NEALT PROCEEDINGS SERIES
VOL. 8

Proceedings of the NODALIDA 2009 workshop

Constraint Grammar and robust parsing

May 14, 2009

Odense, Denmark

Nodalida 2009

Editors

Eckhard Bick

Kristin Hagen

Kaili Müürisep

Trond Trosterud

NORTHERN EUROPEAN ASSOCIATION FOR LANGUAGE TECHNOLOGY

Proceedings of the NODALIDA 2009 workshop
Constraint Grammar and robust parsing

NEALT Proceedings Series, Vol. 8

ISSN 1736-6305

Published by
Northern European Association for Language
Technology (NEALT)
http://omilia.uio.no/nealt

Electronically published at
Tartu University Library (Estonia)
http://hdl.handle.net/10062/14180

Contents

Preface

Constraint Grammar (CG) is a rare species in the Nordic garden of language technology. The framework was invented and developed here, by Fred Karlsson, and it has achieved quite spectacular results. Its success has also been a problem for the framework, since central practitioners have commercialised their results, and withdrawn them from the academic discussion. Whatever the reason is, CG has never drawn a wide audience, not even on its "home ground", the Nordic countries.

The goal of the workshop was partly to make Constraint Grammar and its results more known to collegues, but first and foremost to stimulate the discussion within the CG community, and to facilitate progress. During the last couple of years, CG have improved its way of doing dependency analysis, thereby bridging the gap between "deep" and "shallow" parsing, being both "deep" and "robust". At the same time, the number of applications in which CG is put to use is growing.

The present workshop proceedings contain 4 papers. The two first papers (Trosterud, Bick) present CG parsers for two new languages, Faroese and Esperanto, the latter paper with a focus on dependency grammar. The next paper (Antonsen et al) presents CG in action, for a parser-based intelligent Corpus-Assisted Language Learning (iCALL) program for North Sámi. The fourth paper (Lindström and Müürisep) presents CG in a well-known setting, as a corpus parser, but this time for a corpus of non-standardised language, Estonian dialects. At a time where more and more old dialect archives are digitized, this is a highly relevant topic.

The workshop also contained two presentations which were not submitted for publication: Kevin Brubeck Unhammer presented *Constraint Grammar in Apertium* and Tino Didriksen presented *Latest news*, from the compilator programmer's workbench.

Reviewers for all the papers were Kristin Hagen, Marit Julien and Anssi Yli-Jyrä. As the field is small and transparent, the organising committee deemed a blind reviewing process impossible to carry out (the authors behind all the papers were evident from already published parsers and articles). All papers were accepted.

Eckhard Bick, Kristin Hagen, Kaili Müürisep and Trond Trosterud

A constraint grammar for Faroese

Trond Trosterud
University of Tromsø

Abstract

The present paper presents ongoing work on a finite-state transducer, a Constraint Grammar disambiguator and dependency grammar for Faroese. In Faroese, the classical Germanic system of case, person and number inflection is upheld, but with somewhat more homonymy than in the closely related Icelandic. Rather than conflating homonym categories, the present morphological transducer gives a fully specified analysis of all morphological distinctions.

1 Introduction

The transducer is based upon the lemma list of *Føroysk orðabók* ((Poulsen et al., 1998))[1], and upon the grammatical description found in (Thráinsson et al., 2004).

The Faroese parser uses the computational infrastructure from the Sámi parser project (*giellatekno.uit.no*). It has the same file setup, similar makefiles, etc. There are also benefits in the opposite relation: The Sámi morphophonology testsuite was taken from work on the Faroese *twolc* file.

The Faroese morphological analyser/generator *Ffst* is a finite-state transducer. It is compiled with Xerox transducer compilers: *twolc* for Morphophonology, and *lexc* for lexicon and morphology (cf. (Beesley and Karttunen, 2003) and `http://www.fsmbook.com/`). The disambiguator *Fdis* and dependency grammar *Fdep* are written within the Constraint Grammar framework (see e.g. (Karlsson, 1990), (Karlsson et al., 1995)), and uses the 3rd generation compiler *vislcg3* ((Bick, 2000), `http://beta.visl.sdu.dk/cg3.html`).

[1]Thanks to the authors for making lemmalist and inflection codes electronically accessible, Without it this project would of course not have been realisable.

Sections 2, 3 and 4 present the grammatical analyser Ffst, the disambiguator Fdis and the dependency grammar Fdep, respectively. Section 5 gives an evaluation of the current stand of the parser, and the final section contains future perspectives and a conclusion.

2 The grammatical analyser

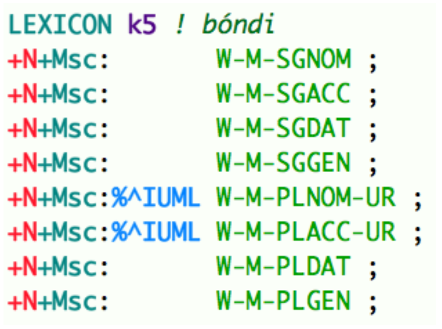
2.1 Lexicon

The Ffst lexicon uses the same inflectional codes as does (Poulsen et al., 1998). Dictionary updates and new words annotated with the same codes may thus be added directly to the Ffst. The analyser has a dynamic compounding component, genitive singular nouns have the basic noun lexicon as one of their continuation lexica, thereby creating a loop allowing any compound with genitive singular first part. This gives rise to a circular transducer, for generation this component must thus be switched off.

Ffst also contains a name guesser. The guesser detects words with capital first letter and non-Faroese phonotax. The candidate words must contain at least one vowel. The final letter cannot be a Faroese suffixal sound (*a, i, u, n, m, r, s, t* (to avoid explicit case endings). The putative name is then assigned Nom, Acc and Dat. If there is any other analysis available, the guessed form is automatically discarded. The guesser is very reliable: Of the 500 most common guesses all 500 were actually names. It is also (too) careful: Banning Faroese case suffixes from the guesser avoids analysig case-inflected forms as baseforms, but at the same time it prevents the parser from making many correct guesses.

2.2 Morphology

The morphological part of *Fdis* is built in several layers. For the nominal morphology, the first layer gives the part of speech and gender tags, and morphophonological flags, as shown in Figure 1, for the noun *bóndi* “farmer”, where the nominative and accusative plural forms show Umlaut.

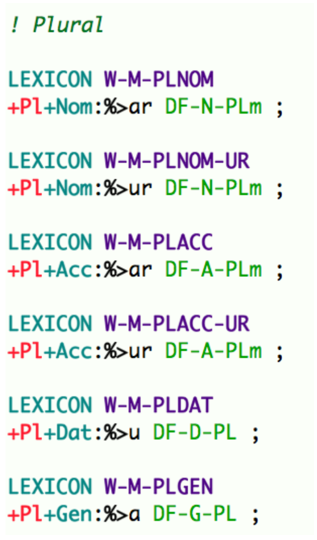


```
LEXICON k5 ! bóndi
+N+Msc:         W-M-SGNOM ;
+N+Msc:         W-M-SGACC ;
+N+Msc:         W-M-SGDAT ;
+N+Msc:         W-M-SGGEN ;
+N+Msc:%^IUML W-M-PLNOM-UR ;
+N+Msc:%^IUML W-M-PLACC-UR ;
+N+Msc:         W-M-PLDAT ;
+N+Msc:         W-M-PLGEN ;
```

Figure 1: Definiteness morphology

The dictionary contains xx nominal declension types, but including singular-only and plural-only declension patterns, and combined patterns (words declined for more than one pattern), the system totals 269 distinct first-layer continuation lexica for nouns, one of them being the $k5$ lexicon in Figure 1.

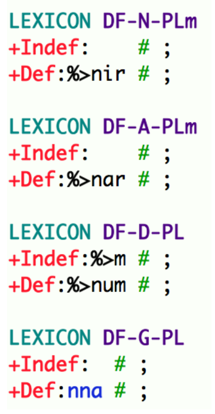
The second layer gives case and number morphology. Figure 2 gives the continuation lexica for weak masculine plurals, i.e., also for *bóndi* and the other $k5$ words.



```
! Plural

LEXICON W-M-PLNOM
+Pl+Nom:%>ar DF-N-PLm ;

LEXICON W-M-PLNOM-UR
+Pl+Nom:%>ur DF-N-PLm ;

LEXICON W-M-PLACC
+Pl+Acc:%>ar DF-A-PLm ;

LEXICON W-M-PLACC-UR
+Pl+Acc:%>ur DF-A-PLm ;

LEXICON W-M-PLDAT
+Pl+Dat:%>u DF-D-PL ;

LEXICON W-M-PLGEN
+Pl+Gen:%>a DF-G-PL ;
```

Figure 2: Second layer - case and number

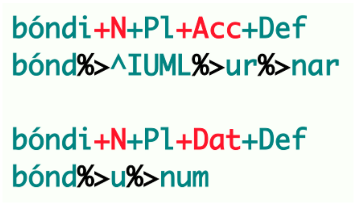
The third layer gives definiteness morphology. Due to the agglutinative nature of Faroese morphology, the lexica either only add the indefinite tag, or the definite tag and suffix. The exception is dative, which shows an *n:m* alternation. Rather than writing a morphophonological rule deleting *m* in front of *num*, the alternation is written into the morphology file.



```
LEXICON DF-N-PLm
+Indef:     # ;
+Def:%>nir # ;

LEXICON DF-A-PLm
+Indef:     # ;
+Def:%>nar # ;

LEXICON DF-D-PL
+Indef:%>m # ;
+Def:%>num # ;

LEXICON DF-G-PL
+Indef:  # ;
+Def:nna # ;
```

Figure 3: Third layer - definiteness

Applying these lexica, we get, among others the accusative and dative plural definite forms shown in Figure 4.

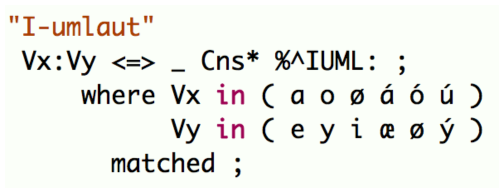


```
bóndi+N+Pl+Acc+Def
bónd%>^IUML%>ur%>nar

bóndi+N+Pl+Dat+Def
bónd%>u%>num
```

Figure 4: The resulting upper and lower lexc strings

2.3 Morphophonology

The lower part of the string pairs from the morphological transducer are then fed to a separate automaton, the morphophonological component. This automaton contains rules for morphophonological alternations, and for non-segmental morphology. The relevant rule in this context is *I-umlaut*, shown in figure 5. The rule works on strings containing any of the vowels in Vx, zero or more consonants, and the Umlaut trigger symbol ^IUML, and changes all vowels in Vx into the corresponding vowels in Vy. In this case, it changes *ó* into *ø*.



```
"I-umlaut"
 Vx:Vy <=> _ Cns* %^IUML: ;
     where Vx in ( a o ø á ó ú )
           Vy in ( e y i æ ø ý )
       matched ;
```

Figure 5: The twolc i-umlaut rule

The morphological and morphophonological transducers are then composed, and the resulting transducers gives a pairing of the upper represenation of the former and the lower represntation of the latter, graphically presented in Figure 6, with the invisible, intermediate strings shown in shaded grey.

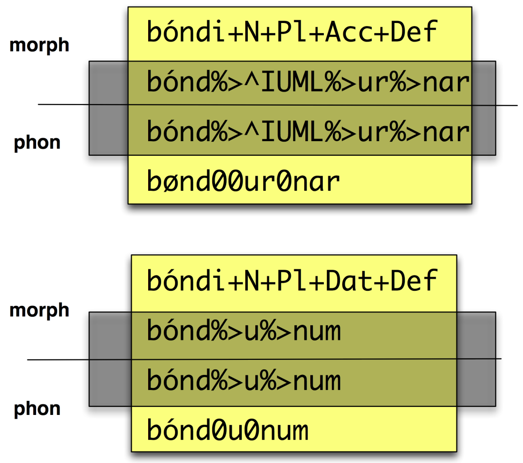


Figure 6: The transducers

Applied to all grammatical words of the lexeme *bóndi*, Ffst gives the paradigm shown in Figure 7.

bóndi+N+Msc+Sg+Nom+Indef	bóndi
bóndi+N+Msc+Sg+Acc+Indef	bónda
bóndi+N+Msc+Sg+Dat+Indef	bónda
bóndi+N+Msc+Sg+Gen+Indef	bónda
bóndi+N+Msc+Pl+Nom+Indef	bøndur
bóndi+N+Msc+Pl+Acc+Indef	bøndur
bóndi+N+Msc+Pl+Dat+Indef	bóndum
bóndi+N+Msc+Pl+Gen+Indef	bónda
bóndi+N+Msc+Sg+Nom+Def	bóndin
bóndi+N+Msc+Sg+Acc+Def	bóndan
bóndi+N+Msc+Sg+Dat+Def	bóndanum
bóndi+N+Msc+Sg+Gen+Def	bóndans
bóndi+N+Msc+Pl+Nom+Def	bøndurnir
bóndi+N+Msc+Pl+Acc+Def	bøndurnar
bóndi+N+Msc+Pl+Dat+Def	bóndunum
bóndi+N+Msc+Pl+Gen+Def	bóndanna

Figure 7: The resulting paradigm for bóndi

A list of the morphophonological rules is given in Figure 8 on page 4.

2.4 Status quo for Ffst

At present (May 2009), the Faroese morphological transducer recognises 94.3 % of all wordform tokens and 62.8 % of all wordform types in running text, for a corpus of 2.7 million words (dominated by newspaper text). The discrepancy indicates that Ffst handles common words better than rare ones.

