A Bare-bones Constraint Grammar

Eckhard Bick

Institute of Language and Communication, University of Southern Denmark
Campusvej 55, DK 5230 Odense M
eckhard.bick@mail.dk

Abstract. This paper presents a solution for overcoming the lexical resource gap when mounting rule-based Constraint Grammar systems for minor languages, or in the face of licensing and financing limitations. We investigate how the performance of a CG disambiguation grammar responds to shifting input parameters, among them lexicon limitations of various degrees, the lack a morphological analyzer or both. We propose solutions for a bare-bones system, introducing endings heuristics and so-called morphological APPEND rules. For English, even with an unadapted disambiguation grammar, our bare-bones tagger achieved F-scores of 90-96% for part of speech, and 94-97% for lemmatization, depending on the modules and mini-lexica used.

Keywords: Less-resourced languages, PoS-tagging, Constraint Grammar, morphological analyzer, lexicon generation

1 Introduction

Conventional wisdom has it that in the realm of natural language parsing (NLP), statistical methods are more cost-efficient and easier to build than rule-based systems. However, the latter are dependent on training data, and machine learning of morphosyntactic analysis relies on the existence of a fair-sized annotated corpus for the language in question. If no such corpus exists, manual annotation of a boot-strapping corpus will be necessary, eliminating part of the cost-effectiveness advantage. Furthermore, in the face of Zipf's law, the limited availability of large linguist-revised corpora (especially treebanks) makes it difficult to achieve good lexical coverage. Rule-based NLP systems, on the other hand, while not dependent on training data, generally require extensive lexica and/or morphological analyzers as input modules. As it would seem, both statistical and rule-based approaches are resource-sensitive and can run into difficulties with minor languages, or in the face of licensing and financing limitations. What we are addressing in this paper, is the assumption that these problems are especially difficult to solve for rule-based systems, among them our own methodology of choice, Constraint Grammar (CG). At the disambiguational level, early work by Chanod & Tapanainen (1995) showed that even a small set of CG rules can compete with an off-the-shelve statistical tagger. However, in their experiment, both systems had access to an extensive lexicon an a mature morphological analyzer (a finite-state transducer). All published CG parsers to date (Karlsson et al. 1995 and Bick 2000) have included extensive lexico-morphological resources as input to their grammatical rules (as have other rule-based approaches like HPSG and LFG). To the best of our knowledge, no previous research has been done on how to build a CG in the absence of such resources.

2 Related work and methodological focus

A great deal of methodological inspiration on the corpus-based creation of lexico-morphological resources can be found in recent research literature, and the different approaches can be clustered into two distinct groups - on the one hand linguistically pre-structured corpus work with an emphasis on paradigm completeness and validation (e.g. Clément, Sagot & Lang 2004 or Forsberg et al. 2006), on the other hand machine learning techniques designed to deduce lexical information from raw text data in an unsupervised fashion (e.g. Goldsmith 2001 or Creutz & Lagus 2005). Automatic extraction techniques have also been suggested for higher level lexical information (e.g. Nikuládóttir & Whelpton 2009 for semantic relations), but the focus of this paper will be on the lexico-morphological level. Though our goal is a full inflexional analysis, we don't intend to build full morphological paradigms like Forsberg et al. or Clément et al. Rather, the idea is to find a balance between fullform mini-lexica from small annotated corpora on the one hand (chapter 5.3) and what Piasecki and Radzieszewski (2007) in their work on Polish call a statistical *a tergo index* on the other hand - a list of corpus- or lexicon-derived pseudo-suffixes or -endings, capturing inflexional and other patterns of word formation at the surface level (chapter 3). While Piasecki and Radzieszewski use a letter tree to spawn individual morphological tags, in our own work on English we investigated complete endings strings of various lengths, each associated not with individual tags, but complete tag chain readings, assuming that for a poorly inflecting language like English the line between analytical morphology and derivation is blurred, and that it is more important to capture possible patterns in word class changes or ethymology at the same time. As a third module, supplementing and modifying suggested readings from both mini-lexica and endings heuristics, we use a new type of Constraint Grammar rules (chapter 4), which are morphologically productive in nature, running counter to the traditional use of CG as reductionist disambiguation technique.

