

**THE 11<sup>TH</sup> NORDIC CONFERENCE ON  
COMPUTATIONAL LINGUISTICS**

Copenhagen, 28-29 January 1998

**NODALIDA '98  
PROCEEDINGS**

Center for Sprogteknologi  
and  
Department of General and Applied Linguistics (IAAS)  
University of Copenhagen  
Njalsgade 80  
DK-2300 Copenhagen S, Denmark

ISBN 87-90708-01-6

©NODALIDA '98 and the authors  
c/o Center for Sprogteknologi, Njalsgade 80, DK-2300 Copenhagen S

Thanks go to all members of the programme committee for their unvaluable help reading the papers for this conference

Copenhagen, March 1998

Bente Maegaard.

# CONTENT

Welcome <i>John Kuhlmann Madsen</i> .....	1
Introduction <i>Bente Maegaard</i> .....	3
LMT at Tivoli Gardens <i>Arendse Bernth, Michael McCord</i> .....	4
Drug Terminology. A multilingual term database. The AVENTINUS project <i>Christian Sjøgreen</i> .....	13
AVENTINUS, GATE and Swedish Lingware <i>Dimitrios Kokkinakis</i> .....	22
Norwegian Computational Lexicon ( <i>NorKompLeks</i> ) <i>Torbjørn Nordgård</i> .....	34
Structural Lexical Heuristics in the Automatic Analysis of Portuguese <i>Eckhard Bick</i> .....	44
An HPSG Marking Analysis of Danish Determiners and Clausal Adverbials <i>Costanza Navarretta, Anne Neville</i> .....	57
A Chart-Based Framework for Grammar Checking. Initial Studies <i>Anna Sågvall Hein</i> .....	68
CP-UDOG: An Algorithm for the Disambiguation of Compound Participles in Danish <i>Jens Ahlmann Hansen, Poul Søren Kjærsgaard</i> .....	81
Evaluation of the Syntactic Parsing Performed by the ENGCG Parser <i>Klas Prytz</i> .....	87
CATCH: A Program for Developing World Wide Web CALL Material <i>Erik F. Tjong Kim Sang</i> .....	94
Assembling a Balanced Corpus from the Internet <i>Johan Dewe, Jussi Karlgren, Ivan Bretan</i> .....	100
Peeking Into the Danish Living Room. Internet access to a large speech corpus <i>Peter Juel Henriksen</i> .....	109

Extraction of Translation Equivalents from Parallel Corpora <i>Jörg Tiedemann</i> .....	120
The Construction of a Tagged Danish Corpus <i>Thomas Bilgram, Britt Keson</i> .....	129
Linguistics isn't always the answer: Word comparison in computational linguistics <i>Lars Borin</i> .....	140
Logic for Part-of-Speech Tagging and Shallow Parsing <i>Torbjörn Lager</i> .....	152
A statistical and structural approach to extracting collocations likely to be of relevance in relation to an LSP sub-domain text <i>Bjarne Blom</i> .....	160
A Reasonably Language Independent, Heuristic Algorithm for the Marking of Names in Running Texts <i>Benny Brodda</i> .....	168
Some Results Regarding Tree Homomorphic Feature Structure Grammar and the Empty String <i>Tore Burheim</i> .....	176
Teaching and learning computational linguistics in an international setting <i>Koenraad de Smedt</i> .....	186

# Welcome

**John Kuhlmann Madsen**

Dean of the Faculty of Humanities, University of Copenhagen

Ladies and gentlemen, dear colleagues,

I have been told that this is the second time you have honoured our Faculty with a Nodalida meeting. The first one took place in the now – computationally speaking, if you allow me – almost prehistoric year of 1979.

Permit me to take some minutes of your precious time to talk about language studies in rather general terms.

The seventies were the heyday of massive student interest in formal linguistics. In later years we discovered a rather disturbing movement away from the formal aspects of language study, more or less corresponding to the exodus from the maths and physics departments. A couple of years ago we made a small survey of the modern language departments, focusing on the themes of the minor theses, (which are compulsory for obtaining the MA degree). It was shocking for me to learn that in two consecutive academic years there was no minor thesis on theoretical aspects of grammar, phonology, or semantics, in English, French, German, or Spanish. The closest we got to linguistics in language studies were the essays on sociological aspects of some speech community in some distant regional areas, and the like. We can detect the same tendency when applicants make their first choice: first in line we have psychology, general literature, film and media studies, history of art, philosophy, theatre studies, and history. I am sure you get the overall picture: the main interest lies within hermeneutics and aesthetics. According to fellow deans from other arts and humanities faculties the results of our survey are not exceptional. We are facing a similar trend in the nineties. The way I see it, we are talking about a trend that must be halted; not to save our language and linguistics departments, but simply because the small Nordic countries cannot do without top-level language education. We like to think that we speak foreign languages better than the foreigners do. I must confess that I am not so sure of the correctness of this opinion. On the other hand, you need not be a formal linguist to speak a foreign language beautifully, but, if only a tiny minority feels attracted by the study of grammar, I fear teachers will soon experience some difficulty correcting their pupils.

I feel that in promoting linguistic language studies we are moving upstream. I am confident that it will be worth our while. However, we must be realistic in our efforts. Some years ago our Minister of Education tried to force students in specific directions. He succeeded only in causing severe damage to many subject areas, especially within the humanities and, ironically enough, in their language departments. We still suffer from his peculiar ideology. The way I see it, the study of language must regain its attraction on its own terms. The jobs are there, the teachers are – still – there, but how do we attract students other than those who have been *au pairs*? The usual trick is to sit back and leave it to the secondary school system. The interest must be created in secondary school, but why leave it all to others. Why do we not collaborate with the language

teachers and their principals to at least gain some understanding of the situation? For our part that is exactly what we are doing now. In mid February headmasters from the eastern part of the country will meet at our Faculty to start a discussion on how we can get to understand each other's needs. Hopefully, that meeting will lead to further meetings on specific subjects, including language and linguistics.

Last year we noticed an increased interest in linguistics among our applicants. Hoping for a turn of the tide I look forward to welcoming many future linguists – computational or not – at our Faculty. I am confident that your work – here and elsewhere – will be inspiring both for your fellow specialists and for the study of language in general.

It's my pleasure to bid you welcome to two days of mutual inspiration.

Thank you

# Introduction

**Bente Maegaard**

Programme chair

The Nordic conference of Computational Linguistics (Nordiske Datalingvistikdage, NODALIDA) is a biannual event, directed to the researchers, students and industrialists in the field of computational linguistics as well as to all interested parties. Most of the participants are from the Nordic countries.

Papers were invited from all areas of computational linguistics, including but not restricted to: morphological, syntactic, semantic and textual analysis and generation, speech processing, machine translation, computational lexicography and processing of monolingual or multilingual text corpora.

The programme featured many of these with a specific focus on corpus related work, i.e. the creation of corpora, and their use for different purposes. Research and development concerning computational lexica was discussed in a couple of papers. Another feature worth mentioning is that several papers discussed the use of the Internet in connection with computational linguistic applications: making corpora available on the Internet, making educational software packages available on the Internet etc.

I wish to thank warmly the programme committee who with their expertise and their readiness to provide feedback to the authors and to me very fast, made it possible to provide a very interesting programme. The programme committee consisted of Kimmo Koskenniemi, Helsinki, Peter Molbæk Hansen, Copenhagen, Torbjørn Nordgård, Trondheim, and Anna Sågvall Hein, Uppsala.

A panel discussion on *The Nordic languages in the Information Society - a responsibility for computational linguistics and computational linguists?* showed that the interest from public funding agencies in supporting computational linguistics has been and is different in the various Nordic countries. The Nordic Council discussed the language issue last May, and it was suggested that a coordinated approach be made to the Nordic Council concerning the protection and reinforcement of the Nordic languages which are 'less used' on a world basis.

# LMT at Tivoli Gardens

**Arendse Bernth and Michael McCord**

IBM T. J. Watson Research Center  
P.O.B. 704, Yorktown Heights, NY 10598  
arendse@watson.ibm.com mccord@watson.ibm.com

## Abstract

This paper reports on certain aspects of English→Danish machine translation, using a newly redesigned version of LMT that is implemented in C and is capable of translating Web pages. We use the translation of a Web page about Tivoli Gardens as a springboard for discussion of LMT's treatment of some phenomena of English→Danish translation, with emphasis on the contributions of a new transformational system for LMT.

## 1 Introduction

We would like to take you to Tivoli Gardens, even though it is midwinter. Our group at IBM Research has been working on a new version of the MT system LMT (McCord, 1989a, 1989b; Bernth and McCord, 1991), and we are applying it currently to the translation of Web pages. We will illustrate a new English→Danish LMT prototype on the translation of a nice Web page about Tivoli Gardens.

For several years, the LMT system was developed in Prolog. The Personal Translator English↔German LMT product (Lehmann, 1995) is implemented largely in Prolog. However, three years ago our group at IBM Research decided to redo our natural language software in C. This move was originally prompted by a desire for increased speed and greater portability; but it turned out to be a great excuse for making various design improvements, even though the overall structure and theory remain the same.

First the Slot Grammar system (McCord, 1980, 1990, 1993) and our grammar checker EasyEnglish (Bernth, 1997) were rewritten, and then the group (including Sue Medeiros) turned to redoing the LMT shell. We have concentrated so far mainly on the English→German version (with Claudia Gdaniec as our German expert), but have also done a partial porting of the existing Prolog English→Danish version. (The Prolog version was begun by Bernth and developed more fully by Hanne Jensen.) The group will soon turn to the development of other language pairs in the new framework, since by now the core engine is rather well developed.

In this paper, we will describe some central aspects of the new LMT system through the English→Danish version (EDLMT). As opposed to some of the other LMT language pairs, e.g. English→German, EDLMT may still be considered a “toy” system; for instance the transfer lexicon is fairly small, about 8000 lemmas. Nevertheless, we can give a good illustration of several technical features of the new LMT in the context of English→Danish.



LMT is a transfer system comprising four major steps: Source analysis, lexical transfer, transformations, and target morphological generation. We will not attempt, in this short paper, to describe the whole system. The transformational step is the most interesting for present concerns, and we will emphasize it in our examples.

The transformational phase uses a newly designed transformational formalism. It is a language in its own right, without any “escapes” to an underlying programming language. It has a Lisp-like syntax, and is interpreted by our C programs in the LMT shell. The transformational formalism is described in detail in (McCord and Bernth, 1998); here we shall only give a brief description in the context of our examples for EDLMT.

## 2 Web Page Translation

The new C version of LMT can process complete HTML files (as well as documents in other mark-up systems), identifying the natural language segments and replacing them by their translations. The treatment of segment-internal HTML tags and punctuation (“mark-up”) is non-trivial, since (1) the target language may have different constituent order and the mark-up in the target output must be placed appropriately, and (2) the meaning of segment-internal mark-up (e.g. a quote) may depend on segment-external mark-up.

Together with W. Rubin, we have developed “WebLMT”, which allows users of a Web browser to get on-the-fly translations of Web pages using a proxy server on which LMT is running. The user can select the target language (so far, English is the only source language). WebLMT shows both the source and target pages in a 2-frame browser window. All of this can be done without any additional installation of software on the user’s machine.

Figure 1 shows the WebLMT translation for English→Danish of our Tivoli Web page. The URL is:

<http://www.danbbs.dk/~ais/copenhagen/tivoli.html>

This Web page was created by Hans-Henrik T. Ohlsen, and is used here with his permission.

For the remainder of this paper, we will use the text of our example as a means of illustrating various features of EDLMT. We organize the discussion in terms of the linguistic phenomena that are treated by EDLMT—which comprise some of the main problems in English→Danish translation. The point is not to break any new linguistic ground, but to show how these phenomena are treated by LMT, with emphasis on the transformational component.

## 3 Treatment of Definite Noun Phrases

Our first problem is the treatment of Danish definite noun phrases—corresponding roughly to English noun phrases with a definite article determiner (“the”). The Danish NP may have an overt definite article determiner or a clitic on the head noun of the noun phrase. We will not deal with the differences between English and Danish in the usage of definiteness, but demonstrate our treatment of definites once definiteness has been established. Definiteness in the Danish NP may be introduced because of a definite article in the English, in which case it is introduced by lexical transfer; or it may be introduced for other reasons, by a transformation.

Whether the Danish NP uses a clitic or not depends on a number of factors. The basic case is clitic and appears when the NP has no adjectival premodifiers (with a few exceptions such as “hel” (“whole”)) and no restrictive relative clause postmodifiers; otherwise an overt definite article is used in Danish. See e.g. (Diderichsen 1987; Koefoed 1987; Jones and Gade, 1985).

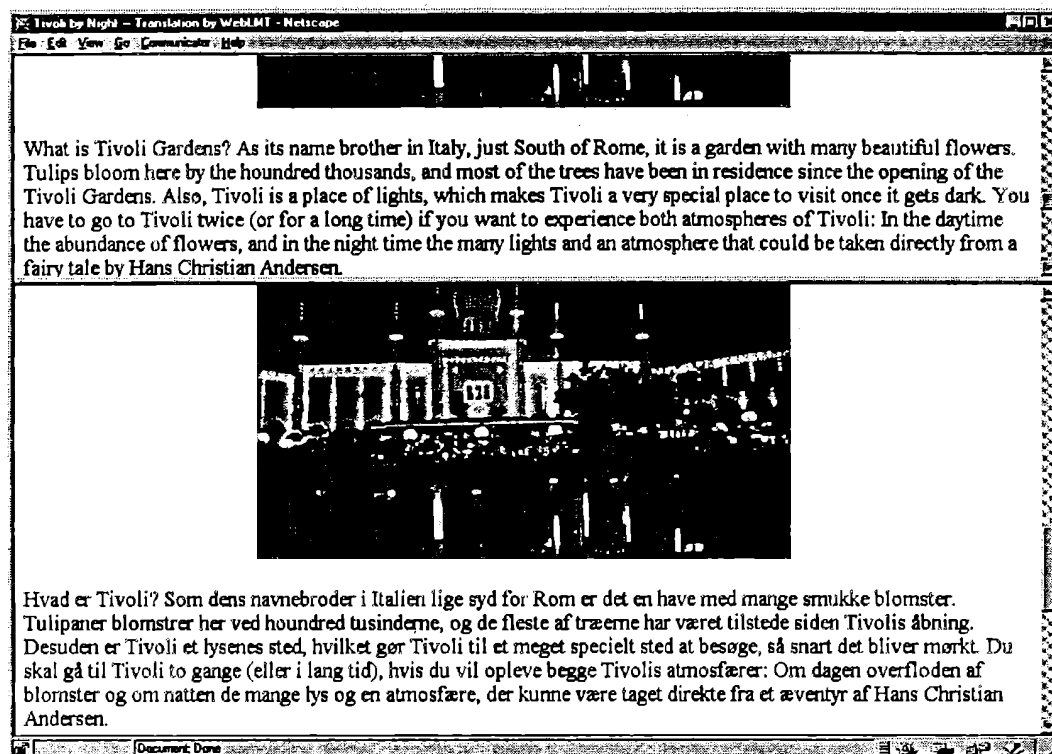


Figure 1. Web Page Translation by WebLMT

The issue is exemplified in just about every sentence of the Web page. Here is one noun phrase that shows both the clitic and the non-clitic cases:

“the abundance of flowers, and in the night time the many lights” ⇒

“overfloden af blomster og om natten de mange lys”.

The factors determining whether to use the clitic or the non-clitic determiner are structural, and specific to the target language (Danish). In LMT, such issues are handled by the transformations, which can explore, test, and change the tree structure as it is converted from the source language form to the target language form. The trees operated on by the transformations are expressed in terms of target language words and features (resulting from lexical transfer), but they begin with source language tree structure.

One of the features shared across a noun phrase (but not across coordination) is the feature “Definite”. During lexical transfer, “Definite” is set to true (meaning a clitic will be used) as the default, and the actual definite determiner is zeroed out (by default) by setting one of its features, the feature “IsZero”, to true. If necessary, the transformational system will later change these defaults, using a transformation called **noncliticdet**.

We will show here a somewhat simplified version of **noncliticdet**. It has two parts: one dealing with the case of a premodifying adjective, and one for the restrictive relative clause. These two rules could be combined into one rule with a disjunction, but it is easier to read separate clauses, which will be tried in order of appearance until one succeeds, or there are no more clauses for the rule (as in Prolog clauses with cuts).

Here is the clause for the case of a premodifying adjectival, to be explained below:

```
noncliticdet
  (tf ((POS . noun)))
  (tm ( *v1
        (v2 (th d))
        (v4 (ts nadj))
        *v5
        head
        *v6 ) )
  (cfl v2 (IsZero . F))
  (cfl (Definite . F))
```

The transformational algorithm will attempt to apply this transformation to every node. As it is tried for a given node *n*, we call *n* the *node in focus*.

This transformation has four top-level operations, with operators `tf`, `tm`, `cfl`, and `cf1`. In the name of an operation, a beginning “t” indicates a *test*, and a beginning “c” indicates a *change*.

After that (in the operation name), we use “f” for features, “m” for modifiers, “h” for head word (citation form), and “s” for slot (filled by the node in question). A variable name is a “v” followed by an integer, and the prefix “\*” on a variable signals that the variable is a *sublist variable*, which can match any sublist of a modifier list.

The first operation, with operator `tf`, tests that the node in focus has part of speech (POS) noun. Generally, `tf` can take any list of feature-value pairs, and can not only test, but also assign values to transformation variables.

The second operation, with operator `tm` (“test modifiers”), gets to the heart of the matter. Its argument is a pattern that is matched against the modifier list of the NP. The pattern has six pattern elements, displayed vertically here.

The first pattern element, `*v1`, matches any predeterminers (a sublist variable may match nil).

The second pattern element, `(v2 (th d))`, illustrates a general feature of the pattern language. A pattern element may be of the form

```
(Var Op1 Op2 ...)
```

where the variable `Var` matches a modifier node and the operations `Opi` act as tests (and must succeed), as if the matched node were the node in focus. These operations can recursively invoke the full power of the transformation language. The variable `Var` may also be a sublist variable, in which case the operations are tried on every node in the matched sublist. In the present case, there is one operation, `(th d)`, which tests that the node matched has head word “d”,

which happens to be the form that the transfer lexicon gives to the Danish definite article before inflection. So this pattern element just matches the definite article.

The third pattern element, (v4 (ts nadj)), requires v4 to match a node which fills the nadj slot. The nadj slot can be filled by adjectives, verb participles, and numbers. (There are additional possibilities, which we will leave out here for the sake of simplicity.) So this pattern element finds a key thing for the transformation, an adjectival premodifier.

After this, we have any modifiers \*v5, followed by the head word, followed by any postmodifiers \*v6.

If this modifier pattern succeeds, then we continue and the first cf1 operation will apply. This operation, (cf1 v2 (IsZero . F)), looks at the determiner node v2 and changes its IsZero feature to F (false). (Recall from above that the default setting (before transformations) is that IsZero = T).

Then we do the next change, (cf1 (Definite . F)). Note that this application of cf1 has only one argument, the feature setting. Generally, when the node argument of an operation is omitted, it is understood to be the node in focus, which in this case is our NP. So this cf1 operation sets the Definite feature of the NP to F.

So if an adjectival premodifier exists, our transformation sees to it that the definite article is overt (IsZero = F) and the head noun will *not* get the (default) clitic (Definite = F).

If this transformation fails on a node, then the next rule for **noncliticdet** is tried on that node. This is the rule that tests for a restrictive relative clause postmodifier.

```
noncliticdet
  (tf ((POS . noun))
    (~ (tsourcepunc ,))

  (tm ( *v1
        v2 (th d))
      *v5
      head
      (v6 (| (ts nrel) (ts nrela)))
      *v7 ) )
  (cf1 v2 (IsZero . F))
  (cf1 (Definite . F))
```

In this version of the rule, we test that the relative clause is *not* preceded by a comma in the English source (via the tsourcepunc operator); this fairly reliably tests that the (English) relative clause is restrictive. The disjunctive test (| (ts nrel) (ts nrela)) on the modifier node v6 tests that v6 is a relative clause.

Note: The relative clause rule for **noncliticdet** is ordered after other transformations that convert certain English NP postmodifiers (like participial clauses) to Danish relative clauses, so that we are really testing for the occurrence of Danish relative clauses, even though the English may have used some equivalent postmodifier.

## 4 Verb-Second Rule

Danish is a verb-second language—meaning that in an independent finite clause, the finite verb is the second constituent. Since this is a structural issue, it is a good candidate for a transformation.

The **verb-second** transformation is exemplified by

“Som dens navnebroder i Italien lige syd for Rom det er en have med mange smukke blomster.” ⇒

“Som dens navnebroder i Italien lige syd for Rom er det en have med mange smukke blomster.”

“Desuden Tivoli er et lysenes sted” ⇒ “Desuden er Tivoli et lysenes sted”

Figure 2 shows the output of the main steps in translating the last example: the source analysis tree, the lexical transfer tree, the tree resulting from application of all the transformations, and the target string resulting from Danish morphological generation.

The **verb-second** transformation interchanges the verb “være” (infinitive for “er”) and the subject “Tivoli”, in order to make “være” second, since there is an initial adverb. Other transformations are involved in this translation.

Syntactic analysis no. 1 Evaluation = 2.321000...

vadv	also1(1)	adv
subj(n)	tivoli1(2)	noun proppn sg
top	be1(3,2,5)	verb vfin vpres sg vsubj
ndet	al(4)	det sg indef
pred(n)	place1(5,u)	noun cn sg
nprep	of1(6,7)	prep
objprep(n)	light2(7)	noun cn pl

Transfer tree...

vadv	også	((POS . adv) (PCS . posi) (HasWh . F) (Case . nom) (
subj	Tivoli	((POS . noun) (NType . proppn) (NClass) (CClass) (Pe
top	være	((POS . verb) (VInfl . vfin) (Tense . vpres) (Mood .
ndet	en	((POS . det) (DType . indef) (Person . pers3) (Numb
pred	sted	((POS . noun) (NType . cn) (NClass r) (CClass) (Per
nprep	af	((POS . prep) (Case . dat) (SSlot . nprep))
objprep	lys	((POS . noun) (NType . cn) (NClass o) (CClass) (Per

Restructured tree...

vadv	desuden	((POS . adv) (PCS . posi) (HasWh . F) (Case . nom) (
top	være	((POS . verb) (VInfl . vfin) (Tense . vpres) (Mood .
subj	Tivoli	((POS . noun) (NType . proppn) (NClass) (CClass) (Per
ndet	en	((POS . det) (DType . indef) (Person . pers3) (Numb
objprep	lys	((POS . noun) (NType . cn) (NClass o) (CClass) (Pers
pred	sted	((POS . noun) (NType . cn) (NClass r) (CClass) (Pers

Desuden er Tivoli et lysenes sted.

Figure 2. Steps of LMT Translation

## 5 Intrasentential Punctuation

Intrasentential punctuation is a highly language-specific issue. For Danish, this is determined by sentence structure (at least in “traditional” Danish punctuation). This is also an ideal candidate

for transformations. The punctuation transformations should be applied at the end of the transformational process, since they need to work on the final structure of the sentence.

The punctuation transformations apply to many of the sentences in the Web page example; even where the English punctuation happens to coincide with the Danish, the punctuation is actually added by a transformation.

One example of the punctuation transformations is shown in:

“Du skal gå til Tivoli to gange (eller i lang tid), hvis du vil opleve begge Tivolis atmosfærer.”

Here a comma is inserted between the main and dependent clauses.

## 6 Compound Nouns

In Danish, compound common nouns are mostly written as one word with a potential connecting sound (usually “e”, “s”, “er”, “en”, or “t”). Morphological rules determine the combining forms of nouns, as illustrated by “name brother” → “navnebroder”.

A simplified version of the particular heuristic employed here is:

*If the noun ends in two consonants, the last of which is not “d”, then the combining form is noun+ “e”.*

Of course there are many exceptions to the general rules, and it is not possible to get *all* combining forms right based on general rules alone; however, in the absence of suitable lexical information, rules provide a noticeable improvement in translation quality.

## 7 Target Word Selection

The LMT transfer lexicon can specify tests that determine the preferred transfer of a given source word (sense), based on context in the source analysis tree. The most common tests are on the *complements* of the source word sense, and these tests may be on semantic types (using a type hierarchy), on morphosyntactic conditions, or on specific words or phrases. However, target selection tests can examine any part of the source tree, and a general mechanism for getting to any part of the tree and making a test is the *path test*.

A path test consists of a “path” (describing a way to get to some part of the tree from the node in question), plus a test to be performed on the destination node. For EDLMT, a path test takes care of translating the relative pronoun “that” into “der” if it is the subject (and not a right conjunct), and into “som” otherwise. This gives a nice variation even though “som” also could be used for the subject.

A simplified version of the entry for “that” shows this. (For the sake of simplicity, all other analyses of this word have been omitted.)

```
that
  < pron sg def
  > if (pth (sf subj) --u (sf nrel)) der
```

```
if (pth --u (sf nrel)) som
else den (png pers3 sg nt)
```

This entry first gives the head word “that”, then the source element (preceded by “<”), and then the corresponding target element (preceded by “>”). In the target element, a path test (signaled by “pth”) tests whether the word fills the subject slot, and whether its mother (“u” for “up one level”) fills a relative clause slot. (The latter is a test that the pronoun is a relative pronoun.) If the test succeeds, the transfer is “der”. If it does not succeed, the next test is tried. This just checks whether it is indeed a relative pronoun (but not a subject), in which case the transfer is “som”. If it is a pronoun, but not a *relative* pronoun, the translation is “den”.

An example is:

```
“an atmosphere that could be taken” ⇒
“en atmosfære, der kunne være taget”
```

The Prolog version of LMT made use of a special anaphora resolution module (Lappin and McCord, 1990a, 1990b; Lappin and Leass, 1994); this is not implemented for the C version yet.

## 8 Unknown Words

Words not found in the lexicon are assumed to be proper nouns and are not translated. Of course, this also applies to spelling errors. The Web page has a spelling error: “by the houndred thousands”, which is the source of a bad translation: “ved houndred tusinderne”. The translation should have been “i hundredetusindvis”.

## 9 Conclusion

Our trip to Tivoli Gardens has illustrated a number of features of EDLMT directly: Treatment of the definite article by way of a transformation; treatment of difference in word order exemplified by the verb-second rule; intrasentential punctuation; combining forms of nouns; and transfer choices in the lexicon.

In addition to these, a number of other features have been illustrated indirectly. The most notable are the following: Our treatment of HTML files, which allows us to hook up LMT to do on-the-fly translation of Web pages; the high quality of the source analysis by English Slot Grammar, which allows accurate transfer on both a lexical and a structural level; and the Danish generation morphology. These features comprise both language-specific parts and language-independent parts.

Our focus has been on the parts that are specific to the language pair English→Danish, but with emphasis on language-independent methods (transformations and lexicons).

All of these features and methods contribute to the quality and usefulness of LMT.

## References

- Bernth, A. (1997). "EasyEnglish: A Tool for Improving Document Quality," *Proc. Fifth Conference on Applied Natural Language Processing*, pp. 159-165. Association for Computational Linguistics.
- Bernth, A. and McCord, M. C. (1991). "LMT for Danish-English Machine Translation," In C.G. Brown and G. Koch (Eds), *Natural Language Understanding and Logic Programming, III*, pp. 179-194, North-Holland.
- Diderichsen, P. (1987). *Elementær Dansk Grammatik*, Gyldendal.
- Jones, W. Glyn and Gade, K. (1985). *Danish—A Grammar*, Gyldendal.
- Koefoed, H. A. (1987), *Teach Yourself Danish*, Hodder and Stoughton.
- Lappin, S. and McCord, M. C. (1990a). "A Syntactic Filter on Pronominal Anaphora for Slot Grammar," *Proceedings of the 28<sup>th</sup> Annual Meeting of the ACL*, pp. 135-142.
- Lappin, S. and McCord, M. C. (1990b). "Anaphora Resolution in Slot Grammar," *Computational Linguistics*, vol. 16, pp. 197-212.
- Lappin, S. and Leass, H. (1994). "An Algorithm for Pronominal Anaphora Resolution," *Computational Linguistics*, vol. 20, pp. 535-561.
- Lehmann, H. (1995). "Machine Translation for Home and Business Users," *Proc. MT Summit V*, Luxembourg, July 10-13.
- McCord, M. C. (1980). "Slot Grammars," *Computational Linguistics*, vol. 6, pp. 31-43.
- McCord, M. C. (1989a). "Design of LMT: A Prolog-based Machine Translation System," *Computational Linguistics*, vol. 15, pp. 33-52.
- McCord, M. C. (1989b). "LMT," *Proceedings of MT Summit II*, pp. 94-99, Deutsche Gesellschaft für Dokumentation, Frankfurt.
- McCord, M. C. (1990). "Slot Grammar: A System for Simpler Construction of Practical Natural Language Grammars," in R. Studer (Ed.), *Natural Language and Logic: International Scientific Symposium*, Lecture Notes in Computer Science, Springer Verlag, Berlin, pp. 118-145.



# **Drug terminology**

## **A multilingual term database. The AVENTINUS project.**

**Christian Sjögreen**  
Språkdata/Dept. Of Swedish language  
Göteborg University  
Box 200  
SE-405 30 Göteborg  
Sweden

### **Abstract**

This paper starts with a brief overall presentation of the AVENTINUS project, merely a list of the different included modules and some comments. Then follows a discussion about drug terminology and finally a description of the design and implementation of a tailored multilingual drug terminology database in MS Access. The used tags and links are presented and discussed and the inputting situation is described. Then some numerical details of the database and a short concluding remark are given.

### **Project description**

The AVENTINUS Project<sup>1</sup> aims at supporting an Advanced Information System for Multinational Drug Enforcement. It is funded by the European Union in the Linguistic Engineering (LE) Program, and has several development and user partners. The goal of the project is to support drug enforcement with multilingual linguistic expertise. AVENTINUS will support communication by providing linguistic tools to overcome language communication barriers. Users should be able to access information and receive results of search requests in their own native language, even if the information is derived from foreign language sources.

The languages dealt with in the first phase are English, German, Spanish, and Swedish.

AVENTINUS will provide modules and components that can be linked to and integrated into the users domestic environments. Modularity and integratability are the most prominent features of the software solutions to be provided.

The participating users are domestic police organisations and intelligence agencies and the Euro-pol Drug Unit (EDU). Interest from other authorities have also been noticed, though.

---

<sup>1</sup> For an exhaustive description of the project, see [THUR97] or take a look at our AVENTINUS web page at

The data to be handled by AVENTINUS are of several types, the most important ones for AVENTINUS being textual data. Texts from **open sources**, mainly newswire texts and **internal communication** texts, like police reports, will be considered.

According to the User Requirements Report, there are two main scenarios that should be supported, the **Indexing (Data Entry) Scenario** and the **Retrieval (Analysis) Scenario**.

In the Indexing Scenario, users are confronted with incoming texts from different sources (fax, telex, electronic) in different languages. They have to decide whether a given input text is relevant or not. AVENTINUS will support this scenario by providing translation support tools, and by providing information-understanding tools (indexing, information extraction).

In the Retrieval Scenario, search support will comprise tools like name search (transliteration, similarity of names) or text search both in structured and textual databases. Translation support tools will be responsible to translate the search requests as well as the search results (structured or textual) into the native language of the searcher

To support the scenarios, AVENTINUS will provide different types of components, such as **Translation Support**, **Information Processing Support**, and **Search Support**.

The project will have three types of translation support tools, *Term Substitution*, *Translation Memory*, and *Machine Translation*. All of them will be available as stand-alone tools accessing common lexical resources, and as components to be called from standard Windows editors such as WinWord. There will be several components to process the incoming texts, and provide further information for later retrieval, for instance *Information Extraction* and *Indexing*

Search Support refers to several requirements, for instance (i) *requests in natural language*, as well as in some *structured form*, (ii) *requests in a native language* instead of the foreign languages of the database to be searched and (iii) *query expansion* and navigation possibilities in the area of text search. Search in both structured databases and in a textual ones will be supported. The components to be offered comprise the following: *Name Search*, *Search in texts*, and *Search in structured databases*. In order to support the AVENTINUS application, three types of linguistic resources will be set up which have to do with both multilingual issues and domain modeling: *Lexical Database*, *Thesaurus*, and *Domain Model*

The architecture of AVENTINUS follows two basic principles. It must be based on components that can be integrated in a very flexible way into the existing system environments of the users and it must be very flexible in the interaction of the internal components. In many components the AVENTINUS functionality may be called from a standard text processing system. The interface will be available on several platforms.

A first version of the AVENTINUS data pool, including test texts and terminology, is implemented, as a pool to create resources and test specifications. The complete system specifications

have been written and reviewed, some of the AVENTINUS components (translation memory, machine translation) are operational and a test plan is available, with Europol as the first testing environment.

### **Drug terminology**

Our assignment in the project is currently twofold. One is to collect, structure, link and 'linguistically' edit the drug terms for all languages involved. The other one is to develop linguistic resources for Swedish. Only the former will be discussed in this paper. It is mainly the responsibility of Maja Lindfors Viklund, Yvonne Cederholm and myself, where my job is and has been to design, implement and maintain a database to store and link the terms locally.

Drug terminology [MLV97] differs from 'normal' terminology in a substantial way—as probably most criminal terminology does—as it's partly used not to make communication easier but rather to hide facts. The fact that many of the terms are slang words or argot only emphasises this difference still more. Normally, one can assume a terminological environment to cover a rather specific, well defined domain, and to be rather consistent with respect to ambiguity and stability in meaning and also often in growth. In the case of drug related terminology we face a quite different situation. The domain includes such opposite areas as street slang, police and custom vocabulary, drug legislation, medical treatment, complex chemical compounds. New products—based upon new chemical formulas (referred to as *designer drugs*)—are constantly developed to keep the trade ahead of the legislation since a drug is not prohibited in our society until it's explicitly put on the list of illegal drugs, i.e. classified as narcotic.

The terms in our drug terminology cover areas such as:

- *cultivation* – geographical areas, traditions and methods
- *production* – handling of substances
- *substance* – types of drugs, names
- *trade* – related places and persons
- *equipment* – tools and accessories
- *abuser* – often nicknames according to the drug in question
- *symptom* – behaviour and experiences

We have until today collected some 13,000 terms all together. Mostly English (roughly 5000), and Swedish (4000), but also some 2000 German and about the same amount of Spanish. Since we work in Sweden it has apparently been easier for us to collect Swedish terms than terms from the other languages involved and that explains why we, relatively spoken, have rather many Swedish ones. It is also quite natural that the number of English terms is equally high, or in fact higher, since we are using English as a pivot language. In addition there is the obvious influence from the Anglo-American cultures upon the western European countries, reflected both in life and language. The relatively low amount of German and Spanish terms is, to some extent, explained by the fact that we still haven't come that far in collecting terms from these languages.

When the project was designed, it was assumed that the users, mainly police and intelligence organisations and the EDU, should provide us with texts and lists of terms. But in reality this has not worked very smoothly, perhaps due to the rather delicate nature of these texts and lists. Most of the police organisations have been very reluctant to give anything away. This is understandable but problematic. There are some exceptions though. The Swedish police organisations have been very co-operative and have supplied the project with a lot of material. For instance, we have a very good relationship with Svenska Narkotikapolisföreningen (SNPF), with the Swedish officials in Europol, and with the Swedish National Police Academy. They have supplied us with a lot of materials and in return, I might add, we have had some opportunities to help them [Hol97].

Among other things SNPF has given us—on diskette—the text to their book *Basfakta om narkotika* [SNPF96] (Basic facts about narcotics). The book deals with almost everything in this context and we have been able to extract numerous terms for drugs, tools, treatment of addicts, legislation, etc. from this text. Another main source for the collection of drug related terms is *Internet*. It's amazing what you can find there. Price lists, recipes, articles, etc. etc. So, surfing the Internet has become more work than pleasure for us, at least in this respect. We have also collected newspaper articles about drugs and other related topics and set up word lists and concordances to find more terms.

### The 'tagset'

The initial bulk of terms that we collected were prepared in a unix environment and imported into the database where the terms then have been further analysed. The tags we use are:

1. *Part of speech*
2. *Type of term* – D: 'drug term', G: 'General language term'
3. *Language* – EN, ES, DE or SE
4. *Definition* – given in English and/or in a 'native' language
5. *Subdomain* – type of drug
6. *Concepts* – links to the domain model
7. *Comment* – free comment
8. *Original language* – Chinese, Swahili, Inca, ... whatever (not used for the moment)

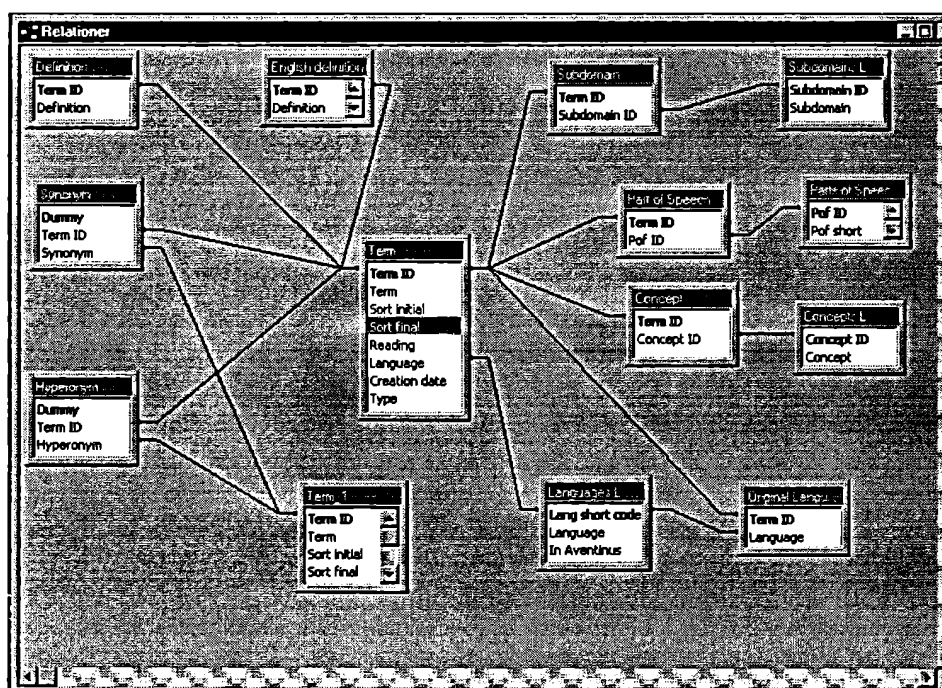
The three first tags are compulsory. The rest are, from the database point of view, optional. Of course, when a term is fully analysed, all relevant tags should be given. The *subdomain* tag is used to classify the terms into drug categories like *cannabis*, *cocaine*, *heroin*, *opium* etc. The *concept* tags classify the terms into categories within the area of drugs and related (criminal) activity and behaviour, for instance *drugs*, *tools*, different types of *geographical locations*, *organisations*, *persons* etc. The set of concepts is decided upon in co-operation with the AVENTINUS group at the university in Sheffield who works with the *ontology* (i.e. the world model, or rather in this case the domain model). The concept tags are hence the links from the drug terminology to the ontology in the project.

We use three semantic links, of which the first one is optional in the same sense as above, while the two others are truly optional.

- Language *equivalence* – to a term in the pivot language
- *Synonym* – to term(s) in the same language
- *Hyperonym* – to term(s) in the same language

## The database

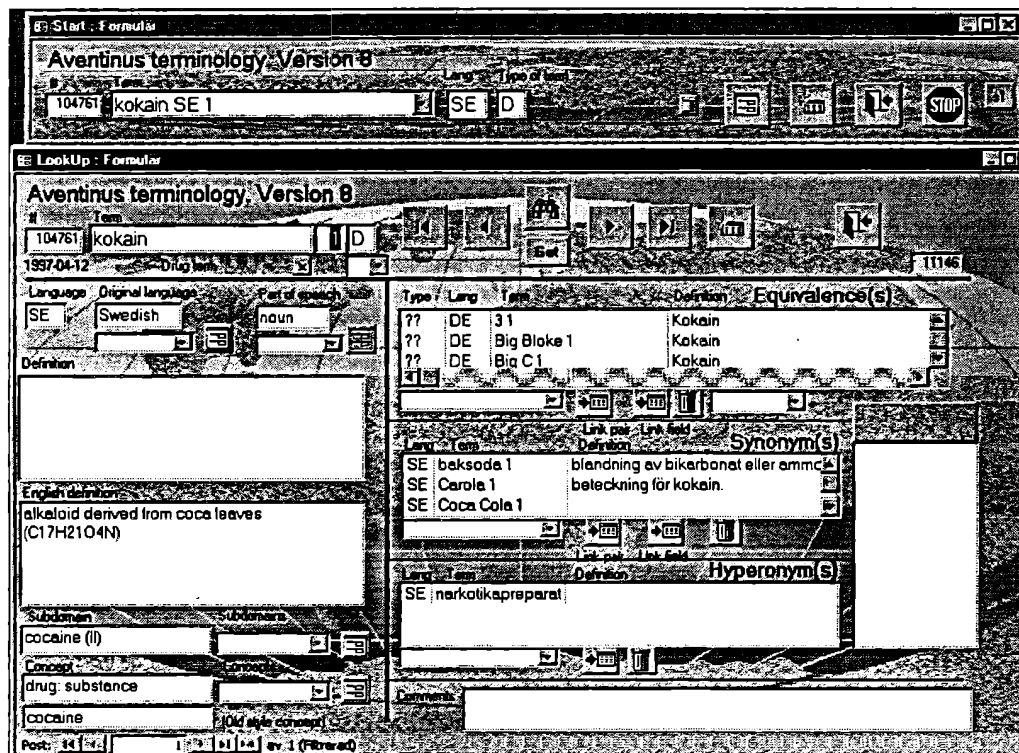
The terms and the tags are stored in a relational database implemented in MS Access 8.0. The main table is called *Term* and contains a unique identification number, the term itself, a normalised form, mainly used for sorting purposes, a reading number (in case of homography or polysemy) and a short form for language. The concepts and subdomains are stored in separate tables *Subdomains L* and *Concepts L* with unique identification numbers and the links in special link tables *Subdomains* and *Concepts* with pointers to *Term* and *Subdomains L* and *Term* and *Concepts L* respectively.



Database relations

There are other tables in the database than those shown in the figure. For instance the rather important table *NextAvailableNumber*, that provides new terms, and some other entities as well, with unique ID-number. All links between terms and properties use the id-numbers instead of the terms themselves. This technique ensures that all established links are kept unchanged even if the terms are corrected for misspellings or if the ordering of reading numbers are changed or whatever. To ensure that no links are set unexpectedly these ID-numbers are never allowed to be reused. That is, if a term for some reason is deleted from the table *Term*, the freed id-number is just thrown away.

## The interface



*The AVENTINUS Drug Enforcement Input Screen*

Existing terms are selected in the smaller upper window in the list field, i.e. a field that lists all entries in the table *Term* in alphabetical order. By keying in one or more letters in the field, the list positions itself to that part of the alphabet. When a term is selected, all tags are automatically fetched and made available for editing in the bigger window.

To enter a new term one clicks on the *New term*-button in either of the two windows. Then a separate input window appears that forces the user to input not just the term, but also the *type* and the *part of speech*. This window will also give information about the presence of any other homographs to the one given and if so gives a warning.

When a non-English term is entered it shall be linked to an English equivalent term. If none exist we try to find one somewhere in our sources. If that isn't possible we can in most cases get a link through a hyperonym and as a last resource we invent a term by 'plain translation'. The linking is carried out by selecting the target(s) from a list field that contains the already existing terms in the same language (synonyms and hyperonyms) or in English (equivalents).

When an equivalence link is established, all other terms in the other two non-pivot languages that are linked to the same pivot term will automatically pop up in the *Equivalence* field. Thus one can get a picture of the popularity of a term (= drug) in the other two non-pivot languages. In the example of the Swedish term *kokain* in the figure there are no less than 88 links, of which one is the pivot term (in English) while the other represent the bouquet of synonyms for this popular drug in German (66) and Spanish (21). The corresponding list of (15) terms in Swedish is shown in the synonym field. To see the English synonyms we just pick the English counterpart as the main entry. In the case of *cocaine* there are no less than 99 synonyms.

When a non-English term is linked to a pivot term in English, it will automatically be linked to a *subdomain* and a *concept* via the already established links from the pivot term. Since not all these terms are linked yet—the project is still in progress—there is an optional possibility to link 'manually' as well.

To each term a definition may be given. The definitions may be given in English and/or in a native (normally Swedish) language. We got a set of terms from the German BundesKriminalAmt (BKA) and some of them were given a kind of rough definition in German. The 'native language definitions' are merely used as a way to give a preliminary definition, perhaps to be translated later or even handed over to a (human) translator. The main purpose of the definitions are today to serve as an aid and a bases for classification and linking.

At irregular occasions, the whole database is exported into textfiles and transmitted via the 'net' to GMS in Munich, where the projects main database is implemented.

### **Statistics and examples**

Here are some numbers, drawn from the database. Note that these numbers may differ from what is mentioned in the text, depending on the fact that all terms are not yet included into the database and certainly not fully analysed. Statistics like these reflect the presumed fact that the amount of terms for a certain drug, or class of drugs, is related to the usage in a society. Perhaps we will eventually find something more exciting than this rather obvious fact. But it's too early to go into this yet, the data has to be more complete first.

	English	German	Spanish	Swedish	Total
Terms	5414	2181	1719	1829	11143
Linked to equivalence		1860	985	1421	4266
<u>Subdomains:</u>					
Amphetamine	283	35	3	49	370
Cannabis	871	68	29	74	1042
Cocaine	600	55	20	56	731
Heroin	247	73	15	35	370
Opium	95	59	4	22	180
<u>Concepts:</u>					
Drug: substance	2631	545	110	229	3515
Drug: tool	65	3	1	18	87
Person: dealer	55	11	4	24	94
Person: user	108	7	11	43	169

The following examples from the database are thoroughly discussed in [MLV97].

*Inheritance example.*

English	German	Spanish	Swedish
deal	deal	dilear	dila
flip (out)	ausflippen	flip	flippa
snort	snorten	snortar	snorta
speed (n)	Speed	espid	speed

*Smuggler example.*

mula (ES)	burro (ES)
Körperschmuggler (DE)	
bodypacker (EN)	
bollbärare (SE)	
sväljare (SE)	culero (ES)
	vaginer (ES)

The 'smuggle' terms all means in principal the same thing, a person who smuggles drugs, but vary from the 'pack animals' via some rather neutral terms to the extremes that describe how, and even where in the body, the drugs are smuggled. This can be seen as an illustration of how cultural attitudes may be reflected in language.

*Capital letter example.* *H* is used as short for heroin<sup>2</sup> and has then been expanded to *horse*, perhaps to indicate the strength in the drug. The same term then shows up in the other languages as *H* and *Pferd*, *H* and *caballero*, and *H* and *häst*.

## Conclusions

The area of drug terminology is to our knowledge a rather unexplored field of research and shows some interesting deviations from 'normal' terminology. All the linking of terms to synonyms and hyperonyms and to English 'equivalents' results in a network of relations between terms that in many respects are not yet fully taken advantage of. So, when the database is 'ready', I think we will have a good framework to start some rather interesting issues.

---

<sup>2</sup> The figure 8 (*H* being the 8:th letter of the alphabet) is also used as a synonym for *heroine*.



There have also been suggestions to broaden the database both with respect to other languages and to introduce new domains. The database could then be used as a foundation to create an international police thesaurus. This may give us a lot to do in the future.

It seems to us that the Anglo-American influence on the drug vocabulary is more pronounced in newly introduced drugs, whilst older more established drugs tend to develop a domestic vocabulary. If this is a tendency that will survive is however hard to tell.

The work has been rather tedious and often quite difficult since none of us are a multilingual drug abuser with knowledge in slang from the streets of Hamburg, Liverpool, Barcelona or Mölndal. We have got good help though, from policemen and from people we know, with insight into the domain and/or the different languages.

At last I want to emphasise the use of MS Access in a number of different areas of research. It can be used to store small or medium sized databases and can easily be learned to master by linguists with a little help from their friends. It has become my standard desktop database tool.

### **Citations**

[Thu97] Thurmair, Gr.: *Multilingual Information Processing: The AVENTINUS system*. Paper given at the FBI conference in Berlin, Sep. 1997.

[MLV97] Lindfors Viklund, M.: *Drug terms*. Dept. of Swedish language, Göteborg University 1977.

[Hol97] Holmén, S.: *Pundartugg*. Narkotikarelaterade slanguttryck. Polishögskolan, 1997.

[SNPF96] Svenska narkotikapolisföreningen: *Basfakta om narkotika*, Göteborg, Sweden 1996.

# AVENTINUS, GATE and Swedish Lingware

**Dimitrios Kokkinakis**

Språkdata/Dept. of Swedish Language  
Göteborg University  
SE 405 30 Göteborg, Sweden  
svedk@svenska.gu.se

**Keywords:** AVENTINUS, GATE, Information Extraction, Swedish Lingware

## **Abstract**

This report presents an outline of the Swedish Lingware components integrated in the Language Engineering application development environment of GATE. This is an ongoing work within the framework of the Language Engineering project AVENTINUS.

AVENTINUS, *Advanced Information System for Multilingual Drug Enforcement*, is a multilingual information processing project financed by the European Union, Language Engineering program, with contract reference LE1-2238 10335/0. The AVENTINUS project is a user- and data-oriented project, addressing the needs of European police and law enforcement agencies on prevention and detection of drug-related offenses, such as money laundering, drug trafficking, drug abuse etc. AVENTINUS is also expected to be extended and used in related to drug enforcement fields such as organized crime.

The report will concentrate on a concise description of the ongoing development, integration and evaluation of Swedish lingware components in GATE. These components as well as the textual material used, that is corpora, of the narcotica subcorpora domain will be discussed.

First, a brief overview of the AVENTINUS project, its goals and characteristics are presented. Then a description of GATE and the information extraction, IE, task of the project will be discussed. Finally, the description of Swedish modules, comprising the necessary lingware for the IE task will be given. These modules and the interaction between them will be illustrated with concrete examples. Furthermore the software/lingware specifications, requirements and, when appropriate, some preliminary performance measures will be given.

## **1. Introduction**

This report presents an outline of Swedish Lingware components integrated in the Language Engineering application development environment of GATE. This is ongoing work within the framework of the Language Engineering project AVENTINUS. First, a brief overview of the AVENTINUS project, its goals and characteristics are presented. Then a description of GATE and the information extraction, IE, task of the project will be discussed. Finally, the description of Swedish modules, comprising the necessary lingware for the IE task will be given. These modules and the interaction between them will be illustrated with concrete examples.

Furthermore the software/lingware specifications, requirements and, when appropriate, some preliminary performance measures will be given.

## 2. The AVENTINUS Project

AVENTINUS<sup>1</sup>, *Advanced Information System for Multilingual Drug Enforcement*, is a multilingual information processing project financed by the European Union, Language Engineering program, (LE1-2238 10335/0). The AVENTINUS project is a user- and data-oriented project, addressing the needs of European police and law enforcement agencies on prevention and detection of drug-related offenses, such as money laundering, drug trafficking, drug abuse etc., referred to, in the rest of the report, as *drug enforcement*. AVENTINUS is also expected to be extended and used in related to drug enforcement fields such as organized crime. The economical and social impact of such project is evident, drug trafficking, dealing and consumption poses one of the greatest threats to the European nations. The goal of the project is thus to support drug enforcement with multilingual linguistic expertise. The AVENTINUS users should be able to access information in their own native language and receive the results of search requests in their own native language as well, even if the information is derived from foreign language resources.

AVENTINUS focuses on three main areas:

### **Translation support and tools**

term substitution, translation memory and machine translation;

### **Information processing**

components for IE, named entity recognition; intelligent indexing;

template generation, in standard as well as in Interpol forms (user requirements);

### **Search support**

query expansion, transliteration and name similarity.

In the first phase of AVENTINUS, the languages analyzed and processed are German, English and Spanish, while Swedish is part of the second phase.

It should be emphasized that the main task of AVENTINUS is not concentrating on constructing a new full-fledged information system for the drug domain, as the AVENTINUS users have their operational environment in place already. Instead, AVENTINUS provides modules and components which can be linked to and integrated into these environments. Thus, the two predominant features of AVENTINUS are modularity and integratability.

## 3. GATE/IE and AVENTINUS

### 3.1 GATE

GATE, *General Architecture for Text Engineering*, Cunningham *et al.* (1995, 1997), Gaizauskas *et al.* (1996b), is an application development environment which supports natural language engineering tasks. GATE has been chosen by AVENTINUS for the information extraction task. In GATE, heterogeneous natural language components/software may be evaluated and refined

---

<sup>1</sup> More information about the project can be found:

individually or may be combined into larger applications. GATE support both researchers and developers working on component technologies (e.g. tagging, parsing) and those working on developing end-user applications (e.g. information extraction, text summarization). GATE is an integrated collection of tools built upon a standardized model of storage and retrieval developed by the TIPSTER program. TIPSTER is an ARPA-funded program of research and development in information retrieval and extraction, involving the creation of a standard architecture/interfaces for systems dealing with information retrieval and extraction.

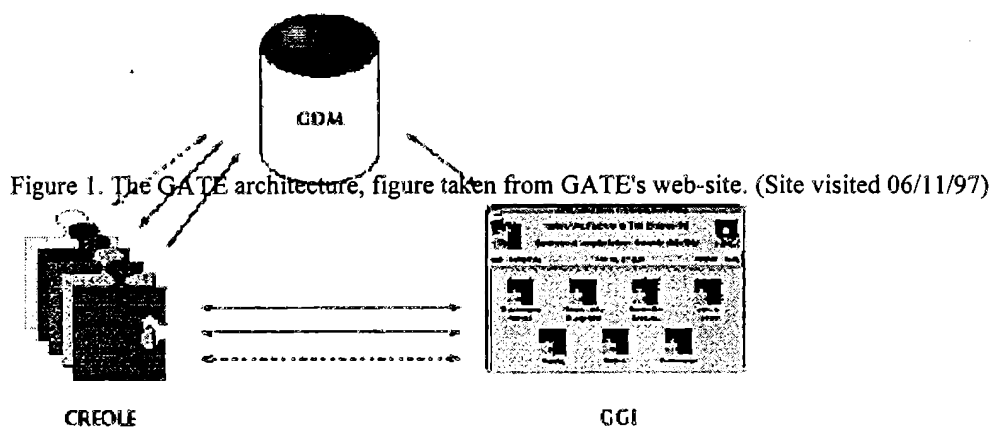


Figure 1. The GATE architecture, figure taken from GATE's web-site. (Site visited 06/11/97)

Figure 1. The Gate architecture, figure taken from GATE's web-site. (sote visited 06/11/97)

GATE consists of three main parts;

- The GATE document manager, **GDM**, based on the TIPSTER document manager, Grishman (1996);
- A Collection of REusable Objects for LE, **CREOLE**, which is a set of language engineering modules integrated with the system, such as taggers and parsers;
- The GATE Graphical Interface, **GGI**, which provides integrated access to the services of the other components.