The results could still be better, but for certain subgenres (such as the Bible), Ffst gives better results (96.3 % and 83.3 %, respectively), results good enough to evaluate the subsequent CG component. Note that even for the known text, Ffst misses approximately 16 % of the wordform types. The reason for this high number is that certain parts of the transducer are still under construction, especially parts of the irregular verbs, and of comparative and superlative forms of adjectives. Also Faroese names are missing, except the most central person names. The foreign names are mainly taken care of by the name guesser.

The top 84 missing wordforms from an 2.7 m wd corpus are shown in Figure 9 on page 4.

The 43093 missing wordforms represent 5.67% of the 2.7 mill corpus. In order to reduce the number of missing wordforms in running text by 50%, the top 2117 wordforms of the missing list would have to be added to the analyser. Important areas for lexicon improvement include the following:

- Adjectival inflection of participles, irregular adjectival forms
- Some irregular strong verbs and verb forms
- Faroese names (other than person names)
- Compounded function words
- Words missing from FO
- Plain errors

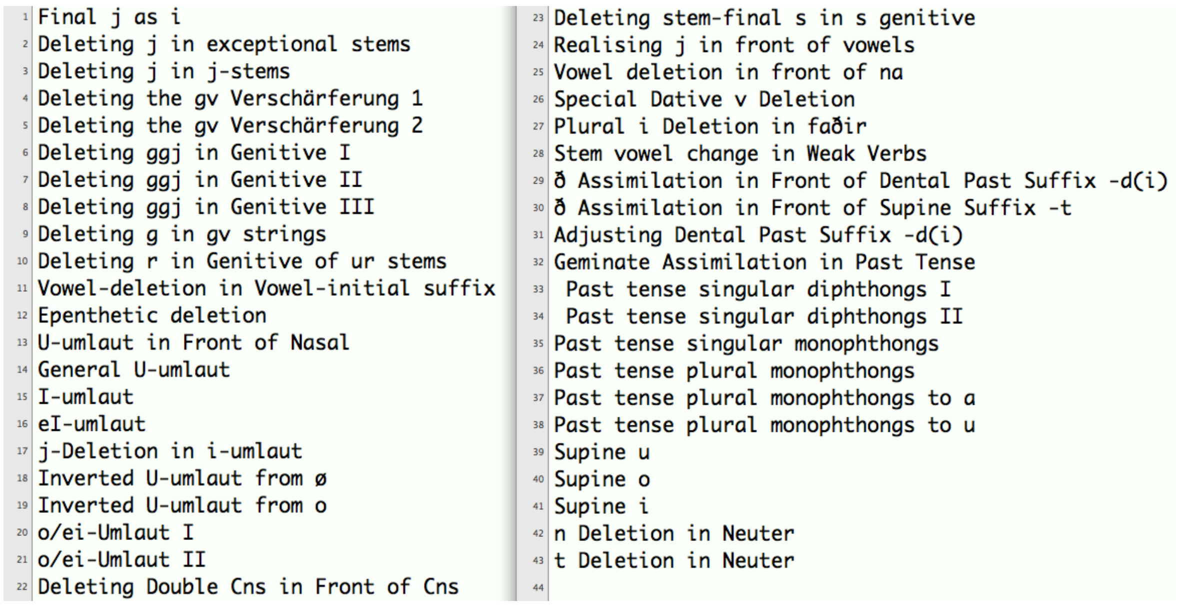
3 The Faroese disambiguator

The disambiguator (Fdis) consists of 166 rules for morphological disambiguation, 67 mapping rules, and 68 rules for disambiguation of grammatical functions. This is a small, but relatively efficient rule set, compared to the disambiguators for some other languages in Table 1[2]. For each language, the table gives number or rules, and the average numbers of readings before and after disambiguation, as applied on a compatible corpus (Genesis and the New Testament.).

3.1 Tag unification

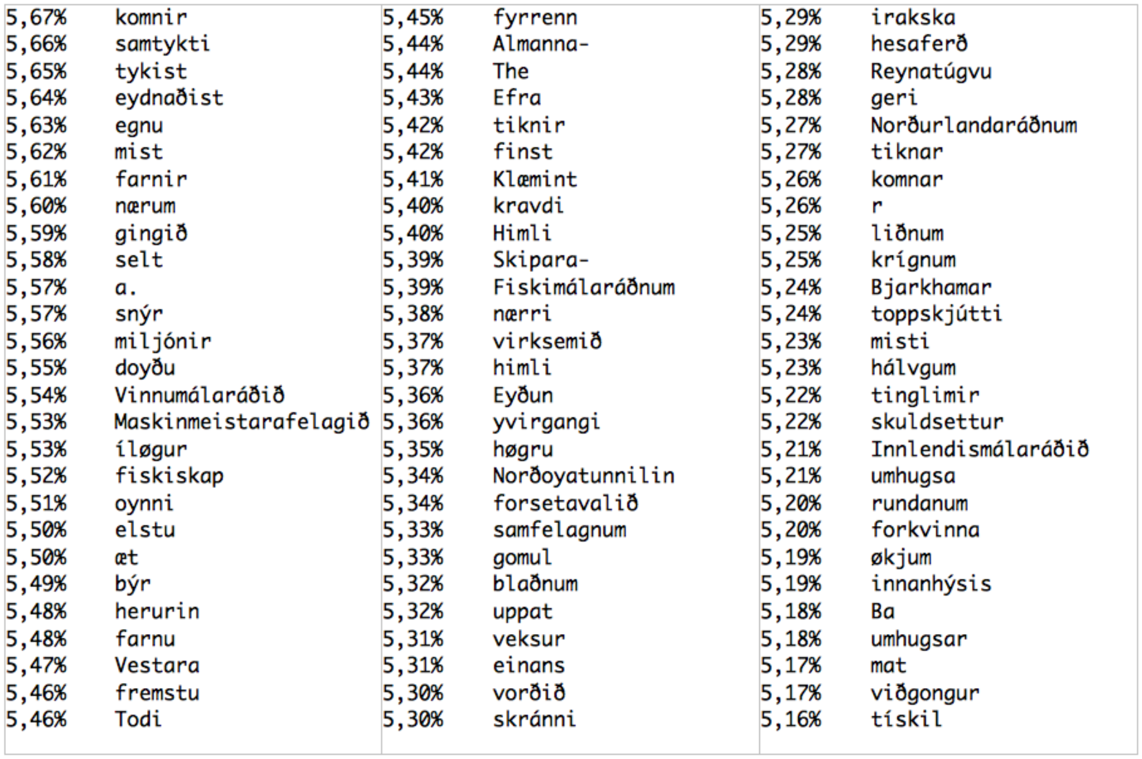
The efficiency of the Fdis ruleset illustrates the efficiency of an innovation in vislcg3, namely set

[2]The Sámi parsers are developed at the University of Tromsø (UiT), the Greenlandic parser is joint work between Oqaasileriffik and UiT, and the Bokmål parser is developed at Tesktlaboratoriet in Oslo. Thanks to Kristin Hagen for running the Bokmål analysis for this comparison.



```
Final j as i
Deleting j in exceptional stems
Deleting j in j-stems
Deleting the gv Verschärferung 1
Deleting the gv Verschärferung 2
Deleting ggj in Genitive I
Deleting ggj in Genitive II
Deleting ggj in Genitive III
Deleting g in gv strings
Deleting r in Genitive of ur stems
Vowel-deletion in Vowel-initial suffix
Epenthetic deletion
U-umlaut in Front of Nasal
General U-umlaut
I-umlaut
eI-umlaut
j-Deletion in i-umlaut
Inverted U-umlaut from ø
Inverted U-umlaut from o
o/ei-Umlaut I
o/ei-Umlaut II
Deleting Double Cns in Front of Cns
Deleting stem-final s in s genitive
Realising j in front of vowels
Vowel deletion in front of na
Special Dative v Deletion
Plural i Deletion in faðir
Stem vowel change in Weak Verbs
ð Assimilation in Front of Dental Past Suffix -d(i)
ð Assimilation in Front of Supine Suffix -t
Adjusting Dental Past Suffix -d(i)
Geminate Assimilation in Past Tense
 Past tense singular diphthongs I
 Past tense singular diphthongs II
Past tense singular monophthongs
Past tense plural monophthongs
Past tense plural monophthongs to a
Past tense plural monophthongs to u
Supine u
Supine o
Supine i
n Deletion in Neuter
t Deletion in Neuter
```

Figure 8: Twol rules



5,67%	komnir	5,45%	fyrrenn	5,29%	irakska
5,66%	samtykti	5,44%	Almanna-	5,29%	hesaferð
5,65%	tykist	5,44%	The	5,28%	Reynatúgvu
5,64%	eydnaðist	5,43%	Efra	5,28%	geri
5,63%	egnu	5,42%	tiknir	5,27%	Norðurlandaráðnum
5,62%	mist	5,42%	finst	5,27%	tiknar
5,61%	farnir	5,41%	Klæmint	5,26%	komnar
5,60%	nærum	5,40%	kravdi	5,26%	r
5,59%	gingið	5,40%	Himli	5,25%	liðnum
5,58%	selt	5,39%	Skipara-	5,25%	krígnum
5,57%	a.	5,39%	Fiskimálaráðnum	5,24%	Bjarkhamar
5,57%	snýr	5,38%	nærri	5,24%	toppskjútti
5,56%	miljónir	5,37%	virksemið	5,23%	misti
5,55%	doyðu	5,37%	himli	5,23%	hálvgum
5,54%	Vinnumálaráðið	5,36%	Eyðun	5,22%	tinglimir
5,53%	Maskinmeistarafelagið	5,36%	yvirgangi	5,22%	skuldsettur
5,53%	íløgur	5,35%	høgru	5,21%	Innlendismálaráðið
5,52%	fiskiskap	5,34%	Norðoyatunnilin	5,21%	umhugsa
5,51%	oynni	5,34%	forsetavalið	5,20%	rundanum
5,50%	elstu	5,33%	samfelagnum	5,20%	forkvinna
5,50%	æt	5,33%	gomul	5,19%	økjum
5,49%	býr	5,32%	blaðnum	5,19%	innanhýsis
5,48%	herurin	5,32%	uppat	5,18%	Ba
5,48%	farnu	5,31%	veksur	5,18%	umhugsar
5,47%	Vestara	5,31%	einans	5,17%	mat
5,46%	fremstu	5,30%	vorðið	5,17%	viðgongur
5,46%	Todi	5,30%	skránni	5,16%	tískil

Figure 9: Top 84 missing wordforms, the percentages showing the percentage of the corpus left unanalysed with a list of missing wordforms up to and including the wordform in question

Table 1: Rules and results for some CG parsers

Parser	Rules	Input	Output
North Sámi	3537	2.42	1.08
Norsk Bokmål	1964	2.13	1.17
Lule Sámi	832	2.18	1.21
Faroese	301	2.45	1.24
Greenlandic	518	2.69	1.42

unification for tags. With the set unification operator $$ it is possible to refer to a set, so that the tag that first satisfies the set must be the same as all subsequent matches of the same set. Cf. the rule (1), which refers to the set (2).

(1) SELECT $$NAGD IF (0 Det)(*1C $$NAGD BARRIER NOT-NP);

(2) SET NAGD = Nom Acc Gen Dat ;

The bulk of the rules aims at disambiguating case, number and gender within the NP. One clue as to determining the correct case is the choice of preposition, as it is for the human listener. Unfortunately, most Faroese prepositions subcategorise for more than one case. What case to choose if there is a tie is ultimately dependant upon the combination of verb and preposition. At the present stage, Fdis selects Accusative for motion verbs and change of relationship PPs, otherwise it chooses Dative.

When disambiguating running text, certain high-frequent words need special attention, both because they get multiple interpretations in the morphological component, and for their key role in the sentence. A common strategy for such words is to write specific rules just for these words. For Fdis, only approximately 15 such words have received special treatment until now, among them the pronouns *hon, vit* and the ambiguous function words *at, ið, men*. Also this is an area for improvement.

The Faroese verbal paradigm shows much homonymy. Ffst follows the practice of the reference grammars, and specifies 3 persons in the singular (also when the conjugation in question shows homonymy), but only one plural form. Naturally, disambiguating of the verbal forms rests heavily upon the person of the subject.

Mapping of grammatical functions is done on the basis of morphological cues and word order, and their disambiguation mainly on the basis of word order. The grammatical function tags are directional (the distinction @OBJ> / @<OBJ indicates whether the governing verb is to found to the right or to the left, respectively). This distinction is heavily utilised in the dependency grammar.

4 The dependency grammar

The dependency grammar quite reliably delimits NPs, and the governed constituents of P and V. Eventual errors here are due to errors in Fdis. The main obstacles for a good depencency analyses are coordination and relative clauses. Attaching appropriate constituents to the clause mother node is quite a reliable process as long as the rest of the analysis is correct. Unfortunately shortcomings in coordination and relative clause analysis, and especially the low coverage of the Ffst gives too many top nodes (2.3 alleged clausal heads per clause on average, compared to the correct 1 head/clause). Even with these shortcomings, the Fdep is already at this stage a good tool for research on basic dependency relations.

5 Evaluation

5.1 Precision and recall

The parser was tested on a small corpus of 1033 words of unseen text from a new genre (Faroese education planning). The results are shown in Table 2.