3 Simulating a full form lexicon

Depending on typological characteristics of the language in question, the balance of importance between a morphological analyzer and its lexicon may shift considerably. Thus, a language with a rich and completely regular inflexion and affixation system can be annotated for part of speech (PoS), number, tense etc. with an analyzer alone, and would need a lexicon only to resolve compounding and root-affix ambiguities. Highly irregular, non-compounding languages with an underspecified morphological system (such as English), on the other hand, can be handled with a so-called fullform list alone, where every word form is associated with one or more reading lines providing PoS and other linguistic categories.

Consider the following (ambiguous) input to a CG disambiguation grammar:

```
The
        "the" <def> ART S/P             (definite article singular/plural)
test
        "test" N S NOM                  (noun, singular, nominative)
        "test" V PR -3S                 (verb present tense, non-third-person)
        "test" V INF                    (verb, infinitive)
        "test"  V IMP                   (verb, imperative)
showed
        "show"  V IMPF                  (verb, past tense)
        "show"  V PCP2                  (verb, past participle)
        "show"  V IMPF                  (verb, past tense)
interpretable
        "interpretable"  ADJ POS        (adjective)
 results
        "result"  N P NOM               (noun, plural, nominative)
        "result"  V PR 3S               (verb, third person singular)
$.
```

In this example, the word form *'results'* can either be looked up in a full form list providing both noun and verb readings, or it can be analyzed as *result+s,* with the s-morpheme triggering the plural and third person inflexional readings, and the root *result* being verified in a separate base form lexicon. Even without lexicon support, there is a fair chance that an English word ending in *-s* will be a plural noun or a third person verb, while such guesses are impossible for uninflected and closed-class words.

In the full form list approach, inflected forms can be generated from lemma-lists using paradigmatic information about the language, or by using an FST in generative mode. But while only generation will ensure the inclusion of all forms of a lemma, many of these forms may be very rare, and a reasonable fullform list may be built simply by extracting all unique word forms (and their different analyses) from an annotated corpus. In a unix environment this can be done with an extremely simply command pipe. Assuming a corpus in .txt format with line-separated tokenization:

```
cat corpus | sort | uniq -c | sort -r -n
```

In the analyzer approach, for languages with a rich inflexion, a full FST can be approximated by pairing a lexeme list (for instance from a translation dictionary) and small analyzer program for all possible inflexion endings, but without paradigmatical information. Such an analyzer will come up with many overgenerated readings, but for each reading the hypothetical root can be checked against the lexeme lexicon, and a certain amount of false readings due to paradigmatical confusion can be weeded out by contextual disambiguation at the CG grammar stage.

In the absence of lexica and analyzers, and with only a small annotated corpus available, we still think a rule-based system can be bootstrapped just as well as a statistical one. What we propose is to combine a mini-lexicon with an word-endings guesser. Though requiring a minimum of manual linguistics, and a text book or other, the mini-lexicon is necessary to cover highly frequent closed-class words (conjunctions, pronouns, functional adverbs) and irregular verb or noun forms. And because, in general, this kind of words is overrepresented in the high-frequency bracket, a large portion may actually be covered by even a small toy corpus. In order to bypass the rest of the lexicon and the morphological analyzer, a second mini-lexicon is compiled from *all* words in the corpus - or even selectively those words in the corpus that are *not* part of the irregular and closed-class first mini-lexicon. Depending on the size of this word inventory, we will regard the last 3-5 letters of each word form as an “endings” string - a heuristic guess at real world inflexion or affixation endings or ethymological word class regularities. The adjective *interpretable,* for instance, can be guessed at from the following 4pattern, derived from more common words, like *available, considerable, acceptable, capable, valuable, unable:*

