, the status of G3 in our pipeline needs to be clarified; from there, evaluative metrics may well follow.

The texts and grammars discussed in this paper are freely available for anybody to examine and use. For further details, see our "Tibetan NLP" page on GitHub, in particular the `tibcg3` repository.

References

Scott DeLancey. 2003. Classical Tibetan. In *The Sino-Tibetan Languages*, edited by Graham Thurgood and Randy J. LaPolla, pp. 255-269. Routledge, London and New York.

Brandon Dotson and Agnieszka Helman-Ważny. 2016. *Codicology, Paleography, and Orthography of Early Tibetan Documents: Methods and a Case Study*. Wiener Studien Zur Tibetologie und Buddhismuskunde, Vol. 89. University of Vienna: Vienna.

Edward Garrett and Nathan Hill. 2015. Constituent order in the Tibetan noun phrase. *SOAS Working Papers in Linguistics* 17:35-48.

Nathan Hill. 2010. An overview of Old Tibetan synchronic phonology. *Transactions of the Philological Society* 108(2):110-125.

Marieke Meelen and Nathan Hill. 2017. Segmenting and POS tagging Classical Tibetan using a memory-based tagger. *Himalayan Linguistics* 16(2).

Roy Miller. 1970. A grammatical sketch of Classical Tibetan. *Journal of the American Oriental Society* 90(1):74-96.

Nicolas Tournadre. 2010. The Classical Tibetan cases and their transcategoriality: From sacred grammar to modern linguistics. *Himalayan Linguistics* 9(2). http://dx.doi.org/10.5070/H99223480. Retrieved from https://escholarship.org/uc/item/94d0447c.

Turrell Wylie. 1959. A standard system of Tibetan transcription. *Harvard Journal of Asiatic Studies* 22:261-267.

Developing a constraint grammar for Lithuanian

Trey Jagiella
Indiana University Bloomington
fjagiell@indiana.edu

1 Introduction

This paper presents a preliminary constraint grammar for Lithuanian. The main objective in developing this constraint grammar was precision. The corpus used to develop this constraint grammar with the Lithuanian ALKSNIS treebank from the Universal Dependencies Project (Bielinskiene et al., 2016). The pipeline consists of a morphological analyser of all possible interpretations for the wordforms in the corpus as well as a constraint grammar. In the test corpus, the constraint grammar has a precision of .9205, a recall of .1845, and an F1 score of .3074.

The paper is organized as follows: section 2 contains a brief review of literature, section 3 describes the analysis pipeline, section 4 describes the development process, section 5 evaluates the results, and section 6 presents the conclusion.

2 Review of literature

There has not yet been a constraint grammar developed for the Lithuanian language. There has, however, been a fair amount of linguistic research on the language. Since the fall of the Soviet Union, there has been greater study into other areas of the Lithuanian language. Little of this work, however, has been translated into other languages (Usoniene et al., 2012). Nevertheless, there are English-language books and translations on Lithuanian grammar as well as Lithuanian dictionaries readily available, which were consulted in the development of this constraint grammar (Mathiassen, 1997; Ramoniene and Pribusauskaite, 2008; Piesarkas, 2006; Piesarkas and Svecevicius, 1995).

3 Analysis pipeline

3.1 Morphological analyser

To create a list of wordforms and their possible interpretations, I used a Python program, which creates a list of all interpretations for a single wordform found in an input.

3.2 Rule writing

The constraint grammar, lt.cg3, is composed of 79 rules, 26 of which are remove and 53 of which are select. These rules were developed by running a sentence of the train CoNLL-U file through the morphological analyser to find a list of its outputs. Based on the ambiguities of the output and the correct lemma in the corpus, rules were written to make the constraint grammar pick the correct interpretation.

4 Development process

To test the rules in the constraint grammar and further develop it a script was run. It took the dev CoNLL-U file of the Lithuanian ALKSNIS treebank and ran it through the analyser and lt.cg3 files and then compared the output of this process to the annotation in the dev file. This script outputs the true and false positives for each rule; the number of input, output, and reference analyses; input, output, and reference ambiguity; total true and false positives and negatives; and precision, recall, and F-score.

From the rule by rule output of the script, poorly performing rules could be eliminated or modified accordingly.

5 Evaluation

5.1 Corpus analysis

The next table summarizes the ambiguity left in the test corpus after being run through the constraint grammar:

Ambiguity in the test corpus

	Input	Reference	Output
Analyses	12629	10118	10932
Ambiguity	1.25	1.0	1.08

From the table above, it can be seen that while there was a noticeable reduction in ambiguity, whether correctly or incorrectly, there still remains a large portion ambiguity within the corpora.

The following table demonstrates the performance of the constraint grammar through precision and recall.

Precision and Recall

	dev	test
Precision	.87	.92
Recall	.66	.18
F1 Score	.75	.31

As can be seen in the table above, the rules are performing relatively accurately. However, the recall is still fairly low, particularly in the test corpus.The F1 score shows that overall, the constraint grammar has a modest performance in disambiguating the corpus.

In the following table the number of true and false positives and negatives for the test corpus are presented:

Positives and Negatives in the test corpus

	Positives	Negatives
True	1563	3367
False	135	6915

From the table above, we can see that the number of true positives is significantly higher than the number of false positives. On the other hand, this is not the case for true and false negatives.

The difference in recall between corpora is intriguing. Given the fact each corpus is relatively small, just over 10,000 tokens each, there is plenty of room for variability and building rules using one corpus may not cover the language sufficiently to allow for effective rule making.

6 Conclusion

Although the precision is not bad at 92% in the test corpus, with a recall of 18% , there is still much work to be done. Any future rules written should be more accurate than the current ones in addition to the rewriting of current rules to increase their accuracy.

References

Agne Bielinskiene, Loic Boizou, Jolanta Kovalevskaite, and Erika Rimkute. 2016. Lithuanian Dependency Treebank ALKSNIS. *I. Skadia and R. Rozis (Eds.): Human Language Technologies The Baltic Perspective*.

Terje Mathiassen. 1997. *Short Grammar of Lithuanian*. Slavica Publishers, Inc., Columbus, OH.

Bronius Piesarkas. 2006. *Didysis Lietuviu-Anglu Kalb Zodynas*. Zodynas Publishers, Vilnius, Lithuania.

Bronius Piesarkas and Bronius Svecevicius. 1995. *Lithuanian Dictionary English-Lithuanian Lithuanian-English*. Zodynas Publishers, Vilnius, Lithuania.

Meilute Ramoniene and Joana Pribusauskaite. 2008. *Practical Grammar of Lithuanian*. Baltos Lankos, Lithuania.

Aurelija Usoniene, Nicole Nau, and Ineta Dabasinskiene. 2012. *Multiple Perspectives in Linguistic Research on Baltic Languages*. Cambridge Scholars Publishing, Newcastle upon Tyne, UK.

Disambiguating homonymous enclitics in Greenlandic

Liv Molich
Oqaasileriffik / The Language Secretariat of Greenland
livm@oqaasileriffik.gl

Abstract

In Greenlandic ambiguities are very common. Some of them concern enclitics which are widely used and can be interpreted as both conjunctional and adverbial particles. The disambiguation of such homographic enclitics was not attempted before 2018, even though a Greenlandic constraint grammar (CG) was initiated more than a decade ago.

In this paper I shall deal with the disambiguational challenges of enclitic particles and discuss where the disambiguation rules should be placed in the CG, and show how some disambiguation problems can be solved by looking at the combination of inflection and enclitic.

This is an important issue, because different renderings of enclitics – like many other morphemes – can change the syntax of the sentence completely.

1 Credits

The Greenlandic CG was initiated in 2008 by Per Langgård in collaboration with Eckhard Bick and Tino Didriksen. The grammar has continuously been improved, especially by Per Langgård, assisted by changing colleagues. A Greenlandic-Danish-Greenlandic machine translation project running from 2017 till the end of 2021 under the auspices of the Language Secretariat of Greenland (Oqaasileriffik 2016) has speeded up the improvement of the CG. Thanks to this work, the Greenlandic CG is now performing better than ever before, but some major issues still need attention.

2 Introduction

Greenlandic is a polysynthetic, agglutinative, split-ergative pro-drop language with a rich morphology. The level of ambiguity is high, with an average number of 3-4 readings per cohort on the morphological level alone (Oqaasileriffik 2010; Molich 2019:4). Most Greenlandic words consist of a root and several derivational morphemes followed by inflection and sometimes one or more enclitics. When the morphemes are combined, a number of morphophonological phenomena (Langgård 1997) blur the morpheme boundaries, raising the level of ambiguity and resulting in a hard job of disambiguation.

In addition to ambiguities caused by morphophonology, there are also ambiguities at the word class level: Common nouns are not distinguishable from adjectives,[1] and nominal participles are often homonymous to verbal participles. Additionally, enclitics can be interpreted as adverbs or conjunctions, but graphically they do not differ. However, a correlation between participles and enclitics can be found, facilitating the disambiguation.

3 The enclitic particles can be conjunctions or adverbs

Three enclitic particles, {luunniit}, {lu} and {li}, share some features: They can be added to any word and may function as conjunctions or adverbs, without changing the syntactic potential or word class of the word they are added to. If the enclitic particle is conjunctional, it coordinates the phrase itself with the phrase to the left of it – disjunctively, additively, or adversatively. If the enclitic is adverbial, it only modifies the clause that it is part of.

The adverbial enclitics often lead to translations

[1] For this reason common nouns and adjectives are treated as belonging to the same word class, nouns.

with subordinate conjunctions if they are added to verbs in the participle or contemporative mood. Such enclitics may therefore be equated to conjunctions, even though they are adverbial seen from a purely Greenlandic point of view.

3.1 An example: {**luunniit**}

The following quote[2] is an example of adverbial and conjunctional use of {luunniit} and shows that disambiguation between the two functions is needed:

EX *Maleruagassa-t*
rule-N.Abs.P[3]
@SUBJ>

*suminngaanneer-aluar-aanni-**luunniit***
be.from.where-even-V.Par.Im-even.Encl.Adv
@CL-ADVL>

atuup-put,
be.in.use-V.Ind.3P,
@PRED

franski-u-gaanni
Frenchman-be-V.Par.Im
@CL-<ADVL

tyrkeri-u-gaanni
Turk-be-V.Par.Im
@CL-<ADVL

*kalaali-u-gaanni-**luunniit***.
Greenlander-be-V.Par.Im-or.Encl.Conj
@CL-<ADVL

> "The rules apply **no matter** where you are from, [no matter] if you are a Frenchman, a Turk **or** a Greenlander."

Both {luunniit} enclitics are added to an impersonal participle *-gaanni*, "you" or "one", but they clearly have different functions, the first being adverbial "**no matter** where you are from" and the second conjunctional, "**or** if you are a Greenlander". Formerly, both {luunniit}s were translated by "even though" (*selvom* in Danish). This is clearly not what we intend, especially not where {luunniit} should be translated as a coordinating conjunction.

[2] KNR 2005. The example has been shortened.

[3] N=noun; Abs=absolutive case; P=plural; V=verb; Ind=indicative mood; Par=verbal participle; Im=impersonal; 3=3rd person; Encl=enclitic particle; Adv=adverbial; Conj=conjunctional.

3.2 Writing disambiguational rules for {**luunniit**}

As the example shows, the syntactic function of the enclitic cannot be deduced from morphology alone, and syntactic rules must therefore be written in order to be able to do the disambiguation. The syntactic function of the enclitic is shown by adding a secondary tag to the word, *Gram/Adv-encl* or *Gram/Conj-encl*. These secondary tags facilitate correct translations of the enclitics into other languages.

If conjunctional, {luunniit} must coordinate two similar phrases. In the following rule, used in the example above, the secondary tag *Gram/Conj-encl* is added to the enclitic, if a verb in the same mood appears to the left of the enclitic, as well as to the right of or in the same word as the enclitic.

```
SUBSTITUTE:conj1luun
   (LUUNNIIT) (LUUNNIIT Gram/Conj-encl)
   TARGET LUUNNIIT + $$MOOD
   IF (-*1 $$MOOD - Gram/Exclm
   BARRIER (*) - KOMMA) ;
```

3.3 Placing the rules for {**luunniit**}

Rules such as the above one work well, but often word class disambiguation prior to enclitic disambiguation could be advantageous. The most logical thing to do is therefore to place the rules for {luunniit} after the word class disambiguation rules. However, this is problematic because the disambiguation of enclitics is useful for disambiguation of word class, and word class disambiguation is useful for the disambiguation of enclitics. Both wishes of course cannot be met at the same time.

Disambiguation of nominal and verbal participle is not done until late in the grammar because of the complexity of the task – as shown later. Because of these and other unsolved ambiguities, I have decided to place the enclitic disambiguation rules in the top of the grammar, rather than in the bottom.

Not being able to disambiguate word class before enclitic, I have decided to divide the rules for the conjunctional and adverbial enclitics into groups of safe, less safe and unsafe rules and arrange them by turns: The group of safe conjunctional, safe adverbial, less safe conjuctional and so on. The outcome of this strategy is a reliable disambiguation, as shown in the evaluation below.

3.4 Running the rules for {luunniit}

The Greenlandic CG is run twice, the first time only using the disambiguation rules and the second time using disambiguation rules as well as mapping rules.[4] This feature is useful for proofreading the enclitic disambiguation rules: If the secondary tag in the first run is different from the one in the second run, the rules might need an adjustment, or the difference may point to a true ambiguity.

At last, the secondary tag is used for the translation of the enclitic. In cases of true ambiguity or enclitics that for some reason have not been targeted by any of the disambiguation rules, the disambiguation must be performed in a later grammar or directly in the translation lexicon rules.

4 An inflectional suffix homonymous to a derivational morpheme

While the impersonal participle *-gaanni* in the example above can be ambiguous, a much more serious problem is represented by the personal participle written *-toq* or *-soq*, identical to a derivational morpheme, the nominal participle {tuq}.[5]

4.1 An example of *-toq*

The various possibilities of interpreting words ending in the ambiguous *-toq* call for radically different syntactic analyses. A word like *atuartoq* can mean either "that he[6] is[7] reading" or "a pupil":

EX *Atuar-**toq***	*taku-ara*
read-Nzr.N.Abs.S	see-V.Ind.1S.3SO[8]
The pupil	I saw him
@OBJ>	@PRED

"I saw the pupil."

EX *Atuar-**toq***	*taku-ara*
read-V.Par.3Sg	see-V.Ind.1S.3SO[9]
That he was reading	I saw it
@CL->CIT	@PRED

"I saw that he was reading / I saw him read."