Table 2: Precision, recall, accuracy and F-ms for a test corpus

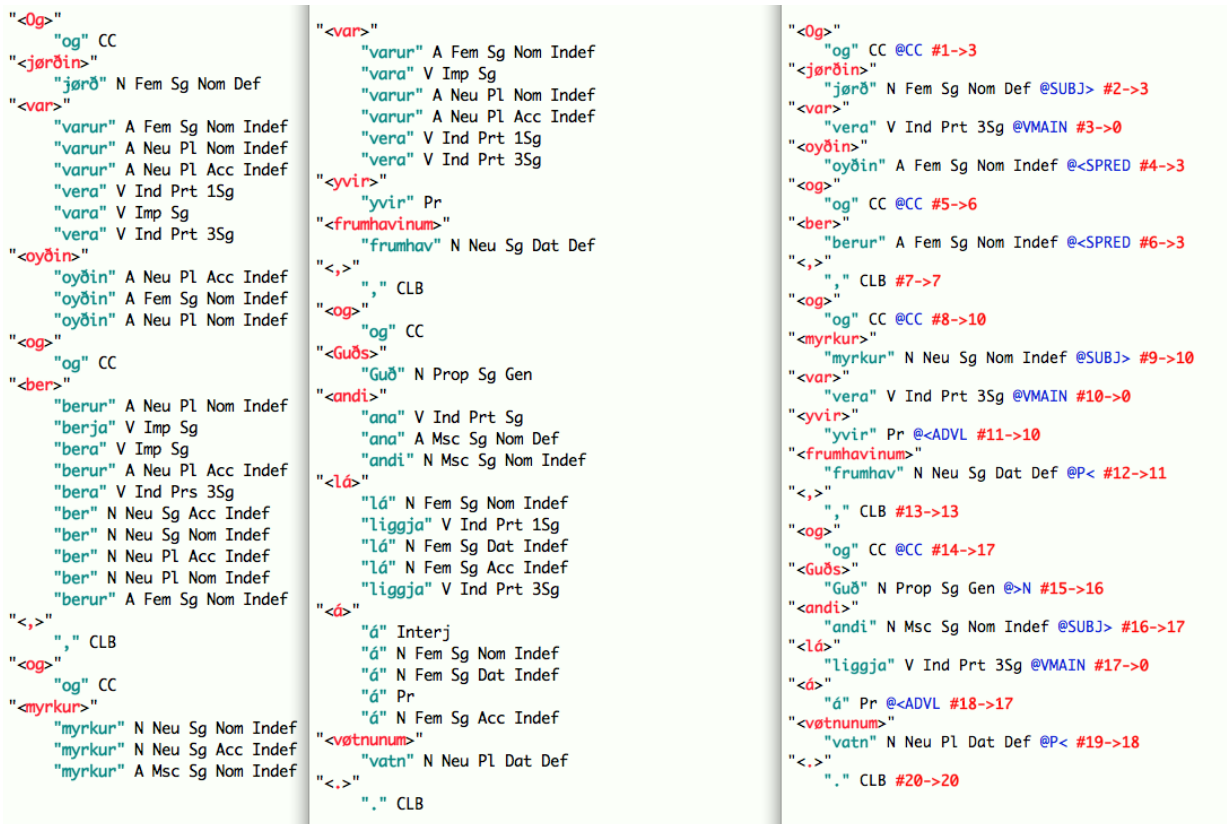
Error type	tp	fp	tn	fn
Morphology	2048	369	2501	101
Syntax	1902	515	2357	245
Dependency	724	316	0	0
	prec	rec.	acc.	F-ms.
Morphology	0.85	0.95	0.91	0.90
Syntax	0.79	0.89	0.85	0.83
Dependency	0.7	1	0.7	0.82

Thus, Fdis is work in progress

As an illustration of the Fdis output, consider Figure 10 on page 6. The two leftmost columns give the output from Ffst, with all possible readings. The third column gives the output from Fdis and Fdep, with ambiguity removed, and grammatical functions and dependency added.

5.2 Processing speed

When it comes to processing speed, it seems that the bottleneck in the system is the disambigua-



```
"<Og>"
	"og" CC
"<jørðin>"
	"jørð" N Fem Sg Nom Def
"<var>"
	"varur" A Fem Sg Nom Indef
	"varur" A Neu Pl Nom Indef
	"varur" A Neu Pl Acc Indef
	"vera" V Ind Prt 1Sg
	"vara" V Imp Sg
	"vera" V Ind Prt 3Sg
"<oyðin>"
	"oyðin" A Neu Pl Acc Indef
	"oyðin" A Fem Sg Nom Indef
	"oyðin" A Neu Pl Nom Indef
"<og>"
	"og" CC
"<ber>"
	"berur" A Neu Pl Nom Indef
	"berja" V Imp Sg
	"bera" V Imp Sg
	"berur" A Neu Pl Acc Indef
	"bera" V Ind Prs 3Sg
	"ber" N Neu Sg Acc Indef
	"ber" N Neu Sg Nom Indef
	"ber" N Neu Pl Acc Indef
	"ber" N Neu Pl Nom Indef
	"berur" A Fem Sg Nom Indef
"<,>"
	"," CLB
"<og>"
	"og" CC
"<myrkur>"
	"myrkur" N Neu Sg Nom Indef
	"myrkur" N Neu Sg Acc Indef
	"myrkur" A Msc Sg Nom Indef
```

```
"<var>"
	"varur" A Fem Sg Nom Indef
	"vara" V Imp Sg
	"varur" A Neu Pl Nom Indef
	"varur" A Neu Pl Acc Indef
	"vera" V Ind Prt 1Sg
	"vera" V Ind Prt 3Sg
"<yvir>"
	"yvir" Pr
"<frumhavinum>"
	"frumhav" N Neu Sg Dat Def
"<,>"
	"," CLB
"<og>"
	"og" CC
"<Guðs>"
	"Guð" N Prop Sg Gen
"<andi>"
	"ana" V Ind Prt Sg
	"ana" A Msc Sg Nom Def
	"andi" N Msc Sg Nom Indef
"<lá>"
	"lá" N Fem Sg Nom Indef
	"liggja" V Ind Prt 1Sg
	"lá" N Fem Sg Dat Indef
	"lá" N Fem Sg Acc Indef
	"liggja" V Ind Prt 3Sg
"<á>"
	"á" Interj
	"á" N Fem Sg Nom Indef
	"á" N Fem Sg Dat Indef
	"á" Pr
	"á" N Fem Sg Acc Indef
"<vøtnunum>"
	"vatn" N Neu Pl Dat Def
"<.>"
	"." CLB
```

```
"<Og>"
	"og" CC @CC #1->3
"<jørðin>"
	"jørð" N Fem Sg Nom Def @SUBJ> #2->3
"<var>"
	"vera" V Ind Prt 3Sg @VMAIN #3->0
"<oyðin>"
	"oyðin" A Fem Sg Nom Indef @<SPRED #4->3
"<og>"
	"og" CC @CC #5->6
"<ber>"
	"berur" A Fem Sg Nom Indef @<SPRED #6->3
"<,>"
	"," CLB #7->7
"<og>"
	"og" CC @CC #8->10
"<myrkur>"
	"myrkur" N Neu Sg Nom Indef @SUBJ> #9->10
"<var>"
	"vera" V Ind Prt 3Sg @VMAIN #10->0
"<yvir>"
	"yvir" Pr @<ADVL #11->10
"<frumhavinum>"
	"frumhav" N Neu Sg Dat Def @P< #12->11
"<,>"
	"," CLB #13->13
"<og>"
	"og" CC @CC #14->17
"<Guðs>"
	"Guð" N Prop Sg Gen @>N #15->16
"<andi>"
	"andi" N Msc Sg Nom Indef @SUBJ> #16->17
"<lá>"
	"liggja" V Ind Prt 3Sg @VMAIN #17->0
"<á>"
	"á" Pr @<ADVL #18->17
"<vøtnunum>"
	"vatn" N Neu Pl Dat Def @P< #19->18
"<.>"
	"." CLB #20->20
```

Figure 10: And the earth was waste and void; and darkness was upon the face of the deep: and the Spirit of God moved upon the face of the waters

tor. Even though it is much smaller than most CG grammars, it performs clearly worse than all the other parts of the pipeline. The reason for this might be the extensive use of set unification.

Table 3: Processing speed, measured on 100000 words of running text, on a 2,4 GHz laptop

Process	Program	Words/sec
Preprocessing	perl	10446
Morphological lookup	fst	42992
Postprocessing	perl	13017
Disambiguation	vislcg3	2042
Dependency	vislcg3	18814

6 Conclusion

The Faroese grammatical analyser presented here is still in the making. It still shows that with a modest number of CG rules, one may achive results good enough for several languaguage processing tasks. Future improvements of the analyser will concentrate upon key parts of the Ffst, upon disambiguation of complex syntactic patterns, and upon the dependency analysis of coordination and relative clauses.

References

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. Studies in Computational Linguistics. CSLI Publications, Stanford, California.

Eckhard Bick. 2000. *The Parsing System "Palavras": Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework*. Aarhus University Press, Aarhus.

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

Fred Karlsson. 1990. Constraint grammar as a framework for parsing running text. In *13th International Conference on Computational Linguistics (COLING-90)*, pages 168–173, Helsinki.

Jóhan Hendrik W. Poulsen, Marjun Simonsen, Jógvan í Lon Jacobsen, Anfinnur Johansen, and Zacharis Svabo Hansen. 1998. *Føroysk orðabók*, volume 1-2. Føroya Fróðskaparfelag, Tórshavn.

Höskuldur Thráinsson, Hjalmar P. Petersen, Jógvan í Lon Jacobsen, and Zacharis Svabo Hansen. 2004. *Faroese: An overview and reference grammar*. Føroya Fróðskaparfelag, Tórshavn.

A Dependency Constraint Grammar for Esperanto

Eckhard Bick
Institute of Language and Communication
University of Southern Denmark
eckhard.bick@mail.dk

Abstract

This paper presents a rule-based formalism for dependency annotation within the Constraint Grammar framework, implemented as an extension of the open source CG3 compiler. As a proof of concept we have constructed a complete dependency grammar for Esperanto, building on morphosyntactically annotated input from the EspGram parser. The system is described and evaluated on a test corpus. With a 4% error rate, and most errors caused by simple error propagation from the morphosyntactic input module, our system has proven robust enough to be integrated into real life applications, such as the Lingvohelpilo spell- and grammar-checker.

1 Introduction

Traditionally, Constraint Grammar (Karlsson et al. 1995) as a descriptive system, has regarded syntax as an extension of morphology, with a shallow syntax based on function tags built on case markers, word order and contextual constraints. This approach to syntax efficiently exploits lexico-morphological clues, and the tag-based annotation allows the grammarian to treat syntax as a disambiguation technique similar to the one used for morphological disambiguation. However, function is only an indirect marker for the relation between words, and it is difficult to express the structural relations of deeper syntax in this fashion. As a first approximation, dependency direction markers were used for the dependents in noun phrases (e.g. @N> or @>N), adjective phrases (@A> or @>A) and prepositional phrases (@P<), a descriptive principle later generalized to clause level functions and subclauses (Bick 2000). In this convention, some obvious underspecifications arise, such as the distinction between short and long attachment in np's, and the scope of coordinators. Nevertheless, two different methods were developed to create full syntactic trees from shallow CG function tags. The first (Bick 2003) uses higher level phrase structure grammars with function tags as terminals, and resolves underspecifications in a generative way. The second, and more robust (Bick 2005), uses ordinary CG rules to add secondary attachment markers (e.g. <np-close>, <np-long>, <co-acc>, <cjt-first>) to resolve underspecification, and creates dependency trees through successive attachment rules. However, the method used an external formalism, with a specially designed dependency rule compiler that also handled issues like uniqueness, circularity and coordination chains.

This paper describes an effort to move this last, tree-building step into the realm of Constraint Grammar proper, thus allowing the user to exploit CG's powerful contextual methodology in the process, to better integrate dependency and functional syntax and to achieve some control over dependency interaction not fully implementable in an the external formalism. The new CG extension was then used to create a dependency CG grammar for Esperanto, and it is this grammar that will be described and evaluated here. The module deep linguistic processing, and can thus be seen as facilitation stepping stone both for further, syntax-dependent annotation (e.g. anaphora, semantic roles) and for various applicative purposes such as machine translation. Currently, the grammar is used in the newly-developed Esperanto grammar checker, Lingvohelpilo (http://lingvohelpilo.ikso.net/), where it provides important contextual information for the checking of accusative/nominative case endings and transitivity affixes, as well as for the identification of long-distance agreement errors, e.g. between subject and subject complement.

2 The formalism

In order to accommodate for dependency, 2 new operators, SETPARENT and SETCHILD, were introduced to GrammarSoft's open-source CG3 compiler (Didriksen 2007), establishing dependency arcs from daughter to mother, or mother to daughter, respectively, addressing one in the SETPARENT/SETCHILD field and the other in a TO field. Both fields of the rule can be independently conditioned with CG contexts in the usual way. The first field works like the TARGET of a MAPping rule, while the TO-end of the dependency is specified by a context condition itself – as seen from the TARGET position. In the case of a LINKed condition, the attachment point can be marked (with a special A operator) as any of the individual contexts checked and "passed". As a default, the dependency arc will attach to the *last* condition of the LINK chain if it can be instantiated. As in the older, external dependency compiler, dependency arcs are expressed as number tokens of the type #n->m, where n is the token ID of the daughter and m the token ID of the mother. Internally, the CG3 compiler uses unique, running IDs (necessary for cross-sentence relations such as anaphora or discourse relations), but in standard dependency output, sentence windows boundaries are respected, using relative IDs. The notation is information equivalent to constituent tree structures, and has been successfully converted into various exchange formats, such as TIGER xml and the VISL cross-language format (constituent trees), as well as MALT xml and CoNNL field format (dependency).