### 3.2 Information Extraction

Information extraction, IE, is the mapping of unseen natural language texts into predefined, fixed-format, structured representations or templates. The templates, when filled, represent an extract of key information from the original text. The resulting data can be further used for intelligent indexing, for storing into a database, for queering or for just displaying it to an end-user. IE systems are often tied to a particular application domain or scenario, and they are computationally as well as knowledge intensive to build.

Four types of IE tasks are identified:

- Named entity recognition**, identification of all the names of people, places, organizations, dates, amounts of money;
- Coreference resolution**, identification of identity relations (anaphoric references) between entities in texts;
- Template element construction**, (TE), association of descriptive information

with the entities, such that "Luleå is\_a city in Sweden";  
**Scenario template construction**, linking of TE into events and relation  
 descriptions.

The VIE, *Vanilla IE*, system, following GATE, has been developed at the Univ. of Sheffield. VIE is used for building a model of a text for different purposes, such as MUC-6 tasks, text summarization and for compositionally constructing semantic representations of sentences integrated into a "discourse model". VIE is a modularized version of the lassie system, *Large Scale Information Extraction*, also developed at the Univ. of Sheffield, (Gaizauskas *et al.* 1995). VIE has been the starting point for the development of the Swedish Lingware.

### 3.3 Multilingual GATE/IE and AVENTINUS

AVENTINUS provides multilingual access to information extracted from texts. Therefore, the IE components have to cope with multiple input languages. This is solved by duplicating parts of the English IE system for each input language. Each language-specific IE system is producing results to each of the AVENTINUS output languages.

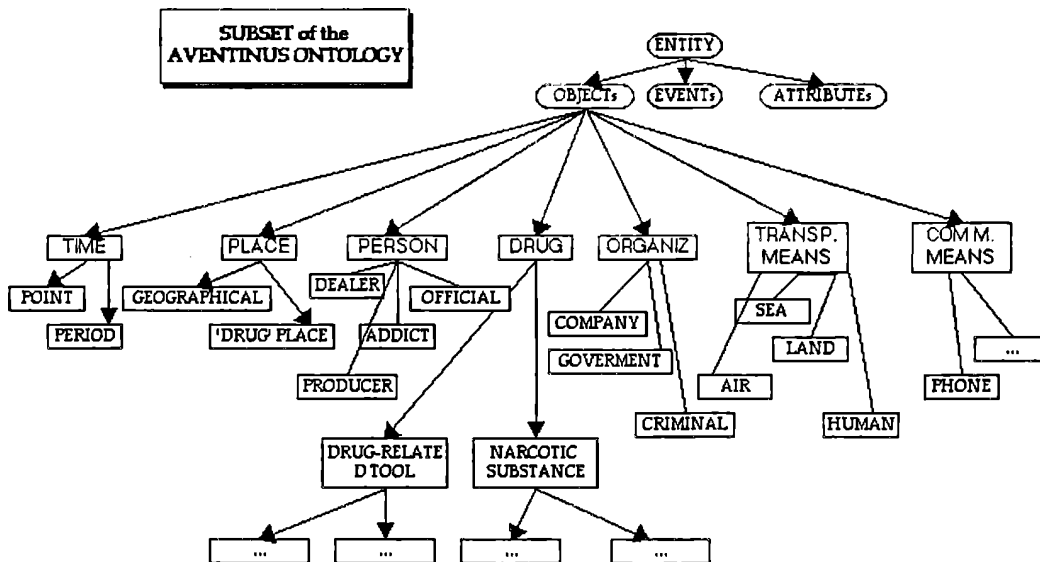


Figure 3. The AVENTINUS ontological objects.

Since AVENTINUS is interested on drug enforcement activities, (cf. Cunningham *et al.* 1996), the entities of interest are persons, communication and transportation means, places, organizations, dates, time sequences and of course drug substances and drug-related tools, (for the latter entities see Lindfors Viklund 1997). A number of relations between these entities, usually in the form of events, is also specified, for instance between people, buyer-seller, between people and companies etc.

## 4. Swedish Lingware in GATE

### 4.1 Introduction

The following chapters give a concise description of the ongoing development, integration and evaluation of Swedish lingware components in the language engineering application development environment of GATE. These components as well as the textual material used, that is corpora, of the narcotica subcorpora domain will be discussed. The following stages are carried out by the IE system:

Language Identification	Text type identification
Tokenization	Sentence boundary identification
Part-of-Speech tagging	Morphological analysis
Name entity recognition	Parsing
Coreference resolution	Discourse interpretation

### 4.2 Corpora, Overview

The text material, corpora, used in the system comprises three different document collections. These are: i)140+ police reports, ii)300+ drug-related newswire-texts downloaded from various Internet web pages and other text collections, and iii)a large number of miscellaneous reports, articles and publications, provided by different sources. The police reports have been provided by the EUROPOL data pool and the Swedish police authorities, Rikspolisstyrelsen. All of them consist of a header identification and a text body. The miscellaneous textual resources consist of machine readable versions of drug related literature, such as the book: "Basfakta om narkotika", provided by SNPF (1996), courtesy of Jonas Hartelius, "Pundartugg", ca 5000 slang words and expressions provided by the author of the book Stefan Holmén (1997), and a large number of on-line publications, reports, wordlists, glossaries and articles about drugs, both encyclopedic and non-encyclopedic.

### 4.3 Language Identification

#### 4.3.1 Overview

This is a module that uses Mark models for language classification developed by Ted Dunning (1994). No linguistic presuppositions are incorporated for this task. The only assumption used is that a text can be encoded as a set of bytes. The method used is to develop a set of level language models from training data and then to use these language models to estimate the likelihood that a particular test string might have been generated by each of the language models.

#### 4.3.2 Training and Output

For the training of this software, the classifier, a collection of texts for the given language(s) (>5k) is required, after that the creation of a profile file is generated for the given language(s). The output of the classifier is a set of scores plus the profile which had the best match. If a file F is in Swedish, the output will be:

```
"1 swe.3 F -415.93 -320.63"
```

where  $n_1$  is the number of profiles which had best match,  $score_{swe}$  is the profile with the best match, here the training is based on trigram sequences,  $F$  is the file being classified,  $-415.93$  is the score for English and  $-320.63$  is the score for Swedish, assuming that we provide profile files for these two languages.

#### **4.4 Tokenization and Text Type Identification**

Tokenization is the process of distinguishing individual tokens and their boundaries in a text-stream and it is of vital interest for lexicon lookup and correct annotation of any kind of labelling, tagging and even for the grammar development. The TOKENIZER module in GATE identifies these boundaries and returns byte offsets to be used as indices in the GDM database. The process is rather advanced and it has the capability to identify over 350 multiword adverbials, prepositions and conjunctions/subjunctions, and a large amount of phrasal verbs. The Text Type Identification (TTI), i.e. if a text is plain, html, email etc., is been carried out at the moment by the tokenizer. This will be a task of an independent module in the future. The TOKENIZER and the TTI are implemented via a set of regular expression patterns which are translated to C++ using flex. The Swedish tokenizer, is following the specifications of the English counterpart, which has been modified in order to account for the Swedish specific set of peculiarities. The TOKENIZER is domain independent. Certain AVENTINUS specific implemented patterns do not influence the analysis of other types of texts. Three major changes have been incorporated into that module.

- i) the integration of separate exclusive start states, accounting for the analysis of Swedish newspaper articles and for the analysis of Swedish police reports;
- ii) identification of SGML markup;
- iii) the integration of the Swedish multiword and phrasal verb recognition.

#### **4.5 Sentence Boundary Identification**

##### **4.5.1 Overview**

The SENTENCE BOUNDARY IDENTIFICATION (SBI), module identifies sentence start and end byte offsets, making use of SGML sentence markup if present. The module is a variant of the provided English module in GATE. This have been tested and modified accordingly, in order to suit the Swedish needs. The SBI module is implemented as a Perl script.

##### **4.5.2 Processing and Performance**

The result of this module is token start and end byte offsets, one pair per line, each followed by the corresponding token string. Newlines from the raw text are also treated as tokens. The byte offset pairs, one per line, represent the sentence start and end positions. The SBI module is reasonably domain independent. Sentence splitting is also affected by the way in which punctuation is handled by the tokenizer, which will vary between different cases. The content of the police reports, for instance, is containing a lot of uppercase tokens, acronyms and abbreviations which require special attention. Some of the stages in SBI are duplicated from the English SBI module, some are modified and some new are added. One of the new stages added is the following:

"A sentence end is assumed if a period is preceded by a sequence of lower case tokens followed by a token starting with a numeral". This is the case of:

"[... bakom smuggling.] [27-åringen häktades ...] "

There is a possibility to analyze texts having SGML markup, in this case, the input is simply searched for "<s>...</s>" pairs, and the corresponding offsets are written out. The performance of the module in the drug enforcement domain is >95% correct identification of sentence boundaries.

## 4.6 Part-of-Speech Tagging

### 4.6.1 Overview and Interface

Two taggers for Swedish have been tested in the current system, namely Brill's rule-based part-of-speech tagger, (Brill 1994), and Cooke's semantic tagger, SemanTag, (Cooke 1996). Brill's tagger have been trained in a subset of the SUC corpus, Ejerhed *et al.* (1992), as well as some other newspaper material. For the training in Swedish texts as well as an extensive description of the tagset see Johansson Kokkinakis *et al.* (1996) and Kokkinakis (1997a). The tagset comprises 14 categories, noun, verb etc., as well as SGML markup as a category of its own, 162 tags are used in total. The idea of using SemanTag is mainly threefold. The SemanTag is 9 to 10 times faster than Brill's, there is a possibility of extending the tagger with semantic information and it uses exactly the same resources as the Brill's tagger, so no re-training is necessary.

### 4.6.2 Processing and Performance

The taggers require four resource files in order to operate: a lexicon, a lexical and a contextual rule list and a list of common bigrams. Various threshold values are decided during training. Tuning flags can be also used. The current lexicon is comprised of 51000 entries, and 750 rules. Multi-word expressions, such as "på\_grund\_av" are part of the lexicon. The original lexical rule file generated was manually supplemented with rules identifying some common unambiguous endings, not generated during training, for instance "ornas\_hassuf\_5\_NCUPGD". The original contextual rule file was also been manually supplemented with rules, for instance with rules distinguishing participles from other verb forms, "VMNOA\_APON(SP)00\_PREV1OR2WD\_blivit". By the addition of these rules the performance has been now improved considerably, after evaluating large Swedish corpora and adding manually rules that resolve that sort of errors. The original performance of 92%-95% correctness is up to 96-99%.

The taggers expect a plain text file as input, formatted with one sentence per line.

Example:

```
"En 23-årig polsk medborgare häktades i går i Trelleborg misstänkt för grovt narkotikabrott ."
```

The default output is a version of the input file with a part-of-speech tag appended to each token.

Example:

```
"En/DIUSO 23-årig/AOPUSNI polsk/AOPUSNI medborgare/NCUPNI häktades/VMISP i_går/R i/S Trelleborg/NP misstänkt/APOUSNI för/S grovt/AOPNSNI narkotikabrott/NCNSNI ./F"
```



The lexicon used by the taggers is slightly biased towards the drug enforcement domain, because their resource files have been extensively tested on drug related texts. The tagger can be used in other domains with the same lexicon and rule sets, without serious performance drawbacks.

## 4.7 Annotated and Unannotated Morphological Analysis

### 4.7.1 Overview

The morphological analysis is producing the stem or root form for each token it is given, plus an affix. This module can be used either with part-of-speech annotated texts or just plain texts and only processing of noun and verb tokens is considered. The morphological analysis is implemented via a set of regular expression patterns which are translated to C++ using flex. These patterns cover over 90% of all the possible patterns for Swedish nouns and verbs, and originate from the analysis and the morphological grouping of the 11th edition of the Swedish Academy word list (SAOL), see Kokkinakis *et al.* (1997). In this analysis 208 morphological classes of verbs and 244 classes of nouns have been distinguished, very unusual or elderly patterns have been omitted for this task, though.

### 4.7.2 Performance

The input of this module is plain text on standard input with each token optionally annotated by VERB or NOUN to restrict the search time. The performance of the module in the annotated material is over 98% accuracy, while for unannotated texts is considerably lower and it ranges from 70-95%. This depends to a great extent on the ambiguity between identical verbal and noun suffixes, "ar/er", as well as homography, "hamnar/händer".

Example: "...föaren har gripits... => ...föare+n ha+r gripa+its..."

## 4.8 Name Entity Recognition, Gazetteer Lookup

The Gazetteer lookup module is trying to identify keywords and phrases related to named entities, as defined for AVENTINUS. This is accomplished by searching a series of pre-stored lists of organizations, locations, person names, date forms, currency names, etc. Most of the lists have been derived from various Internet resources, such as the *Genet Names Server* for a taxonomic list of several thousand of Swedish cities. These lists are translated into a series of finite-state recognisers using flex. The output of this process is a tag and a type attribute for every recognized named entity pattern. Most of the gazetteers are domain independent. Consider for instance a small subset of the content of the title gazetteer for the Swedish police enforcement agencies:

```
Rikspolischef                ställföreträdande Rikspolischef
Biträdande (Rikspolischef| Länspolismästare)  Länspolismästare
(byråchef|Rektor) på PHS      kriminal(kommissarie|inspektör)
polis(kommissarie| inspektör| assistent| aspirant|chefsaspirant|\
    sekreterare| mästare|överintendent|intendent)
```

The performance of this module is over 97% correct identification, while AVENTINUS requires that the quality recognition is >80% in recall and precision.

## 4.9 Parsing

### 4.9.1 Overview

Two separate grammars are used for the parsing task. First a named entity grammar is applied to construct proper noun phrases, then a full sentence grammar is applied. The parser is a bottom-up chart parser which builds a semantic representation compositionally, and a best parse algorithm is applied to each final chart, providing a partial parse if no complete sentence span can be constructed. The parser is written in Prolog. There are about 50 rules for the named entities with terminals like `title_NP`, `person_NP`, and 110 for the sentence grammar. The sentence level grammar has been derived by first acquiring a surface level partial grammar which has been manually completed by inspecting the results of applying it on the drug enforcement corpus, Kokkinakis (1997b).

The formal chart/5 term in the grammar is defined as follows:

```
chart(sentence:N1,
      first_token:T1,
      last_token:Tn,
      edges: [edge(Start_token,End_token,
                  Category(f1:v1,f2:v2,...),
                  Level,Start_offset,End_offset,Category,Parent,Child,ID)]
      next_edge_number:Ne).
```

Consider for instance the syntactic encoding and representation of the sentence:

"SILK skulle vilja använda lättare droger."

```
chart(sentence_n:1, 1,7,edges:[

      edge(1, 2, pn(s_form:'SILK', m_root:'silk', number:sing),1,...),
      edge(1, 2, list_np(s_form:'SILK', m_root:'silk', ne_tag:person,
                        ne_type:person_last),2,...),
      edge(2, 3, md(s_form:'skulle', m_root:'ska'),1,...),
      edge(3, 4, md(s_form:'vilja', m_root:'vilja'),1,...),
      edge(4, 5, v(s_form:'använda', m_root:'använda', tense:none,
                  voice:active,vform:base),1,...),
      edge(5, 6, adj(s_form:'lättare',m_root:'lätt',degree:_),1,...),
      edge(6, 7, n(s_form:'droger',m_root:'drog',number:plural),1,...),
      edge(7, 8, period(s_form:'.', m_root:'.'),1,...)]
      next_edge_number:8).
```

```
syntax 0 37 s
      syntax 0 4 np
            syntax 0 4 basic_np
      syntax 5 37 vp
            syntax 5 24 vpcore
                  syntax 5 24 modal_vpcore
                        syntax 5 11 md
                                syntax 12 17 md
                                      syntax 18 24 nonmodal_vpcore1
                                            syntax 18 24 vpcore1
                                                  syntax 18 24 v
      syntax 29 37 np
            syntax 29 31 adj
            syntax 32 37 n
```

### 4.9.2 QLF

The best parse selected during parsing is returned in a predicate-argument representation or quasi-logical form, QLF, for each sentence processed. The QLF produced marks the point where a common representation begins to emerge between the different AVENTINUS languages, Gaizauskas *et al.* (1997). The QLF is just conjunctions of first order logical terms. Some of the

QLF predicates are the unary `city` and `organization`, the binary and language independent `time` and `name`, and the relational binary `logical subject` and `logical object`, Azzam *et al.* (1997). All NP's and VP's lead to the introduction of a unique instance constant in the semantics which serves as an identifier for the object or event referred to in the text.

Consider for instance the semantic/QLF representation of the sentence:

"SILK skulle vilja använda lättare droger."

name 0 4 person e5

semantics 0 37

```
[name(e5,'SILK'), ne_tag(e5,offsets(0,4)), person(e5),
realisation(e5,offsets(0,4)), använda(e4), modal(e4,skulle,vilja),
time(e4,present), aspect(e4,simple), voice(e4,active), lobj(e4,e6),
drog(e6), number(e6,plural), head(e6,droger), adj(e6,lätt),
realisation(e6,offsets(29,37)), realisation(e4,offsets(5,37)), lsubj(e4,e5)]
```

#### 4.10 Coreference Resolution

The coreference resolution module is not trying to recognize new proper names but adds identity relations between those found by the parser. Some of the rules are: i) if one of the tokens of one name match a name in a text, then they are identical: "Barclays Bank/banken, Natklubben 'Sports'/klubb/klubben"; ii) Aliases are considered equal: "Ove Lennart Andersson alias 'Armar och Ben'".

#### 4.11 Discourse Interpretation

The DISCOURSE INTERPRETER, (DI), module translates the QLF representation produced by the parser into a semantic net representation of instances, their AVENTINUS specific ontological classes and their attributes. The AVENTINUS domain model is using, at the moment, the XI knowledge representation language, Gaizauskas *et al.* 1996a. The XI is written in Prolog. The principal task for the discourse processing module is to integrate the semantic representations of multiple sentences into a single model of the text from which the information required for filling a template may be derived.

This is the only module not been implemented yet for Swedish in any form, i.e. prototypical or fully operational. This depends on the fact that the AVENTINUS domain model or ontology is, at this moment (November 1997), on the last phases of its integration into GATE.

### 5. Template Generation

The next step of the monolingual IE task is to search the discourse model for all the instances of objects and their (inherited) attributes, which are formatted according the AVENTINUS user specifications, e.g. Interpol forms. This phase is not implemented yet, since it is dependent on the DI. The accuracy required for the template element production in f-measure, i.e. minimum precision and recall is 65%.

### 6. Conclusion and Further Development

This paper has described the ongoing work for building an information extraction system for Swedish, within the framework of the LE project AVENTINUS. Swedish lingware components have been presented and discussed. The deadline for accomplishing the task is the summer '98, and there is still a lot of work needed in order to meet the AVENTINUS goals stated in the specifications. Nevertheless, there is a strong belief that the time framework allows us for a

successful accomplishment of the task. By mid-summer '98 the system will be operational into unix and NT environments.

## Acknowledgment

Many thanks to the Sheffield IE team for all the support regarding the installation of GATE.

## References

Azzam S., Cunningham H., Wilks Y., Humphreys K. and Gaizauskas R., (1997), *The AVENTINUS Multilingual QLF Interface*, AVENTINUS internal technical report, Univ. of Sheffield.

Brill E., (1994), *Some Advances In Rule-Based Part of Speech Tagging*, In Proc. of the 12th AAAI '94, Seattle Wa.

Cooke G., (1996), *The SemanTag Project*, <http://www.rt66.com/gcooke/SemanTag/>. Site visited 07/11/97.

Cunningham H., Gaizauskas R., Wilks Y. (1995) *A General Architecture for Text Engineering (GATE) - A New Approach to Language Engineering R&D*, Techn. rep. CS - 95 - 21, Univ. of Sheffield, Dept of Computer Science, <http://www.dcs.shef.ac.uk/research/groups/nlp/gate/>. Site visited 06/11/97.

Cunningham H., Azzam S. and Wilks Y., (1996), *Domain Modelling for AVENTINUS, (WP4.2)*, AVENTINUS internal technical report, Univ. of Sheffield.

Cunningham H., Freeman M., Black W.J. (1997) *Software reuse, object-oriented frameworks and NLP*, In *New Methods in Language Processing*, Jones D. and Somers H. (eds), *Studies in Computational Linguistics*, UCL Press, pp. 357-366.

Dunning T., (1994), *Statistical Identification of Language*, CRL, New Mexico State University.

Ejerhed E., Källgren G., Wennstedt G. and Åström M., (1992), *The Linguistic Annotation of the Stockholm-Umeå Corpus project*, Technical Report No. 33, Univ. of Umeå.

Gaizauskas R., Wakao T., Humphreys K., Cunningham H., and Wilks Y., (1995) *Description of the LaSIE System as used for MUC-6*, In Proc. of the Sixth Message Understanding Conference (MUC-6), Morgan Kaufmann, 1995, pp. 207-220, <http://www.dcs.shef.ac.uk/research/groups/nlp/funded/lasie.html>. Site visited 11/11/97.

Gaizauskas R. and Humphreys K., (1996a), *XI: A Simple Prolog-based Language for Cross-Classification and Inheritance*, In *Proceedings of the 7th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA96)*, Sozopol, Bulgaria, pp. 86-95.

Gaizauskas R., Cunningham H., Wilks W., Rodgers P., and Humphreys K., (1996b) *GATE: An Environment to Support Research and Development in Natural Language Engineering*, In

Proceedings of the 8<sup>th</sup> IEEE International Conference on Tools with Artificial Intelligence, Toulouse, France. :

Gaizauskas R., Humphreys K., Azzam S., Wilks Y., (1997), *Concepticons vs Lexicons: An Architecture for Multilingual Information Extraction*, AVENTINUS internal technical report, Univ. of Sheffield *GEOnet Names Server*, <http://www.nima.html/gns/html>. Site visited 26/10/97.

Grishman R., (1996), *TIPSTER Text Phase II Architecture Design*, Version 2.3, New York University.

Holmén S., (1997), *Pundartugg. Narkotikarelaterade slanguttryck*, Polishögskolan, Solna.

Johansson-Kokkinakis S., Kokkinakis D., (1996), *Rule-Based Tagging in Språkbanken*, Research Reports from the Department of Swedish, Göteborg University, GU-ISS-96-5.

Kokkinakis D., (1997a), *Linguistic Toolset for Swedish: Tokenisation, Tagging and Lemmatization, (WP4.3)*, AVENTINUS System Specifications, Vol. 2.

Kokkinakis D., (1997b), *Partial Parsing using a mini-lexicalized CFG, as seed to a bottom-up chart parser*, unpublished manuscript.

Kokkinakis D., Johansson-Kokkinakis S., (1997), *A Robust, Modularized Lemmatizer/Tagger for Swedish Based on Large Lexical Resources*, Research Reports from the Department of Swedish, Göteborg University, GU-ISS-97-1.

Lindfors Viklund M., (1997), *Drug Terms*, Dep. of Swedish, Göteborg Univeristy.

SNPF (Svenska Narkotikapolisföreningen), (1996), *Basfakta om narkotika*, Göteborg, Sweden.

*TIPSTER*, <http://cs.nyu.edu/cs/faculty/grishman/tipster.html>, and <http://www.tipster.org>, Sites visited 19/10/97.

# Norwegian Computational Lexicon (*NorKompLeks*)

Torbjørn Nordgård

Dept. of Linguistics  
Norwegian University of Science and Technology (NTNU)  
Trondheim  
Norway

## Background

There is no generally accessible lexicon for the Norwegian language. This situation prevents development of general language engineering applications for this language. In addition, linguists cannot use modern lexical tools in their research. Given this background, the lexicon project «Norsk komputasjonelt leksikon» (Norwegian Computational Lexicon), abbreviated as *NorKompLeks*, has the following goals:

1. Create a morphological lexicon for the written standard Bokmål
2. Create a morphological lexicon for the written standard Nynorsk
3. Provide phonological descriptions for both standards
4. Describe argument structures for relevant lexical items (verbs, prepositions, certain nouns)

«Bokmålsordboka» (65000 items) and «Nynorskordboka» (90000 items), both owned by Section of Lexicography at the University of Oslo, define the coverage of the two lexicons. These dictionaries contain the «official» lexicographic inventory of Norwegian, as defined by Norsk Språkråd (The Norwegian Language Council). In what follows the results of the project will be briefly described, and special attention is given to argument structure descriptions and how lexical descriptions can be converted into linguistic formalisms.

The project «Norsk komputasjonelt leksikon» (Norwegian Computational Lexicon) is funded by the Norwegian Research Council (NFR) and Telenor (The National Telematic Company).<sup>1</sup> The project period is from January 1996 through December 1998. The partners are

- Linguistics Department, NTNU, Trondheim
- «Dokumentasjonsprosjektet» (a national documentation project within the humanities), University of Oslo
- Computing Centre for the Humanities, University of Bergen
- Telenor

---

<sup>1</sup> Project assistants: Jardar Eggesbø Abrahamsen (Bokmål and Nynorsk morphology), Bodil Aurstad (Bokmål morphology and argument structure), Kristin Eide (Nynorsk morphology), Bente Moxness (Phonology), Eli Sætherø (Argument structure).

## Morphology

The morphology of the written standard Bokmål (graphemic base forms and morphological paradigms) was finished in December 1996. The format of the lexicographic entries in Bokmålsordboka is

NB00 <i>n</i>	Base Form
NB00 <i>na</i>	Inflectional paradigm(s)
NB00 <i>nb</i>	Version number (in case of homonymy)
ARTNR	Identification label (an integer)
TR007	Lexicographic information (etymology, definitions, examples, etc.)

An example from the Bokmål source (irrelevant typographical codes included, i.e. \$C, \$B, @, etc.):

```
NB001 fremme
NB001a v24,v24a,v25
NB001b 2
ARTNR 17805
TR007
..OPP #>$Cll fremme@ v1
..ETY (norr $Bfremja@, av $Bfram@)
..DEF $C1@ hjelpe fram, øke, påskynde, stimulere
..UTR $Bf- en sak, et formål !@
..UTR $Btiltak som f-r kommunens økonomi !@
..UTR $Bf- salget av, interessen for !@
..DEF adj i pr pt:
..UTR $Bvirke f-nde på@
..FOR stimulerende
..DEF $C2@ legge fram til behandling, ta opp, reise
..UTR $Bf- et forslag !@
..UTR $Bf- en sak for Stortinget, retten@
..DEF $C3@ sette i verk, gjennomføre
```

In the more compact computational lexicon the morphological information is described, in Prolog, as

```
bm(fremme,[v24,v24a,v25],17805).
```

Lexicographic information is removed, but can be obtained via the numeric identifier *17805*. *v24,v24a* and *v25* are morphological inflection codes which describe inflection patterns for the lexical item *fremme* («propose»). After expansion by these codes, we get the following paradigm, in accordance with Faarlund, Lie & Vannebo (1996):

```
nkl_ff(frem,[verb,imperativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremma,[adj,mfn,pos,best,pl,17805,fremme]).
nkl_ff(fremma,[adj,mfn,pos,best,sg,17805,fremme]).
nkl_ff(fremma,[adj,mfn,pos,ubest,pl,17805,fremme]).
nkl_ff(fremma,[adj,mfn,pos,ubest,sg,17805,fremme]).
nkl_ff(fremma,[verb,perf_part,indikativ,aktiv,hovedverb,17805,fremme]).
```

```

nkl_ff(fremma,[verb,perf_part,indikativ,passiv,hovedverb,17805,fremme]).
nkl_ff(fremma,[verb,pret,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremme,[verb,infinitiv,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremmede,[adj,mfn,pos,best,pl,17805,fremme]).
nkl_ff(fremmede,[adj,mfn,pos,best,sg,17805,fremme]).
nkl_ff(fremmede,[adj,mfn,pos,ubest,pl,17805,fremme]).
nkl_ff(fremmende,[adj,mfn,pos,best,pl,17805,fremme]).
nkl_ff(fremmende,[adj,mfn,pos,best,sg,17805,fremme]).
nkl_ff(fremmende,[adj,mfn,pos,ubest,pl,17805,fremme]).
nkl_ff(fremmende,[adj,mfn,pos,ubest,sg,17805,fremme]).
nkl_ff(fremmende,[verb,pres_part,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremmer,[verb,presens,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremmes,[verb,presens,indikativ,passiv,hovedverb,17805,fremme]).
nkl_ff(fremmet,[adj,mfn,pos,ubest,sg,17805,fremme]).
nkl_ff(fremmet,[verb,perf_part,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremmet,[verb,perf_part,indikativ,passiv,hovedverb,17805,fremme]).
nkl_ff(fremmet,[verb,pret,indikativ,aktiv,hovedverb,17805,fremme]).
nkl_ff(fremmete,[adj,mfn,pos,best,pl,17805,fremme]).
nkl_ff(fremmete,[adj,mfn,pos,best,sg,17805,fremme]).
nkl_ff(fremmete,[adj,mfn,pos,ubest,pl,17805,fremme]).

```

(best=definite, ubest=indefinite,mfn=male and female, hovedverb= main verb, perf\_part=past participle, pos=positive, pres=present tense, pret=past tense, pres\_part=present participle)

This expansion is designed so that it meets the requirements of a tagger being developed at the University of Oslo. The expansion can be defined differently, for instance with word class information only (the details of this process will be explained below):

```

nkl_ff(frem,[verb]).
nkl_ff(fremma,[adj]).
nkl_ff(fremma,[verb]).
nkl_ff(fremme,[verb]).
nkl_ff(fremmede,[adj]).
nkl_ff(fremmende,[adj]).
nkl_ff(fremmende,[verb]).
nkl_ff(fremmer,[verb]).
nkl_ff(fremmes,[verb]).
nkl_ff(fremmet,[adj]).
nkl_ff(fremmet,[verb]).
nkl_ff(fremmete,[adj]).

```

Full form expansion is controlled by the information connected to the inflection codes. Consider the code v24 (from the set of paradigm codes for the verb *fremme*):

```

code(v24,
    [inf_v:>0,           % fremme      (infinitive)
    imp_v:>[1:0,1:0],    % frem      (imperative)
    pres_v:>"r",         % fremmer   (present)
    pret_v:>"t",        % fremmet   (past)
    p_part_v:>"t",      % fremmet   (past participle)
    pr_part_v:>"nde",   % fremmende (present participle)
    pass_part_v:>"t",   % fremmet   (passive participle form)

```



```

s_pass_v:>"s",           % fremmes      («s-passive»)
pos_ub_sg_v:>"t",       % fremmet      (adjective,positive,indefinite,singular)
pos_b_sg_v:>"de",       % fremmede     (adjective,positive,definite,singular)
pos_pl_v:>"de",         % fremmede     (adjective,positive,plural)
pos_v:>"nde"]].         % fremmende   (adjective,positive)

```

The expression *inf\_v:>0* means that the *inf\_v* (the infinitival form) is achieved if nothing is done to the lexical stem. The imperative is generated by deleting the two rightmost characters of the stem, the present tense requires that the string «r» is added to the stem, and so on. Thus, each generated form is tied to a morphosyntactic code (*inf\_v*, *imp\_v*, ...). These codes must be given an interpretation according to the requirements of the lexicographer, linguist or system designer who is going to use the lexicon. The symbol *pres\_v* has the interpretation verb, present tense, indicative, active, main verb. In Prolog:

```
m_kode(pres_v,[verb,presens,indikativ,aktiv,hovedverb]).
```

Note that the list [verb,presens,indikativ,aktiv,hovedverb] is a sublist of [verb,presens,indikativ,aktiv,hovedverb,17805,fremme], i.e. the morphological information associated with the present form *fremmer* in the example above. By changing the interpretation of the code *pres\_v* new lexical descriptions can be generated, for instance in accordance with TEI or EAGLES specifications. The same is true of all morphological codes in the lexicon. Thus, the codes is a method for describing the properties of morphological distinctions in Norwegian, and the descriptions can be adapted to various theories and formalisms, simply by changing the interpretation of the codes. A trivial example is the alternative where word class information is the only information, as in the example above:

```
m_kode(pres_v,[verb]).
```

The same coding system will be used for the Nynorsk material, and this work will be finished in the spring of 1998.<sup>2</sup>

## Phonology

The project has completed phonological descriptions of approx. 65.000 Bokmål entries. The descriptions are written in SAMPA. Consider some examples (*begynne* = begin):

```

fremme      ""frem@      V01  17805
begynne     b@"jyn@     v21  4974 .

```

Observe that stress and tones are marked: A single " means stress with toneme 1, and double "" signals stress with toneme 2.<sup>3</sup>

---

<sup>2</sup> The derivation of «verbal adjectives» is different in the Nynorsk lexicon where a verb licenses an adjective base form together with an adjective code. This pair serves as the basis for generating a normal adjective paradigm. The description of verbal adjectives in the Bokmål material will be changed to this format when time and resources are available, hopefully in the fall 1998.

The phonological descriptions are based on the most common pronunciation patterns in the South Eastern parts of Norway, where most Norwegians live. In controlling the phonological representations the developers use an automatic speech generation system developed by Telenor Research.

As in the morphology phonological paradigms will be generated, using the same description format. Thus, when the project is finished phonological representations will be available together with morphological forms, both for the base entries in the lexicon and in the expanded («full form») lexicon.

Phonological coding of nynorsk entries has started and will be finished in 1998. Generation and control of phonological paradigms will be accomplished in 1998.

### Argument Structure in NorKompLeks

There is a well-known tension between linguistic sophistication in lexical descriptions and the time available for developing a lexicon of an adequate size. An important question in *NorKompLeks* is how to describe argument structure in a linguistically sound way, and at the same time being able to make these descriptions in a finite amount of time, i.e. 20 months work for one person.

It is very important for the reusability of a lexicon that theoretical adaptation can be achieved without rebuilding the entire lexicon. Consequently, the lexicon should be theory neutral (without being anti-theoretically). The project builds on previous lexicons for Norwegian: the *TROLL* lexicon<sup>4</sup> and *NorLex* verb lists (University of Bergen, 1992 - 1993).

Absolute theory independence is impossible (and undesirable). But theoretical «idiosyncrasies» and «hang-ups» must be avoided, for instance linguistic theories which insist that reference to grammatical functions is irrelevant in the lexicon, as in Government and Binding and Minimalist approaches, see e.g. Chomsky (1981, 1986) and Chomsky (1994). A criterion for success is that the lexicon can be used to generate argument descriptions in various theories, for instance Lexical-Functional Grammar (see e.g. Dalrymple, Kaplan, Maxwell & Zaenen 1995), Head-Driven Phrase Structure Grammar (see Pollard and Sag 1994) and GB-type argument descriptions (see the references above), by some limited adaptations.

The argument structure for a verb must contain information about the basic *construction types* that the verb can be engaged in. This information can be encoded at various levels of abstraction, and linguistic theories will typically tend to represent this information in a compact format which can be interpreted by theory internal mechanisms. A computational lexicon with reusability ambitions should take the opposite approach: Describe the construction types transparently, and show how these construction types can be interpreted by linguistic theories.

---

<sup>3</sup> The code V01 is an original morphological code from Bokmålsordboka. This code will be replaced by new morphological codes (v24,v24a,v25) and phonological codes (work in progress).

<sup>4</sup> See Hellan and Johnsen 1988.

Consider some intransitive construction types with examples:

- Intransitive verb with expletive subject  
*Det regner (It rains)*
- Intransitive verb with expletive subject and required adverbial  
*Det kvakk i henne («It suddenly-surprised her», 'she was suddenly surprised by something')*
- Intransitive verb with agentive subject  
*Studentene tenker (The students think)*
- Intransitive verb with experiencer subject  
*Gutten fryser (The boy is freezing)*

A construction type is characterised by its obligatory arguments. Each construction type is given a unique label. Thus, the construction type «Intransitive verb with expletive subject» is called *nullv* (meaning *nullverdig* or «zero valency»). The arguments of a construction type is described as a triple with information about syntactic function, thematic role and categorial realisation. The code *nullv* is interpreted as

```
arg_code(nullv,[arg1:su::norole::np]).
```

This simply means that a verb with this code can take a subject NP with no thematic role, i.e. an expletive subject. The code for intransitive verb with agentive subject is

```
arg_code(intrans1,[arg1:su::ag::np]).
```

That is, a construction with a subject NP which has the thematic role agent.

Passivization is a regular process in most languages, but certain verbs can not enter the passive construction. A well-known example in the syntactic literature is so-called ergative verbs, like *arrive*. In NorKompLeks passivization possibilities are tied to construction types. Therefore, intransitive verbs of the «arrive»-type has the label *intrans2* which is composed as

```
arg_code(intrans2,[arg1:su::th::np,--passiv]).
```

That is, a verb with a thematic subject. The tag «--passiv» signals that this construction type can't undergo a lexical passivization process, whereas the type *intrans1* does not have this restriction. The tag «--passive» is in a sense redundant, given that it is empirically true that verbs which take a thematic subject cannot be passivized, but this tag makes the codes more explicit and thus reduces the risk of wrong code assignments.

The passivization procedure can be defined as

*If a verb has the code intrans1, it also has the possible realisation [arg1:su::norole::np] when it appears with passive morphology.*

Consider two codes for transitive constructions:

- `arg_code(trans1,[arg1:su::ag::np,arg2:obj::th::np]).`  
Example: *dekomponere* (*decompose*)
- `arg_code(trans2,[arg1:su::ag::np,arg2:obj::th::s1]).`  
Example: *dokumentere* (*to document*, with clausal complement)

Note that each argument in the construction is represented as a triple, but the argument structure is conceived as a *list* of such triples.<sup>5</sup>

Control verbs require that the number of arguments is specified together with the controller of the infinitival subject. This is done as follows in *NorKompLeks*:

- `arg_code(trans3,[arg1:su::ag::np,arg2:obj::th::inf, arg2:su=arg1:su]).`  
Example: *prøve* (*try*)

The expression *arg2:su=arg1:su* says that the subject of the second argument is identical to the subject of the first argument, i.e. subject control.

Let us now turn to some examples from the argument structure descriptions:

```
w(dages,9088,[nullv]).           (dawning, «The day is dawning»)
w(dampe,9220,[trans1,intrans2]). (steam)
w(dandere,9237,[trans1]).        (shape, arrange)
w(dangle,9242,[intrans2,trans1]).(dangle, swing)
w(danse,9265,[intrans1,trans1]).(dance)
w(danske,9279,[intrans1]).       (speak Danish affectedly)
w(dirre,9285,[intrans2]).        (quiver, vibrate)
```

Note that a verb can enter into a set of construction types, as the verb *danse* (dance) which can be used transitively or as a standard intransitive verb.

## Interpretation of Argument Structures in Grammatical Frameworks

The argument structure information in *NorKompLeks* satisfies the basic requirements of current generative theories. «Classical» LFG, i.e. the 1982 version, needs information about syntactic function in order to specify the argument structure portion of the lexical descriptions. More recent versions of LFG has argument linking information where syntactic functions and thematic roles are connected. *NorKompLeks* seems to have the required information for both versions of LFG. A preliminary version of a compiler which translates argument descriptions in *NorKompLeks* into LFG-82 descriptions has been developed. The Prolog session below illustrates the idea of translating *NorKompLeks* descriptions of verbs into LFG-82 format (the information is given in Prolog syntax). The program consults the lexical descriptions of a word form (e.g. *babel*), translates the code(s) into LFG-82 format, and returns the translation. The usual LFG notation is also given.

---

<sup>5</sup> Such a list is of special interest for subcategorization in HPSG.

27 ?- translate\_word(bable,Trans,lfg82).     *bable = babble*  
Trans = [[up pred = bable(up subj)]]

LFG notation: ( $\uparrow$  PRED) = 'bable<( $\uparrow$  SUBJ)>'

28 ?- translate\_word(baktale,Trans,lfg82).     *baktale = slander, speak ill of*  
Trans = [[up pred = baktale(up subj, up obj)]]

LFG notation: ( $\uparrow$  PRED) = 'baktale<( $\uparrow$  SUBJ), ( $\uparrow$  OBJ)>'

29 ?- translate\_word(boble,Trans,lfg82).     *boble = bubble*  
Trans = [[up pred = boble(up subj)], [up pred = boble(up subj)]]

LFG notation: ( $\uparrow$  PRED) = 'boble<( $\uparrow$  SUBJ)>'

30 ?- translate\_word(beta,Trans,lfg82).     *beta = impress, fascinate*  
Trans = [[up pred = beta(up subj, up obj)]]

LFG notation: ( $\uparrow$  PRED) = 'beta<( $\uparrow$  SUBJ), ( $\uparrow$  OBJ)>'

31 ?-translate\_word(begynne,Trans,lfg82).     *begynne = begin*  
Trans = [[up pred = begynne(up subj)], [up pred = begynne(up subj, up obj)],  
[up pred = begynne(up subj, up vcomp), up subj = (up vcomp, subj)]]

LFG notation:  $\left[ (\uparrow \text{ PRED}) = \text{'begynne}<(\uparrow \text{ SUBJ})>\right]$   
 $\left[ (\uparrow \text{ PRED}) = \text{'begynne}<(\uparrow \text{ SUBJ}), (\uparrow \text{ OBJ})>\right]$   
 $\left[ (\uparrow \text{ PRED}) = \text{'begynne}<(\uparrow \text{ SUBJ}), (\uparrow \text{ VCOMP})>\right]$   
 $\left[ (\uparrow \text{ SUBJ}) = (\uparrow \text{ VCOMP SUBJ}) \right]$

The last version of the verb *begynne* (begin) has a control equation which states that the matrix subject is the same as the subject of the vcomp. Observe that translation 29 in the Prolog session gives two identical argument structures, which means that the lexicon makes a distinction that this LFG compiler doesn't.

Lexical descriptions in HPSG need information about categorial realisation and ordering among the arguments. The relevant ordering is implicitly encoded (arg1 is the subject, arg2 is the object, etc.). The semantics of lexical items HPSG is not provided by *NorKompLeks*, simply because this semantic information is more or less internal to HPSG.

Government and Binding approaches to syntax is not widespread in computational linguistics, but GB grammars (and Minimalist approaches) are popular among syntacticians. Because the *NorKompLeks* lexicon is meant to be useful also for syntacticians who are not NLP practitioners, GB translations of lexical descriptions has some interest. Lexical descriptions in GB require information about thematic roles, categorial information and whether an argument is «external»

or «internal». The former two types of information is encoded directly, but the «external» / «internal» dichotomy is present indirectly. Agentive subjects are «external», but thematic subjects are «internal». Thus, the code *intrans1* has one external argument in GB terms, and *intrans2* has one internal argument. Some examples of the translation program:

```
22 ?- translate_word(bable,Trans,gb).
    Trans = [bable lex_gb verb args [ag : np - 'E']]
```

The expression *bable lex\_gb verb args [ag : np - 'E']* means that *bable* is a verb with one external agent argument which is realized as NP.

```
23 ?- translate_word(baktale,Trans,gb).
    Trans = [baktale lex_gb verb args [ag : np - 'E', th : np]]
```

Here the verb *baktale* has two arguments, an external agent NP and an internal thematic NP.

```
24 ?- translate_word(boble,Trans,gb).
    Trans = [boble lex_gb verb args [], boble lex_gb verb args [th : np]]
```

The empty list signals a verb with no theta-marked arguments (the first version of *boble*). The second version of the verb has one argument which is internal, which is the theoretical description of an ergative verb.

The verb *beta* has a thematic external argument and an internal argument with the role experiencer:

```
25 ?- translate_word(beta,Trans,gb).
    Trans = [beta lex_gb verb args [th : np - 'E', exp : np]]
```

The actual translation from NorKompLeks descriptions to GB descriptions is governed by Prolog predicates like

```
arg_translation(gb,[arg1 : su :: ag :: np], [ag : np - 'E']).
```

and

```
arg_translation(gb,[arg1 : su :: ROLE :: np, -- passiv], [ROLE : np]).
```

The third arguments, i.e. *[ag : np - 'E']* and *[ROLE : np]* are GB-translations of NorKompLeks codes. Recall that *[arg1 : su :: ag :: np]* is the interpretation of the code *intrans1*, and the verb *bable* has this code. *ROLE* is a Prolog variable which has the effect of transferring the NorKompLeks theta role onto the GB description. LFG translations is accomplished by the same method, but with other interpretations, of course:

```
arg_translation(lfg,[arg1:su::_ARG1ROLE::np], up pred = [up subj]).
```

Note that the thematic role is of no interest here, and it is made invisible by the anonymous Prolog variable *\_ARG1ROLE*.

## Argument Information and Morphological Information

The verb

w(debutere,9434,[intrans1]). (*make one's debut*)

has an identifier 9434 which is a «pointer» into the stem lexicon. The stem lexicon has information about morphological properties (conjugation classes and spelling). Generating full forms with morphosyntactic (see the morphological section above) and argument structure information is a rather trivial matter, but the details are of course dependent upon theoretical and formal considerations.

## Conclusions

The basic ingredients of the morphological, phonological and syntactic-semantic properties of the *NorKompLeks* lexicon has been described. Most of the work is finished. Today the lexicon is used at the University of Oslo in a rule-based POS-tagger, and it will be used in the Norwegian part of the ongoing EU-project SCARRIE. TEI interpretations of the morphological and syntactic-semantic information will be made later this year.

## References

- Hellan, L., Johnsen, L. and Pitz, A. : *TROLL* (The Trondheim Linguistic Lexicon Project). Ms, Dept. of Linguistics, NTNU, Trondheim.
- Bresnan, J., ed. (1982): *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, Mass.
- Bresnan, J., and R. Kaplan (1982): Introduction. In Bresnan (1982).
- Chomsky, N. (1981): *Lectures on Government and Binding*. Foris, Dordrecht.
- Chomsky, N. (1986): *Knowledge of Language. Its Nature, Origin and Use*. Praeger, New York.
- Chomsky, N. (1995): *The Minimalist Program*. MIT Press, Cambridge, Mass.
- Dalrymple, M., R.M. Kaplan, J.T. Maxwell, A. Zaenen (1995): *Formal Issues in Lexical-Functional Grammar*. CSLI Publications, Stanford, CA.
- Faarlund, J.T., S. Lie and K.I. Vannebo (1997): *Norsk Referansegrammatikk*. Universitetsforlaget, Oslo.
- Pollard, C. and I. Sag: *Head-Driven Phrase Structure Grammar*. The University of Chicago Press, Chicago.

# Structural Lexical Heuristics in the Automatic Analysis of Portuguese

Eckhard Bick

Department of Linguistics, Århus University, Willemoesgade 15 D, DK-8200 Århus N  
tel: +45 - 8942 2131, fax: +45 - 86 281397, e-mail: lineb@hum.aau.dk  
<http://visl.hum.ou.dk/Linguistics.html>

## Abstract

The paper discusses, on the lexical level, the integration of heuristic solutions into a lexicon based and rule governed system for the automatic analysis of unrestricted Portuguese text. In particular, a morphology based analytic approach to lexical heuristics is presented and evaluated. The tagger involved uses a 50.000 entry base form lexicon as well as prefix-, suffix- and inflexion endings lexica to assign part of speech and other morphological tags to every wordform in the text, with recall rates between 99.6% and 99.7%. Multiple readings are subsequently disambiguated by using grammatical rules formulated in the Constraint Grammar formalism. On the next level of analysis, tags for syntactical form and function alternatives are mapped onto the wordforms and disambiguated in a similar way. In spite of using a highly differentiated tag set, the parser yields correctness rates - on running unrestricted and unknown text - of over 99% for morphology/PoS and 97-98% for syntax. A test site with a variety of applications (parsing, corpus searches, interactive grammar teaching and - experimental - MT has been established at <http://visl.hum.ou.dk/Linguistics.html>.

## 1 Background

In corpus linguistics, most systems of automatic analysis can be classified by measuring them against the bipolarity of rule based versus probabilistic approaches. Thus Karlsson (1995) distinguishes between “pure” rule based or probabilistic systems, hybrid systems and compound systems, i.e. rule based systems supplemented with probabilistic modules, or probabilistic systems with rule based “bias” or postprocessing. As a second parameter, lexicon dependency might be added, since both rules based and probabilistic systems differ internally as to how much use they make of extensive lexica, both in terms of lexical coverage and granularity of lexical information.

The *constraint grammar* (CG) formalism (e.g. Karlsson et al., 1995), which I have been using in my own system<sup>1</sup> for the automatic analysis of unrestricted Portuguese text (Bick, 1996 [1] and 1997 [2]), is both rule governed and lexicon based, focusing on disambiguation of multiply

---

<sup>1</sup> The system was developed in the framework of a Ph.D.-project at Århus University, over a period of three years, drawing on lexicographic research on Portuguese from an earlier Master's Thesis.



assigned lexical and structural readings as the main tool of analysis. Readings are expressed as sets of word based modular tags. Syntactic structure is covered by using function tags and dependency markers (Bick, 1997 [1]), but I will here concentrate on the lexico-morphological level. Before any Constraint Grammar rules can apply, all (morphologically) *possible* readings have to be identified, and I have to this end developed a preprocessor, that identifies wordforms, polylexical units and sentence boundaries, as well as a morphological analyser for Portuguese using an adapted electronic version of a previously published dictionary (Bick, 1993) in combination with affix- and inflection endings lexica supplemented by corresponding alternation rules for word formation (Bick, 1995). In the analyser's output, every word form is followed by as many tag lines as there are potential readings:

- (1) "<revista>"  
 "revista" <+n> <CP> <rr> N F S  
 "revestir" <vt> <de^vtp> <de^vrp> V PR 1/3S SUBJ VFIN  
 "revistar" <vt> V IMP 2S VFIN  
 "revistar" <vt> V PR 3S IND VFIN  
 "rever" <vt> <vi> V PC

With a CG-term, such an ambiguous list of readings is called a *cohort*. In the example, the word form 'revista' has one noun-reading (female singular) and four (!) verb-readings, the latter covering three different base forms, subjunctive, imperative, indicative present tense and participle readings. Conventionally, PoS and morphological features are regarded as primary tags and coded by capital letters. In addition there can be secondary lexical information about valency and semantical class, marked by <> bracketing.

A *constraint grammar rule* brings the ambiguity problem to the foreground by specifying which reading (out of a cohort of ambiguous readings for a given word) is impossible (and thus to be discarded) or mandatory (and thus to be chosen) in a given sentence-context. For instance, a rule might discard a finite verb reading after a preposition (2a) , or when another - unambiguous - finite verb is already found in the same clause, with no coordinators present (2b).<sup>2</sup>

- (2a) REMOVE (VFIN) IF (-1 PRP)  
 [discard finite verb readings (VFIN) if the first word to the left (-1) is a preposition PRP]
- (2b) REMOVE (VFIN) IF (\*1C VFIN BARRIER CLB OR KC) (NOT \*-1 CLB-WORD)  
 [discard VFIN if there is another unambiguous (C) finite verb (VFIN) anywhere to the right (\*1) with no clause-boundary (CLB) and coordinating conjunction (KC) interfering (BARRIER). Discard only if there is no subordinator (CLB-WORD) anywhere to the left (\*-1)]

---

<sup>2</sup> Ordinarily, this disambiguation process works on whole cohort lines, i.e. distinguishes between PoS, base form and inflection, but tolerates competing valency options. However, on a higher level of analysis, I have introduced valency and semantical disambiguation, too. This can be very useful for polysemy resolution, like in "rever", where the transitive <vt> - intransitive <vi> distinction has a meaning correlate: 'tornar a ver' [see again] vs. 'transudar' [leak through]. Likewise, "revista" followed by a name <+n> or being read (semantical class <rr>) is more likely to be a newspaper than an inspection (semantical class <CP> for action: +CONTROL, +PERFECTIVE).

With current software, before an analysis run, all rules are translated into a finite state network by a compiler program, yielding the actual parser. The Portuguese grammar was originally written in the formalism suggested by Pasi Tapanainen's first compiler implementation, but later rewritten to match the notation used in his new CG-2 parser compiler (Tapanainen, 1996).

By applying the rule set several times, the parser renders more and more words in the sentence unambiguous, and in the end, only one reading is left for every word. Since the individual rule can be made very "cautious" by adding more context conditions, and since the last surviving reading will never be discarded, the formalism is very robust. Even imperfect input will yield *some* parse. Unlike probabilistic systems, where "manual interference" as in the introduction of bias on behalf of irregular phenomena often has an adverse side-effect on the overall performance of the parser (due to interference with the ordinary statistical "rules" based on the *regular* "majority" phenomena), Constraint Grammar tolerates and even encourages the incremental "piecemeal" addition of exceptions and context conditions for individual rules (For a comparison of statistical and constraint-based methods see Chanod & Tapanainen, 1994).

## 2. System Performance

If they can be made to work on free text, rule based systems can achieve very low error rates. While state-of-the-art probabilistic taggers still have error rates of over three percent<sup>3</sup>, even for PoS tagging, CG based systems fare somewhat better. For English word class error rates of under 0.3% have been reported at a disambiguation level of 94-97% (Voutilainen, 1992). For my own Portuguese CG system, test runs with near 100% disambiguation on fiction and news texts suggest a correctness rate of over 99% for morphology and part of speech, when analysing unknown unrestricted text<sup>4</sup>. For syntax the figures are 98% for classical literary prose (Eça de Queiroz, "O tesouro") and 97% for the more inventive "journalese" of news magazine texts (VEJA, 9.12.1992), as shown in table(3):

---

<sup>3</sup> Compare, for English, (Garside et.al., 1987) on the HMM based CLAWS system, (Francis and Kucera, 1992) on recovering PoS tags from the Brown corpus, Ratnaparkhi's maximum-entropy tagger trained on the Penn Treebank (Marcus et al., 1993) or Brill's stochastic tagger using automated learning (Brill, 1992). For German the Morphy system described in (Lezius et. al., 1996) achieved an accuracy of 95.9%.

<sup>4</sup> The test texts used were not part of the benchmark corpus used to develop the rules, and fresh text chunks were used for every new test. The present grammar, however, still being improved, does incorporate changes made as a result of test run errors.

(3) System performance on the PoS and syntactic levels:

Text: Error types:	<i>O tesouro</i> ca. 2500 words		<i>VEJA 1</i> ca. 4800 words		<i>VEJA 2</i> ca. 3140 words	
	errors	correct- ness	errors	correct- ness	errors	correct- ness
Part-of-speech errors	16		15		24	
Base-form & flexion errors	1		2		2	
<b>All morphological errors</b>	17	<b>99.3 %</b>	17	<b>99.7 %</b>	26	<b>99.2 %</b>
syntactic: word & phrases	54		118		101	
syntactic: subclauses	10		11		13	
<b>All syntactic errors</b>	64	<b>97.4 %</b>	129	<b>97.3 %</b>	114	<b>96.4 %</b>
"local" syntactic errors due to PoS/morphological errors	- 27		- 23		- 28	
<b>Purely syntactic errors</b>	37	<b>98.5 %</b>	106	<b>97.8 %</b>	86	<b>97.3 %</b>

### 3. Lexico-morphological heuristics

Yet even in a rule based CG system, heuristics can be quite useful (for English, see Karlsson et al., 1995). Thus rules are usually grouped according to their "safety", i.e. their statistical tendency to make errors. Less safe rules can be added as a heuristic level on top of a kernel of safe rules, and will be applied *after* these. Also, statistical inspired "rarity tags" (<Rare>) can be added to certain less probable readings in the lexicon, and then referred to by contextual disambiguation rules. A third field for the application of heuristics is on the *analyser level*, i.e. concerns the (lexico-morphological) *input* of the disambiguation rule system. It is this third type of heuristics I am concerned with here.