xxxable ADJ

Using a frequency breakdown of these “endings”, we can then assign PoS and linguistic categories to unknown word forms, using a longest match strategy. For endings with multiple readings, only the most frequent ones will be included in the annotation and passed on to the CG disambiguation grammar. As a consequence of this strategy, corpus errors, misspellings and hapaxes will automatically be weeded out, and the noisy frequency tail of the word inventory ignored. For our English miniature test corpus with only 4.000 words, the top frequency end of the resulting 3-letter endings lexicon looked like this (with prefixed frequency counts):

```
50 xxxion,  N S NOM
44 xxxing,  V PCP1
30 xxxons,  N P NOM
21 xxxing,  N S NOM
18 xxxers,  N P NOM
17 xxxity,  N S NOM
15 xxxlly,  ADV
14 xxxnts,  N P NOM
```

```
14 xxxies,  N P NOM
14 xxxent,  N S NOM
13 xxxnce,  N S NOM
12 xxxble,  ADJ POS
10 xxxing,  ADJ POS
10 xxxely,  ADV
10 xxxcal,  ADJ POS
 9 xxxess,  N S NOM
 8 xxxtes,  N P NOM
 8 xxxted,  V PCP2 PAS
 8 xxxent,  ADJ POS
 8 xxxces,  N P NOM
```

As can be seen, the list nicely manages both inflexion (*-ion/-ions)* and derivation *(-ble,-lly)*. For *-ing,* verbonominal ambiguity is covered, as is the N/ADJ ambiguity of *-ent.*

In principle, there are two diffent sources for endings patterns and their statistics. On the one hand, corpora can be used, leading to an emphasis on tokens and true frequency. On the other hand a dictionary can be used, with a type focus, and “frequency” meaning that there are many words in a language with a given ending. While the former is a better model for a bare-bones system relying heavily on endings also for common words, the latter is better suited to predict the analysis of a small remaining percentage of rarer, unknown words. Therefore, in out evaluation, we used one method for the lexicon-free system, and the other for experiments with lexicon sizes.

4 A Constraint Grammar analyzer

A classical CG rule, as allowed in the cg-1 (LingSoft Oy) and cg-2 (Tapanainen 1996) formalisms, is a disambiguation rule that discards or selects reading lines from a readings cohort, or individual tags from a single readings line:

```
REMOVE (VFIN) IF (-1C ART OR DET) ; # remove a finite verb reading if the word to the left is a safe (C) article or determiner
SELECT (VFIN) IF (NOT *-1 VFIN) (NOT *1 VFIN) ; # select a finite verb reading if there is no other finite verb neither left (*-1) or right (*1)
```

This reductionist methodology was complemented by ADD and MAP rules used for adding syntactic function tag canditates (subject, object etc.) for higher level disambiguation. However, in the original formalism, these tag-adding rules could not make reference to word form string elements, and could not be used to add entire reading lines to unknown tokens. In our own, new CG rule formalism, cg-3, we therefore implemented a new rule type, APPEND, specifically targeting morphological work in the face of lacking or incomplete lexical resources. We also added full support for regular expressions and string variables in both target and context matches. Apart from the obvious application, morphological analysis of the target word, regular expression matches can also be used for on-the-fly reference to grammatical sets of words in context conditions, at least for features with some form of regular surface representation, such as using the *'-ize'* affix as a marker for verb transitivity in English, or aspect prefixes in slavic languages.

For closed-class words like prepositions and uninflecting words like adverbs, an APPEND rule is very simpel, and equivalent to a full form list:

```
APPEND ("$1"v <atemp> ADV) TARGET ("<(always|ever|never|now|soon|today|tomorrow|yesterday)>"r) ;
```

For affixation[1], as in the adjective example below, we use <safe> and <heur> (heuristic) markers, the idea being that safe mappings will prevent later rules from adding further, heuristic readings for competing word classes:

```
  APPEND ("$1"v <safe> ADJ) TARGET ("<(.*(ic|oid|ous))>"r) ; # spastic, android, gaseous
  APPEND ("$1"v <heur> ADJ) TARGET ("<(.*(al))>"r) ; # accidental
```