The rule below is an example of a dependency-creating rule for prenominal dependents (@>N), attaching to np-heads (@NP-HEAD) or nouns in the nominative (N NOM), to the right (*1).

```
(a) SETPARENT @>N TO (*1 @NP-HEAD
    OR (N NOM) BARRIER PRP) ;
```

Once established, dependency arcs can be used by later rules – even by other dependency-mapping rules – using three types of dependency relators: p (parent), c (child) and s (sibling). The p-, c- and s-relators replace what would otherwise be position markers in a traditional CG context. Thus, rule (a) exploits semantic prototype roles to select +HUM subjects in the presence of cognitive verbs, while (b) implements the syntactic uniqueness principle for direct objects (@ACC).

```
(a) SELECT (%hum) (0 @SUBJ) (p <Vcog>)
(b) SELECT (@ACC) (NOT s @ACC)
(c) ... (*-1 N LINK c DEF)      -> definite np
    recognized through dependent
(d) ADD (§AG) TARGET @SUBJ (p V-HUM
    LINK c @ACC LINK 0 N-NON-HUM) ;
```

Rule (c) is an example of a rule context used to recognize a definite np through its determiner, and (d) assigns the semantic role tag of agent (§AG) to subjects of "human" verbs with a non-human direct objects.

3 The Esperanto grammar

The preposition barrier (PRP) in the np rule in the last section is a sensible safety measure for English and French, but fails to account for pre-nominal pp's as they do occur in e.g. Esperanto and German. The next rule therefore allows prenominals to search right (**1) *across* the first np-head to a later one that is not part of a prenominal pp (as implied by @P<). Note that the SET target has its own condition excluding targets that already have a parent (using the (*) convention for "any tag"). Since rule application order supersedes token order, this will have the effect of not undoing the pp-free prenominal attachments already mapped by the first rule.

```
SETPARENT @>N (NOT p (*))
    TO (**1 @NP-HEAD OR (N NOM))
    (NOT 0 @P<) ;
```

At the clause level, it is a fair assumption that all left-pointing functions attach to the closest main verb (&MV), unless an intervening subclause ending is marked by punctuation (CLB):

```
SETPARENT @<FUNC
        TO (*-1 &MV BARRIER CLB) ;
```

For right-pointing functions (@FUNC>), the blocking condition is a subclause "complementizer" (relative/interrogative pronoun or a subordinating conjunction), which – unlike English - is an obligatory feature in Esperanto. In a subsequent rule, long-distant attachment across relative clauses can be performed for still unattached subjects (NOT p (V)), by linking to the next main verb that does not already have a subject (NOT c @SUBJ>):

```
SETPARENT @SUBJ> (NOT p (V))
    TO (**1 &MV)
    (*-1 NON-V LINK NOT 1 PCP)
    (NOT c @SUBJ>)
```

Note the additional context condition in the TO field that identifies the first verb in a possible verb chain and conditions it as not being a participle – since participle clauses don't have left subjects.

In our grammar, coordination is handled as "parallel" attachment, not chained Mel'cuk-style, and in the absence of uniqueness-demanding contexts, ordinary attachment rules will therefore handle coordination, too. However, the clause boundary barrier discussed before poses a problem where a chain of conjuncts contains not only a coordinator, but also commas. Therefore, a somewhat more complicated rule becomes necessary to attach comma-isolated conjuncts:

```
SETPARENT $$@FUNC (NOT p (V))
    TO (*-1 IT BARRIER NON-PRE-N/ADV
    LINK *-1 $$@FUNC BARRIER @FUNC
    LINK p (V)) ;
```

This rule exploits the new uniqueness feature in CG3 to attach any as yet unattached function if the *same* function ($$@FUNC) can be found to the left of an immediately adjacent (BARRIER NON-PRE-N/ADV) iterator (IT = coordinator or comma), with no other functions in between (BARRIER @FUNC). The dependency head will be the mother (p V) of the same-function antecedent found. Further rules, not discussed here, attach the coordinator token itself, and assign secondary conjunct tags to all conjuncts, in order to distinguish between first and later conjuncts should the need for a Mel'cuk-style transformation arise.

4 Evaluation

Compared to the complexity of morphological and syntactic CGs, our dependency CG module is strikingly rule efficient, achieving robust annotation with just 66 rules, compared to the thousands of rules in lower-level CGs, and the couple of hundred rules in a CG-based PSG. Of course, it has to be born in mind, that our rules rely heavily on syntactic functions and attachment direction markers introduced by preceding CG modules. Also, at the time of writing, we have not yet incorporated the distinction between close and long postnominal attachment, ellipsis and quoted sentences which will unavoidably add to the number of rules.

Speedwise, CG-dependency is also quite efficient. A 75.000 word corpus consisting of 50% news magazine text and 50% classical texts, was analyzed with the EspGram tagger (Bick 2007) at the syntactic-functional level, and the annotated corpus was then tagged with our dependency CG on a 2.4 GHz laptop. In this experiment, the analysis chain up to the syntactic function level ran at 72 words/s, while the dependency level alone ran at 6336 words/s, using 10.2 % of overall processing time. Compared to the external dependency system (608 words/s), this implies a speed improvement by almost one order of magnitude.

A rough inspection of annotation results for a sample of 1000 words indicate an overall error rate for the dependency annotation of about 4%. Of these, about half were attachment failures (no mothernode for non-topnode functions), half were wrong attachments (wrong daughter-mother relation). With most errors being caused by syntactic-function errors in the input, the error rate of the dependency module itself was very low, under 1%.

5 Conclusion and outlook

Given the necessary formal changes to the CG compiler software, it appears to be feasible, even with a relatively small set of rules, to handle the creation of dependency tree structures for CG-analyzed input within the CG formalism itself. Our experiments with such a grammar for use in an Esperanto spell- and grammar-checker produced robust results, both quantitatively and qualitatively. In particular, the dependency module proved to be considerably more robust than the syntactic function module, inheriting most of its errors from the former. We therefore believe that CG dependency modules can be created with comparatively little effort, to turn existing CG function annotations into dependency treebanks without substantial loss of information. Future research should allow us to shed light on the question to what degree our dependency grammar, given a compatible set of morphological and syntactic input tags, is language independent - as the size and simple nature of our rule set indicates.

References

Bick, Eckhard (2000), "The Parsing System PALAVRAS - Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework", Aarhus: Aarhus University Press

Bick, Eckhard. (2003) A CG & PSG Hybrid Approach to Automatic Corpus Annotation, In: Kiril Simow & Petya Osenova (eds.), Proceedings of SProLaC2003 (at Corpus Linguistics 2003, Lancaster), pp. 1-12

Bick, Eckhard. (2005) "Turning Constraint Grammar Data into Running Dependency Treebanks". In: Civit, Montserrat & Kübler, Sandra & Martí, Ma. Antònia (red.), Proceedings of TLT 2005 (4th Workshop on Treebanks and Linguistic Theory, Barcelona, December 9th - 10th, 2005), pp.19-27

Bick, Eckhard (2007), Tagging and Parsing an Artificial Language: An Annotated Web-Corpus of Esperanto, In: *Proceedings of Corpus Linguistics 2007, Birmingham, UK*. Electronically published at (http://ucrel.lancs.ac.uk/publications/CL2007/, Nov. 2007)

Didriksen, Tino (2003). "Constraint Grammar Manual", http://beta.visl.sdu.dk/cg3/single/

Karlsson, Fred et al. (1995): Constraint Grammar - A Language-Independent System for Parsing Unrestricted Text. Natural Language Processing, No 4. Berlin & New York: Mouton de Gruyter.

Appendix: Annotation sample

Post 12 jaroj da reformoj, la efikeco de la ĉeĥa ekonomio ne signife transpaŝas la nivelon atingitan en la jaro 1989.
(Ater 12 years of reforms, the efficiency of the chech economy has not significantly surpassed the level reached in [the year of] 1989,)

Post	[post] <*> PRP @ADVL> #1->14
12	[12] <card> <cif> NUM P @>N #2->3
jaroj	[jaro] <dur> <per> N P NOM @P< #3->1
da	[da] PRP @N< #4->3
reformoj	[reformo] <sem-c> <act> N P NOM @P< #5->4
la	[la] ART @>N #6->7
efikeco	[efikeco] <f> N S NOM @SUBJ> #7->14
de	[de] PRP @N< #8->7
la	[la] ART @>N #9->11
cxehxa	[cxehxa] <jnat> ADJ S NOM @>N #10->11
ekonomio	[ekonomio] <domain> N S NOM @P< #11->8
ne	[ne] <amod> <setop> ADV @>A #12->13

```
signife      [signife] ADV @ADVL> #13->14
transpasxas  [transpasxi] <mv> <vt>V PR @FS-STA
             #14->0
la           [la] ART @>N #15->16
nivelon      [nivelo] <ac> N S ACC @<ACC #16->14
atingitan    [atingi] <mv> <vt> V PCP PAS IMPF ADJ
             S ACC @ICL-N< #17->16
en           [en] PRP @<ADVL #18->17
la           [la] ART @>N #19->20
jaro         [jaro] <dur> <per> N S NOM @P<
             #20->18
1989         [1989] <year> <card> <cif> NUM S @N<
             #21->20
$.
```

The following fields are used in the annotation scheme, and expressed as feature attribute pairs in xml: wordform, [base form/lemma], <semantics>, @syntactic_function, #dependency-link

(*part of speech tags:* N=noun, V=verb, ADJ=adjective, ADV=adverb, PRP=preposition, ART=article, NUM=numeral; *inflexion:* S=singular, P=plural, NOM=nominative, ACC=accusative, PCP=participle, PAS=passive, PR=present tense, IMPF=past tense; *syntactic function:* @SUBJ=subject, @ADVL=adverbial, @ACC=direct object, @>N=pre-nomina modifier, @N<=post-nominal modifier, @P<=argument of preposition, @ICL=non-finite clause, @FS=finite clause, @STA=statement; *semantic prototypes:* <dur> duration, <ac> abstract countable, <domain> domain, <sem-c> semantic product, <act> action, <f> feature, <jnat> nationality, <mv> main verb; *valency:* <vt> transitive)

Constraint Grammar in Dialogue Systems

Lene Antonsen
University of Tromsø
Norway
`lene.antonsen`
`@uit.no`

Saara Huhmarniemi
University of Tromsø
Norway
`saara.huhmarniemi`
`@helsinki.fi`

Trond Trosterud
University of Tromsø
Norway
`trond.trosterud`
`@uit.no`

Abstract

This article discusses and gives examples of the use of Constraint Grammar as parser engine in parser-based CALL programs for North Sámi. The parser locates grammatical errors in a question-answer program and a dialogue program, and is also used for navigating inside the dialogue.

1 Introduction

The present paper discusses the use of Constraint Grammar (vislcg3) in two different dialogue systems for learning North Sámi: *Vasta* – a QA-drill with open questions, and *Sahka* – a dialogue between program and user within a scenario. The underlying pedagogical goals for both programs are exercising verb inflection, choosing the correct case, and extending the vocabulary of the student.

Constraint Grammar (CG) rules are used for adding tutorial feedback about grammatical errors, navigating in the *Sahka*-dialogue based on the user's answers, and for identifying parts of the user's answer for use in variables later in the dialogue.

Our leading idea was to utilize our existing analyser for Sámi when developing pedagogical programs for language instruction. With vislcg3 we had the possibility of making an intelligent tutoring system with sophisticated error analysis where student tasks could go beyond multiple-choice or string matching algorithms.

Sámi is a language with complex morphology, and it demands much practising before the student reaches necessary skills. However, since Sámi is a minority language, it is common that Sámi students do not receive enough opportunities to practise the language in a natural way. There is also a lack of teaching materials. Therefore, programs accessible on the Internet may be a supplement to the instruction given at school or in universities.

In the following section we describe the basic algorithm for generating questions for *Vasta* and analysing user's input in *Vasta* and *Sahka*. Section 3 shows how CG is used for navigation in the dialogues in *Sahka*, and section 4 shows how tutorial feedback is given with the help of CG rules. In section 5 we present an evaluation of how the system works in real life. The final sections present future perspectives and a conclusion. The programs are available on a web-based learning platform at internet (`http://oahpa.uit.no/`), which contains six programs (Antonsen, Huhumarniemi and Trosterud, 2009).

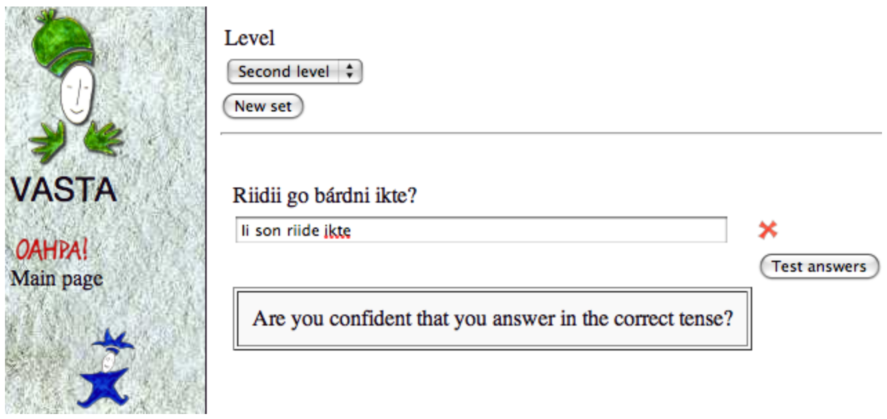
2 The system

2.1 Basic grammatical analysis

The basic grammatical analysis of North Sámi is done with finite state transducers (fst) and a constraint grammar parser made at UiT. The relevant resources are the following:

- a morphological fst analyser/generator, compiled with the Xerox compiler *xfst* (Beesley and Karttunen, 2003).
- a morphological disambiguator based on constraint grammar with 3300 manually written rules and a syntactic analyser which adds grammatical function (vislcg3).

The CG parser framework shows extraordinary results for free-text parsing, and Vislcg3 is also used in the VISL-suite of games developed at SydDansk Universitet for teaching grammatical analysis on the Internet (`http://visl.sdu.dk/`). One of their programs accepts free user input in some of the 7 supported languages. The input is analysed or changed into grammar exercises (Bick, 2005).