Since the higher levels of the parsing system (for example, PoS and syntax) are technically rule based disambiguators, they need *some* reading for every word to work on, which is why even unanalyzable word forms (i.e. word forms that can not be reduced to a root found in the analyser's lexicon) need to be given one or more heuristic readings with regard to word class and flexion morphology. The majority of such cases is accounted for by unknown proper nouns (1-2% of all words, depending on text type), and can be handled by assigning heuristic PROP tags to *all* capitalised words in certain contexts, and then *adding* any competing *analytical* analysis, especially in the case of sentence initial position, leaving the final decision to the rule based disambiguation module. This way, though 80% of all nouns in my corpus need heuristical treatment, the error rate for the PROP class as a whole can be kept at 2%, not too far from the taggers overall PoS error rate (< 1%).

### 3.1 "Unanalyzable<sup>5</sup> words": typology and statistics

Though accounting for only 0.3-0.5% of word forms in running text, other types of analysis failures (i.e. word forms that can not be reduced to a root found in the analyser's lexicon) are more difficult to handle, due to their functional diversity and the lack of a clear morphological marker. Table (4) provides an error typology for a 131.981 word literature and secondary literature corpus (The RNP depository of Brazilian literature), containing 604 unanalyzable words in the test run.

Three main groups may be distinguished, comprising of roughly one third of the cases each:

- a) orthographical errors (shaded in the table, and partially corrected before heuristics proper by an accent module recognising regular regional spelling variations)
- b) unknown and underivable Portuguese words or abbreviations
- c) unknown foreign loan words

(4) Error types in "unanalyzable" words:

DOMAIN	NUMBER OF TOKENS	PERCENTAGE	
Foreign	232	38.4	
orthographic variation (European/accenuation)	125	20.7	<i>Correctables</i>
other port. Orthographic	74	12.3	<i>Misspellings</i>
non-capitalised names and abbreviations	37	6.1	<i>Encyclopaedic lexicon failures</i>
names and name roots	18	3.0	
abbreviations	19	3.1	
root not found in lexicon	119	19.7	<i>Core lexicon failures</i>
found in Aurelio <sup>6</sup>	91	15.1	
not found in Aurelio	28	4.6	
derivation/flexion problem	15	2.5	<i>Affix lexicon failures</i>
suffix	8	1.3	
prefix	3	0.5	
flexion ending	2	0.3	
alternation information	2	0.3	
other	2	0.3	
SUM	604	100.0	

### 3.2 Analytical morphological heuristics

For optimal performance, the three groups mentioned above would require different strategies. Foreign words appearing in running Portuguese text are typically nouns or noun phrases, and

<sup>5</sup> In this paper I intend "unanalysable word forms" to mean word forms that cannot - by derivation and/or inflexional analysis - be reduced to a root found in the analyser's lexicon. Of course, only part of these - typing errors and foreign language quotes - are *really* unanalysable, while others might be covered by enlarging the lexicon or enhancing the scientific derivation list.

<sup>6</sup> "Novo Dicionário Aurelio" is the largest monolingual dictionary of Brazilian Portuguese.

trying to identify verbal elements only causes trouble. In "real" Portuguese words without spelling errors, structural clues - like flexion endings and suffixes - should be emphasised. These will be meaningful in misspelled Portuguese words, too, but, in addition, specific rules about letter manipulation (doubling of letters, missing letters, letter inversion, missing blanks etc.) and even knowledge about keyboard characteristics might make a difference.

Motivated by a grammatical perspective rather than probabilistics, my approach has been to emphasise groups (a) and (b) and look for *Portuguese* morphological clues in words with unknown stems. Since prefixes have very little bearing on the probability of a word's word class or flexional categories, only the flexion endings and suffix lexica are used. As it also does in ordinary morphological analysis, the tagger tries to identify a word from the right, i.e. backwards, cutting off potential endings or suffixes and checking for the remaining stem in the root lexicon (the main lexicon). Normally, for (multiply) *analysed* words, using Karlsson's law (Karlsson 1992, 1995)<sup>7</sup>, the Portuguese analyser would try to make the root as long as possible, and to use as few derivational layers<sup>8</sup> as possible. For (system-internally) *unanalyzable* words, however, I use the opposite strategy: Since I am looking for a hypothetical root, flexion endings and suffixes are all I've got, and I try to make their half of the word (the right hand part) as large as possible.

Working with a minimal root length of 3 letters, and calling my hypothetical root 'xxx', I will start by replacing only the first 3 letters of the word in question by 'xxx' and try for an analysis, then I will replace the first 4 letters by 'xxx', and so on, until - if necessary - the whole word is replaced by 'xxx'.<sup>9</sup> For a word like *ontogeneticamente* the rewriting record will yield the chain below. Here, the full chain is given, with all readings it would encounter on its way. In the real case, however, the tagger - preferring long derivations/endings to short ones - would stop searching at the *xxxticamente* -level, where the first group of readings is found. In fact, the adverbial use of an adjectively suffixed word is much more likely than hitting upon, say, a "root-only" noun whose last 9 letters happen to include both the '-ico' and the '-mente' letter chains by chance.

(5) *ontogeneticamente* -> no analysis

*xxxogeneticamente*  
*xxxgeneticamente*  
*xxxeneticamente*  
*xxxneticamente*

---

<sup>7</sup> Karlsson's law states, that of two morphological analyses of different derivational complexity, the one with *fewer* elements is almost always the correct one.

<sup>8</sup> Karlsson's law can be applied to any string of free (i.e. compounding), derivational or inflexional morphemes, but the frequency of ambiguity types with respect to these three elements will differ from language to language - thus, in Portuguese, compounding is much rarer than in most Germanic languages, while Swedish, the language for which Karlsson's law was originally formulated, does have compounding, but not as rich an inflexion morphology.

<sup>9</sup> A similar method of partial morphological recognition and circumstantial categorization might be responsible for a human being's successful inflectional and syntactic treatment of unknown words in a known language; the Portuguese word games "collorido" (president Collor & *colorido* - 'coloured') and "tucanagem" (the party of the *tucanos* & *sacanagem* - 'dirty work'), for instance, will not be understood by a cultural novice in Brazil, even if he is a native speaker of European Portuguese - but he will still be able to identify both as singular, the first as a past participle ('-do') and the second as an abstract noun ('-agem') of the feminine gender.

<i>xxxeticamente</i>	
<i>xxxiticamente</i>	-> suffix '-ico' (variation '-tico') + adverbial ending '-mente' "ontogene" <DERS -ico [ATTR]> <deadj> ADV
<i>xxxicamente</i>	-> suffix '-ico' + adverbial ending '-mente' "ontogene" <DERS -ico [ATTR]> <deadj> ADV
<i>xxxicamente</i>	
<i>xxxamente</i>	-> adverbial ending '-mente' (variation '-amente') "ontogenetico" <xxxo> <deadj> ADV
<i>xxxmente</i>	
<i>xxxente</i>	-> "present participle"-suffix '-ente' "ontogeneticamer" <DERS -ente [PART.PR]> ADJ M/F S "ontogeneticamer" <DERS -ente [AGENT]> N M/F S -> causative suffix '-entar' <sup>10</sup> + verbal flexion ending '-e' "ontogeneticam" <DERS -ar [CAUSE]> V PR 1/3S SUBJ VFIN
<i>xxxnte</i>	
<i>xxxte</i>	
<i>xxxte</i>	-> verbal flexion ending '-e' "ontogeneticamenter" <xxxer> V IMP 2S VFIN "ontogeneticamentir" <xxxir> V IMP 2S VFIN ### "ontogeneticamenter" <xxxer> V PR 3S IND VFIN "ontogeneticamentir" <xxxir> V PR 3S IND VFIN ### "ontogeneticamentar" <xxxar> V PR 1/3S SUBJ VFIN
<i>xxx</i>	-> no derivation or flexion "ontogeneticamente" <xxx> N F S "ontogeneticamente" <xxx> N M S

Roots with 'xxx' are present in the core lexicon alongside the "real" roots, including the necessary stem alternations<sup>11</sup> for verbs (here, BbCc for different root-stressed forms and AaiD for endings-stressed forms):

(6)

root	word class	alternation subclass	lexeme ID	target of analysis
xxx	<sm>		54573	masculine noun, typically foreign
xxxar	<amf>		59547	Portuguese '-ar'-adjective*
xxxar-	<vt>	AaiD	54578	endings-stressed forms of '-ar'-verbs
xxxer	<sm>		54666	masculine noun, typically English*
xxxia	<sf>		54665	feminine noun, Latin-Portuguese*
xxxo	<sm>		54582	masculine noun, typically Portuguese

<sup>10</sup> This suffix is regarded as a variant of '-ar', and therefore normalized in the DER-tag: <DERS -ar [CAUSE]>.

<sup>11</sup> Here, BbCc for different root-stressed forms and AaiD for endings-stressed forms, with D, for example, meaning a root to be used with future subjunctive endings.

Besides the typical stems ending in '-o', '-a' and '-r', default stems consisting of a plain 'xxx' have been entered to accommodate for foreign nouns with "un-Portuguese" spelling. Like many other languages, Portuguese will force its own gender system even onto foreign loan words, so a masculine and a feminine case must be distinguished, for later use in the tagger's disambiguation module.

Since the analyser's heuristics for unknown words prefers readings with endings (or suffixes) to those without, and longer ones to shorter ones, verbal readings (especially those with inflexion morphemes in 'r', 'a' or 'o') have a "natural" advantage over what really should be nouns or adjectives, especially when these appear in their uninflected singular base form. Lexicon-wise, this tendency is countered by adding three of the most commonly ignored nominal cases specifically into the lexicon: (a) English '-er' nouns otherwise only taken as Portuguese infinitives, (b) Latin-Portuguese '-ia' nouns otherwise only read as verbal forms in the imperfeito tense, and (c) '-ar' adjectives otherwise analysed only as infinitives.

Rule-wise, verbal readings alone are not allowed to stop the heuristics-machine, it will proceed until it finds a reading with another word class. So, the process is set to ignore *verbal* readings on its way down the chain of hypothetical word forms with ever shorter suffix/endings-parts. Thus, the heuristics-machine will *record* verbal readings, but only stop if a noun, adjective or adverb reading is found in that level's cohort (list of readings). In this context, participles and gerunds - though verbal - are treated as "adjectives" and "adverbs", respectively, because they feature very characteristic endings ('-ado', '-ido', '-ando', '-endo', '-indo').

This raises the possibility of the heuristics-machine progressing from multi-derived analyses (with one or more suffixes) to simple analyses (without suffixes) before it encounters a non-verbal reading. In this case, the application of Karlsson's law does still make sense, and when the heuristics-machine hands its results over to the local disambiguation module, this will select the readings of lowest derivational complexity, weeding out all (read: verbal!) readings containing more (read: verbal!) suffixes than the group selected. In the misspelled French word '*entaente*', for example, the verbal reading:

(7a) "enta" <DERS -(ent)ar [CAUSE]> V PR 1/3S SUBJ VFIN,  
from the 'xxxente'-level, is removed, leaving only underived verbal readings - from the 'xxx'-level, along with the desired noun singular reading from the 'xxx'-level.

(7b) entaente ALT xxxaente ALT xxxe ALT xxx  
       "entaenter" <xxxer> V IMP 2S VFIN  
       "entaenter" <xxxer> V PR 3S IND VFIN  
       "entaentar" <xxxar> V PR 1/3S SUBJ VFIN  
       "entaente" <xxx> N F S  
       "entaente" <xxx> N M S

Since all disambiguation not related to Karlsson's law is referred to the CG-module, the word class choice between V and N will be contextual (and rule based), as well as the morphological sub-choice of mode (IMP - PR) for the verb, and gender (M - F) for the noun. In the prototypical case of a preceding article, the verb reading is ruled out by:

(8a) REMOVE (V) IF (-1 ART)  
and the gender choice is then taken by agreement rules such as:

- (8b) REMOVE (N M) IF (- 1C DET) (NOT -1 M)  
 REMOVE (N F) IF (- 1C DET) (NOT -1 F)

Consider the following examples of "unanalyzable" words from real corpus sentences, where the final output, after morphological contextual disambiguation, is given:

- (9a) inventimanhas ALT xxxas (also: one ADJ and three rare V-readings)  
 "inventimanha" <xxx> N F P 'tricks'
- (9b) araraquarenses ALT xxxenses (3 other ADJ readings removed by local disambiguation)  
 "araraquar" <DERS -ense [PATR]> <jh> <jn> ADJ M/F P 'from Araraquara'
- (9c) sombrancelhas ALT xxxas (also: one ADJ and three rare V-readings)  
 "sombrancelha" <xxx> N F P '=sobrancelhas - eye brows'
- (9d) cast ALT xxx (also: N F S)  
 "cast" <\*1> <\*2> <xxx> N M S 'English: cast'

In (9a) and (9b) the parser assigns correct readings to unknown, but wellformed Portuguese words. Depending on the orthodoxy of the fusion process, these affixes may be recognised (9b), or not (9a). That affix recognition is important, can be seen from the fact that all competing analyses in (9b) - but not in (9a) - have the correct PoS tag. What is special about (9c), is the (phonetical?) misspelling ('sombrancelhas') of an otherwise ordinary Portuguese word. Even so, with the help of the surviving morphological clues and contextual disambiguation, the parser is able to assign the right analysis in most cases, especially if the words still look Portuguese. (9d), finally, is the hard case - foreign loan words. English 'cast' does not fit with any Portuguese flexion ending, therefore the default reading N is assigned, gender disambiguation relying on NP-context.

In order to test the parser's performance and to identify the strengths and weaknesses of the heuristics strategy of the parser, I have manually inspected 757 "running" instances<sup>12</sup> of lower case word forms where the parser's disambiguation module received its input from the tagger's heuristics module. The first column shows the word class analysis chosen, and inside the three groups (errors, Portuguese, foreign) the left column gives the number of correct analyses, whereas the right column offers statistics about the mistakes, specifying - and quantifying - what the analysis *should* have been.

---

<sup>12</sup> The words comprise all "unanalysable" word forms in my corpus, that begin with the letters 'a' and 'b'. Since the relative distribution of foreign loan words and Portuguese words depends on which initial letters one works on ('a', for one, is over-representative of Portuguese words, whereas 'x', 'w' and 'y' are English-only domains), no conclusions can be drawn about these two groups' relative percentages. Inside the Portuguese group, however, the distribution between real words and misspellings may be assumed to be fairly alphabet-independent. Any way, the sampling technique has no significance for error frequencies or distribution in relation to word class, which was the main objective in this case.



(10) Word class distribution and parser performance in "unanalyzable" words (VEJA news text)

analysis	A) orthographical errors		B) Portuguese words		C) foreign words <sup>13</sup>		all	
	correct	other	correct	other	correct	other	correct	other
N	119	ADJ 8 ADV 8 VFIN 3 PRON 1 DET 1 PRP 1	212	ADJ 3	226	ADV 11 ADJ 3 PRON 2 PRP 2	557	43
ADJ	25	N 8 GER 2	95	N 7	8	-	128	17
ADV	3	-	5	-	-	-	8	-
VFIN	13	N 4 PCP 1 ADV 1	9	N 4 ADJ 2	-	N 7 ADJ 1	22	20
PCP	10	-	16	-	-	-	26	-
GER	3	-	-	-	-	-	3	-
INF	9	-	4	-	-	N 4	13	4
	182	38	341	16	234	30	757	84
		(17.3%)		(4.5%)		(11.4%)		(10.0%)

The table shows that, when using lexical heuristics, the parser performs best - not entirely surprisingly - for wellformed Portuguese words (B). Of 323 nouns and adjectives in group B, only 16 (5%) were misanalysed as false positives or false negatives. The probability for an assigned N-tag being correct is as high as 98.6%, for the underrepresented adverb and non-finite verbal class even 100%. All false positive nominal readings (N and ADJ) are still in the nominal class, a fact that is quite favourable for later syntactic analysis.

Figures are lower for group C, unknown loan words, where the chance of an N-tag being correct is only 92.6%, even when allowing for a name-chain-like N-analysis of English adjectives integrated in noun clusters of the type 'big boss'. Finite verb readings, though rare (due to lacking flexion indicators), are of course all failures, and only the little adjective group was a hit, the few cases being triggered by morphologically "Portuguesish" Spanish or Italian words.

The results in group A (misspellings) resemble those of group B, with a good performance for classes with clear endings, i.e. non-finite verbs and '-mente'-adverbs, and a bad performance for finite verb forms. For the large nominal groups figures are somewhat lower: 84.4% of N-tags, and only 71.4% of ADJ-tags are correct - though most false positive ADJ-tags are still within the nominal range. The lower figures can be partly explained by the fact that misspelled closed class words (adverbs, pronouns and the like) will get the (default, but wrong) noun reading - a technique that works somewhat better and more naturally for foreign loan words (C), which often are "terms" imported together with the thing or concept they stand for, or names. Also, the

<sup>13</sup> Only individual words and short integrated groups are treated, foreign language sentences or syntactically complex quotations are treated as "corpus fall-out" in this table.

percentage of "simplex"<sup>14</sup> words without affixes is much higher among the misspellings in group A than in group B, where all simplex words - being spelled correctly - would have been recognised in the lexicon anyway, due to the good lexicon coverage *before* getting to the heuristics module. Therefore, nouns and adjectives in group A lack the structural information of suffixes that helps the parser in group B: 'xxxo' looks definitely less adjectival than 'xxxístico'. In particular, 'xxxo' invites the N/ADJ-confusion, whereas many suffixes are clearly N or ADJ. Thus, '-ístico' yields a safe adjective reading.

#### 4. Special - "deviant" - word class probabilities for the heuristics module

Is it possible, apart from morphological-structural clues, to use "probabilistics pure" for deciding on word class tags for "unanalyzable" words? In order to answer this question, I will - in table (14) - rearrange information from table (13) and compare it to whole text data (in this case, from a 197.029 word stretch of the mixed genre Borba-Ramsey corpus). Here, I will only be concerned with the open word classes, nominal, verbal and '-mente'-adverbial.

(11) Open word class frequency for "unanalyzable" words as compared to whole text figures

	whole text		"unanalyzable" words						
			orthographical errors		Portuguese words		foreign words		all heuristics
analyse s	%	cases	%	cases	%	cases	%	cases	%
N	47.38	131	63.59	232	63.39	237	95.18	600	73.08
ADJ	12.79	33	16.02	100	27.32	12	4.82	145	17.66
ADV <sup>15</sup>	1.26	3 (+9)	1.46	5	1.37	- (+11)	-	8	0.97
VFIN	24.96	16	7.77	9	2.46	-	-	25	3.05
PCP	4.96	11	5.34	16	4.37	-	-	27	3.29
GER	2.47	3	1.46	-	-	-	-	3	0.37
INF	6.17	9	4.37	4	1.09	-	-	13	1.58
		206		366		249		821	

Among other things, the table shows that the noun bias in "unanalyzable" words is much stronger than in Portuguese text as a whole, the difference being most marked in foreign loan words. The opposite is true of finite verbs which show a strong tendency to be analysable. Finite verbs are virtually absent from the unknown loan word group. For the non-finite verbal classes the distribution pattern is fairly uniform, again with the exception of foreign loan words.

<sup>14</sup> "Simplex" words are here defined as words that can be found in the root lexicon without prior removal of prefixes or suffixes. Of course, the larger the lexicon the higher the likelihood of an (etymologically) affix-bearing word appearing in the lexicon, - and thus not needing "live" derivation from the parser.

<sup>15</sup> Only deadjectival '-mente'-adverbs can meaningfully be guessed at heuristically, and therefore only they should enter into the statistics for word class guessing. Also the base line figure of 1.26% for normal text is for '-mente'-adverbs only, the overall ADV frequency is nearly 12 times as high. Since non-'mente'-adverbs are a closed class in Portuguese, the latter will be absent from the heuristics class of wellformed unknown Portuguese words, but in the foreign loan word group and the orthographical error group they will appear in the false positive section of other word classes (numbers given here in parentheses). In the orthographical error group, both '-mente'-adverbs and closed class adverbs can occur, the first as correct ADV-hits, the other usually as false positive nouns (for instance, 'aimda').

As might be expected, among the "unanalyzable" words, orthographical errors and correct Portuguese words show a remarkably similar word class distribution.

A lesson from the above findings might be to opt for noun readings and against finite verb readings in "unanalysable" words, when in doubt, especially where no Portuguese flexion ending or suffix can be found, suggesting foreign material. As a matter of fact, this strategy has since been implemented in the system, in the form of heuristical disambiguation rules, that discard VFIN readings and chose N readings for <MORF-HEUR> words, where lower level (i.e. safe) CG-rules haven't been able to decide the case contextually.

## 5. Conclusion

It can be shown that lexico-morphological heuristics - at least for a morphology-rich language like Portuguese - can be based on structural clues and the systematic exploitation of derivational and inflectional sublexica. Applied to improve analyser recall on the input level of a Constraint Grammar system, the described technique positively contributed to the overall performance of a lexicon based rule governed tagger/parser. Correctness rates of more than 99% were achieved for the morphological/PoS tagger module, with heuristic error rates running at 2% for proper name heuristics and 4.5% for the heuristical analysis of other unrecognised, but correctly spelled Portuguese word forms. In all, heuristic analysis was needed for 80% of all proper nouns (amounting to ca. 2% of running word forms in news text), but for less than 0.4% of non-name word forms. Finally, word class frequency counts suggest that PoS probabilities for "unanalyzable" words in Portuguese texts are quite different from those for the language on the whole.

## References

Bick, Eckhard, *Portugisisk - Dansk Ordbog*, Mnemo, Århus, 1993, 1995, 1997.

Bick, Eckhard, *The Parsing System "Palavras", Documentation*, unpublished Ph.D. project evaluation, 1995, 1997.

Bick, Eckhard, "Automatic Parsing of Portuguese", in *Proceedings of the Second Workshop on Computational Processing of Written Portuguese*, Curitiba, 1996.

Bick, Eckhard, "Dependensstrukturer i Constraint Grammar Syntaks for Portugisisk", in: Brøndsted, Tom & Lytje, Inger (eds), *Sprog og Multimedier*, Aalborg, 1997.

Bick, Eckhard, "Automatisk analyse af portugisisk skriftsprog", in: Jensen, Per Anker & Jørgensen, Stig. W. & Hørning, Anette (eds), *Danske ph.d.-projekter i datalingvistik, formel lingvistik og sprogteknologi*, pp. 22-20, Kolding, 1997.

Brill, Eric, "A Simple Rule-based Part of Speech Tagger", in *Proceedings of the Third Conference on Applied Natural Language Processing*, ACL, Trento, Italy, 1992.

Chanod, Jean-Pierre & Tapanainen, Pasi, "Tagging French - comparing a statistical and a constraint-based method", adapted from: *Statistical and Constraint-based Taggers for French*, Technical report MLTT-016, Rank Xerox Research Centre, Grenoble, 1994.

Francis, W.N. & Kucera, F., *Frequency Analysis of English Usage*, Houghton Mifflin, 1982.

Garside, Roger & Leech, Geoffrey & Sampson, Geoffrey (eds.), *The Computational Analysis of English. A Corpus-Based Approach*, London, 1987.

Karlsson, Fred, "SWETWOL: A Comprehensive Morphological Analyser for Swedish", in *Nordic Journal of Linguistics* 15, 1992, pp. 1-45.

Karlsson, Fred & Voutilainen, Atro & Heikkilä, Juka & Anttila, Arto (eds.), *Constraint Grammar, A Language-Independent System for Parsing Unrestricted Text*, Mouton de Gruyter, Berlin 1995.

Lezius, Wolfgang & Rapp, Reinhard & Wettler, Manfred, "A Morphology-System and Part-of-Speech Tagger for German", in: Dafydd Gibbon (ed.): *Natural Language Processing and Speech Technology*, Berlin, 1996.

Marcus, Mitchell, "New trends in natural language processing: Statistical natural language processing", paper presented at the colloquium *Human-Machine Communication by Voice*, organized by Lawrence R. Rabiner, held by the National Academy of Sciences at *The Arnold and Mabel Beckman Center* in Irvine, USA, Feb. 8-9, 1993.

Tapanainen, Pasi, "The Constraint Grammar Parser CG-2", University of Helsinki, Department of Linguistics, Publications no. 27, 1996.

Voutilainen, Atro & Heikkilä, Juka & Anttila, Arto, *Constraint Grammar of English, A Performance-Oriented Introduction*, Publication No. 21, Department of General Linguistics, University of Helsinki, 1992.

# An HPSG Marking Analysis of Danish Determiners and Clausal Adverbials

Costanza Navarretta and Anne Neville  
Center for Sprogteknologi  
Njalsgade 80, 2300 Copenhagen S, Denmark  
costanza@cst.ku.dk anne@cst.ku.dk

## Abstract

Determiners and adverbials in many languages occur in a fixed order which is not an issue that has received a great deal of attention in HPSG94[9]. To deal with the order of Italian determiners Allegranza ([2] and [3]) proposes a revision of HPSG94. We have adopted Allegranza's approach to account for the order of Danish determiners and we have extended it to describe the order of cooccurring clausal adverbials. Our extension provides evidence that Allegranza's revision of HPSG can be used not only to account for determiners in other languages than Italian, but also other phenomena that exhibit a similar combinatorial behaviour.

## 1 Introduction

Danish determiners, like determiners in other languages, combine in a fixed order within the noun phrase. The order of determiners is not an issue that has received much attention in HPSG94[9]. In HPSG94 the class of determiners forms a uniform category having no subcategories. Determiners are lexically assigned the function specifier which is restricted to the category of *functionals*. *Functionals* include the typical minor categories lacking phrasal projections, but are also meant to include words that project [9, chapter 9]. This analysis cannot account for the categorial diversity of determiners which is essential to any description of their order. Since determiners in HPSG94 belong to the same category, the combination of determiners cannot be constrained by letting determiners or nouns select subcategories of determiners. Also, certain determiners may function as either specifier or adjunct depending on the context they occur in. Since function-specification takes place in the lexicon irrespective of context, this analysis would require two lexical entries for each of these determiners, one for when they function as specifier and one for when they function as adjunct.

Danish clausal adverbials also combine within the clause in a fixed mutual order and this order can be described in terms of their subcategories. In this respect they resemble determiners. This property of adverbials is not dealt with in [9] either.

In (2)<sup>1</sup> we present Danish data illustrating the mutual order of determiners in noun phrases and clausal<sup>2</sup> adverbials in main and subordinate clauses. Then, in (3), Allegranza's ([2] and [3]) alternative analysis of Italian determiners, within the framework of HPSG, is presented. In (4) we describe how we have adapted Allegranza's approach to formalize Danish determiners which do not exhibit the same cooccurrence patterns as Italian determiners. We further show that the approach can be extended to constrain the mutual order of clausal adverbials<sup>3</sup>. Finally, in (5) we conclude and propose that the order of other categories may be constrained in a similar fashion.

## 2 The Order of Danish Determiners and Clausal Adverbials

### 2.1 Determiners

Determiners include the following categories:

- (1) a. **articles:** *den* (the), *en* (a)
- b. **demonstratives:** *denne* (this), *disse* (these), ...
- c. **possessives:** *min* (my), *din* (your), ...
- d. **quantificational det.:** *alle* (all), *enhver* (every/each), *mange* (many), ...
- e. **cardinals:** *en* (one), *to* (two), ...
- f. **ordinals:** *første* (first), *anden* (second), ...

Determiners are commonly classified according to their distribution, and hence divided into predeterminers, central determiners and postdeterminers (cf. [10]). The classification is based on their position in the noun phrase with respect to each other, pre- and postdeterminers pivoting on the central determiner. Quantificational determiners cut across all three classes and we may subdivide them into quantificational pre-, central and postdeterminers. Articles, demonstratives and possessives are central determiners, cardinals and ordinals are postdeterminers. (2) shows the successive attachment of a pre-, central and postdeterminer.

- (2) a. *alle katte*  
(all cats)
- b. *alle disse katte*  
(all these cats)
- c. *alle disse mange katte*  
(all these many cats)

---

<sup>1</sup>The work described in this paper in part originates from research carried out within the framework of two EU funded projects, MLAP93-09[11] and LSGRAM (LRE 61029[7]).

<sup>2</sup>By clausal adverbials we mean adverbials which are placed in the so-called Actualization Field (in Danish "Nexus Felt" following Diderichsen[4]) i.e. they follow the finite verb or the finite verb and the subject in main clauses, while in subordinate clauses they occur in-between the subject and the finite verb.

<sup>3</sup>In this paper we focus exclusively on the syntax of determiners and adverbials.

The classification in [10] which is based on English determiners presupposes that determiners from the different classes are mutually exclusive. However, in Danish, possessives and the definite article may in certain contexts cooccur with a central determiner, as shown in (3) (cf. [5]):

- (3) a. *dette mit eneste ønske*  
(this my only wish)  
b. *mine de røde vanter*  
(my the red gloves)  
c. *dette det første forsøg*  
(this the first attempt)

Thus in (3a) the possessive which usually functions as a central determiner cooccurs with a demonstrative determiner. In (3b) the definite article is preceded by a possessive determiner, and finally in (3c) the definite article is preceded by a demonstrative determiner.

## 2.2 Clausal Adverbials

Adverbs, prepositional phrases, temporal nominal phrases (e.g. *hele dagen* (all day)), participles and subordinate clauses may all function as adverbials modifying clauses.

Main clauses and subordinate clauses have different word order. In main clauses adverbials can occur in the three positions:

- final position, after the main verb and its complements:  
*Jeg rejser til Italien om en måned*  
(I will travel to Italy in a month)
- initial position, before the finite verb<sup>4</sup>:  
*Om en måned rejser jeg til Italien*  
(In a month I will travel to Italy)  
(lit. In a month will I travel to Italy)
- in the so called “Actualization field”, after the finite verb or after the finite verb and the postpositioned subject<sup>5</sup>:  
*Jeg vil faktisk rejse til Italien om en måned.*  
(I will actually travel to Italy in a month.)  
and  
*Til Italien vil jeg faktisk rejse om en måned.*  
(To Italy, I will actually travel in a month.)  
(lit: To Italy will I actually travel in a month)

Some adverbials can occur in all positions (free adverbials), some only in the Actualization field, (clausal adverbials), some cannot occur in the Actualisation field, some occur both

<sup>4</sup>Note that the subject follows the finite verb in main clauses with an initial topicalized element.

<sup>5</sup>In main clauses the subject is postpositioned in simple interrogative clauses and in comment clauses in addition to the previously mentioned clauses with a topicalized element.

in initial position and in the Actualisation field. A few adverbials have different meanings depending on whether they occur in the Actualisation field or not.

In subordinate clauses adverbials can only occur in the Actualisation field (after the subject and before the finite verb) and/or after the main verb and its complements (final position).

Clausal adverbials modify the entire clause and have a fixed order, e.g.

- (4) a. *Jeg kan altså faktisk sjældent komme tidligere.*  
(I can therefore really seldom come earlier.)
- b. *Jeg ved at jeg altså faktisk sjældent kan komme tidligere.*  
(I know that I can therefore really seldom come earlier.  
Lit. I know that I therefore really seldom can come earlier.)

are correct clauses while the following two are not:

- (5) a. \* *Jeg kan sjældent faktisk altså komme tidligere.*
- b. \* *Jeg ved at jeg sjældent faktisk altså kan komme tidligere.*

Clausal adverbials are divided into four groups by Allan et al.[1] as follows:

- (6) 1. **Short modal adverbs:** *da* (surely), *jo* (certainly), *skam* (you know), *vel* (presumably)...
2. **Conjunctive adverbs:** *altså* (consequently), *derfor* (therefore), *desuden* (additionally), *dog* (still/yet)...
3. **Longer, modal adverbs and prepositional phrases:** *antagelig* (probably), *egentlig* (really), *faktisk* (actually), *i det hele taget* (all together), *oven i købet* (in addition), *trods alt* (in spite of everything), *virkelig* (really...)
4. **Negations:** *aldrig* (never), *ikke* (not)...

The four groups reflect the relative order of clausal adverbials cooccurring in a clause, i.e. short modal adverbials occur first, followed by conjunctive, longer modal and negation adverbials as illustrated in (7):

- (7) *Peter har vel derfor faktisk aldrig været i København*  
                  1      2      3      4  
(Peter has presumably therefore actually never been in Copenhagen)

To the fourth group (negations) we have added expressions of frequency such as *altid* (always), *sjældent* (seldom), *ofte* (often). However, the relative order of clausal adverbials is not as simple as described in [1]. Adverbials from the first three groups can cooccur freely with adverbials of the same type<sup>6</sup>. Moreover, in main clauses the main verb occurs before the clausal adverbials while in subordinate clauses it follows them.

<sup>6</sup>Actually, there are some restrictions on the combination of adverbials of the same type, which are not mentioned in [1]. We will not describe them in this paper.



### 3 Allegranza's Revised Selection and Marking

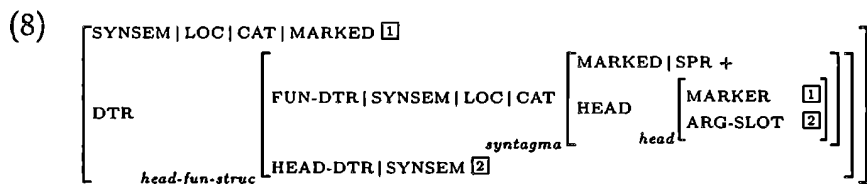
As we have shown in section 2 both determiners and clausal adverbials combine in a fixed order. In order to constrain the combination of Italian determiners Allegranza ([2] and [3]) proposes a revision of HPSG94 [9]. This revision allows for a subcategorization of determiners and the specification of a single lexical entry for determiners which may be adjunct or specifier according to the context in which they occur.

In HPSG94 determiners and adverbials are specifiers and adjuncts respectively and, together with markers (complementizers and conjunctions), they are non-head selectors.

The functional specifiers and markers, select their head sister by means of the SPEC head feature according to the HEAD-SPECIFIER and the HEAD-MARKER SCHEMATA. In addition, in the HEAD-SPECIFIER SCHEMA the head daughter selects the specifier daughter via the category feature SPR<sup>7</sup>. The substantive adjuncts select the adjacent head-daughter by means of the head feature MOD in the HEAD-ADJUNCT SCHEMA[9, p. 46].

Marking is restricted to markers and is used to block the cooccurrence of markers in the same construction. The value of MARKING, i.e. *marking*, is subtyped into *marked* and *unmarked*. The sort *marked* is subtyped into *complementizer* and *conjunction*, and *complementizer* is again subtyped into *that* and *for*.

Allegranza, similarly to Netter ([8]), generalizes HPSG non-head selection to cover selection by a general functor and extends the marking mechanism to cover all non-head functors. In the revised HPSG a *head-functor-structure* replaces HPSG *head-specifier-structure*, *head-marker-structure* and *head-adjunct-structure* and a FUNCT-DTR substitutes SPR-DTR, ADJ-DTR and MARK-DTR. The FUNCT-DTR has two head features a selecting ARG-SLOT which replaces SPEC and MOD, and a marking feature MARKER. The value of MARKER is structure-shared with the MARKED value of the mother node in a *head-functor-structure*. The SPR which in HPSG was a valency list is treated as a simple boolean feature similar to Netter's FCOMPL<sup>8</sup>. The HEAD-FUNCTOR SCHEMA replacing the HEAD-SPECIFIER, HEAD-ADJUNCT and HEAD-MARKER Schemata in [9] is shown in example (8) (c.f. [3, p. 34]):



Determiners are now functors and thus have the property of marking their head sisters. This is exploited in Allegranza's account of Italian determiners. Italian determiners have the attribute MARKER with the value *marking* which is subtyped as follows:

- (9) Partitions of *marking*: *unmarked*, *marked*.  
Partitions of *marked*: *determination (det)*, ...

<sup>7</sup>This mutual selection giving rise to a recursive structure has been criticized in both [8] and [3].

<sup>8</sup>Netter[8] introduces the concept of functional completeness in addition to subcategorization to account for projection levels of nominals and suggests an analysis of German NP specifiers as functional heads.

Partitions of *det*: *switch-det*, *source-det*.

Partitions of *switch-det*: *outer-det*, *inner-det*.

Partitions of *source-det*: *inner-det*, *baretype-det*.

The various partitions are further specified as in (10)-(14)<sup>9</sup>:

(10) *marked*: [SPR *boolean*]

(11) *det*: [ QUAMARK *boolean*  
ORDMARK *boolean*  
POSSPRO *list(synsem)* ]

(12) *outer-det*: [SPR +]

(13) *inner-det*: [SPR -]

(14) *baretype-det*: [SPR +]

Allegranza's approach solves the problems the HPSG analysis of determiners presented. Firstly, the type hierarchy reflects the categorial diversity of determiners. The attributes QUAMARK, ORDMARK and POSSPRO reflect the fact mentioned above that determiners do not form a uniform group. Quantificational determiners (QUAMARK), ordinals (ORDMARK) and possessives (POSSPRO) are different subcategories and enter into each their fixed position in the noun phrase. The attributes are used to constrain selection by determiners of their nominal head-sister.

Secondly, the functional variation of determiners as specifiers or adjuncts is formalized by the subtyping and the attribute SPR. Determiners of type *outer-det* are specifiers if they project SPR - nominals into SPR + nominals, in other words if they select *inner-det* nominals. To this group belong Italian articles, demonstratives and central quantifiers, such as *il*, *questo*, *ogni* (the, this, every). If, on the other hand, they project SPR + nominals into SPR + nominals, i.e. pass on the SPR value of their head-sister, they are adjuncts (Italian predeterminers are *tutto*, *entrambi* (all, both)). Determiners of type *inner-det* are always adjuncts and project SPR - nominals. Underspecified *source-det* determiners whose projections are resolved to nominals of type *baretype-det* by external selection are always adjuncts, and the SPR + value originates from the head noun (Italian ordinals and possessives, e.g. *secondo*, *mio* (second, mine)). *Source-det* determiners that are resolved to *inner-det* by the attachment of a preceding determiner are adjuncts, i.e. SPR -. The type *switch-det* is likewise underspecified. A determiner lexically underspecified as a *switch-det* determiner may end up as either an *outer-det* determiner if no other determiner precedes, hence specifier, or an *inner-det* determiner, hence adjunct, when another determiner precedes. In Italian this holds for cardinals and

<sup>9</sup>Note that POSSPRO takes a list as its value. Nouns are not assumed to subcategorize for possessives. However, Allegranza notes that possessives are subject to binding theory (cf. [2, p. 69]), and consequently must appear on the subcategorization list of the head noun. To achieve this he structure-shares the possessive's *synsem* with an element on the head noun's subcategorization list via POSSPRO. Therefore the value of POSSPRO is not boolean. It should also be noted that the marking mechanism presupposes a binary branching analysis of constituent structure, which is a departure from HPSG94.

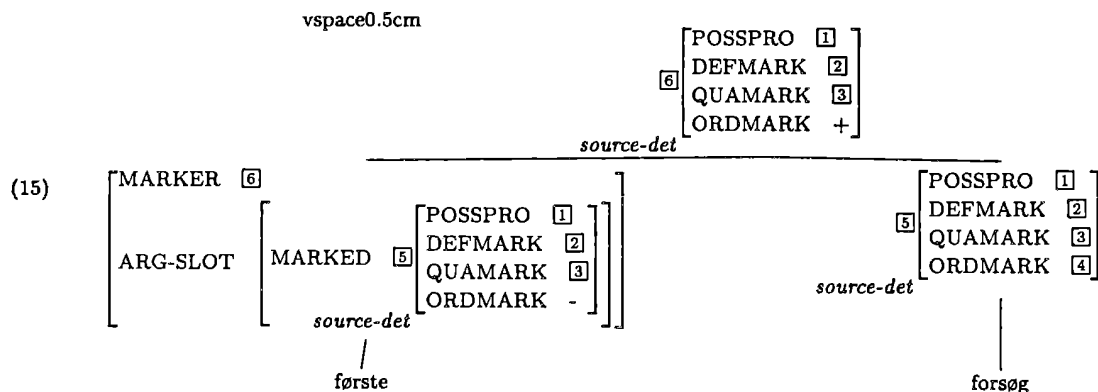
quantificational postdeterminers, e.g. *due*, *molti* (two, many), as shown in the following two examples: *due bambini* (two children) and *i due bambini* (the two children) where *due* is a specifier and an adjunct respectively.

## 4 Formalizing Danish Determiners and Clausal Adverbials

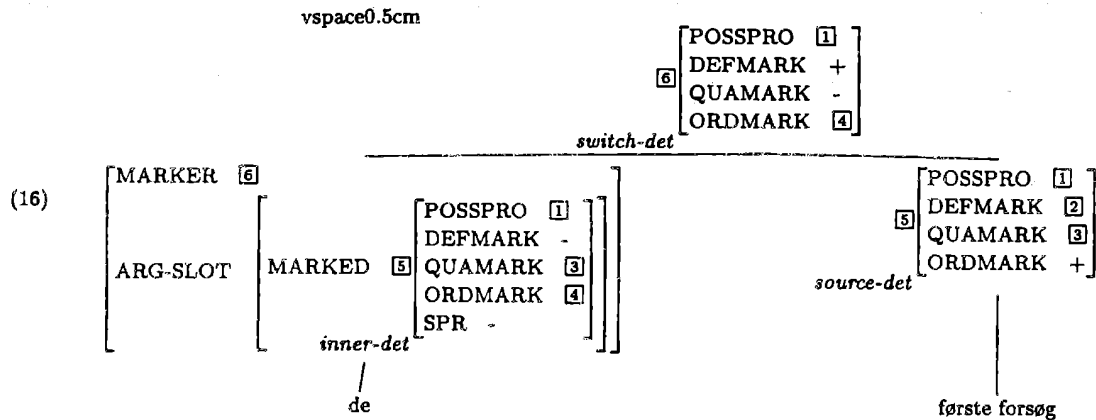
### 4.1 Determiners

In the following we will show how Allegranza's marking typology can be applied to the formalization of Danish determiners, by going through the successive attachments of determiners in the NP *alle disse de første forsøg* (all these the first attempts).

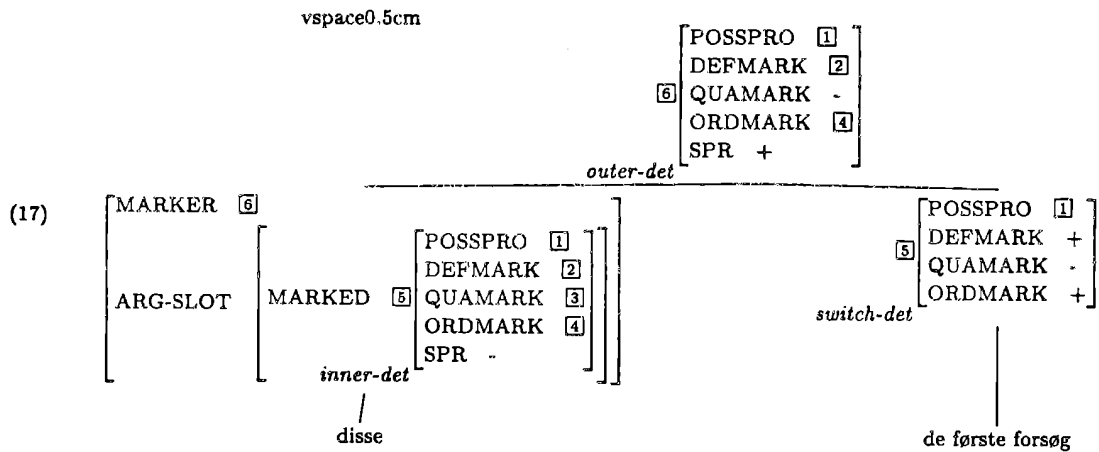
Danish ordinals, like the Italian ordinals, project ORDMARK + *source-det* nominals and select ORDMARK - *source-det* nominals. Ordinals pass on unchanged the SPR value of the nominal to which they attach. Apart from the ORDMARK marking, the representation of ordinals correspond to the representation of adjectives.



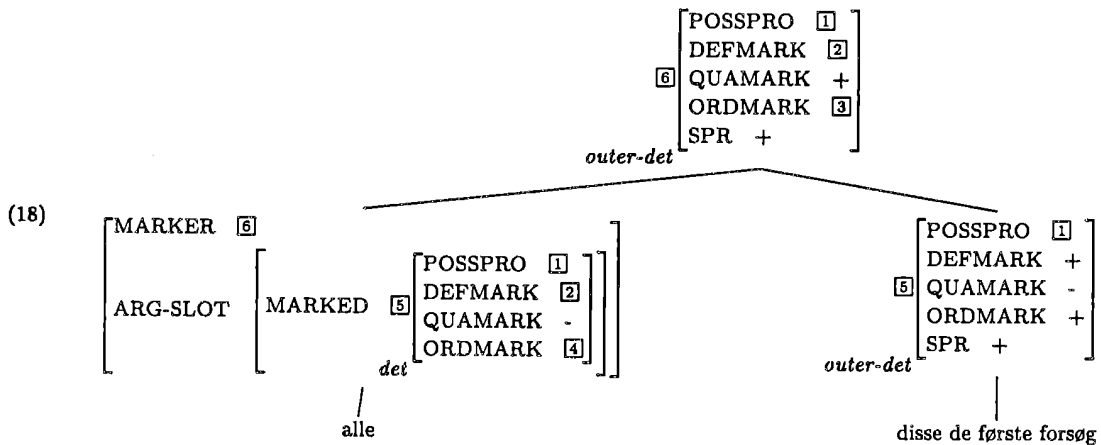
The Danish definite article may follow a demonstrative or a possessive. This is accounted for by treating it as *switch-det*. To avoid iteration of definite articles the attribute DEFMARK is introduced for Danish. The definite article selects DEFMARK - nominals and projects DEFMARK + marked nominals. They undo a potential QUAMARK + resulting from a previous attachment of a quantificational postdeterminer, allowing predeterminers to attach after a demonstrative even though the nominal may already contain a quantificational postdeterminer.



Danish demonstratives have the same marking and selection properties as Italian demonstratives, selecting an *inner-det* which is SPR -, and projecting an *outer-det*, resulting in an SPR + marked nominal.



Finally, the distribution of Danish predeterminers differs from that of Italian predeterminers. Italian predeterminers select *outer-det* marked nominals. As (18) shows, Danish predeterminers select for the less specific type *det* further constrained as QUAMARK -, avoiding immediate cooccurrence of quantificational determiners.



## 4.2 Clausal Adverbials

In the following we give a formal account of the mutual order of Danish clausal adverbials by further extending Allegranza's marking system. To control the placement of adverbials in a clause, we introduce a head feature *PLACEM* the value of which is subtyped into *front*, *end*, *nexus*. The value of *PLACEM* is encoded lexically in the entries of clausal adverbials. The marking system we describe concerns clausal adverbials, i.e. adverbials with the value *nexus*.

We have added the subtype *actualization* to Allegranza's *marked* partition as in (19):

- (19) **Partitions of *marked*: *actualization* (*act*),...**  
**Partitions of *act*: *main.cl*, *subord.cl*.**

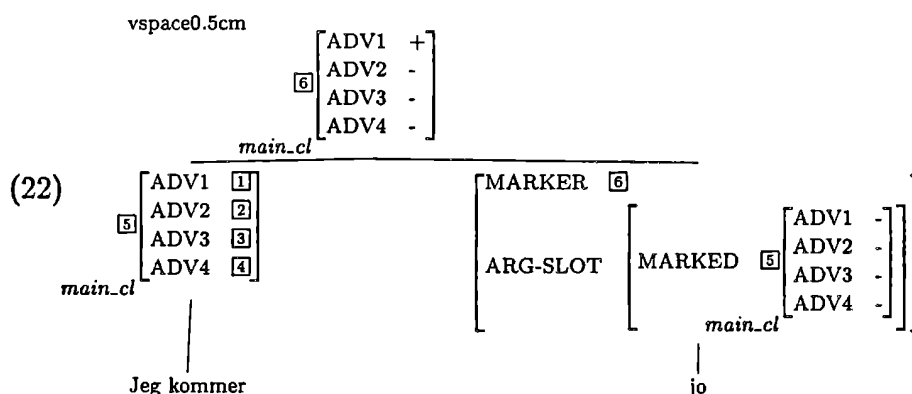
*main.cl* and *subord.cl* constitute the two subtypes of the marking system referring to adverbials in main clauses and in subordinate clauses respectively. The partitions are further specified in (20 and 21):

- (20)  $main.cl: \begin{bmatrix} ADV1 & boolean, & ADV2 & boolean \\ ADV3 & boolean, & ADV4 & boolean \end{bmatrix}$

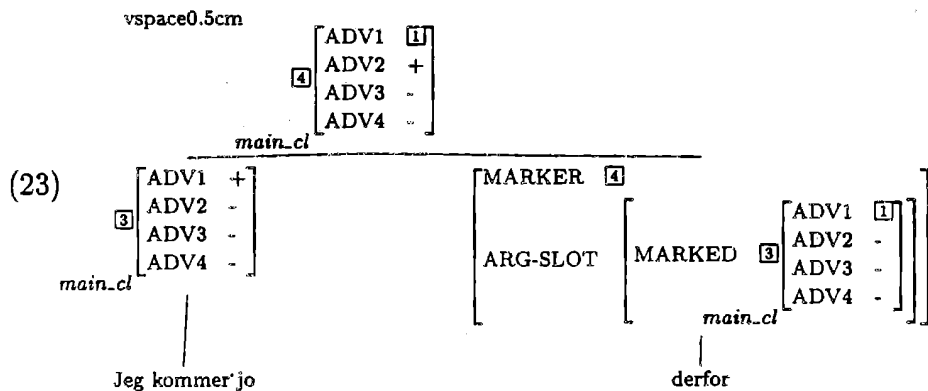
- (21)  $subord.cl: \begin{bmatrix} BADV1 & boolean, & BADV2 & boolean \\ BADV3 & boolean, & BADV4 & boolean \end{bmatrix}$

We now illustrate how the above formalization accounts for the attachment of clausal adverbials described in section 2.2. In particular we look at the attachment of the adverbials in the sentence *Jeg kommer jo derfor* (I come certainly therefore).

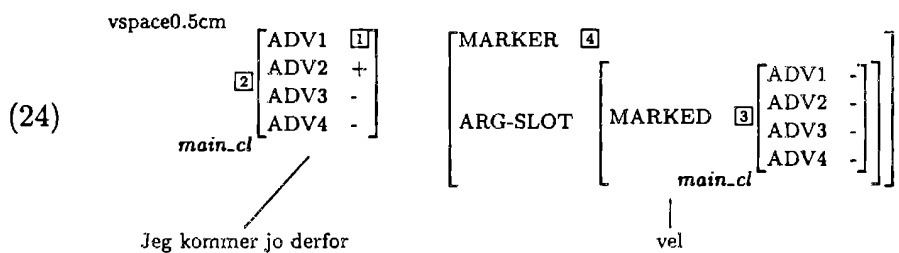
Short modal adverbials select main clauses that may contain an adverbial of their type, and project *ADV1 +*. The short modal *jo* attaches to the clause *Jeg kommer*, which does not contain any adverbials. Thus the marking of the clause unifies with the selecting marking of the short modal and the projected clause receives a positive value for *ADV1* as shown in (22).



When the conjunctive adverbial *derfor* (therefore) attaches to the clause *Jeg kommer jo*, its selecting marking unifies with that of the clause as shown in (23). The projected clause structure-shares the value of *MARKED* with the value of *MARKER* of the selecting adverbial.



Attaching the short modal *vel* to the projected clause *Jeg kommer jo derfor* in (23) is prevented by the marking mechanism. In fact the marking of the clause does not unify with that of the selecting adverbial as shown in (24).



## 5 Concluding Remarks

The order of cooccurring determiners in nominal phrases and of clausal adverbials in clauses is not treated in HPSG94. To constrain the ordering of Italian determiners Allegranza proposes a revision of HPSG selection and Marking Principle. The categorial and functional diversity of determiners is reflected in the formalized marking system.

In this paper we have adopted Allegranza's approach to treat the order of Danish determiners and clausal adverbials. Our extension of the marking system provides evidence that Allegranza's approach can be used not only to describe determiners in other languages than Italian, but also to deal with another selecting category that occurs in a fixed order.

Also in other languages than Danish there are order restrictions on certain adverbials. For example in English *He can actually always get away with arriving late* is a correct sentence while *He can always actually get away with arriving late* is not. Although the mutual order of such adverbials is quite flexible, it is possible to find some general rules [10]. The same applies for the combination restrictions on punctuation markers [6]. The ordering involved in these kinds of phenomena can be constrained by means of the extended marking system in a way similar to that presented in this paper.

## References

- [1] R. Allan, P. Holmes, T. Lundskær-Nielsen. 1995. *Danish - A Comprehensive Grammar*. Routledge, London.
- [2] V. Allegranza. 1995. *Integrated Report on Determination and Quantification*. LINDA - MLAP93-015, gruppo DIMA, Torino.
- [3] V. Allegranza. forthcoming. Determiners as Functors: NP Structure in Italian. In S. Balari & L. Dini (eds) *Romance in HPSG* CSLI, Stanford.
- [4] P. Diderichsen. 1946. *Elementær Dansk Grammatik*. Gyldendal, Copenhagen.
- [5] E. Hansen. 1994. *Kvalificeret bestemthed*. Unpubl. manuscript, Copenhagen University.
- [6] B. Jones. 1996. Towards a Syntactic Account on Punctuation. In *Proceedings of COLING-96*, Copenhagen.
- [7] B. Music and C. Navarretta. 1996. Documentation of the Danish Lingware. LRE 61029, LSGRAM Deliverable CST,E-D8-DK, Copenhagen. Aix-en-Provence.
- [8] K. Netter. 1994. Towards a Theory of Functional Heads: German Nominal Phrases. In J. Nerbonne, K. Netter and C. Pollard, editors, *German in Head-Driven Phrase Structure Grammar*. CSLI, Stanford.
- [9] C. Pollard and I.A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. CSLI, the University of Chicago Press, Chicago.
- [10] R. Quirk, S. Greenbaum, G. Leech, J. Svartvik. 1985. *A Comprehensive Grammar of the English Language*. Longman, London.
- [11] N. L. Underwood (ed), C. Povlsen, P. Paggio, A. Neville, B. S. Pedersen, L. D. Jørgensen, B. Ørsnes, A. Braasch. 1996. *LINDA - Linguistic Specifications for Danish*. MLAP93-09, CST, Copenhagen.

# A Chart-Based Framework for Grammar Checking Initial Studies

**Anna Sgvall Hein**

Department of Linguistics  
Uppsala University  
anna@ling.uu.se

## 1 Introduction

A language checker, typically, has two basic components, a spell-checker and a grammar checker. Whereas the spell-checker, usually, limits its operation to the inspection and correction of individual text words, the grammar checker has to cope with errors that can only be detected in contexts that are larger than the word. Among the latter we find not only syntactic errors in the traditional sense, but also punctuation errors, soft lower case errors, and other violations of graphical conventions (Wedbjer Rambell 1998). Here we will only discuss errors that may be denoted construction errors, i.e. syntactic errors and certain types of punctuation errors, e.g. separators such as comma. Spelling errors that result in proper words but wrong constructions and thus cannot be captured by a spell-checker also belong to this group. For the handling of soft lower case errors and other violations of graphical conventions including erroneous use of dash, quotations mark etc. a simpler machinery may be envisaged.