Here, the second rule will allow a later noun mapping for *-al (e.g. withdrawal).*

Obviously, rule order is important, and more specific rules, with more morphological information to draw on, must precede default rules, the last of which will be a singular noun reading for most languages, given the dominance and productivity of nouns in the lexicon. The following noun rules show how a certain amount of root alterations can be handled, overriding more general rules:

```
  APPEND ("$1y"v <heur> N P NOM) TARGET ("<(.*)ies>"r) (NOT 0 <lex>) ; # flies
  APPEND ("$1"v <safe> N S GEN) TARGET ("<(.*)'s>"r) (NOT 0 <lex> OR (N)) ;  # fly's
  APPEND ("$1"v <safe> N P GEN) TARGET ("<(.*s)'>"r) (NOT 0 <lex> OR (N)) ;  # flies'
  APPEND ("$1"v <heur> N P NOM) TARGET ("<(.*)s>"r) (NOT 0 <lex> OR (N)) ;  # pies
```

Note the ie->y change in the analysis of “flies”, and the fact that the genitive-s analysis, deemed <safe>, will overrule the general plural s-stripping in the fourth rule.

All in all, for English, some 20 APPEND rules can provide very good morphological coverage on top of a small lexicon of irregular base forms, and with no analyzer in the program chain. A certain degree of unavoidable over-generation of ambiguity can be handled by an ordinary disambiguation CG - the same contextual rules working for true ambiguity (e.g. N/V for *tests*) will also tackle heuristic ambiguity (e.g. N/V for *xxxs*). For inflecting or agglutinating languages, more APPEND rules will be needed, but they will also be much more precise and be less prone to systematic heuristic ambiguity. We have evaluated the performance effect of our morphological CG heuristics for English, but expect it to be better rather than worse, for these other language types.

5 Evaluation

In order to evaluate the various suggested setups for a bare-bones CG system, we are using our English CG system as a chain of replaceable black box modules. Thus, we reduced the original morphological analyzer to a mere look-up program with access only to a mini-lexicon of irregular and closed-class words, inactivating the real, verbo-nominal lexicon. Likewise, the disambiguation CG was not allowed access to valency or semantic class tags from the main lexicon. To maintain tokenization compatibility, and thus matchability with the gold standard annotation, we retained the system's own preprocessor, but for a new language a working version could be produced with a few lines of code, basically line-breaking on space and punctuation, with a few exceptions for abbreviations and - depending on the language - contractions, enclitics and possibly names.

All experiments were performed on a random text chunk from the English Wikipedia (en.wikipedia.org), with 3957 tokens (3510 functional words, 1699 word form types). 48.6% of running word form tokens (1705 words), or 80% of word form types (1368) were “unknown” once the main lexicon had been inactivated.

[1] Rules like these can be improved considerably, and be made to include much longer and more fine-grained lists of derivational morphemes. On the other hand, many suffixes will also be captures by a good endings heuristics (cp. below), which is why our rule was not made more elaborate in the current setup.

5.1 The effect of APPEND rules and endings heuristics

In all evaluation scenarios we used a small CG module with about 40 APPEND rules for the heuristic assignment of part of speech (PoS) and inflexion tags to unknown words. Without a lexicon or analyzer, this module alone permitted the disambiguation CG a part-of-speech F-Score of 87. Without a lexicon, the original analyzer helped only a little (89.6 for PoS), and its effect could be matched and surpassed by using a bare-bones analyzer using endings heuristics learned from a corpus:

Table 1: performance with various input combinations for CG disambiguation grammar

	No lexicon no analyzer	No lexicon but analyzer	No lexicon but endings heuristics	Full system
full lexicon				+
base lexicon	+	+		+
original analyzer		+		+
mini-analyzer with endings heuristics (4 letters)			+ (3.5 - 310 M words)	
morph. APPEND cg	+	+	+	
CG disambiguation	+	+	+	+
PoS F-score	87	89.6	89.1-89.8	99.6
Morphology F-score	85.1	84.8	84.1-86.4	98.5
Base form F-score	93.1	93.4	93.3-94.6	99.2