Riidii go bárdni ikte?

Ii son riide ikte

Are you confident that you answer in the correct tense?

Figure 1: A generated question and a user's answer in *Vasta*. ("Did the boy ride yesterday?" "No, yesterday he does not.")

2.2 Sentence generator

The question-answer drill *Vasta* consists of randomly chosen questions – yes/no-questions and wh-questions. In order to be able to create a large number of potential tasks, we implemented a sentence generator. With the generator we can easily offer variation to the user, instead of tailoring every task with ready-made questions.

A template question matrix contains two types of elements: constants and grammatical units for words selected from the pedagogical lexicon, constrained by semantic sets. The pedagogical lexicon forms a collection of about 2400 words that are considered relevant for the learners of North Sámi in schools and universities. The dialectal variation is taken into account in the lexicon as well as in the morphological generator, and the user may choose eastern or western dialect for the tasks. The sentence generator handles agreement, e.g. between subject and the main verb.

Figure 2 shows a question template in which the main verb (MAINV) is fixed to indicative past tense, but the person and number inflection may vary freely. In Figure 1 on page 2 the same template is realised as a task in *Vasta*. The user's answer triggers a feedback message about the tense of the main verb. Since the content of the MAINV and SUBJ are drawn from the lexicon, the example template may generate around 15 000 different questions.

The question matrices are marked for level, corresponding to the level option chosen by the

```
<q level="2" id="go_ikte">
  <qtype>PRT</qtype>
  <question>
    <text>MAINV go SUBJ ikte</text>
    <element id="MAINV">
  <grammar tag="V+Ind+Prt+Person-Number"/>
  <sem class="ACTIVITY"/>
    </element>
    <element id="SUBJ">
  <sem class="HUMAN"/>
  <grammar pos="N"/>
    </element>
  </question>
</q>
```

Figure 2: A question template (MAINV question-particle SUBJ yesterday).

user, e.g. the basic level 1 has only indicative and no past tense. Because of this we have to fix the inflections in every template to some extent, and there are as many as 111 matrix questions.

2.3 The analysing process

Both the question and the answer are analysed with the morphological analyser and then the result is postprocessed to cg3-format and passed to the CG3 rule component (cf. Figure 3). The question and user's answer pairs are merged, and analysed as one text string. The question mark in the question is exchanged for a special symbol ("^qst" or "^sahka" QDL), as shown in the analysed question-answer pair in Figure 5 on page 4. We use these symbols, rather than the question mark itself, in order not to introduce a sentence delimiter in the analysis, since we want to refer to the question and the answer separately in the rules (left or right side of the QDL), but also treat the question-answer part as one unit. Many of the constraints are based upon the grammar and semantics of the question – e.g. the tense and person inflection of the verb, the case of NP in the answer and so on. The question itself restricts the possible interpretation of the input.

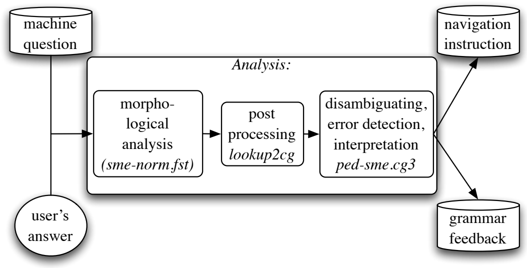

machine question
navigation instruction
Analysis:
morpho-logical analysis (sme-norm.fst)
post processing lookup2cg
disambiguating, error detection, interpretation ped-sme.cg3
user's answer
grammar feedback

Figure 3: Schematical view of the process.

The vislcg3-rule set consists of two parts. The first part is a rule set, which disambiguates the user's input only to a certain extent. The rule set is relaxed compared to the ordinary disambiguator, in order to be able to detect relevant readings despite of a certain degree of grammatical and orthographic errors in the input. The second part of the rule set contains rules for giving feedback to grammatical errors, and rules for navigating to the next question or utterance in the dialogue, based on the user's answer. In this paper, we concentrate on the rules for giving feedback for the user and navigating in the dialogue.

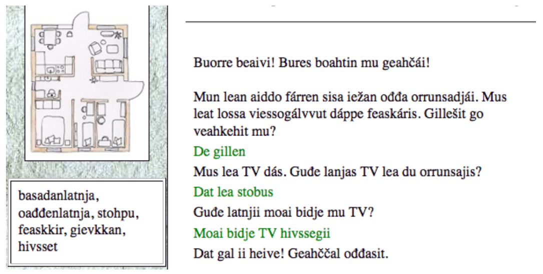
3 Navigating in the dialogues

In the *Sahka* dialogues the main goal has been to create a feeling of a natural dialogue. One of the ways to achieve that goal is reacting to the user's input. When the input is morphologically analyzed, the CG rules are used for assigning tags to the question-answer pairs, which are then used for selecting appropriate questions and navigating in the dialogue. The dialogues deal with different topics. The "first meeting dialogue", for example, treats topics such as age, family, working place/school, car and so on. Navigation between the topics is achieved by recognizing and tagging the content of the user's answer in CG rules and providing the analysis to the *Sahka*-engine. In addition, it is possible to assign a target tag to certain information types; the system may e.g. collect name, car brand and so on, and use it as a variable in the follow-up questions.

The CG rules used in the dialogue processing may be divided into two types: general rules that may target any question-answer pair and question-specific rules that are tailored for a specific question.

3.1 Rules for specific questions

Since the functionality of *Sahka* is more dependent upon correct analysis of the content of user's answer, the questions in the dialogues do not vary freely as in *Vasta*. Every question is a text string and has its own unique name assigned to the QDL. This enables writing question specific CG rules and accessing the question from other questions.


basadanlatnja, oađđenlatnja, stohpu, feaskkir, gievkkan, hivsset

Buorre beaivi! Bures boahtin mu geahčái!

Mun lean aiddo fárren sisa iežan ođđa orrunsadjái. Mus leat lossa viessogálvvut dáppe feaskáris. Gillešit go veahkehit mu?

De gillen

Mus lea TV dás. Guđe lanjas TV lea du orrunsajis?

Dat lea stobus

Guđe latnjii moai bidje mu TV?

Moai bidje TV hivssegii

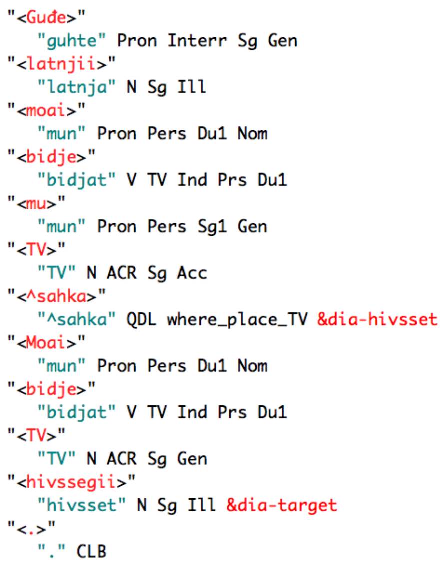
Dat gal ii heive! Geahččal ođđasit.

Figure 4: From *Sahka*. ("In which room should we place the TV?" "We should place it in the toilet." "That is not a good idea. Try again.")

Consider an example dialogue from *Sahka*. In Figure 4 the setting is a visit to a friend who has moved into a new flat, and needs a helping hand with moving the furniture. We have come to the third question and the next question in the dialogue is selected depending on the answer. In Figure 5 the analysis assigns two navigation tags to the question-answer pair. The rule for assigning the tag *&dia-hivsset* is shown in Ex. (1), the other one is explained in section 3.2.

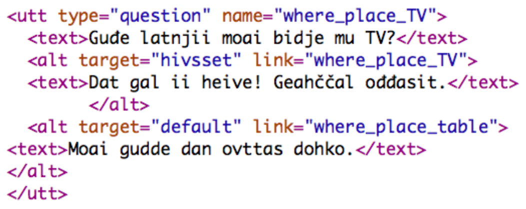
(1)
```
MAP (&dia-hivsset) TARGET QDL IF
(0 (where_place_TV))
(*1 ("hivsset") BARRIER Neg OR ROOMS) ;
```

This special rule for the question with the identifier *where_place_TV* adds the tag *&dia-hivsset* to the QDL in the question-answer pair if the answer contains the word *hivsset* (toilet). The barrier prevents the rule from working if the negation verb or a word from the set denoting rooms intervenes between the QDL and the word *hivsset*. The barrier will prevent assigning the tag to answers, which negate the possibility of putting the TV to the toilet, or giving the toilet as only one of more possibilities.



```
"<Guđe>"
   "guhte" Pron Interr Sg Gen
"<latnjii>"
   "latnja" N Sg Ill
"<moai>"
   "mun" Pron Pers Du1 Nom
"<bidje>"
   "bidjat" V TV Ind Prs Du1
"<mu>"
   "mun" Pron Pers Sg1 Gen
"<TV>"
   "TV" N ACR Sg Acc
"<^sahka>"
   "^sahka" QDL where_place_TV &dia-hivsset
"<Moai>"
   "mun" Pron Pers Du1 Nom
"<bidje>"
   "bidjat" V TV Ind Prs Du1
"<TV>"
   "TV" N ACR Sg Gen
"<hivssegii>"
   "hivsset" N Sg Ill &dia-target
"<.>"
   "." CLB
```

Figure 5: Assignment of navigation tags is done together with the disambiguation.



```
<utt type="question" name="where_place_TV">
  <text>Guđe latnjii moai bidje mu TV?</text>
  <alt target="hivsset" link="where_place_TV">
  <text>Dat gal ii heive! Geahččal ođđasit.</text>
        </alt>
  <alt target="default" link="where_place_table">
<text>Moai gudde dan ovttas dohko.</text>
</alt>
</utt>
```

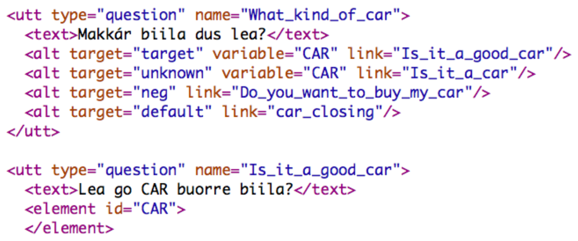
Figure 6: From the a dialogue file. ("In which room should we place the TV?" Alt. 'toilet': "That is not a good idea. Try again." Default: "We carry it there together.")

When the *Sahka*-engine reads the CG-output, it recognizes the dia-tag and searches for a next instruction based on the tag. Every question contains links to alternative questions that are selected based on the recognized tag. In addition, there is a default link in case a navigation tag was not present in the CG-input. In Figure 6 there are two alternative links for the answers to the question in Figure 5. One of them is connected to the *&dia-hivsset* tag and will give the answer "That is not a good idea. Try again." The other link is default and leads to the next question in the dialogue.

Another example of a question specific dialogue navigation rule comes from yes/no-questions where the user often provides more information than what was asked for. E.g. to the question 'Do you have children?', the user can answer 'Yes, I have two children.' In the dialogue, the next question would normally be 'How many children do you have?'. To avoid this question when the information was already provided, we have a pass-tag for omitting the next question. In this case, the pass-tag is added to the question with identifier *do_you_have_children* if the answer contains a numeral, as shown in example (2):

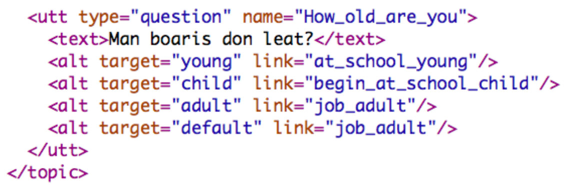
(2) MAP (&dia-pass) TARGET QDL
(0 (do_you_have_children) LINK *1 Num) ;

Let us consider a couple of examples showing how the dialogue may be branched to different questions and topics. In Figure 7 there are different follow-up questions for the answer to "What kind of car do you have?" If the car brand is in the lexicon, the system picks up the car type and uses it in a variable in the next question, e.g. "Is Ford a good car?", and if it is not in the lexicon (it can e.g. be a spelling error or a joke from the user), the next question will be "Is it a car?". There is also an alternative link for a negative answer ("Do you want to buy my car?"), and the default leads to a comment, which closes this topic.



```
<utt type="question" name="What_kind_of_car">
  <text>Makkár biila dus lea?</text>
  <alt target="target" variable="CAR" link="Is_it_a_good_car"/>
  <alt target="unknown" variable="CAR" link="Is_it_a_car"/>
  <alt target="neg" link="Do_you_want_to_buy_my_car"/>
  <alt target="default" link="car_closing"/>
</utt>

<utt type="question" name="Is_it_a_good_car">
  <text>Lea go CAR buorre biila?</text>
  <element id="CAR">
  </element>
```

Figure 7: Alternative links due to the answer of 'What kind of car do you have?'.



```
  <utt type="question" name="How_old_are_you">
    <text>Man boaris don leat?</text>
    <alt target="young" link="at_school_young"/>
    <alt target="child" link="begin_at_school_child"/>
    <alt target="adult" link="job_adult"/>
    <alt target="default" link="job_adult"/>
  </utt>
</topic>
```

Figure 8: Alternative branches due to the age of the user. The question is 'How old are you?'.

Whenever a topic is closed, the dialogue proceeds to the next topic. For example, an answer from the user about her age will induce a tag which is used for navigating to different branches of the dialogue based on the age of the user, as in Figure 8. The tag for age is assigned with a regular expression inside a CG rule, as in the examples (3), (4) and (5):

(3) MAP (&dia-adult) TARGET Num
(*-1 QDL LINK 0 (How_old_are_you))
(0 ("([2-9][0-9])"r)) ;

(4) MAP (&dia-young) TARGET Num
(*-1 QDL LINK 0 (How_old_are_you))
(0 ("([1][0-9])"r)) ;

(5) MAP (&dia-child) TARGET Num
(*-1 QDL LINK 0 (How_old_are_you))
(0 ("([0-9])"r)) ;

Users in the age-group below 20 proceed to a topic about going to school, while the older users are asked about their work. The question contains a default link as well, since some users have fun telling they are 1000 years old.