We will start by examining different levels of functionality of a grammar checker. Then we will study what can be achieved by a combination of robust partial parsing and the application of local error rules. In specific, a chart-based implementation of such a framework will be presented and from our experience with this framework we will draw some conclusions.

## 2 Functionality of a grammar checker

In the first place, a grammar checker has to *detect* a construction error. Further, in order to *correct* it, or recommend a correction, it must provide a *diagnosis*. As regards the diagnosis, Uszkoreit (1996), in addition, suggests a level of *recognition*. Recognition means identifying the nature of the error in terms of localisation and constraint violations, e.g. violation of subject – verb agreement. The diagnosis should identify the source of the error as a basis for correction, e.g. changing the subject, or changing the verb. Such a four level scheme seems to provide a useful framework for the discussion of grammar checking functionality.

### 1. Detection

Identification of segments possibly containing an error

### 2. Recognition

Localisation and identification of constraints possibly violated



### 3. Diagnosis

Identification of possible error sources

### 4. Correction

- a) finding or construction alternatives
- b) ordering of alternatives
- c) substituting most highly ranked alternatives

Figure 1: Possible tasks in grammar correction (from Uszkoreit 1996)

---

A parser with a complete grammar of the language or language fragment to be checked appears to be the primary tool for error detection. It will be capable of making a primary division between correct and incorrect sentences according to its grammar. All constructions outside the scope of the grammar will be flagged as erroneous. It will not, however, per se provide the functionality that is required for the recognition and diagnosis of an error. In order to do so, the parser must have quite detailed knowledge of the kinds of errors that may occur. A grammar rule may be violated in several ways, and there is no limit to the kinds of errors that may occur, at least if accidental performance errors are to be considered. Consequently, a complete account of all possible kinds of errors is out of reach. This is the motivation for the high priority given to error collection and error modelling in the Scarrie project<sup>1</sup>.

Approximately 9,000 proof-reading errors have been analysed and stored in an error database, the Scarrie *Error Corpora Database*, *ECD*. Each database entry comprises an error fragment and its correction marked with an error type code in accordance with an error typology that was also developed in the Scarrie project (see Wedbjer Rambell 1998). The error material was provided by two prominent Swedish newspapers, *Svenska Dagbladet*, a contracted partner of the project, and *Uppsala Nya Tidning*, one of its subcontractors. See further (Wedbjer Rambell et al. 1998).

Below we will examine the applicability of the Uszkoreit scheme to the Scarrie error typology. The examples will all be chosen from the ECD.

A first example from the ECD:

---

Error text	Proof-reader's correction	Error type code
<i>*det tidiga 1800-talen</i>	<i>det tidiga 1800-talet</i>	GPNPAG01

---

The error type code reflects the four levels of the typology.

Group      GP:    grammar problem

---

<sup>1</sup> The Scarrie project aims at the development of a proof-reading tool for the Scandinavian publishing industry, see further url:<http://guagua.echo.lu/langeng/en/le3/scarrie/scarrie.html>.

Category                    NP:    nominal phrase  
Subcategory    AG:    agreement  
Specification 01:    number

The proof-reader has chosen to correct the noun, changing its number from plural to singular. However, a correction of the determiner by a change of number from singular to plural is also an alternative to consider. The example with the two possible corrections fits well into the Uszkoreit scheme:

---

*\*det tidiga 1800-talen*

Detection:                NP is flagged

Recognition:    violation of number agreement in premodifier – noun

Diagnosis:                1. plural instead of singular in noun  
                              2. singular instead of plural in premodifier

Correction:                1. *det tidiga 1800-talet*                    (one correction step)  
                              2. *de tidiga 1800-talen*                    (one correction step)

Figure 2: A simple example from the ECD in the Uszkoreit (1996) framework

---

Below we present an example of an error that can be analysed in more than one way. The first analysis is in accordance with the correction made by the proof-reader. The second one was found by ScarCheck, our prototype grammar checker (to be described below).

---

*\*det slutgiltiga siffrorna*

Detection:                NP is flagged

Recognition:    1. violation of number agreement in premodifier - noun  
                              2. violation of gender and number agreement in premodifier - noun

Diagnosis:                1. singular instead of plural in premodifier  
                              2. neuter instead of utrum in premodifier and  
                                     plural instead of singular in noun

Correction:                1. *de slutgiltiga siffrorna*                    (one correction step)  
                              2. *den slutgiltiga siffran*                    (two correction steps)

Figure 3: An example of alternative error analyses in the Uszkoreit (1996) framework

---

The accomodation of this example in the Uszkoreit scheme is not quite straightforward. Splitting the recognition level up into two, i.e. the localisation of the error (premodifier - noun agreement), and the identification of the constraints possibly violated is an alternative to be considered. Apart from that, we think that this scheme provides a good basis for discussing and comparing grammar checking functionality.

In its present version, the ScarCheck grammar checking prototype does not go beyond the recognition level. It does, however, provide mechanisms for ordering alternatives at this level.

### 3 Constraint relaxation and the application of local error rules

Bustamante&Sánchez (1996) make a primary division of grammatical errors into non-structural and structural errors. Non-structural errors are mismatchings of features that do not affect configurational issues. Examples of non-structural errors are agreement violations. Structural errors are mismatchings of features that represent syntactic categories. Examples of structural errors are wrong head-argument relations, word order errors, and substitution of categories. For the handling of non-structural errors the authors propose constraint-relaxation techniques. Such techniques are also used for the detection of structural violations concerning the use of prepositions in head-argument relations. The approach has been implemented in prolog, and applied to Spanish. The resulting demonstrator covers intra- and intersyntactic agreement, errors due to wrong usage of bound prepositions in head argument relations, and errors on portmanteau words. Certain stylistic aspects are also covered. The authors also provide a comprehensive background to grammar checking.

A similar approach has been chosen for Swedish. The main difference is to be found in the Swedish attempt to use an underspecified grammar in order to achieve a realistic coverage. Thus a combination of robust partial parsing and the application of local error rules is being explored. Phrase constituents (NP, AdjP, AdvP, PP, VP (verbal core)) are analysed by means of rules that accept feature violations. Anticipated errors at clause and sentence level, be they structural or non-structural, are handled by means of local error rules. The local error rules operate on the results of the parsing process. They may be formulated in terms of phrase categories, lexical categories, lemmas, and morpho-syntactic features. All the rules are integrated in one single parsing grammar. Typically, each rule covers both the positive and the negative case.

#### 3.1 ScarCheck - a chart-based implementation of a grammar checker

The grammar checking strategy outlined above has been implemented in a chart-based framework. The prototype checker, *ScarCheck*, has two basic modules, a chart parser and a chart scanner. The parser builds as much structure as the grammar allows, and the scanner traverses the chart collecting and reporting errors. Below we present an example of the interaction with ScarCheck. (So far, there is no interface connecting the two modules). For the sake of the presentation, comments in quotation marks are added.

```
"User: invoking the parser:"  
  > (p "de tidiga 1800-talet ")           'the early 19th century'  
  
"System: respons"
```

1 parse, 22 vertices.

T

"User: printing the result:"

>(pr)

"System: respons"

de tidiga 1800-talet :

```
(* = (PHR.CAT = NP
      FIRST = +
      DEF = DEF
      FORM = DEF
      NUMB = PLUR
      GENDER = NIL
      DET = (LEM = DEN1.AL
            WORD.CAT = ART)
      ATTR = (PHR.CAT = ADJP
            HEAD = (1 = (LEM = TIDIG.AV
                      WORD.CAT = ADJ
                      DEGREE = POS))
            GENDER = NEUTR
            NUMB = <* NUMB>
            DEF = NIL2
            FUNC = ATTR
            A-FORM = A
            CASE = BASIC
            SEX = NIL)
      ERR = (1 = GPNPAG01)
      CASE = BASIC
      HEAD = (WORD.CAT = NOUN
            LEM = 1800-TAL.NN)))
```

OK

"User: invoking the chart scanner"

> (reportchart)

"System: respons"

---

<sup>2</sup> The NIL value unifies with any value.

```
FELTEXT 'erroneous text': de tidiga 1800-talet
FEL 'error': number agreement in premodifier - noun
```

**Figure 4: An example of a simple interaction with ScarCheck**

---

The error message generated by reportchart is an interpretation of the (first and only) error feature value (ERR = (1 = GPNPAG01)) that was found in the chart. The message provides two kinds of information, i.e. the local context in which the error was recognised, and a spelling-out of the error type code. In the example the local context is identical to the full input. This is not always so, as we will see below. First, however, we will present a case where two errors are found:

```
>(p "Det slutgiltiga siffrorna ") 'the final figures'

1 parse, 27 vertices.
T
> (reportchart)

FELTEXT: Det slutgiltiga siffrorna
FEL: 1 number agreement in premodifier - noun
      2 gender agreement in premodifier - noun
```

**Figure 5: Two errors recognised by ScarCheck**

---

The two errors are presented in the same order as they appear in the chart. This is also the preferred order. Violation of number agreement is, by far, the most common agreement violation in the ECD. Ordering the errors according to priority is the task of the grammar rule writer.

### 3.1.1 Parsing for errors

The parser, UCP (Sågvall Hein 1983) uses a bottom-up strategy. This is in accordance with the partial nature of the parsing process; in the general case, there will be no edge spanning the whole chart, but a sequence of edges representing words and phrases. The grammar is formulated in a procedural formalism (Sågvall Hein 1983), and grammar rules (incl. local rules of anticipated errors) are triggered from the grammar. For instance, NP rules are triggered at the recognition of lexical categories that may appear as NP introducers.

Clause initial position and clause final position are important clues to the detection of structural errors at sentence level. As a means of identifying the beginning of a clause when it appears as the first constituent of the sentence, chart initial position is recorded as a feature in the morphological description of the first word of the input. An example of a rule where such a signal is motivated is the local error rule designed to catch erroneously deleted finite verbs in main clauses introduced by NPs. The invocation of the rule is conditioned by the position of the NP; it is only invoked if the NP appears at the potential beginning of a clause.

Unification of feature structures is the basic operation for test and assignment in the procedural UCP formalism.

Errors are recorded as features with values set in accordance with the error typology. In principle, the parser accepts any number of errors in a constituent.

Reportchart expects the errors to be located at the top level of the feature description. Consequently, the parser has to account for proper error propagation.

### 3.1.1.1 Error propagation

For instance, if an error is found in one of two coordinated NPs, it has to be propagated to the top level of the description of the coordinated NP (see fig. 6). It may be analysed in several ways. In particular, there are several options as regards the scope of the determiner (*det*) and the adjective (*större* 'bigger'). Do they modify both nouns (*maskinerna* 'machines' and *traktorerna* 'tractors') or only the first one? In cases like this one, the parser will provide only one analysis, and it will choose the most superficial one according to which the premodifiers modify only the first, and closest, noun. This decision is motivated by processing economy; we want the checker to be as fast and efficient as possible.

Let's examine the consequences of this choice. Two (alternative) agreement errors will be recognised in the first NP, whereas there are no errors to be found in the second one consisting of a singular noun. The errors that are found in the first NP are propagated to the top level of the description, i.e. the level of the coordinated NP. The coordinated NP constitutes the error context and will be presented as such to the user (see fig. 7). It may be argued that the context is too wide and thus counterintuitive. If so, the most obvious solution would be to make Reportchart take a deeper look into the structure at the expense of processing economy. We would not, however, be in a better position in this respect by making a "deeper" analysis.

```
det större maskinerna och traktorerna 'the bigger machines and tractors':
```

```
(* = (PHR.CAT = NP
      ERR = (1 = (1 = GPNPAG01
                2 = GPNPAG02))
      1 = (PHR.CAT = NP
          FIRST = +

          DEF = DEF
          FORM = DEF
          NUMB = SING
          GENDER = NEUTR
          DET = (LEM = DEN1.AL
                WORD.CAT = ART)
          ATTR = (PHR.CAT = ADJP
                 HEAD = (1 = (LEM = STOR1.AV
                               WORD.CAT = ADJ
                               DEGREE = COMP))
                 GENDER = <* 1 GENDER>
```

```

                NUMB = <* 1 NUMB>
                DEF = NIL
                FUNC = ATTR
                A-FORM = NIL
                CASE = BASIC
                SEX = NIL)
    ERR = <* ERR 1>
    CASE = BASIC
    HEAD = (WORD.CAT = NOUN
           LEM = MASKIN.NN) )
2 = (LEM = OCH.CN
    WORD.CAT = CONJ)
3 = (PHR.CAT = NP
    NUMB = PLUR
    GENDER = UTR
    CASE = BASIC
    DEF = DEF
    HEAD.FORM = <* 3 DEF>
    HEAD = (WORD.CAT = NOUN
           LEM = TRAKTOR.NN) ) ) )

```

**Figure 6: An example of a propagated error**

---

> (reportchart)

```

FELTEXT:   det större maskinerna och traktorerna
FEL:       1 number agreement in premodifier - noun
           2 gender agreement in premodifier - noun

```

**Figure 7: A report of a propagated error**

---

### 3.1.1.2 Changing or postponing a decision

Sometimes the parser needs to change its decision about an error. For instance, in a Swedish NP introduced by the definite determiner *den*, the head noun should be in the definite form, e.g. *de områdena* 'those areas' unless it is followed by a restrictive relative clause introduced by the pronoun *som* 'which'. In the latter case, the indefinite form is the proper one, e.g. *de områden som* 'those areas which'. The parser can handle such problems using its look-ahead facility (see further Sägval Hein 1983, p. 12) in combination with an error cancelling NOERR feature feature. It neutralises an ERR feature with the same value. For instance, *de områden* will be analysed as an NP with an agreement violation of the species value: GPNPAG03 (see fig. 8) whereas the error will be cancelled in the analysis of *de områden som* (see fig. 9).

```

de områden :
(* = (PHR.CAT = NP
     FIRST = +
     DEF = DEF
     FORM = DEF
     NUMB = PLUR

```

```

GENDER = NEUTR
DET = (LEM = DEN1.AL
      WORD.CAT = ART)
ERR = (1 = GPNPAG03)
HEAD = (WORD.CAT = NOUN
      LEM = OMRÅDE.NN)))

```

```

FELTEXT:  de områden
FEL:      1 species agreement in premodifier - noun

```

**Figure 8: An error that may be cancelled**

---

```

de områden [som] :
(* = (PHR.CAT = NP
     FIRST = +
     DEF = DEF
     FORM = DEF
     NUMB = PLUR
     GENDER = NEUTR
     DET = (LEM = DEN1.AL
           WORD.CAT = ART)
     ERR = (1 = GPNPAG03)
     HEAD = (WORD.CAT = NOUN
            LEM = OMRÅDE.NN)
     NOERR = (0 = GPNPAG03)))

```

**Figure 9: Cancelling an error**

Reportchart will consider the ERR feature cancelled by the NOERR feature. It will face a situation where there are two competing edges of the same length, one with an error in it and one with no error (error cancelled). It will prefer the error free one.

### 3.1.1.3 Parsing local error rules

Whereas a robust grammar rule recognises configurationally well-formed units, a local error rule, typically, applies to fragments of constituents. Such fragments are represented by edges in the chart but not used in the further analysis. For instance, a local error rule will apply to the initial fragment of the main clause *Det bli sång och musik* 'There will be song and music' and recognise an error in the verb form (infinitive/imperative instead of finite), see fig. 10. The rule was designed to capture non-structural violations of the verb-second rule in Swedish main clauses.



```

Det bli [sång och musik] :      'There will be song and
music'
(* = (ERR = (1 = GPVFFV01)
      PHR.CAT = CL.FRAG))

```

```

FELTEXT:  Det bli
FEL:      1 infinite verb instead of finite

```

**Figure 10: A non-structural error recognised by a local error rule: main clause**

---

An analogous rule was formulated for the recognition of improper verb forms in the position of the finite verb in subjunctive clauses, see fig. 11.

```

Om människor börja [tro] :      'If people begin [to believe]'
(* = (ERR = (1 = GPVFFV01)
      PHR.CAT = CL.FRAG))

```

```

FELTEXT:  Om människor börja
FEL:      1 infinite verb instead of finite

```

**Figure 11: A non-structural error recognised by a local error rule: conditional clause**

---

In the ECD there are also examples of deleted finite verbs. Since the position of the finite verb is fixed in Swedish, deleted finite verbs may be recognised by means of local error rules, provided that they are capable of recognising the preceding constituent (and, in subjunctive clauses, an optional adverb) in its full extension. Hereby, heavy demands are made on the parser's coverage, in specific, with respect to NPs. An example of a deleted finite verb that is recognised by means of a local error is presented in fig. 12. In this case the parser easily found the preceding constituent.

```

Det nödvändigt [att tänka i nya banor] :      'It necessary [to think in
new                                         ways]
(* = (PHR.CAT = CL.FRAG
      ERR = (1 = GPVVMV01)))

```

```

FELTEXT:  Det nödvändigt
FEL:      1 finite verb missing

```

**Figure 12: A structural error recognised by a local error rule**

---

### 3.1.1.4 Partial parsing

The parser starts its operation at the level of the characters. It records unknown strings, and saves them for the protocol. When a rule fails or there is no applicable rule it just moves on to the next word and the rules that are associated with that word are triggered. For instance, in fig. 13 we

present the error messages that were generated as a result of the analysis of a sentence in three independent segments. There is no edge covering the whole sentence (0 parses) but three partial parses are stored in the chart and interpreted by the chart scanner. The first segment was analysed by means of a local error rule, the second was reported missing from the dictionary, and the third segment was analysed by means of a robust NP rule.

The example *Det bli förmodligen det slutgiltiga siffrorna*. 'It become probably the final figures'. was constructed for the sake of the presentation, and for the same reason the adverb *förmodligen* was removed from the dictionary. (Sentences with this simple structure are likely to be covered fully by the grammar.)

```
> (p "Det bli förmodligen det slutgiltiga siffrorna. ")
0 parses, 48 vertices.
NIL
> (reportchart)

FELTEXT:  Det bli
FEL:  infinite verb instead of finite

FELTEXT:  förmodligen 'probably'
FEL:  Ordet finns ej i lexikonet.      'The word is not in the
dictionary.'
```

**Figure 13: Error messages based on partial parsing**

---

It is of vital importance that the parser is capable of recovering when rules fail or there are gaps in the grammar or the dictionary. Otherwise, it will not be capable of recognising errors that appear after that initial part of the sentence that is covered by grammar rules<sup>3</sup>.

As is normally the case in chart parsing, the parser ends its work when no more rules apply and the agenda is exhausted.

### 3.1.2 Scanning the Chart for Errors

The chart scanning module Reportchart traverses the chart in search for error features. Starting at the first vertex, it inspects the top level of the feature description of the longest inactive edge. If an error is found, a corresponding message is formulated and taken to the error protocol, otherwise the inspection just goes on to the final vertex of the current edge and the edgelist going out from that vertex. When there is a choice between several edges, the scanner always chooses

---

<sup>3</sup> An alternative framework for grammar checking in which robust parsing may be combined with the application of rules of anticipated errors has been proposed and implemented by Vosse (1994). The parser works bottom-up; still there seems to be no way of implementing partial parsing in an efficient way.

the longest one disregarding the others. When there are more than one edge of maximum length, and one of them has no errors, this edge is preferred and no error is recorded in the protocol. Before finally accepting an error, Reportchart verifies that the error has not been cancelled by means of a corresponding NOERR feature.

#### **4 Conclusions and future work**

The proposed approach has been tested in the implemented framework, both with regard to feature relaxation techniques for the handling of non-structural errors and the application of local error rules for the handling of structural errors and some types of non-structural errors at clause level. The results that were achieved so far are encouraging.

In specific, the handling of non-structural errors is found to be fairly straightforward. An issue to be further investigated though is the number of feature violations to be accepted for the various phrase types. If too many errors are accepted, the checker may overgenerate. Another problem that should be further explored concerns feature propagation. When and how far should an error feature be propagated?

The coverage of the checker will be modelled after the Scarrie Error Corpora Database. Preliminary studies indicate that a great number of error rules will be needed to account for the variety of error types that are represented in the base.

The prototype checker is also being evaluated in an application to controlled language at Scania (Sågvall Hein et al. 1997).

#### **References**

Bustamante, Flora Ramírez & León, Fernando Sánchez, 1996, *GramCheck: A Grammar and Style Checker*, in Proceedings of the 16<sup>th</sup> International Conference of Computational Linguistics (Coling -96): 175 – 181.

Sågvall Hein, Anna, 1983. *A Parser for Swedish. Status Report for Sve.Ucp. February 1983*. Report No. UC DL-R-83-2. Uppsala University. Center for Computational Linguistics.

Sågvall Hein, Anna, Almqvist, Ingrid, & Starbäck, Per, 1997. *Scania Swedish – A Basis for Multilingual Machine Translation*. Translating and the Computer 19. Papers of the Aslib conference held on 13 & 14 November 1997.

Starbäck, Per, forthcoming, *ScarCheck – a Software for Word and Grammar Checking*. Technical Report. Uppsala University. Department of Linguistics.

Uszkoreit, Hans, 1996. *Grammar Checking. Theory, Practice, and Lessons learned in LATESLAV*. Concluding oral presentation at the final review meeting of the Lateslav Project (PECO 2824). Prague. August 1996.

Wedbjer Rambell Olga et al., 1998. *An Error Database of Swedish*. SCARRIE, Deliverable 2.1.3.2, version final 1.0. Uppsala University. Department of Linguistics.

Wedbjer Rambell, Olga, 1998. *Error Typology for Automatic Proof-reading Purposes*. SCARRIE, Deliverable 2.1, version 1.1. Uppsala University. Department of Linguistics.

Vosse, Theo G., 1994. *The Word Connection*. Grammar-based Spelling Error Correction in Dutch. Amsterdam.

# CP-UDOG

## An Algorithm for the Disambiguation of Compound Participles in Danish

Jens Ahlmann Hansen and Poul Søren Kjærsgaard  
Odense University

### Abstract

This paper describes some aspects of the linguistic analysis which has been applied to disambiguate Danish compound participles (CPs) (sect. 1-3), followed by an introduction to the implementation of the disambiguation process (sect. 4-6).

### 1. Introduction

A CP consists of a present or past participle, to which another word is prefixed. This compound functions as an adjectival modifier in a noun phrase (clause level) or as a predicate to the subject/direct object (sentence level). Two examples:

1. *elproducerende vindmøller*
2. *vindmølleproduceret el*

The results of the analysis consist partly of the unfolding of a CP into a finite verb construction, either as a sentence or as a relative clause. The resulting unfolded construction allows for the identification of the syntactic and semantic functions of the elements integrated in the CP, the former by means of traditional structuralist methods, the latter by means of lexical information.

Within the framework of the UDOG-project (Research into Danish Vocabulary and Grammar), Poul Søren Kjærsgaard and Bjarne Le Fevre Jacobsen have sought to infer and to formalize the syntactic, semantic and lexical rules which underlie the formation of CPs (e.g. Jacobsen & Kjærsgaard, 1995). Specifically, they have addressed the following 3 aspects:

1. A description of CPs based on the valency and the semantic selectional restrictions of CPs.
2. A contrastive analysis of CP construction patterns and the finite construction patterns of their derivations.
3. The application of valency number and selectional restrictions in the analysis of syntactic structure and distribution of semantic roles in CP constructions.

The CP-UDOG implementation disambiguates compound-participles (CPs) in Danish. Furthermore, the application generates monolingual paraphrases in which the syntactic and semantic relationships between the participle form, its prefix and the head-noun of the NP become fully explicit.

CP-UDOG has both an actual and a potential function:

- I. A tool for testing linguistic hypotheses concerning the rules governing the generation of CPs in Danish.
- II. A prototype module for the automatic generation of translation equivalents in Romance languages from CPs in Germanic languages.

## 2. Linguistic data

The data that the investigation is based on was collected primarily from 1993 to 1996 from Danish newspapers and television teletext. The corpus amounts to more than 3000 examples of simple and compound participle constructions covering approx. 1000 different verbs.

## 3. Algorithm

The algorithm is data-driven and consists of two operations: analysis and generation (see appendix 1). The starting point is, first, the observation, resulting from several studies, that there exist some archetypical syntactic and semantic relations between the participle and a prefix on the one hand and between the (compound) participle and the head of a NP (clause level) or the subject/object (sentence level) on the other. These relations can be inferred from examples 1-2:

syntax:	direct object-present participle	subject
semantics:	patient	agent
1.	<i>el-                    producerende</i>	<i>vindmøller</i>
syntax:	subject-                    past participle	direct object
semantics:	agent	patient
2.	<i>vindmølle-            produceret</i>	<i>el</i>

The starting point is, second, the observation, resulting from the UDOG project, that there exist a number of deviating patterns which do not comply with the archetypes. The algorithm to be presented here aims at producing unfolded, hence disambiguated, paraphrases of both groups.

**The analysis module** consists of five parameters:

**The first parameter** the algorithm operates on is the type of participle: present or past. Danish present participles have an invariant ending: *-ende*; while past participles have a limited number of morphemes: *-et, -t, -te, -ede, -ne*. In either case, a second condition to be satisfied is that the root of the word under consideration can be identified as a verb. This operation is carried out by a dictionary lookup.

**The second parameter** is the valency of the verb that the participle is derived from. It is determined by a dictionary lookup.

Verbal valency is defined as the minimal number of dependents (including the subject) a verb presupposes in order for the verbal action to take place. Apart from a valent verbs, we distinguish rather traditionally between monovalent, bivalent and trivalent verbs. The number of dependents is crucial to the disambiguation process since it may identify for instance a nominal prefix which does not fill the function which the archetype would otherwise predict. Syntactic valency is also

important since it contributes to distinguish between direct and prepositional objects (the preposition belonging to the latter is deleted during the formation of compound participle).

**The third parameter** is the class the prefix belongs to. Most prefixes are lexical items in their own right, e.g. *el* and *vindmølle*. A few, however, are not, e.g. *be-*, *u-*). In either case, the class is determined by a dictionary lookup.

The archetypical patterns take into account only nominal prefixes, but the set of possible prefixes includes, among others, adjectives, adverbs, prepositions, particles and bound morphemes. If the prefix does not belong to the class of nouns, it cannot fill a nominal function such as subject or direct object. Further, the third parameter may cooperate with the second parameter to identify the function of the prefix. This happens when a nominal prefix represents an ellipsis of a PP (the preposition being elided) or vice versa when a preposition represents a PP whose head was elided.

**The fourth parameter** examines the compatibility between the selectional restrictions of the verb under consideration and semantic features belonging to the dependent. The metaphor that may best illustrate this phenomenon is two wheels whose gears may or may not mesh. Incompatibility between the selectional restrictions of a verb and the semantic features of a dependent overrules the archetypical relation. In such instances, the algorithm assigns syntactic and semantic functions which are different from the archetypical ones. The check is performed by using a rather primitive set of semantic features (eight).

**A fifth parameter** intervenes when past participles are considered. This is ergativity. An inaccusative verb inverts the normal semantic relationship between the participle and its dependents (see appendix 1).

**The generation module** of the algorithm assigns syntactic and semantic functions to the prefix and the head. This enables the production of an unfolded equivalent of the CP. Two aspects deserve mentioning. First, the tense of the unfolded present participle is, by default, set to the present. This is true as far as a context-free PP is concerned. But it would not hold if the CP to be unfolded was embedded in a sentence in the past tense. The tense of the unfolded past participle is set to present or present perfect depending on the verb's aktionsart (imperfective verbs deriving present tense and perfective verbs present perfect).

Second, the paraphrase is by default set to active voice. Hence, subject and object are linearized in an order that does not necessarily coincide with the order that would prevail if the CP occurred in a context-sensitive environment. This means that elided dependents are indicated formally by X1, X2,...

#### **4. Database implementering**

For the purposes of systematic analysis, categorization and hypothesis formation, a database for the collected CP examples has been constructed. The manual data-handling consists of 3 sections:

1. Actual reading
2. Lexeme
3. Paraphrase

Section 1 contains information concerning the form and function of the particular CP. Section 2 describes the valency and selectional restrictions of the related infinitive from an LFG perspective. In section 3, the CP construction is analyzed: syntactic structures and semantic relationships are formulated in terms of LFG. Finally, the CP example is transcribed into a semantically equivalent construction with a finite VP, e.g.

- I. *elproducerende vindmøller* → *vindmøller*, *der producerer el*  
(electricity-producing windmills → windmills, that produce electricity)

The paraphrase resolves any potential ambiguous relationships between the participle form, its prefix and the NP head-noun. From the paraphrase, it is relatively less problematic to generate translation-equivalents in Romance languages:

- II. *vindmøller*, *der producerer el* →  
*des éoliennes / qui produisent de l'électricité / productrices d'électricité/*

## 5. Architecture of the CP-UDOG algorithm

In order to test the potential of the aforementioned rule sets for analysis of CP constructions, the CP-UDOG algorithm, which further formalizes the linguistic analysis and the associated terminology, was implemented. Due to concerns for transparency of the program code and compatibility with other language processing systems, the algorithm was implemented in the programming language Pascal.

The static structure of the program consists - in a rather simplified description - of dictionary files, a database file, an NP-parser and a module for enhanced analysis with ensuing rewrite rules.

In addition to standard morphological values, nouns are categorized within a coarse-grained semantic hierarchy. The database file - a reduced version of the database mentioned in section 4 - contains mainly information concerning the morphology, valency (selectional restrictions), aktionsart and ergativity of verbs.

The NP-parser reads the input, identifies the word units, confirms the syntax and segments the CPs. Following the identification, further semantic-syntactic information from the database file is added to the verb form. The information from the NP-parser is passed on to the rewrite module, where the data is further analyzed before the input - non-finite CPs in NPs - is transcribed into finite constructions by means of rewrite rules (see example I, section 4).

Thus, the manual treatment of data in the database and the automatic analysis of the CP-UDOG-algorithm interact on several levels:

1. The database constitutes the setting for inductive hypothesis formations, while the implementation acts as a 'top-down' hypothesis test.
2. Database information is exported directly to the database file of the implementation.
3. The manually produced paraphrases of the database are the measures by which the output of the implementation is judged.



4. The logical consistency of the implementation has contributed towards a more precise meta-terminology, in particular concerning such aspects as valency (selectional restrictions) and aktionsart.

## 6. Tests, maintenance and further perspectives

The implementation of the main structure of the program is concluded, though the refinement of certain rewrite rules is an ongoing process.

As mentioned in section 5, detection of potentially rule based deviant CP types has been facilitated by the application's ability to process large corpora rapidly and consistently. Basically, the method consists of using files of systematically categorized CP examples as test data to achieve a better overview. Therefore, the run-time tests of the program have had the dual function of evaluating existing linguistic hypotheses, while providing data for new analysis theories.

Future tests, ideally, would include a Danish parser and a full-scale dictionary in order to achieve a clearer picture of the efficiency of the algorithm.

Finally, an extension of the CP-UDOG-project will include a description and an analysis of CPs in English and German, with the primary aim of establishing analogous methods for disambiguation of CP constructions in Danish, English and German.

## References

Bresnan, J. 1982. "The Passive in Lexical Theory". *The Mental Representation of Grammatical Relations*, edited by J. Bresnan. Cambridge, Mass: The MIT Press.

Jacobsen, B. L. F. & Kjærsgaard, P. S. 1995. "Adjektiviske deverbaler i dansk". *UDOG-rapport 2*, edited by Maegaard, B. and Pedersen, B. S., pp. 3-26. København: CST.

Kjærsgaard, P.S. 1996. "Danske participialer og valens". *Adjektivernes Valens*, edited by VanDurme, K. , pp. 49-84. Odense, Institut for Sprog og Kommunikation: Odense Universitet.

Pümpel-Mader, M. et al. 1992. *Deutsche Wortbildung. Typen und Tendenzen in der Gegenwartssprache*, V. *Adjektivkomposita und Partizipialbildungen*. Innsbruck. Düsseldorf: Pädagogischer Verlag Schwann.

Stewart, P. 1995. "Brugen af en database til behandling af danske participialer". *Datalingvistisk Forenings 5. Årsmøde*, pp. 53-66. Odense, Institut for Sprog og Kommunikation: Odense Universitet.

Analysis					Generation				
Participle type	Valency	Prefix wordclass	Semantic Selectional Restrictions		Head	Prefix	Participle	X1	X2
			Prefix	Head					
PPI	1	+n		+S	S	ADV	Vpres		
			+S	-S	ADV	S	V:pres		
		-n			S	ADV	Vpres		
	2	+n	+O	+S	S	O	Vpres		
			-O	+S	S	ADV	Vpres	O	
		-n			S	ADV	Vpres	O	
	3	+n			S	O	Vpres	prep.O	
		-n			S	ADV	Vpres	O	prep.O

Analysis						Generation					
Participle type	Valency	Prefix wordclass	Sem. Select. Restrict.		Ergativity	Head	Prefix	Participle Aktionsart	X1	X2	
			Prefix	Head							
PPII	1	+n				S	ADV	pres. / pres. perf.			
		-n				S	ADV	pres. / pres. perf.			
	2	+n	+S				O	S	pres. / pres. perf.		
			-S	+O	inergative		O	ADV	pres. / pres. perf.	S	
				+S	inaccusative		S	O	pres. / pres. perf.		
			-S, +O	-S, -O			ADV	O	pres. / pres. perf.	S	
	-n					O	ADV	pres. / pres. perf.	S		
	3	+n					O	ADV	pres. / pres. perf.	S	
		-n					O	ADV	pres. / pres. perf.	S	prep.O

Appendix 1.

# Evaluation of the Syntactic Parsing Performed by the ENGCG Parser

Klas Prytz

Department of Linguistics  
Uppsala University  
Box 513, S-751 20 Uppsala, Sweden  
Klas.Prytz@ling.uu.se

## 1. Introduction

This paper presents an evaluation of the syntactic parsing performed by the ENGLISH Constraint Grammar parser (ENGCG). The parser is described in '*Constraint Grammar; A Language-Independent System for Parsing Unrestricted Text*' (Karlsson, Voutilainen, Heikkilä, Anttila, 1995) and a version of it is available on Internet. Although ENGCG is both a morphological and a syntactic parser, this paper deals exclusively with the syntactic component. A small corpus has been compiled from five short texts from different text categories and has been sent to the parser. The returned analysis has been checked and the performance of the parser has then been evaluated.

The following section provides an introductory presentation of the ENGCG parser, the method used in parsing and the representation of the analysis. The third section describes the method used in the evaluation while the fourth presents the result of the evaluation. The fifth section contains some comments about errors found in the analysis, the sixth points to some factors that may affect the performance and the last section consists of some conclusions.

## 2. English constraint grammar

Constraint Grammar (CG) is a language independent framework for morphological and syntactic parsing of natural language(s) (Karlsson, 1995, p. v). Any language should be possible to parse within this framework provided there is a correct description of that language. ENGCG is an implementation of CG for the parsing of English. Words are assigned possible analyses by tag assigning modules and the output from these modules are then disambiguated by the application of constraining rules. A word may be assigned several morphological readings as well as syntactic analyses. The ambiguous output from the tag assignment may look as follows:

```
"<tended>"  
  "tend" <SV> <SVO> V PAST VFIN @+FMAINV  
  "tend" <SV> <SVO> PCP2 @APP @-FMAINV
```

The first line contains the word form found in the text. The following lines contain two different morphological analyses. The word surrounded by quotation marks is the lexical form of the word. Then follows a set of tags expressing the analyses. The tags surrounded by angular brackets, here, indicate that both readings of the word are either intransitive or monotransitive.

The rest of the morphological and syntactic analyses is represented by the tags that follow. V PAST VFIN indicates that the word is a finite verb in past tense, while PCP2 on the third line indicates that the word is a participle. As far as the morphological analysis is concerned, one line expresses one unambiguous analysis. A single line may, however, contain more than one syntactic analysis. Syntactic analyses are represented by tags preceded by @. @FMAINV indicates that the word is a finite main verb. In the example above, the second reading is ambiguous between an analysis as an apposition and a non-finite main verb.

In order to solve ambiguities Constraint Grammar rules are applied to this analysis. All Constraint Grammar rules have the same basic construction. A rule consists of a target description that singles out a particular analysis, a set of context conditions that has to be satisfied for the rule to apply, and an action that is undertaken if the rule is applied. As a result of this action the current analysis may be discarded or singled out as the correct one and all other analyses discarded. The morphological constraints discard complete lines together with syntactic tags. The remaining syntactic tags are then subject to the application of syntactic constraints. In this way a run through the parser may reduce the number of analyses. Consecutive runs through the parser may reduce this number further since discarding certain analyses may create context that allows the application of other rules. The last reading is always retained. In this way the output from the parser will always contain at least one analysis for each word.

There are two different set of rules used by the parser. The first set consists of 'safe' rules, that are supposed to always yield a correct result, while the second set consists of heuristic rules that may result in erroneous analyses. The user has the option of choosing the application of both sets of rules or the application of only the non-heuristic set.

### 3. Method

For the evaluation of the syntactic parsing performed by the ENGCG parser a small corpus was compiled consisting of five texts from different categories, all in all 2628 words, representing different genres (see table below).

Category	Tokens	Average sentence length
Learned	560	43.1
Fiction	488	40.7
Biography	514	39.5
Governmental	562	62.4*
Press	504	14.5
TOTAL	2628	32.0

Table 1. Division of text sample

- In order to obtain text pieces of reasonable length the last 355 words in this text sample have been divided into four clauses using semicolon as a delimiter.

The texts were divided into 'chunks' of about three hundred words that were sent to the on-line ENGLISH Constraint Grammar parser accessible on Internet (<http://www.lingsoft.fi/cgi-pub/engcg>). The parser gives the option of analysing the text with or without the use of heuristics. Both these options have been used for this study. The same text sample has, thus, been analysed twice, once with the use of the heuristic set of rules in addition to the ordinary set and once without.

The output from the parser has been manually checked. The number of words in each sentence has been counted as well as the number of syntactic units. *Words* here refers to word tokens and letters but not to punctuation marks. Syntactic units are single words or groups of words entered on single lines (see example below).

- (1). a. "<press\_conference>"  
 b. "<as=if>"

Both these sequences of words are fitted together on a single line by the Constraint Grammar Preprocessor and are considered as one unit but two words. The '\_' character is used for compounds while phrasal idioms and multi-word prepositions are fitted together with the character '='.

Different morphological analyses are represented as separate lines; readings, along with syntactic analyses (see example below).

- (2). "<\*you>"  
 "you" <\*> <NonMod> PRON PERS ACC SG2/PL2 @OBJ  
 "you" <\*> <NonMod> PRON PERS NOM SG2/PL2 @SUBJ

In this case the word *you* has two morphological readings with one syntactic analysis each.

When checking the analysis, tags that are deemed to be incorrect have been marked and correct ones, if missing, have been added to the analysis. All these tags have been counted yielding figures for returned tags, correct tags, and intended correct tags for each sentence.

In some cases a word or unit has been marked with more than one tag that was deemed to be correct.

- (3). "<the>"  
 "the" <Def> DET CENTRAL ART SG/PL @DN>  
 "<\*m25>"  
 "m25" <\*> ABBR NOM SG @<P  
 "<in>"  
 "in" PREP @<NOM @ADVL  
 "<\*kent>"  
 "kent" <\*> <Proper> N NOM SG @<P  
 "<\$.>"

In the example above, the preposition *in* is tagged both with an @ADVL tag, marking it as a independent adverbial, and a @<NOM tag, marking it as a post-modifying preposition. Both these analyses are possible and this unit will for this reason yield two correct tags and two intended correct tags. In other cases different morphological readings may yield the same correct syntactic analysis.

- (4). "<more=than>"  
 "more=than" ADV @ADVL  
 "more=than" <CompPP> PREP @ADVL

In this example the ADV reading is incorrect and the PREP reading correct. Both readings are, however, marked with the correct syntactic tag (@ADVL). In this case both analyses are deemed correct and this unit is, thus, assigned two correct tags. This may be considered a kind of overgeneration of syntactic tags. It does not, however, affect the result in a negative way. In most cases there is only one intended correct tag for each syntactic unit.

#### 4. Result

Since the analysis may yield more than one tag, correct as well as incorrect, for one syntactic unit, it is reasonable to divide the result into two parts: recall and precision. Recall is calculated by the formula: returned correct tags divided by intended correct tags, while precision is calculated by the formula: returned correct tags divided by all returned tags. For example: the sentence '*All in all, Berlin and Kay appear to have dealt a severe blow to the notion of linguistic relativism.*' consists of 19 words. *All in all* is considered an idiom and all three words are fitted together on a single line. The sentence, thus, has 17 units. No unit is morphologically ambiguous so the number of readings is also 17. The preposition *to* before the last noun phrase is syntactically ambiguous between @<NOM @ADVL yielding a number of 18 syntactic tags in total. The @<NOM analysis for this reading is considered incorrect. This gives a number of 17 correct tags equalling the number of intended correct tags. The recall for this sentence will be 100% (17 correct tags divided by 17 intended correct tags. The precision is about 94.4% (17 correct tags divided by 18 returned tags)

The recall and precision for each test text is presented in table 2 below.

		Learned	Fiction	Biogr.	Govern.	Press	Total
No Heur.	Recall	93.9	97.4	96.0	96.4	99.4	96.5
	Precision	70.1	65.7	71.3	68.4	78.1	70.5
Heur.	Recall	93.4	96.4	95.7	96.4	98.8	96.0
	Precision	74.2	67.1	73.3	68.5	78.1	72.0

Table 2. Recall and precision for the text sample (%)

The recall is higher for the non-heuristic parsing than for the heuristic parsing. This is true in all categories except Governmental. In that category, recall shows identical figures for the heuristic and the non-heuristic analyses. The precision is lower for the non-heuristic analysis than for the heuristic. This is true in all categories except press. The immediate conclusion to be drawn from

this is that the optional addition of heuristics lowers the recall but increases the precision. The heuristic rules discard many erroneous tags along with some of the correct ones.

## 5. Comments

Verb sequences are often correctly analysed and there is only a small amount of overgeneration.

- (5). "<will>"  
"will" V AUXMOD VFIN @+FAUXV  
"<be>"  
"be" <SV> <SVC/N> <SVC/A> V INF @-FAUXV  
"<returned>"  
"return" <SVOC/A> <SV> <SVO> PCP2 @-FMAINV

Adverbial phrases are often correctly tagged. Modifiers in noun phrases, such as determiners, pre- and post-modifying nouns and adverbials mostly get correct analyses.

- (6). "<his>"  
"he" PRON PERS MASC GEN SG3 @GN>  
"<late>"  
"late" A ABS @AN>  
"<years>"  
"year" N NOM PL @<P

Nominal heads cause a considerable amount of overgeneration. The parser is often unable to relate heads of nominal phrases to the rest of the sentence.

- (7). "<term>"  
"term" N NOM SG @SUBJ @OBJ @<P

## 6. The Effect of Sentence Length and Morphological Analysis on the Performance of the Parser

Recall and precision do not seem to be affected at all by sentence length when single sentences are compared. Even the extremely long sentences in the test text have quite high recall and precision. If, however, the average recall and precision for each text category are compared to the average sentence length of corresponding category, sentence length affect both recall and precision. Text types with longer average sentence length have lower recall and precision than text types with shorter. This may, however, be caused by the difference in text types rather than by sentence length itself. A text type with longer sentences may be more complicated and more difficult to analyse. To determine to what extent this is the case is, unfortunately, outside the scope of this study.

The number of morphological analyses also affects recall and precision. If the number of morphological readings per unit is compared to the precision, there is a visible correlation in that a high number of morphological readings per unit gives a low figure for precision. Overgeneration of morphological analyses seems to feed overgeneration of syntactic labels. The

correlation between the number of morphological readings per word and recall seems to be reversed. A high number of morphological readings gives a high recall, indicating that a high number of morphological readings increases the chances of correct syntactic analyses.

## 7. Conclusions

This study has shown that the recall for the syntactic parsing seems to lie around 93-99%, higher for the non-heuristic parsing and lower for the heuristic. The precision is considerably lower and falls between 66-78%, higher for the heuristic parsing and lower for the non-heuristic. There seems to be a balance between recall and precision. The addition of heuristics in the parsing lowers the recall but makes the precision higher as the use of heuristic rules discards both erroneous and correct analyses. This connection between recall and precision seems to be quite constant throughout all text categories.

There seems to be no connection between sentence length and the performance of the parser in terms of recall and precision. Text types with longer average sentence length, however, have both lower recall and precision than text types with shorter average sentence length.

Morphological ambiguities seem to feed syntactic ambiguity. This is not at all surprising since every reading of a single syntactic unit will receive at least one syntactic label. Overgeneration of morphological readings feeds overgeneration of syntactic analyses since every morphological reading more than necessary generates at least one syntactic tag more than necessary. The overgeneration of syntactic tags does, however, not always affect recall and precision in a negative way. The overgeneration of syntactic tags seems to be the reason why the connection between recall and morphological overgeneration seems to be reversed. If a word has many syntactic tags attached to it, the chance of finding the correct one among them is greater than if only one syntactic tag is present.

The test text has been compiled from following sources:

### Learned

Geoffrey Sampson, *Schools of Linguistics*, 1980, Hutchinson, London, Melbourne, Sydney, Auckland, Johannesburg, pp. 98-100.

### Fiction

Margaret Drabble, *The Needle's Eye*, 1972, Penguin Books, pp. 190-191.

### Biography

T. C. Duncan Eaves and Ben D. Kimpel, *Samuel Richardson, A Biography*, 1971, pp 11-12.

### Governmental

Document authorised by The Council of The Stock Exchange under section 154 (1) (b) of the Financial Service Act 1986, Terms and Conditions of Application.

### Press

John, Steele, Friday, May 24, 1996, *The Daily Telegraph*, pp. 1-2.



## **References**

Karlsson, Fred; Voutilainen, Atro; Heikkilä, Juha; Anttila, Arto. eds. (1995). *Constraint Grammar; A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin, New York.

# CATCH: A Program for Developing World Wide Web CALL Material

**Erik F. Tjong Kim Sang**

Department of Linguistics  
University of Uppsala  
erik.tjong@ling.uu.se

## 1 Introduction

Computer-Assisted Language Learning, in short CALL, is the research area which deals with the development and evaluation of computer software and hardware that is used in a language learning environment. The recently published book *Computer-Assisted Language Learning – Context and Conceptualization* by Michael Levy focuses on evaluating CALL systems and providing a high-level description for them. We are interested in obtaining answers to more basic questions: What educational methods can be used in CALL material? And what software can be used for developing CALL material?

This paper provides answers to these two questions. We will start with describing a basic collection of educational methods that are usable in CALL material. After that we will introduce our program CATCH<sup>1</sup> which supports the development of CALL lessons that make use of these educational methods.

We have chosen the World Wide Web (WWW) as the target platform for our CALL material. The advantage of developing CALL material for WWW is that it will become accessible and usable from almost all operating systems with standard software that is available on most computers these days. This solves portability problems and opens the possibility for using the CALL material longer than until the next software or hardware update<sup>2</sup>.

## 2 Educational methods in CALL

In this section we will present a basic collection of educational methods that can be used in CALL material. Presenting a complete collection is impossible and therefore we will concentrate on the methods that we consider most important. We will start with describing exercise formats and continue with help and feedback facilities. The section will be concluded with a general part in which some other educational methods will be described briefly.

### 2.1 Exercises

The ability of CALL software to check the language level of students and subsequently determine the course material they should be subjected to gives CALL an advantage over language learning by using books or audio-visual methods. Checking the language level of students takes place in an exercise environment. Two advantages of exercises with automatic

---

<sup>1</sup> The program CATCH is available at <http://stp.ling.uu.se/~erikt/catch/>

<sup>2</sup> Reusability of CALL material is a big problem according to [Lev97]. CALL material is often discarded at the next software or hardware update.

answer checking are the possibility for testing students from one group on different topics simultaneously and the opportunity for discovering students which problems immediately without a large investment of time by a human teacher [Sci95]. It is hard to overestimate the importance of exercises for CALL material.

Exercises can be characterized by four features:

**Exercise features:**

Purpose: practice and test.

Task: translation, question answering and gap filling.

Format: text, speech and images.

Input format: text, speech and multiple-choice answers.

In our view exercises can be used for two purposes. One is for making the student practice with the material. In this case the performances of the students on the exercise do not have to be registered. Students may change their answers as many times as they want and attempt to use the feedback provided by the CALL software to get all answers right. It is also possible for using an exercise for determining the level of a student. In that case the exercise should be regarded as a test. The results of the students on the test exercises are important for the software and the teacher. The answers should be registered and the students should not be allowed to modify them.

There are three basic exercise tasks. The first one is translation. In a translation exercise the students have to translate words or phrases from one language to another. The second task is question answering. Here students have to answer questions, for example about a text they have read. The third exercise task is gap filling. Here students are presented with a text in which words or phrases have been removed. Their task is to fill in the deleted material.

There are three possible exercise formats. The first one is the most common one: presenting an exercise as text. It is also possible to present an exercise as spoken material. This is necessary for testing the listening capabilities of the students. However one should be aware of the fact that development, storage and retrieval of speech samples requires much more time and computer resources than doing the same with written material. The same is true for the third exercise format: images. Exercises can, of course, also make use of a combination of these three formats.

There are three formats of answer response from students. The most simple format is the multiple-choice answer. For every question the students are provided with a number of possible answers and they have to choose the correct one. This makes dealing with the answers much easier for the CALL software since every possible answer is known by the developer of the material. If the student is allowed to respond to a question by inputting text then it is not possible to foresee every possible answer. Recognizing errors in answers becomes harder especially when the answer consists of a sequence of words rather than a single word. The third input format, speech, is even more challenging for the CALL developer than text input. Beside the answer handling problem, which is equally difficult as for text input, it also includes technical problems dealing with the recognition of the texts uttered by different students.

Developers of CALL material are interested in finding out which exercises will help increasing the language level of the students in the best way. A.G. Sciarone has provided some interesting answers to this question for the topic of text understanding [Sci95]. The language level of second language learning students can be increased by making them spend more time on reading written texts and listening to spoken material. Students will spend more time on these tasks if they perform tests with obligatory result levels regularly. Dictation exercises are the best method for testing listening capabilities and gap filling is the best exercise task for testing text understanding [Sci95]<sup>3</sup>.

## 2.2 Feedback and help

CALL software reacts to input of students by providing feedback. In order for the feedback to be useful the CALL system should be able to recognize and classify the input. This is one of the most challenging tasks for developers of CALL material. Too often language teachers that evaluate CALL programs complain that a student has given a correct answer to a question but that the computer has failed to recognize it [vdL97]. The problem here is that computer software is unable to understand human language.

A consideration of the position of CALL in language education is necessary here. CALL systems are not being used for replacing human teachers but as an extra tool for aiding their work. This means that CALL programs do not have to be able to evaluate every possible language learning task<sup>4</sup>. CALL software may concentrate on the tasks that it performs best and use input formats that are relatively easy to check.

If CALL programs allow only simple input of students like multiple-choice answers or single word responses then we can attempt to specify each relevant answer in advance. In that case it is possible for the CALL developer to define useful feedback for these predefined answers. The answer set will benefit from an evaluation of the answers given by the students while using the CALL material [Ake98]. It is therefore desirable that CALL software is able to store the unforeseen answers of students so that they be evaluated by its developers.

Apart from general help facilities about program usage CALL software needs special aids for students. The most important of these is a searchable online dictionary. Language students often encounter unknown words which they have to look up and it is an advantage if they can use the CALL system for doing that as well. A problem is that foreign language students may have different first languages and this means that the dictionary may need more than one source language.

---

<sup>3</sup> Sciarone has compared multiple-choice exercises, question answering exercises and gap filling exercises. His conclusion was that multiple-choice exercises allowed too much guessing, answers on questions are too hard to check for the available computational linguistics technology while gap filling exercises provide both good testing facilities and allow checking answers with the available technology [Sci95].

<sup>4</sup> An example of a language learning evaluation task which seems impossible for a CALL system with the present technology is testing the communicational capabilities of language learning students. This task requires a human teacher.

## 2.3 Other methods

With exercises, feedback and help facilities our overview of educational methods that are usable in CALL systems is by no means complete. Another important method is narrative text which can be either written or spoken. However texts need to be accompanied by exercises to make sure that the students have read and understood them. Equally important are explanations about grammatical features of the language that is to be learned. These can be incorporated in a general CALL lesson but they are at least as useful as a part of the feedback provided to the students. A third educational method which is offered by some commercial CALL programs are language games. We regard these as free-format practice exercises.

## 3 CATCH

In this section we describe our authorware program CATCH. We will start with a general description of the software. After this we will present the different parts of the CALL lesson description language and give an example exercise definition. We will conclude the section with experiences of CATCH users.

### 3.1 Background

The World Wide Web (WWW) is an interesting medium for CALL material because it solves the portability and reusability problems that comes with many of the current CALL applications. However there are some problems with this medium as well:

1. a lot of technical knowledge about document encoding and programming is necessary in order to be able to develop a working WWW CALL application<sup>5</sup>.
2. a large amount of encoding information is necessary for even the simplest exercise<sup>6</sup> and
3. because students might have a slow network connection to the material CALL developers are not able to apply freely all techniques they would like to use (they should be careful with using many images, animations and sound files).

These problems hinder the development of CALL WWW material. There is not much we can do about the third problem. However the first and the second problem are solvable. Our proposed solution for these two problems is to create a CALL development language which enables language teachers to define information necessary for CALL applications without having to worry about the peculiarities of WWW document encoding. An external program will be used for converting the specifications to a WWW document. Our program CATCH works exactly like that.

CATCH is a document conversion program written in the programming language Perl [Wal96]. It converts documents in a CALL encoding language to documents in the encoding language of WWW: HTML. The CALL encoding language consists of markup tags which define the type of

---

<sup>5</sup> For example, [Gol96] reports that while many departments at the University of British Columbia were interested using WWW based course material in the early nineties both quantity and quality of this material was highest at the Computer Science Department.

<sup>6</sup>[Ahl98] reports that for a WWW file with an eleven-question gap exercise 89% of the file consisted of encoding tags and software.

text parts. The tags used in this language are SGML tags: they start with the < character and end with >. There are two variants of each markup tag: the <xxx> tag which opens context xxx and the </xxx> tag which closes context xxx. Examples of these tags are the start of lesson tag <lesson> and the end of test exercise tag </exercise>.

The users of CATCH can specify CALL material without having to worry about technical details like answer checking and WWW document encoding. The CALL format documents are both smaller and easier to understand than the corresponding HTML documents. It is possible to use CATCH without having any knowledge of HTML. However we have foreseen that CALL developers will want to use HTML features that are not included in the standard output format of CATCH. For that reason CATCH allows HTML tags to be present in its input file. The program will accept these tags and copy them to its output file. In this way it is possible for CATCH users to adapt the format of their CALL lessons to their own wishes.

