Arboretum, a hybrid treebank for Danish

Eckhard Bick

Institute of Language and Communication, University of Southern Denmark

lineb@hum.au.dk, Rugbjergvej 98, DK-8260 Viby J

tel. +45-86283524, fax +45-86281397

Abstract

This paper presents ongoing work on a newspaper and mixed genre treebank for Danish, describing both the corpus as such (data, format, size etc.) and the tools and routines used to create it. The corpus is annotated by layered use of automatic CG and PSG grammars (with subsequent revision), drawing robustness from the former og depth from the latter, and yielding a hybrid format specifying both dependency, constituent structure and syntactic function. Descriptive issues such as discontinuity, coordination and valency are discussed, and the interplay between CG- and PSG parsing rules is evaluated. Using a special attachment CG, automatic analysis of running text resulted in well-formed trees in 70-75% of cases, the median number of trees per sentence being 1.41.

1. Introduction

Syntactic treebanks are a useful but labour-intensive resource both for linguistic research, NLP tool-development and grammar teaching. Not least for a small language like Danish, it makes sense to optimise the interplay of efficient parsing tools and manual revision in treebank building, and to strive for "filterability", i.e. to encode grammatical information in reasonably theory-neutral ways. Thus, the current project, *Arboretum,* makes use of existing Constraint Grammar technology for Danish, and maintains notational compatibility with VISL's[1] teaching tree banks in 22 languages (http://visl.sdu.dk), as well as a sister treebank for Portuguese (*Floresta Sintá(c)tica*, Afonso et.al. 2002), and a recently initiated treebank for French[2], *L'Arboratoire* (http://corp.hum.sdu.dk/ arboratoire.html).

Arboretum can be said to be "hybrid" in two ways: On the one hand, its format is maintained in 2 parallel, but not wholly information-equivalent, formats, (a) word based shallow dependency tags, and (b) VISL-style constituent trees, with both formats sharing tags for syntactic function and morphological form. On the other hand, the treebank construction process is hybrid, too, mirroring the parallel format in a sequential way, progressing

[1] VISL ("Visual Interactive Syntax Learning") is a multi-language research initiative at the University of Southern Denmark, developing NLP-tools, corpora and internet based grammar teaching tools.

[2] A joint venture with Susanne Salmon-Alt and Ane Dybro Johansen (ATILF - Loria-LED).

from a lexicon and rule driven morphosyntactic analysis to shallow dependency parsing and finally a function based constituent analysis, by progressively applying various Constraint Grammar (CG) and Phrase Structure Grammar (PSG) modules. A third format (c), with full dependency specification, has recently been introduced. Like (b), it is generated by working on output from (a) - in this case applying not a PSG, but a Prolog based dependency matcher[3].

2. Tools for automatic annotation

Morphosyntactic disambiguation, syntactic function and dependency annotation are handled with Constraint Grammar tools (Karlsson et.al. 1995), which are also used to add shallow word based semantic information, like case roles or named entity types. Finally, CG output is used as input to a PSG system, with rewriting rules using as terminals not words, but syntactic categories and form types from the CG level. Within the VISL project (http://visl.sdu.dk), I have designed such hybrid systems for a number of Romance and Germanic languages[4], in particular for Danish (Bick 2001) and Portuguese (Bick 2000). Though corpus annotation and grammar teaching have been the prime applicational targets, other areas, like MT and information extraction, have been explored in pilot studies.

The Danish hybrid parser, *DanGram*, uses around 7000 CG-rules and 500 PSG meta rules, gaining robustness from the former and depth from the latter. Constraint Grammar is, in its essence, a reductionist disambiguation technique, that will - unlike systems built exclusively on rewriting rules - always yield at least one analysis for any given sentence. This inherent robustness is enhanced by the fact that global context can override for instance local agreement errors, and that constituent order restrictions are expressed implicitly (by REMOVE rules and BARRIER contexts) rather than explicitly. Hence, a PSG relying on CG function terminals rather than word terminals, can then allow itself the luxury of less stringent rewriting rules, since - unlike ordinary PSG, HPSG, LFG etc. - it does not have to determine word and constituent function, but only "assemble" constituents from functions. Even unorthodox function chains can thus be allowed since the danger of overgenerating is greatly reduced by the fact that all input (i.e. CG-output) is "function-proof" and will only allow one function (or, in the case of true ambiguity, few functions) for any given constituent candidate.

DanGram's lexico-semantic data base covers around 100.000 lexemes and 40.000 names, and for most items valency information and a semantic prototype class is provided. For running text, correctness rates (F-scores) for word class and syntactic function are around 99% and 95%,

---

[3] The Prolog dependency program is being developed by Søren Harder as part of his Ph.D.-research at VISL.