3.2 General rules

Most of the cg3-rules are general rules that apply to all question-answer pairs. Consider example (7), which generalizes over the question marker set in (6):

(6) LIST TARGETQUESTION-ACC = ("mii" Acc)
("gii" Acc)("galle" Acc) ("gallis" Acc) ;

(7) MAP (&dia-target) TARGET NP-HEAD + Acc IF
(*-1 QDL BARRIER S-BOUNDARY LINK *-1
TARGETQUESTION-ACC LINK NOT 0 Num)
(NEGATE *1 (N Acc) BARRIER VERB OR
CC)(NOT 0 NOTHING) ;

This is a general target rule for questions, which requires an answer in the accusative. S-BOUNDARY is a set of words and tokens which marks the end of the (sub)sentence. NOTHING is a set of indefinite pronouns like "nothing" and "nobody". There are similar rules for other cases.

There are also general rules for tags marking whether the answer is interpreted as affirmative or negative, as in Ex. (8):

(8) MAP (&dia-pos) TARGET QDL IF
(*-1 Qst OR go)(NOT *1 Neg);
MAP (&dia-neg) TARGET QDL IF
(*1 Neg BARRIER S-BOUNDARY);

In Sámi a yes-no question is indicated by a question particle "go", which can be a separate word or cliticized to the word to the left, which then gets a *Qst* tag in the analysis.

3.3 Storing information

It is useful to store some information about the user during the dialogue, such as name and age of the user. This information may be used in questions later, and give an impression of familiarity. These are implemented using special tags, such as in the examples (9) and (10):

(9) If the name is not in the lexicon:
MAP (&dia-target) TARGET QMRK IF
(*-1 QDL BARRIER (&dia-target) LINK 0
(What_is_your_name)) ;

(10) The name is in the lexicon:
MAP (&dia-target) TARGET Prop IF (*-1 QDL
BARRIER (&dia-target) LINK 0
(What_is_your_name)) ;

The set QMRK contains the question mark, and is given if the name is not in the lexicon, which is quite common with names. Both rules have &dia-target as barrier so it will hit only the first name, if there are many. There are similar rules and tags for information concerning place names, car brands and so on, and the information is used by the system in variables in tailored questions or utterances.

4 Tutorial feedback

The system gives tutorial feedback about grammar errors both in *Vasta* and *Sahka*. The feedback is generated from the grammar error tags, which are assigned during the disambiguation analysis. It should be noted that the system uses the grammatical analyser on the fly, exploiting full lexicons. This allows the user's answer to contain any Sámi word, also words that are not restricted to the pedagogical lexicon.

4.1 Grammar errors

```
"<Guđe>"
    "guhte" Pron Interr Sg Gen &grm-missing-Ill
"<latnjii>"
    "latnja" N Sg Ill
"<moai>"
    "mun" Pron Pers Du1 Nom
"<bidje>"
    "bidjat" V TV Ind Prs Du1
"<mu>"
    "mun" Pron Pers Sg1 Gen
"<TV>"
    "TV" N ACR Sg Acc
"<^sahka>"
    "^sahka" QDL where_place_TV
"<Moai>"
    "mun" Pron Pers Du1 Nom
"<bidje>"
    "bidjat" V TV Ind Prs Du1
"<TV>"
    "TV" N ACR Sg Gen
"<gievkkanis>"
    "gievkkan" N Sg Loc
"<.>"
    "." CLB
```

Figure 9: A grammar error tag is assigned.

In the question in Figure 9, the systems asks "In which room should we place the TV?" The user answers ”Moai bidje TV gievkkanis” (’We should place the TV in the kitchen’), with locative “gievkkanis” rather than the correct illative “gievkkanii”. The CG parser disambiguates the input, and the sentence matches the structural description of the general CG rule in example (11):

(11)
```
MAP (&grm-missing-Ill) TARGET (”guhte”) IF
(1 (N Ill) LINK *1 QDL LINK NOT *1 Ill OR
ADV-ILL OR Neg BARRIER S-BOUNDARY) ;
```

The rule adds a grammar-error-tag *&grm-missing-Ill* to the sentence analysis triggered by the interrogative pronoun followed by a noun in illative. This combination requires an illative form in the answer, when there is no illative form nor adverb with illative interpretation nor negation verb in the answer. The *Sahka*-engine generates a tutorial message based on the error-tag, given in example (12):

(12)
```
<message id=”grm-missing-Ill”>The
answer should contain an illative. </message>
```

One of the pedagogical goals behind the programs is that the user should practice inflecting the finite verb correctly. A central requirement is thus that the user answers with full sentences containing a finite verb. To encourage the user to practice also difficult verbs, she has to use the same verb as in the question. The CG rule in example (14) controls the choice of verb for the answer, and it uses a regular expression-based tag (a so-called *sticky tag*). The verb is identified via a regular expression $.*$ (cf. (13)), and the rule in (14) is triggered if it does not find the same verb lemma in both the question and the answer.

(13)
```
LIST VERBLEMMA = (”.*”r) ;
```

(14)
```
MAP (&sem-answer-with-same-verb) TARGET
FINVERB (NOT 0 Neg OR AUX-SET) (0
$$VERBLEMMA LINK *-1 QDL BARRIER
S-BOUNDARY OR FINVERB LINK NOT 0
EXEPTION-QUESTIONS LINK *-1 FINVERB
-1 BOS LINK NOT 1 $$VERBLEMMA)) ;
```

BOS is the left border of the sentence. Proverbs get a special treatment, and a question containing a pro-verb will accept any verb in the answer. There are also exceptional rules for some auxiliary verbs and for some questions, like for the question ”What is your name?”, which will more naturally be answered without a verb.

In *Vasta* the pronouns are not allowed to be interpreted inclusively (*we / you*, not *we / we*), but in *Sahka* they follow the logic of the scenario. This is the main reason for why *Sahka* has a slightly different rule set compared to *Vasta*. To indicate the type of the program in the morphological analysis, the delimiter between question and answer in *Sahka* is ”^sahka” instead of the delimiter tag ”^qst” used in *Vasta*.

Some of the questions in the *Sahka* dialogues are made for special grammatical training such as adjectival comparison. These questions populate a whole section of rules in the CG file. The rules add specific feedback to the potential errors.

The user will get only one feedback at a time, so the error tags are ordered partly as natural progress for error correction, and partly according to the likeliness of the error. First of all, the user will get feedback about spelling errors. If there is no agreement between subject and verb, then she will get feedback on the verb form, and not on the pronoun, given the assumption that the error is in the verb form rather than in the pronoun.

Grammar errors we have rules for, include

- verbs: finite, infinite, negative form, correct person/tense according to the question
- case of argument based upon the interrogative
- case of argument based upon valence
- locative vs. illative based upon movement
- subject/verbal agreement
- agreement inside NP
- numeral expressions: case and number
- PP: case of noun, pp based upon the interrogative
- time expressions
- some special adverbs
- particles according to word order
- comparison of adjectives

4.2 Misspellings

The user’s misspellings form the largest distinct problem for the functionality of the game. If the spelling error gives rise to a non-existing word form, then the message to the user is ”The word form is not in our lexicon, can it be a spelling error?”, which often is not of enough help to the user. A human reader would be able to read the answer in a robust way, and detect what the user intended to write. Simulating this ability is not an easy task.

Running the feedback through an ordinary speller engine is not a good solution, since the speller will come up with a large number of suggestions, without being able to choose between them. A possible solution would be to run a morphological analysis on the speller suggestions, and let a CG component pick the most likely candidates. The problem is that the current North Sámi speller (`http://divvun.no`) is made for native speakers and corrects mainly typing errors. In *Vasta* and *Sahka*, we would need a correction mechanism for errors due to wrong choice of affixes.

As a partial solution, we have added rules to the morphophonological rule file for typical spelling errors in e.g. place names. This enables the system to give a specific feedback in case of typical misspellings of place names. If the place name still is not recognized by the analyser, the feedback in the dialogue is ”I haven't heard about X. Is it a place?”, and the navigation proceeds to the next question.

The misspelling can also give rise to another word form of the same lemma. For such cases we have made rules based on the sentential context. The challenge is to give a feedback according to what the user thinks she has written, because she is probably not aware of the unintended word form. E.g. if the consonant gradation is incorrect in an attempted singular locative, the word form will be a nominative with possessive suffix Sg3. The learner will probably not know the possessive suffixes yet, so referring to it would not be useful. Instead, she gets the feedback: ”Do you mean locative? Remember consonant gradation.”

A more difficult problem emerges when the spelling error gives rise to an unintended lemma. Then the challenge is again to give feedback according to what the user thinks she has written. The feedback has to be tailored to what we know about the user's interlingua – and we have made some rules for sets of typical unintended lemmas. Some of them are systematic, such as the Sg2 of a verb incorrectly used after the negative verb, will result in a ConNeg form of a derived verb.

4.3 Metacomments

The *Sahka* program is intended to mimic a natural dialogue. But there are some restrictions in the possible input from the user; the system has to be able to analyse the input, and the answers should be pedagogically meaningful for the user. To remind the user of that, the system sometimes give metacomments to the user, like the following:

- ”Answering *I-don't-know* is too simple. Try again.”
- ”Your answer must always contain a finite verb.”
- ”You must use one of the words in the wordlist in the left margin.”
- ”You have not used the correct adjective. Try again.”

5 Evaluation

The evaluation of *Sahka* and *Vasta* was done when the programs had been available on internet for three months. The user's input and the feedback from the system were logged for the last two weeks of the period. The log shows that *Sahka* has six times as many queries as *Vasta*, so users clearly prefer the former one.

The system gave 156 tutorial feedbacks for the two programs during the two weeks. Breaking down the precision numbers on type of feedback, we got the picture shown in Table 1. Of 27 erroneous judgements, 16 were due to technical malfunction, 9 to wrong syntactical and 2 to wrong lexical analysis.

Rule type	**corr.**	**wrong**	**corr. %**
wrong tense	7	0	100,0
wr. V after neg	3	0	100,0
no infinite V	1	0	100,0
orth. error	44	2	95,7
wr. case V-arg	26	4	86,7
no finite verb	19	4	82,6
wr. S-V agreem.	17	8	68,0
wrong V choice	7	4	63,6
wrong word	4	4	50,0
wr. case after Num	1	1	50,0

Table 1: Feedback precision for different rule types.

As shown in Table 1 not all of the rule types mentioned in 4.1 have been in use during this period. These rule types have not been used:

- agreement inside NP (except for numeral expressions)
- nominal case inside PP
- time expressions
- word order errors for particles

The reason is probably that the users do not write more complex language than they have too. E.g. they don't answer with a complex NP if they can answer with just a pronoun or a noun, they don't write a time-expression with PP if the can answer with an adverb instead, and they don't use optional particles if they are unsure of where to put them. The price we pay for the free input strategy is that the users are not forced to exercise more complex language.

Table 2 on page 9 shows different kinds of error types the system has identified in the user's sentences, these we call *positives*. If it really is an error, then we call it it a *true positive*, if not, then it is a *false positive*. A sentence not flagged as an error by the system is counted as a *negative*, and we distinguish between *true negatives* (correct answers) and *false negatives* (erroneous answers which the system did not detect).

We measured *precision* (correctly identified errors/all diagnosed errors), *recall* (correctly identified errors/all errors), and *accuracy* (correct judgements/cases). For the error types we target, precision = 0.85, recall = 0.93, and accuracy = 0.89 (N=277). Better recall than precision indicates that very few errors slip through, at the price of erroneously identifying some correct forms as errors. The system is thus a bit too critical towards the students: It almost never lets through a (targeted) mistake. In this pedagogical setting, a goal for future work is improving precision (avoiding erroneous error flagging), perhaps even with the risk of a lower recall.

6 Future perspectives

We have started the work with improving the system. Among our future plans are:

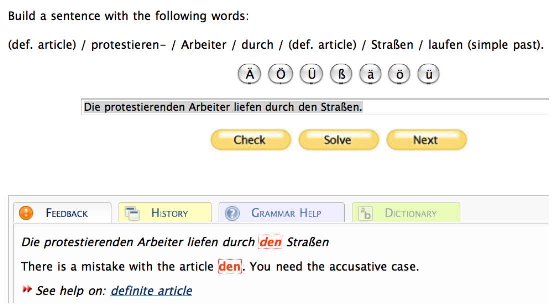
Implementing a speller. Because the misspellings are the biggest problem for the users, we will implement a speller. We will give relevant suggestions to the user by analysing the list of suggestions according to the context with CG, and also implement weighted lexical transducers, see (Linden and Pirinen, 2009). For the weighting we will use the pedagogical lexicon and the North Sámi corpus as a training corpus.

Implementing a topic option in *Vasta*. Today *Vasta* generates questions randomly within each level based on grammar difficulty. The log shows that this program is not as popular as the *Sahka*. We are planning to make it more interesting for the users by restricting the semantic sets for the variables in the question templates according to topics, and give the user's a topic option as well.

Sentence building from a fixed set of lemmas. We are also considering forcing the user to construct more complex phrases and also use more particles, by deciding what lemmas the user should use, as a supplement to the other programs. Available on internet is *e-tutor* – a program for teaching German to foreigners (Heift, 2001), at `http://e-tutor.org/`. e-tutor gives very good feedback to student's errors, but the possible input is restricted to a set of lemmas by means of which she has to construct a sentence. In this way the user is forced to write more complex phrases. Figure 10 shows an example from the program.