Both interpretations are equally possible and depend on the context alone. Therefore, in cases such as this one, the choice between nominal and verbal participle should only be made with context taken into account.

4.2 Combination of participle *-toq* and enclitic {lu}

Fortunately, real ambiguities are rare. In some cases the ambiguities can be solved by looking at the combination of inflection and enclitic:

EX *Taama*	*oqar-**tor-lu***
That.Part	say-V.Par.3S-when.Encl.Adv[10]
That	when he said
@ADVL>	@ADVL>

paatsiveerup-punga
become.confused-V.Ind.1S
I became confused
@PRED

"When he said that, I became confused."

Here, the adverbial, enclitic particle {lu} and the verbal participle *-toq* are used in combination to show that the first two words form a temporal subordinate clause, "**when** he said that", which is not coordinated, and should therefore not be translated as *"**and when** he said that".

This combination of enclitic and participle can be written in a simple rule:

```
SUBSTITUTE:Adv02lux
  (LU) (LU Gram/Adv-encl)
  TARGET LU + Par
  IF (NEGATE *-1 V) ;
```

Here, {lu} is marked adverbially if it is added to a verbal participle, and if no other verb is found to the left of it.

In this way, the word class can sometimes be determined by looking at the combination of enclitic and inflection.

5 Evaluation

The disambiguation of enclitic {luunniit}, {lu} and {li} is in most cases done in the first run of

[4] According to Tino Didriksen, 97,5% of the disambiguation is done in the first run, and most of the remaining disambiguation is done on the basis of the syntactic tags. The grammar could therefore easily have been split into a disambiguating grammar and a mapping grammar (personal communication, August 2019).

[5] *u* and *o* are orthographic variations of the phonemic vowel /u/.

[6] In Greenlandic, gender is not a grammatical category. Here and later, "he" could just as well have been "she" or "it".

[7] Present and past tense is normally not marked morphologically, so depending on the context, it could be translated as "was".

[8] Nzr=nominalizer; N=noun; Abs=absolutive case; S=singular; V=verb; Ind=indicative mood; 1=1st person; 3=3rd person; O=object.

[9] V=verb; Ind=indicative moor; Par=participle mood; 1=1st person; 3=3rd person; S=singular; O=object.

[10] Part=particle; V=verb; Ind=indicative mood; Par=participle mood; 1=1st person; 3=3rd person; S=singular; Encl=enclitic particle; Adv=adverbial.

the grammar. In a minor test corpus (Lynge 1976) of 16,400 tokens in 1,951 sentences, 1,395 of the tokens contained one of the three enclitics – almost one per sentence. Among these were 118 {luunniit}s.

Disambiguation of enclitic {luunniit}			
Total	**Correct**	**Incorrect**	**Not disamb.**
118	98	3	17

Of the 17 {luunniit}s that could not be disambiguated after the implementation of my rules, 7 had no morphological analysis at all, and 2 lacked a morphological analysis of the next word in the sentence concerned. 2 were differently marked in the first and the second run of the grammar, and 1 was not disambiguated until the second run. I expect that the rate of correctly disambiguated enclitics will increase when completely analyzed text becomes available.

The effectiveness of the {luunniit} rules in the top of the grammar was tested by the use of a minimal grammar only consisting of {luunniit} rules. The tagging was here very similar to that of the full grammar, but of course without the word disambiguation. This shows that the internal rule order is effective, and that the potential advantage of running the grammar twice is minimal.

As regards the correlation between enclitic {luunniit} and participle, the table below clearly shows that the combination of nominal participle and adverbial enclitic is the least probable one.[11]

Correlation between participle and enclitic		
	Nominal	**Verbal**
Conjunctional	611	456
Adverbial	318	429

This knowledge can be used for disambiguating the last enclitics and/or participles, on a statistical basis.

6 Conclusion

Disambiguation problems abound in Greenlandic, and they sometimes call for creative thinking. Some of the problems can be solved by distinguishing between adverbial and conjunctional enclitics. Disambiguation of word class and disambiguation of enclitic are interdependent, and when the disambiguation rules for enclitics are put in the right order, reliable disambiguations can be achieved, resulting in better disambiguation of participles, and more correct translations.

[11] In Oqaasileriffik's big corpus with more than 13,000,000 words, 2,614 words contain the combination of participle and {luunniit}. Not all enclitics were analyzed or disambiguated so the numbers in the table do not sum up to the total.

Acknowledgements

My dear husband Flemming A. J. Nielsen deserves big thanks for correcting linguistic errors, and for always cheering me up.

References

Kleist, Mira, Juana Petrussen and Carla Rosing Olsen. 2017. *Ataqqinartuaraq.* Nuuk: Milik Publishing. Translated from Antoine de Saint-Exupéry. 1943. *Le Petit Prince.* New York: Reynal & Hitchcock.

KNR (Kalaallit Nunaata Radioa). 2005. *Birthe Rønn Hornbech: Siunnersuut akuersaarneqarsinnaanngilluinnarpoq*, https://knr.gl/kl/nutaarsiassat/birthe-r%C3%B8nn-hornbech-siunnersuut-akuersaarneqarsinnaanngilluinnarpoq.

Langgård, Per. 1997. *Forsøg til en Forbedret Grønlandsk Pædagogisk Grammatica.* Nuuk: Atuagkat.

Lynge, Hans Anton. 1976. *Seqajuk*. Atuakkiorfik.

Molich, Liv. 2019. *Solving translational problems through constraint grammar - a practical case study of possession homographs.* Bachelor thesis. Nuuk: University of Greenland. https://uni.gl/media/4822434/solving-translational-problems-through-constraint-grammar-a-practical-case-study-of-possession-homographs-official.pdf.

Oqaasileriffik. 2010. *A bit of History.* https://oqaasileriffik.gl/langtech/a-bit-of-history/.

Oqaasileriffik. 2016. *Ukiumoortumik nalunaarut 2016-imoortoq.* https://oqaasileriffik.gl/wp-content/uploads/2018/08/2016-Oqaasileriffik-final.pdf.

CG Roots of UD Treebank of Estonian Web Language

Kadri Muischnek
University of Tartu
Estonia
kadri.muischnek@ut.ee

Kaili Müürisep
University of Tartu
Estonia
kaili.muurisep@ut.ee

Dage Särg
University Of Tartu
Estonia
dage.sarg@ut.ee

Abstract

This paper describes a method building UD Treebank of Estonian Web Language from scratch. First, the texts were parsed using Estonian CG parser and the parser output was manually checked by two human annotators. After that, the CG annotations were converted into UD annotations by means of CG rules and external scripts. Apart from providing a detailed overview of this method, the paper also discusses benefits and limitations of this approach.

1 Introduction

This contribution reports on a project of building a preliminary version of the UD Treebank of Estonian Web Language (EWTB) and lessons learnt in the course of this effort.

Universal Dependencies (UD) is an open community effort to create cross-linguistically consistent treebank annotation for many languages within a dependency-based lexicalist framework (Nivre et al., 2016).

As the Estonian UD Treebank (EDTB) has been part of the UD treebank collection since its Version 1.2 (Muischnek et al., 2014b), the corpus of web language has been included since Version 2.4. The main Estonian UD Treebank contains 30,723 trees, 434,245 tokens. EDTBs texts represent the "classical" genres of written language: fiction, newspaper and scientific texts. EWTB (1660 trees, 27,000 tokens) includes a small sample of texts from the corpus Estonian Web 2013.

The main aim of the UD effort is to facilitate developing better parsing techniques and better parsers. By "better" one also bears in mind better coverage of texts that are "out there" and need to be parsed for practical purposes. These texts include also the user-generated internet content containing a large variety of genres, differing from the normed language usage of the "classical" texts and also from each other in orthography, lexicon and even in the preferred syntactic structures. So we are extending the coverage of Estonian UD and as a pilot project we have annotated a small collection of web texts and published it as a UD Treebank of Estonian Web Language (EWTB) in UD Version 2.4.

In the UD repository different internet genres (blogs, web, social, reviews) are distinguished. Of those, EWTB contains blogs, social (forum posts) and other web texts, but no reviews.

2 UD Treebank of Estonian Web Language (EWTB)

EWTB includes a small sample of texts from the corpus Estonian Web 2013[1]. Estonian Web 2013 belongs to the so-called Ten-Ten corpus family. The texts have been crawled from the web, cleaned from non-textual material, tokenized and analysed morphologically (lemmatized). The same tools were used for tokenizing and lemmatizing classical written texts and more informal web texts, so the quality of the original morphological analysis was not reliable. Thus we preserved the tokenization but created new morphological annotation, including lemmas.

The creation of EWTB proceeded in two steps. First, the texts were annotated using the Estonian Constraint Grammar annotation scheme for morphological analysis and dependency parsing (Muischnek et al., 2014a). The annotation standard was the same as used for annotating the Estonian Dependency Treebank (Muischnek et al., 2014b), but one additional syntactic label has been introduced, namely that of discourse particle. The initial annotations were created using the Constraint Grammar parser for Estonian and the parser output was manually checked by two human anno-

[1] DOI:10.15155/1-00-0000-0000-0000-0011FL

tators. The preliminary Constraint Grammar style treebank of web texts is described by Särg et al. (2018) and is freely available[2].

3 The conversion procedure

The CG annotations were converted into UD annotations by means of Constraint Grammar rules. The conversion rules and conversion process are discussed in detail in Muischnek et al. (2016). Resulting UD annotations were again manually checked, but this time by one person. Also, several consistency checks were made using the Udapi tool (Popel et al., 2017).

Such a procedure - creating UD treebank by converting Constraint Grammar annotations into UD annotations - has also been used while creating the North Sámi UD treebank (Sheyanova and Tyers, 2017).

The annotation scheme of UD has been enhanced on each release, as well as the developers of the corpora are becoming more and more demanding for the correctness and consistency of the annotation.

3.1 Clausal dependencies

Estonian CG annotation scheme is quite fine-grained for annotating intra-clausal phenomena, and thus the transfer of annotation is not very complicated inside the clause. But although we annotate the dependency relations that hold between the clauses, our scheme does not distinguish the names of those relations, and the annotation only shows that there is a dependency relation between the clauses. That should be considered one of the main shortcomings of our CG annotation scheme. The heads of subclauses have been annotated by label *dep* and then corrected manually. Figure 1 illustrates the sentence (1) in the CG scheme and its correct syntactic counterparts in the UD schema are presented in Figure 2 (the column of morphological features is omitted). The label of the 5th token has been corrected manually.

(1) *Usutakse* *et* *puud* *suudavad*
believe-IMPRS that trees can
talletada *piisavalt* *CO*$_2$.
strore enough CO_2 .

It is believed that trees can store enough CO_2.

```
"<Usutakse>"
   "usku" Ltakse V main indic pres imps af @FMV #1->0
"<,>"
   "," Z Com CLB #2->2
"<et>"
   "et" L0 J sub @J #3->5
"<puud>"
   "puu" Ld S com pl nom @SUBJ #4->5
"<suudavad>"
   "suut" Lvad V main indic pres ps3 pl ps af @FMV #5->1
"<talletada>"
   "talleta" Lda V main inf @OBJ #6->5
"<piisavalt>"
   "piisavalt" L0 D @ADVL #7->6
"<CO2>"
   "CO2" L0 Y nominal ? @OBJ #8->6
```

Figure 1: CG annotation of the sentence.

1 Usutakse	usku	VERB	V	...	0	root	_	_
2 ,	,	PUNCT	Z	_	5	punct	_	_
3 et	et	SCONJ	J	_	5	mark	_	_
4 puud	puu	NOUN	S	...	5	nsubj	_	_
5 suudavad	suut	VERB	V	...	1	ccomp	_	_
6 talletada	talleta	VERB	V	...	5	ccomp	_	_
7 piisavalt	piisavalt	ADV	D	_	6	advmod	_	_
8 CO2	CO2	SYM	Y	...	6	obj	_	_

Figure 2: UD annotation of the sentence.

When converting the new corpus from CG to UD, we found that in addition to known problems in determining the function of clauses, it was also necessary to check determiners, names, copulas, elliptical constructions etc.

3.2 Determiners

Estonian CG annotation employs only pronoun part-of-speech, while UD also uses determiners. Although the transfer is mostly straightforward and lexicon based, there are some cases which only human could solve. In example (2), word-form *nende* can be determiner *these* or modifier *their*.

(2) *nende* *hindade* *puudumisel* ...
this-PL-GEN price-PL-GEN missing ...
they-PL-GEN price-PL-GEN missing ...

Missing of these/their prices ...

3.3 Names and appositions

The annotation of names and appositions is different in CG and UD. The leftmost part of a multi-word name is the head in UD while Estonian CG

[2] https://github.com/EstSyntax/EDT

annotates the last part of a multi-word name as the head. As for appositions, Estonian CG annotation scheme treats them as attributes. So, in example (3), the head of the name phrase is the rightmost node (comitative case of *Malouli*) in the CG annotation, and the leftmost (*finalist*) in UD.

(3) *Lahing Nordecon Openi finalisti*
battle Nordecon Open-GEN finalist-GEN
Laurent Malouliga
Laurent Malouli-COM

Battle with Laurent Malouli, the finalist of Nordecon Open

3.4 Copular constructions

The annotation of copula clauses is different in Estonian CG. Also, the definition of copula clause is wider as it is in CG and the straightforward rule-based conversion is not possible (Muischnek and Müürisep, 2017). CG annotation considers verb *be* as a root of the sentence (4), while it is a copula in the UD annotation. Figure 3 illustrates the differences of trees.

(4) *Homasho õige koht on*
Homasho-GEN right place be-3.SG
Goeido järel
Goeido-GEN after

Homasho's right place is after Goeido.

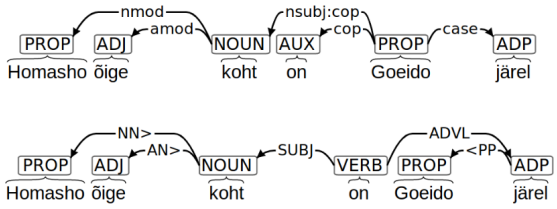

Figure 3: Copular constructions.

3.5 Elliptical constructions

CG-based treebank does not have any special label for ellipsis or orphan nodes. This annotation has been included into UD by special rule-based detector that can recognize some elliptical clauses but not all. As atypical elliptical clauses are quite frequent in the corpus of web language, they needed manual reannotation. Empty nodes have been included into UD syntax trees and the whole clause has an extra annotation of enhanced dependencies. In the sentence (5), the verb *pay* is omitted in the coordinated clause. Enhanced dependencies are marked as dotted arcs in figure 4.