### **3.2 Parts of the program**

The present version of CATCH (2.0) recognizes fifteen CALL material encoding tags. The choice of these tags has been based on the theoretical outline presented in section 2. The largest part of the tags deals with exercises (9 tags). Apart from that there are some general structure tags (3) and some tags that deal with other educational methods (3). The general structure tags deal with defining the CALL lesson context, the headings and menus with pointers to the different parts in the CALL lesson. CATCH offers three other educational methods for the CALL material: texts, word lists and grammar explanations.

CATCH offers two exercise types: practice exercises and test exercises. Students can work with the practice exercises on a stand-alone computer. However for working with the test exercises they will need a computer with a network connection because answer checking will usually be performed by a central computer. Both exercises allow two input formats: multiple-choice input and text input. The decision about what exercise format and what exercise task to use has been left to the developers of the CALL material.

### **3.3 User experiences**

The first version of CATCH<sup>7</sup> has been tested by second-year Language Engineering students at Uppsala University in a one-point CALL introduction course. The students have used the program for developing CALL material for Swedish and French. They were positive about the software. The students had some recommendations for additional functionality which have been used for extending the previous version of the program. The CALL material developed by the students has subsequently been published on WWW<sup>8</sup>. We have received quite some enthusiastic reactions on these lessons from its users and requests for developing more.

Currently the program is being used in a five-point CALL course for developing material for Spanish and Russian. In this course we are cooperating with a professional language teacher. We have received many interesting recommendations from both her [Ake98] and the students

---

<sup>7</sup> The first version of CATCH was called call2html.

<sup>8</sup> The CALL material is available at <http://stp.ling.uu.se/call/>

[Ahl98]. Some of their recommendations for additional functionality, like practice exercises, have already been added to CATCH.

#### 4 Concluding remarks

The World Wide Web is an excellent medium for the development of CALL material. However the amount of technical knowledge and the amount of HTML coding that are required for creating CALL lessons may prevent language teachers from using WWW for developing their own teaching material. In this paper we have described a general tool called CATCH<sup>9</sup> which can be used to overcome this barrier. CATCH converts descriptions of CALL material written in an easy to understand encoding format to HTML files. By working in this way it is possible for people without technical knowledge to develop CALL material in a faster way than by creating HTML files from scratch.

#### References

- [Ahl98] Viktoria Ahlbom. *Interactive Grammar Exercises in Russian*. Department of Linguistics, Uppsala University. (In Swedish, to appear).
- [Ake98] Kariné Åkerman Sarkisian. *A Pilot Project for Using CALL for Russian*. Department of Slavic Languages, Uppsala University. (In Swedish, to appear).
- [GSS96] Murray W. Goldberg, Sasan Salari and Paul Swoboda. World Wide Web Course Tool: An Environment for Building WWW-based Courses. *Computer Networks and ISDN Systems*, 28, 1996. Also available on <http://homebrew.cs.ubc.ca/webct/papers/p29/>
- [Lev97] Michael Levy. *Computer-Assisted Language Learning*. Claredon Press - Oxford, 1997. ISBN 0-19-823631-X.
- [Sci95] A.G. Sciarone. *The Computer in Second Language Learning*. Boom Amsterdam, 1995. ISBN 90-5352-205-0. (In Dutch).
- [vdL97] Hanny van der Linden. Personal communication, 1997.
- [WCS96] Larry Wall, Tom Christiansen and Randal L. Schwartz. *Programming perl*. O'Reilly & Associates, Inc., 1996. ISBN 1-56592-149-6.

---

<sup>9</sup> The program CATCH is available at <http://stp.ling.uu.se/~erikt/catch/>. It runs on the platforms MSDOS, Windows and Unix provided that the programming language Perl is available.

# Assembling a Balanced Corpus from the Internet

**Johan Dewe**, Telia Research,  
**Jussi Karlgren**, SICS, and  
**Ivan Bretan**, Telia Research

Address for correspondence:

Jussi Karlgren, SICS, Box 1263, 164 29 Kista, Sweden

Fax: + 46 8 751 72 30

Jussi.Karlgren@sics.se

## Balanced Corpora for Textual Research

For empirically oriented textual research it is crucial to have materials available for extraction of statistics, training probabilistic algorithms, and testing hypotheses about language and language processing in general. In recent years, the awareness that text is not just text, but that texts comes in several forms, has spread from more theoretical and literary subfields of linguistics to the more practically oriented information retrieval and natural language processing fields. As a consequence, several test collections available for research explicitly attempt to cover many or most well-established textual *genres*, or *functional styles* in well-balanced proportions (Francis and Kucera, 1982; Källgren, 1990).

The creation of such a collection is a complex matter in several respects. Our research area is to build retrieval tools for the Internet, and thus, for our purposes, the choice of genres to include is one of the more central problems: there is no well-established genre palette for Internet materials. To find materials to experiment with, we need to create them in a form suitable for our purposes. This is a double edged problem, involving both vaguely expressed user expectations and establishing categories using large numbers of features which taken singly have low predictive and explanatory power. This paper gives an outline of the methodology we use for determining which genres to include.

## Stylistic Variation and Genre

Texts exhibit considerable variation. While the variation in topic or content is quite obvious and the basis for most categorization enterprises in information retrieval research variation in style is as noticeable, and forms a second basis for categorization: poetry, prose, non-fiction, reference materials, and so forth are all stylistic categories or genres.

Stylistic variation shows through *stylistic items*: observable choices of linguistic items. Stylistic items can be observed on any level of linguistic abstraction: lexical, for the choice between words of similar meaning but different connotations; syntactic, for the choice between equivalent constructions with different communicative import; textual, for decisions of textual organization.

Each stylistic item is of little import, but taken together they are indicative of systematic differences. A set of documents with a perceived consistent tendency to make the same stylistic



choices is called a genre or, specifically, if it has an established communicative function, a functional style (see e.g. Enkvist, 1973; Vachek, 1975).

Stylistic variation between genres or language varieties can be detected reliably using a large battery of quite simple stylistic items such as pronoun counts or relative frequencies of certain types of constructions such as agentless passives (Biber, 1988, 1989; Karlgren and Cutting, 1994), utilized for authorship determination by simple calculations of average word length distributions (Mendenhall, 1887), and with some success predictively for information retrieval (Karlgren, 1996; Karlgren and Straszheim, 1997; Stralkowski et al, 1996).

## Establishing Genres

### Method

In previous similar studies, we have used introspective methods: we have established genres mainly based on personal experience (Ben Cheikh and Zackrisson, 1994; Hussain and Tzikas, 1995). Other text collections organized by genre, genre is largely equated with *source*. Texts from some organization are categorized together with texts from similar organizations, without regard for text usage: e.g. journalistic press archives, personal letters, technical documentation. (e.g. Källgren, 1990). For this study, we wished to have a better foundation for our genre palette. Our basic source of knowledge is interviewing users about their perceptions of what types of material they find and interact with online. We collate the impressions and try to define genres that are both reasonably consistent with what users expect and observable and conveniently computable using measures of stylistic variation as outlined in the previous section. Cf. Figure 1, see last page of this paper.

### Questionnaire

The questionnaire in Figure 2 was sent to 648 computer users - students, researchers, and teachers at Stockholm University and the Royal Institute of Technology. We received 7 error messages and 67 responses, which gives a response rate of 10 per cent.

Hi. I need two minutes of your time.  
For my M Sc project I will classify WWW documents by genre.  
What is a genre? A genre is a group of documents with similarities as regards form. Journalistic material, for instance, gives us several examples of genres. We find scientific materials, short stories, news items, advertisements, and so forth. In a larger perspective a newspaper itself is a genre, as compared to crime fiction, parliamentary records, and chat group text.  
Similarly, it should be possible to categorize materials from the WWW in genres. The obvious ones I can figure out myself, but I do not want to constrain myself to a single perspective. So I need your help to gain a wider view:  
\* What genres do you feel you find on the WWW?  
Take a minute to think over the question, and send me a list of the genres that occur to you. All replies are useful to me!  
Thank you for your time,  
/Johan Dewe, d92-jde@nada.kth.se

Figure 2. The genre questionnaire (This is an English translation. The Swedish original can be found at <http://www.stacken.kth.se/~dewe/dropjaw/enkat.txt>)

## Compiling the results

```
Science, Entertainment, Information
Here I am, Sales pitches, Serious material
Home pages
Data bases
Guest books
Comics
Pornography
FAQs
Search pages
Corporate info
Product info
Reference materials
My immediate reaction is that genres from general society
will be found on
the WWW as well. We get stuck in old conventions. ... e.g. e-
mail
conventions follow paper letter conventions. I would start by
using genres
from ordinary life and see if they are applicable to WWW.
Home pages
Public info
Non-government organization info
Search info
Corporate info
Informative advertisements
Non-informative advertisements
Research materials
Games and pornography
News
Economic info
News
Tourism
Sports
Games
Adult pages
Science
Culture
Language
Media
Public documents, Internal documents,
Personal documents
Information
"Check out what a flashy page I can code"
"I guess we have to be on the net too"
```

**Figure 3.** Some translated excerpts from the answers to the questionnaire. (The answers in their entirety can be found at <http://www.stacken.kth.se/~dewe/dropjaw/enkatsvar.txt>).

The answers ranged from very short to extensive discussions - some examples are shown in Figure 3. It was very clear to us from that most readers conflated genre and form on the one hand with content and topic on the other: "tourism", "sports", "games", "adult pages". This is not surprising. Genre and topic are not independent dimensions of variation, and a typical library categorization reflects both dimensions simultaneously. Several respondents did give examples of more cleanly form-oriented genres as well: "home pages", "data bases", "FAQs", "search pages", "reference materials". Some respondents gave explicit references to *paper genres* - one lengthy quote is given among the examples in Figure 3. The *intention* of the information provider showed up as a genre formation criterion in several responses: "here I am", "sales pitches", "serious material"; or, as an alternative formulation of the same criterion, the type of author:

"commercial info", "public info", "non-governmental organization info". Some responses explicitly brought up *quality*: "boring home pages" and text *ecology* or intended environment: "public documents", "internal documents", "personal documents".

We have attempted to systematize some of the user perceived distinctions, namely those that are predictable enough to be modeled with simple metrics, in the genre palette shown in Figure 4.

<b>Informal, Private</b> Personal home pages.
<b>Public, commercial</b> Home pages for the general public.
<b>Searchable indices</b> Pages with feed-back: customer dialogue; searchable indexes.
<b>Journalistic materials</b> Press: news, reportage, editorials, reviews, popular reporting, e-zines.
<b>Reports</b> Scientific, legal, and public materials; formal text.
<b>Other running text</b>
<b>FAQs</b>
<b>Link Collections</b>
<b>Other listings and tables</b>
<b>Asynchronous multi-party correspondence</b> Contributions to discussions, requests, comments; Usenet News materials.
<b>Error Messages</b>

Figure 4. The current genre palette.

When trying to assign textual materials to the various categories automatically we expect to find that some genres are not as useful as they may seem at first sight; we will find that some of these categories may have to be adjusted - merged, split, or redefined - as the collection is evaluated using statistical methods. The categories shown in Figure 4 are starting points for research, not final results.

## Finding Samples

We use three methods to collect data from the World Wide Web.

Firstly, we take queries used for the Text Retrieval Conference (Harman, 1996) (TREC queries nos. 251-300; fields "topic" and "description") and run them through Altavista, a search service on the Internet. We use the top ten hits for each query to retrieve about 500 documents.

Secondly, we take sixty queries from Magellan, another search service on the Internet. Magellan provides a "voyeur page" (<http://mckinley.voyeur.com/voyeur.cgi#voyeur?1>) which displays real

user queries in real time. We run the sixty queries through Magellan, and similarly obtain about 600 documents.

Thirdly we use history files from local Netscape users to retrieve about 700 additional documents.

URL source	TREC via Altavista	Magellan Voyeur	History List	Total
01 Informal, Private	11	67	50	128
02 Public, Commercial	23	87	87	197
03 Searchable indices	4	14	55	73
04 Journalistic materials	50	28	16	94
05 Reports	106	5	2	113
06 Other running text	73	49	38	160
07 FAQs	0	4	8	12
08 Link Collections	31	50	67	148
09 Listings, Tables	17	138	70	225
10 Discussions	16	0	8	24
11 Error Messages	55	36	93	184
-----				
Total	386	478	494	1358

Figure 5. The current composition of the corpus.

### Evaluating the choice of genres

To evaluate the genre palette we sent out the list of genres we settled on to the same recipients we originally solicited the genre distinctions from, with a question if they understood what the genres represented and if any obvious genre was missing. We received 102 responses. Most respondents claimed to understand what type of text our genre labels were intended to cover, and while most categories got some comments of one form or another, most comments were caused by our giving too few examples of what the genres were intended to cover. Most comments concerned the category "Interactive pages". Many respondents were annoyed by the fact that the category was not of the same type as the other types. Some respondents objected or did not understand the labels -- e.g. "FAQ" or "Listings, tables" or "Error messages"; many asked for a download page or ftp database category; some wondered about the all-inclusiveness of "Other running text"; several asked for a specific category for "Search engines"; several suggested more content based genres.

Many pointed out that some of the categories were less suitable for search in that they did not imagine themselves ever searching for "Error messages" or "Interactive pages" specifically. Several respondents pointed out that the categories were not mutually exclusive. In summary, the most central objections were either such that would be remedied in an interactive situation where examples are readily available, or requests for more flexible genre assignment.

### Recognizing genres automatically

The genre palette, besides being intuitively understandable, needs to be workable for automatic analysis. We calculate a quite large number of textual features for each individual text and work

them together for a categorization decision using a machine learning algorithm. The pioneering work by Douglas Biber(1988, 1989) on computational corpus-based stylistics has been descriptive rather than predictive, aiming to find distinctions between different registers or varieties of spoken and written language. It has made use of large numbers of stylistic features collected from previous, non-computational work and weighing them together using standard methods from multivariate statistics. We use this work as a basis for ours. Most of Biber's features we use here are rather lexical in nature, for ease of processing: the relative frequency of certain classes of words such as personal pronouns, emphatic expressions, or downtoning expressions, for instance. We add more general textual and genre specific features: relative number of digits, or average word length, for instance (Karlgrén, 1996; Karlgrén and Straszheim, 1997). Others yet are vectored specifically to the Internet material we have been using for experimentation: number of images or number of HREF links in the document, for instance. We normalize the measurements by mean and standard deviation, and combine them -- 40 of them, at present -- into simple if-then categorization rules using C4.5, a non-parametric categorization tool (Quinlan, 1993).

```
If- there are more "because" than average,  
- longer words than average,  
- type-token ratio is above average,  
then  
- the object is of class Textual  
with  
- a certainty of 90.0%.
```

Figure 6. An example classification rule.

We have a few dozen rules to categorize texts into one of the eleven genres defined in the above sections. The genres partition into two major hypercategories: textual (04, 05, 06, 07, 10) and non-textual (01, 02, 03, 08, 09, 11); each of them in turn splits to one of five or six sub-categories. These splits are of varying quality: the first does quite well, something like a ninety per cent success rate, while the subsplits make the wrong choice somewhere between once in three or four times. With additional features and a better defined genre palette results will improve. However, to get really useful results the categorization should not be exclusive. Every object should potentially be of several genres.

## Conclusions

Internet users have a vague sense of genres among the documents they retrieve and read. The impressions users have of genre can be elicited and to some extent formalized enough for genre collection. The names of genres should be judiciously chosen to be on an appropriate level of abstraction so that mismatches will not faze readers.

## References

1. Douglas Biber. 1988. *Variation across speech and writing*. Cambridge University Press.
2. Douglas Biber. 1989. "A typology of English texts", *Linguistics*, 27:3-43.
3. Naoufel Ben Cheikh and Magnus Zackrisson. 1994. "Genrekategorisering av text för filtrering av elektroniska meddelanden" (Genre Classification of Texts for Filtering of Electronic Messages) Stockholm University Bachelor's thesis in Computer and Systems Sciences, Stockholm University.
4. Nils Erik Enkvist. 1973. *Linguistic Stylistics*. The Hague: Mouton.
5. Donna Harman (ed.). 1996. *The Fourth Text REtrieval Conference (TREC-4)*. National Institute of Standards Special Publication 500-236. Washington.
6. Fahima Polly Hussain and Ioannis Tzikas. 1995. "Ordstatistisk kategorisering av text för filtrering av elektroniska meddelanden" (Genre Classification of Texts by Word Occurrence Statistics for Filtering of Electronic Messages) Stockholm University Bachelor's thesis in Computer and Systems Sciences, Stockholm University.
7. Jussi Karlgren. 1996. "Stylistic Variation in an Information Retrieval Experiment" In *Proceedings NeMLaP 2*, Bilkent, September 1996. Ankara: Bilkent University. (In the Computation and Language E-Print Archive: cmp-lg/9608003).
8. Jussi Karlgren and Douglass Cutting. 1994. "Recognizing Text Genres with Simple Metrics Using Discriminant Analysis", *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto. (In the Computation and Language E-Print Archive: cmp-lg/9410008).
9. Jussi Karlgren and Troy Straszheim. 1997. "Visualizing Stylistic Variation." In the *Proceedings of the 30th HICSS*, Maui.
10. W. N. Francis and F. Kucera. 1982. *Frequency Analysis of English Usage*, Houghton Mifflin.
11. Gunnel Källgren. 1990. *The First Million is Hardest to Get: Corpus Tagging*. *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)* Hans Karlgren (ed.), Helsinki.
12. T.C. Mendenhall. 1887. "The Characteristic Curves of Composition." *Science* 9: 237-49.
13. J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann.
14. Tomek Strzalkowski, Louise Guthrie, Jussi Karlgren, Jim Leistensnider, Fang Lin, Jose Perez-Carballo, Troy Straszheim, Jin Wang, Jon Wilding. 1996. "Natural Language Information

Retrieval: TREC-5 Report" Proceedings of The Fifth Text REtrieval Conference (TREC-5).  
Donna Harman (ed.). National Institute of Standards Special Publication. Washington.

15. Josef Vachek. 1975. "Some remarks on functional dialects of standard languages". In  
Styleand Text - Studies presented to Nils Erik Enkvist. Håkan Ringbom. (ed.) Stockholm:  
Skriptor and Turku: Åbo Akademi.

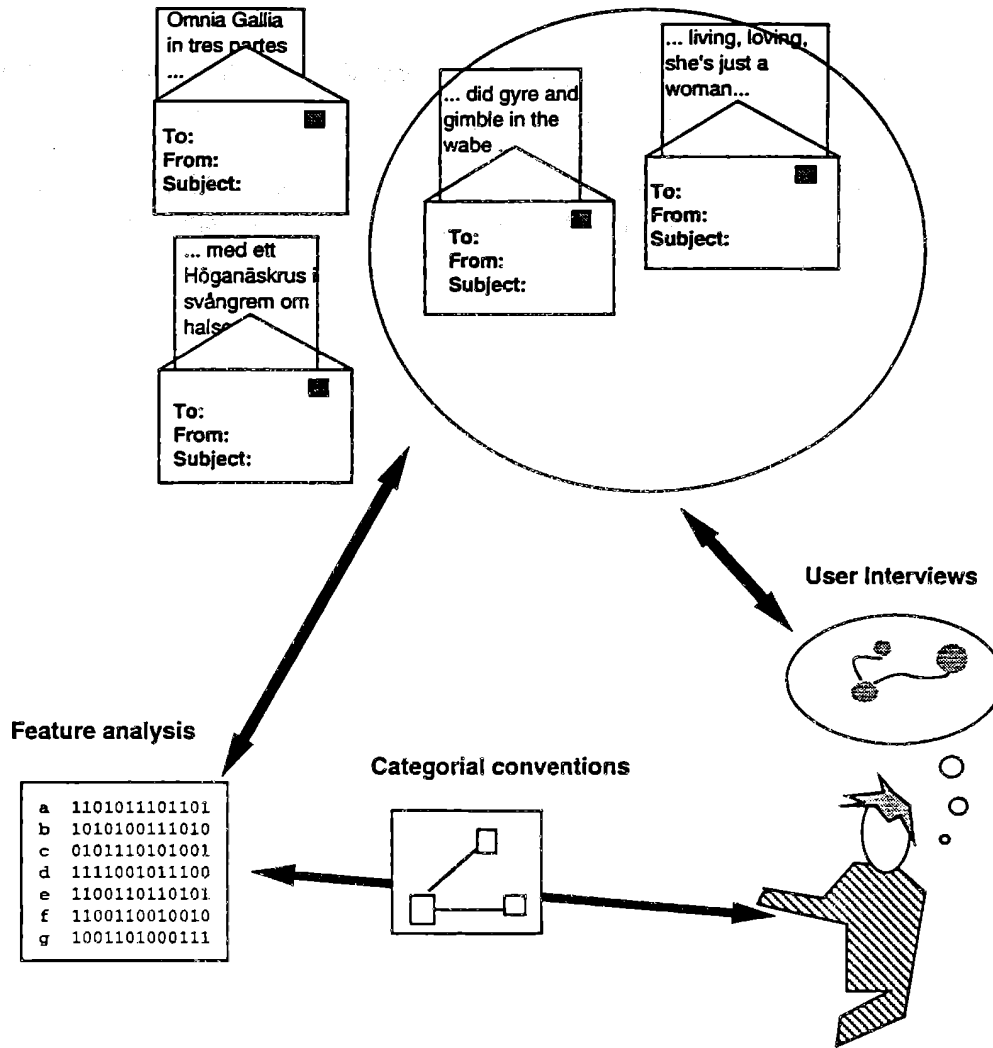


Figure 1. A snapshot of the methodology shows the interplay between vaguely expressed user expectations and observable and conveniently computable categories.



# Peeking Into the Danish Living Room

## Internet access to a large speech corpus

**Peter Juel Henriksen**

Dept. of Danish Dialectology and  
Dept. of General and Applied Linguistics (IAAS)  
Univ. of Copenhagen, Njalsgade 80, DK-2300 S, Denmark  
pjuel@cphling.dk

### 1. Introduction

Our newly opened Internet site offers a view to a >10<sup>6</sup> word corpus of informal Danish conversations. The corpus and the search engine situated at IAAS can now be reached and used as easily as any homepage on the World Wide Web, offering a tool for serious investigations into informal speech.

After a few introductory remarks, we shall present the corpus and the search engine as seen from the user's point of view, following up the presentation with a few example queries. In conclusion, some reflections on the possibilities that the Internet has to offer in utilisation, maintenance, and control of large corpora of semi-confidential data.

The new site may be reached directly at the URL <http://phoneme.cphling.dk/BySoc>, or else via the IAAS home page at <http://www.cphling.dk>

#### 1.1 Why study informal speech?

Ordinary people are common. Most of the day, they talk casually, taking part in informal conversations. So, by far the largest part of the language national product must be plain and simple vernacular. Moreover, most children acquire their mother tongue exposed to this style only, arguably making it the most essential part as well.

Still, the syntax and semantics of informal speech have hardly been studied at all with the exact means of modern formal grammar. Why?

Firstly, because informal speech is *irregular*, seen from a traditional syntactic point of view, making it much more recalcitrant to work with than educated written style. Secondly, because it is hard to get access to large, reliable samples of genuine everyday conversation; the use of bugs is of course unethical, and perhaps illegal, making this abundant source of language rather elusive as a scientific object (which may well seem a bit paradoxical). Finally, because generative linguistics have turned these two obstacles into a claim that informal speech is theoretically uninteresting.

This paper suggests a solution to one of the problems.

## 2. The corpus

The searchable corpus consists of transcriptions of approximately 80 conversations<sup>1</sup>. The participants are all native Danes, most of them born and raised in Nyboder in central Copenhagen. In each case, the linguist who participated in the conversation took great care to make the situation as comfortable and familiar for the informant as possible. Most of the conversations thus took place in private homes, with the linguist as a so-called 'intimate stranger', rather than a TV-style interviewer. The strategy was the following: Prior to his first meeting with the informant, the linguist would try to get acquainted with a neighbour or such, enabling him to present himself to the informant as 'a friend of your friend'. "Normally, people do not confide in a stranger, but some have done so when they have met *an intimate stranger*" (Gregersen & al (91) p.53, following Milroy & al (77)). After a while, many informants reach a relaxed, or even confidential state. This state is of course desirable, since it is exactly then that we get the informal style free of conscious control. For this reason, the sessions recorded were rather long, almost always more than an hour, in some cases more than three hours.

The recordings were transcribed using the convention Dansk Standard 2, as defined by The Danish Language Council (Dansk Sprognævn), a basically orthographic code, enriched with special symbols. In addition to Dansk Standard 2, a so-called *score format* was designed, showing the onset of the utterances relative to each other.

### *Transcription sample*

```
-----  
1> ja ja nej~  
2> nå nå (ler) # nå I startede i de små lejligheder ik' nå  
A>ja vi starter helt forfra (ler)  
-----  
1>  
2># det troede jeg?(ellers)? ja ja  
A> ja fordi der der er dem der er mindre f der er dem der er på halvandet  
-----  
1>på~ atten kvadratmeter ja~ sådan en fik vi tilbudt f  
2> ja det var hvad vi kunne få  
A>varelse ik' ja~ f  
-----
```

<sup>1</sup> The searchable corpus covers about two thirds of the recordings made in connection with the project *Urban Sociolinguistics* (Projekt Bysociolingvistik), carried out in the late eighties by a group of Danish sociolinguists rooted in Institute of Danish Dialectology and Institute of Nordic Philology at the University of Copenhagen. An important purpose of the project was to study Danish vernacular as a function of age, sex, and social class. However, according to leading member Frans Gregersen, the group was very anxious not to have this particular purpose influence the data collection. The interviewers were thus instructed to let the informants choose as well topic as style, not urging them into narration, say (pers.comm.). The background and results of Project Bysociolingvistik are reported in Gregersen & al (91).

*Special symbols:* A> refers to an interviewer; 1>, 2> ,... to informants; ~ stands for 'tøven' (hesitation); ε for 'pause' (pause); # for 'pause med vejrtrækning' (breathing); ( ... ) for 'transskriptørkommentar' (transcriber's comment) ?( ... )? for 'usikker transskription' (transcription uncertain)

The searchable corpus (*BySoc* from here on) consists of 1.4 million words, distributed over some 35,000 word forms. The corpus has recently been thoroughly proof read and converted into a well-defined data structure (cf. Henrichsen (97)). *BySoc* has no grammatical annotations.

### 3. The search engine

The central window in the *BySoc* site consists of a number of frames, among which the most important are the frames **Søgeområde** (Search Area), **Søgeprofil** (Search Profile), and **Resultat** (Result). Using the buttons and tables in these frames, the user can specify

- A search area (a subpart of the corpus)
- A search profile (a string of letters and special symbols)
- An output format (*count*, *list of occurrences*, or *text samples*)

#### 3.1 Search Area

The search domain can be cut along the following dimensions (among others).

*Speaker:* sex, age, role (linguist/informant), social group (working/middle class), name, id  
*Conversation:* number of participants, time slice (e.g. 'exclude the first ten minutes')  
*Style:* language (Danish/foreign), enunciator(*selfquote*), origin(*transcription/comment*)  
*Transcription:* transcriber, confidence (*reliable/uncertain*), version (*main/secondary/encrypted*)

#### 3.2 Search Profile

A simple orthographic word will do as a search profile, while in general any *regular expression* is allowed. Our dialect of RE is as in the programming language *Perl* (Wall (91)), enriched with special symbols for 'space character' ( \_ ) 'letter' ( □ ), 'vowel' ( ½ ), and 'consonant' ( § ). Some special symbols of RE:  $x?$  matches zero or one instances of  $x$ ;  $x\{n,m\}$  matches  $n$ - $m$  instances of  $x$  in sequence ( $m$  may be omitted if there is no upper bound);  $x^+$  is equivalent to  $x\{1,\}$ ;  $x^*$  equivalent to  $x\{0,\}$ .

### Example profiles

Search profile	matches	does not match
Nyboder_Skole	Nyboder Skole	NyboderSkole
[Nn]y.ode(r ns)	Nyboder nymodens ny oder	yboder nyodens ny ode
Ÿybod?er	Nyboder byboer	Ayboder bybodder
(ja)*da	da jajajada	jjjda jaada
π{1,2}y\$od+er	Nyboder blylodder	yboder polyboder Nyboer
π+	Nyboder slikasparges	Nyboder Skole slik- asparges

Some linguistically relevant profiles are demonstrated in sections 0 and 0 below.

### 3.3 Result format

Different queries call for different kinds of output from the search engine. The user can choose output format by clicking one or more buttons in the table 'Resultatets form' in the frame **Resultat**. There are three families of output formats available.

- Count number
- Annotated word list, sorted by frequency or alphabetically
- Text samples, in one or two dimensions (single speaker or full score)

The user may also choose to have the result sent as email.

### 3.4 Query check

While the user is working on his query, his dispositions are being checked for consistency by a background process. This process is controlled by a *JavaScript* program (cf. ref.), embedded in the HTML source code. The process is invisible to the user until he makes an illegal choice – such as excluding men *and* women from the search area, or putting restrictions on 'S3' while his profile contains only *two* bracketed subparts. In such cases, the background process corrects the error, if possible, or otherwise resets the parameter in question to its default setting.

A simple example is the check procedure associated with the query parameter *sex*. The corresponding table in the frame **Søgeområde** (Search area) is generated by the following HTML code.

### HTML sample

```
<a name=sex><i><u>Køn</u></i></a><br>
<select name=sex multiple onchange=notempty(this,2)>
  <option value="F" selected>kvinder
  <option value="M" selected>mænd
</select>
```

Each time the user clicks on one of the options, *male* or *female*, the function `notempty` is called.

### JavaScript sample

```
function notempty(s,e) {
  var flag=false, i=-1;
  while (s.options[++i]) {
    if (s.options[i].selected) {flag=true; break}
  }
  if (!flag) {for (i=0; i<e; i++) {s.options[i].selected = true}}
}
```

If the user happens to deselect *all* options – in the case at hand: both sexes – the background process resets the table to its default setting [*male selected*, *female selected*]. Otherwise, the process stays invisible.

## 3.5 Submitting a query

When the user has finished his query, he clicks the **Start søgning** button (Start search). A JavaScript process then collects all parameter settings, encloses them in a virtual envelope, and submits them to the server. At the server side, a UNIX daemon unpacks the envelope, starts a Perl session, and feeds the search engine with the user input as % ENV variables (for technical details cf. Holtse and Henrichsen (*in preparation*)).

Before the result is returned to the client, certain proper names are encrypted, namely those that do not occur in Retskrivningsordbogen (the Standard Danish Orthographic Dictionary). This means that non-local names like *Grønland* and *Roskilde* are allowed, while *Delfingade* (a local street name) and *Margrete* (a personal name) are not. Of course, this does not exclude the possibility of misuse, but it does prevent the sincere user from being presented with confidential data unwillingly (in case of deliberate misuse, appropriate actions can be taken, since each user must sign a sincerity declaration to obtain a password).

The output of the search engine is then translated into an HTML document, enclosed in a virtual envelope, and sent via the Common Gateway Interface (CGI) to the client where it is interpreted and presented to the user.

## 4. Sophisticated queries

This section may be skipped by the busy reader.

### 4.1 Cascaded filters

One of the special features of our search engine is the *cascaded filter*. By enclosing certain subparts of a profile in angle brackets ( < and > ), one can refer to them individually. The first bracketed substring is then referred to as '\$1', the second as '\$2', and so on. By putting individual restrictions on these named subparts, one can construe sophisticated filters piecemeal.

Consider an example (this section may be skipped by the busy reader). Find adverbial clusters included in compound verb groups of the type 'har ... været' – clusters such as 'har sikkert nok været' and 'har jo heller ikke altid været'<sup>2</sup>. For practical reasons, we will restrict the goal to 2-6 word clusters.

As a first try, we define the profile `_har<_(□+_) {2,6}>været_` and specify that only \$1 is to be registered, i.e. the 2-6 uninstantiated words between the angle brackets. Now the unbracketed parts of the profile serve as context constraints. This is not good enough yet, though. Along with the wanted adverbials, many subject NPs in detopicalized position are admitted: 'har du sely været...', 'nogen gange har min mor og far været...', etc. So we impose a restriction on \$1 that it may not match `_(jeg|du|min|din|han|hun|nogen|alle|vi|de|I|det|den|der)_`. The words 'jeg', 'du', ... are the most frequent Danish pronouns in the nominative and indeed the most frequent subject NP elements in *BySoc*.

Now the profile works, but of course it misses out a lot of relevant adverbial clusters, namely all those contained in *other* participle constructions, such as 'har ... haft' og 'har ... kunnet'. So we soften up the profile: `_har<_(□+_) {2,6}>(haft|□+et)_`, now allowing for a spectre of participle constructions such as 'har faktisk ikke haft en chance' and 'har nu ikke altid løbet så stærkt'. However, a new restriction is called for, since the `(haft|□+et)_` part of the profile happens to match, not only participles, but also pronouns in the neuter, as in e.g. 'har et eller andet imod mig'. So we put `(haft|□+et)_` in angle brackets and specify that \$2 must not match `(det|noget|meget|andet)_`, the four by far most frequent PRO<sub>neuter</sub> in spoken Danish. These little adjustments can be added in a bottom-up fashion, simply by analysing the output of the search engine.

Now the filter performs quite well, making it reasonable to start drawing conclusions from the results delivered by the search engine. Still, there is plenty of room for improvements, on the one hand by extending the hunting ground with modal verbs ('kan'), new auxiliary verbs ('havde'), subordinate clauses ('at jeg ikke har haft den'), etc. – and on the other hand by sharpening the constraints.

Should we later want to focus on special adverbial clusters only – say, those containing negative polarity items, or 'ikke', or discourse particles like 'jo', 'vel' and 'sgu' – this can be done by simply adding more restrictions on the \$-variables.

---

<sup>2</sup> In terms of the *feltskema*, Diderichsen's reknown description of the Danish sentence syntax, the adverbials in question belong to the *a*-slot of the nexus field (Diderichsen (46)).

## 4.2 Two-dimensional profiles

In informal conversation, the word changes rapidly and impulsively – the more so, the more typical the conversation. Hence, any analysis of the semantics and pragmatics of conversation without access to the distribution of utterances over speakers would be impaired, or even meaningless.

So of course such *BySoc* information should be accessible, and this calls for a new entry to the search engine. In order not to have the interface look like the dashboard of a Jumbo jet, we have chosen to add just one new convention: ‘%*n*’. As mentioned in the previous section, ‘\$*n*’ refers to the *n*th bracketed subpart of the profile. ‘%*n*’, in contrast, refers to the same text position as ‘\$*n*’ (i.e. the same time slice), but *anyone but* the same speaker.

Consider an example. In Danish informal speech, the tag *ik’* usually triggers a positive response (we take positive responses to be *jo*, *ja*, and *mm*). Find all exceptions to this rule.

One solution would be to search for `_ik' _<_.{2}>`. This profile finds each utterance-final occurrence of *ik’*, thanks to the two adjacent spaces. The bracketed part of the profile then corresponds to the first 3 characters of the right context – and when referred to with ‘%1’, it applies to the full score, or rather: all speakers with the exception of the speaker saying *ik’*. Now we need only to impose the restriction on ‘%1’ that it may not match `jo|ja|mm`.

Two of the *ik’* occurrences in question occur in the transcription sample in section 0.

## 5. Linguistic sessions (examples, not science!)

### 5.1 Trigrams, n-grams

Trigrams – groups of three adjacent words – are well-known creatures in corpus linguistics. *BySoc* trigrams may be found with the search profile `_(w+_){3}` (easily modified to cover 4-grams etc.).<sup>3</sup>

The search engine reports 780,513 hits, the most frequently occurring trigrams being:

---

<sup>3</sup> Select the output format ‘*list*’ (deselect ‘*text samples*!’). As queries of this kind put a considerable load on the server, you may have to split up the corpus. You could, for instance, cut it in three time slices: 0-49th minute of each interview, 50th-99th minute, and 100th-and-on, and search them individually.

Rank	Form	Count
#1	'det er jo'	1881
#2	'og sådan noget'	1589
#3	'men det er'	1340
#4	'i hvert fald'	1248
#5	'det er det'	1089
#6	'ja det er'	961
#7	'og det er'	858
#8	'det kan jeg'	800
#9	'det ved jeg'	783
#10	'jeg ved ikke'	781

### 5.2 Differences in word selection: men versus women

As mentioned, the corpus can be torn along various seams, including *sex* and *age*, providing for investigations into word selection. For instance, the frequency of a few pronouns, 'jeg', 'man', 'du', 'min', seem to vary quite a bit as a function of sex and age (search profile: `_(jeg|man|du|min)_`).

Frequency and rank	'jeg'	'man'	'du'	'min'
M, 14-25 years	3.8% (#2)	1.0% (#21)	0.63% (#31)	0.30% (#58)
M, 25+ years	2.7% (#5)	1.2% (#16)	0.80% (#27)	0.24% (#66)
F, 14-25 years	5.4% (#2)	0.71% (#29)	0.59% (#34)	0.36% (#45)
F, 25+ years	3.5% (#2)	0.85% (#25)	0.84% (#26)	0.34% (#49)

### 5.3 George Kingsley Zipf revisited

G. K. Zipf was one of the first to suggest general statistical hypotheses about language. The following has become known as 'Zipf's law' (Zipf's own formulation was a bit different, but equivalent).

*In any corpus of considerable size, in any language,*  $\forall z \in \text{Word\_forms: Rank}(z) \times \text{Count}(z) = c,$  *c constant*

Zipf's claim does hold quite nicely in a variety of Danish text corpora, at least for words of rank >15.



Rank	Rank × Count / 1000 <i>newspapers</i>	<i>magazines</i>	<i>scientific journals</i>
#1	8	8	8
#2	13	13	15
#4	15	17	21
#8	25	27	31
#15	28	33	37
#30	22	27	25
#60	19	24	20
#120	22	24	19
#250	23	22	22
#500	25	24	23
#1000	26	24	25
#2000	27	24	24
#4000	25	23	23

Figures based on Maegaard & al (86)

Does informal spoken Danish show the same kind of regularity?

To answer this question, we compute a frequency list using the profile  $\_ \# \_$ , or, more efficiently<sup>4</sup>:  $\_ \# \_$ .

Total: 1,348,083 hits.

Rank	Count	Wave length = Total / Count	Frequen cy	Accumulate d frequency	Rank × Count /1000	Word form
#1	74191	18	5.50%	5.50%	74	'det'
#2	45779	29	3.40%	8.90%	92	'ja'
#3	41338	32	3.07%	12.0%	124	'og'
#4	39497	34	2.93%	14.9%	158	'jeg'
#5	39218	34	2.91%	17.8%	196	'er'
#6	36211	37	2.69%	20.5%	217	'så'
#7	32194	41	2.39%	22.9%	225	'der'
#8	25315	53	1.88%	24.8%	203	'ikke'
#15	17284	77	1.28%	35.0%	259	'ik'
#30	10337	130	0.77%	49.9%	310	'også'
#60	3467	388	0.26%	62.9%	208	'vil'
#120	1359	991	0.10%	72.6%	163	'ude'
#250	477	2826	0.035%	80.3%	119	'hvorfor'
#500	178	7573	0.013%	85.7%	89	'døren'
#1000	70	19258	0.005%	89.73%	70	'situation'
#2000	27	49929	0.002%	92.9%	54	'forstod'
#4000	10	134808	0.0007%	95.4%	40	'bøgerne'

<sup>4</sup> The engine has two search algorithms, one lazy and one meticulous. The lazy one is the faster, but it does not find overlapping hits, so it wont work with the profile  $\_ \# \_$ . On the other hand, it does work with  $\_ \# \_$ , and this profile is (nearly) equivalent, since the + operator is 'greedy', always matching as many characters as possible.

As can be seen in the table, corpus *BySoc* does not follow Zipf's law. The distributional patterns of informal speech seem to be radically different from those of written texts. While this is hardly a controversial observation in itself, it does trigger a series of challenging questions: *How different? Where? And why?*

In conclusion, the reader is invited to try another Zipf-hypothesis on corpus *BySoc*.

"The product of the number of words of a given occurrence, when multiplied by the square of their occurrences, remains constant for the great majority of the different words of the vocabulary in use, though not for those of highest frequency." (Zipf (36) p.41ff).

Visit <http://phoneme.cphling.dk/BySoc>, enter the profile `_m+`, and select the output format *Zipf list*.

## 6. Concluding remarks

Using the Internet as a keyhole, there is little need for distributing copies of a corpus. A single copy residing at the server side is sufficient, for many purposes. This has some significant advantages.

Firstly, the researcher can document his results, simply by reporting the parameter settings of his central queries. Secondly, maintenance of the corpus is fast and efficient. Each user is a potential proof reader, and his reported corrections can be implemented and utilised in minutes or hours. Thirdly, confidential data can be encrypted, and if necessary, effectively barred within an instance. This improved efficiency of control can be foreseen to actually make the corpus *more freely* available, since the corpus administrator can now mete out access much more precisely – if necessary, even on an individual basis – and does not have to employ 'better too little than too much' kinds of precautions. The precision in control could also make the supplier of data feel more confident, as he can now be given (i) a qualified promise that no unauthorised party will gain access, and (ii) a detailed description of what authorised users will be allowed to see and do (one could even trade with a reluctant informant, offering the withdrawal of data in case, say, the private situation of the informant should change).

The client-server model offers *platform independence* on the client side, and *specialisation* on the server side. This may lead to a new and attractive situation: even the slowest of today's 486s or Macs is sufficient for the most powerful searches, as long as there is an Internet connection (and a willing corpus provider at the other end). The implicit anarchy and decentralisation of the Internet could thus strengthen the research community by counteracting a notorious technocratic and anti-individualistic barrier in the corpus based research. Via the Internet, everyone, including individual researchers and low-budget institutions, can have equal access to resources and methods. In the same vein, the technically insecure linguist can make as intricate queries as can the UNIX or Perl wizard.

Last but not least, no one is forced into buying a specific software product, thanks to the system independence of HTML and Java – an obvious democratic advantage, which the Department of Justice in Washington right now is struggling to preserve.

## 7. Acknowledgements

Thank you to the *BySoc* group for the enormous work they have done collecting this unique corpus. It must have been a battle. Thanks also to Dorte Haltrup Hansen who swept the battle ground for left-over misspellings and more.

The work reported in this paper had not been possible without the steady and unconditioned support from Department of Danish Dialectology, particularly Inge Lise Pedersen, for which I am very grateful.

## 8. References

Gregersen &al (1991) *The Copenhagen Study in Urban Sociolinguistics*, 1+2; Copenhagen: Reitzel.

Milroy, L. and J. Milroy (1977) *Speech and Context in an Urban Setting*; Belfast Working Papers in Language and Linguistics, vol. 2,1.

Henrichsen, P. J. (1997) *Talesprog med Ansigtssløftning*; IAAS, Univ. of Copenhagen: Instrumentalis 10/97 (in Danish).

Holtse, P.; P. J. Henrichsen (*in preparation*) *Multi-modal Corpus Presentation via Internet*.

Maegaard, B. &al (86) *Hyppige Ord i Danske Aviser, Ugeblade og Fagblade*; Gyldendal.

Wall, L. &al (1991) *Programming Perl*; O'Reilly.

Zipf, G.K. (1936) *The Psycho-biology of Language – Introduction to Dynamic Philology*; London: Routledge.

# Extraction of Translation Equivalents from Parallel Corpora

**Jörg Tiedemann**

Department of Linguistics

Uppsala University

*joerg@stp.ling.uu.se*

January 25, 1998

## 1 Introduction

In the past much effort was devoted to the compilation of multi-lingual parallel corpora for the purpose of linguistic information retrieval. This paper aims to introduce and evaluate three simple strategies for the extraction of translation equivalents from structured parallel texts. The goal is to support the production of bilingual dictionaries for domain-specific applications. The approaches described in the paper assume sentence alignment, strict translations, and historical relations between considered language pairs. They take advantage of corpus characteristics like short aligned units and structural & orthographic similarities in order to obtain results with a high level of precision. Furthermore, it will be shown that automatic filtering can be used to improve the precision of the extracted material. Simple techniques are used to detect translation candidates that are most likely wrong.

## 2 Pre-processing the Corpus

Important pre-processing steps include paragraph and sentence boundary detection, tokenization, compilation of collocations, and multilingual sentence alignment. The precision of extracted material depends highly on the quality of these pre-processing steps. All three extraction methods introduced in the paper assume tokenised and aligned corpora.

Tokenization is one of the fundamental tasks in preparing the corpus for linguistic information retrieval. The separation of lexical tokens from punctuation symbols is important. Here, special pattern can be defined to segment the corpus. Difficulties arise due to the ambiguity of the used symbols. For further discussions see [GT94].

Automatic sentence alignment can be done by statistical methods like those proposed in [GC93], [FC94], [FM94], [Chu93], [SFI92], and [Mel96]. These approaches yield high precision especially for closely related language pairs.

## 3 Extraction by Iterative Size Reduction

In this section an iterative extraction method is introduced that takes advantage of aligned parallel corpora with a large number of short aligned text structures. This phenomenon often appears in highly structured texts like technical documentation. The following investigations are based on the parallel Scania corpus ([Sca93]) which contains many short structures due to brief descriptions, list elements, tables, picture captions, and headers. These elements can be easily recognised, aligned, and analysed. Among the Swedish/English alignments of the Scania corpus with a total number of 35,818 alignments there are 2,628 aligned header structures, 8,558 aligned

table cell structures, and 8,423 aligned list item alignments. About every third alignment represents a 1:1 token alignment, although many of them are copies of each other.

First, we extracted 1:1, 1:x, and x:1 token alignments from the corpus to compile a basic set of translation equivalents. This basic dictionary was used to analyse the remaining alignments in an iterative process by removing known translations from the total set of corpus alignments. As the result, the size of the alignments (in tokens) decreases. Then, we extracted newly obtained 1:1 token alignments and added them to the set of known translations. This extended set of translation pairs was used in the next step to analyse the remaining alignments from the former step. This procedure may be repeated until no new 1:1 token alignments appear.

This simple algorithm produces a large number of new translation equivalents in case of many short aligned structures. When applied to the Scania corpus the method provides results of high precision as shown in table 1.

Step	Swedish/English	Swedish/German
1	184 (96.2 % correct)	223 (96.7 % correct)
2	90 (93.3 % correct)	231 (98.7 % correct)
3	26 (100 % correct)	165 (98.8 % correct)
4	7 (85.7 % correct)	85 (95.3 % correct)
5	2 (100 % correct)	31 (93.5 % correct)
recall (incl. basic dictionary) <sup>1</sup>	12.2. %	14. 6 %

Table 1: The number of translation equivalents (1:1 token pairs) extracted with the iterative size reduction method.

The quality of the results in each step highly depends on the set of translation equivalents obtained in former steps. In general, wrong pairs will produce wrong alignments in the next reduction step. Table 1 shows that the precision of Swedish/German results decreases slightly in steps 4 and 5. However, the numbers of newly discovered Swedish/English pairs are too small in the final steps to observe a similar behaviour.

The algorithm so far is limited to the extraction of 1:1 pairs. However, in language pairs with different compounding rules (like Swedish/English) many 1:x correspondences appear. Unfortunately, remaining 1:x alignments cannot be considered as correct translation equivalents automatically. Some tokens may belong to previously removed tokens. The remaining tokens are usually grammatical function words. Fortunately, most of the 1:x alignments can be transformed to correct translation equivalents by simply removing these words using language specific function word lists. Adequate lists may be compiled by extracting the most frequent words from the corpus. The efficiency of this simple approach can be seen in table 2.

---

<sup>1</sup>The recall value is estimated by comparing the number of extracted pairs with the total number of source tokens used in the corpus.

	before	after
Swedish/English 1:x	46.9 %	84.5 %
Swedish/English x:1	1.5 %	79.1 %
Swedish/German 1:x	4.4 %	84.6 %
Swedish/German x:1	44 %	69.2 %

Table 2: Precision estimates for extracted Swedish/English and Swedish/German 1:x respectively x:1 alignments before and after the removal of function words.

The removal of function words improves the precision values greatly. Applying a more sophisticated list of function words would produce even better results and would admit the usage of 1:x pairs for the iterative extraction process. This would represent a major improvement especially for language pairs like Swedish/English.

Finally, it is worth mentioning that any bilingual dictionary may be used by the algorithm, provided that it suits the domain of the corpus under consideration.

#### 4 Considerations to String Similarities

Evaluations to string similarities can be used for different tasks in computational linguistics. One application is the identification of morphological deviations. This may be used in the size reduction method, which is introduced in the previous section, to identify slightly modified translation pairs (see further [Tie97]). Another application is the identification of cognates from bilingual texts in case of similar character sets and historical relations between the languages under consideration. In both cases simple string matching algorithms can be used to compare word pairs. Applications to technical texts are especially profitable because of internationalisation and similarities in the origin of technical terminology.

One simple algorithm based on character comparison is the 'longest common sub-sequence ratio' (LCSR) which is defined as follows: The LCSR score is calculated by the length of the longest common, not necessarily contiguous, sub-sequence of characters divided by the character length of the longer string ([Mel95]).

Mostly, even more simple algorithms represent sufficient measures for string similarity. We used algorithms to search initial and final character sequences, and algorithms to compare fixed character sequences. More complex algorithms may raise the recall but not the precision of results when applied with similar threshold values. See further [Bor98] for a comparison of different approaches.

The selection of word pairs, which are considered for the string comparison, is important for the quality of the extraction of cognates. A major improvement represents sentence alignment. The chance of getting 'false friends' remains very low if only word pairs from aligned text structures are considered. When applied to the Swedish/English and Swedish/German alignments of the

Scania corpus with a threshold of 70% the method provides results with high precision as shown in table 3.

	Swedish/English	Swedish/German
pairs	360	843
precision	94.7 %	97.9 %
recall <sup>2</sup>	2.3 %	5.5 %

Table 3: The number of cognates that were identified with string similarity measures.

Sentence alignment also allows further evaluation of similarity measures. Scores from token pairs can be combined for further analyses (see also [Tie97]). Usually, words that are very similar to only one word of the corresponding alignment are more likely translation equivalents than words that show similarities to several words from the corresponding alignment. This assumption can be expressed by the following similarity score combinations:

$$diff_s(x, y) = sim(x, y) - \sum_z^{z \in V} sim(x, z)$$

$$diff_l(x, y) = sim(x, y) - \sum_z^{z \in L} sim(z, y)$$

Here, the term  $sim(x,y)$  represents the similarity score for the token pair  $(x,y)$ . In this way new scores for each token pair are calculated which may be evaluated by specific threshold filters. Including evaluations of similarity score combinations the similarity method yielded a recall of 3.2% with an estimated precision of 92.3% for Swedish/English alignments, and a recall of 7.7% with an estimated precision of 96.2% for Swedish/German alignments.

## 5 Extraction based on Statistical Measures

Many statistical measures are based on co-occurrence measures. The basic input data is the absolute frequencies of the occurrence of single words or word groups and the co-occurrence frequencies of word pairs and pairs of word groups in corresponding subparts of the text (for instance aligned sentences). Using these values, statistical measures can be estimated for monolingual as well as for multi-lingual input.

Similar to word distribution measurements based on Mutual Information like in [FC94] another statistical ratio, the Dice coefficient can be used to measure the co-occurrence of words or word groups. This ratio is used, for instance, in the collocation compiler XTract ([Sma93]) and in the lexicon extraction system Champollion ([SMH96]). It is defined as follows:

---

<sup>2</sup> The recall value is hard to estimate. The value mentioned in table 3 is measured by comparing the number of resulting pairs with the set of source words used in the corpus. According to expectations, the recall is very small for both language pairs because only a small subset of word pairs actually represent cognates.

$$Dice(x, y) = \frac{2P(x, y)}{P(x) + P(y)}$$

$P(x, y)$  represents the probability for the occurrence of  $x$  and  $y$  in corresponding parts;  $P(x)$  and  $P(y)$  are the single probabilities that  $x$  and  $y$  occur. All these values can be estimated by corresponding frequency values. Notice that  $x$  and  $y$  may be single words as well as word groups.

The major advantage of statistical measures is language independence. The major problem, however, is the proper selection of monolingual text units for the considerations. The chosen text units have to be comparable in their semantic complexity, otherwise statistical measures produce incorrect and incomplete results. Consider for instance the different usage of compounds in Swedish and English. In Swedish, compounds are mostly used in the form of compositions while in English corresponding expressions appear as sequences of separated words. Therefore, a sophisticated compilation of monolingual collocations should be performed for the identification of non-compositional compounds before performing statistical extraction.

A second problem is the different variety of morphological forms in different languages. To solve this problem all word forms should be reduced to their stem forms before counting occurrence frequencies.

A last problem arises with infrequent words. Because of its definition the introduced ratio may produce high scores for pairs containing infrequent text units although they do not correspond. This fact is due to the high probability that infrequent words appear in corresponding text parts by chance. Therefore, infrequent text units should be excluded from this statistical consideration.

However, infrequent text units can be analysed in a different way. Like in the 'size reduction' method (see section 3) we presume short aligned text units. By using monolingual frequency counts, all terms with a higher frequency than a specific threshold value  $t_1$  are removed from the complete set of alignments. Then, the remaining alignments are examined. With the assumption that infrequent terms are translated into infrequent terms in the other language, all remaining infrequent one-to-one term pairs (a term may be a word or a word phrase) are extracted which occur less frequently than another threshold value  $t_2$ . These pairs may be considered as translation equivalent candidates. Again, the biggest difficulty is the selection of proper terms for frequency counts.

Due to time constraints the algorithms were applied without previous compilation of collocations and without stemming. Sentences were simply segmented into space separated tokens and frequency counts were based on these text elements. Filtering the Dice coefficient with a threshold of 0.7 provided results with a recall of 7.8% and an estimated precision of 86% for Swedish/German alignments and a recall of 4.1% with an estimated precision of 70% for Swedish/English alignments. The extraction of low frequent word pairs with  $t_1=100$  and  $t_2=10$  provided results with a recall of 6.7% and a good precision of 93.5% for Swedish/German alignments, and a recall of 2.8% and an unsatisfactory precision of 46% for Swedish/English alignments.



## 6 Automatic Evaluation Filter

The results from the three extraction methods introduced above are certainly not perfect and free of mistakes, although precision yielded promising values. Fortunately, automatic filters can be used to remove pairs that are most likely wrong. Statistical and empirical evaluations can be applied to analyse the set of translation candidates. Special considerations should be attached to terms, for which multiple translation candidates were extracted. Most errors are to be found here. The translation candidates can be compared with each other and the most unlikely candidates can be removed automatically. Several approaches are applicable:

**Length based filter:** A length difference ratio can be calculated by dividing the length of the shorter string by the length of the longer string. In case of multiple translation candidates the pair with the highest score identifies the most likely translation. Now, all scores for alternative candidates are compared with this score and all pairs, which do not pass a chosen threshold, are removed. The length difference can also be used to compare alternative translations with the most likely translation. Furthermore, it may be also used to exclude translation candidates with unreasonable length differences to the source language term.

**Similarity filter:** Word comparison algorithms can be applied to extracted pairs. The highest resulting score identifies the most likely translation among multiple translation candidates. Then, all similarity scores of alternative candidates are compared with the similarity score of the most likely translation. Another possibility is to calculate similarities between the most likely translation and alternative candidates. This presumes multiple translations to appear because of slightly morphological modifications rather than because of synonym translations.