Conduct studies on Oahpa in actual use. Investigating how Oahpa works in actual use in the classroom will be important in the work with improving the system.

Porting the programs to more Sámi languages. For Lule Sámi a morphological analyser is available, and we have started making a CG disambiaguator. For South Sámi a morphological analyser will be finished in 2010.



Build a sentence with the following words:

(def. article) / protestieren– / Arbeiter / durch / (def. article) / Straßen / laufen (simple past).

Die protestierenden Arbeiter liefen durch den Straßen.

Die protestierenden Arbeiter liefen durch den Straßen

There is a mistake with the article den. You need the accusative case.

See help on: definite article

Figure 10: An alternative to free input is e-tutor.

7 Conclusion

The paper has shown how we use vislcg3 for pedagogical dialogue systems for North Sámi. Vislcg3 is used in many ways: By relaxing the analysis of the input string, we are able to find errors made by the user, and assign feedback tags to the analysis. Secondly, by analysing the semantics of the user's input, and assigning semantic tags to the input, we are able to navigate through the dialogue according to user feedback. And finally, we can assign

Error type	true pos.	false pos.	true neg.	false neg.	precision	recall	accuracy	F-ms.
Gramm. error	641	234	769	7	0,73	0,99	0,85	0,84
Semant. error	805	69	764	12	0,92	0,99	0,95	0,95
Orthogr. error	875	0	776	0	1	1	1	1
Other error	695	180	751	25	0,79	0,97	0,88	0,87
	3016	483	3060	44	0,86	0,98	0,92	0,92

Table 2: Precision, recall and accuracy for different error types.

tag to information in the user's input and use it in the program's questions or utterances.

The CG formalism has a great potential for use in pedagogical settings. It is robust enough to handle erroneous data, and at the same time flexible enough to give both general corrections, and corrections targeted at specific words in specific settings.

We have seen that a major problem is spelling errors. Whether CG is able to offer a solution for this problem as well, remains a topic for future research.

Acknowledgments

Thanks to the faculty of Humanities at the University of Tromsø, and the Sámi Parliament in Norway, for funding the project.

References

Lene Antonsen, Saara Huhumarniemi and Trond Trosterud. 2009. Interactive pedagogical programs based on constraint grammar. *Proceedings of the 17th Nordic Conference of Computational Linguistics*. Nealt Proceedings Series 4.

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI publications in Computational Linguistics. USA.

Eckhard Bick. 2005. Live use of Corpus data and Corpus annotation tools in CALL: Some new developments in VISL. Holmboe, Henrik (ed.): *Nordic Language Technology, Årbog for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*, 171–185. København: Museum Tusculanums Forlag.

Krister Lindén and Tommi Pirinen. 2009. Weighted Finite-State Morphological Analysis of Finnish Compounding with HFST-LEXC. *Proceedings of the 17th Nordic Conference of Computational Linguistics.*. Nealt Proceedings Series 4.

Trude Heift. 2001. Intelligent Language Tutoring Systems for Grammar Practice. *Zeitschrift fur Interkulturellen Fremdsprachenunterricht [Online]* 6(2).

Fred Karlsson, Atro Voutilainen, Juha Heikkilä and Arto Anttila. 1995. *Constraint grammar: a language-independent system for parsing unrestricted text.* Mouton de Gruyter.

VISL-group. 2008. *Constraint Grammar*. http://beta.visl.sdu.dk/constraint_grammar.html University of Southern Denmark.

Parsing Corpus of Estonian Dialects

Liina Lindström
University of Tartu
Estonia
liina.lindstrom@ut.ee

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

Abstract

This paper introduces our work for adapting a rule based parser of spoken Estonian to the morphologically unambiguous part of the corpus of dialects. A Constraint Grammar based parser was used for shallow syntactic analysis of Estonian dialects. The recall of the grammar was 96-97% and the precision 87-89%.

1 Introduction

The goal of this research was to find a method for automatic syntactic annotation of the Corpus of Estonian Dialects (CED)[1].

The dialect corpus was compiled by two institutions – the University of Tartu and the Institute of the Estonian Language. The Corpus of Estonian Dialects consists of:

1) dialect recordings;
2) phonetically transcribed dialect texts;
3) dialect texts in simplified transcription;
4) morphologically tagged texts;
5) a database containing information about informants and recordings.

The texts in the corpus are spoken dialect interviews.

By the end of 2008, the corpus contained about 1,000,000 transcribed text words and 500,000 morphologically tagged text words.

We have used morphologically tagged texts as input for the syntactic parser.

The texts of the dialect corpus represent spoken language and have been transcribed using quite similar principles as used for the Corpus of Spoken Estonian (Hennoste et al., 2000). For this reason, we decided to test the parser of spoken language (Müürisep and Nigol, 2007, also Müürisep and Nigol, 2008) on the texts of dialects. It should be noted that the parser of spoken language is an adaption of parser for written language (Müürisep et al., 2003).

The parser for written Estonian is based on Constraint Grammar framework (Karlsson et al., 1995). The CG parser consists of two modules: morphological disambiguator and syntactic parser. In this paper, we presume that the input (transcribed speech) is already morphologically unambiguous and the word forms have been normalized according to their orthographic forms.

The parser gives a shallow surface oriented description to the sentence where every word is annotated with the tag corresponding to its syntactic function (in addition to morphological description). The head and modifiers are not linked directly, only the tag of modifiers indicates the direction where the head may be found.

```
aga                                          ;; but
   aga+0 //_J_ coord  //    **CLB @J
timä                                         ;; he
   tema+0 //_P_ pers ps3 sg nom //   @SUBJ
!!!=
ol'l'                                        ;; was
   ole+0 //_V_ main ps indic impf sg ps3 // @+FMV
latsõst                                      ;; childhood
   laps+0 //_S_ com sg el //   @P>
saan'iq                                      ;; since
   saadik+0 //_K_ post #el //   @ADVL
!!!=
tark                                         ;; clever
   tark+0 //_A_ pos sg nom //   @AN>
poiss                                        ;; boy
   poiss+0 //_S_ com sg nom //   @PRD
```

Fig. 1: An extract from syntactically annotated corpus of dialect Võru**:** *aga timä oll latsõst saaniq tark poiss* 'but he was a clever boy already since childhood'. @J - conjuction, @SUBJ - subject, @+FMV - finite main verb, predicate, @P> - complement of postposition, @ADVL - adverbial, @AN> - premodifying attribute, @PRD - predicative or complement of subject. Morphological tags are between "//"-characters.

[1] see *http://www.murre.ut.ee/korpus.html* (in Estonian)

Figure 1 depicts the format and tag set of syntactically annotated sentence. The parser of written text analyzes 88 - 90% of words unambiguously and its error rate is 2% (if the input is morphologically disambiguated and error-free). The error rate for the corpora of dialects is higher: 3-5%, but approximately 89-92% of words are assigned exactly one syntactic tag. The words which are hard to analyze remain with two or more tags.

As mentioned before, the parser is rule based. The grammar consists of 1200 handcrafted rules. The grammar rules implement a conservative parsing strategy - they rather leave the word form ambiguous than remove the correct tag.

The remainder of this paper is organized as follows. We will give an overview of the Corpus of Estonian Dialects in section 2. Section 3 describes the conversion of texts from XML format to the textual format (see Fig. 1 and 2) and section 4 deals with the modification of the grammar. We will give an overview of the parser evaluation process in section 5. In section 5, we also discuss the main shortcomings of the parser: the error types and ambiguity classes and compare the results of the parser with the results of the spoken language parser.

2 Overview of the Corpus

The Corpus of Estonian Dialects (CED) is an electronic data collection which includes authentic dialect texts from all Estonian dialects. In order to create a solid base for further research, the dialect data in CED were well-chosen and meticulously transcribed. There is roughly the same amount of material from every Estonian dialect in the corpus. The first part of CED was composed from the oldest available tape-recorded dialect texts and contains about 1 million text words.

The corpus is based on dialect recordings which have mainly been made in the 1960s and 1970s. However, the first recordings are much older – they date from 1938. The recordings are usually interviews conducted at the home of the dialect informant.

The dialect texts in Fenno-Ugric phonetic transcription constitute one of the main parts of the corpus. The aim has been to transcribe the texts as accurately as possible; the phenomena accompanying spontaneous speech (e.g. the discourse particles, corrections, repetitions, etc.)

```
<u who="KJ">
<mark><sne>no</sne><msn>no</msn><mrf
slk="Par"/> </mark>
<mark><sne>tsuvvaq</sne><msn>tsuug</
msn><tah>pastel</tah><mrf slk="S">pl
n</mrf></mark>
</u>
```

Fig. 2: Example of morphologically annotated utterance

have been added to the text which usually have not been considered important in dialect research.

All of the phonetically transcribed texts have been transformed in one-to-one fashion without information loss into the simplified transcription. In addition, the comments, the text of the informant(s) and the interviewer have been annotated. This annotation is preserved also in morphologically tagged texts.

Texts in the simplified transcription are morphologically tagged. The tagged texts are in XML format. Words have been divided into 26 word classes according to their morphological inflections, syntactic characteristics and semantics. This classification is based on the system of word classes presented in Estonian grammars (Erelt et al., 1995: 14–41); however, more subclasses can be distinguished (e.g. proadverbs, affixal adverbs; see Lindstrom et al., 2006). In addition, the annotation includes 2 numbers, 15 cases and possessive suffixes for nomens, and 25 features and endings for verbs. The XML annotation consists also of meta information (dialect, informant, transcriber, annotator etc.), remarks about background activities, and sometimes also the meaning of the word form.

Figure 2 demonstrates an extract from a short dialogue turn from CED where the informant (<u who="KJ">) says *no tsuvvaq*, *no* is a particle and *tsuvvaq* is a plural noun in nominative case meaning *pastel* 'soft leather shoe'.

According to the traditional approach (cf. Pajusalu, 2003), Estonian dialects are divided into three dialect groups. These dialect groups are further divided into different dialects, the dialects are divided into parish dialects (sub-dialects). The following dialect groups and dialects are represented in the dialect corpus:

1) North Estonian dialect group: Mid, Eastern, Western, Insular dialects;

2) South Estonian dialect group: Võru, Mulgi, Tartu, Seto dialects;

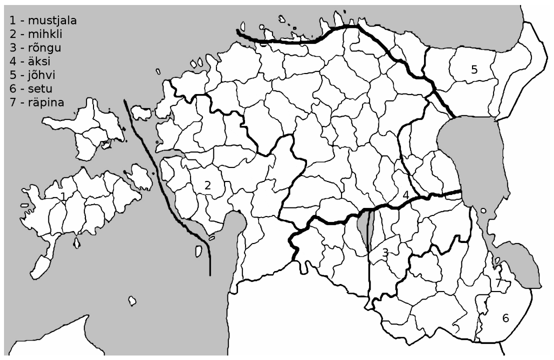


Fig. 3: The map of of the parish dialects used in the experiment

3) North-Eastern Coastal dialect group: North-Eastern (Alutaguse), Coastal dialects.

In our research for automatic syntactic annotation of dialects, we use subcorpus of 19,000 words from 7 different parish dialects (see Fig. 3).

The Äksi parish dialect (4 in the map) represents the central Mid dialect which is also the basis for standard Estonian. Mustjala (1) represents the Insular dialect and Mihkli (2) represents the Western dialect, both belonging to the North-Estonian dialect group. Jõhvi (5) belongs to North-Eastern Coastal dialect group which is rather different from the North Estonian dialect group; also, it has many similarities to Finnish dialects.

Three parish dialects – Rõngu (3), Räpina (7) and Seto (6) – represent the South-Estonian dialect group which is even more different from North Estonian (and standard Estonian) than North-Eastern Coastal dialect. Rõngu belongs to Tartu dialect which has historically had more connections to North Estonian than Räpina and Seto.

Parish dialect	**Word count**
Äksi	3569
Mustjala	1013
Rõngu	1457
Jõhvi	2975
Seto	3122
Räpina	2559
Mihkli	4303
Total	18998

Table 1: The list of used subdialects and their size

Table 1 presents word counts for these corpora.

3 Conversion of the Corpus

In order to apply constraint grammar parser to the corpus of dialects, we had to convert it to the appropriate format (see Fig. 1). As the original format of the corpus was well documented and automatically generated, the transformation process was fairly smooth. The hardest task was the mapping of differencies in word class tagging.

The original annotation did not distinguish modal verbs from main verbs but this information is crucial for syntactic rules. For this reason, every potential modal verb (4 verbs) got an additional morphological reading.

Also, the original mark-up lacks the detailed classification of pronouns. This was added using a special database. Since the dialects may have different pronouns (for example *sjoo* means 'this' in Seto subdialect) there might be a need to update the database before analysing new dialect.

Grammar rules use the valency database of adpositions. Dialect specific adpositions should be added to this before automatic transformation. Before applying the conversion program to a new dialect one should check the list of adpositions.

The tags which exist in the dialect corpus but do not exist in the corpus of spoken language remain in the annotation in the same form (for example, the case of instructive).

All words without morphological annotation, irrelevant transcription tags, records of meanings and remarks are commented out with a special tag *!!!*, so they do not influence the work of the parser (see Fig. 1).