(5) *ja siis jälle maksab mees ja siis*
and then again pay-3.SG man and then
jälle mina jne.
again I etc.

And then the man (husband) will pay and then I (will pay) again etc.

Figures 5 and 6 illustrate the format of CG and UD annotation of the EWTB sentence (5).

```
"<Ja>"
    "ja" L0 J crd CLB @J #1->4
"<siis>"
    "siis" L0 D @ADVL #2->4
"<jälle>"
    "jälle" L0 D @ADVL #3->4
"<maksab>"
    "maks" Lb V main indic pres ps3 sg ps af @FMV #4->0
"<mees>"
    "mees" L0 S com sg nom @SUBJ #5->4
"<ja>"
    "ja" L0 J crd CLB @J #6->9
"<siis>"
    "siis" L0 D @ADVL #7->9
"<jälle>"
    "jälle" L0 D @ADVL #8->9
"<mina>"
    "mina" L0 P pers ps1 sg nom @SUBJ #9->4
"<jne>"
    "jne" L0 Y adverbial @ADVL #10->9
"<.>"
    "." Z Fst #11->11
```

Figure 5: CG annotation of the sentence.

4 Future plans and conclusion

The conversion rule set consists of approximately 1000 rules which transfer texts from CG format to UD. Some conversion steps need human knowledge and their rule-based automation is impossible (or hard). As for future research, we plan to increase the treebank and improve it by adding coreference annotation.

Acknowledgments

This study was supported by the Estonian Ministry of Education and Research (IUT20-56), and by the European Union through the European Regional Development Fund (Centre of Excellence in Estonian Studies).

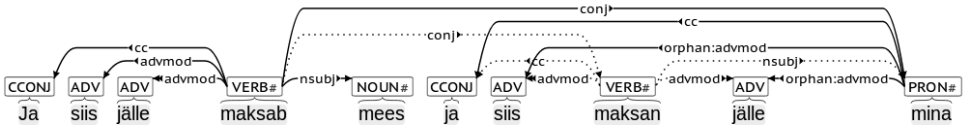

Figure 4: Elliptical constructions.

```
# sent_id = ewtb2_149414_34
# text = Ja siis jälle maksab mees ja siis jälle mina jne.
1    Ja      ja      CCONJ  J  _                                                                    4  cc           4:cc           _
2    siis    siis    ADV    D  _                                                                    4  advmod       4:advmod       _
3    jälle   jälle   ADV    D  _                                                                    4  advmod       4:advmod       _
4    maksab  maksma  VERB   V  Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin|Voice=Act     0  root         0:root         _
5    mees    mees    NOUN   S  Case=Nom|Number=Sing                                                 4  nsubj        4:nsubj        _
6    ja      ja      CCONJ  J  _                                                                    9  cc           7.1:cc         _
7    siis    siis    ADV    D  _                                                                    9  orphan:advmod 7.1:advmod    _
7.1  maksan  maksma  VERB   V  Mood=Ind|Number=Sing|Person=1|Tense=Pres|VerbForm=Fin|Voice=Act     _  _            4:conj         _
8    jälle   jälle   ADV    D  _                                                                    9  orphan:advmod 7.1:advmod    _
9    mina    mina    PRON   P  Case=Nom|Number=Sing|Person=1|PronType=Prs                          4  conj         7.1:nsubj      _
10   jne     jne     ADV    Y  Abbr=Yes                                                            4  conj         4:conj         SpaceAfter=No
11   .       .       PUNCT  Z  _                                                                    4  punct        4:punct        _
```

Figure 6: UD annotation of the sentence.

References

Kadri Muischnek and Kaili Müürisep. 2017. Estonian copular and existential constructions as an UD annotation problem. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies*, pages 79–85. Linköping University Electronic Press.

Kadri Muischnek, Kaili Müürisep, and Tiina Puolakainen. 2014a. Dependency Parsing of Estonian: Statistical and Rule-based Approaches. In *Baltic HLT*, volume 268 of *Frontiers in Artificial Intelligence and Applications*, pages 111–118. IOS Press.

Kadri Muischnek, Kaili Müürisep, and Tiina Puolakainen. 2016. Estonian Dependency Treebank: from Constraint Grammar Tagset to Universal Dependencies. In *Proc. of LREC 2016*.

Kadri Muischnek, Kaili Müürisep, Tiina Puolakainen, Eleri Aedmaa, Riin Kirt, and Dage Särg. 2014b. Estonian Dependency Treebank and its annotation scheme. In *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 285–291. University of Tübingen.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan T. McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *LREC*. European Language Resources Association (ELRA).

Martin Popel, Zdeněk Žabokrtský, and Martin Vojtek. 2017. Udapi: Universal API for universal dependencies. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*. Linköping University Electronic Press.

Dage Särg, Kadri Muischnek, and Kaili Müürisep. 2018. Annotated Clause Boundaries' Influence on Parsing Results. In *Proceedings: 21st International Conference on Text, Speech and Dialogue*.

Mariya Sheyanova and Francis M. Tyers. 2017. Annotation schemes in North Sámi dependency parsing. In *Proceedings of the 3rd International Workshop for Computational Linguistics of Uralic Languages*, pages 66–75.

Modelling Plains Cree Negation with Constraint Grammar

Katherine Schmirler
University of Alberta
Department of Linguistics
schmirle@ualberta.ca

Antti Arppe
University of Alberta
Department of Linguistics
arppe@ualberta.ca

Abstract

This paper explores negation in a Plains Cree corpus, using Constraint Grammar to model various aspects of negation (verbal, nominal, particle). Plains Cree, an Algonquian language of North America, displays rich morphological marking on nouns and verbs, but also makes use of a large class of indeclinable particles, including negative markers. Combined with non-configurational syntax and few strict word order patterns, modelling syntactic relationships involving particles is far from straightforward. Using previous grammatical descriptions and corpus observations, we describe the process of modelling relationships involving negative particles in Plains Cree, present the patterns that emerge, and identify issues for further modelling.

1 Introduction

Previous work on Plains Cree syntactic modelling has aimed to identify basic syntactic relationships between verbs and core arguments (e.g. Schmirler et al., 2018), which can, for the most part, be identified on the basis of previously marked lexical features (i.e. noun class or verb class) and morphological features output by a morphological model (cf. Snoek et al., 2014; Harrigan et al., 2017). For further development, considerations beyond simple morphological features need to be made. In the present work, we detail the ongoing process of modelling relationships involving Plains Cree negative particles.

2 Background

2.1 Plains Cree

Plains Cree is a member of the Algonquian language family, which ranges across much of North America, from the Blackfoot and Cree languages spoken as far west as Alberta, Canada and Montana, USA, to a number of languages spoken on the eastern coast of the continent. Plains Cree is the westernmost member of the Cree language continuum, which includes Cree dialects across Alberta, Saskatchewan, northern Montana, northern Manitoba, and northern Ontario, as well as the closely related Montagnais-Naskapi dialects, spoken in Quebec and Labrador (Wolvengrey, 2011).

Algonquian languages are known for their rich morphology and non-configurational syntax, such that syntactic roles are determined by morphological agreement rather than word order. Of their various typological features, the noun classification system of animacy and the hierarchical alignment system are of particular interest for syntactic modelling. *Animacy* refers to the noun classification or gender system in Plains Cree, which divides nouns into two classes: *animate* and *inanimate*. Though this system corresponds closely to conceptual animacy (i.e., all humans, animals, and trees are animate, while most other objects are inanimate), many nouns demonstrate that it is a grammatical classification: for example, *asikan* ‘sock’ and *ayôskan* ‘raspberry’ are animate, while *maskisin* ‘shoe’ and *otêhimin* ‘strawberry’ are inanimate.

The animacy of nouns also bears on the types of verbs with which they can occur. The verbal system of Plains Cree includes four classes of verbs, determined by their transitivity and the animacy of the nouns they license. Thus, intransitive and transitive verbs each have two subclasses for each animate and inanimate nouns. Inanimate intransitive verbs (VII) take one inanimate participant and include attributive verbs for describing inanimate

nouns and zero-place predicates such as weather terms. Animate intransitive verbs (VAI) take one animate participant and include intransitive actions and attributive verbs for animate nouns. The terminology for transitive verbs differs slightly; it is assumed that the actor[1] is animate, and so the classification refers to the animacy of the goal. Transitive inanimate verbs (VTI) take an animate actor and an inanimate goal. Transitive animate verbs (VTA) take two animate participants. Examples of different classes are given in (1) and (2); note the pairs of verbs for nouns of differing animacy. Detailed morphological breakdowns are presented in these examples to demonstrate the rich morphology of Plains Cree, but are not included in later examples, as morphology is not the focus of the present work.[2]

(1) Intransitive clauses

a. VII
otêhimin mîhkwâw
otêhimin mîhkwâ- -w
strawberry.N.IN be.red.VII 0SG
'the strawberry is red'

b. VAI
ayôskan mîhkosiw
ayôskan mîhkosi- -w
raspberry.N.AN be.red.VAI 3SG
'the raspberry is red'

(2) Transitive clauses

a. VTI
wâpahtam maskisin
wâpaht- -am maskisin
see.VTI 3SG shoe.N.IN
'she/he/it (animate) sees a shoe'

b. VTA
niwâpamâw asikan
ni- wâpam- -âw asikan
1 see.VTA 1/2SG>3SG sock.N.AN
'I see a sock'

Alongside the transitivity/animacy classes of verbs, we also briefly introduce verbal orders, or inflectional patterns with different semantic and syntactic functions. Wolfart (1973) describes three orders for Plains Cree: independent, conjunct, and imperative. As is frequent in other descriptions of Plains Cree, our morphological model also identifies a subclass of the conjunct order, the future conditional, so we include this as a fourth order herein. The functions of the orders are summarised briefly here. The independent order is generally used for matrix clauses while the conjunct order can be used for either matrix or subordinate clauses. Cook (2014) identifies these as indexical and non-indexical clauses respectively: in very simple terms, independent clauses require no prior knowledge or context, while conjunct clauses, if they occur as matrix clauses, are not syntactically embedded but instead pragmatically "embedded" in an established context (in the real world or established within a discourse). Future conditional verbs are generally subordinate clauses, as they occur with meanings such as 'if, when, whenever'. Finally, imperative verbs may be either immediate ("do action now") or delayed ("do action later") and are not considered subordinate clauses. With the exception of imperatives, which do not occur for VIIs, all verb classes can occur in all verbal orders.

The rich agreement morphology allows for straightforward modelling of core argument relationships while particles, the most frequent word class evidenced in texts, bear essentially no inflectional morphology, and include words with a variety of different functions, which have not yet been given a detailed classification for Plains Cree. Without such a classification, development of particle constraints in the Plains Cree parser is an ongoing process, such as that described for negative particles in section 3.

2.2 A Plains Cree corpus

The texts to which the Plains Cree CG parser is applied are referred to herein as the Ahenakew-Wolfart (A-W) corpus (Arppe et al., in press). The A-W corpus consists of several texts (totalling ∼73,000 words of Plains Cree) collected in the 1980s and 1990s, which have been transcribed, edited, and in some cases translated, then published in several volumes (Ahenakew, 2000; Bear et al., 1998; Kâ-Nîpitêhtêw, 1998; Masuskapoe, 2010; Minde, 1997; Vandall and Douquette, 1987; Whitecalf et al., 1993). Digital versions, which

[1] In accordance with Algonquianist tradition, we use actor and goal to label syntactic roles in place of the more common subject and object (Bloomfield, 1946).

[2] Grammatical abbreviations: N = noun, IN = inanimate, AN = animate, VII = inanimate intransitive verb, VAI = animate intransitive verb, VTI = transitive inanimate verb, VTA = transitive animate verb, 0SG = singular inanimate agreement, 3SG = third person singular animate, 1 = first person prefix, 1/2SG>3SG = first or second person singular acting on third person singular animate.

display less editing than the published texts (e.g. more fragments, commas representing pauses, etc.), have been supplied for the digital corpus by H.C. Wolfart. This corpus has been morphosyntactially analysed using a finite-state parser (Snoek et al., 2014; Harrigan et al., 2017), the results of which have been hand-verified by two researchers, and subsequently tentatively disambiguated and parsed for core arguments (Schmirler et al., 2018) using CG-3 (Bick and Didriksen, 2015). The corpus is available upon request at URL: http://altlab.ualberta.ca/korp. The corpus contains a variety of genres, including historical narratives, personal narratives, funny stories, speeches/lectures, and dialogues; future research is planned to explore the ways in which genre affects morphosyntactic patterns in Plains Cree.

3 Considering negation in the Plains Cree CG parser

3.1 Negative particles

The initial implementation of negative particles in the CG parser was straightforward. First, a LIST of negative particles was created, and negative particle phrases were similarly identified; examples of particles are given in (3) and examples of phrases are given in (4).[3] These were assigned a morphological tag `Neg` for reference in later constraints and in corpus searches. The same morphemes appear repeatedly in these particles: *namôy*, *môya*, and *môy* are reduced forms of *namôya*; *kâwiya*, *êkâya*, *êkây*, *kâya*, and *êkâ* of *êkâwiya*; and *mwâc* of *namwâc*.[4] The other form in this list, *nama*, occurs only particle phrases in the A-W corpus, though older texts demonstrate that it was once a common negative particle; additionally, *namôya* is historically derived from *nama* plus a focus or emphatic particle *wiya*. Similarly, though only a subset of the negative particle phrases are given in (4), the same phrases also occur with the reduced forms of *namôya*.[5]

(3) Negative particles

```
LIST NEG = "namôya" "namôy"
"môya" "môy" "êkâwiya"
"kâwiya" "êkâya" "êkây" "kâya"
"êkâ" "nama" "namwâc" "mwâc" ;
```

(4) Negative particle phrases

`"namôya_wîhkâc" Iph Neg` ‘never’
`"namôya_ahpô" Iph Neg` ‘not even’
`"namôya_cî" Iph Neg` ‘it is not so?’

Though not relevant to the initial implementation of negation relationships in the present parser, the distribution of these different negative particle types (*namôya*-type, *êkâwiya*-type, *namwâc*-type) is referenced throughout section 4 below. Two main accounts are given for the distribution of *namôya*-type and *êkâwiya*-type negative particles.[6] The most common of these is a syntactic explanation: *namôya*-types (generally) occur with matrix clauses and *êkâwiya*-types (generally) occur with subordinate clauses. Additionally, imperatives are always negated with *êkâwiya*-type particles (e.g. Dahlstrom, 1991; Wolfart, 1996).