[4] VISL as a teaching and "edutainment" system has small treebanks for 22 languages, while Constraint Grammars and PSG's exist for Danish, English, German, Portuguese, Spanish and French, allowing live corpus based tree-building.

respectively. These numbers are comparable to CG-results for other languages, and compare favourably to non-CG tagging/parsing techniques (Bick 2003). Category-specification of F-scores indicates a certain variation across categories, with, for instance, left-subjects and right-objects performing better than right-subjects and left-objects, suggesting manual revision should focus more on the latter than the former.

At present, subsequent constituent analysis of *uncorrected* CG-input yields complete "legal" (i.e. well-formed) trees for about 70-75% of Danish sentences, and since "break down points" in the remaining partial trees are often isolated and clearly identifiable, heuristic repair mechanisms can be imagined as a future line of research.

On fully corrected input from a 200 sentence test chunk, 95% of sentences were assigned at least one full tree structure, with an added attachment error rate of around 0.8% after tree-ranking[5].

3. The corpus

The current data target for the Danish treebank is a 10 million word subset from DSL's sentence randomised Korpus90/2000 (Asmussen 2002 and Bick 2002). This mother corpus contains 52 million words, which have previously been *DanGram*-annotated up to the CG-syntactic level (http://corp.hum.sdu.dk). Though in principle mixed genre, the newest half of this corpus has a strong news text bias in quantitative terms. It was intended as a "linguistic snapshot" of the Danish language around the millenium shift, and provides good coverage of current creative language usage, with a challenging syntactic complexity and without too many orthographical errors.

3.1. The annotation scheme

Though most treebanks are intended as reference corpora for broad syntactic research in a given language, it is difficult to please all users, and a methodological or descriptional bias towards one linguistic theory or other is all but unavoidable. "Classical" treebanks like the English Penn and SUSANNE treebanks (www.cis.upenn.edu/~treebank/ and www.grsampson.net/ RSue.html), are in principle based on bracketing structure, but enriched with function labels, while Dependency Grammar has been embraced by a number of more recent initiatives like the large Czech PDT (quest.ms.mff.cuni.cz/ pdt/Corpora/PDT_1.0/index.html), and smaller treebanks for Turkish (www.ii.metu.edu.tr/ ~corpus/treebank/), Russian (www.iitp.ru/ iitp/lab15e.htm), Danish (www.id.cbs.dk/ ~mtk/treebank/) and Italian (www.di.unito.it/ ~tutreeb/). Since large treebanks profit from automatic parsing technology, NLP-tools interact with descriptional issues. Examples are the use of HPSG in the Dutch Alpino-Treebank

[5] This last number would turn out less favourable, though, if "fully corrected" did *not* include helping tags from the additional attachment CG.

(odur.let.rug.nl/ ~vannoord/trees/) and the Bulgarian BulTreeBank (www.bultreebank.org/), or the use of LFG in the Spanish UAM Treebank (www.lllf.uam.es/~sandoval/UAMTreebank.html) and last not least the hybrid PSG/Dependency TIGER-treebank for German (www.ims.uni-stuttgart.de/ projekte/TIGER/), where LFG is used as one of several modules.

Arboretum itself comes in two parallel flavours, (a) a dependency treebank with word based CG-annotation (fig. 1) and (b) a PSG-treebank with constituent annotation (fig. 2). Both versions allow crossing branches/discontinuity and specify function as well as structure, and both can be converted into graphical formats (fig. 3). Tokenisation is in part lexicon-driven (polylexical function words like 'på_grund_af', 'ikke_desto_mindre'), in part grammar driven (especially name chains), and in some cases subject to disambiguation. Though Arboretum does provide token-level information (lemma, PoS, inflexion, semantic class of names), this discussion will focus primarily on syntax, arguably the main raison-d'être of treebanks.

3.1.1. Dependency, form and function

At the CG-level, each Arboretum-token is assigned both a function tag and a directed shallow CG-dependency, pointing to a head-category explicitly (@>N prenominal) or implicitly (@<SUBJ subject right of verb). These markers are, together with the uniqueness principle and secondary attachment tags, used to compute a full numbered dependency (e.g. #5 = dependent of word 5). AT the PSG-level, a head-function (H) is retained in groups, as well as group-specific dependency-functions (e.g. DN for nominal groups).