**Frequency based filter:** Absolute and co-occurrence frequencies are calculated for extracted pairs. Using these values, the Dice coefficient is estimated and used for the removal of translation candidates with unreasonable low scores. In case of multiple translations the Dice score can also be used to identify the most likely translation and the score for alternative candidates is compared with the score of this translation.

**Combined filter:** The filter described above may be combined. Generally, one approach may be used for the identification of the most likely translation and another approach may be used for the comparison of alternative translations with this candidate. One possible combination is for instance the usage of the Dice coefficient for the identification of the most likely translation and a similarity filter to compare this candidate with alternative ones.

**Subset filter:** Translation candidates may be incomplete. Therefore, if one translation candidate is completely included in another candidate it should be removed. Translation candidates can be considered as sets of words. Translations are removed if the complete set of words is included in an alternative translation candidate.

A non-trivial task is to adjust these filters to obtain the best result. The improvement level depends on the quality of the previously applied extraction methods.

## 7 Discussion and Conclusion

In this paper three methods are described for the retrieval of translation equivalents from aligned bilingual corpora. They represent independent approaches that use different assumptions. Table 4 shows corpus characteristics and preparations that are necessary for each single approach.

method	highly structured text	similar character set	tokenization	sentence alignment	compilation of collocations	removal of function words
size reduction	X		X	X		X
similarity measures		X	X			
score combinations		X	X	X		
Dice statistics			X	X	X	
low frequent terms	X		X	X	X	

Table 4: Necessary characteristics and preparations in order to apply the different extraction methods.

The size reduction method takes advantage of corpora with many short aligned text structures. Quantity and quality will decrease if less structured texts are used even if a large set of translation equivalents is used in the initial step. For highly structured texts this method provides fast and precise results. The method is not necessarily limited to 1:1 word pairs, although the extraction of phrase alignments needs additionally the removal of function words. An advantage is that any otherwise compiled dictionary may be used by the algorithm as long as it suits the domain of the corpus. In this way, the size reduction method was applied again to the Scania corpus using an initial dictionary that was compiled by different extraction methods. The recall<sup>3</sup> of the final dictionary amounts to 28.3% with an estimated precision of 96.5% for Swedish/English and 49.4% with an estimated precision 96.7% for Swedish/German.

The string similarity approach aims to extract closely related word pairs. It is limited to 1:1 token pairs. The method is applicable to historically related language pairs only. Precision and recall can easily be adjusted by modifying the threshold value. The application described in the paper provides results with precision and recall similar to the size reduction method. However, the extracted pairs represent different sets for both approaches. In contrast to the other approaches the cognate extraction method depends highly on the language pair and assumes similar character sets.

Statistical extractions depend on the quality of the text segmentation and the frequency counts. The simple approach described in the paper provides results with recall values similar to the other two extraction methods. However, the precision is much lower for both language pairs. Especially for Swedish/English alignments many incomplete pairs appear. A more sophisticated segmentation and previously applied stemming would improve the precision greatly.

<sup>3</sup>The number of extracted pairs is compared with the total number of source tokens that are used in the corpus to estimate the recall.

When applied to the Swedish/German alignments all three approaches provided results with much higher precision and recall than when applied to Swedish/English alignments, although the size reduction method and statistical evaluations should be mostly language independent. However, essential pre-processing steps like collocation compilation and stemming were not carried out before applying these algorithms and therefore, remarkable differences in quality and quantity of the achieved results arose.

## References

[Bor98] Lars Borin. Linguistics isn't always the answer: Word comparison in computational linguistics. In Proceedings of the 11th Nordic Conference on Computational Linguistics NODALIDA '98, Copenhagen, 1998.

[Chu93] Kenneth W. Church. Char align: A Program for Aligning Parallel Texts at the Character Level. In Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives, ACL. Association for Computational Linguistics, 1993.

[FC94] Pascale Fung and Kenneth W. Church. K-vec: A New Approach for Aligning Parallel Texts. In Proceedings of the 15th International Conference on Computational Linguistics, Kyoto/Japan, 1994.

[FM94] Pascale Fung and Kathleen R. McKeown. Aligning Noisy Parallel Corpora Across Language Groups: Word Pair Feature Matching by Dynamic Time Warping. In Proceedings of the 1st Conference of the AMTA, Columbia/Maryland, 1994. Association for Machine Translation in the Americas.

[GC93] William A. Gale and Kenneth W. Church. A program for aligning sentences in bilingual corpora. Computational Linguistics, 19(1), 1993.

[GT94] Gregory Grefenstette and Pasi Tapainen. What is a word, What is a sentence? Problems of Tokenization. In Proceedings of the 3rd International Conference on Computational Lexicography (COMPLEX '94), Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, 1994. Rank Xerox Research Centre, Grenoble Laboratory.

[Mel95] I. Dan Melamed. Automatic Evaluation and Uniform Filter Cascades for Inducing N-best Translation Lexicons. In Proceedings of the 3rd Workshop on Very Large Corpora, Boston/Massachusetts, 1995.

[Mel96] I. Dan Melamed. A Geometric Approach to Mapping Bitext Correspondence. In Conference on Empirical Methods in Natural Language Processing, Philadelphia/USA, 1996.

[Sca93] Scania project - Homepage. <http://stp.ling.uu.se/~corpora/scania/> Uppsala University, Linguistics Department, 1997.

[SF192] Michael Simard, George F. Foster, and Pierre Isabelle. Using Cognates to Align Sentences in Bilingual Corpora. In Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation, Montreal/Canada, 1992.

[Sma93] Frank Smadja. Retrieving Collocations from Text: XTRACT. Computational Linguistics, 1993.

[SMH96] Frank Smadja, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. Translation Collocations for Bilingual Lexicons: A Statistical Approach. In Association for Computational Linguistics. Association for Computational Linguistics, 1996.

[Tie97] Jörg Tiedemann. Automatical Lexicon Extraction from Aligned Bilingual Corpora. Diploma thesis, Magdeburg University, Department of Computer Science, 1997.

# The Construction of a Tagged Danish Corpus

**Thomas Bilgram**

Dept. of Linguistics, University of Aarhus, Denmark. bilgram@ling.aau.dk

**Britt Keson**

DSL, Christians Brygge 1,1. 1219 København K, Denmark. parole@coco.ihl.ku.dk

## Abstract

The object of this paper is to present ongoing work on the construction of a morphosyntactically tagged Danish corpus, which is an integral step in the making of a Constraint Grammar (CG) parser for Danish and also constitutes a part of the Danish contribution to the European PAROLE project. This paper discusses various aspects of the morphological description of Danish used here as well as some of the guidelines developed for the manual disambiguation process. Finally, it also briefly gives an overview of the objectives of the two projects involved.

## 1. Introduction

The work on the construction of a morphosyntactically tagged Danish corpus as described here was undertaken as a joint effort by stud. PhD Thomas Bilgram, University of Aarhus, and cand. mag. Britt Keson and stud. mag. Dorte Haltrup Hansen from Det Danske Sprog- og Litteraturselskab (DSL), one of the two Danish partners in the PAROLE project. One of the aims of the PAROLE project is to produce a morphosyntactically tagged corpus of approx. 250,000 running words for the languages of each of the European partners. For Thomas Bilgram, the tagged corpus will serve as testing material for the development of a Constraint Grammar automatic tagger/parser for Danish.

The Danish text material to be tagged was composed of a random selection of approx. 1600 excerpts containing one or more consecutive paragraphs (each excerpt totalling approx. 150 - 180 words) extracted in part from the 40 mill. word corpus of the Danish Dictionary at DSL. This subcorpus was then analysed using a Two-level description of Danish morphology, DAN-TWOL, to assign to each word all of the possible morphosyntactic analyses, after which the correct analysis was chosen during a manual disambiguation process. In order to make the result as consistent as possible, this disambiguation process was for the most part carried out in parallel on the same texts, and later consensus on possible differences was achieved among the human taggers. The first 50,000 running words were treated as a pilot project to ensure that the information yielded by DAN-TWOL was sufficiently detailed for the purposes of both projects, and to develop a tagging manual to guide the human taggers in the disambiguation process.

## 2. DAN-TWOL

### 2.1 TWOL: A short introduction

The Two-level algorithm was originally designed by Kimmo Koskenniemi [Koskenniemi 83]. A TWOL is based on the principle that a word can be regarded as having two separate levels, a surface and a lexical level, as well as a description of the relation between these two levels. The surface level is the word as it appears in a text. The lexical level consists of (i) the morphemes (i.e. lemmas, derivational morphemes and inflectional morphemes), (ii) a description of the morphotax – i.e. inflections, derivations, and compounding – and (iii) a description of allomorphy. The DAN-TWOL was made during the graduate studies of Thomas Bilgram [Bilgram 94].

#### The TWOL tagset vs. other tagsets

The fundamental idea in a TWOL is that a word is analysed by a process of accepting one letter of the word at a time. When a sequence of letters is accepted as a morpheme, the part of the analysis that this morpheme constitutes is appended to the analysis string. Hence, the resulting analyses have a very modular appearance<sup>1</sup> (See Fig. 1).

Fig. 1.

bil+∅+∅+∅	=	<b>N</b>	<b>FLS</b>	<u>SG</u>	<b>UBEST</b>	<b>NOM</b>
bil+∅+∅+s	=	<b>N</b>	<b>FLS</b>	<u>SG</u>	<b>UBEST</b>	<b>GEN</b>
bil+er+∅+∅	=	<b>N</b>	<b>FLS</b>	<u>PL</u>	<b>UBEST</b>	<b>NOM</b>
bil+er+∅+s	=	<b>N</b>	<b>FLS</b>	<u>PL</u>	<b>UBEST</b>	<b>GEN</b>
bil+∅+en+∅	=	<b>N</b>	<b>FLS</b>	<u>SG</u>	<b>BEST</b>	<b>NOM</b>
bil+∅+en+s	=	<b>N</b>	<b>FLS</b>	<u>SG</u>	<b>BEST</b>	<b>GEN</b>
bil+er+ne+∅	=	<b>N</b>	<b>FLS</b>	<u>PL</u>	<b>BEST</b>	<b>NOM</b>
bil+er+ne+s	=	<b>N</b>	<b>FLS</b>	<u>PL</u>	<b>BEST</b>	<b>GEN</b>

This modularity makes the analyses look somewhat different from other PoS analysis systems (such as the CLAWS word tagging system [Garside 97]), where a single, and often quite compact, tag bears all the morphological (and often syntactic and semantic) information. However, in general the full set of features in the TWOL can be compared to a tag in other systems, and this leads to a larger and much more detailed tag inventory. The definitions of the tags (i.e. strings of features) in a TWOL is focused on the contents of the morphological features, and hence the number of tags is a direct product of the number of features. At present, a Danish corpus of 250,000 running words was given just over 500 different tags when analysed by DAN-TWOL. This figure can be compared to other tag systems, such as the Penn Treebank corpus (approx. 50 tags) and the Brown Corpus (just over 200 tags).<sup>2</sup>

#### 2.1.2 DAN-TWOL

The result of a DAN-TWOL analysis is a list of all the possible morphosyntactic analyses of a given word. More than half of the words in the corpus were ambiguous, and the average level of ambiguity was approx. 2 analyses for every word. Making this ambiguity explicit is the primary goal for the TWOL, and a word and the possible analyses given to it by DAN-TWOL is referred to as a cohort.

<sup>1</sup> The PoS and gender information (bold type) is part of the lexicon. The number, definiteness and case information (underlined) is a result of the acceptance of – possibly nil – morphemes.

<sup>2</sup> The differences in the tagsets in different taggers render it virtually impossible to make a comparison of performance and output from different automatic taggers. An attempt has been made by Jochen Leidner in a CLUE report [Leidner 97]. The figures mentioned here are from this report.

DAN-TWOL is composed of a description of a lexicon of Danish (i) lemmas, (ii) inflectional and (iii) derivational morphemes, as well as a description of some allomorphy. The lemmas in the lexicon are based on a machine-readable version of the 1986 edition of *Retskrivningsordbogen* [Sprognævn 86] the official Orthographical Dictionary of Danish. From this, all the regular lemmas were extracted and converted into the format needed in DAN-TWOL. For example, nouns were coded for gender, number and definiteness morphemes, and verbs were coded for tense and participle morphemes. All irregular lemmas were handcoded in the lexicon. The lexicon has since been incrementally updated from its original size of approx. 42,000 entries to presently just over 49,000 entries. The inflectional and derivational parts were implemented on the basis of [Arndt 92] and are very similar to the morphology outlined in the Orthographical Dictionary [Sprognævn 96].

The assumption in TWOL that the lexicon offers full coverage of all the words found in a given text is very rarely true. Large real-life corpus texts will always contain words not included in the lexicon, and these words are returned as unanalysed by DAN-TWOL, and hence have to be handled by other means<sup>3</sup>. A consequence of the relatively free compounding in DAN-TWOL is that words which should not have received an analysis by DAN-TWOL erroneously are analysed as various compounds. For example, the name 'Bilgram' is analysed as a compound of 'bil' (*car*) and 'gram' (*gram*). This situation can be regarded as a continuum; at one end we have a system based on a closed list of fully inflected words recognised by the analyser, and at the other end we have a very liberal morphotactical system. The former system yields very good analyses of all the recognised words, but leaves a great number of words unrecognised, while the latter system yields an analysis for almost every word in the text, but the analyses proposed by the system are not always acceptable. The TWOL system is clearly located at the liberal end of this continuum.

The Orthographical Dictionary [Sprognævn 96] contains a few notes (§ 35 and §36) on the change of PoS that takes place in connection with the inflected forms of certain lemmas. For example, past participles can appear inflected according to number and definiteness in the same way that adjectives are, and (typically the *-t* inflected form of) adjectives can also occur with an adverbial function. In the CG approach there is a clear-cut distinction between the PoS and the syntactic function and also an acceptance of the fact that words of a certain PoS in the lexicon can appear in syntactic positions typically occupied by words of another PoS. Hence, the option of adding, for example, all past participles to the lexicon as possible adjectives was deliberately avoided. Instead, the possibility of adding this syntactic information during the manual disambiguation process was introduced.

## 2.2 PoS Definition in DAN-TWOL

The PoS definition in DAN-TWOL is based on the difference in the set of applicable features. Nouns have one set of features, verbs another, pronouns a third, etc. Fig. 2 is a listing of the different PoS categories and possible features for those used in DAN-TWOL.

---

<sup>3</sup> A natural part of a CG tagger is a heuristic module that supplies the unrecognised word with a cohort.

If this is Fig. 2.

compared to the PAROLE tagset in section 5, one will notice that generally the same features are present. The

PoS	1	2	3	4	5	6	7
N	Gender	Number	Definiteness	Case			
V	Mood	Tense	Voice	(Gender)	Number	Definiteness	Case
A	Degree	Gender	Number	Definiteness	Case		
PRON	Number	Definiteness	Case				
NUM	Ord/Cad	Case					
EGEN	Case						

major differences are due to the TWOL distinction between the truly morphologically derivable information and information that is more syntactically orientated. The latter information is given in '<>': E.g. U <cc>, U <cs>, U <inf>, U <prep>, U <adv>, U <midsub>. The '<>' are used to represent other kinds of distributional or textual information as well: For example, <\*> (initial capital letter), <upl> (noun without a plural form) and <x-sg> (referent of pronoun is singular).

### 3. The Disambiguation Process

During the disambiguation process, the human tagger evaluated all the analyses in the cohort and marked the analysis regarded as correct with the tag <Correct!>. As mentioned in 2.1.2, the tagset defined for the PAROLE necessitated the addition of 'transcategorization' tags during this disambiguation process (examples in section 4.2).

The actual disambiguation process was performed using telnet connections to a linux server. The texts were handled in a restrictive mode for the text editor emacs, allowing only movement in the text and the addition and deletion of the various types of 'correct' tags. Hence, it was not possible to edit the actual texts by mistake.

### 4. Some examples from the tagging guidelines

#### 4.1.1 Common Nouns

In general, a capitalised noun which has received a morphosyntactic analysis as a common noun from the DAN-TWOL was tagged as a common noun, unless the word clearly is the name of person, a geographical location or the like. The motivation for this distinction between common and proper nouns was: (i) to take into consideration the widespread variation in the use of capitalisation in various 'naming' expressions, and (ii) to avoid overpopulating the lexicon with potential (i.e. capitalised) proper nouns, creating an undesirable ambiguity for words in sentence-initial position. Since complex naming expressions like 'det konservative folkeparti' (*the conservative party*) are not recognised as a single unit by the morphological analyser, they had to be treated on a strict word-by-word basis. As seen below, a number of Danish personal names indeed overlap with common noun readings, but in these cases the proper noun tag was preferred (marked by ☞), and if missing, was added later to complete the cohort (marked by ☛).

Fig. 3.

common noun	>
proper noun	>
foreign word	



nu har statsministeren bedt departementschefen undersøge sagen  
(now te prime minister has asked the permanent undersecretary to investigate the matter)  
☞ "stat\#minister" N FLS SG BEST NOM

Det konservative Folkeparti er det største oppositionsparti  
(The conservative Party is the largest opposition party)  
"den" <\*> PRON SG BEST NOM <x-int> <w-pers>  
"det" <\*> U <midsubj>  
☞ "den" <\*> PRON SG BEST NOM <x-int> <w-demo/art>  
"konservativ" A POS UK PL UB NOM  
☞ "konservativ" A POS UK SG BEST NOM  
☞ "folk\#parti" <\*> N INT SG UBEST NOM

"Det giver en masse rutine," siger Thomas Bjørn  
(It gives a lot of experience," Thomas Bjørn'bear' says)  
"bjørn" <\*> N FLS SG UBEST NOM  
☞ "bjørn" EGEN NOM

Det kunne være interessant at møde Statsministeren  
(It could be interesting to meet the Prime Minister)  
☞ "stat\#minister" <\*> N FLS SG BEST NOM

et folketingsmedlem for Det Konservative Folkeparti  
(a member of parliament for The Conservative Party)  
"den" <\*> PRON SG BEST NOM <x-int> <w-pers>  
"det" <\*> U <midsubj>  
☞ "den" <\*> PRON SG BEST NOM <x-int> <w-demo/art>  
"konservativ" <\*> A POS UK PL UB NOM  
☞ "konservativ" <\*> A POS UK SG BEST NOM  
☞ "folk\#parti" <\*> N INT SG UBEST NOM

Peter Søndergård har leveret en levende montage  
(Peter Søndergård'southern farm' has given a vivid montage)  
"sønder\#gård" <\*> N FLS SG UBEST NOM  
☞ "søndergård" EGEN NOM

#### 4.1.2 Proper Nouns

By far the most words to be marked as proper nouns during the disambiguation process were names of people and geographical locations (countries, cities, rivers etc.). In addition, the proper noun tag was given to the 'proper noun element' of complex naming expressions for companies, institutions, sports teams, music groups, book titles, film titles etc., leaving all words that overlap with lexical common nouns to be tagged as common nouns. Early on, it became clear that most of the capitalised word unrecognised by the DAN-TWOL analyser in fact turned out to be foreign words with what could be regarded as proper noun denotations, and hence the 'proper noun element' definition was extended to include either (i) the (capitalised) name of a person or geographical location or the like, or (ii) any capitalised word unknown to the DAN-TWOL analyser. In the few instances where the DAN-TWOL analyser had assigned a Danish morphosyntactic analysis to a capitalised foreign word that was part of a non-Danish phrase, it was decided to disregard this analysis and give consistent proper noun tags to each element of the whole expression.

det betalingsstandsede firma Accumulator Invest  
(the suspended company 'Accumulator Invest)  
☞ "accumulator" EGEN NOM  
☞ "invest" EGEN NOM

kvindlige og mandlige betjente fra Københavns Politis station  
(female and male police officers from Copenhagen Police station)  
☞ "københavn" EGEN GEN  
☞ "politi" <\*> <upl> N INT SG UBEST GEN  
☞ "station" N FLS SG UBEST NOM

i foråret 1990 blev Rungsted Gymnasium inviteret  
(in spring 1990 Rungsted Grammar School was invited)  
☞ "rungsted" EGEN NOM  
☞ "gymnasium" <\*> N INT SG UBEST NOM

bogen, som kommer fra Royal Botanic Gardens  
(the book, which comes from the 'Royal Botanic Gardens)  
"royal" <\*> A POS FLS SG UBEST NOM  
☞ "royal" EGEN NOM  
☞ "botanic" EGEN NOM  
"garde" <\*> N FLS SG BEST GEN  
☞ "gardens" EGEN NOM

Compound nouns are relatively frequent in Danish, and here the generative element of the DAN-TWOL morphological analyser proved particularly useful for identifying the compound words not already listed in the lexicon. Incorporated in the analyser is the decision always to assign the PoS of the last element of a compound word, regardless of the PoS of the preceding element(s).

dansetruppen har banandansen med i programmet  
(the dance group has the banana dance in its program)  
"banan\#dans-en" <kentaur> N FLS SG UB NOM  
☛ "banan\#dans" N FLS SG BEST NOM

at give den gamle Dickens-historie en flyvende start  
(to get the old Dickens story off to flying start)  
☛ "\*\*dickens\#-historie" N FLS SG UBEST NOM

### 4.1.3 Foreign Words

Another common potential dilemma exists between analysing a word as a (common or proper) noun or as a foreign word. In our work, the <foreign word> tag was retained only as an absolute last resort for unrecognised, non-capitalised words that are not of Danish origin, and which, due to their relative obscurity or infrequency, we felt would be unreasonable to add to the lexicon, and which therefore remained completely unanalysed after the disambiguation process. Hence, the <foreign word> tag was *not* used to mark contemporary foreign loan-words that for one reason or another do not appear in the Orthographic Dictionary. Instead, these loan-words were given 'proper' Danish morphosyntactic analyses, whenever it was felt that enough information was present to do so. The decision to tag capitalised unrecognised words as proper nouns and non-capitalised unrecognised words (most frequently) as foreign words proved to be useful in most instances, an obvious exception being foreign song/film/book etc. titles and names of people, where capitalised and non-capitalised words may occur in succession.

Tjeneren undrede sig over aquadenten  
(the waiter wondered about the aquadente)  
☛ "aquadente" N FLS SG BEST NOM

fra den dag tilhører han Erlanders pojkar  
(from that day on he belongs to Erlander's pojkar)  
☛ "pojkar" <foreign word>

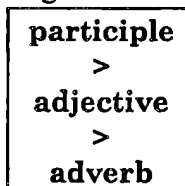
at sidde med en pint og "The Sun"  
(to sit with a pint and "The Sun")  
"pine" V INF PCP2 UK SG UBEST NOM  
☛ "pint" N FLS SG UBEST NOM

den saudiske generaløjtnant Khalid bin Sultan  
(the Saudi lieutenant-general Khalid bin Sultan)  
☛ "\*\*khalid" EGEN NOM  
☛ "bin" EGEN NOM <foreign word>  
"sultan" <\*> N FLS SG UBEST NOM  
☛ "\*\*sultan" EGEN NOM

## 4.2 Adjectives, Adverbs and Participles

Adjectives, adverbs and participles also represent an interesting ambiguity class in Danish. In this case, with regard to the different needs of the potential users of the tagged texts, it was decided to adhere closely to the PoS assignments in the Orthographic Dictionary for these words, and not to assign the members of this ambiguity class the various possible PoS by expanding the DAN-TWOL lexicon. Instead, 'transcategorization' tags, such as <use-adv>, were used to mark the syntactic usage of these words in a particular context and thereby create a more informative, and hence more flexible, resulting tagset. As will become evident from the examples below, the general tendency with this ambiguity class was to prefer a participle analysis to an adjectival analysis, and to prefer an adjectival analysis to an adverbial analysis (See Fig. 4.)

Fig. 4.



## 4.2.1 Adjectives and adverbs

When a lexical adjective functions syntactically as an adverb in a given context, it was decided to retain the adjective PoS analysis and to add the syntactic information with the <use-adv> tag. The only exceptions to this rule are words which have been assigned both an adjective PoS and an adverb PoS in the Orthographic Dictionary, usually because of a significant difference in meaning.

når blodet langsomt kommer i kog  
(when the blood slowly begins to boil)  
☛ "langsom" A POS INT SG UBEST NOM <use-adv>

desværre dør hun tidligt i filmen  
(unfortunately she dies early in the movie)  
☛ "tidlig" A POS INT SG UBEST NOM <use-adv>

naboerne er lige ved at ringe til børneværnet  
(the neighbours are just about to call the RSPCC)

☛ "lige" U <adv>  
"lige" A POS UK UT UB NOM  
"lige" N FLS SG UBEST NOM  
"lig" A POS UK PL UB NOM  
"lig" A POS UK SG BEST NOM

## 4.2.2 Participles

As Danish lexical present and past participles often appear with an adjectival or adverbial syntactic function on par with lexical adjectives, it was decided to use transcategorization tags for participles as well. Attributive adjectival use of present and past participles is indicated with the <use-adj> tag as shown below. Hence no attempt has been made to convert lexical participles to adjectival PoS, even when the (past) participle appears inflected for number, definiteness and (occasionally) gender. Attributive adverbial use of present and past participles is marked with the <use-adv> tag.

hele familien overvejer følgende punkter  
(the whole family is considering the following points)  
☛ "følge" V INF PCP1 NOM <use-adj>

Han var iført meget stramme, slidte cowboybukser  
(He was wearing very tight, worn jeans)

☛ "slide" V INF PCP2 UK PL UB NOM <use-adj>  
"slide" V INF PCP2 UK SG BEST NOM

min mor klarer sig forbavsende godt  
(my mother does surprisingly well)  
☛ "forbavse" V INF PCP1 NOM <use-adv>

en stor gruppe formodet raske østeuropæiske børn  
(a large group of presumably healthy East European children)

☛ "formode" V INF PCP2 UK SG UBEST NOM <use-adv>  
"for | mod" <upl> N INT SG BEST NOM

When both a participle analysis and adjectival analysis are possible, the participle analysis is usually preferred, unless (i) the corresponding verbal lemma does not exist, or (ii) the two analyses differ too greatly in meaning (rarely the case). The former case is true when the DAN-TWOL morphological analyser has productively generated an analysis as the participle form of a non-existing verbal lemma.

kræft i livmoderen eller tilgrænsende organer  
(cancer in the uterus or adjacent organs)  
☛ "tilgrænsende" A POS UK UT UB NOM  
"til | grænse" V INF PCP1 NOM

Man er født med et bestemt antal hårsække  
(One is born with a certain number of hair follicles)  
☛ "bestemme" V INF PCP2 UK SG UBEST NOM  
"be | stemme" V INF PCP2 UK SG UBEST NOM  
"bestemt" A POS UK SG UBEST NOM



## 5. The PAROLE Project

The purpose of the EU-funded LE-PAROLE project is to produce a large-scale harmonised set of 'core' corpus and lexicon resources for the languages of the European Union. One of the two main objectives of the corpus part of the PAROLE project is to produce large monolingual corpora of approx. 20 million running words, which will obey certain common mark-up

conventions, namely the PAROLE Corpus Encoding Standard (CES) as presented in [Ridings 96], which is in line with the EAGLES/MULTEXT CES guidelines [Ide et al 95]. The other main objective is to construct a subcorpus of approx. 250,000 running words, which are to be morphosyntactically tagged according to predefined tagsets that are compatible with the PAROLE lexicons. These resources produced by PAROLE will be made available through the European Language Resources Association (ELRA).<sup>4</sup>

Fig. 5.

PoS	Type	3	4	5	6	7	8	9	10	11
Noun		Gender	Number	Case			Definite-ness			
Verb		Mood	Tense	Person	Number	Gender	Definite-ness	TrCat	Voice	Case
Adj		Degree	Gender	Number	Case		Definite-ness	TrCat		
Pron		Person	Gender	Number	Case	Possessor	Reflexive	Register		
Det		Person	Gender	Number	Case	Possessor				
Art		Gender	Number	Case						
Adv		Degree	Function	Wh-ness						
Adpos		Formation	Gender	Number						
Conj		Ctype	Coord-posit							
Num		Gender	Number	Case						
Interj										
Residual										
Unique										

 = categories and features used in the Danish PAROLE corpus  
 = categories and features not used in the Danish PAROLE corpus

The Danish PAROLE tagged subcorpus will be based on an automatic conversion of the DAN-TWOL morphosyntactic tags to the Danish PAROLE tagset format (see Fig. 5). The common PAROLE tagset format was specified in the PAROLE *Multilingual Corpus Tagset Specifications* [Volz and Lenz 96], which constitutes an enhanced version of the EAGLES/MULTEXT tagset as presented in [Monachini and Calzolari 96]. In this format, the first position in the tag string contains a character for the PoS, and the second position contains a character representing one of various predefined PoS subcategorization types. The positions 3 - 7 contain morphological features taken from a common PAROLE feature set, while the positions 8 and up contain morphological features that are language-specific. Underspecification and ambiguities are expressed using a combination of various ambiguity markers (and characters), while symbols such as '=' and '.' are used as fillers. In the tagged subcorpora, a morphosyntactic tag is expressed using the following notational format: <W lemma=word\_lemma msd="tag\_string">word</w>.

The following example (discussed in section 4.2) demonstrates what the conversion to the Danish PAROLE tagset and format looks like:

<sup>4</sup> More information about the PAROLE project in general can be found at <http://www-tei.uic.edu/orgs/tei/app/le02.html>. Information about the two Danish PAROLE partners in particular is available at [http://coco.ihl.ku.dk/~parole/par\\_eng.html](http://coco.ihl.ku.dk/~parole/par_eng.html) (Det Danske Sprog- og Litteraturselskab) and <http://www.cst.ku.dk/home.html> (Center for Sprogteknologi).

i foråret 1990 blev Rungsted Gymnasium inviteret:

<W lemma="i" msd="SP">i</W> <W lemma="forår" msd="NCNSU==D">foråret</W>  
 <W lemma="1990" msd="AC---U=--">1990</W> <W lemma="blive" msd="VADA=----A-"  
 ">blev</W>  
 <W lemma="Rungsted" msd="NP--U=-">Rungsted</W>  
 <W lemma="gymnasium" msd="NCNSU=I">Gymnasium</W>  
 <W lemma="invitere" msd="VAPA=S[CN][ARV]-U">inviteret</W>

## 6. Using the Tagged Corpus as a Testing Corpus

As mentioned above, the tagged corpus material produced through the process described here also represents an integral step in the testing of an automatic tagger/parser, the DAN-CG. The Constraint Grammar (CG) is a knowledge-based system in which every piece of linguistic information is hand-coded in the system by a linguist. The system does not make use of statistical information, and does not apply any kind of automatic learning algorithms on the corpus material. The system is considered to be automatic in that it can take any kind of text as input and deliver PoS and a – shallow – syntactic analysis for the words contained therein as output. [Karlsson 94] is an introduction to CG and its principles, and a specific description of the ENG-CG can be found in [Voutilainen 92].

The foundation of the CG tagging/parsing process is that one of the analyses in any cohort is the correct one, and this reduces the actual process of tagging/parsing to a choice between the analyses in the cohort, or – in CG terms – to a deletion of the contextually wrong analyses.

The input to the DAN-CG is DAN-TWOL material of the kind used as input in the manual disambiguation process as described above. A set of CG rules is applied to this material in order to remove the analyses in the cohort that are considered to be wrong in the context. In the sentence in Fig. 6 one can see that 'nye' is ambiguous between the A SG and the A PL analyses, and 'cykler' between the N PL and V FIN analyses. The contextually wrong analyses are marked in italics.

The CG rules used to disambiguate at present<sup>5</sup> have the appearance as shown in Figs. 7 and 8.

Fig. 6.

Nye cykler ruster hurtigt	
(New bikes rust quickly)	
"<nye>"	
"ny" A POS UK PL UB NOM	
"ny" A POS UK SG BEST NOM	<i>&lt;= see fig. 7.</i>
"<cykler>"	
"cykle" V FIN PRS AKT	<i>&lt;= see fig. 8</i>
"cykel" N FLS PL UBEST NOM	
"<ruster>"	
"ruste" V FIN PRS AKT	
"<hurtigt>"	
"hurtig" A POS INT SG UBEST	

Fig.7.

REMOVE	(A SG NOM)
(1C	N-PL-NOM)
(NOT	*-1 V-TRI/VAL) ;
<i>Remove the A SG NOM from the cohort if the next word to the right (1) definitely (C) is a N PL NOM, and no (NOT) trivalent verbal (TRI-VAL) can be found anywhere (*) to the left (-) starting the search from word 1(1).</i>	

<sup>5</sup> This sentence was constructed as an example, since real texts most often contain more ambiguity. The rules are taken from the present CG rule file, which is changing constantly.

When applying the standard method of developing a CG, one makes use of the testing mode of the CG software which is designed to return a warning whenever an analysis marked <Correct!> is removed from the cohort. With this mode on, one can quantify the output of the CG tagger/parser by counting how many analyses marked <Correct!> are removed by any given rule in the system, compared to how many analyses *not* marked <Correct!> are removed. Hence, if a rule happens to remove the correct A PL analysis of 'nye' above and leaves the incorrect A SG BEST analysis, this would appear in the log file, and the rule can be located and corrected. These testing procedures are needed since the number of rules in a fully developed CG is quite large (> 1000), and the amount of running words needed for developing it is large as well (often 50,000 to 100,000 words). Hence manually evaluating the output would simply be too time-consuming.<sup>6</sup>

Fig. 8.

REMOVE	(V FIN)
(NOT 0	VERB-NO-CS)
(*1C	V-FIN BARRIER CLS-BOUND/POS)
(*-1	(>>>)
BARRIER	V-FIN
OR	CLS-BOUND/POS) ;

*Remove the V FIN analyses from the cohort if the same (0) word is not (NOT) a verbal of the kind that can make a clause without a clause marker (E.g. 'høre' and 'se') (VERB NO CS), and a word whose only reading is that of a finite verb (V FIN) is found somewhere (\*) to the right starting from the first position (1), and not preceded (BARRIER) by anything that can be a clause boundary (CLS BOUN/POS), and - to the left - no finite verbs (V FIN) or possible clause boundaries (CLS-BOUND/POS) are found before (BARRIER) the beginning of the sentence (>>>).*

## 7. Perspective

The co-operative effort between the PAROLE project and DAN-TWOL/Constraint Grammar projects has proved to be very fruitful for both projects. The common tagset and common disambiguation procedure also rendered all the information necessary for both projects, and the parallel performance of the manual disambiguation has greatly improved the quality of the resulting tagged corpus. We hope that the product of our efforts will also be useful for other users in need of a large morphosyntactically tagged Danish corpus in the future.

## Bibliography:

Arndt, H. (1992): "Towards a Syntactic Analysis of Danish Computer Corpora" in *Proceedings of the XIIth Scandinavian Conference of Linguistics*.

Bilgram, T. (1994): *Computerstyret analyse af dansk*, Specialeopgave, Dept. of Linguistics, University of Aarhus, Denmark.

Dansk Sprognævn (1986): *Retskrivningsordbogen* (the Orthographical Dictionary), Dansk Sprognævn. [also as a machine-readable text version].

Dansk Sprognævn (1996): *Retskrivningsordbogen* (the Orthographical Dictionary), Aschehoug.

Garside, Leech, Sampson (eds.) (1987): *The Computational Analysis of English, a corpus based approach*. London & New York, Longman.

<sup>6</sup> The making of the CG-tagger/parser for Danish is still in progress, but the present version can be tested at <http://ling.hum.aau.dk/~bilgram/CG.html>.

Ide, N., D. Durand, G. Priest-Dorman, and J. Veronis (1995) *MULTEXT: Corpus Encoding Standard*, LRE Project 62-050, CNRS. [The most recent version of the CES can be found at <http://www.cs.vassar.edu/CES/>].

Karlsson, F., A. Voutilainen, J. Heikillä, and A. Anttila (eds.) (1994): *Constraint Grammar: a Language-independent System for Parsing Unrestricted Text*. Berlin and New York: Mouton de Gruyter.

Koskenniemi, K. (1983): *Two-level Morphology. A General Computational Model for Word-form Production and Generation*. Publication No. 11, Dept. of Linguistics, University of Helsinki.

Leidner, J. (1997): "Evaluation Taggers for English: Some Evidence" in *CLUE-TR-971101*. Friederich-Alexander-Universität, Erlangen-Nürnberg, Institut für deutsche Sprache und Literaturwissenschaft. This report is available at: <ftp://ftp.linguistik.uni-erlangen.de/pub/reports/CLUE-TR-971101>.

Monachini, M. and N. Calzolari (1996): *Synopsis and Comparison of Morphosyntactic Phenomena Encoded in Lexicons and Corpora. A Common Proposal and Applications to European Languages*, EAGLES Document EAG-LSG/IP-T4.6/CSG-T3.2, EAGLES [This is also available at: <http://www.ilc.pi.cnr.it/EAGLES96/morphsyn/morphsyn.html>].

Ridings, D. (1996): *Text Representation in PAROLE*, MLAP PAROLE 63-386 WP 4.1.3, Göteborg universitet [Also available at: <http://svenska.gu.se/~ridings/textrep/textrep.html>]. Volz, N. and S. Lenz (1996): *Multilingual Corpus Tagset Specifications*, MLAP PAROLE 63-386 WP 4.1.4, IDS, Mannheim.

Voutilainen, A. (1992): *Constraint Grammar of English, A Performance-Oriented Introduction*, Helsinki University Press.

# Linguistics isn't always the answer: Word comparison in computational linguistics<sup>1</sup>

Lars Borin

Department of Linguistics  
Uppsala University  
Lars.Borin@ling.uu.se

## Abstract

String similarity metrics are important tools in computational linguistics, extensively used e.g. for comparing words in a variety of problem domains. This paper examines the sometimes made assumption that the performance of such word comparison methods would benefit from the use of linguistic, viz. phonological and morphological, knowledge. One linguistically naive method and one incorporating a moderate amount of linguistic sophistication were compared on a bilingual and a monolingual word comparison task for a range of languages. The results show the performance, measured as recall and precision, of the linguistically naive method to be superior in all cases.

## 1. Introduction

A good method for string comparison, or *string similarity metric*, is an important and useful item in the computational linguist's tool kit. Its special case, word (or rather: word form) comparison is extensively used in dealing with the following tasks, among others:

- Spelling checking and correction, where it is used to find lexical entries similar to putative misspellings (Kukich 1992; Oflazer 1996);
- Historical linguistics and dialectology, for the reconstruction of earlier language stages and for the subgrouping of related languages or dialects, through the comparison of suspected cognates (Guy 1994; Kessler 1995; Covington 1996);
- Information retrieval, in particular for finding proper names and trademarks (Kukich 1992; Siegfried 1992; Lambert 1997);
- Multilingual corpus linguistics, for sentence and word alignment and for the establishment of translation equivalences on the word level (Simard *et al.* 1992; Melamed 1995; McEnery and Oakes 1996; Tiedemann 1997);
- Extraction of lexical information from monolingual text corpora, where word form comparison can be used for finding morphologically related word forms.

---

<sup>1</sup> The research reported in this paper was carried out within the project *Creating and annotating a parallel corpus for the recognition of translation equivalents* in the research program *Translation and interpreting: A meeting between languages and cultures*, funded by the The Bank of Sweden Tercentenary Foundation. I wish to thank Kamal Khaledzadegan for his help with the transliteration of the Arabic and Persian material, Bo Utas and Carina Jahani for introducing me to the intricacies of the Arabic script, and the anonymous reviewers for helping me to clarify some of the more muddled points of my exposition.



Even though word comparison is normally used in conjunction with other methods specific to each task (such as dictionary lookup for spelling checking, other alignment methods, including incremental translation dictionary construction, for finding translation equivalents etc.), it is ubiquitous enough that it is important to know which of several available methods is the most effective for a particular task. Strangely enough, however, evaluations of the relative efficacy of these methods are scarce in the literature, as a rule. Some notable exceptions are Lambert (1997), who compares edit distance with bigram and trigram Dice scores for assessing the confusability of drug names, McEnery and Oakes (1996), who compare edit distance, truncation (i.e., initial substring matching), and bigram Dice scores for finding translation equivalents in bilingual parallel corpora, and Tiedemann (1997), who compares a variety of substring matching methods, also for finding translation equivalents in a bilingual parallel corpus. Lambert reports recall, fallout (or false positive rate), and accuracy for the methods, McEnery and Oakes give precision scores and estimated recall, while Tiedemann only calculates precision.

From a linguistic point of view, an important dividing line is that between word comparison methods which use some linguistically, *viz.* phonologically and morphologically, motivated comparison metric, and those which do more linguistically 'naive' character string comparisons. The methods compared in the works referred to above all belong in the latter category. In the literature, it is sometimes assumed—perfectly reasonably, in my view—that the performance of certain word comparison tasks would benefit from the use of a more linguistically motivated comparison method (e.g., Brasington *et al.* 1988; Borin 1991; Kessler 1995; Covington 1996). By this is usually understood a non-language-specific method<sup>2</sup>, i.e. the idea is that the *general* performance—across a range of problems and languages—of such a method could be enhanced by the introduction of some linguistic sophistication.

To my knowledge this assumption has not earlier been explicitly tested by experiment. The purpose of the research reported here is to do this, by applying both a linguistically naive and a linguistically more sophisticated string comparison method to the same two word comparison tasks and the same material for a range of languages, and comparing the performance of the two methods on these tasks in terms of precision and recall.

As a representative of the class of linguistically naive methods was chosen a metric that henceforth will be referred to as *LCS*, which is defined as the length of the longest common subsequence (hence the name *LCS*) of the two strings, i.e. the maximum number of exact character matches between the two strings, possibly with non-matching characters in-between, divided by the length of the longer string, thus yielding a real value in the range 0–1. The algorithm for calculating *LCS* is a special case of a well-known string alignment method with a time complexity  $O(nm) \gg O(n^2)$ , i.e. roughly quadratic<sup>3</sup> (Sankoff and Kruskal 1983).

---

<sup>2</sup> Which should exclude spelling checking and information retrieval algorithms incorporating explicitly language-specific pronunciation information. Such language-specific methods are useful and needed, to be sure, but their existence for some languages does not exclude the search for general methods.

<sup>3</sup>  $n$  and  $m$  are the lengths of the two strings. The calculation is simplified by the assumption that, on the average,  $n \approx m$ .

LCS was chosen under the assumption that the results obtained by it will be largely valid for other linguistically naive string comparison metrics described in the literature. These are methods such as

- common substrings, e.g., initial (Simard et al. 1992; McEnery and Oakes 1996; Tiedemann 1997), final (Tiedemann 1997), or in any position (Zhang and Kim 1990);
- edit distance, when it is understood as the special case of Levenshtein distance where all insertions, deletions and substitutions are given the weight 1 (Sankoff and Kruskal 1983);
- *n*-gram comparisons, often expressed as Dice scores<sup>4</sup> (Lambert 1997; McEnery and Oakes 1996).

For the linguistically more sophisticated method, a method described by Covington (1996; henceforth called *COG*, for *COGNate alignment*) was chosen. This is a depth-first algorithm for ranking sound alignments in word pairs, as the first step in the reconstruction of proto-forms by the comparative method used in historical linguistics. The algorithm has exponential time complexity, producing (in the worst case) approximately  $3^{n-1}$  alignments. It works by performing a depth-first enumeration, or search, of all possible alignments of the segments of the two strings with each other<sup>5</sup> or with  $\emptyset$ . Each kind of alignment incurs a 'cost', according to the phonological nature of the segments aligned. The cost assignment scheme is shown in figure 1.

C(onsonant) with identical C	0
V(owel) with identical V	5
short V with long V, or V with S(emi-V)	10
V with different V	30
C with different C	60
completely dissimilar segments	100
segment- $\emptyset$ (a 'skip') after segment- $\emptyset$	40
segment- $\emptyset$ otherwise	50

Figure 1: *COG* cost assignment scheme

Additionally, the configuration  $\begin{array}{c} - a \ \emptyset - \\ - \ \emptyset \ b - \end{array}$  i.e., alternating skips, is not allowed.

Thus, Covington's algorithm uses fairly coarse linguistically relevant features of the word pairs to guide the alignment process, namely the phonological trichotomy V-C-S, i.e., syllabicity, according greater importance to consonants than to vowels or semivowels in determining the preferred alignment, and consequently also in determining the similarity score. There is also some linguistically motivated context sensitivity in the cost assignment scheme, since contiguous skips are preferred (and consequently also contiguous segment-segment alignments), reflecting the morphological fact that morphs tend to be contiguous. Hence, the *COG* method utilizes both

<sup>4</sup> The Dice score for an *n*-gram comparison of two strings is calculated as:  $2C/(A+B)$ , where C is the number of unique *n*-grams common to the two strings, and A and B the total number of unique *n*-grams in each string.

<sup>5</sup> Although the search space is minimized by the heuristic of only allowing a search to continue as long as its accumulated cost is less than the lowest total cost found so far.

phonological and morphological knowledge, admittedly of a fairly coarse and rudimentary kind, in determining the similarity score.

## 2. Material

The material used for the experiments consisted of a small sample from a multilingual parallel corpus under construction at our department. The corpus is made up of articles from *Invandartidningen*, a Swedish periodical appearing in 42 issues yearly, in eight parallel language versions: Arabic, English, Finnish, Persian, Polish, Serbocroatian, Spanish and simple Swedish<sup>6</sup>. We also had access to the Swedish original manuscript version, which is not published as such, but on the basis of which the other language versions are produced.

All language versions except Finnish and simple Swedish were used in the experiments. Six short bilingual parallel texts were prepared, with Swedish as L1, and Arabic, English, Persian, Polish, Serbocroatian and Spanish as L2. The parallel texts were manually aligned at the sentence level, and the Arabic, Persian and Slavic texts were transliterated into a Latin-1 coding devised specially for the purposes of the experiments. Figure 2 shows some statistical data about the texts used, and in figure 3, some of the parallel sentences from the material are shown.

<i>language</i>	<i>sentences</i>	<i>words</i>	<i>words/sentence</i>	<i>=/sentence</i>	<i>corr/sentence</i>
Arabic	28	291	10.39	0	1.32
English	28	336	12	1.18	1.43
Persian	26	329	12.65	0	1.42
Polish	28	292	10.43	1.18	1.04
Serbocr.	28	315	11.25	0.89	1.18
Spanish	28	287	10.25	1.04	1.25
Swedish	28	238	8.5	—	—
<i>average</i>	28	298	10.78	0.71 / 1.07	1.27

*Figure 2: Text statistics. The last two columns show the average number of exact string matches (mostly proper names) per sentence, and the average number of other correspondences per sentence (see section 3), respectively.*

## 3. The experiments

The text material used for the experiments was deliberately kept small, primarily so that it would be possible to calculate the recall of the two methods—i.e., the ratio of the number of found correspondences to the total number of correspondences—a parameter which is important to know in order to compare the methods fairly. Where string similarity metrics are used on large text corpora (e.g., McEnery and Oakes 1996; Tiedemann 1997), it is generally not feasible to calculate recall, other than as an estimate (as is done by McEnery and Oakes), mainly because

<sup>6</sup> The following language name abbreviations are used in this paper:

Ar = Arabic; En = English; Pe = Persian; Pl = Polish; Sc = Serbocroatian; Sp = Spanish; Sw = Swedish.

the total number of valid correspondences must be determined by a human judge, or judges<sup>7</sup>, and the sheer size of most corpora normally precludes this.

The performance of the two word similarity metrics were evaluated for two kinds of computational linguistic problem, namely those of

(1) bilingual word comparison, or the problem of finding word correspondences<sup>8</sup>, i.e., putative translation equivalents, in parallel texts in two languages (for the six language pairs mentioned in the previous section);

(2) monolingual word comparison, or the problem of finding morphologically related words in a text in one language (for five of the languages; Arabic and Persian were not included in this experiment).

These two tasks, of course, are instances of the last two types mentioned in the introduction, i.e., the establishment of translation equivalences on the word level and extraction of lexical information from monolingual text corpora, respectively.

---

<sup>7</sup> Actually, if there existed a method by which recall could be automatically determined in these cases, we would of course use this method instead of the one that we are evaluating for the task at hand. Note also that calculating recall by sampling a smaller part, or smaller parts, of a larger corpus basically reduces to the procedure described here, although using several samples would presumably make the results more reliable.

<sup>8</sup> Often called 'cognates' in the corpus linguistics literature. I prefer the term 'correspondence', mainly because 'cognate' has a well-established use as a term in historical linguistics, which in practice is disjoint with that now being introduced in corpus linguistics; most of the items picked out by methods for finding 'cognates' in bilingual corpora (e.g. by Simard *et al.* 1992, or Melamed 1995) are actually loanwords (e.g. the English-French word pair *fraternity* ~ *fraternité*), i.e. virtually the opposite of cognates in the historical linguist's sense (a real English-French cognate pair, related to the previous one, would be *brotherly* ~ *fraternel*).

	(1)	(2)
<b>AR</b>	<b>l<sup>ntq</sup>l ly rnkby</b>	tqdm lxfr v l <sup>mr</sup>
<b>EN</b>	<b>Move to Rinkeby</b>	<b>Reds and Greens gain ground</b>
<b>PE</b>	<b>bh rynkhby nql mk<sup>n</sup> kny d</b>	srxh <sup>a</sup> v sbzh <sup>a</sup> pyę myrvnd
<b>PL</b>	<b>Przenieśb do Rinkeby</b>	Czerwoni i Zieloni coraz popularniejsi
<b>SC</b>	<b>Preseliti nadleštvo u Rinkeby</b>	Crveni i zeleni na usponu
<b>SP</b>	<b>Trasladarse a Rinkeby</b>	Rojos y verdes adelante
<b>SW</b>	<b>Flytta till Rinkeby</b>	Framåt för de gröna och röda
	(3)	
<b>AR</b>	<b>lys b<sup>mk<sup>n</sup></sup> 83 fy l<sup>m</sup>½t l<sup>tkyr</sup> b<sup>l<sup>tcv</sup>yt</sup> l<sup>c</sup>l<sup>o</sup> °zb l<sup>dymqr</sup>&amp;yt l<sup>jdy</sup>dt</b>	
<b>EN</b>	<b>Eighty-three per cent could not imagine voting for <i>New Democracy</i></b>	
<b>PE</b>	<b>83 drcd °z mrdm nmytv<sup>nnd</sup> tcvr r½y d<sup>dn</sup> bh °zb dmkr<sup>sy</sup> <i>nyyn</i> r<sup>a</sup> dr sr</b>	
<b>PL</b>	<b>bprvr<sup>nnd</sup></b>	
<b>SC</b>	<b>83% ankietowanych nie wyobraża sobie aby mogli poprzeb w wyborach <i>nowá</i></b>	
<b>SP</b>	<b>democracjê</b>	
<b>SP</b>	<b>Oko 83 odsto ne bi moglo da glasa za <i>Novu demokratiju</i></b>	
<b>SW</b>	<b>El 83 por ciento no puede pensarse votar por <i>nueva</i> democracia</b>	
	<b>83 procent kan inte tänka sig att rösta på <i>ny</i> demokrati</b>	

Figure 3: Three parallel sentences from the material (in adapted orthography). Valid correspondences are boldfaced. Cognates excluded by the minimum-length criterion are boldfaced and italicized.

For both sets of word comparison experiments, the texts were normalized: All texts were lower cased, and most vowel accent marks were removed. Additionally, all word tokens shorter than four characters were removed from the texts. Consequently, the comparisons were made on word token pairs where both tokens were at least four characters long. This minimum length criterion, borrowed from Simard *et al.* (1992), excludes mainly function words, which tend not to correspond to each other in the sense understood here, although it turned out that it also excluded some cognates, such as the Swedish word *ny* 'new' and its correspondences in the other languages, except Arabic (which, not being Indo-European, understandably does not share this cognate).

The bilingual word comparison experiments were carried out as follows. Each word in each Swedish sentence was compared with each word in the corresponding sentence in the target language, and the two similarity values (LCS and smallest COG) were calculated. For various thresholds, the string pairs falling above and below the threshold (excluding exact string matches) were compared with a precompiled list of valid correspondences (see below), whereby three important values were obtained:

*P*(recision), i.e. the number of valid correspondences found, divided by the total number of correspondences found;

*R*(ecall), i.e. the number of valid correspondences found, divided by the total number of valid correspondences for the text pair in question;

*Effectiveness*, or *F* ( $= 2 \cdot P \cdot R / (P + R)$ ), a frequently used combination value of *P* and *R* (see van Rijsbergen 1979, ch. 7).

The kinds of correspondences recognized as *a priori* valid in the parallel texts were (easily recognizable) *loanwords* (including proper nouns) and *cognates*, not necessarily belonging to the same part of speech in the two languages, but containing the same number of lexical morphemes<sup>9</sup>, for example:

	<i>L1 (Swedish) word form</i>		<i>L2 correspondence</i>	
Sw-Ar	europa	<i>Europe</i> N	¹l¹vrvby	
Sw-Sp	kritiserar	<i>criticize</i> V	critica	<i>criticism</i> N
Sw-Sc	populärare	<i>more popular</i> A	populärnost	<i>popularity</i> N
Sw-En	skepsis	<i>scepticism</i> N	scepticism	
	slakt	<i>slaughter</i> N	slaughtered	

Only the words in the last correspondence pair in this list (*slakt* — *slaughtered*) are cognates. All the others are loanwords (including proper names).

The monolingual word comparison experiments were carried out in a similar fashion: For each text, a list of its word tokens was prepared. Each item in this list was compared with each of the items following it, and the two similarity values (MEL and smallest COG) were calculated for each word pair thus compared. P, R and F were calculated in the same way as for the parallel texts, but checked against a different list of valid correspondences than in the bilingual experiments, of course. The criteria used in compiling this list were that the correspondences should be morphologically related words, by the morphological mechanisms of (not necessarily productive) *inflection*, *derivation*, or *both derivation and inflection*, but excluding *compounding*, for example:

			<i>corresponds to</i>	
En	democracy		democratic	
	democracy		democrats	
Pl	nowy	<i>new</i> A	wznowieniem	<i>renewal</i> N
	głosowałoby	<i>would vote</i> V	głosy	<i>votes, voices</i> N
Sw	minskar	<i>diminish</i> V	minskning	<i>diminishing</i> N
	företagandet	<i>the business</i>	företagens	<i>the businesses'</i> N
	activity N			

#### 4. Results

Each of the experiments yielded a range of precision/recall values, which (together with the resulting F values) were plotted against the corresponding threshold values<sup>10</sup>, resulting in a number of diagrams like the one shown in figure 4, where the performance of both methods on monolingual Serbocroatian data is displayed.

<sup>9</sup> In practice, this criterion excludes all words containing more than one lexical morpheme, since, out of the languages investigated here, only Swedish and English make extensive use of compounding and only in Swedish are compounds written as one word (hence the low word count for Swedish in the table in figure 2).