Further investigations have noted that this description, while an excellent starting point, does not fully capture the distribution of *namôya*-types and *êkâwiya*-types. Instead, this distribution can be explained in terms of realis and irrealis contexts. However, conveniently, we can use the morphosyntactic features of verbal order to approximate this distinction in the present work. Thus, independent verbs, as matrix clauses, are most likely realis; conjunct verbs, just as they can be either matrix or subordinate clauses, they can represent situations with either realis or irrealis semantics; future conditional verbs are subordinate clauses, but can be either realis or irrealis; and finally imperative verbs are considered always irrealis (and thus the only verbal order that is negated only by *êkâwiya*-types) (e.g. Cook, 2014).

3.2 Constraint development

After a LIST of negative particles was created, an initial constraint then assigned the function tag `@Neg>` when any of these words or phrases appeared immediately before a verb. While this does produce adequate results, it does not by any means fully capture negation in Plains Cree. First, there can be some intervening material between the negative particle or particle phrase and the verb, some

[3] The LIST approach was implemented for the initial development stage described herein; in future development, tags for particle functions can be added in the morphological model.

[4] All of these are glossed as ‘no, not’.

[5] Parser abbreviations: `Neg` = negative, `Ipc` = particle, `Iph` = particle phrase, `@Neg` = any negative tag, `@Neg-V>` = negative dependent on a verb to the right, `@Neg-N>` = negative dependent on a noun to the right, `@Neg-Ipc>` = negative dependent on a particle to the right.

[6] Little is said for Plains Cree on the *namwâc*-type particles.

of these negative particles can modify nouns or other particles as well as verbs, and some of these particles modify only particular elements. For example, variations of *êkâwiya* modify particular conjugation patterns, and *nama* only occurs in particle phrases in the A-W corpus.

A cursory exploration of the A-W corpus begins to demonstrate the complexities of negation: of the ∼1510 negative particles (excluding particle phrases), ∼580 occur immediately before verbs, which are identified by the original simple constraint. Of the remaining particles, ∼25 occur immediately before nouns, ∼100 before pronouns, ∼550 before particles, ∼160 before punctuation, and the remainder before as-yet-unlabelled elements. While an exploration of how negatives interact with all of these categories is beyond the scope of this paper, those before nouns can be explored readily. Several occur before a noun (or noun phrase) which is followed by punctuation, and so the negative particle does appear to modify the noun. In other cases, however, the noun is followed by a verb, without intervening punctuation, suggesting that the verb is negated. One sentence displays a series of negated elements, including nouns and a verb; a portion is given in (5).

(5) *...namôya nipiy,*
not water
namôya sâkahikana k-âtâmitân,
not lakes I buy from you
namôya kinosêw; ...
not fish
'...I do not buy the water, nor the lakes, from you, nor the fish...' (Kâ-Nîpitêhtêw, 1998, pp. 110-13)

In example (5), where *namôya* occurs before *nipiy* 'water' or *kinosêw* 'fish', it is straightforward to analyse each negative as dependent on the following noun. Before *sâkahikana* 'lakes', however, the verb is negated (cf. the English translation, where the verb is negated once and the remaining nouns are preceded by 'nor'). With this observation in mind, a constraint was written for nouns that makes use of clause boundaries, and the verbal constraint was modified to allow for intervening nominals.

A similar constraint was also written for particles, and the verbal constraint modified once again to allow for any intervening material, excluding clause boundaries. The current three negation constraints are given in (6). While these three constraints allow for many of the negative particles to receive a dependency tag, further examination of the corpus is required to determine their accuracy.[7]

(6) Negation constraints for Plains Cree
```
MAP:NegV @Neg-V> TARGET Neg
IF (*1 V BARRIER CLB) ;
MAP:NegN @Neg-N> TARGET Neg
IF (1 N) ;
MAP:NegIpc @Neg-Ipc> TARGET
Neg IF (1 Ipc) ;
```

This empirical approach quickly reveals issues of scope: often, a negative particle appears to negate a verb later in a sentence, rather than the immediately following noun or particle, regardless of punctuation. While these constraints can begin to give some idea of how negation works in Plains Cree, and can be used to further develop more accurate constraints, additional research into the scope of negation is also required.

4 Negation in a Plains Cree corpus

4.1 Negative particles and other word classes

The parser identifies 1732 negative particles, of which 1480 receive an `@Neg` tag (of any type, V, N, or Ipc). Of those that receive a syntactic function tag, 1249 are identified as modifying a verb, 25 as modifying a noun, and 206 as modifying a particle. Details for each type of negative particle (*namôya*-type, *êkâwiya*-type, and *namwâc*-type) are given in Table 1 (with a significant co-occurrence distribution (χ^2 (4, N = 1470) = 33.76, $p < .001$)). In Table 2,[8] the posthoc analysis of this significant distribution is presented, demonstrating the likelihood of each negative particle type to modify verbs, nouns, or particles.[9] Verbs are more likely to be modified by *êkâwiya*-types, while particles are more likely to be modified by *namôya*-types or *namwâc*-types.

For these negative particles, we can also explore how they modify different subclasses. In Table 3,

[7] As negation in Plains Cree is *symmetric* (negation does not strictly occur with other clausal changes from positive utterances, such as the addition of an auxiliary verb in English) (Miestamo, 2013), we cannot use clues from other morphosyntactic features when modelling negation.

[8] Here, '+' indicates a positive association (signficant over-co-occurrence), '-' a negative association (significant under-co-occurrence), and '0' a non-significant co-occurrence. See also Table 6.

[9] See Arppe (2008, p. 82-4), based on standardised Pearson residuals as described in e.g. Agresti (2002, p. 81).

	namôya	*êkâwiya*	*namwâc*
Total	1347	331	54
@Neg	1150	305	25
@Neg-V	945	290	14
@Neg-N	21	4	0
@Neg-Ipc	184	11	11

Table 1: Negative particles and negative syntactic tags for all word classes.

	namôya	*êkâwiya*	*namwâc*
@Neg-V	-	+	-
@Neg-N	0	0	0
@Neg-Ipc	+	-	+

Table 2: Posthoc analysis of the co-occurrences of negative particle types with verbs, nouns and particles.

the frequency of each verbal transitivity/animacy class with each negative particle type is presented. There was no significant preference for a particular negative type for any verb class (χ^2 (6, N = 1280) = 11.09, p = .086). However, as seen in Table 4, VTIs are negated nearly twice as often as other verb classes (significantly so, (χ^2 (3, N = 21332) = 112.59, $p < .001$): a pattern worth exploring in future research.

	namôya	*êkâwiya*	*namwâc*
VII	84	21	0
VAI	342	128	8
VTI	313	75	3
VTA	229	73	4

Table 3: Negative particle types and verbal transitivity classes.

Verbal order, which we use here as a (very rough) approximation of realis and irrealis, presents more readily interpretable results, as given in Table 5 (with a significant co-occurrence distribution (χ^2 (6, N = 1249) = 374.33, $p <$ .001)). We see, not surprisingly, the majority of independent verbs (matrix clause verbs, roughly equivalent to realis) are negated with *namôya*-types far more often than *êkâwiya*-types. For conjunct verbs, which may be either matrix or subordinate clauses (and either realis or irrealis), are negated $\sim$60% of the time by *namôya*-types and $\sim$39% of the time by *êkâwiya*-types. Conditional forms, which are always considered subordinate and often irrealis are most often negated by *êkâwiya*-types ($\sim$76%), though *namôya*-types are not infrequent at $\sim$24%. Finally, as described in the literature, we see that all of the negated imperative verbs in the corpus occur with *êkâwiya*-type negative particles. The posthoc analysis is presented in Table 6, confirming the above observations. As seen for verb classes, we can also explore the percentage of each order that is negated. In Table 7, independent verbs are seen to be the most frequently negated, while conjunct verbs are the least frequently negated. These patterns may suggest that matrix clauses are more likely to be negated than subordinate clauses, though further investigation into conjunct subtypes is required.

	Negated	**In corpus**	%
VII	105	1989	5.28
VAI	478	9269	5.16
VTI	391	4100	9.54
VTA	306	5974	5.12

Table 4: Negative particle types and verbal transitivity classes.

	namôya	*êkâwiya*	*namwâc*
Independent	561	4	5
Conjunct	377	244	9
Conditional	5	16	0
Imperative	0	28	0

Table 5: Negative particle types negating verbs by order.

	namôya	*êkâwiya*	*namwâc*
Independent	+	-	0
Conjunct	-	+	0
Conditional	-	+	0
Imperative	-	+	0

Table 6: Posthoc analysis of the co-occurrences of negative particle types with verbal orders.

Nominal features (both class, animate or inanimate, and number, singular or plural) are presented with respect to negative particles in Table 8. Inanimate nouns are negated more often than animate nouns, though animate nouns are more frequent in the A-W corpus. An explanation for this pattern is not immediately evident and as such further investigation is required. However, when it

	Negated	**In corpus**	%
Independent	570	6181	9.22
Conjunct	630	13995	4.50
Conditional	21	331	6.34
Imperative	28	374	7.50

Table 7: Negative particle types and verbal transitivity classes.

comes to number, singular nouns are more common than plural (though animate plural are far more common than inanimate plural), and so these results align well with overall corpus results. Still, future examination would not be amiss.

	namôya	*êkâwiya*	*namwâc*
Inanimate	12	2	0
Animate	7	1	0
Singular	17	2	0
Plural	3	1	0

Table 8: Negative particle types negating nouns by feature.

Finally, results for negated particles are given in Table 9. As these particle classifications are only rudimentary, little can be said at this time. Temporals and quantifiers are more likely to be negated than locatives; one can readily imagine phrases such as "not long ago" and "not much", which might arise from negating these subclasses. It is perhaps also worth noting that *namôya*-type particles are considerably more common for negating other particles, though the proportion of *namôya* -types compared to all particles with an `@Neg` tag in the overall corpus (∼80%) is not dissimilar to the proportion of *namôya*-types compared to all particles with an `@Neg-Ipc` tag (∼90%).

	namôya	*êkâwiya*	*namwâc*
Locatives	5	1	0
Quantifiers	14	1	1
Temporals	21	0	5
Other	141	9	0

Table 9: Negative particle types negating particles by function.

4.2 Comments on *êkâwiya* and *namwâc*

In the above results, we see that for some curious patterns, there are but few cases that can easily be explored in more detail. Two of these patterns we briefly note here, though fuller investigations are beyond the scope of the present work.

The first of these is the occurrence of *êkâwiya*-type particles negating nouns, for which only three instances occur. Two of these occur in the context of conjunct verbs, which we can interpret as irrealis; due to the clause boundaries in the text, the negative particle was simply identified as dependent on the noun. In the third case, given in (7), however, there is no morphosyntactic means of identifying an irrealis context: instead, the semantics of the particle *tapiskôc* 'as if' and the negative *êkây* must be taken into account.

(7) *kiwayawîtisahokawin*
you (sg.) are sent away
tapiskôc *êkây* *âskîhkân*
as if not reserve
'you are sent away as if it were not a reserve'
(Bear et al., 1998, pp. 300-1)

The second question is the occurrence of *namwâc*, in particular where it modifies another word rather than occurring in isolation as 'no'. The *namwâc*-types are by far the least frequent negative particle type discussed here and, unlike *namôya*- and *êkâwiya*-types, over half occur in isolation. Additionally, no information on the use of *namwâc* can be found in the literature for Plains Cree. Of those 25 that do occur with an `@Neg` tag, over half occur in clauses the bear an interesting feature: some degree of uncertainty, conveyed either through evidential particles (*êsa*, *êtikwê* 'apparently') or the combination of the negative particle and the verbs *kiskêyihtam* 's/he knows (it)' or *kiskisiw* 's/he remembers (it)'. While an investigation of these features with other negative particle types has not yet been conducted, these patterns offer a number of questions for future investigation of Plains Cree syntax and semantics.

5 Discussion

5.1 Modelling process

The straightforward modelling process outlined in section 3.2 ignores key facts of negation in Plains Cree. First, in only looking at nouns, verbs, and particles, we exclude pronouns (e.g. *môy nîsta* 'me neither'), which has certainly led to other inaccuracies in our results. Our primary reason for excluding pronouns at this time is that the third person singular pronoun *wiya* also occurs as a focus

or emphatic particle and we have yet to determine the best course of action for handling this ambiguity in the parser. Currently, a crude technique is implemented that identifies *wiya* as a pronoun when a verb with third person features occurs in the same clause and as a particle otherwise, though this is known to be inaccurate. This ambiguity has likely led to over-generation of `@Neg-Ipc>` tags, as these have been applied to *wiya* even in cases where it behaves as a pronoun. While a solution for the disambiguation of *wiya* may not be immediately apparent, constraints to identify negated pronouns will be a priority in future modelling.

Second, recent research has noted discontinuous particle phrases in Plains Cree (Wolvengrey, 2019); for example, the phrase *namôya wîhkâc* 'never' may occur with intervening material, including the verb. While we have not explored phrases in the current paper, such discontinuous phrases will have resulted in the over-application of the negative tag constraints, applying to *namôya* as though it were a lone particle rather than a member of a phrase. Future model development must determine the best course of action for identifying such phrases automatically.

These are but two issues we have identified in the process of modelling negation in Plains Cree. These issues, among general scope issues as mentioned in section 3.2 are left to future research.

5.2 Morphosyntactic vs. semantic patterns

In the results presented above, we make reference to morphosyntactic features (i.e., those referred to in the morphological and syntactic models and overtly presented in the results), but also comment on broader syntactic features such as matrix and subordinate clauses, as well as semantic features, namely the realis/irrealis distinction. While the morphosyntactic features of verbal order can represent some of the distinctions between matrix and subordinate clauses and realis and irrealis semantics, much is not captured. At minimum, other morphosyntactic features can be used to further refine the distinction; Plains Cree makes use of a class of morphemes known as *preverbs*, which occur before the verb stem within a verb and bear a number of functions (e.g. tense, adverbial, modal). Among these preverbs we find those that mark different type of conjunct clauses: *ê-* marks basic clauses, *kâ-* marks relative clauses, others bear tense and aspect information. While not all of these are well-defined, their interactions with conjunct suffixes and the realis/irrealis distinction have been investigated to some degree (e.g. Déchaine et al., 2018). Using such research as a base, we can further identify syntactic and semantic functions of conjunct verbs and their relationships with negative particles and phrases.