(Figure 1)

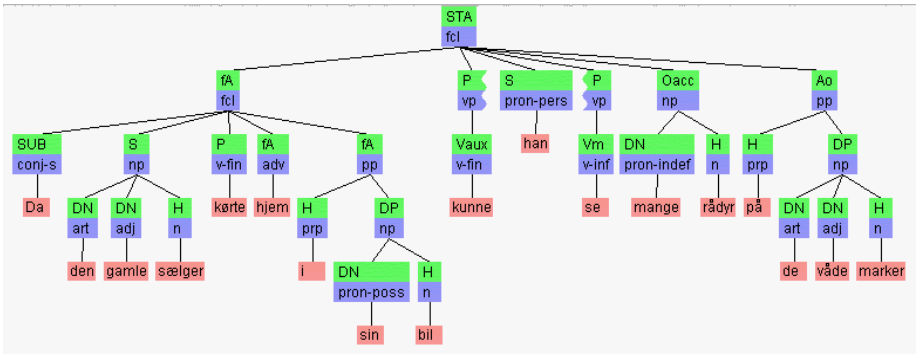
```
1.Da (When)          [da] KS                          @SUB     #5
2.den (the)          [den] ART UTR S DEF               @>N     #4
3.gamle (old)        [gammel] ADJ nG S DEF NOM         @>N     #4
4.sælger (salesman)[sælger] N UTR S IDF NOM          @SUBJ>   #5
5.kørte (drove)      [køre] <mv> V IMPF AKT         @FS-ADVL>#10
6.hjem (home)        [hjem] ADV DIR                   @<SA     #5
7.i (in)             [i] PRP                          @<ADVL   #5
8.sin (his)          [sin] <refl> DET UTR S             @>N    #9
9.bil (car)          [bil] N UTR S IDF NOM              @P<    #7
,
10.kunne (could)     [se] <aux> V IMPF AKT           @FAUX     #0
11.han (he)          [han] PERS UTR 3S NOM           @<SUBJ    #10
12.se (see)          [se] <mv> V INF AKT             @AUX<     #10
13.mange (many)      [mange] <qu> DET nG P NOM        @>N      #14
14.rådyr (deer)      [rådyr] N NEU P IDF NOM         @<ACC     #12
15.på (in)           [på] PRP                        @<OA      #12
16.de (the)          [den] ART nG P DEF                @>N     #18
17.våde (wet)        [våd] ADJ nG P nD NOM             @>N     #18
18.marker (fields)   [mark] N UTR P IDF NOM            @P<     #15
```

Each node in the tree-format is marked for both *function* and *form* (FUNCTION:form), while depth (in the internal format, fig. 2) is expressed as indentation (number of '='-signs at line start.

(Figure 2)

STA:fcl		
fA:fcl		
=SUB:conj-s	('da')	*Da (When)*
=S:np		
==DN:art	('den' UTR S DEF)	*den (the)*
==DN:adj	('gammel' nG S DEF NOM)	*gamle (old)*
==H:n	('sælger' UTR S IDF NOM)	*sælger (salesman)*
=P:v-fin	('køre' IMPF AKT)	*kørte (drove)*
=As:adv	('hjem' DIR)	*hjem (home)*
=fA:pp		
==H:prp	('in')	*i (in)*
==DP:np		
===DN:pron-poss	('sin' <refl> UTR S)	*sin (his)*
===H:n	('bil' UTR S IDF NOM)	*bil (car)*
P:vp-		
=Vaux:v-fin	('kunne' IMPF AKT)	*kunne (could)*
S:pron-pers	('han' UTR 3S NOM)	*han (he)*
-P:vp		
=Vm:v-inf	('se' AKT)	*se (see)*
Od:np		
=DN:pron-indef	('mange' <quant> nG P NOM)	*mange (many)*
=H:n	('rådyr' NEU P IDF NOM)	*rådyr (deer)*
Ao:pp		
=H:prp	('på')	*på (in)*
=DP:np		
==DN:art	('den' nG P DEF)	*de (the)*
==DN:adj	('våd' nG nD NOM)	*våde (wet)*
==H:n	('mark' UTR P IDF NOM)	*marker (fields)*

Figure 3



Zero constituents are avoided by marking the function in question on the nearest daughter, and by letting dependent type determine group type: 'De syge lider' *(the sick suffer)*, for instance, will carry a CG-subject reading on

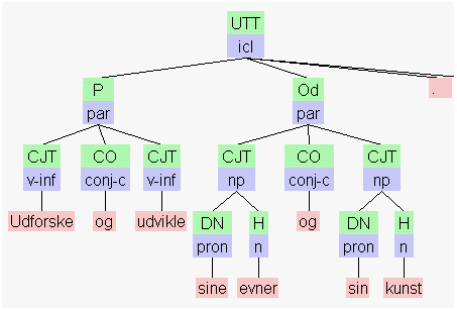
'syge', while maintaining adjective-PoS. At the PSG-level, the article-prenominal 'de' causes the group 'de syge' to be marked as np (noun phrase). Subject function will be passed on to this np, and 'syge' will become H. Similarly, no 1-daughter-nodes are allowed, and 'Bjørnen sover' will not contain an overt np-node, but rather carry the subject-tag directly on the noun (S:n) - as, of course, at the CG-level.