Since endings heuristics are a very cheap resource to produce, we searched for the optimal number of letters to use, and investigated the effect of the size of the corpus used for learning endings patterns. In the table below, we looked at the last 2,3,4,5 or 6 letters, respectively, and distinguish between 3 different corpus sources:

1. Wikipedia test chunk endings (ca. 3.500 tokens revised), i.e. same corpus
2. Leipzig Internet corpus[2] (ca. 3.500 words revised), i.e. small, but different corpus
3. Leipzig Internet corpus (ca. 310.000 words unrevised), i.e. moderate treebank size

Table 2: performance with various input combinations for CG disambiguation grammar (full disambiguation or choose first reading)

endings heuristics number of letters	2	3	4	5	6
PoS F-score	87.3 84.2 87.7	89.3 86.1 89.2	91.9 **89.1** **89.8**	**92.8** **89.1** 89.6	92.1 88.6 89.5
Morphology F-score	86.4 83.6 86.4	88.2 83.2 **86.7**	90.3 **84.1** 86.4	**90.6** **84.1** 85.4	88.2 83.2 84.8
Base form F-score	92.2 91.9 92.1	93.9 92.7 93.7	95 **93.3** **94.6**	**95.6** **93.3** 94.1	94.9 92.9 93.

[2] (Quaesthoff et al. 2006)

As one might expect, the use of same-corpus data performed best, both for genre and lexicon reasons, but this is not a realistic scenario outside the realm of Wall Street Journal linguistics. Interestingly, the best cross-corpus F-Score for PoS was achieved with 4-letter endings[3]. The likely reason for this is that fewer letters will not capture enough morphemes, and leave too much ambiguity, while using too many letters may lead to a lower level of "abstraction", and hence, coverage problems. Performance rose when using more corpus data[4], even though we used unrevised annotations for the larger data set, and revised annotation for the small one.

Base form recognition performed better than PoS, and morphology worse, but both was to be expected, since, for instance, English verbs and nouns may share a common base form *(a house - to house),* while especially verbs have a great deal of 0-morpheme inflexion in English *(e.g. house V INF, house V -3S, house V IMP),* allowing for morphological mistaggings even in the face of a correctly recognized V part of speech.

5.2 Recall ceilings before disambiguation

All of the above tests targeted overall performance, through the optics of an existing English disambiguation grammar. However, the rules in this grammar were written for input from a regular analyzer drawing on a large lexicon (> 200.000 lexemes) complete with valency and semantic class information. In order to get an idea about the potential performance ceiling of a new grammar, optimized for the kind of heuristic, over-ambiguous input our bare-bones setup is providing, it is interesting to know the point of departure for a disambiguation grammar, i.e. the recall and precision of a bare-bones analyzer with access only to an endings heuristics and morphological APPEND rules.

Table 3: Recall / precision / F-score for bare-bones analyzer **without** disambiguation CG (endings lexicon from unrevised 310.000 word corpus)

endings heuristics	3 letters Recall/Prec./F-score	4 letters Recall/Prec./F-score	5 letters Recall/Prec./F-score
PoS	**95.6** /47.7 / 63	95 / 50.6 / 66	94.3 / 51.5 / 66.6
Morphology	**93.9** / 46.9 / 62.6	92.7 / 49.4 / 64.5	91.4 / 49.9 / 64.6
Base form	95.6 / 81.6 / 88	**96.1** / 84.8 / 90.1	95.4 / 85.3 / 90.1

The numbers show that for writing a disambiguation grammar from scratch, it might actually be a good idea to use shorter ending strings for heuristics, since the implicit over-generation will ensure a higher recall. Conversely, precision is higher with the longer 5-letter endings heuristics, giving an existing (reductionist) disambiguation grammar less room to make errors. The latter effect, however, is dependent on a sufficiently large corpus base - with a small corpus, even a manually revised one (!), *both* recall and precision will be higher with the more generalizing 3-letter endings:

Table 4: recall / precision / F-score for bare-bones analyzer without disambiguation CG (endings lexicon from different, revised 3.500 word corpus)

endings heuristics	3 letters	4 letters	5 letters
PoS F-score	**93.9** / 53 / 67.8	93.6 / 51.9 / 66.8	93.6 / 51.6 / 66.6

[3] For same-corpus training, the optimal number of letters may be a little higher, since longer string matches may mean a more narrow modeling of individual, frequent words in the corpus.

[4] Still, this is also true of full form lexica built from the same corpora, so with bigger corpora available, the need to have endings heuristics in the first place will decrease.

Morphology F-score	**91.4** / 51.6 / 66	90.5 / 50.2 / 64.6	90.6 / 49.9 / 64.4
Base form F-score	**94.5** / 84.7 / 89.3	94.2 / 84.6 / 89.1	94.1 / 84.6 / 89.1

5.3 The effect of fullform mini-lexica

Finally, we simulated the availability of a corpus-derived fullform lexicon, to enhance our bare-bones system, assuming that no real lexicon resource would be available, and that available corpora would not be human-revised. For this experiment we used the English Leipzig Internet corpus (2 million words, Quaesthoff et al. 2006), automatically annotated with the EngGram parser (Bick 2009). The corpus contained a maximum of 180.000 different non-hapax word form types, not counting numerical expressions and compound names. Of these, we used the 1000, 10,000 and 100,000 most frequent forms to enhance the base lexicon of the bare-bones system. As before, we used both an endings-based heuristics and morphological APPEND rules to handle remaining unknown forms, and ran the evaluation metrics after disambiguation. For the endings heuristics we settled for a string length of 4, and used the same frequency chunk to build the endings lexicon, i.e. combining the 1.000 word min-lexicon with an endings lexicon likewise built from only 1.000 word forms, etc. One important difference between a fullform lexicon derived from a small corpus, and one generated from complete lexical paradigms (or simply written by hand) is that the former does not guarantee cohort completeness - for instance, the most frequent analyses of the word *'close'* (say, adjective and infinitive) may be represented in the corpus, while others (say, finite verb and imperative) are missing. To simulate a generated cohort-complete resource for comparison, we ran an alternative evaluation where rarer forms below the chosen frequency threshold were added, *if* another, more frequent reading for the same form had been found *above* the threshold. Figures for both scenarios are given in the table below, the cohort-complete results in parentheses.

Table 5: The influence of mini-lexicon size (endings files built from same lexicon size)

mini lexicon size in words	1 000	10 000	100 000	180 000 (all)
PoS F-score	90 (89.8)	93.7 (93.3)	**95.3** (95.2)	95.2 (95.2)
Morphology F-score	87 (87.2)	91.8 (91.7)	**94.1 (94.1)**	**94.1 (94.1)**
Base form F-score	93.6 (93.3)	96 (95.8)	**96.8 (96.8)**	**96.8 (96.8)**

The figures show the expected performance increase with bigger corpora, but also that growth is asymptotic - the Zipf tail of the frequency, i.e. frequencies close to the hapax level, do not contribute. An interesting finding is that completing cohorts with rarer readings does not help. The expected beneficial recall effect is canceled out either by the loss of implicit statistical disambiguation, or by corpus annotation errors found in the low frequency bracket. In this light, it must be emphasized that no human revision of either corpus or lexicon was involved, and that cohort-completeness was simulated from corpus data. In a development setting, better results might be achieved with revised corpora (treebanks), or by manually cohort-completing the top frequency bracket of a corpus-derived lexicon.