5.3 Future considerations

Future research is planned to explore how negative particles interact with different verbal morphology patterns beyond those discussed herein, as well as more detailed looks at nouns, particles, and pronouns. This also requires a deeper investigation of particle phrases and their functions; of particular interest are negative particle phrases with *kîkway* 'thing', as these often function as nominals. Many phrases also seem to negate clauses, and thus verbs, rather than nouns or particles—an impressionistic observation that requires further consideration.

Beyond negative particles, overall improvements and further developments in the syntactic model will also be necessary. For example, to better understand negation and the interclausal relationships and semantic patterns discussed in section 5.2 above, we will need to undertake an important yet daunting step in the syntactic model: modelling interclausal relationships, including relative clauses. Thus far, we have limited ourselves to clauses as delineated by punctuation, though interclausal and text-level relationships will be instrumental in corpus investigations of Plains Cree.

6 Conclusions

Despite their morphological simplicity, negative particles in Plains Cree have presented an interesting exercise in modelling their relationships with nouns, verbs, and other particles. The combinatory freedom of particles and flexible word order of Plains Cree present an ongoing challenge for the development of a parser. However, the identification of broad functions within the particle class, such as negation, has revealed various avenues for further modelling, and has been an important step toward more detailed and accurate syntactic function tags for Plains Cree.

7 Acknowledgements

This research was supported by the Social Sciences and Humanities Research Council of

Canada (SSHRC) through a Doctoral Fellowship (752-2017-2105) and a Partnership Grant (895-2019-1012). We also thank H.C. Wolfart for providing the digital files of the A-W corpus.

References

Alan Agresti. 2002. *Categorical data analysis*, second edition. John Wiley & Sons, Hoboken.

Alice Ahenakew. 2000. *âh-âyîtaw isi ê-kî-kiskêyihtahkik maskihkiy / They Knew Both Sides of Medicine: Cree Tales of Curing and Cursing Told by Alice Ahenakew*. University of Manitoba Press, Winnipeg.

Antti Arppe. 2008. Univariate, bivariate, and multivariate methods in corpus-based lexicography: A study of synonymy. Publications of the Department of General Linguistics, University of Helsinki, No. 44. http://urn.fi/URN:ISBN:978-952-10-5175-3.

Antti Arppe, Katherine Schmirler, Atticus G. Harrigan, and Arok Wolvengrey. in press. A morphosyntactically tagged corpus for Plains Cree. In *Papers of the 49th Algonquian Conference*.

Glecia Bear, Minne Fraser, Mary Wells, Alpha Lafond, and Rosa Longneck. 1998. *Our Grandmothers' Lives: As Told in Their Own Words*, bilingual edition edition. University of Regina Press, Regina.

Eckhard Bick and Tino Didriksen. 2015. Cg-3—beyond classical constraint grammar. In *Proceedings of the 20th nordic conference of computational linguistics, NoDaLiDa 2015, May 11-13, 2015, Vilnius, Lithuania*, 109, pages 31–39. Linköping University Electronic Press.

Leonard Bloomfield. 1946. Algonquian. In *Linguistic structures of Native America*, volume 6, pages 85–129. Viking Fund Publications in Anthropology, New York.

Clare Cook. 2014. *The Clause-Typing System of Plains Cree: Indexicality, Anaphoricity, and Contrast*. Oxford University Press, Oxford ; New York.

Amy Dahlstrom. 1991. *Plains Cree morphosyntax*. Garland Pub.,, New York :.

Rose-Marie Déchaine, Monique Dufresne, and Charlotte Reinholtz. 2018. (In)variance in the Cree dialect continuum: evidence from (IR)REALIS syntax. Paper presented at the 50th Algonquian Conference, October 27, 2018, Edmonton, AB.

Atticus G. Harrigan, Katherine Schmirler, Antti Arppe, Lene Antonsen, Trond Trosterud, and Arok Wolvengrey. 2017. Learning from the computational modelling of Plains Cree verbs. *Morphology*, 27(4):565–598.

Jim Kâ-Nîpitêhtêw. 1998. *ana kâ-pimwêwêhahk okakêskihkêmowina / The Counselling Speeches of Jim Kâ-Nîpitêhtêw*. University of Manitoba Press, Winnipeg.

Cecilia Masuskapoe. 2010. *piko kîkway ê-nakacihtât: kêkêk otâcimowina ê-nêhiyawastêki*. Algonquian and Iroquoian Linguistics, Winnipeg.

Matti Miestamo. 2013. Symmetric and asymmetric standard negation. In Matthew S. Dryer and Martin Haspelmath, editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig. https://wals.info/chapter/113.

Emma Minde. 1997. *kwayask ê-kî-pê-kiskinowâpatihicik / Their Example Showed Me the Way: A Cree Woman's Life Shaped by Two Cultures*. University of Alberta Press, Edmonton.

Katherine Schmirler, Antti Arppe, Trond Trosterud, and Lene Antonsen. 2018. Building a Constraint Grammar Parser for Plains Cree Verbs and Arguments. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Conor Snoek, Dorothy Thunder, Kaidi Lõo, Antti Arppe, Jordan Lachler, Sjur Moshagen, and Trond Trosterud. 2014. Modeling the Noun Morphology of Plains Cree. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 34–42, Baltimore, Maryland, USA. Association for Computational Linguistics.

Peter Vandall and Joe Douquette. 1987. *wâskahikaniwiyiniw-âcimowina / Stories of the House People, Told by Peter Vandall and Joe Douquette*. University of Manitoba Press, Winnipeg.

Sarah Whitecalf, H. C. Wolfart, and Freda Ahenakew. 1993. *kinêhiyawiwiniwaw nêhiyawêwin / The Cree Language is Our Identity: The La Ronge Lectures of Sarah Whitecalf*. University of Manitoba Press, Winnipeg.

H. Christoph Wolfart. 1973. *Plains Cree: A Grammatical Study*, volume 63.5 of *Transactions of the American Philosophical Society, New Series*. American Philosophical Society, Philadelphia.

H. Christoph Wolfart. 1996. Sketch of Cree, an Algonquian Language. In *Handbook of American Indians. Volume 17: Languages*, pages 390–439. Smithsonian Institute, Washington.

Arok Wolvengrey. 2019. *tānispīhk wīhkāc ōma kē-āpatahk!?*: *wīhkāc* as a Polarity Item In Plains Cree. Paper presented at the 51st Algonquian Conference, October 24-27, 2019, Montreal, QC.

Arok Elessar Wolvengrey. 2011. *Semantic and pragmatic functions in Plains Cree syntax*. Netherlands Graduate School of Linguistics.

Many shades of grammar checking – Launching a Constraint Grammar tool for North Sámi

Linda Wiechetek linda.wiechetek@uit.no
Sjur Moshagen sjur.n.moshagen@uit.no
Børre Gaup borre.gaup@uit.no
Thomas Omma thomas.omma@uit.no

Divvun UiT Norgga árktalaš universitehta

1 Introduction

This paper discusses the characteristics and evaluation of the very first North Sámi spell- and grammar checker. At its launch it supports *MS Word* and *GoogleDocs*[1], cf. Figure 1. We describe its component parts, the technology used, such as *Constraint Grammar* (Karlsson, 1990; Karlsson et al., 1995; Bick and Didriksen, 2015) and *Hfst-pmatch* (Hardwick et al., 2015), and its evaluation with a new evaluation tool specifically designed for that purpose.

Only the modules of the full-scale grammar checker described in Wiechetek (2017) that have been tested and improved sufficiently for the public to use are released in this launch. More advanced syntactic errors will be included at a later stage. The grammar checker modules are an enhancement to an existing North Sámi spellchecker (Gaup et al., 2006), following a philosophy of *release early, release often*, and using continuous integration (*ci*) and continuous delivery (*cd*) to deliver updates with new error correction types, as new parts of the grammar checker are sufficiently tested. Releasing at an early stage of development gives the user community early access to improved and much needed tools and allows the developers to improve the tools based on the community's feedback, essentially preferring early and frequent releases over feature-based releases to allow for a close feedback-based relation between developers and users.[2]

We started moving towards a "release early, release often" strategy in 2018 when working on the *Divvun installer*. The *Divvun installer* is a tool to simplify the installation and updates of the tools developed by the *Divvun* group. The users only have to install the *Divvun installer* app, and then select the languages they are interested in – everything is then installed and configured automatically. It also checks for updates to the installed tools with regular intervals, and allows us to provide updates to our users automatically. That makes it the center-piece of our release early, release often strategy.

Version	Date
Divvun 1.0	(2007-12-12)
Divvun 1.0.1	(2007-12-21)
Divvun 1.1	(2008-12-17)
Divvun 2.0	(2010-12-08)
Divvun 2.1	(2011-03-21)
Divvun 2.2	(2011-11-23)
Divvun 2.3	(2013-02-08)
Divvun 2.3	(2013-02-08)
Divvun 3.0	(2013-06-13)
Divvun 4.0	(2015-12-17)
Divvun 4.0.1	(2016-03-17)
Divvun 4.1	(2016-12-15)
Divvun 4.2	(2018-12-20)
GramDivvun 1.0 beta	**(2019-09-27)**

Table 1: *Divvun* release history (2007-2019)

This is a change of strategy based on previous experience with the feature-based release strategy, which presupposed extensive manual testing to avoid regressions. This leads to long release cycles and up to two years between the releases as can be seen in table 1. It also leads to less word coverage for the users while the system had already access to a larger lexicon.

Over the years, we have introduced automatic testing methods to maintain quality and avoid regressions. The automatic tests are integrated into our CI/CD system and cover errors we have experienced earlier, and as we find new, uncovered

[1] https://gsuite.google.com/marketplace/app/divvun_grammar_checker/611280167256

[2] https://medium.com/@warren2lynch/scrum-philosophy-release-early-release-often-a5b864fd62a8

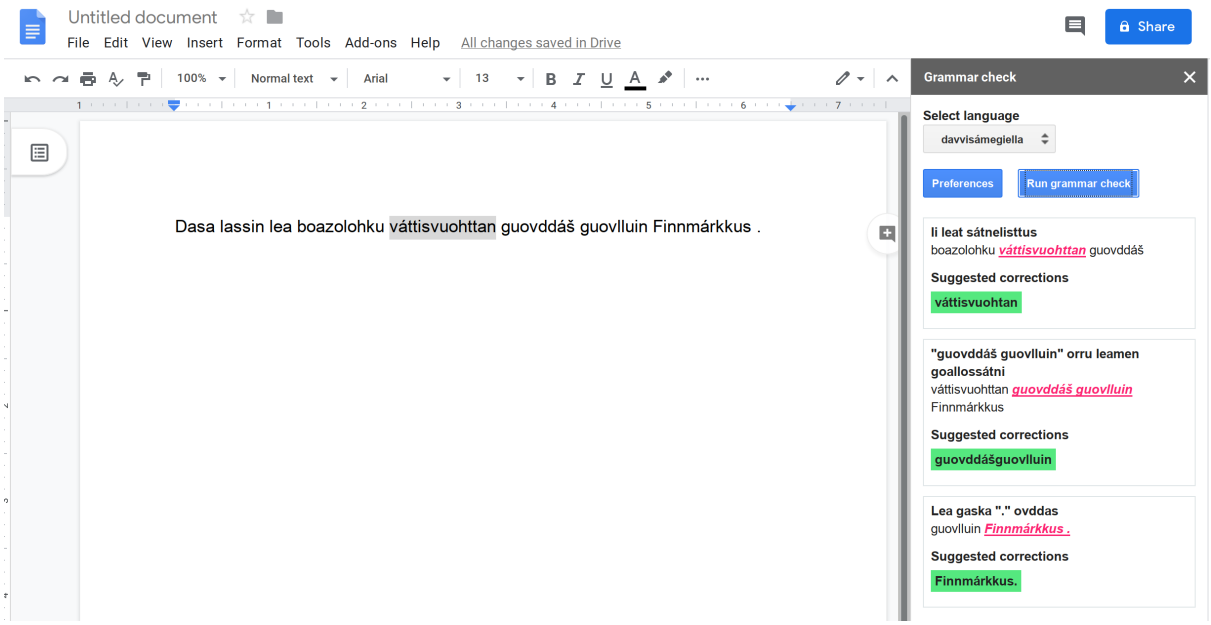

Figure 1: *GramDivvun 1.0 beta* in GoogleDocs

areas, we add tests for these as well. Automatic CI/CD means that if new commits do not break our tests, a new version of our software is built and released automatically. If tests fail, the nightly releases will not be built before the tests pass again. This assures us that the nightly releases will not contain regressions compared to earlier releases, and yet incorporates new words and features.

North Sámi is a Uralic language spoken in Norway, Sweden and Finland by approximately 25 700 speakers (Simons and Fennig, 2018). These countries have other majority languages, making all Sámi speakers bilingual. Bilingual users frequently face bigger challenges regarding literacy in the lesser used language than in the majority language due to reduced access to language arenas (Outakoski, 2013; Lindgren et al., 2016). Therefore, tools that support literacy, like spelling and grammar checkers, are more important in a minority language community than in a majority language community.

The released grammar checker is a light grammar checker in the sense that it detects and corrects errors that do not require rearranging the whole sentence, but typically just one or several adjacent word forms based on a grammatical analysis of the sentence. Additionally, a number of formatting errors are covered. The main new features are implemented by means of several Constraint Grammar-based modules. These include correction of formatting and punctuation errors, filtering of speller suggestions, much improved tokenisation and sentence boundary detection, as well as advanced compound error analysis and correction.

This paper shows how a finite-state based spellchecker can be upgraded to a much more powerful spelling and grammar checking tool by adding several Constraint Grammar modules.

2 Framework

2.1 Language tools

An open-source spelling checker for North Sámi has been freely distributed since 2007[3], the beginnings of which have been described by Gaup et al. (2006). The tool discussed in this paper includes the open-source spelling checker referenced above, but further developed and using the hfst-based spelling mechanism described in Pirinen and Lindén (2014). The spelling checker is enhanced with five Constraint Grammar modules, cf. Figure 2. It should be noted that the spelling checker is exactly the same as the regular North Sámi spelling checker used by the language community, but with the added functionality that all suggestions are returned to the pipeline with their full morphological analysis. The analyses are then used by subsequent Constraint Grammar rules during disambiguation and suggestion filtering.