The most substantial difference in the annotation of dialects and spoken language is in the mark up of participles. Namely, the participles which act similarly to adjectives (attributes and predicatives) are annotated as adjectives with extra tag *partic* in the corpora of spoken and written language. The mark up of dialect corpus does not distinguish different types of participles, all participles carry the POS tag of verb. As the participles act in dialects mainly as parts of verb chain (they form perfect and past perfect tense) and quite seldom as attribute or predicative, the introduction of a new morphological ambiguity was not reasonable.

```
aga                                   ;; but
   aga+0 //_J_ coord //   @J
siss                                  ;; then
   siis+0 //_D_ //   @ADVL
!!!=
e                                     ;; ee
   e+0 //_B_ //   @B
!!!$.
!!!   $. //_Z_ Fst //
*pulmad                               ;; weddings
   pulm+0 //_S_ com pl nom // @REP
*pulmad                               ;; weddings
   pulm+0 //_S_ com pl nom //   @SUBJ
õlid                                  ;;were
   ole+0 //_V_ main ps indic impf pl ps3 //   @
+FMV
*ikke                                 ;; still
   ikka+0 //_D_ //   @ADVL
*suure+perälised                      ;; marvellous
   suure+pärane+0 //_A_ pos pl nom //   @PRD
minul                                 ;; I
   mina+0 //_P_ pers ps1 sg ad //   @ADVL
küll                                  ;; indeed
   küll+0 //_B_ //   @B
```

Fig. 4: An extract from syntactically annotated corpus of dialect Võru. 'I had indeed marvellous weddings'

4 Conversion of the Grammar

Comparison of dialect texts with texts of spoken language revealed that the largest modifications in grammar should be related to a) inner clause boundary detection rules due to lack of intonation mark up; b) differences in annotation scheme; c) differences in vocabulary.

We inspected all rules for clause boundary detection thoroughly. In addition to the fact that dialect corpus lacks the intonation mark up, we had to consider that dialect texts resemble monologues, the utterances are longer than in everyday conversations or information dialogues.

Two types of pauses were transcribed in the dialect corpus, the shorter and the longer. The experiments showed that the use of shorter pauses as delimiters is dangerous since they occur quite often inside a phrase when a speaker is looking for an appropriate word, and their use was rather an obstacle during parsing.

In most cases the morphological description contains the normalized form of the stem which was mostly the same as in written language. There were some exceptions: we had to amend negational words (*ei* 'not', new words *ep*, *es*), add *nakkama* to the set of *hakkama* 'begin, start', etc. Also, we had to add new items to the sets related to temporal adverbial with folk calendar days like *jüripäev* 'St. George's day', *jaanipäev* 'midsummer day', *mihklipäev* 'St. Michael's day'. Fortunately, these modifications of rules were marginal.

We did not find a good solution for the analysis of participles which have different annotation scheme than used in other text corpora. It turned out that the ratio of precision and recall was best if we left the grammar willingly erroneous since the participles act seldom as attributes or predicatives in dialects.

We had to remove some seemingly correct rules from the grammar since they caused many errors due to erroneous clause boundary detection. First of all this holds for the principle of uniqueness: every main verb may have one uncoordinated subject. The same principle is also valid for objects and predicatives. These rules generate a lot of errors during the analysis of utterances with disfluencies or ellipses (see example (1)).

(1)	ja	ilus	**ein**	onn	väga	ilus
	and	beautiful	hay	is	very	beautiful
	ein	sin	all	...		
	hay	here	below	...		

'and it is a very beautiful hay here below'

We use the same method for the detection of simpler disfluencies as used for contemporary spoken language: an application of external script which removes repeats and simpler self-repairs before the parsing process and restores them in the output with a special tag after the analysis.

Modification and addition of rules took place with the help of a training corpus of 5700 words which was manually syntactically annotated. The training corpus allowed to research how the rules function and interact on dialect texts, which rules should be modified, which ones should be removed and which ones to be added. The texts of the training corpus were basically from Central, Western and Insular parishes.

During the rule design process, we attempted to minimize their error rate. If the reasonable error rate for written language is below 2% then error rate for dialects turned into 3-3.5%. The further debugging of rules gave only small effect since most of remaining errors had been caused by the phenomena specific to spoken language: disfluencies, elliptical utterances, unfinished utterances, agreement conflicts etc.

5 Evaluation

Table 2 demonstrates the gained results for different corpora. The test corpora have not been used during the process of grammar development. The results have been calculated on the automatic comparison of manually annotated corpora with automatically parsed corpora. Corpora have been annotated mainly by one human expert but the complicated utterances have been discussed by several researchers.

Dialect and type	**Word count**	**Recall**	**Precision**
Mustjala (training)	1013	97.14	86.54
Mihkli (training)	2140	96.87	90.01
Mihkli (test)	2163	96.44	85.88
Rõngu (training)	1457	96.98	89.96
Äksi (training)	977	96.52	88.56
Äksi (test)	2592	96.45	87.81
Jõhvi (test)	2975	96.12	87.35
Seto (test)	3122	95.26	88.59
Räpina (test)	2559	95.82	86.49
Training total	5587	96.89	89.09
Test total	13441	95.93	87.24

Table 2: The precision and the recall of the parser.

The table illustrates that the correctness in test corpora is almost 1% lower than in training corpora, and the precision is lower by 2%. The results are significantly worse on the corpora of Southern Estonian dialects. This may have two reasons: first, Southern Estonian texts were not used during the training and development process of the grammar. On the other hand, the Souther Estonian dialects differ significantly from standard Estonian which is based on North-Estonian central dialect. Also, one should take into account that every dialect text in this experiment represents only one speaker and the results of the dialect parsing depend on the fluency of speech of this speaker. For example, the informant for Jõhvi dialect was an elderly woman who had difficulties with speaking fluently.

The comparison of results of parsing dialects and spoken language indicates that the parser performs 1-2% worse on dialects (see Table 3). But also, we have to consider the influence of the genre to the outcome. For example, everyday conversations are easier to parse than information dialogues (this means that the precision and recall are higher). For this reason, we included a short radio interview to the comparison corpora which has a genre most similar to dialect corpora. The results of parsing this corpus are comparable to the results of parsing dialect corpora.

Corpus	**Type**	**Recall**	**Precision**
Everyday conversation	training	97.46	89.66
	test	97.58	91.84
Information dialogues	training	97.06	87.63
	test	96.77	87.42
Radio interview	test	96.80	88.47
Dialects	training	96.89	89.09
	test	95.93	87.24

Table 3: Comparison of parsing results for spoken language and dialects

5.1 Error types

The analysis of error types has been generated on the basis of subcorpus of Mihkli parish dialect of 2500 words.

We tried to group the errors in a generic fashion, individual cases which were hard to generalize have been categorized as Other. Table 4 gives overview of error types and their occurrence in the subcorpus.

In some cases it is very difficult to detect the clause boundary (see example (2)) and these errors are hard to avoid.

(2)	rukis	andis	ikka	väiksema	saagi	ia	ei	olnud
	rye	gave	still	smaller	harvest	good	not	was

'Rye gave a smaller harvest. It wasn't good.'

The errors of syntactic rules may occur also during the analysis of other types of corpora, they may be caused by unusual word order, small unfixed error in context conditions of a rule or some other shortcomings of rules.

Error	Count
clause boundary detection	12
syntactic rules	11
a np-phrase before or after a clause	11
ellipse	9
mapping rules	6
kõik/all	6
predicative	4
disfluency detector	2
unknown syntactic error	2
dialect specific	3
other	11
Total	77

Table 4: Count of different error types

An solitary noun phrase causes always confusion since the clause boundary detection rules could not find the border between the phrase and a new clause. Mostly the problematic noun phrases locate before the clause as in example (3).

(3)

üks	sort	need	on	väga	kibedad
one	sort	these	are	very	bitter

'One sort. These are very bitter.'

But they can also be found after the clause as in example (4).

(4)

kui	aeg	seokke	oli	seemne
when	time	such	was	seed
tegemise	aeg			
making	time			

'When time was such. It was time for sowing seeds.'

Ellipse is also a frequent phenomenon in spoken language. Often the missing element is *be*-verb as in example (5).

(5)

üks	ees	teene	taga
one	before	other	behind

'One is before, the other is behind"

In some cases, the correct syntactic tag is never added to the word form. Typically this is a case where adjective acts as a noun but in dialect texts, there are also cases where pronouns were used as discourse particles or as a part of exclamation (*oh sa taevas* 'oh you heaven').

Unexpectedly, the word *kõik* 'all' caused a number of errors which are all hard to avoid. *kõik* 'all' can act as a normal pronoun but quite often it is premodifying or postmodifying attribute locating outside the phrase (see example (6)).

(6)

pääbad	oli	jaettud	kõik
days	were	divided	all

'All days were divided'

kõik 'all' may also be found as a discourse marker as in example (7).

(7)

pangad	olid	raha	täis	ja	kõik	jahh
banks	were	money	full	and	all	yes

'The banks were full of money and ...'

There was a regular pattern of incorrect analysis of predicatives in the test corpus as in example (8).

(8)

Põllud	ond	neokst	kitsad
Fields	were	such	narrow

'Fields were such narrow.'

One could consider this as a shortcoming of syntactic rules.

There were only 3 errors which may be classified as dialect specific, 2 of them occur with indefinite pronoun *keegi* 'nobody' which was used instead of *miski* 'nothing'.

Disfluency detector made 2 errors, and 2 errors were related with words which syntactic functions were not possible to decide.

5.2 Ambiguities

As the error rate of the grammar was 3-4% then the second important indicator of parsing efficiency was ambiguity rate. The percentage of remaining syntactic readings is given in Table 5 (on the basis of test corpus of 13,411 words).

92% of words become unambiguous, 5.8% of words have two syntactic tags, and 1.9% of words have 3-5 syntactic tags.

The ambiguity class of subject and object dominates among ambiguity classes (see Table

6), followed by the ambiguity of subject and predicative, adverbial and subject, and finally followed by the ambiguity classes containing attributes.

Count of syntactic tags	Percentage
1	92.36
2	5.80
3	1.56
4	0.23
5	0.05

Table 5: The percentage of the count of syntactic tags in the test corpus

The domination of the ambiguity class of object and subject may be explained by the inexact clause boundary detection - it is not clear which word belongs to which verb and the decisions are made rather by the form of the noun.

Ambiguity class	Count
@OBJ @SUBJ	212
@PRD @SUBJ	134
@ADVL @SUBJ	68
@ADVL @NN>	64
@NN> @OBJ	60
@NN> @SUBJ	57
@ADVL @OBJ	56
@-FMV @ADVL	55
@ADVL @OBJ @SUBJ	53
@OBJ @PRD @SUBJ	36
@ADVL @PRD @SUBJ	35
@<NN @ADVL	30

Table 6: The main ambiguity classes

6 Conclusions

Our experiment of using a parser of spoken language for syntactic analysis of the corpus of dialects can be regarded fairly successful. Although the error rate of the analysis is 1-2% higher than for the spoken language parser, most of the errors are hard to avoid. The parser and its grammar that are based on Constraint Grammar framework are robust enough to deal with non-fluent speech and syntactic constructions specific to dialects. Approximately 10% of words remain ambiguous in the output of the parser but fortunately these ambiguities will not obstruct linguistic research.

We plan to analyze the whole corpus in an automated fashion and make it available on the web. Also, we are planning to create a publicly available search engine for the corpus, in order to facilitate further studies of Estonian syntax and dialects.

References

Erelt, Mati, Reet Kasik, Helle Metslang, Henno Rajandi, Kristiina Ross, Henn Saari, Kaja Tael, Silvi Vare. 1995. *Eesti keele grammatika*, vol. 1. Eesti TA Keele ja Kirjanduse Instituut, Tallinn.

Hennoste, T., Lindström, L., Rääbis, A., Toomet, P., Vellerind, R. 2000. Tartu University Corpus of Spoken Estonian. In Seilenthal, T., Nurk, A., Palo, T., eds.: *Congressus Nonus Internationalis Fenno-Ugristarum* 7.-13. 8. 2000. Pars iv. Dissertationes sectionum: Linguistica I, Tartu (2000) 345–351

Karlsson, F., Voutilainen, A., Heikkilä, J., Anttila, A. 1995. *Constraint Grammar: a Language Independent System for Parsing Unrestricted Tex*t. Mouton de Gruyter, Berlin and New York.

Lindström, Liina, Liisi Bakhoff, Mari-Liis Kalvik, Anneliis Klaus, Rutt Läänemets, Mari Mets, Ellen Niit, Karl Pajusalu, Pire Teras, Kristel Uiboaed, Ann Veismann, Eva Velsker. 2006. Sõnaliigituse küsimusi eesti murrete korpuse põhjal. – E. Niit (ed.) *Keele ehe*. Tartu Ülikooli eesti keele õppetooli toimetised 30, Tartu: 154-167

Müürisep, Kaili, Helen Nigol. 2007. Disfluency Detection and Parsing of Transcribed Speech of Estonian. *Proc. of Human Language Technologies as a Challenge for Computer Science and Linguistics*. 3rd Language & Technology Conference (ed. Zygmunt Vetulani). Oct 5-7, 2007, Poznan, Poland. Fundacja Uniwersitetu im. A. Mickiewicza. pp. 483-487.

Müürisep, Kaili, Helen Nigol. 2008. Where Do Parsing Errors Come From: The Case of Spoken Estonian. In Sojka, P.; Horak, A.; Kopecek, I.; Karel, P. (eds.). LNCS 5246. *Text, Speech and Dialogue*. Springer-Verlag. pp. 161 - 168.

Müürisep, Kaili, Tiina Puolakainen, Kadri Muischnek, Mare Koit, Tiit Roosmaa, Heli Uibo. 2003. A New Language for Constraint Grammar: Estonian. *International Conference Recent Advances in Natural Language Processing*. Proceedings. Borovets, Bulgaria, 10-12 September 2003, pp. 304-310.

Pajusalu, Karl. 2003. Estonian Dialects. – Mati Erelt (ed.) *Estonian Language*. Linguistica Uralica supplementary series, vol. 1. Estonian Academy Publishers, Tallinn: 231-272.