6 Conclusions and Outlook

We have shown how the necessary morphological input for the disambiguation rules of a Constraint Grammar tagger can be generated in a resource-poor environment. To bypass the usual lexico-analytical setup (analyzers, lexica, finite state machines), we compared and combined 3 methods - (a) morphological APPEND rules, (b) endings heuristics and (c) corpus derived mini-lexica. With an annotated, but unrevised 2-million word corpus as “maximal resource", the best combination of a-c achieved F-scores of 95.3 for PoS and 96.8 for lemmatization, respectively, even in a cross-genre evaluation. Even with a lexicon of only closed-class words, a combination of (a) and (b) came close to 90% correctness for PoS, with a theoretical recall ceiling of 95.6 without disambiguation. Future research should determine if using a newly written, tailor-made disambiguation grammar could exploit this recall ceiling better than the existing grammar made for non-heuristic lexico-analytical input.

An important lesson learned, and a nice aspect for resource-savers, was the fact that for generating data-driven lexica, corpus size matters more than whether the corpus annotation has been human-revised - this being true both for full form lexica and endings heuristics. For the latter, we established 4 letters as the optimal string length to use for English[5].

We chose English for our experiments not because CG has a special affinity or effectiveness for English, but simply to make our research maximally accessible for the research community as a whole. However, our intuition is that the effectiveness of both endings-based heuristics and morphological APPEND rules should be higher for languages with a richer morphology, as would be the case for most other Indo-European languages, for instance. On the other hand, the coverage of same-size fullform lexica should be lower for those languages. In a bare-bones setting we believe the treatment of unknown word forms to be more important than lexicon coverage, but this intuition need to be corroborated by future research on representative members of various language families.

References

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

Bick, Eckhard. 2009. "Introducing Probabilistic Information in Constraint Grammar Parsing". *Proceedings of Corpus Linguistics 2009, Liverpool, UK.* Electronically published at: ucrel.lancs.ac.uk/publications/cl2009/

Chanod, Jean-Pierre and Pasi Tapanainen. 1995. *Tagging French – comparing a statistical and a constraint-based method*. In Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics, pp. 149–156, Dublin, Ireland, March. ACL.

Clément, Lionel, Benoît Sagot and Bernard Lang. 2004. Morphology based automatic acquisition of large-coverage lexica. In *Proceedings of the 4th Language Resources and Evaluation Conference* (LREC'04), pages 1841–1844, Lisbon, Portugal.

Creutz, Mathias and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05), pp 106–113, Espoo, Finland.

Forsberg, Markus, Harald Hammarström and Aarne Ranta. 2006. Morphological Lexicon Extraction from Raw Text Data. In *Proceedings of FinTAL'2006.* pp.488~499

Goldsmith, John A. 2001. Unsupervised learning of the morphology of a natural language. Computational Linguistics, 27(2): pp. 153–198.

Karlsson, Fred, Atro Voutilainen, Juha Heikkilä, and Arto Anttila, editors. 1995. *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Natural Language Processing, No 4. Mouton de Gruyter, Berlin and New York

Nikulásdóttir, Anna Björk and Matthew Whelpton. 2009. Automatic Extraction of Semantic Relations. In: Pedersen, Bolette Sandford et al. (eds): *Proceedings of the NODALIDA 2009 workshop WordNets and other Lexical Semantic Resources -*

[5] Without disambiguation, in pure recall terms, 3 letters were better, but at too high a price in terms of ambiguity.

between Lexical Semantics, Lexicography, Terminology and Formal Ontologies. NEALT Proceedings Series, Vol 7. ISSN 1736-6305

Quasthoff, U.; M. Richter; C. Biemann. 2006. “Corpus Portal for Search in Monolingual Corpora”. *Proceedings of the fifth international conference on Language Resources and Evaluation, LREC 2006, Genoa.* pp. 1799-1802

Piasecki, Maciej and Adam Radziszewski. 2007. Polish Morphological Guesser Based on a Statistical A Tergo Index . In: *Proceedings of the International Multiconference on Computer Science and Information Technology*. pp. 247–256 . ISSN 1896-7094

Tapanainen, Pasi. 1996. *The Constraint Grammar Parser CG-2*. No 27, Publications of the Department of General Linguistics, University of Helsinki.