<sup>10</sup> In order to make the values obtained by the two methods comparable, COG thresholds were mapped into the LCS threshold range using the following functions:  $(500 - \text{COG}) / 500$  (bilingual case) and  $(230 - \text{COG}) / 200$  (monolingual case)

Sensible LCS threshold values tended to fall in the range 0.45–0.8, in steps of 0.05, while the values for COG varied with the type of problem. Values of 50–250 in steps of 25 turned out to be a suitable range for for the bilingual experiments, with F values peaking for thresholds in the range 150–200, while the monolingual experiments needed less range and finer granularity: 50–150 with step size 10 and the maximal F values occurring in the threshold interval 90–110.

### SERBOCROATIAN MONOLINGUAL DATA

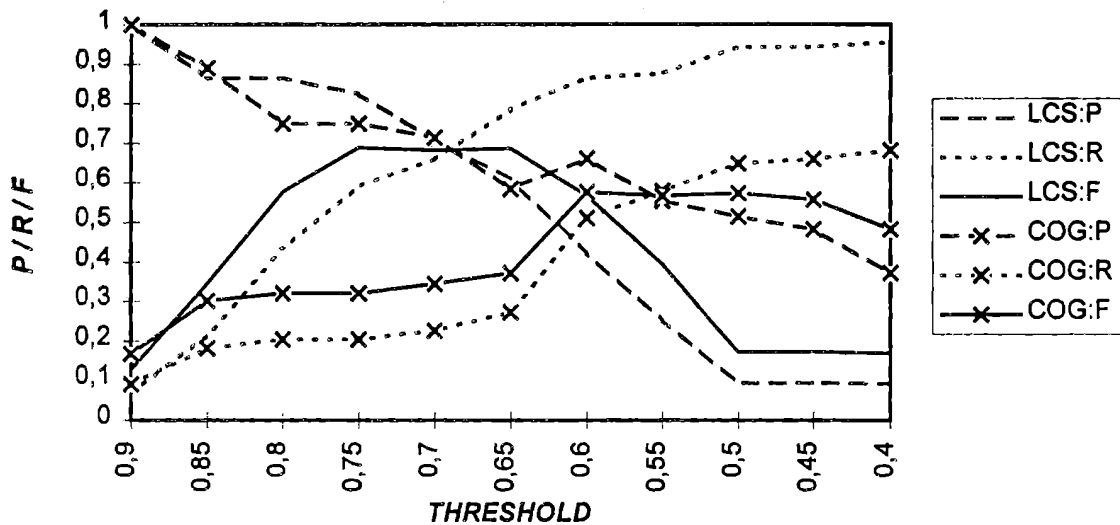


Figure 4: Precision, recall and F for the monolingual Serbocroatian experiment.

For LCS, too, the results were systematically different for the two kinds of problem. We find that maximal F values were obtained for thresholds between 0.6 and 0.5 in the bilingual case, but in the threshold interval 0.8–0.7 in the monolingual experiments. Figure 5 shows the maximal F values obtained in all the experiments.

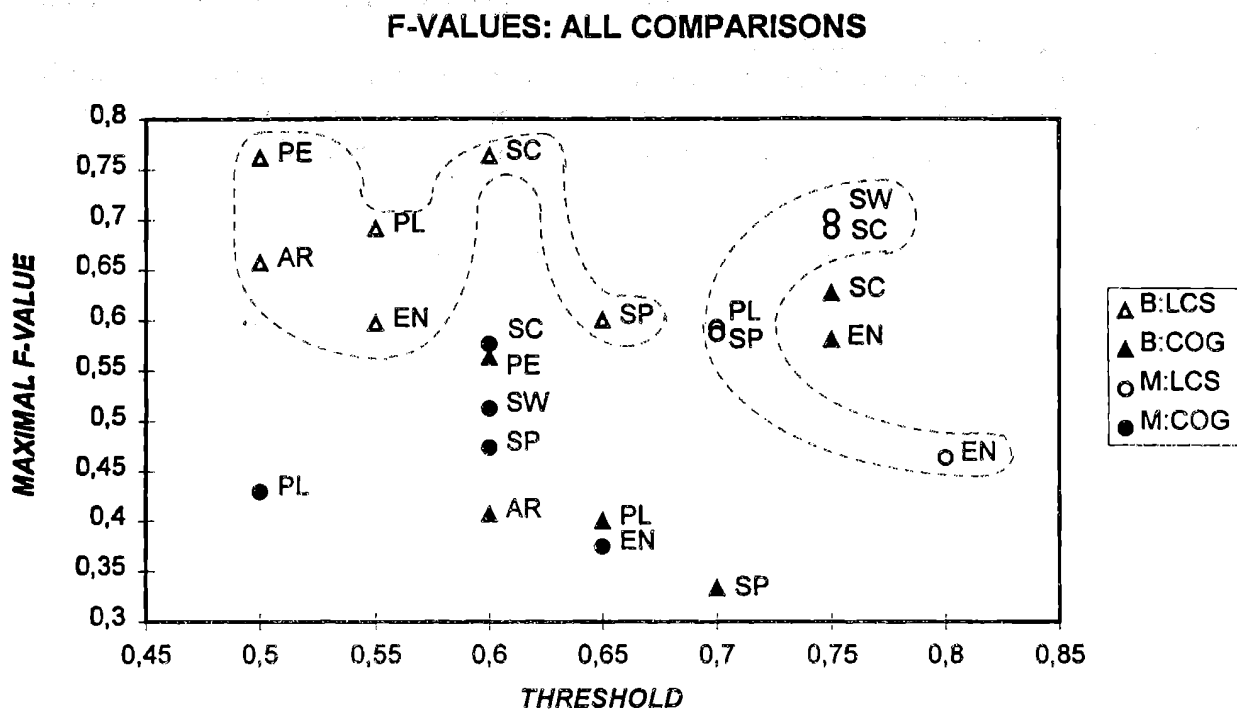


Figure 5: Maximal *F* values obtained in the monolingual (circles) and bilingual (triangles) experiments, for both LCS (unfilled) and COG (filled). The 'LCS regions' are marked with dashed lines.

## 5. Discussion and conclusions

The picture which emerges from the results of the experiments leads to a clear, but from a linguistic point of view somewhat depressing conclusion: LCS performs better overall. This is surprising; the expectation was that COG—with its greater linguistic sophistication—would have shown better performance (or been on a par with LCS), at least for the parallel texts, which are, after all, its 'natural domain' as an algorithm for comparing related words in different languages.

It is true that the material examined is very small, and thus no very reliable statistical correlations can be established on the basis of it. On the other hand, however, for all the language pairs and languages, LCS performs consistently better than COG.

Some conceivable reasons for this could be:

(1) The original formulation of COG uses a coarse phonetic transcription instead the conventional orthography used here. Thus, it could be that the performance of the algorithm is degraded by orthographic 'noise'. This cannot apply for the monolingual experiments, however.

(2) COG was originally devised for a different type of problem—namely that of aligning word forms for historical reconstruction—than the two to which it was applied here. This *should* not



be a problem, since the mechanisms and processes that must be reckoned with are, by and large, the same for all three problem domains (see, e.g., Lass 1977).

(3) There is not enough (or not the right kind of) linguistic knowledge in COG. It has sometimes been suggested, (e.g. Borin 1991; Kessler 1995; Covington 1996) that a distance metric for phonological segments could be used as a cost scheme for linguistic string comparisons. Preliminary experiments were carried out on parts of the text material using a modified version of the COG algorithm, where consonants are compared according to their place and manner of articulation, and vowels according to (binary) phonological feature values. The results were far from encouraging, however; recall increased to some extent, but at the cost of much lower precision scores, with a resulting overall worse performance, i.e. the method found too many invalid correspondences. More experiments with various cost assignment schemes are needed, however, before this idea can be dismissed as not workable.

On the positive side, LCS is the less resource-demanding of the two algorithms, with quadratic complexity, instead of the exponential complexity of COG. We also know that bigram comparisons are about as good as LCS, at least for some word comparison tasks. McEnery and Oakes (1996) find that bigram Dice scores perform slightly better than edit distance (i.e., a method comparable to LCS) for finding translation equivalents in parallel corpora, while Lambert (1997) arrives at the opposite result: edit distance outperforms bigram and trigram comparison on the task of determining the confusability of (American) drug names. Given that the time complexity of bigram comparison is linear in the sum of the lengths of the strings, and given the results presented here, bigram comparison should be the method of choice for many word comparison applications. However, LCS (or a similar method, such as edit distance) should be chosen if there is a need to keep a statistical record of character substitutions, e.g. for adapting the word comparison method to a specific language or language pair and a specific problem domain.

## **6. Future work**

Our future work within this problem area will concentrate on the following issues:

- (i) Test whether the results hold for larger text materials and more languages;
- (ii) Investigate other string comparison methods, e.g. non-symbolic methods such as neural networks, or other automatic learning methods such as that described by Ristad and Yanilos (1996), which have been systematically left out of the preceding discussion;
- (iii) Investigate whether the language or language pair and the kind of correspondence (e.g. loan-word, cognate, inflectionally or derivationally related) shows a correlation with the performance of the comparison methods. The results presented in the previous section do show differences for the two kinds of problem: LCS thresholds were higher, and COG was more sensitive, in the monolingual experiments, but this was not investigated at this time;
- (iv) Pursue further the idea of devising, on linguistic grounds, a more effective cost assignment scheme for the word comparison problems considered here (see the preceding section). This could involve, e.g., additional preprocessing of the text in order to make the orthography more phonological.

## References

- Borin, Lars. 1991. The automatic induction of morphological regularities. Reports from Uppsala University, Department of Linguistics (RUUL) 21.
- Brasington, Ron, Steve Jones and Colin Biggs. 1988. The automatic induction of morphological rules. *Literary and Linguistic Computing* 3(2):71-78.
- Covington, Michael A. 1996. An algorithm to align words for historical comparison. *Computational Linguistics*, 22(4):481-496.
- Guy, Jacques B. M. 1994. An algorithm for identifying cognates in bilingual wordlists and its applicability to machine translation. *Journal of Quantitative Linguistics*, 1:35-42.
- Kessler, Brett. 1995. Computational dialectology in Irish Gaelic. <http://xxx.lanl.gov/{ ps | e-print | format } /cmp-lg/9503002>.
- Kukich, Karen. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24:377-439.
- Lambert, Bruce L. 1997. Predicting look-alike and sound-alike medication errors. *American Journal of Health-System Pharmacy*, Vol. 54, No. 10, pp. 1161-1171.
- Lass, Roger. 1977. Internal reconstruction and generative phonology. *Transactions of the Philological Society 1975*, 1-26. Oxford: Basil Blackwell.
- McEnery, T. and M. Oakes 1996. Sentence and word alignment in the CRATER project. *Using Corpora for Language Research. Studies in Honour of Geoffrey Leech* ed. by J. Thomas and M. Short. London: Longman.
- Melamed, I. Dan. 1995. Automatic evaluation and uniform filter cascades for inducing N-best translation lexicons. *Proceedings of the 3rd Workshop on Very Large Corpora*. Boston, MA.
- Oflazer, Kemal. 1996. Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction. *Computational Linguistics*, 22(1): 73-89.
- van Rijsbergen, C. J. 1979. *Information Retrieval*. 2nd ed. London: Butterworths. [References here are to the electronically available version: <http://www.dcs.gla.ac.uk/Keith/Preface.html>]
- Ristad, Eric Sven and Peter N. Yamilos. 1996. Learning string edit distance. Dept. Of Computer Science, Princeton University, Research Report CS-TR-532-96.  
[=<http://xxx.lanl.gov/{ ps | e-print | format } /cmp-lg/9610005>]
- Sankoff, David and Joseph B. Kruskal (eds.)1983. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Reading, MA: Addison-Wesley.

Siegfried, Susan. 1992. Synoname™: A personal name-matching program for use in the humanities. *Literary and Linguistic Computing*, 7(1): 64-67.

Simard, M., G. Foster and P. Isabelle. 1992. Using cognates to align sentences in parallel corpora. *Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation*, Montreal, 67-81. [References here are to the electronically available version: <http://www-rali.iro.umontreal.ca/Publications/sfiTMI92.ps>]

Tiedemann, Jörg. 1997. Automatical lexicon extraction from aligned bilingual corpora. Diploma thesis in Computer Science. Otto-von-Guericke-Universität Magdeburg.

Zhang, Byong-Tak and Yung-Taek Kim. 1990. Morphological analysis and synthesis by automated discovery and acquisition of linguistic rules. *Proceedings of the 13th International Conference of the Association for Computational Linguistics*, Helsinki, Vol. 2, 431-436.

# Logic for Part-of-Speech Tagging and Shallow Parsing

Torbjörn Lager,  
Department of Linguistics,  
Uppsala University  
Torbjorn.Lager@ling.uu.se

## Abstract

In this paper, a purely logical approach to part-of-speech tagging and shallow parsing is explored. It has a lot in common with reductionist parsing strategies such as those employed in Constraint Grammar (Karlsson et al. 1994) and Finite-State Intersection Grammar (Koskenniemi 1990), but rules are formulated entirely in logic, and a model generation theorem prover is used for part-of-speech tagging and parsing.

## 1 Introduction

In this paper, a purely logical approach to part-of-speech tagging and shallow parsing is explored. It has a lot in common with *reductionist* parsing strategies such as those employed in Constraint Grammar (Karlsson et al. 1994) and Finite-State Intersection Grammar (Koskenniemi 1990), but rules are formulated entirely in first-order logic, and a model generation theorem prover is used for part-of-speech tagging and parsing. For the purpose of demonstrating the approach, a small WWW-based system has been implemented.

## 2 Logic

Recent years have seen several logic programming languages that extend pure Prolog to handle disjunction and various forms of negation. The particular kind of extension used in this paper have clauses of the following form:

$$(1) \quad A_1 ; \dots ; A_k :- B_1, \dots, B_n, \sim D_1, \dots, \sim D_m$$

That is, the consequent of a clause may consist of a disjunction of atomic formulas, and the antecedent may contain negated formulas. Negation here is not classical negation though, but non-monotonic, 'default' negation (like *not* or  $\sim$  in Prolog). For reasons we need not touch upon here, clauses must be *range-restricted*, which means that all variables occurring in a clause must occur in at least one of the positive body atoms  $B_1, \dots, B_n$ .

Explicit negative information can be given as follows:

$$(2) \quad \sim B_1, \dots, B_n, \sim D_1, \dots, \sim D_m$$

This logic has the expressive power of full first-order logic, and the non-monotonic operator only adds to that. A collection of such clauses is called a *disjunctive normal logic program*.

It is well-known that every *model* of a logic program can be represented by a set of ground atomic formulas. Furthermore, a model  $M$  is a *minimal model* of a theory  $T$  if there exists no other model of  $T$  which is included (in the set-theoretic sense) in  $M$ . Whereas a theory in the

language of pure Prolog always has exactly *one unique* minimal model, a disjunctive theory in general does not (cf. e.g. Fernández & Minker 1992).

The *minimal model state* of a theory  $T$  is the set of positive ground disjunctions all of whose minimal models satisfy  $T$ . Thus it provides a very compact representation of a set of minimal models.

### 3 Input

Let  $yield(s)$  denote the description of an input string of words  $s$ . For example,  $yield("he can can a can")$  consists of the following clauses:

(3) `word(1-2,he). word(2-3,can). word(3-4,can). word(4-5,a). word(5-6,can).`

Note that the forms of the words, as well as their relative positions, are encoded in these clauses.

### 4 Grammar

A grammar can be conceived of as a set of *constraints*, and constraints can be expressed by clauses. In the approach advocated here, *lexical entries* are encoded as clauses, often with disjunctions in their consequents, as in (4).

(4) `cat(S,v(nonsg3,inf)) ; cat(S,n(sg)) ; cat(S,aux(fin)) :- word(S,can).`

This clause can be paraphrased as “every instance of the word ‘can’ is a noun or a verb or an auxiliary”.

Other constraints express negative information, and tend to look at the *local* context of the word occurrences to be disambiguated. Here’s an example:

(5) `:- cat(P0-P1,det), cat(P1-P2,v(_,_)).`

This sentence can be paraphrased as “determiners and verbs, in that order, do not occur together”.

Constraints can also be written that ‘tag’ occurrences of words with syntactic functions, and which perform disambiguation on that level too. Also, by means of a recursive definition of “zero or more words”, words which are not strictly adjacent may be related, e.g. as follows:

(6) `:- fun(P0-P1,subj), zero_or_more_words(P1-P2), fun(P2-P3,subj).`

This clause expresses the constraint that a sentence may not contain more than one subject.

### 5 Part-of-Speech Tagging and Parsing

Let  $T$  be a grammar, correct but not necessarily complete, in the form of a set of constraints. Let  $yield(s)$  be a set of constraints deriving from the input string of words  $s$ . Then the notion of *grammaticality* can be defined as follows:

**Def. 1:** An input string of words  $s$  is grammatical if the sentencehood of  $s$  is a logical consequence of  $T \dot{\cup} yield(s)$ , else the grammaticality of  $s$  is *unknown*.

Notice that if the sentencehood of  $s$  cannot be demonstrated, we really don’t know whether  $s$  is grammatical. Yet, this kind of strategy is often supplemented with the *closed world assumption*,

and with an *indirect* definition of ungrammaticality based on a negation by failure inference step, in which the negative is traded for the unknown:

**Def. 2:** An input string of words  $s$  is ungrammatical if it cannot be shown to be grammatical in the sense of Def. 1.

Parsing by means of traditional generative grammars can be performed within such a framework (cf. Johnson 1994). In contrast, the *reductionist approach* can be seen as based on a notion of *ungrammaticality* defined as follows:

**Def. 3:** An input string of words  $s$  is ungrammatical if  $T \dot{E} \text{ yield}(s)$  has no model, i.e. if *false* is a logical consequence of  $T \dot{E} \text{ yield}(s)$ , else the grammaticality of  $s$  is *unknown*.

This definition can be supplemented with the following indirect definition of *grammatical*:

**Def. 4:** An input string of words  $s$  is grammatical if it cannot be shown to be ungrammatical in the sense of Def. 3.

In the traditional approach, represented by the definitions 1 and 2, everything not explicitly allowed (or 'licensed') is forbidden, whereas a reductionist grammar, based on the definitions 3 and 4, allows everything which it does not explicitly disallow.

The difference can be seen very clearly in the extreme case where we have *no grammar at all*. Then, in the traditional approach, since no input string of words can be parsed, *every* such string is deemed ungrammatical, whereas in the reductionist approach, *no* input string of words is deemed ungrammatical.

## 6 Output

If *false* cannot be demonstrated, then the grammar has at least one minimal model which shows that the sentence is allowed by the grammar and which encodes the linguistic properties of the input string of words. That is, a well-formed linguistic representation is any structure that satisfies the constraints.

There are two equivalent ways in which linguistic structures can be represented: as a set of minimal models, where each model represents a structure that 'survives' the constraints, or as a minimal model state (i.e. as a set of disjunctions of ground atomic sentences).

## 7 Examples

### 7.1 Part-of-Speech Tagging

The above ideas can be applied to part-of-speech tagging. A yield representing the sentence "he can can a can" as follows:

(7) word(1-2,he). word(2-3,can). word(3-4,can). word(4-5,a). word(5-6,can).

and the relevant lexicon entries as follows:

(8) cat(S,pron(sg3)) :- word(S,he).  
cat(S,det) :- word(S,a).  
cat(S,v(nonsg3,inf)) ; cat(S,n(sg)) ; cat(S,aux(fin)) :- word(S,can).

has no less than twenty-seven minimal models. We then add the following constraints (which all seem reasonably true) to the theory:

- (9) :-cat(P0-P1,det), cat(P1-P2,v(,\_)).  
 :-cat(P0-P1,det), cat(P1-P2,aux(\_)).  
 :-cat(P0-P1,pron(\_)), cat(P1-P2,n(sg)).  
 :-cat(P0-P1,pron(sg3)), cat(P1-P2,v(nonsg3,\_)).  
 :-cat(P0-P1,aux(\_)), cat(P1-P2,n(sg)).  
 :- cat(P0-P1,aux(fin)), cat(P1-P2,aux(fin)).

Computing the set of minimal models, we find that only one minimal model remains.

- (10) {word(1-2,he). word(2-3,can). word(3-4,can). word(4-5,a). word(5-6,can).  
 cat(1-2,pron(sg3)). cat(2-3,aux(fin)). cat(3-4,v(nonsg3,inf)). cat(4-5,det).  
 cat(5-6,n(sg)). }

This model encodes the part-of-speech of the words in the input string. Thus the constraints serve well to discard certain minimal models and, by doing this, disambiguate the input string of words on the level of part-of-speech.

## 7.2 Shallow Parsing

Now, let us consider an example from syntax. Given the representation of the Swedish sentence “lisa älskar pelle” (English = “lisa loves pelle”) in (11), the lexical entries in (12), the rules for assigning syntactical functions to part-of-speech in (13), and the negative information in (14), there are two minimal models, displayed in (15) and (16), respectively.

- (11) word(1-2,pelle). word(2-3,älskar). word(3-4,lisa).
- (12) cat(S,pn) :- word(S,pelle).  
 cat(S,pn) :- word(S,lisa).  
 cat(S,pron(nom)) :- word(S,hon).  
 cat(S,v) :- word(S,älskar).
- (13) fun(S,subj) ; fun(S,obj) :- cat(S,pn).  
 fun(S,subj) :- cat(S,pron(nom)).  
 fun(S,pred) :- cat(S,v).
- (14) :- fun(P0-P1,subj), zero\_or\_more\_words(P1-P2), fun(P2-P3,subj).  
 :- fun(P0-P1,obj), zero\_or\_more\_words(P1-P2), fun(P2-P3,obj).
- (15) { word(1-2,lisa). word(2-3,älskar). word(3-4,pelle). cat(1-2,pn).  
 cat(2-3,v). cat(3-4,pn). fun(1-2,subj). fun(2-3,pred). fun(3-4,obj). }
- (16) { word(1-2,lisa). word(2-3,älskar). word(3-4,pelle). cat(1-2,pn).  
 cat(2-3,v). cat(3-4,pn). fun(1-2,obj). fun(2-3,pred). fun(3-4,subj). }

In Swedish, this ambiguity is actual, since these two analyses are both possible. However, analyses assigning the same syntactic function to both names are ruled out by the description, as they should be.

These two analyses can be ‘packed’ into one minimal model state, as follows:

- (17) { word(1-2,lisa). word(2-3,älskar). word(3-4,pelle). cat(1-2,pn).  
 cat(2-3,v). cat(3-4,pn). fun(1-2,obj);fun(1-2,subj). fun(1-2,obj);fun(3-4,obj).  
 fun(1-2,subj);fun(3-4,subj). fun(2-3,pred). fun(3-4,obj);fun(3-4,subj). }

Thus, a minimal model state can always provide a dense, ‘underspecified’ representation of analyses represented as a set of minimal models.

Finally, note that the sentence “hon älskar pelle” would only receive one analysis, since the nominative form of the personal pronoun “hon” (English = “she”) disambiguates the sentence on the level of syntax.

### 7.3 More Complex Constraints

The constraints we have seen so far have been fairly simple, so let’s consider a more complex one. In (Karlsson et al. 1994, p. 59) the following example of a constraint is given:

(18) (“<for>“ =! CS (NOT -1 VFIN)(1 TO)(2 INF)(\*3 VFIN))

The idea behind this constraint is that “the subjunction reading (CS) of the word-form “<for>“ is correct if the preceding word is not a finite verb, if the base form of the next word is “to”, if the word after this is an infinitive, and if there is a finite verb in or rightwards of positions 3.”

Here’s one way to write this in logic:

(19) cat(P1-P2,cs) :-  
 word(P1-P2,for),  
 ~cat(P0-P1,v(fin)),  
 cat(P2-P3,to),  
 cat(P3-P4,v(inf)),  
 zero\_or\_more\_words(P4-P5),  
 cat(P5-P6,v(fin)).

Note that this is a positive constraint. Note also the use of a negative condition, made possible by the inclusion of negation in the language, and the use of the zero-or-more-words predicate, introduced above.

Lets test this rule on a real-world example. In the *Brown corpus* (Francis & Kucera 1982), there is one (but only one!) example that satisfies the conditions of this rule.

*Without saying so, she was really grateful; for to attend the dying was something she had never experienced ...*

Given a representation of the relevant part of this text as in (20), the lexicon entries in (21), and the constraints in (19) and (22), we are able to compute the minimal model state in (23).

(20) word(1-2,for). word(2-3,to). word(3-4,attend). word(4-5,the). word(5-6,dying).  
 word(6-7,was). word(7-8,something). ...

(21) cat(S,cs) ; cat(S,prep) :- word(S,for).  
 cat(S,to) ; cat(S,prep) :- word(S,to).  
 cat(S,v(ing)) ; cat(S,n) :- word(S,dying).  
 cat(S,det(def)) :- word(S,the).  
 cat(S,v(fin)) :- word(S,was).  
 cat(S,v(inf)) :- word(S,attend).  
 cat(S,pron) :- word(S,something).

(22) :- cat(P0-P1,prep), cat(P1-P2,v(inf)).  
 :- cat(P0-P1,det(\_)), cat(P1-P2,v(\_)).



(23) {word(1-2,for). word(2-3,to). word(3-4,attend). word(4-5,the).  
word(5-6,dying). word(6-7,was). word(7-8,something). cat(1-2,cs).  
cat(2-3,to). cat(3-4,v(inf)). cat(4-5,det(def)). cat(5-6,n). cat(6-7,v(fin)). }

One final thing worth noting about this example is that there is a lot of interaction between constraints, but that the order in which they interact does not matter at all.

## 8 Implementation

On top of the model generation theorem prover *DisLog* (Seipel & Thöne 1994), which is capable of generating the set of minimal models or the minimal model state which satisfies a disjunctive normal logic program, I have built a simple but user-friendly World Wide Web application which tailors the *DisLog* prover to the task of tagging and parsing with grammars encoded in logic. The entry page is at <http://www.ling.gu.se/~lager/ICCG/iccg.html>.

## 9 Summary and Conclusion

In the *logic grammar* tradition, phrase structure grammars as well as unification-based extensions of phrase structure grammars are viewed in the light of the two equations “grammar = theory” and “parsing = theorem proving”. This view is simple, natural and in many other ways attractive: Formal logic has an old tradition and is well-understood; logic is close to natural language, it comes with a clear notion of truth, and is declarative in the strongest possible sense. Also, properties like consistency and monotonicity have been extensively studied in logical frameworks. Indeed, if there is a *lingua franca* of knowledge representation, it ought to be formal logic.

Reductionist grammars – grammars that use mostly negative and usually local constraints to cut away ambiguities introduced by a lexical/morphological analysis phase – constitute an interesting alternative to traditional phrase structure grammars, and have been developed for several languages.

In this paper I have tried to take a logic grammar view on reductionist grammars. I hope to have shown that this view is conceptually simple and transparent, and well worth to explore. It remains to develop the basic ideas further, to scale up, to assess its practical merits and disadvantages, and to compare it with other approaches when tried on real corpora.

By trying to ‘reconstruct’ reductionist grammars in logic, we may begin to see why and how they work, if and when they deserve to be called ‘declarative’, what we could/should mean when we speak of them as ‘inconsistent’, ‘redundant’ or ‘robust’, etc. We may also be able to improve our understanding of how reductionist parsing relates to other kinds of methods such as parsing with phrase structure grammars, and tagging with statistical methods. For example, since phrase structure grammars can be expressed in logic, it would be interesting to elaborate on the relation of reductionist grammars to phrase structure grammars, and also, to try to combine the two.

The use of a *minimal model state* for representing ambiguity in a principled and very compact way appears to be related to work by Dymetman (1997), where an ambiguous representation of the grammatical properties of an input string of words is seen as a specialisation of the grammar

for that particular input string.<sup>1</sup> This indeed is exactly how a minimal model state is related to the kind of grammars explored in the present paper.

As far as scaling up is concerned, the logical approach is hopefully sufficiently similar to Constraint Grammar and Finite-state Intersection grammar to allow large portions of such grammars to be converted into logic. Moreover, since everything is expressed in logic, there may be off-the-shelf methods – such as these developed within the field of Inductive Logic Programming (Muggleton 1992) – for automatically acquiring constraints by learning from a corpus.

As regards efficiency, it remains to be shown that a logic-based reductionist grammar could ever become practical as a method for tagging or parsing. There is a well-known trade-off between expressive power and computational complexity that always threatens to make particular uses of such powerful formalisms intractable. On the other hand, there are, as far I know, no results that show that the specific kind of theories that I have in mind here actually exercise the worst case complexity of the general case. Be that as it may, since in any case, logic may still have a role to play as an analytical tool for investigating how different kinds of knowledge about language can be brought to bear on the problem of part-of-speech tagging and shallow parsing.

### Acknowledgements

This work was conducted within the TagLog Project, supported by NUTEK and HSFR. I am grateful to my colleagues at Uppsala University and Göteborg University for useful discussions, and in particular to Joakim Nivre in Göteborg.

### References

- Dymetman, M. (1997) Chants, Interaction-free Grammars, and the Compact Representation of Ambiguity, *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence NAGOYA, Aichi, Japan, August 23-29, 1997*.
- Fernández, J. A. & Minker, J. (1992) Disjunctive Deductive Databases, *Proceedings of the Logic Programming and Automated Reasoning Conference*.
- Francis, W. & Kucera, H. (1982) *Frequency Analysis of English Usage*. Houghton Mifflin.
- Johnson, M. (1994) Two Ways of Formalizing Grammars. *Linguistics and Philosophy* 17(3).
- Karlsson, F., Voutilainen, A., Heikkilä, J. & Antilla, A. (1995) *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton De Gruyter.

---

<sup>1</sup> Dymetman's approach is in turn related to methods proposed by researchers working in the LFG framework. The connection to LFG and thus indirectly to the work of Dymetman was kindly suggested to me by an anonymous reviewer of a draft version of the present paper.

Koskenniemi, K. (1990) Finite-state parsing and disambiguation. In Hans Karlgren (ed.) *COLING-90. Papers presented at the 13<sup>th</sup> International Conference on Computational Linguistics*, Vol. 2, Helsinki, Finland.

Muggleton, S. (Ed.). (1992) *Inductive Logic Programming*, Academic Press.

Seipel, D. & Thöne, H. (1994) DISLOG - A System for Reasoning in Disjunctive Deductive Databases, In: Antoni Olivé (Ed.): *Proceedings of the Fifth International Workshop on the Deductive Approach to Information Systems and Databases*. September 19-21, 1994, Aiguablava, Costa Brava.

# A statistical and structural approach to extracting collocations likely to be of relevance in relation to an LSP sub-domain text

Bjarne Blom

Department of Lexicography and Computational Linguistics  
The Aarhus Business School  
bb@lng.hha.dk

## 1. Background.

This article sketches a method by means of which likely text-relevant collocational word strings may be extracted from sub-domain LSP-texts. Because of the repetitive nature of collocations, a statistical and structural approach is suggested. The goal of this project is two-fold: (a) to explore the degree to which computational methods are suitable for extracting collocations; (b) to explore the degree to which it is possible to extract information particular to the overall topic or domain of a LSP text, i.e. terminology, by means of knowledge-poor techniques. As this study has run on a time-limited basis, most importance has been placed on (a).

## 2. What is understood by "collocation".

A "collocation" is often defined as either "an arbitrary and recurrent word combination" (Benson 1990) or "the cooccurrence of two or more words within a short space of each other" (Sinclair 1991). However, such definitions tend to leave two important questions out of consideration, cf. the two following examples:

- (a) ... shouldn't have done *that*. *This* prompted.....
- (b) ..... seems unlikely *that this* should be the case

The two definitions above, which are in effect fairly representative of the major part of definitions offered, do not take two points into considerations, namely the distinction between lexical collocations like *prime minister* and syntactic collocations like *that this* - or the issue whether words should be allowed to cross sentence boundaries. Relying on either of the above definitions would result in massive overgeneration, so consequently we will define the concept of "collocation" as to better suit a quantitative approach:

A collocation is a word string consisting of a minimum of two words<sup>1</sup> with the following characteristics:

---

<sup>1</sup>This is a brief description of the method designed and the findings made in connection with the research project UDOG (Udforskelse af Dansk Ordforråd og Grammatik - *Exploration of Danish Vocabulary and Grammar*) under the auspices of The Danish Research Council for the Humanities. A broader outline is given in Danish in Blom 1997 and (1998 - forthcoming).

<sup>1</sup>For the purposes of this study, we will impose the restriction on the minimum word string length that it must contain at least one content word, cf. section six.

- (1) the entire word string is occurring within a given textual segment  $a$ .
- (2) the least frequent word(s) in the word string occur(s) at least  $b$  times in the corpus.
- (3) all words in the word string occur with a span of  $c$  words to its neighbour collocate
- (4) all words in the word string occur in  $d$  particular sequences

This definition is rather a set of parameters to be adjusted according to the particular scientific context in which they are to be utilised, thus (1) refers to whether a word string is considered to be a phrase, a syntagma, an entire sentence, or even permitted to cross sentences boundaries; (2) refers to a lower threshold value decided on for a given purpose in order to filter off infrequent occurrences ; (3) refers to the issue whether interrupted structures should be taken into account; (4) refers to the question whether inverted structures should be taken into account. In this article we let  $a$  be sentence level, i.e. we accept a clausal structure as a collocation rather than imposing some arbitrary value as a maximum word string length; we let  $b$  be four, i.e. a word occurring three times or less is not considered to be significant; we let  $c$  be zero, i.e. we do not include interrupted structures, as such structures require quite an extensive statistical setup (cf. Ikehara, Shirai & Uchino 1996), and such structures are outside the scope of this study; we let  $d$  be left-to-right, in the sense that we do not lemmatize bigram structures, i.e. *afgift betales* = *betales afgift* in order to secure the inclusion of underrepresented items. However, in case inverted structures are significantly represented in the corpus, they will be included as valid collocations.

### 3. Finding likely relevant words (unigram level).

The approach is founded on the assumption that a word has both statistical and grammatical properties which may serve as a clue to its terminologicalness, at least this is believed to be the case for very hard-core LSP texts. As an example of such a text, the Danish VAT act has been chosen. The legislative genre has been carefully selected, as legislative terminology tends to be unambiguous in the sense that one particular concept is usually backed by one particular orthography. As opposed to LGP, the use of synonymy, paraphrases and other stylistic ways of representing a concept orthographically is minimised in order to avoid interpretative scope on the legal circumstances relating to a given concept. As a bonus, we may expect polysemy not to constitute a serious problem for this kind of study.

Zipf (1949) suggested that there seems to exist an inversely proportional interdependence between frequency and rank in a text, in that a very limited number of words occur with very high frequencies, whereas a very large number of words occur with very low frequencies. Damerau (1965), Dennis (1967) and Stone & Rubinoﬀ (1968) discovered a link between frequency, typicality - and word class, so that so-called function words (which tend to account for quite a considerable part of words found at the high-frequency end of the Zipfian distribution) and content words have different statistical behaviour in texts, as the former category follows a poisson distribution closely, whereas the latter does not. These works provide theoretical foundation for the intuition that a word like "that" has a greater intertextual dispersion than a word like "thermodynamics". Bookstein & Swanson (1974) (1975) and Harter (1975) sharpened

this point by offering an explanation why certain content words (like e.g. *red, say, man*) have greater intertextual dispersion than other content words (like e.g. *enterprise, sub-clause, taxable*). Their claim was that words which show a random distribution in a poisson process are likely not to contain information about the text in which they occur, whereas words which do not follow a poisson distribution tend to contain information about the text in which they occur.

For this study, we will rely on the very simple assumption that given the fact that we are dealing with a very specialised sub-domain LSP text, i.e. a legislative genre text within the domain of taxation, text-relevant words are likely to be found among high-frequency content words. For this end, three steps have been necessary: (1) the text has been tagged for word class, (2) words sharing an identical stem have been lemmatised to avoid insignificance arising from the underrepresentation of a given inflected form, and (3) a very simple and limited stop list has been applied to filter off spatio-temporal items (eg. names of various EU countries or words referring to a deadline by which a given task is to be carried out), which tend to be represented with some frequency owing to the nature of the text, however is likely to be irrelevant for the particular legal domain in question. In order to determine a suitable threshold value for filtering off words with no or very little relevance, a number of sub-sections of the VAT act have been randomly picked in order to compare the frequency of occurrence of words in the sub-section with their frequency of occurrence in the entire text. This was done in order to find a suitable cut-off point below which words are likely not to be of relevance to the text (for an example of this, see Blom 1997), and there seemed to be a case for omitting items below the frequency of four.

#### **4. Finding words with collocational potential (unigram level).**

The output consists of a list of unigrams rendered text-relevant<sup>2</sup> by the method. What we need to find out now is which of these unigrams tend to occur as either single words or as part of a multi-word unit.

There are existing statistical methods for testing the "bondness" of word pairs, the most prominent one being mutual information, cf. Church & Hanks (1990). Mutual information is a widely used frequency-based formalism that calculates the probability whether a word pair occurs together or separately in a text. However, this approach is not adequate for this study, as we place our focus on content words. In mutual information statistics, both words of a bigram receive equal weight irrespective of grammatical status. This would mean that function words might exercise undue influence over the overall MI-value and thus bias the measure. This is no unreasonable assumption, if we consider the fact that the most frequent words in a Zipfian distribution tend to be function words. In this study, we restrict our scope to keeping content words as "nodes" and then examine the way they combine with their either left or right adjacent "collocates".

We will use the heuristics that a content word's ability to enter into a "bond" with another word depends on its contextual distribution, i.e. its number of adjacent either left or right words in a text. The fewer such adjacent words, the more a word's unigram frequency will be distributed upon *particular* collocates. In case a content word has a large number of collocates, its unigram

---

<sup>2</sup> Future versions of the method will try to incorporate more sophisticated heuristics such as the way a given content word is distributed over the entire number of sentences, or the number of content words used to describe a given content word in the entire text.

frequency will be watered out. Consider a content word which occurs fourteen times in a text and has an equal number of collocates, then each bigram gets a frequency of one. On the basis of these statistics, we can safely assume that this content word definitely constitutes a single word unit. If, on the other hand, a content word occurring fourteen times is represented two times with one particular adjacent word and twelve times with another particular adjacent word, we can deduct that this content word is predominantly part of a multi-word unit. For the purposes of this heuristics, we apply the following formula in which  $u$  is the unigram frequency of the content word and  $c$  is the number of collocates of the content word:

### **5. Finding word-pairs with collocational strength (bigram level).**

Our present output consists of the unigram list from the previous step exclusive of words which the method does not render potential multi-word items. We will now explore the above idea a bit further, as we will focus on the strength of any given bigram in which a particular content word occurs in order to find the strength of the bond of the word-pair. The bond strength depends on the number of times a word occurs in a given bigram compared to how often the word occurs as a unigram. Our assumption is that *content word n* exercises good "attractive power" on *word x* if the bigram frequency ( $f_{qnx}$ ) accounts for a major part of the unigram frequency of content word  $n$  ( $f_{qn}$ ). We will try to make this statistical parameter differentiate between words with the same unigram/bigram-ratio, but with different frequencies, eg. the instance where two words with ten collocates each occur with a unigram frequency of ten and a hundred, respectively. We will use this formula, in which  $u$  is the unigram frequency and  $b$  is the bigram frequency, to discriminate between superior and less superior collocational strength:

### **6. Expanding the bigrams (sentence level).**

The output from the last two steps consists of a list of bigrams which are, on the basis of the statistics applied, considered to be of a multi-word nature rather than a single-word nature. Now we will expand each bigram to find each possible word string in which the bigram occurs. In stead of merely extracting the longest possible bigram like for instance Smadja (1993), we will permit all syntactically well-formed word strings in which a candidate bigram occurs. To this end a contextual array is applied, in which a given bigram is shown in all its contexts, so that the horizontal axis accounts for syntagmatic values, while the vertical axis accounts for paradigmatic values. The following rules apply to the expansion of orthographic word strings: (1) Only word strings (non-expanded as well as expanded bigrams) occurring at least three times are accepted; (2) A bigram may be expanded by  $\pm$ one position only if the same word occurs at the same syntagmatic position in the contextual array; (3) A bigram is left out of consideration in case a longer word string occurs with at least the bigram frequency minus one occurrence. This is to avoid syntactically incomplete structures, the frequent occurrence of which can be ascribed to syntactic reasons only. The longer word string which the same or nearly the same frequency as the bigram is likely to be the valid one. An easier solution would be to favour certain syntactic structures, typically noun syntagmas or predicative-like structures like Smadja (1993), or to limit the study to bigrams containing only content words, however this would idiosyncratically favour *betale afgift* {C-C} to *betaling af afgift* {C-F-C}.

In a Zipfian distribution of words, the high-frequency end is dominated by grammatical words

like prepositions, conjunctions and the like, which is why a great amount of syntactic collocations is to be expected in any given bigram output. In order to reduce the likely amount of overgeneration, we will take into account only bigrams that contain at least one content word and which does not include a punctuation mark. Having taken this preventive step, we will assume that syntactic noise is likely to occur at the left- and rightmost positions in a word string where a function word may be found. In stead of merely leaving an expert with the tedious task of filtering off items irrelevant owing to syntactic noise, we will apply a set of syntactic rules to serve as a filter for such syntactic noise.

There are four syntactic main rule classes (1-3), one syntactic exception rule class (4) and one statistical exception rule class (5) as follows: (1) a particular tag (or tag sequence) cannot take up the leftmost slot (or slots) of a word string; (2) a particular tag (or tag sequence) cannot take up the rightmost slot (or slots) of a word string; (3) a particular tag sequence cannot form either an entire word string or a part of a given word string counting from the second left- or rightmost slot; (4) a particular tag sequence discharged by a syntactic rule may qualify, if it matches a specific syntactic pattern; (5) a particular tag sequence left out by a syntactic rule may qualify if a content word is immediately succeeded and/or preceded by a preposition in at least 80% of all cases. The latter rule is used to statistically identify phrasal verbs (*berettige til*), as well as complex prepositions (*i henhold til*). All occurrences of {vb-prep} and {prep-sb-prep} have been examined in order to evaluate the performance of this 80%-rule. All instances of phrasal verbs and complex prepositional phrases are correctly identified. Consequently, this technique might even prove to be a theoretical spinoff of this project.

Example of a Class (1) rule:

input: "den i stk. #num#" {det-prep-noun.abbr-num}  
 main rule: a word string cannot be initiated by tags {det-prep}- reduce word string by these words.  
 output: "stk. #num" {noun.abbr-num}

Example of a Class (2) rule:

input: "registrere som landbrug og" {vb.inf-konj-noun-konj}  
 main rule: a word string cannot be ended by tag {konj}- reduce word string by this word.  
 output: "registrere som landbrug" {vb.inf-konj-noun}

Example of a Class (3) rule:

input: "opgøres på grund af" {vb.pres.pass-prep-noun-prep}  
 main rule: a word string cannot be equivalent to tags {vb.opt<sup>3</sup>.opt-prep-noun-prep} where \*prep-noun-prep\* is a complex prepositional phrase - omit entire word string.  
 output: Ø

Example of a Class (4) rule:

input: "den i stk #num# omhandlede afgift" {det-prep-noun.abbr-num-vb.part.adj-noun}  
 main rule: a word string cannot be initiated by tags {det-prep}- reduce word string by these words.

---

<sup>3</sup>This notation refers to optional filling of slot. The slots may be filled by a tag referring to a verbal tense, voice and mood.



exception: a word string initiated by tags {det-prep} cannot be reduced, if the word string is ended by tags {num-vb.pret.adj-noun}<sup>4</sup>.  
 output: "den i stk #num# omhandlede afgift" {det-prep-noun.abbr-num-vb.part.adj-noun}

Example of a Class (5) rule:

input: "fritage for" {vb.inf-prep}  
 main rule: a word string cannot be ended by tag {prep}- reduce word string by this words.  
 exception: a word string ended by tag {prep} cannot be reduced, if {prep} has in its immediate left position the tag {vb.opt.opt}, and {vb.opt.opt} has in all its lemmatized forms the orthographic representation of {prep} in 80% or more of all cases.  
 output: "fritage for" {vb.opt.opt-prep}

..... mstændigheder	<b>fritage</b>	en virksomhed for.....
..... kan indrømme	<b>fritagelse</b>	for.....
.....er kan meddele	<b>fritagelse</b>	for forhøjelse af.....
.....omhandlende	<b>fritagelser.</b>	d . Spørgsmål 1.....
.....f virksomheder	<b>fritage</b>	for at svare afgift.....
.....arer ville være	<b>fritaget</b>	for afgift efter §.....
.....dre EF-lande er	<b>fritaget</b>	for afgift : 1 ).....
.....heder ville være	<b>fritaget</b>	for afgift . Stk. 5.....
.....ndet ville være	<b>fritaget</b>	for afgift . 2 ).....
..... stk. 1 og 3 , er	<b>fritaget</b>	for at svare afgift.....

Fig. 1. Concordance for the lemma "fritage[-]" (frequency of occurrence: 10) and its immediate right collocates. In eight out of ten cases, the lemma "fritage" has "for" as its adjacent right collocate. In this case, the {prep}tag is deemed to be a particle rather than a preposition, and thus a valid part of a phrasal verb.

## 7. Evaluating the syntactic filter.

The syntactic filter was created on the basis of the orthographic output strings. The output was examined in order to distinguish between word strings which should either be permitted as syntactically well-formed or omitted as syntactically ill-formed. This was done in order to deduct syntactic rules to be generalised with a view to creating the best possible recall-precision rate.

48% of all orthographic word strings were syntactically ill-formed, whereas 52% were well-formed. This 50-50 ratio does certainly imply a certain higher principle of randomness, which might make syntactic formalization impossible. However, if we examine syntactic tag sequences among the group of omitted word strings, we discover that some 96% are actually unique,

<sup>4</sup>This rule aims at including a syntactic pattern particular to the legal domain in Danish in which the head of a nominal phrase is pre-modified by a determiner followed by a preposition.

whereas some 4% share a syntactic sequence with a member of the group of syntactically well-formed word strings. Within the group of omitted word strings with a unique tag sequence, 99% may be subject to formalization, whereas only 1% could not be formalised into any operational rule. This one per cent account for a minor loss of precision (-0,6%). The formalization of the 99% word strings did not conflict with rules applicable to the syntactically well-formed word strings. All of the remaining 4% ambiguous word strings could be formalised, however with a moderate loss of recall (-3,4%). The syntactic filter works with 96,6% recall and 99,4% precision, and provides some indication that syntax usage in sub-domain LSP texts like the present one is of a nature which accommodates formalisation rules.

### **8. How to evaluate the terminological relevance of the word strings extracted.**

The syntactic filter ensures a well-formed output, but this is of course no guarantee that the output meets a certain qualitative standard as to text typicality. How can we know that the word strings extracted are not merely commonplace collocations found in virtually any text. One obvious way of judging the output would be to have an expert evaluate it, however one obvious drawback to this approach is the circumstance that experts tend to have different and sometimes even conflicting opinions. However another method might be to test how the word strings are dispersed on a cross-section of texts from various domains in order to find out whether the same collocations tend appear in one or more other texts, or whether they tend to be restricted to the Danish VAT Act. Obviously, we cannot look for the collocations in all texts ever written, but we can ascertain with statistical certainty whether a given word string is restricted to the VAT Act or not. Evaluation of LSP relevance has been left out of consideration for this present study, but is an interesting issue for further research.

### **References**

- Benson, M. (1990): Collocations and general-purpose dictionaries. *International Journal of Lexicography* 2, pp. 1-14.
- Blom, B. (1997): Om statistisk og strukturel afgrænsning af sandsynlige teksttypiske kollokationer i Momsloven. In: *UDOG-rapport 6*, pp.3-23.
- Blom, B. (1998 - forthcoming): A method for identifying collocations likely to be relevant in relation to a sub-domain LSP-text. *UDOG-rapport 7*.
- Bookstein, A. & Swansson, D. R. (1974): Probabilistic models for automatic indexing. *Journal of the American Society for Information Science*, 25, pp. 312-318.
- Bookstein, A. & Swansson, D. R. (1975): A decision-theoretic foundation for indexing. *Journal of the American Society for Information Science* 26, pp. 45-50.
- Church, K. W. & Hanks, P. (1990): Word association norms, mutual information and lexicography. *Computational Linguistics* 16-1 pp. 22-29.
- Damerau, F. J. (1965): An experiment in automatic indexing. *American Documentation* 16, pp. 283-289.

- Dennis, S. F. (1967): The design and testing of a fully automatic indexing-search system for documents consisting of expository text. In: G. Schechter (ed.) *Information Retrieval: A critical review*, pp. 67-94. Thompson Books Co., Washington.
- Frantzi, K. T. & Ananiadou, S. (1996): Extracting nested collocations. *Proceedings from COLING'96*, pp. 41-46.
- Harter, S. P. (1975): A probabilistic approach to automatic keyword indexing. Part 1: On the distribution of specialty words in a technical literature, Part 2: An algorithm for probabilistic indexing. *Journal of the American Society for Information Science* 26, pp. 197-206; pp. 280-289.
- Ikehara, S., Shirai, S & Uchino, H. (1996): A statistical method for extracting uninterrupted and interrupted collocations from very large corpora. *Proceedings from COLING'96*, pp. 574-579.
- Sinclair, J. (1991): *Corpus, concordance and collocation*. Oxford University Press.
- Smadja, F. (1993): Retrieving collocations from text: Xtract. *Computational Linguistics* 19-1, pp. 143-178.
- Stone, D. C. & Rubinoff, M. (1968): Statistical generation of technical vocabulary. *American Documentation*, pp. 411-412.
- Zipf, G. K. (1949): *Human behaviour and the principle of least effort*. Addison-Wesley, Cambridge, Massachusetts.

# A Reasonably Language Independent, Heuristic Algorithm for the Marking of Names in Running Texts

Benny Brodda  
Stockholm

## 0. Introduction

The identification of names and abbreviations in a raw text is an important subactivity in the "tokenization" process, i.e. the identification of the basic units of the text: paragraphs, sentences and words. Tokenization is in its turn an important subactivity in the "normalization" of texts, the totality of operations and preparations a text has to undergo before it is suitable for being added to a text corpus.

This specific paper deals indirectly with the sentence recognition problem. In most cases a configuration in a text of the type

(1) major\_delim + word\_interspace + up-case\_letter\_+ one or more low-case\_letters;

indicates the end of one sentence and the beginning next, where a "major delim(ite)r" stands for any of the characters '.', '!', '?', ..., and a "word interspace" a non-empty string of "white" characters (spaces and/or a linebreaks). There are, however, quite a few exceptions to this rule, and one important set of such exceptions involves combinations of abbreviations and names, e.g. '...; cf. Brodda 1998' or 'As H.G. Wells writes in ...'. Neither Brodda nor Wells are here sentence openers. There are many other ways sentence breaks may be indicated (or not indicated) in a text, and there are other types of exceptions (or "tricky configurations") but the mentioned configuration in (1) certainly provides for the vast majority of cases.

When starting this project I set for myself a kind of specification: In principle I should be able to take a raw text of any kind and in any language, "throw" it through my algorithms and get the names in it marked. Now, it is never as simple as that. One always has to make at least some preliminary investigations, (finding out which character set is used, which word delimiters there are, if the words are hyphenated, if the texts contain printing codes, etc.) and some preliminary preprocessings (making something useful out of possible printing codes and/or remove them, fuse hyphenated words, perform a preliminary paragraph separation, etc.)

But still, in principle, if these preliminary preprocessings are done, I think I can do precisely this. I have made extensive test for Swedish, where my algorithms work quite satisfactory, and I have performed some preliminary tests on French, Italian and English, with comparable results so far. We have also tested it on Estonian, where the results were not that impressive (but still useful). I have not tested it on German, but that would probably be a waste of time; as long as they insist on spelling their nouns with capital initial letters my algorithms just won't work; Sorry, Germans.

As we will see, for Swedish I can reach a recall rate of at least 98% and a precision rate of about 95+%.

All computer programs used in the experiments reported here are written in the PCBeta programming system; cf. Brodda, 1998, forthcoming.

## 1. Preliminaries

In this paper I will mainly consider **prototypical** names, names that adhere to what I call the **Ux-format**, one upper case letter followed by one or more common letters. Furthermore, prototypical names should have a (**reasonably**) **stable orthography** in the sense that they always retain the Ux-format, and they only take some restricted types of "bending"; in Swedish and most other west European languages quite regularly the genitive, (in Swedish and in the other Nordic languages by just adding an *s* to the name: *Paul*, *Pauls*). Commercial names for physical objects tend to become used as common nouns and are correspondingly flexed: *Saab*, *Saaben*, *Saabar*, *Saabarna*. This type of flexion is a marginal problem, and I will not account for it here.

Of course there are other types of names than the prototypical ones. 'USA', for instance, represents one important type of names, which I will leave out here (they pose a minor problem). More troublesome are names of intellectual works, which names sometimes occur directly in the text without surrounding quotation marks (as they probably should have, if the proof reader had done his/her job properly). 'Dylan writes in *The Times* they are *Achanging*, that ...'. Now, neither of the last three Ux-words have a stable orthography (in an English text). At least the word *The* would certainly appear in the text(s) also as *the*, and correspondingly for the word *Times* (maybe). For the word *Achanging* there is a lesser chance of finding the word *achinging* in the text(s); it would probably be accepted as a name according to my algorithms. Thus, what is accepted as a name may depend on the type of texts we are investigating and the total amounts of them.

The words *The*, *Times* and *Achanging* in the example above are examples of a type which I will call **formal names**, words adhering to the Ux-format and appearing in an **interior position**; at least one word (or correspondingly) away from either a paragraph beginning or the nearest major delimit(er) to its left. The intended meaning of "interior position" is, of course, "sentence interior position", but as I intend to use my name identification procedure(s) as an aid for sentence detection, I can not, at this point, assume that I know which these are. The "interior position"-concept, is, by and large, stronger than any reasonable meaning of "sentence interior". The addition "or correspondingly" above is to account for situations of the type '..., cf., e.g., *Brodda 1998*' where *Brodda* now is interior (',' should not occur in sentence beginnings), whereas this word is not interior in the corresponding example in the Introduction.

Now a few words about "major delimit(er)s" and about "what a word is". If a sentence begins with '10:15 this morning *Benny Brodda* presented his epoch-making theory on name identification' then '10:15' is, of course, a word; it is part of a topicalized time adverbial. If you have an itemized list of the type '10:15 *Benny Brodda* on a name identification method; 11:00 *Kimmo Koskenniemi* on FS-parsing; 11:45 ... (etc.)', then it is not at all clear whether '10:15' should be considered as part of the following sentence or not. (Personally I would say

not). Properly speaking, the time specifications here should normally be delimited by something (a full stop, a colon, or so), but the problem is what you do if they are not.

The characters usually assumed to belong to the set of “major delims” are ‘.’, ‘!’ and ‘?’, but there are other characters which to a varying degree have similar functions, at least in certain situations. The most important of these are ‘:’ (colon) and ‘;’ (semicolon). Especially if these characters occur in configurations like (1), above, then the best procedure is to consider them to be major delims. In Swedish writing the ‘:’ in a configuration of the type ‘... xxx: “Uxx xxx ... .’ most certainly indicates that the Ux-word commences a sentence-like object; this is the ordinary way direct quotations are written. In the procedure described below I will to begin with take the conservative position that a quote-in character always is equivalent to a major delimiter, if the quoted stretch of text comprises more than one word. Dash, ‘—’, finally, in ASCII-texts often occurring as an ordinary hyphen ‘-’ surrounded by white characters, is again a character with somewhat unstable use as a sentence begin/end indicator; I assume that it is, if it occurs before an Ux-word.