[3] http://divvun.no/korrektur/korrektur.html

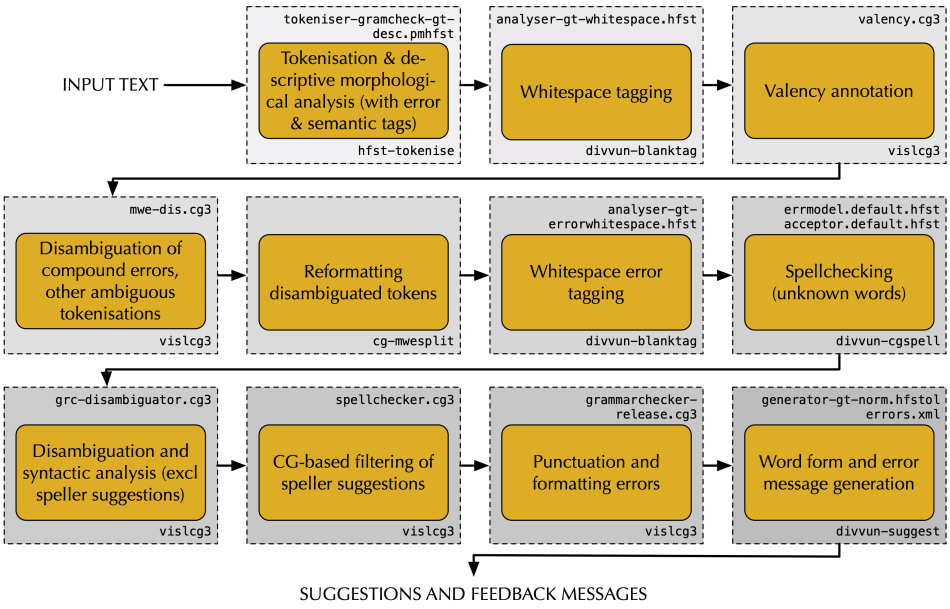

Figure 2: System architecture of *GramDivvun 1.0 beta*, the North Sámi grammar checker

All components are compiled and built using the *GiellaLT* infrastructure (Moshagen et al., 2013). Five constraint grammar modules, i.e. a valency grammar (*valency.cg3*), a tokenizer (*mwe-dis.cg3*), a morpho-syntactic disambiguator (*grc-disambiguator.cg3*), a disambiguation module for spellchecker suggestions (*spellchecker.cg3*) and a module for more advanced grammar checking (*grammarchecker-release.cg3*) are included in the spelling and grammar checker.

The current order of the modules has shown to be the most optimal one for our use and has been established during the work with the grammar checker. It follows the principle of growing complexity, and information necessary to subsequent modules is made available to them. Valencies for example are used in the disambiguation of compounds, which is why the module precedes the multi-word disambiguation module. The first whitespace tagging module precedes sentence boundary disambiguation because it is used to insert hints about the text structure. It marks, for example, first and last words in the paragraph, which is relevant when deciding if a period marks the end of sentence or is part of an abbreviation or a numeric expression. The second whitespace analyser is applied after the multiword disambiguation, but could also be applied later. Its purpose is to tag potentially wrong use of whitespace, and must be added before the final grammar checking module, but any position between the multiword disambiguation and grammar checking is going to work. Spellchecking is performed before disambiguation so that more sentence context is available to the syntactic analyser and disambiguator.

It should be noted that we do not use a guesser for out-of-vocabulary words. A major part of new words are formed using productive morphology of the language, like compounding and derivation, both of which are encoded in the morphological analyser. Also, the lexicon is constantly being updated, so that proper nouns and other potential out-of-vocabulary words will quickly be covered. As updates are made available frequently, this should not be a major issue for users, but some issues are discussed towards the end of this article.

There are both language specific and language independent parts in the *GiellaLT* infrastructure. The *GiellaLT* infrastructure is developed at *UiT The Arctic University of Norway* to support the development of language technology for the Sámi languages. It covers language independent components,[4] as well as language-specific code with a supporting build system.[5] Although initially developed for the Sámi languages, the infrastructure has been built with language independence in mind, and presently there are about a hundred languages in various stages of development in our infrastructure. The linguistic code is gen-

[4] cf. `https://github.com/divvun`

[5] cf. `https://gtsvn.uit.no/langtech/trunk/`

erally language-specific. However, the annotation of syntactic functions and dependencies is generalized for several related languages (South Sámi, Lule Sámi, North Sámi, and Pite Sámi) (Antonsen et al., 2010). The technical code on the other hand is language-independent. The infrastructure is not only valid for North Sámi, but can directly be used by any language in the *GiellaLT* system, e.g. other Sámi languages as well as any other language. Presently there is an early version of a working Faroese grammar checker in addition to the North Sámi one, and initial work has started for a number of the other Sámi languages.

The system does error detection at four different stages of the pipeline. Non-word typos are marked by means of the spellchecker. Secondly, a Constraint Grammar module marks whitespace errors in punctuation contexts based on input from the second whitespace analyser. Compound errors are identified by means of a Constraint Grammar-based tokenisation disambiguation file. And a fourth Constraint Grammar module for advanced grammatical errors and punctuation marks quotation errors.

As a tool intended to be used in production by regular users, it targets all types of errors, from technical typesetting errors such as wrong quotation marks and faulty use of spaces, via spelling errors to advanced grammatical error detection and correction. In the evaluation all of these are counted as grammar checker errors, as we want to evaluate the overall performance of the tool. The only errors not included in the evaluation are those that we do not target at all (which are quite a few in this first beta release).

The sentence in ex. (1) includes a typo (*Norgag* should be *Norgga* 'Norway' (Gen.)), a space error (before '.') and a compound error (*iskkadan bargguin* should be *iskkadanbargguin*) 'survey' (Loc. Pl.). In addition, there is a congruence error, i.e. *dáid* (Gen. Pl.) 'these' should be *dáin* (Loc. Pl.), i.e. it should agree in case and number with *iskkadan bargguin*. The launched grammar checker can detect the first three errors, but not the last one, since syntactic error rules are not included in this initial launch of the grammar checker.

(1) Oktiibuot 13 **Norgag** doaktára leat
altogether 13 Norway.GEN doctor.GEN have
leamaš mielde dáid
been with these.GEN
iskkadan bargguin_.
testing work.LOC.PL
'Altogether 13 Norwegian doctors have participated in these surveys.'

The whitespace analyser detects an erroneous space before '.' The suggested correction is *"<iskkadan bargguin.>"*. The tokenisation disambiguation module detects the compound error and suggests the combination of lemma and tags *iskkadanbargu+N+Sg+Com* resulting in the form *iskkadanbargguin*. The tokeniser is used to disambiguate between syntactically related n-grams and misspelled compounds, where the misspelling is an erroneous space at the word boundaries.

This module is clearly checking more than spelling conventions, i.e. grammar, as writing, for example, two consecutive nouns as one or two words has syntactic implications. Such noun-noun combinations do not necessarily need to be compounds even if the first element is in nominative case. They can also be syntactically related as in the agent construction in ex. (2) where *addin* 'giving' is a (nominalized) non-finite verb that modifies the second noun, *vuođđu* 'basis'.

This grammar checker finds compound errors by distinguishing between syntactic readings as the previous one and compound readings as *addinvejolašvuođa* 'giving possibility (Gen.)'.

(2) luossabivdu lea lunddolaš
salmon.fishing is natural
Golf-rávnnji **addin**
Golf-stream.GEN give.ACTIO.NOM
vejolašvuođa vuođđu
possibility.GEN basis
'salmon fishing is a natural resource for a possibility given by the Golf-stream'

2.2 Evaluation tools

Previously, we had evaluated our tools manually, but now we wanted to be able to measure the improvement of our tools more consistently.

2.2.1 Annotated corpus

An evaluation tool presupposes an annotated corpus. The corpus we use for evaluation contains 226 336 words and is manually annotated by a North Sámi speaker/linguist according to the following principles.[6]

[6]The raw corpus texts for the openly accessible corpus can be found at `https://gtsvn.uit.no/freecorpus/orig/sme`. The corpus used for the evaluation in this article can also be found at `https://gtsvn.uit.no/`

The tokens involved in the error are enclosed in parenthesis followed by a sign for their general error type (orthographic, real word, morpho-syntactic, syntactic, lexical, formatting, foreign words), which is followed by another parenthesis containing error subclassification and the expected correction. The subclassification may contain annotation that is customized for the main six different error types. Sometimes part of speech, and morpho-syntactic criteria are specified or the error is further explained, e.g. regarding congruence. The description is followed by a correction of the error after the pipe sign. In ex. (3), the tokens involved in the error are nouns, the syntactic error is a compound error and the correction is *riikkačoahkkinmáhpas*.

(3) Áššebáhpáriid gávnnat (riikkačoahkkin máhpas)¥(noun,cmp|riikkačoahkkinmáhpas)

Orthographic errors (marked by $) include non-words only. They are traditional misspellings confined to single (error) strings and the traditional speller should detect them. Real word errors (marked by ¢) cannot be detected by a traditional speller. Morpho-syntactic errors (marked by £) are case, agreement, tense, mode errors. They require an analysis of (parts of) the sentence or surrounding words to be detected. Syntactic errors (marked by ¥) require a partial or full analysis of (parts of) the sentence or surrounding words; word order, compound errors, missing words, redundant words and so on. Lexical errors (marked by €) include wrong derivations. Foreign words (marked by ∞) include single words in other languages like, for example, Norwegian *og* and *august*. Formatting errors (marked by ‰) include spacing errors in combination with punctuation.

(4) an.lávdegoddi
?.committee

When doing error mark-up, it can be challenging to find the appropriate corrections in cases like ex. (4), where a copy-paste error results in the first part of the word missing. Without the original text, it can be close to impossible to reconstruct the intended form from the remaining text.

freecorpus/orig/nodalida2019

2.2.2 Evaluation tool

The evaluation tool is written in python. It reads the corpus files, runs the grammar checker on the original version of each paragraph, compares the output with the gold standard markup in the corpus, and collects the results. The output is written to a text file, with a report for each paragraph and also the overall evaluation measures.

The paragraph report covers mark-up errors that do not correspond to grammar checker errors (usually false negatives) and grammar checker errors that do not correspond to mark-up errors (false positives or missing mark-up). When mark-up and grammar checker errors align, the report covers whether or not the mark-up errors correction is among the grammar checker suggestions, and reports any missing suggestions by the grammar checker.

The overall report provides precision, recall and F-score for all errors and also breaks these numbers down for the respective error classes.

As the grammar checker does not presently cover neither lexical, morpho-syntactic nor real word errors, these error types are filtered away as they are read in by the evaluation tool, such that the expected correct text as given by the corpus mark-up is the text used as input to the testing. As we expand the coverage of the grammar checker, new error types will be used in testing it as well.

3 Evaluation

3.1 Quantitative evaluation

We calculate both precision and recall.

$$\textbf{Precision} = \frac{\text{number of items correctly retrieved}}{\text{number of items actually retrieved}}$$

$$\textbf{Recall} = \frac{\text{number of items correctly retrieved}}{\text{number of items that should have been retrieved}}$$

A previous (manual) evaluation of compound error detection, which is the linguistically most advanced error type in *GramDivvun*, has resulted in a precision of 76.6% and a recall of 78.6% (Wiechetek et al., 2019). The development of the grammar checker evaluation tool presented in this paper has been informed by the previous evaluation, and many errors in the infrastructure and in the Constraint Grammar modules have been corrected. Therefore, higher precision and recall are expected in this evaluation.

As opposed to the evaluation done in Wiechetek et al. (2019), the evaluation of *GramDivvun 1.0 beta* is automatic. The evaluation is based on version *r183544* of the grammar checker[7]. An automatic evaluation is meaningful as the error corrections intended in this launch typically include a limited amount of adjacent word forms, which is relatively straightforward. Reference-less approaches as proposed in Napoles et al. (2016) are not an option as they require pre-existing and independently developed tools that are sensitive to grammatical errors, a luxury not available to most minority languages.

A part of the North Sámi *SIKOR* corpus (SIKOR2016) containing manual error markup for orthographical and grammatical errors is used as the evaluation corpus. It consists of 226 336 words. The genres represented in the evaluation corpus are news, blogs, and teaching materials.[8]

In the evaluation, we only consider spelling errors (only non-words), compound errors, space errors and punctuation errors (only quotation marks for now), i.e. only the error types we actually try to correct. The performance of the grammar checker is measured by means of precision and recall. Good precision has typically priority over good recall as users tend to react more critically to flagging correct input as errors as opposed to not flagging error input.

The results are presented in Table 2. Both precision and recall are above 85%, i.e. above the previous results for compound error detection. As a reference, when Bick (2015) evaluates his full-fledged grammar checker *DanProof*, for correcting both, spelling and compounding errors precision is 90.8% and recall is 86.8%. While our recall is close to Bick's results, precision can definitely be improved. Orthographic error detection (spell checking) performs best (87.3%), a good improvement compared to earlier results.[9] Formatting error detection performs slightly worse (82.2%). The results for syntactic (compound) error detection are lowest of all error types (73.9%), and slightly lower than the results in Wiechetek et al. (2019). This may be partly due to a smaller amount of compound errors in this corpus (344) compared to the one in Wiechetek et al. (2019) (458). The qualitative evaluation below will shed light on other possible reasons for *GramDivvun* shortcomings.

Measure	Overall	Orth	Format	Syn
Precision	85.4%	86.1%	85.0%	75.0%
Recall	85.8%	88.5%	79.6%	72.9%
F-Score	85.6%	87.3%	82.2%	73.9%
TP	1.319	991	277	51
FP	226	160	49	17
FN	219	129	71	19

Table 2: Precision, recall and F-Score of GramDivvun 1.0 beta (TP = true positives, FP = false positives, FN = false negatives)

3.2 Qualitative evaluation

In this section we analyse and discuss reasons for shortcomings in precision and recall in different modules of the grammar checker. Common reasons for false positives and false negatives are lexicon- and fst-related issues. These are, for example, missing items in the lexicon, cited fragments in other languages that do not receive an analysis and errors the treatment of clitics. Furthermore, there are named entity related issues (427 cases), misspelled compounds (i.e. written apart) that are not listed as such in the lexicon, and misspelled compounds that are not listed as such (split compounds with an error in the first part). In addition to lexicon-related issues, there are shortcomings in the disambiguation rules and in the error detection modules of the grammar checker, where Constraint Grammar rules do not recognize syntactically related words, and analyses them as compounds. Other reasons are related to the whitespace analyser, where space errors are erroneously detected because we do not recognize first/last words in a sentence.