3.1.2. Discontinuity

Discontinuity (crossing branches), which is not a rare feature in Danish, is expressed by double-arrow-dependents in CG (meaning "cross over the next legal head, to another word of compatible type), and by "broken node"-markers in the constituent trees. In *'Uvejret havde jeg ikke regnet med',* for instance, the argument of preposition ('uvejret') is marked @>>P at the CG-level, and at the PSG-level, it is daughter of a left-half marked pp (Op:pp-) matching a right-half parent (-Op:pp) of 'med'. Discontinuous clauses *('Peter tror jeg ikke du kan slå')* are treated in a similar fashion (@ACC>> and Od:fcl-). For verb chains, discontinuity only arises at the constituent level (vp- ... -vp), since CG annotates main verbs as arguments of auxiliaries, attaching subjects to the first verb (i.e. the auxiliary) and objects to the last (i.e. the main verb).

3.1.3. Coordination

While traditional CG underspecifies coordination in a flat way (coordinated dependents are assumed to potentially attach to all "legal" heads at the same level, and coordinated heads are assumed to potentially govern all "legal" dependents at the same level), this is not satisfactory at a semantic level, for coordinated ellipsis, or when working on selection restrictions. At the price of an added error-burden, our constituent notation strives to remedy these shortcomings by specifying paratagmata both for known functions and "unknown" nexus-functions. Fig. 4 shows a coordinated predicator (P), where both verbs jointly govern the two np's of a coordinated direct object (Od).



P = Predicator, UTT = Utterance
Od = Direct/Accusative object
CJT = Conjunct, CO = Coordinator
H = Head, DN = Nominal dependent
par = paratagma, icl = non-finite clause
np = noun phrase, pron = pronoun
conj-c = coordinating conjunction
n = noun, v-inf = verb/infinitive

Figure 4

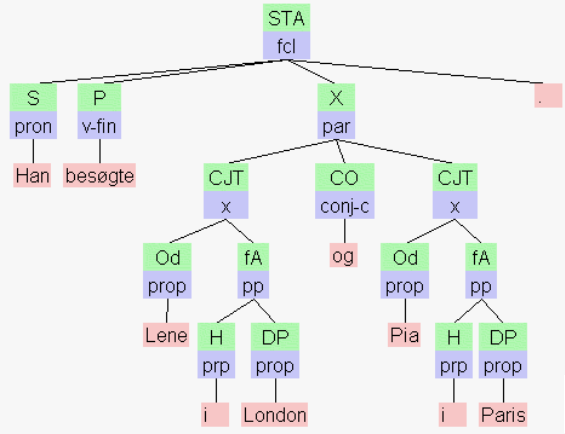


Fig. 5

Figure 5 shows a paratagma with unknown *function* (X), whose conjunct-constituents are assembled by joining 2 pairs of direct object (Od) and free adverbial (fA) into constituents of unknown *form* (x).

3.1.4. Valency

Drawing on its extensive valency lexicon, DanGram does annotate for valency, which in CG is marked both as a (disambiguated) valency-marker on the head (for instance, <va+LOC> for a locative adverbial argument), and - at the clause level - by different functions for the dependent. Maintaining these function tags, the treebank does not need to express the argument/adjunct-distinction by structural means. Pp-nodes, for instance, can carry the following function tags at clause level,

- (free) adunct adverbial (fA): *Peter arbejder i Paris. (P. works in Paris)*
- (bound) argument adverbial with subject relation (As):
  Peter bor i Paris. (Peter lives in Paris)
- (bound) prepositional object (Op): *Peter holder af Lene. (P. likes Lene)*

while valency so far has been underspecified in the group-level function tag ('arg' - 'argument' and 'mod' - 'modifier' are *not* used with DN/DA at present):

- adnominal dependent (DNmod): *en kvinde med stil (a woman with style)*
- adverbial dependent (DAarg): *vild med Maria (in love with Maria)*

Experimentally, case roles like Actor, Patient etc. are assigned by a special layer of CG rules[6], using function context, valency and lexical information handed down by the other CG-modules.