## 2. The test texts

The big idea with my investigation is to (pre)process raw text, with no markings in it to begin with. In order to be able to evaluate my name identification algorithm, I needed, however, a text with the names already marked. Luckily, we had precisely that at my department, viz. a substantial - though preliminary - subset of the so called SUC-corpus (SUC = the Stockholm-Umeå Corpus; cf. Källgren: [www.ling.su.se/forskning/SUC/distr&anv.htm](http://www.ling.su.se/forskning/SUC/distr&anv.htm)). The texts in SUC are SGML-marked (according to the Text Encoding Initiative conventions), meaning that they contain a lot of spurious information for my specific purposes, so I extracted from it the text itself together with (a transformed variant of) the name tags. I also kept the paragraph structure (essentially in the form of an empty line after each paragraph.) Here follows a small but rather typical sample of such a text. (The asterisks do not occur in the SUC-text itself. They are the results of my algos.)

```
aa03a 035 bådas framtida toppmöten kommer att minska .+ Med hänvisning
aa03a 036 till att relationen mellan [ USA ] och [ *Sovjetunionen ] nu
aa03a 037 undergått avgörande "kvalitativa" förbättringar beslöt sig
aa03a 038 [ president *George *Bush ] och *Sovjetledaren [ *Gorbatjov ]
aa03a 039 från och med nu att hålla regelbundna , sannolikt årliga ,
aa03a 040 ... +
```

Fig. 1: A text sample from one of the input texts.

“aa03a” in the line beginnings is the name of the corresponding (sub)text in SUC, and “035, 036, etc.” are the (temporary) line numbers within that text I use during my experiments. The ‘+’-signs in the text above are what remain of the sentence-in and sentence-out tags in SUC; here they are strict sentence end markers. The ‘[’ and ‘]’ characters correspond to the name-in and name-out tags of SUC, respectively. My name marking programs just disregard the characters mentioned in this paragraph.

Altogether I have now a corpus of some 300,000+ words, but for the experiment reported here I have used a subset of some 77,000+ running words.

When evaluating my procedures I do, of course, take the name-in and name-out characters into account - they delimit a "name region" - and I can immediately see which words are "false hits", "true hits" and "misses". The last two concepts are defined relative to what I call **SUC-names**, Ux-words occurring inside a name region. **True hits** (or just hits) are the marked SUC-names and **misses** the non-marked ones. (NB! 'USA' in Fig. 1 is **not** a miss; cf. last para of this section). The **false hits** are the asterisked words outside any name region (e.g., '\*Sovjetledaren', "the Soviet leader"; in Swedish a compound like that may or may not be spelled with a capital first letter, but either way it should not be counted as a name. (The English counterpart is also a compound, but I think it is not entirely wrong to consider the word 'Soviet' in it to be a name.)

In the evaluations of my algorithm(s) I use the standard descriptive statistics **recall** and **precision** - originally emanating from the I/R-world - and (here) defined as:

$$(2) \quad \text{recall} = \frac{\text{number\_of\_hits}}{\text{number\_of\_SUC-names}};$$
$$\text{precision} = \frac{\text{number\_of\_hits}}{\text{number\_of\_hits} + \text{number\_of\_false\_hits}};$$

Both these statistics are commonly expressed in percent: 100% recall (here) = all the SUC-names were actually "recalled", i.e. asterisked; 100% precision = all asterisked words are also SUC-names (= no disturbing "noise" in the output).

Before I proceed I must explain very carefully what my claims are. For the time being I only consider Ux-words, meaning that I here simply disregard words like "USA". (In the next warp I will try to catch such names, too). Words like *president* in *president George Bush* the SUC-people think is part of the name, but I will not even try to catch them; I simply think, that their view is wrong. Another thing that I have not tried to catch so far, is that name combinations like *George Bush* actually refer to one and the same object, the **constellation** must be considered as one name. At present they are counted as two separate names. I have done some preliminary testing for identifying such complex names, and for the moment the best - in terms of recall and precision - à priori procedure seems to consider **all** name meetings as being coreferential, but this is too early to say. Peculiarly enough, the presence of certain non-Ux-words seems to enhance such affinities: *Otto von Bismarck*, *Louis de Funès*, *Jan van der Velde*, *Gabriel de la Gardie*, etc. Such words or word combinations I call **name continuators**, and I have about 10 of them in a permanent lexicon, which is used in the procedures described in section 4.1 and 4.2. (In fact, this is the **only** lexicon I use, besides the temporary ones created during the processing; cf., e.g., next section).

The name marking is now performed in five distinct steps, two steps when looking at single words in isolation, two when immediate contexts are taken into account and one final left to right pass, marking such formal names which for some reasons have not been marked, so far.

### 3. Looking at names in isolation

The basic here idea is to extract all formal names in the text(s), make a lexicon of them, and then use that lexicon for marking (allegedly) names occurring anywhere in the text, including their original positions.

#### 3.1 Step 1: The marking of formal names

The very first step is to mark all formal names in the text; this is the basis for all later processings. The text I use here is a text called AA-AF.TXT, which comprises 77,000+ words of running text, and the result text is called AA-AFM0.TXT. When running the evaluation program on this text, I get the following basic statistics (Fig. 2).

```
LOG Output AA-AFM0.LOG   date:  28/2 1998
RuleFile:      NAMELOG.RUL
TextFile in:   AA-AFM0.TXT, Lines in: 13764

Event statistics!
wds           77664          %%total # of wrds
nms           5575          %%total # of SUC-names (Ux-words only)
fse           254           %%false hits: only marked by rules
mss           589           %%misses: only SUC-marked
hts           4986          %%hits: both marked by rules and by SUC
rcl:          89.4          %%recall:    100*hts/nms
pre:          95.2          %%precision: 100*hts/(fse+hts)
```

Fig 2. The evaluation of text AA-AFM0.txt: formal names marked

Please, observe, that the recall rate as given in Fig. 2 is for the whole text. If only interior areas are considered - i.e. those words which Step 1 comprises - the recall rate is considerably higher, viz. about 98%.

Among the 254 false hits about 220 are compounds of the type *\*Sovjetledaren* (cf. sect. 2), and I will return to a comment about such "errors" in the final section.

#### 3.2 Step 2: Apply the information obtained in Step 1 to non-interior Ux-words.

##### 3.2.1 Step 2.1: "No morphology".

As a preliminary exercise we now extract all marked words in AA-AFM0.TXT, make a lexicon of them and mark those Ux-words in AA-AFM0.TXT that were not marked in step 1, i.e. essentially words in sentence beginnings, we obtain a remarkably high recall - viz. 98.1% - but at a cost of a rather bad precision 78.1%. If we only consider such words that this step actually comprises, i.e. non-interior words, the result is a near catastrophe. We get a precision rate of only 29.1%, which means that more than 70% of all freshly marked words are non-names. Not very impressive.



The source of this disaster are to be found among “hits” in Step 1 of the following types:

```
ac04c 014 köper nu ut [ *Den norske Bank ] , [ Union ...
ae07d 024 ar ingen roll *Det år det som år det fina m...
```

In the first example *Den* in *Den norske bank* (“The Norwegian Bank”) is (part of) a name, which so far is OK (i.e. in Step 1). In the second example, *Det* is marked because of a printing error in the SUC-text (or, rather, in the original raw text). *Det* here simply opens a new sentence, and there should have been a full stop before it. (As said, my test texts are derived from a preliminary version of the SUC-corpus. In fact, using Step 1 and then a search for false hits in interior positions is an excellent way of locating errors of the mentioned type, thus providing valuable up-date information for the final version.

Now, the words *Den* and *Det* are both ordinary words and, incidentally, also very common. Furthermore, they are typical “sentence openers”, meaning there is a good chance to find them in sentence beginnings and spelled with a capital initial letter. Altogether there was a handful of different words of that type marked in Step 1, and these few words together totally destroyed this simplistic approach.

This step is now modified in the following way. I compare the frequencies of all formal names obtained in Step 1 with the frequencies of the same words spelled with common letters. Then I keep only those formal names that are more common as formal names as they are as common words. Simple.

After step 2.1: recall = 95.8% ; precision = 95.8.%.

### 3.2.1 Step 2.2: Now taking morphology into account.

As said in Preliminaries I will assume that names can only take the genitive as flexion, which in Swedish is a simple: an -s added to the name (and no apostrophe): *Paul*, *Pauls*, and if the name already ends in an -s, then nothing is added: *Nils*, *Nils*.

The procedure is now: For every formal name found in the text, remove an -s if it ends in one, add an -s if it does not, and then treat the two forms as separate words as in Step 2.1, i.e. match the frequency of them as names against the frequency of them as common words, and knock those name forms out that had a lower frequency as their common counterparts. If the two forms of an alleged name both survive this sieve, they are again fused and treated according to a default rule, saying that a found item may or may not have an extra -s on it in order to be accepted. Thus *Nils* occurs in the lexicon only as *Nil*, but treated according to the default rule, (no *Nil* was found in the text, but who cares). *Svan*, which is a common Swedish family name survived this ordeal (a few *svan*, “swan”, were not many enough), but *Svans* did not; a few *svans*, “tail”, knocked that variant out. For the common Swedish first name *Hans* neither form survived: *hans*, “his”, knocked *Hans* out, *han*, “he”, knocked *Han* out; the latter was less catastrophic, because no *Han* actually appeared in the texts. In Step 3 and 4 we will see how to save *Hans*.

The algorithm in this final version of Step 2 was also modified in order to account names like *d'Aille* and *Prince's*. The solution here was simply to consider the ‘ ’ to be a word delimiter in such contexts, meaning that no special arrangements needed for them. (I see no problem in using this simple “trick” to solve the genitive problem for English.) Anyhow, the final “scores” for Step 2 are now:

After Step 2: recall = 96.0% ; precision = 95.7%.

#### **4. Looking at names in context**

##### **4.1 Step 3: “The property of being a name is contagious”.**

Names often come in “bursts”, the earlier mentioned example *George Bush* is by no means rare. On the contrary, they abound: *Hans \*Gustavsson*, *Svenska \*Handelsbanken* (“The Swedish Business Bank”), *Nya \*Zeeland*, *\*Paris All Stars*, *The International Tennis \*Weekly*, *\*Perstorp \*Plastic Systems*, *\*Sturkö \*Design och Rökeri* and many others (The asterisks here are those put therein Step 2.).

Step 3 is now very simple, we apply what I call “asterisk propagation”: An Ux-word in interior position receives an asterisk, if a neighbour of it has one, and this procedure is applied recursively. The “infection” may also spread over “name continuators”, (cf. sect. 2), and *och* is one of them. Through this mechanism all the unasterisked Ux-words in the examples above now become marked. The tennis journal through a repetitive asterisk left propagation, the Paris stars through a corresponding right asterisk propagation. The word *Rökeri*, “smoke house”, receives its asterisk through a rightwards jump over *och*. 13 occurrences, for instance, of the name *Hans* were recovered through Step 3.

After Step 3: recall = 97.0% ; precision = 95.7%.

##### **4.2 Step 4: “Once a forename, always a forename”.**

In this step we define an asterisked word occurring immediately to the left of another asterisked word as a **forename**, possibly with a word continuator in between. We start this step by extracting all forenames, make a lexicon of them, and apply this lexicon to the words in the sentence beginnings (well, in non-interior positions). A found word is then assumed to be a name, if the word after it is asterisked.

There were not many names caught in this process (five for this text, to be precise, including one *Hans*) and there were three false hits. In a larger sample of the SUC-texts, AA-ED.TXT, comprising somewhat more than 300,000 words, the corresponding figures were 57 hits and 30 false hits, so the increase in recall may not be that negligible.

After Step 4: recall = 97.5% ; precision = 95.7%.

In the next and final step we again look at interior words in isolation.

## 5. Step 5: "Formal names are names, after all"

In sect. 3.2.1 we briefly discussed the example

```
ac04c 014 k per nu ut [ *Den norske Bank ] , [ Union ...
```

The word *Den* in this example was asterisked in Step 1, but it lost its marking through the knock-out procedures of Step 2. This *Den* has then stayed unasterisked after that. But *Dendefines*, after all, at least the beginning of a name, and it is a SUC-name according to our definitions. How do we capture this observation?

In this final pass we once more scan the text from left to right and mark so far unasterisked formal names, now with somewhat slackened conditions compared to Step 1. There we excluded Ux-words occurring immediately after a quote-in sign. Now we accept such words unless this character is preceded by a colon (when the Ux-word opens a direct quotation and may or may not be a name). Ux-names now marked in this process are more often than not (the beginning of) either a name of an intellectual work or just an ordinary, complex name, as the one in the example discussed above.

After Step 5: recall = 98.7%; precision = 95.2%.

## 6. Conclusions

We see a slight decrease of precision rate in Steps 3 through 5. This decrease would most certainly be hard to verify statistically, but I think it is significant. The reason is that these steps (or rules) might be a little bit too heuristic, meaning that the number of new false hits will grow relatively faster than the number of new true hits. It might be possible, I think, to marginally refine these steps, but for my purposes the present figures are quite satisfactory.

Anyhow, I have run the same set of rules on the much larger text AA-AD.TXT, and found absolutely comparable results. Theoretically there should, though, exist some optimum text size for applying my algorithms, but my findings so far seem to indicate that the algorithms are very insensitive to text size.

Compounds of the *\*Sovjetleader* type (cf. sect. 2) represent the by far most common type of false hits, and they considerably contribute to the smaller figure of the precision rate. If you are actually trying to identify all (and only) the names in a text, such words will, of course, create trouble. But in the context of what I am trying to do - identify sentences - the question is, whether the acceptance of such words as names would be so bad, after all. I need to identify words with a stable Ux-spelling, and the mentioned compounds have exactly that.

# Some Results Regarding Tree Homomorphic Feature Structure Grammar and the Empty String\*

Tore Burheim  
Telenor Research and Development  
PO.Box 23, N-2007 Kjeller  
Tore.Burheim@fou.telenor.no

## Abstract

In this article we focus on some restrictions we may impose on Tree Homomorphic Feature Structure Grammar (THFSG) regarding the empty string. After defining the restrictions, we prove some closure properties of the two new classes of languages these restrictions give us. Furthermore, we establish that the membership problem is PSPACE-complete for THFSGs without empty productions in the phrase structure rules.

## 1 Introduction and some definitions

In many linguistic theories the empty string plays the prominent role of the empty category. This is also the case for early Lexical-Functional Grammar (LFG) [KB82]. However, later the use of the empty category in LFG has been questioned. Kaplan and Zaenen [KZ89] argue that by the introduction of functional uncertainty (implemented as regular expressions in equation schemata) there is no place for the empty category in LFG. Grammaticality is enforced through the completeness, coherence and consistence constraints on the functional structure. Bresnan [Bre95] on the other side, argues against this view. With no intention of going into this discussion, we may at least say that, from a linguistic point of view, it remains a matter of dispute whether the empty category is needed in LFG-like grammars which allow regular expressions in the equation schemata.

From a more technical perspective, the empty string requires special care with respect to parsing: The question is, when do we introduce an empty category/string? Even if it sounds a bit overwhelming, we may introduce an unlimited number of empty categories all over the string. This is a problem e.g. for bottom-up parsing. Both the linguistic and the technical perspectives make it interesting to study grammar formalisms with respect to the empty string.

Tree Homomorphic Feature Structure Grammar (THFSG) [Bur97b] is an LFG-like feature structure grammar formalism which may allow regular expressions in the equation schemata.<sup>1</sup>

---

\*This article is a short version of [Bur98].

<sup>1</sup>In [Bur97a] it was proved that we may introduce regular expressions in the equation schemata for THFSG without extending the class of languages described.

As LFG, it has a context-free phrase structure backbone and it allows the empty string on the right hand side in the phrase structure rules. In the work presented here, we study the classes of languages we get by making two different restrictions regarding the empty string. First we define the class of languages we get by just removing the empty string from the languages defined by THFSG. Then we define a special version of THFSG in which we do not allow the empty string in the phrase structure rules, the so-called  $\varepsilon$ -free THFSG. But first, some words about THFSG itself.

## 1.1 Tree Homomorphic Feature Structure Grammar

THFSG was introduced by Burheim in [Bur97b]. It is based on Lexical Functional Grammar [KB82, DKMZ95] and work by Colban [Col91]. The formalism has a context-free phrase structure backbone and add equations to the nodes in the phrase-structure tree as is done in LFG. These equations describe feature structures. In the formal framework there are two main differences from LFG: First, due to a restriction that is imposed on the equations in the grammar, the referred part of the feature structure is a tree which includes a homomorphic image of the phrase structure tree. On the other side, with this restriction we do not need the off-line parsability constraint to make the grammar decidable.

We give a brief introduction to THFSG. Since it is based on feature structures we will start with an informal definition of feature structures.

A *feature structure* over a set of attribute symbols  $\mathcal{A}$  and value symbols  $\mathcal{V}$  is a four-tuple  $\langle Q, f_D, \delta, \theta \rangle$  where  $Q$  is a finite set of nodes,  $f_D : D \rightarrow Q$  is a function, called the name mapping,  $\delta : Q \times \mathcal{A} \rightarrow Q$  is a partial function, called the transition function, and  $\theta : Q \rightarrow \mathcal{V}$  is a partial function called the atomic value function. We extend the transition function by its transitive and reflexive closure to be a function from pairs of nodes and *strings* of attribute symbols, and we assume that  $D$  is implicit defined by  $f$ . A feature structure is *well defined* if it is describable (from named nodes), acyclic and atomic.

THFSG uses equations to talk about feature structures, such that a feature structure may or may not satisfy each equation. A feature structure *satisfies* the equation  $x_1 u_1 \doteq x_2$  if and only if  $\delta(f(x_1), u_1) = f(x_2)$ , and the equation  $x_3 u_3 \doteq v$  if and only if  $\alpha(\delta(f(x_3), u_3)) = v$ , where  $x_1, x_2, x_3 \in D$ ,  $u_1, u_3 \in \mathcal{A}^*$  and  $v \in \mathcal{V}$ . These path and value equations are the only kind of equations we use in THFSG. The well defined feature structure  $M$  satisfies the set of equations  $E$ , if and only if  $M$  satisfies every equation in  $E$ .

A *Tree Homomorphic Feature Structure Grammar*, THFSG, over the set of attribute symbols  $\mathcal{A}$  and value symbols  $\mathcal{V}$ , is a 5-tuple  $\langle \mathcal{K}, \mathcal{S}, \Sigma, \mathcal{P}, \mathcal{L} \rangle$  where  $\mathcal{K}$  and  $\Sigma$  are two finite and disjoint sets of symbols, called categories and terminals, and  $S \in \mathcal{K}$  is the start symbol. Moreover  $\mathcal{P}$  is a finite set of production rules

$$A_0 \rightarrow \begin{array}{c} A_1 \dots A_m \\ E_1 \quad E_m \end{array} \quad (1)$$

where  $m \geq 1$ ,  $A_0, \dots, A_m \in \mathcal{K}$ , and for every  $i$ ,  $1 \leq i \leq m$ , is  $E_i$  a finite set with *one and only one* equation schema on the form  $\uparrow u_1 = \downarrow$  where  $u_1 \in \mathcal{A}^*$ , and a finite number of equation schemata on the form  $\uparrow u_2 = v$  where  $u_2 \in \mathcal{A}^+$  and  $v \in \mathcal{V}$ . At last  $\mathcal{L}$  is a finite set of lexicon

rules

$$A \rightarrow \begin{matrix} t \\ E \end{matrix} \quad (2)$$

where  $A \in \mathcal{K}$ ,  $t \in (\Sigma \cup \{\varepsilon\})$ , and  $E$  is a finite set of equation schemata on the form  $\uparrow u_3 = v$  where  $u_3 \in \mathcal{A}^+$  and  $v \in \mathcal{V}$ .

The constituent structure (c-structure) is a phrase structure tree decorated with symbols and equation schemata sets according to the production and lexicon rules the usual way. As in LFG, the up and down arrows in the equation schemata are metavariables. To instantiate the arrows we substitute them with nodes in the c-structure, which then become the name domain in the name mapping function. This function is then a mapping from the nodes in the c-structure to the nodes in the feature structure. A c-structure is *feature consistent* if and only if the union of all the equations in the c-structure is satisfied by a well defined feature structure.

## 2 THFSG and $\varepsilon$

In this section we study some relations between different restrictions we may impose on THFSG regarding the empty string  $\varepsilon$  and some closure properties on the classes of languages we get when imposing these restrictions. Two restrictions may be imposed: the absence of the empty string in the grammar and the absence of the empty string in the language generated. Let us start with the definition:

### Definition 1

- **$\varepsilon$ -free form:** A THFSG is in  $\varepsilon$ -free form if and only if the empty string  $\varepsilon$  does not occur on the right hand side in any lexicon rule in the grammar.
- **$\varepsilon$ -free THFSG-languages:** The class of  $\varepsilon$ -free THFSG-languages,  $\mathcal{C}(\text{THFSG})_\varepsilon$ , is defined as

$$\mathcal{C}(\text{THFSG})_\varepsilon = \{L \in \mathcal{C}(\text{THFSG}) \mid \varepsilon \notin L\} \quad (3)$$

THFSGs in  $\varepsilon$ -free form only have lexicon rules as

$$A \rightarrow \begin{matrix} t \\ E \end{matrix} \quad (4)$$

where  $t$  is a symbol in the alphabet  $\Sigma$ , and *not* the empty string. THFSGs in  $\varepsilon$ -free form are almost identical to the grammar formalism GF1 defined by Colban [Col91]. The only difference is that GF1 only allows path equation schemata on the forms  $\uparrow \doteq \downarrow$  and  $\uparrow a \doteq \downarrow$  where  $a$  is a single attribute symbol. This restriction on attribute strings in GF1 is used in a normal form<sup>2</sup> for THFSG [Bur97b]. Following the lines in the proof of normal form, it is easy to see that we may rewrite any grammar with longer attribute strings into an equivalent grammar with at most one single attribute symbol in each path equation schema, without violating the  $\varepsilon$ -free restriction. Hence the  $\varepsilon$ -free THFSG describe the same class of languages as Colban's GF1.

<sup>2</sup>This normal form also requires exactly two elements on the right hand side in the production rules

When studying classes of languages, we often want to study what happens when we impose different algebraic operations on a class, i.e. we want to study its closure properties. In this context, trios and Abstract Families of Languages (AFL) are of particular interest. But before we go into details on trios and AFL's, let us clarify what we mean by a *class* of languages. A class of languages is a set of languages  $\mathcal{C}_\Gamma$  over a countable set of symbols  $\Gamma$ , such that for each language  $L \in \mathcal{C}_\Gamma$  there exists a finite alphabet  $\Sigma \subseteq \Gamma$  such that  $L \subseteq \Sigma^*$ . A class of languages given by a grammar formalism  $\mathcal{C}_\Gamma(\mathcal{GF})$  is a class of languages such that for each language  $L'$  in  $\mathcal{C}_\Gamma(\mathcal{GF})$  there exists a grammar  $G$  in  $\mathcal{GF}$  such that  $L(G) = L'$ , and for each grammar  $G$  in  $\mathcal{GF}$   $L(G)$  is in  $\mathcal{C}_\Gamma(\mathcal{GF})$ . In the rest of this paper we assume that  $\Gamma$  is given and omit  $\Gamma$  as subscript.

A *trio* is a class of languages closed under  $\varepsilon$ -free homomorphism, inverse homomorphism, and intersection with regular languages. A *full trio* is a class of languages closed under arbitrary homomorphism, inverse homomorphism, and intersection with regular languages. By intersection with regular languages, we mean the traditional binary set theoretic operation. A *string homomorphism* is a function  $h : \Delta^* \rightarrow \Sigma^*$  such that for every  $w \in \Delta^*$  and  $a \in \Delta$  we have

$$h(\varepsilon) = \varepsilon \quad (5)$$

$$h(aw) = h(a)h(w) \quad (6)$$

A homomorphism is  $\varepsilon$ -free if  $h(a) \neq \varepsilon$  for all  $a \neq \varepsilon$ . The string homomorphic image of a language  $L \subseteq \Delta^*$  under a string homomorphism  $h : \Delta^* \rightarrow \Sigma^*$  is the language  $\{h(w) \mid w \in L\}$ . The inverse string homomorphic image of a language  $L' \subseteq \Sigma^*$  is the language  $\{w \mid h(w) \in L'\}$ .

An *Abstract Family of Languages* is a trio which is also closed under concatenation, union, and positive closure. A *full abstract family of languages* is a full trio closed under concatenation, union, and Kleene closure. By union we mean the traditional binary set-theoretic operation. The concatenation of two languages  $L_1$  and  $L_2$ , is the language  $\{w_1w_2 \mid w_1 \in L_1 \text{ and } w_2 \in L_2\}$ . By positive closure of a language  $L$  we mean the language  $\{w_1 \dots w_n \mid n \geq 1 \text{ and } w_1, \dots, w_n \in L\}$ , and by Kleene closure of a language  $L$  we mean the language  $\{w_1 \dots w_n \mid n \geq 0 \text{ and } w_1, \dots, w_n \in L\}$ . The only difference between Kleene closure and the positive closure is the empty string in Kleene closure.

To show that a class of languages is a full trio, it is sufficient to show closure under NFT-mapping. This due to the fact that the NFT-image of a class of languages gives us the least full trio containing the class [Gin75]. If we restrict the NFT-mapping to  $\varepsilon$ -free mappings, we get the least trio. A *Nondeterministic Finite Transducer* (NFT) is a 6-tuple  $M = \langle Q, \Delta, \Sigma, \delta, q_0, F \rangle$  where  $Q$  is a finite set of states,  $\Delta$  is an input-alphabet,  $\Sigma$  is an output-alphabet,  $\delta$  is a function from  $Q \times (\Delta \cup \{\varepsilon\})$  to finite subsets of  $Q \times \Sigma^*$ ,  $q_0 \in Q$  is the initial state, and  $F \subseteq Q$  is a set of final states. An NFT is  $\varepsilon$ -free if  $\delta$  is a function from  $Q \times (\Delta \cup \{\varepsilon\})$  to finite subsets of  $Q \times \Sigma^+$ . We extend the transformation function  $\delta$  as follows: (1) For every  $q \in Q$ ,  $\langle q, \varepsilon \rangle \in \delta(q, \varepsilon)$ . (2) If  $\langle q_2, x \rangle \in \delta(q_1, w)$  and  $\langle q_3, y \rangle \in \delta(q_2, a)$ , then  $\langle q_3, xy \rangle \in \delta(q_1, wa)$  for every  $q_1, q_2, q_3 \in Q$ ,  $a \in (\Delta \cup \{\varepsilon\})$ , and  $w \in \Delta^*$ . Let  $\widetilde{\mathcal{NFT}}$  be the set of NFTs, and  $\mathcal{NFT}$  be the set of  $\varepsilon$ -free NFTs.

For any NFT  $M = \langle Q, \Delta, \Sigma, \delta, q_0, F \rangle$ , the image under  $M$  of a string  $w \in \Delta^*$  and a language  $L \subseteq \Delta^*$  are

$$M(w) = \{x \mid \exists q \in F : (q, x) \in \delta(q_0, w)\} \quad (7)$$

$$M(L) = \bigcup_{w \in L} M(w) \quad (8)$$

The inverse images under  $M$  of a string  $x \in \Sigma^*$  and a language  $L' \subseteq \Sigma^*$  are

$$M^{-1}(x) = \{w \mid x \in M(w)\} \quad (9)$$

$$M^{-1}(L') = \bigcup_{w \in L'} M^{-1}(w) \quad (10)$$

For any class  $\mathcal{C}$  of languages, the NFT-image and  $\varepsilon$ -free NFT-image of  $\mathcal{C}$ ,  $\widetilde{\mathcal{M}}(\mathcal{C})$  and  $\mathcal{M}(\mathcal{C})$ , are defined as

$$\widetilde{\mathcal{M}}(\mathcal{C}) = \{M(L) \mid L \in \mathcal{C} \text{ and } M \in \widetilde{\mathcal{NFT}}\} \quad (11)$$

$$\mathcal{M}(\mathcal{C}) = \{M(L) \mid L \in \mathcal{C} \text{ and } M \in \mathcal{NFT}\} \quad (12)$$

An important result from [Gin75] is that for any class  $\mathcal{C}$  of languages:

$$\begin{aligned} \widetilde{\mathcal{M}}(\mathcal{C}) &= \{h_2(h_1^{-1}(L) \cap R) \mid L \in \mathcal{C}, R \in \mathcal{R}, \\ &\quad h_1, h_2 \text{ are homomorphisms}\} \end{aligned} \quad (13)$$

$$\begin{aligned} \mathcal{M}(\mathcal{C}) &= \{h_2(h_1^{-1}(L) \cap R) \mid L \in \mathcal{C}, R \in \mathcal{R}, h_1 \text{ is a homomorphism} \\ &\quad \text{and } h_2 \text{ is an } \varepsilon\text{-free homomorphism}\} \end{aligned} \quad (14)$$

where  $\mathcal{R}$  is the class of regular languages. From this result, Ginsburg [Gin75] shows that  $\widetilde{\mathcal{M}}(\mathcal{C})$  is the least full trio containing  $\mathcal{C}$  and  $\mathcal{M}(\mathcal{C})$  is the least trio containing  $\mathcal{C}$ . An interesting case arises when  $\mathcal{C}$  only consists of a single language.

Now let us state some closure properties for  $\varepsilon$ -free THFSG-languages. In [Bur97b], it was proved that the class of THFSG-languages,  $\mathcal{C}(\text{THFSG})$ , is a full Abstract Family of Languages. From this fact, we get the following result directly.

**Lemma 1** *The class of  $\varepsilon$ -free THFSG-languages,  $\mathcal{C}(\text{THFSG})_\neq$ , is an abstract family of languages and*

$$\mathcal{C}(\text{THFSG})_\neq = \{L - \{\varepsilon\} \mid L \in \mathcal{C}(\text{THFSG})\} \quad (15)$$

**Proof:** From [Gin75, p21], we know that if  $\mathcal{C}$  is an abstract family of languages, then  $\{L \in \mathcal{C} \mid \varepsilon \notin L\} = \{L - \{\varepsilon\} \mid L \in \mathcal{C}\}$  is an abstract family of languages. Since  $\mathcal{C}(\text{THFSG})$  is a (full) abstract family of languages [Bur97b], we know that  $\mathcal{C}(\text{THFSG})_\neq$  is an abstract family of languages and we know that

$$\mathcal{C}(\text{THFSG})_\neq = \{L - \{\varepsilon\} \mid L \in \mathcal{C}(\text{THFSG})\} \quad (16)$$

■

From the definition of  $\varepsilon$ -free THFSG-languages and Lemma 1, we have that

$$\{L \in \mathcal{C}(\text{THFSG}) \mid \varepsilon \notin L\} = \{L - \{\varepsilon\} \mid L \in \mathcal{C}(\text{THFSG})\} \quad (17)$$

Another way to view this is that

$$\mathcal{C}(\text{THFSG}) = \{L, L \cup \{\varepsilon\} \mid L \in \mathcal{C}(\text{THFSG})_\neq\} \quad (18)$$

Since  $\mathcal{C}(\text{THFSG})_\neq$  is an abstract family of languages, and hence a trio we have that it is closed under  $\varepsilon$ -free NFT-mapping

$$\mathcal{M}(\mathcal{C}(\text{THFSG})_\neq) = \mathcal{C}(\text{THFSG})_\neq \quad (19)$$

It is straightforward to establish the closure under arbitrary NFT-mapping:



**Lemma 2** *The NFT-image of  $\mathcal{C}(\text{THFSG})_\neq$  is  $\mathcal{C}(\text{THFSG})$ , that is*

$$\widetilde{\mathcal{M}}(\mathcal{C}(\text{THFSG})_\neq) = \mathcal{C}(\text{THFSG}) \quad (20)$$

**Proof:** Since  $\mathcal{C}(\text{THFSG})$  is closed under NFT-mapping and  $\mathcal{C}(\text{THFSG})_\neq$  is a subset of  $\mathcal{C}(\text{THFSG})$ , we have that  $\widetilde{\mathcal{M}}(\mathcal{C}(\text{THFSG})_\neq) \subseteq \mathcal{C}(\text{THFSG})$ . Since  $\mathcal{C}(\text{THFSG}) = \{L, L \cup \{\varepsilon\} \mid L \in \mathcal{C}(\text{THFSG})_\neq\}$  and it is straightforward to define an NFT  $M$  such that  $M(w) = \{w, \varepsilon\}$  for each string  $w$ , we have that  $\mathcal{C}(\text{THFSG}) \subseteq \widetilde{\mathcal{M}}(\mathcal{C}(\text{THFSG})_\neq)$ . ■

Now we turn our attention to  $\varepsilon$ -free THFSGs. Recall that these are THFSGs without the empty string on the right hand side in lexicon rules. Immediately, we see that  $\varepsilon$ -free THFSGs define only  $\varepsilon$ -free THFSG-languages, hence

$$\mathcal{C}(\varepsilon\text{-freeTHFSG}) \subseteq \mathcal{C}(\text{THFSG})_\neq \quad (21)$$

However,  $\varepsilon$ -free languages may be generated by THFSGs with  $\varepsilon$  in the lexicon rules.

When we now turn our attention to closure properties for  $\varepsilon$ -free THFSGs, let us first establish that the class of languages they define is an Abstract Family of Languages.

**Lemma 3** *The class of languages described by  $\varepsilon$ -free THFSGs,  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$ , is an abstract family of languages.*

**Proof:** The proof that  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is closed under union, concatenation, and positive closure, is almost identical to the proof of closure under union, concatenation, and Kleene star for  $\mathcal{C}(\text{THFSG})$  [Bur97b]. The only difference is that we do not need, and are not allowed to introduce, the lexicon rule which generates the empty string for Kleene closure.

The proof that  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is closed under  $\varepsilon$ -free NFT-mapping is identical to the proof that  $\mathcal{C}(\text{THFSG})$  is closed under NFT-mapping in [Bur97b], except that we must restrict the NFT to be  $\varepsilon$ -free. Hence the class  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is an Abstract Family of Languages. ■

Since an Abstract Family of Languages is also a trio, and the  $\varepsilon$ -free NFT-image of a class of languages is the least trio containing the class, we have that  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is closed under  $\varepsilon$ -free NFT-mapping, that is

$$\mathcal{M}(\mathcal{C}(\varepsilon\text{-freeTHFSG})) = \mathcal{C}(\varepsilon\text{-freeTHFSG}) \quad (22)$$

Now, if we look for the least full abstract family of languages containing  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  we get the following result.

**Lemma 4** *The NFT-image of  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is  $\mathcal{C}(\text{THFSG})$ , that is*

$$\widetilde{\mathcal{M}}(\mathcal{C}(\varepsilon\text{-freeTHFSG})) = \mathcal{C}(\text{THFSG}) \quad (23)$$

**Proof:** Since  $\mathcal{C}(\varepsilon\text{-freeTHFSG}) \subseteq \mathcal{C}(\text{THFSG})$ , and  $\mathcal{C}(\text{THFSG})$  is closed under NFT-mapping, we know that  $\widetilde{\mathcal{M}}(\mathcal{C}(\varepsilon\text{-freeTHFSG})) \subseteq \mathcal{C}(\text{THFSG})$ .

To prove that  $\mathcal{C}(\text{THFSG}) \subseteq \widetilde{\mathcal{M}}(\mathcal{C}(\varepsilon\text{-freeTHFSG}))$ , we show that for each THFSG  $G$  there exists an  $\varepsilon$ -free THFSG  $G'$  and an NFT  $M$  such that  $M(L(G')) = L(G)$ . Let  $G = \langle \mathcal{K}, \mathcal{S}, \Sigma, \mathcal{P}, \mathcal{L} \rangle$ , and assume that  $a \notin (\mathcal{K} \cup \Sigma)$ . Then let  $G' = \langle \mathcal{K}, \mathcal{S}, \Sigma \cup \{a\}, \mathcal{P}, \mathcal{L}' \rangle$  where  $\mathcal{L}'$  is the least set such that for each

$$A \rightarrow \begin{matrix} t \\ E \end{matrix} \quad (24)$$

in  $\mathcal{L}$ ,

$$A \rightarrow \begin{matrix} t' \\ E \end{matrix} \quad (25)$$

is a rule in  $\mathcal{L}'$ , where  $t' = a$  whenever  $t = \varepsilon$ , and  $t' = t$  else. Clearly,  $G'$  is an  $\varepsilon$ -free THFSG. Now define a string homomorphism  $h$  such that

$$h(t) = \begin{cases} \varepsilon & \text{if } t = a \\ t & \text{otherwise} \end{cases} \quad (26)$$

It is straightforward to see that  $h(L(G')) = L(G)$ . Since any string homomorphism may be represented by an NFT we know that there exists an NFT  $M$  such that  $M(L(G')) = L(G)$ . ■

The results in this section with respect to NFT-images may be summarised in the following theorem.

### Theorem 1

$$\begin{array}{ccc} & \mathcal{C}(\text{THFSG}) & \\ & \nearrow \tilde{M} & \nwarrow \tilde{M} \\ \mathcal{C}(\varepsilon\text{-free THFSG}) & \subseteq & \mathcal{C}(\text{THFSG})_{\neq} \end{array} \quad (27)$$

**Proof:** Directly from Lemmas 1, 2, 3, 4, and the fact that  $\varepsilon$ -free THFSGs only generate  $\varepsilon$ -free languages. ■

## 3 $\varepsilon$ -free THFSG and PSPACE

As noticed earlier,  $\varepsilon$ -free THFSG is almost identical to the grammar formalism GF1 defined by Colban [Col91]. Colban shows that the membership problem for GF1 is NP-hard. In this section we strengthen this result by stating that the membership problem for  $\varepsilon$ -free THFSG is PSPACE-complete. The difference between GF1 and  $\varepsilon$ -free THFSG—the length of attribute strings in the equations—does not make any difference here, hence the given result is also valid for GF1. In fact, consulting the proof of normal form in [Bur97b] we see that there is a polynomial (even linear) algorithm which transforms any  $\varepsilon$ -free THFSG into a GF1 grammar. Conversely, GF1 is a special case of  $\varepsilon$ -free THFSG.

We show the PSPACE completeness as usual in two steps, first showing that the problem is PSPACE-hard, and then that it is in PSPACE.

**Lemma 5** *The membership problem for  $\varepsilon$ -free THFSG is PSPACE-hard.*

The proof is by transforming the Finite State Automata Intersection Problem (FSAI) into the membership problem for  $\varepsilon$ -free THFSG. The FSAI is given on the following instances. Given a finite set of deterministic finite state automata  $A_1, \dots, A_n$  over a common alphabet  $\Sigma$ , let  $L(A_i)$  be the language accepted by automaton  $A_i$ ,  $1 \leq i \leq n$ . The FSAI question is then: is there a string  $u \in \Sigma^* : u \in L(A_1) \cap \dots \cap L(A_n)$ ? FSAI was shown by [Koz77] to be PSPACE-complete. Given any instance of this problem we show how to define an  $\varepsilon$ -free THFSG

$G$  such that  $t^n \in L(G)$  for a given atomic symbol  $t$ , if and only if there exists a string in the intersection as described.<sup>3</sup>

Since  $\varepsilon$ -free THFSG is a special case of THFSG we have the following corollary directly from Lemma 5:

**Corollary 1** *The membership problem for THFSGs is PSPACE-hard.*

With the result from Lemma 5 we need the following lemma to establish that the membership problem is PSPACE-complete.

**Lemma 6** *The membership problem for  $\varepsilon$ -free THFSGs is in PSPACE.*

To prove this, we show how to construct a polynomial space nondeterministic Turing Machine which decides the language defined by an  $\varepsilon$ -free THFSG. Since the construction can be done in polynomial time for any  $\varepsilon$ -free THFSG, and  $\text{NPSpace} = \text{PSPACE}$  [Sav70], this implies that the given membership problem is in PSPACE. The main idea in the proof is that we may traverse both the c-structure and feature structure top-down simultaneously with only a limited view of the rest of the structures. This due to the homomorphism between the two structures and the restrictions on the equation schemata.<sup>4</sup>

From Lemma 5 and 6 we have the following result.

**Theorem 2** *The membership problem for  $\varepsilon$ -free THFSGs is PSPACE-complete.*

## 4 Summary and remarks

In this article we have proved a set of closure properties for THFSG with two Restrictions regarding the empty string. We have shown that both the class of  $\varepsilon$ -free THFSG-languages and the class of languages defined by  $\varepsilon$ -free THFSGs are abstract families of languages, and that the least full-abstract family of languages including each of these two classes is the class of languages defined by THFSG. We also saw that the membership problem for  $\varepsilon$ -free THFSGs is PSPACE-complete.

There is one important open question here, and that is whether or not  $\mathcal{C}(\varepsilon\text{-freeTHFSG})$  is equal to  $\mathcal{C}(\text{THFSG})_{\neq \varepsilon}$ . If this is the case, the only difference between these two subclasses and  $\mathcal{C}(\text{THFSG})$  is the absence of the empty string in the languages generated. This would make it possible to define an  $\varepsilon$ -isolated normal form for THFSG in which the start symbol  $S$  never occurs on the right hand side of production rules, and the empty string only occurs on the right hand side of lexicon rules if the  $S$  occurs to the left. This kind of normal form is often used. However, the question is left for further research.

In the beginning of this article we noticed that in later LFG the empty category does not play a prominent role, even if there is some disagreement about the need for it [DKMZ95, p134] [KZ89, Bre95]. Functional uncertainty implemented as regular expressions in the equation schemata is used to handle linguistic phenomena previously handled by empty categories (e.g. long-distance dependencies). If we look at the use of regular expressions to implement functional

---

<sup>3</sup>The proof is given in [Bur98].

<sup>4</sup>The proof is given in [Bur98].

uncertainty as introduced in [KZ89], we only find one regular expression in each set of equation schemata, and this is a path equation schema.

Now, what about regular expressions in conjunction with  $\epsilon$ -free THFSG? In [Bur97b] it is proved that we may introduce regular expressions in the equation schemata in THFSG without extending the class of languages described, and that the emptiness and membership problems remain decidable. The proof of equivalence between THFSG with and without the possibility of regular expressions, introduces the empty string on the right hand side in new lexicon rules. However, by examining the proof it is straightforward to see that if we restrict regular expressions to only occur in path equation schemata, a corresponding introduction of the empty string in lexicon rules is no longer needed. As a result, the class of languages we get by allowing regular expressions in the path equation schemata in  $\epsilon$ -free THFSG is the same as the class of languages defined by clean  $\epsilon$ -free THFSG.

## Acknowledgements

I would like to thank Tore Langholm for his advice during the work that led to this paper, and for comments on early versions of the manuscript. I would also like to thank Kjell Jørgen Hole and Marianne Fjelltveit Hole for proofreading.

## References

- [Bre95] Joan Bresnan. Linear order, syntactic rank, and empty categories: On weak crossover. In Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III, and Annie Zaenen, editors, *Formal Issues in Lexical-Functional Grammar*, number 47 in CSLI Lecture Notes. CSLI-Publications, 1995.
- [Bur97a] Tore Burheim. Emptiness, membership and regular expressions for tree homomorphic feature structure grammars. In Tilman Becker and Hans-Ulrich Krieger, editors, *Proceedings of the Fifth Meeting on Mathematics of Language – MOL5*, D-97-02, Saarbrücken, Germany, 1997. Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI. Short version.
- [Bur97b] Tore Burheim. A grammar formalism and cross-serial dependencies. In Patrick Blackburn and Maarten de Rijke, editors, *Specifying Syntactic Structure*. CSLI-Publications, 1997.
- [Bur98] Tore Burheim. Three homomorphic feature structure grammar and the empty string. Manuscript., 1998.
- [Col91] Erik A. Colban. *Three Studies in Computational Semantics*. PhD thesis, University of Oslo, 1991.
- [DKMZ95] Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell III, and Annie Zaenen. *Formal Issues in Lexical-Functional Grammar*. Number 47 in CSLI Lecture Notes. CSLI Publications, 1995.

- [Gin75] Seymour Ginsburg. *Algebraic and Automata-Theoretic Properties of Formal Languages*. North-Holland Publishing Company, Amsterdam, The Netherlands, 1975.
- [KB82] Ronald M. Kaplan and Joan Bresnan. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*. The MIT-Press, Cambridge, Massachusetts, USA, 1982.
- [Koz77] D. Kozen. Lower bounds for natural proof systems. In *Proc. of 18th. Annual Symposium on Foundations of Computer Science*, pages 254–266, Long Beach, USA, 1977. IEEE Computer Society.
- [KZ89] Ronald M. Kaplan and Annie Zaenen. Long-distance dependencies, constituent structure, and functional uncertainty. In Mark R. Baltin and Anthony S. Kroch, editors, *Alternative conceptions of phrase structure*, pages 17–42. The University of Chicago Press, 1989.
- [Sav70] W.J. Savitch. Relationship between nondeterministic and deterministic tape complexity. *Journal of Computing System Science*, 4:177–192, 1970.

# Teaching and learning computational linguistics in an international setting

**Koenraad de Smedt**

[www.uib.no/acohum](http://www.uib.no/acohum)

[acohum@uib.no](mailto:acohum@uib.no)

## Summary

Computational Linguistics has long been a forerunner in the use of humanities computing technology. However, there are many organisational problems to be addressed to maintain and improve the quality of teaching and learning of Computational Linguistics. Advanced Computing in the Humanities (ACO\*HUM) is an international network which investigates the use of new technologies in Humanities teaching and learning. It promotes international co-operation a.o. for teaching and learning Computational Linguistics.

## Aims

In so far as Computational Linguistics belongs to the human sciences or humanities, it has long had a privileged position. It has long been a forerunner in the application of formal methods and of computational techniques from Artificial Intelligence. It has during the past twenty or so years established itself as an advanced field of research and study, and it has transformed both computer science and linguistics. However, its special position among humanities disciplines is rapidly vanishing. The sciences of language are today not at all unique in their adoption of advanced computing. History, for example, is rapidly absorbing computing as an important research methodology, for example for the intelligent searching of archives. History of art is rapidly adopting advanced visual processing techniques, the building of pictorial databases, etc. Even literature, which has long been a place of traditional scholarship based on books, is increasingly using computer-aided research techniques, for example in formal stylistics, and is beginning to take hypertext serious as basis for a new rhetoric. And with Sophie's world on CD-ROM, even philosophers are trading their pens for shiny disks. Briefly, the whole humanities are today faced with a challenge to innovate their learning and teaching.

Back to Computational Linguistics, what are the challenges Computational Linguistics is facing? The field of natural language processing has expanded enormously in the past decades and is continuing to expand with respect to theory, methods, and applications. This has allowed many different schools and many niches of special research. While this variety is useful and needs to be maintained, it is not so easy in everyday life to offer quality teaching in Computational Linguistics covering an up to date part of the Computational Linguistics spectrum. For one thing, the Computational Linguistics staff at most universities is rather small. For another, the development of good teaching materials is way behind the research developments. This may affect the quality of teaching in such ways that students are not sufficiently familiar with what research requires, students are not prepared for computational linguistics professions, and students do not learn some things which are considered basic at other universities, which results in the hindrances for their mobility. All this argues for increased co-operation, also on

international levels, to develop and offer better teaching and training in Computational Linguistics.

These challenges are food for thought for the SOCRATES / ERASMUS thematic network on Advanced Computing in the Humanities (ACO\*HUM). What do we envisage? Part of the efforts of ACO\*HUM is directed towards curricula innovation. We aim at curricula containing a range of basic modules with a wide international agreement, while allowing for local specialisation's. Here we build on seminal work done by the former ERASMUS network on natural language processing. Another part of our efforts is directed to agreeing on equivalencies of degrees, and possibly the creation of a new international masters degree in natural language processing. We also aim at the development of new teaching materials for Computational Linguistics, including web-based teaching. Web-based courses offer not only the opportunity to learn across national boundaries, but also to teach across national boundaries. Several individuals in the Computational Linguistics community are experimenting, but lack up till now a forum for the exchange of experiences. ACO\*HUM is actively stimulating the development and testing of more such courses, in co-operation with ELSNET. We intend to use the network as a forum for exchanging experiences on a trans-national basis. In this context we also stimulate also the transfer of research results to teaching materials, so that today's students learn to work with systems and data which are the state of the art. Finally, we promote Computational Linguistics as a professional profile, and we want to co-operate with professional organisations such as EACL and with research institutions and companies to study the correspondence between what students learn and what the real world expects from them as professionals.

Not only Computational Linguistics, but also mainstream linguistics is affected by the developments. Mainstream scholars of language are increasingly using tools which are produced by Computational Linguistics research. One needs to distinguish here between the linguist, who is a user of linguistic tools, and the computational linguist, who is a developer of linguistic tools. Even without understanding how a parser works, a parser can be a practical tool for a linguist who wants to play with grammars. This argues for the inclusion of tools courses in linguistics curricula. In fact, at the University of Bergen, all first year linguistics students use such tools as the LFG workbench and Tarski's world. A co-operation between linguists and computational linguists is needed in the updating of linguistics curricula, and also here a co-operation is useful. In fact, ACO\*HUM does not aim at an isolation of those using advanced computing in the humanities and those who don't. Rather, we want to bring advanced computing to as large a part of humanities students as possible.

### **Organisation of the network**

Advanced Computing in the Humanities (ACO\*HUM) is an international network project which investigates the use of new technologies in Humanities teaching and learning. The project started in September 1996 and will continue for 3 years. It is a network project, which means that it is based on the mutual exchange of information and on voluntary agreements. The network is no regulating agency, but offers a forum for exchange among the partners. This exchange takes the form of working groups which regularly meet, workshops and conferences. In September, a large conference will be organised in Bergen; this will be announced on the web and announcements will be sent to Nodali. Through voluntary co-operation in the network, the

project aims at facilitating mobility including virtual mobility and common educational resources. Eventually we hope to find a common ground for international degrees in Computational Linguistics, international pooling of educational materials and computational training materials for Computational Linguistics, and better contact between educational institutions and future employers.

The network has an office based in Bergen. They can be reached at [www.uib.no/acoHum](http://www.uib.no/acoHum). Currently, the network has six working groups which are represented schematically in Figure 1.

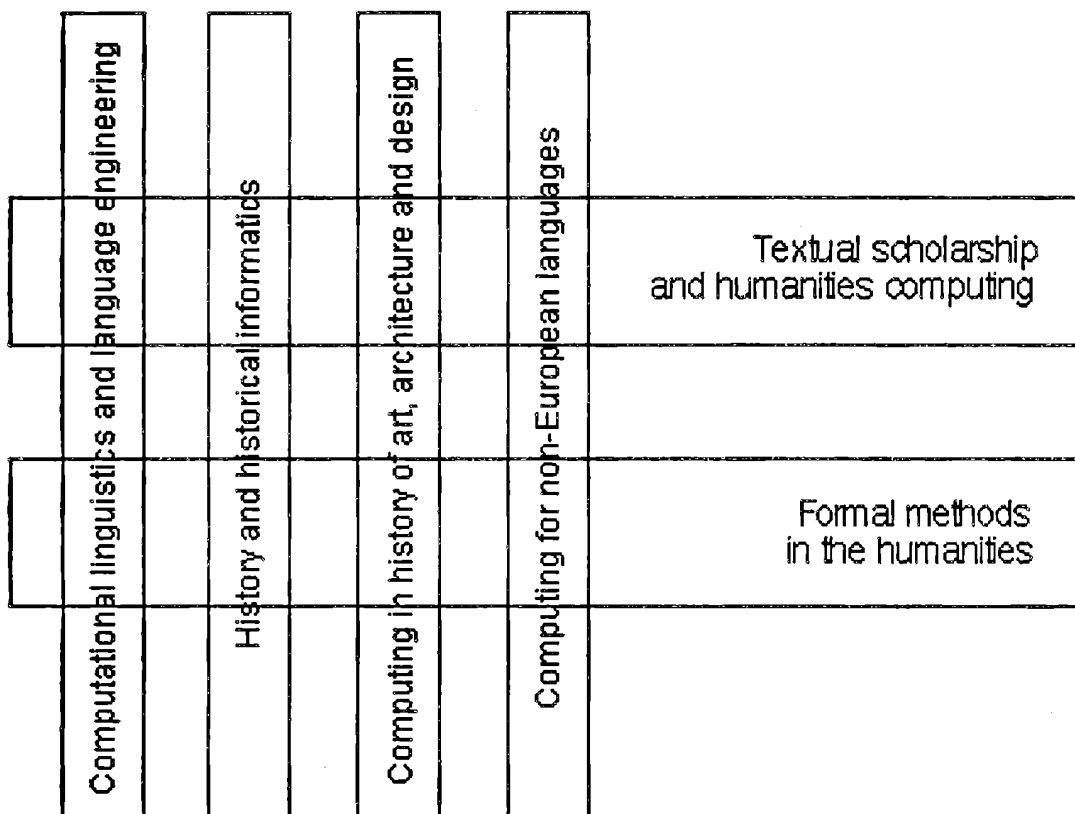


Figure 1. ACO\*HUM working groups

As shown in the figure, there is an interplay of vertical themes and horizontal themes. Vertical themes correspond largely to traditional disciplines within the humanities, while horizontal themes are humanities wide areas of interest.

The working group on Computational Linguistics and language engineering is composed of Bill Black and Marie Hayet (Manchester), Walter Daelemans (Antwerpen and Tilburg), Laurence Danlos (Paris), Koenraad de Smedt (Bergen), Gert Durieux (Antwerpen), Joakim Nivre (Göteborg), Hans Uszkoreit and Brigitte Krenn (Saarbrücken), Paul Mc Kevitt (Ålborg), Julia Lavid and Felisa Verdejo (Madrid), Torbjørn Nordgård (Trondheim) and Andy Way (Dublin). These members have been active in setting up discussions and actions related to the issues mentioned earlier, and will by the end of the project produce a report and recommendations.



## **Preliminary conclusions**

The whole project has been active for one and a half year, which is relatively short for a large network with over a hundred partners. The conclusions can therefore only be preliminary. Our partners have up to now indicated that within several humanities disciplines, clear trends are noticeable based on real student needs. Among these trends we mention the following:

1. There is increasing demand for collaborative practice. This is perceived to be a radical shift away from the cult of the individual towards the development of a new perception of the individual practitioner as a member of a team, or part of a network of relationships, both within and across disciplines. Information and communication technology is perceived as instrumental for new collaborative models. Students, also in Computational Linguistics, have to learn how to communicate and work together with others.
2. Graduates in Computational Linguistics are badly wanted on the job market in many regions of Europe, a.o. in Scandinavia. However, students need to be equipped with up to date knowledge which is needed in the real world, not just classical theories and methods. This means they need access to good course modules and up to date learning materials.
3. The use of information and communication technology should be applied to teaching and learning situations in order to improve the efficiency and quality of academic education. Bringing the computer into teaching in a reasoned way should liberate students from time and space limitations on learning, enable life-long and distance learning as well as augmenting traditional degree schemes. Concerted actions are needed to create technical and organisational conditions for such developments on an international basis.