[7] To obtain this version run: `svn co -r183544 https://gtsvn.uit.no/langtech/trunk langtech; cd langtech/giella-core; ./autogen.sh; ./configure; cd ../giella-shared; ./autogen.sh; ./configure; cd ../langs/sme; ./autogen.sh; ./configure -with-hfst -without-xfst -enable-grammarchecker -enable-tokenisers -enable-alignment -enable-reversed-intersect; make -j`

[8] `https://giellalt.uit.no/proof/nordplus/StavekontrolltestingOgNorplusprosjektet.html` (Retrieved 2019-06-19)

[9] `https://gtsvn.uit.no/biggies/trunk/techdoc/proof/spelling/testing2/sme/to/goldstandard/20180207-1355-corpus-gs-results.html`

3.2.1 False positives

To identify shortcomings in precision false positives need to be analysed.

One issue are shortcomings in the compound disambiguation rules, where in the case of two correct alternative spellings (hyphenated vs. non-hyphenated version and one word vs. two-word version), a correct one is replaced by the other correct one. In ex. (5), *Riddu Riđđu* is written without a hyphen (which is correct). However, the grammar checker corrects it to *Riddu-Riđđu*. This issue can be resolved by explicitly tagging the multiword spellings in the lexicon and making an exception in the Constraint Grammar rules that annotate an error.

(5) Guollefestivála geassemánus ja
fish.festival june.LOC and
Riddu Riđđu suoidnemánus.
Riddu Riđđu July.LOC
'The fishing festival in June and Riddu Riđđu in July'

In the case of first and last names, hyphenated and non-hyphenated versions of the same name are very common. The spelling of human names exhibits a great variation, and names are hard to standardize.

In ex. (6), *Inger Anna* is falsely corrected to *Inger-Anna* by the compound error detection module. Because of the great productivity of names, Constraint Grammar rules should contain special exceptions for human names.

(6) Sámegillii: **Inger Anna** Gaup Gustad.
Sámi.ILL: Inger Anna Gaup Gustad
'In Sámi: Inger Anna Gaup Gustad.'

In ex. (7), *rahppo* receives an error tag by *GramDivvun* even though it is correct. The form *rahppo* receives a regular derived analysis (i.e. the passive analysis of *rahpat* 'open') and an underived lexical error analysis (i.e. the indicative analysis of *rahppot* 'be opened') by the morphological analyser. The disambiguator should ideally remove the lexical error analysis in favour of the correct one. The solution is an adaption of the disambiguator so that possible error-tag analyses that compete with correct readings are removed.

(7) Go Norga **rahppo** "Európái"
when Norway open.up Europe.ILL
'When Norway opens up to Europe'

There are also syntactic issues as in ex. (8). The grammatical tokeniser does not recognize syntactically related words like *prošeakta* 'project' and *virggálaččat* 'professionally', but analyses them as a compound *prošeaktavirggálaččat* '?project professionally' based on the fact that it is a denominal derivation of *prošeaktavirgi* 'project position'. Constraint Grammar rules should be changed as to recognizing the syntactic relation between the words. An alternative solution is blocking the compound reading of two-word expression for derivations in the lexicon.

(8) Dát lea čielga hálddahuslaš dássi, gos
this is clear administrative level, where
prošeakta **virggálaččat** registrerejuvvo
project professionally register.PASS.3SG
ja álggahuvvo.
and start.PASS.3SG
'This is a clear administrative level, where a project is registered and started professionally.'

3.2.2 False negatives

In order to find the reasons for shortcomings in recall we have to look at false negatives. In ex. (9), the proper noun compound *Sámiid Hilat* is not hyphenated as it should be according to the norm. However, *Sámiid* is in genitive case, and in this case the compound disambiguation module removes the error reading, as genitive nouns can be prenominal modifiers. A possible solution is to list and error-mark the non-hyphenated *Sámiid Hilat* in the lexicon.

(9) Artihkkal aviissas **Sámiid Hilat**,
Article newspaper.LOC Sámi.GEN Hilat,
nr. 2 - 1978
nr. 2 - 1978
'The article in the newspaper Sámiid-Hilat, nr. 2 - 1978'

Lexicon issues include the treatment of clitics. We treat clitics and stand-alone particles in the same way. That means that certain words get erroneously analysed as combinations with stand-alone particles even if these are never to be found in combination with other words. In ex. (10) *dálkadat* has a spelling error, and should be *dálkkádat*. However, since it receives an analysis based on *dálkká* and the particle *dat*, the spelling error is not found.

(10) Suohkanis lea buorre **dálkadat**
county.LOC is good climate
'The county has a good climate.'

Another problem is the exact span of the marked-up area. Space errors, for example, cannot be marked on the (potentially missing) space itself. Instead, the surrounding words are marked. Since the Constraint Grammar rule for tagging these errors are not properly constrained, error tags are erroneously added to further words. In ex. (11), the double space error between *šat* and *ságastallat* is not found.

(11) ja dalle dáid birra eat
and then these.GEN.PL about not.3PL
dárbbaš šat__ságastallat
need anymore talk
'and then we don't need to talk about these things anymore'

This is an instance of error overlapping, i.e., two errors have overlapping contexts, and only the last error is flagged and corrected. This is another result of mixing user interface considerations with error detection and correction code. We encode the linear scope of the visual error marking in the Constraint Grammar rules, including the relevant surrounding words. This should better be left to a component closer to the user interface, and the Constraint Grammar rules should only tag the error strings themselves. Then two consecutive errors would not get conflicting mark-up, and the user interface component can resolve adjacent errors in the most appropriate way.

3.3 Discussion of the automatic evaluation

There is a certain amount of discrepancy between the identified errors reported by *GramDivvun*, and the manually marked-up errors in the corpus.

The manual markup of ex. (12) classifies *skuvla(* 'school(' as missing space before the parenthesis and *Oahpit* 'students' as a typo. *GramDivvun* does also find these two errors, but does not distinguish between them in the string positions for the errors: it uses the same area for both of them. That means that the identified typo includes the opening parenthesis as well in the identified erroneous part of the input, even though technically only *Oahpit* should be a typo.

However, as mentioned above, the manual mark-up marks *skuvla(* 'school(' and *Oahpit* 'students' as separate errors with different string spans. The evaluation tool reformats and adjusts the spans of the *GramDivvun* output to match the manual markup. It is therefore possible to do meaningful and usable comparisons, which is necessary for an automatic evaluation.

(12) skuvla(Oahpit
school(students

The major insight from the automatic evaluation is that our present grammar checker design is not optimal. We currently encode user interface information together with error detection and correction information at an early stage, adding information of the span of the error. This complicates the Constraint Grammar rules for error markup and correction. Rules extend the error span to neighbouring words, which themselves can have errors, creating a spaghetti of interdependences.

Untangling this mess has been a major effort during the evaluation. A cleaner design that avoids these problems is one of the objectives for the final release of *GramDivvun 1.0*. This means marking the span of an error at a later state, closer to the user interface, so it does not interfere with the rules for the actual error identification and correction.

For now, the automatic evaluation tool identifies the proper span of the errors. However, some errors may remain, both, in the evaluation and in the reported results.

4 Conclusion

In this paper we have presented *GramDivvun 1.0 beta*, the first released combined spelling and grammar checker tool for North Sámi that, in addition to spelling errors, detects and corrects punctuation errors, compound errors and white space errors. Additionally, this work also describes the complete infrastructure for a full-scale grammar checker and facilitates the implementation of any kind of grammatical error correction as soon as these are considered to be working well enough to be released. The infrastructure is available for any language within the *GiellaLT* framework.

We have exploited many shades of Constraint Grammar at all stages of the grammar checking process. Constraint Grammar is used in most error detection and correction. This includes context-aware filtering of spelling suggestions. The overall F-Score is 85.6%. Based on the qualitative evaluation and systematic error-debugging many (frequent) error types could be resolved after evaluating *GramDivvun 1.0 beta*, and better results are expected for future evaluations. Overall precision (85.4%) is at the moment not better than overall

recall (85.8%).

In addition, we have developed an automatic evaluation method, which will facilitate quality maintenance and allow us to release updated versions of the grammar checker more frequently. We have learned that manual corpus mark-up can get substantial support from the automatic testing, as the grammar checker often finds more errors than the ones visible to the human eye. We have also learned that the evaluation tool needs constant surveillance to ensure concordance with the desired features to be evaluated. Finally, diverging principles of error mark-up and grammar checker error detection pose challenges to the automatic evaluation.

Future plans include adding a name guesser, improving lexicon coverage, adding Constraint Grammar rules for variations in the lexicon (avoiding false positives) and refining the syntactic rules in the compound error detection module. Additionally, we plan to remove the user-interface elements from the present Constraint Grammar rules. As the infrastructure is ready, a wide range of real word errors and syntactic error detection rules will be included in future releases and turn *GramDivvun* into a full-scale North Sámi grammar checker.

Acknowledgments

We want to thank Kevin Brubeck Unhammer for his profound work with the grammar checker infrastructure, for very valuable discussions regarding the design of the grammar checker, for helping out with certain types of advanced Constraint Grammar rules, and for building our first prototype in LibreOffice.

References

Lene Antonsen, Linda Wiechetek, and Trond Trosterud. 2010. Reusing grammatical resources for new languages. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*, pages 2782–2789, Stroudsburg. The Association for Computational Linguistics.

Eckhard Bick. 2015. DanProof: Pedagogical spell and grammar checking for Danish. In *Proceedings of the 10th International Conference Recent Advances in Natural Language Processing (RANLP 2015)*, pages 55–62, Hissar, Bulgaria. INCOMA Ltd.

Eckhard Bick and Tino Didriksen. 2015. CG-3 – beyond classical Constraint Grammar. In *Proceedings of the 20th Nordic Conference of Computational Linguistics (NoDaLiDa 2015)*, pages 31–39. Linköping University Electronic Press, Linköpings universitet.

Børre Gaup, Sjur Moshagen, Thomas Omma, Maaren Palismaa, Tomi Pieski, and Trond Trosterud. 2006. From Xerox to Aspell: A first prototype of a north sámi speller based on twol technology. In *Finite-State Methods and Natural Language Processing*, pages 306–307, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sam Hardwick, Miikka Silfverberg, and Krister Lindén. 2015. Extracting semantic frames using hfst-pmatch. In *Proceedings of the 20th Nordic Conference of Computational Linguistics, (NoDaLiDa 2015)*, pages 305–308.

Fred Karlsson. 1990. Constraint Grammar as a Framework for Parsing Running Text. In *Proceedings of the 13th Conference on Computational Linguistics (COLING 1990)*, volume 3, pages 168–173, Helsinki, Finland. Association for Computational Linguistics.

Fred Karlsson, Atro Voutilainen, Juha Heikkilä, and Arto Anttila. 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin.

Eva Lindgren, Kirk P H Sullivan, Hanna Outakoski, and Asbjørg Westum. 2016. Researching literacy development in the globalised North: studying trilingual children's english writing in Finnish, Norwegian and Swedish Sápmi. In David R. Cole and Christine Woodrow, editors, *Super Dimensions in Globalisation and Education*, Cultural Studies and Transdiciplinarity in Education, pages 55–68. Springer, Singapore.

Sjur N. Moshagen, Tommi A. Pirinen, and Trond Trosterud. 2013. Building an open-source development infrastructure for language technology projects. In *NODALIDA*.

Courtney Napoles, Keisuke Sakaguchi, and Joel R. Tetreault. 2016. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2109–2115.

Hanna Outakoski. 2013. Davvisámegielat čálamáhtu konteaksta [The context of North Sámi literacy]. *Sámi dieđalaš áigečála*, 1/2015:29–59.

Tommi A. Pirinen and Krister Lindén. 2014. State-of-the-art in weighted finite-state spell-checking. In *Proceedings of the 15th International Conference on Computational Linguistics and Intelligent Text Processing - Volume 8404*, CICLing 2014, pages 519–532, Berlin, Heidelberg. Springer-Verlag.

SIKOR2016. 2016-12-08. SIKOR UiT The Arctic University of Norway and the Norwegian Saami Parliament's Saami text collection. **URL:** *http://gtweb.uit.no/korp (Accessed 2016-12-08)*.

Gary F. Simons and Charles D. Fennig, editors. 2018. *Ethnologue: Languages of the World*, twenty-first edition. SIL International, Dallas, Texas.

Linda Wiechetek. 2017. *When grammar can't be trusted – Valency and semantic categories in North Sámi syntactic analysis and error detection*. PhD thesis, UiT The Arctic University of Norway.

Linda Wiechetek, Kevin Brubeck Unhammer, and Sjur Nørstebø Moshagen. 2019. Seeing more than whitespace – Tokenisation and disambiguation in a North Sámi grammar checker. In *Proceedings of the third Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 46–55.

Constraint Grammar as a Hand-Crafted Transformer

Anssi Yli-Jyrä
Helsinki Centre for Digital Humanities (HELDIG)
P.O. Box 24, 00014 University of Helsinki, Finland
anssi.yli-jyra@helsinki.fi

Abstract

The differences between the rule-based NLP such as CG and the deep neural networks, such as the Transformer (Vaswani et al., 2017) are so striking that it is really hard to see any relevant conceptual links between them. However, this paper sketches a thought experiment that assumes an equivalent input-output behaviour by both systems and aligns certain structural aspects of the computation behind a practical Constraint Grammar with the computation structure of Transformer. Based on this scene, several findings are presented that state some functional similarities in the computation graphs of the systems.

1 Introduction

The Transformer architecture (simply *Transformer*) (Vaswani et al., 2017), and Constraint Grammar parsing framework (simply *CG*) are currently in the opposite ends of the continuum for different NLP technologies (Table 1). The main contrasts between these relate to the representation of word senses and the way in which the systems implement machine learning. Learning in both systems is *error-driven*, but CG can use *transformation-based learning* algorithms (Brill, 1995; Lager, 2001) that differ greatly from the backpropagation algorithm used as a part of the gradient descent optimisation of Transformer. A more striking, but superficial difference is the way how the systems traditionally represent their lexicons. A Transformer network (*a* Transformer) assumes a lexicon of word parts. This maps the word parts, or tokens, directly to a high-dimensional vector space. A Constraint Grammar parser (*a* CG) has typically access to a finite-state based lexicon that assigns, to each word, a set of morpho-syntactic readings and categories. These sets are called *cohorts*.

Since there is no obvious link between constraint grammars and deep neural networks, the two methods are seldom studied in parallel. If there are some links, they are hardly ever been pointed out. One harmful consequence of this state of affairs is that it is not known how to combine these technologies in a synthetic design. Therefore, it is especially valuable to investigate how these unrelated technologies could be aligned and even married with one another. Accordingly, the aim of the current paper is to start a discussion that seeks for cross-design understanding and synthesis. This discussion may lead to ideas that allow us to create NLP that takes advantage of both expert knowledge and big data.

2 Premise and Methodology

Talman et al. (2019) compared the performance of a CG-based machine translation (MT) system and a Transformer-based sys-

Table 1: Some contrasts

	Transformer	*CG*
based on:	neural networks	restarting automata
data need:	large	moderate
tokens:	word parts	inflected words forms
input sequence:	word embeddings	cohorts of readings
word senses:	continuous	discrete tag sequences
features:	learned feature representations	template or lattice based features
special use:	pretrained embeddings	gold annotation
learning:	backpropagation and stochastic gradient descent (SGD)	composition and transformation-based learning (TBL)

tem, bringing them thus to the same table for comparison in terms of their respective translation quality. Our aim is to consider the theoretical consequences of a different comparison that is based on a thought experiment, a powerful method mastered by G. Galilei and A. Einstein in particular.

The Premise of Equivalence. According to our thought experiment, we assume the imaginary situation where both systems would happen to compute the same, nontrivial function. Although this state of affairs is unlikely to be generally achievable, there is a realistic posibility that a Transformer is able to learn to compute the same input-output mapping as a typical CG.

The Computation Graph Method. In such a world where the same mapping would be computed by two kinds of systems, it is natural to ask whether the equivalent behaviour has something to do with a similar structure that is shared by both systems. Perhaps the kind of structure with most promising parallels is the high-level computation graph of both systems. A computation graph is a directed graph that shows the flow of information in a system that consists of several connected processing modules.

The Alignment Hypothesis. The current hypothesis is that both systems actually have corresponding computation steps that can be functionally aligned with each other.

Since the experiment facilitates the detection of analogies in design, it has potential value for further research: we may want to build robust systems where CG and statistical models complement one another, or hybrid systems that contain some computation steps from CG and some other steps from a neural encoder-decoder architecture.

3 Aligned Encoders

To argue for conceptual connections between Constraint Grammar and Transformer, we start from very general observations.

The encoder-decoder components. A Transformer is a composition of a stack of encoder networks and a stack of decoder networks: *enc* ∘ *dec*. The layers of the *encoder networks* (*enc*) build an internal representation for the input string. Then, a multi-layer *decoder* network (*dec*) produces an output string based an internal representation.

Finding 1. *Both systems contain an* encoder component *that embeds the input sequence to a sequence of contextually disambiguated feature representations of tokens at each position.*

Proof. This is clearly true for Transformer. Constraint Grammar is an encoder that maps the sequences of ambiguous cohorts to sequences of (nearly) unambiguous cohorts that represent the contextual reading of each token in a sentence. □

The existence of a decoder component in both systems can also discussed. CG does not usually have a decoder component, but it has sometimes been extended with modules that can be seen as decoders. For example, Hurskainen (1999) and (Hurskainen and Tiedemann, 2017) describe CG-based systems where the disambiguated input string is processed further by "decoding" modules. These modules implement a mapping from the contextualised token representations to a representation of the corresponding target language tokens, and a mapping from the original word order to the target word order, etc. Since the internal structure of these modules is still somewhat different from the decoders of the Transformer architecture, their similarities cannot be demonstrated in the current work.

Information reduction. By design, the encoder stack of Transformer is a multi-layer neural network. The information content of the output is a subset of the information content that is available in the input. The rest of the input information is irrelevant for the output and is thrown out during the encoding process.

Finding 2. *The encoders of both systems are functions and thus reductionistic: the amount of information that is relevent for the output representation is not increasing inside the encoders.*

Proof. The output layer of a Transformer is a function of the input layer. A Constraint Grammar is generally known to be an iterated function that transforms the input sequence to a less ambiguous one. □

Feature vectors. Although the tokens are, in many ways, ambiguous in the beginning, Transformer assumes a finite lexicon of tokens. It embeds the tokens to a space of continuous representations.

Finding 3. *Both systems represent the input tokens as feature vectors.*

Proof. Yli-Jyrä (2011a) has demostrated how to embed the input cohorts of a CG system into finite vectors. These vectors contain the values of those hand-engineered features that are used in rule conditions and are thus relevant for the function defined by the CG grammar. Such vectors can be extended to a lossless representations from which the input cohorts in a finite lexicon can be decoded. Such a representation is comparable with the token embedding vectors used by Transformer. □

A finite number of layers. The encoder network in Transformer contains typically 6–64 similar layers (with different weights).[1]

Finding 4. *The number of layers in both encoder architectures is potentially finite.*

Proof. The finite bound for layers holds for Tranformer by definition. Yli-Jyrä (2017b) conjecture that, in practice, each CG can be viewed as a finite-visit Turing machine, which is known to be equivalent to a functional one-way finite-state automaton or transducer, see Yli-Jyrä (2017a). Such a machine model has a reading-writing head that does not cross any position in the sentence more than k times. According to the argument, this optimisation is made possible by the assumption that at most a finite amount of information needs to be communicated across each sequence position. Hulden (2011), Peltonen (2011) and Yli-Jyrä (2011a) present similar analyses for separate CG rules but they do not reach the conjecture that finite visits and bounded crossings per position would be sufficient for the correct function semantics of the whole CG parser where the rules are supposed to apply iteratively. Once the equivalence with a finite-state transducer is established, it is easy to see that a multi-layer composition of several finite-state transducers can be much more succinct way to compute the same function. Thus, both encoders have a finite number of layers in the case where the finite-visit conjecture holds. □

[1] This raises a question, could the encoder network have recurrent layers that share their weights. This would make the encoder even more similar to a CG.

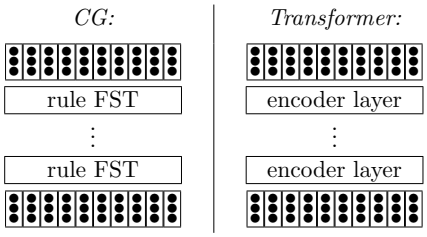

Figure 1: Alignment of the encoders

Given these four observations, we can draw a diagram (Figure 1) that aligns the computational structure of the encoder of Transformer with a CG system.

4 Aligned Sublayers

The alignment inside the corresponding layers of the encoders requires us to dig into the more detailed structure inside both CG and Transformer. There are at least four ways to approach a CG parser, but only the last one helps us to see the analogy between the two systems:

(i) Iterative Rule Application. Some CG parsers have a control mechanism that iterates over the positions of the sentence and over the disambiguation rules, trying to apply each rule at each position at a time. If non-monotonic rules are included, the parser becomes Turing complete (Kokke and Listenmaa, 2017; Yli-Jyrä, 2017b). Since Transformer has been specifically designed to have a bounded number of layers, the iterative rule application differs too much from it.

(ii) Constraint Programming. Another view on constraints treats the constraints conjuctively, requiring the output sequence to satisfy all the constraints. For some inputs, the system can be over-constrained, which would make the parser to fail, unless some constraints are relaxed (Listenmaa, 2019). Such constraint programming approach is quite different from the Transformer architecture whose encoders are position-wise, without any constraint relaxation.

(iii) Maximum Subgraph Parsing. A new approach to constraint relaxation is to re-

formulate the constrained parsing as a maximum subgraph problem over encoded graphs. Yli-Jyrä and Gómez-Rodríguez (2017) consider complete dependency graphs with arc-factored weights and sketches an efficient parsing algorithm for maximun weighted noncrossing subgraphs that satisfy some hard constraints, such as acyclicity and connectivity. The weights are needed because the system is under-constrained. With a more general encoding (Yli-Jyrä, 2019), it is possible to remove the limitation of noncrossing parses from the parsing algorithm.

(iv) Separate Context Queries. In Yli-Jyrä (2011a), the context conditions of CG rules are represented by an efficient query-FSA (Figure 2) that can match thousands of complex conditions in parallel. The position-wise satisfaction of context conditions is encoded into strings using *position-wise flag diacritics* (Yli-Jyrä, 2011b), a generalisation of Liang's hyphenation algorithm (Liang, 1983). These diacritics of the query-FSA encode a positionwise vector that tells which contexts are present at each position.

Finding 5. *The two subnetworks of the encoder in Transformer can be compared with the computation steps used in Yli-Jyrä (2011a).*

Proof. In Transformer, each encoder network consists of two position-wise networks: (1) a self-attention network and (2) a feed-forward (FF) network. The first network queries, in parallel for every input position, an attention-weighted average vector that describes an aspect of its context in the sentence. After this, the FF network manipulates the vector that encodes the contents of each position based on the information gathered via self-attention.

The computation structure of Transformer corresponds to the fourth way to implement Constraint Grammar. The CG implementation of Yli-Jyrä (2011a) tests firsts the the context conditions of all disambiguation rules in all positions. After this, the systen chooses an input cohort, manipulates its feature representation (and the cohort), and updates the tests around the changed cohort. The system could also take some risk of suboptimal search and change multiple cohorts at the same time.

If sufficiently many positions are changed during one iteration, the CG parser can be implemented with a finite cascade of finite-state transductions. This is the general case where the system becomes quite Transformer-like. A special case of similar, cascaded application of parallel rules has been explored by Hulden (2011). □

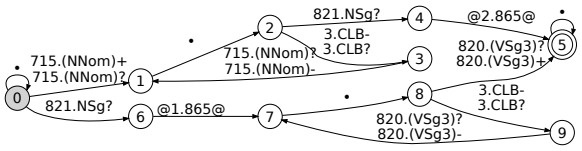

Figure 2: a small portion of a "self-attention" query network in CG

CG:	*Transformer:*
Feature-based cohort manipulations	FF network for position-wise manipulations
Recurrent "self-attention" query network	Non-recurrent self-attention network

Figure 3: Sublayers of the encoders

The alignment of the sublayers gives us a picture (Figure 3). Further analysis is needed to check whether the conditions of CG rules are simple enough to be simulated with the self-attention mechanism.

5 Conclusion

In this paper, we have aligned the high-level computational structures of CG parser and a Transformer under their functional equivalence. This resulted in findings

1. on their encoder-decoder decomposition,
2. on their reductionistic nature,
3. on their vectorized token representations,
4. on their finite number of layers, and
5. on the two steps in each encoder layer.

We also noted that the alignment is not perfect. For example, it is probable that the cohort vectors and the word embedding vectors do not represent the lexical or morphological ambiguity in the same way. Understanding the significance of this difference would be crucial for the discussion about interpretability and invertibility of token representations. Nevertheless, the alignment suggests the possibility of hybrid parsing models that would combine these architectures and their complementary strengths in NLP.

References

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Comput. Linguist.*, 21(4):543–565.

Mans Hulden. 2011. Constraint grammar parsing with left and right sequential finite transducers. In *Proceedings of the 9th International Workshop on Finite State Methods and Natural Language Processing*, pages 39–47, Blois, France. Association for Computational Linguistics.

Arvi Hurskainen. 1999. SALAMA Swahili Language Manager. *Nordic Journal of African Studies*, 8(2):139–157.

Arvi Hurskainen and Jörg Tiedemann. 2017. Rule-based machine translation from english to finnish. In *Proceedings of the Second Conference on Machine Translation, WMT 2017, Copenhagen, Denmark, September 7-8, 2017*, pages 323–329. Association for Computational Linguistics.

Pepijn Kokke and Inari Listenmaa. 2017. Exploring the expressivity of constraint grammar. In *Proceedings of the NoDaLiDa 2017 Workshop on Constraint Grammar - Methods, Tools, and Applications, 22 May 2017*, pages 15–22, Gothenburg, Sweden. Linkoping University Electronic Press.

Torbjörn Lager. 2001. Transformation-based learning of rules for constraint grammar tagging. In *NODALIDA*.

Franklin Mark Liang. 1983. *Word Hy-phen-a-tion by Com-pu-ter*. Ph.D. thesis, Stanford University, Department of Computer Science.

Inari Listenmaa. 2019. *Formal Methods for Testing Grammars*. Ph.D. thesis, Department of Computer Science and Engineering, Chalmers University of Technology and University of Gothenburg, Gothenburg, Sweden.

Janne Peltonen. 2011. Rajoitekielioppien toteutuksesta äärellistilaisin menetelmin (On finite-state implementation of Constraint Grammars). Master's thesis, University of Helsinki, Department of Modern Languages, Helsinki.

Aarne Talman, Umut Sulubacak, Raúl Vázquez, Yves Scherrer, Sami Virpioja, Alessandro Raganato, Arvi Hurskainen, and Jörg Tiedemann. 2019. The university of Helsinki submissions to the WMT19 news translation task. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 412–423, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, USA. Curran Associates Inc.

Anssi Yli-Jyrä. 2011a. An efficient constraint grammar parser based on inward deterministic automata. In *Proceedings of the NODALIDA 2011 Workshop Constraint Grammar Applications*, volume 14 of *NEALT Proceedings Series*, pages 50–60.

Anssi Yli-Jyrä. 2011b. Explorations on position-wise flag diacritics in finite-state morphology. In *Proceedings of the 18th Nordic Conference of Computational Linguistics, NODALIDA 2011, May 11-13, 2011, Riga, Latvia*, pages 262–269. Northern European Association for Language Technology (NEALT) / Tartu University Library (Estonia) / ACL.

Anssi Yli-Jyrä. 2017a. Forgotten islands of regularity in phonology. In *$K + K = 120$: Papers dedicated to László Kálmán and András Kornai on the occasion of their 60th birthdays*.

Anssi Yli-Jyrä. 2017b. The power of Constraint Grammar revisited. In *Proceedings of the NoDaLiDa 2017 Workshop on Constraint Grammar - Methods, Tools, and Applications, 22 May 2017*, pages 23–31, Gothenburg, Sweden. Linkoping University Electronic Press.

Anssi Yli-Jyrä. 2019. Transition-based coding and formal language theory for ordered digraphs. In *Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing*, pages 118–131, Dresden, Germany. Association for Computational Linguistics.

Anssi Yli-Jyrä and Carlos Gómez-Rodríguez. 2017. Generic axiomatization of families of noncrossing graphs in dependency parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1745–1755. Association for Computational Linguistics.