[6] This grammar was written by Søren Harder and can be tested at http://visl.sdu.dk, but is not used in the current version of the treebank.

3.2. Corpus creation

The treebank itself is created in a circular interplay between automatic analysis and manual linguistic revision. Thus, the CG format (fig.1) is revised *before* constituent trees are generated from it. Then, a *second round* of revision is performed (fig.2 format), with a more structural focus. Since PSG is more stringent and less robust than CG, even previously overlooked form- and function errors may be found at this stage, and if so, both formats will be updated in parallel. Finally, a graphical tree inspection (fig.3) is performed using VISL's ordinary tree manipulation tools. This triple revision, involving three different formats and at least two different annotators for any given sentence, obviously enhances revision accuracy, but also helps identify inter-annotator-disagreement regarding unresolved or un-standardized descriptional issues, which can then be marked, discussed, documented and - ultimately - implemented not only in future revision work, but also in the automatic parsing grammar.

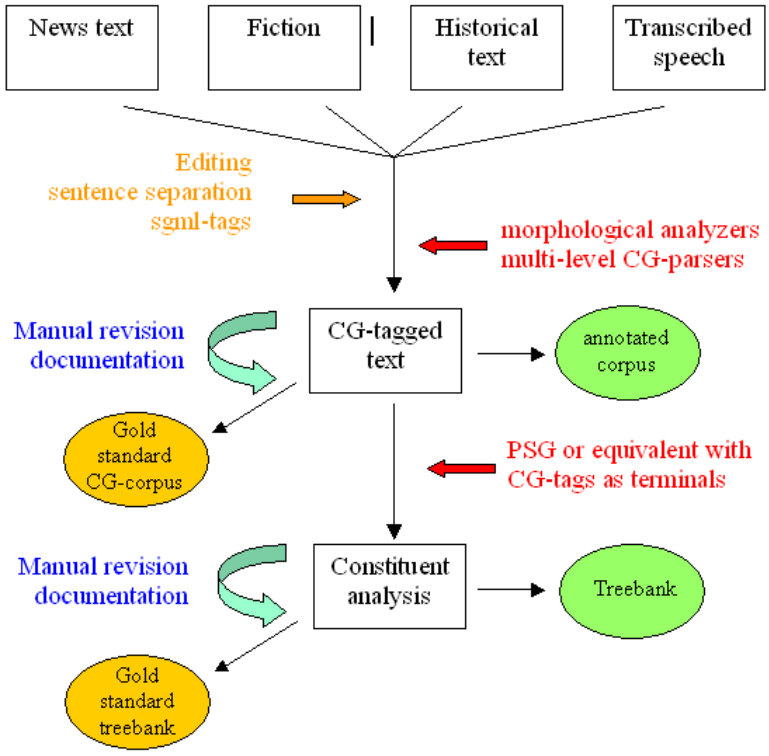


Fig. 6

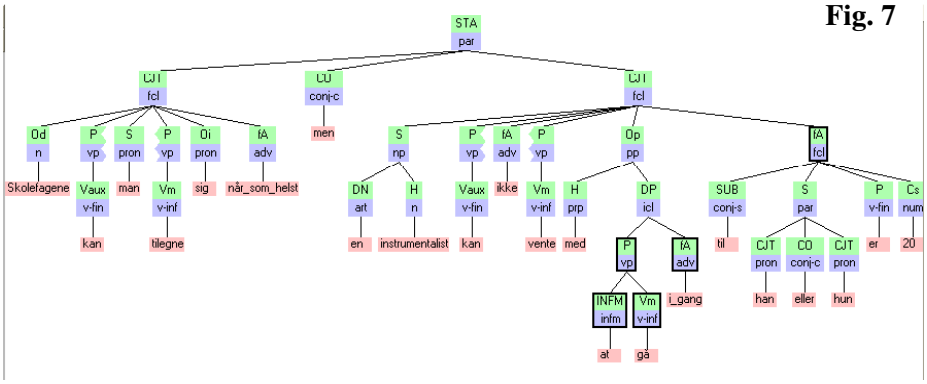
4. Corpus search tools

Search tools and graphical internet interfaces exist for both VISL-formats (CG and VISL-PSG), both in-house and - in the context of joint ventures involving other languages also using the VISL-tree-format - out-house. Apart

from simple grep-based tools (http://corp.hum.sdu.dk), CQP[7]-based interfaces have been written for the VISL-format by Diana Santos (Linguateca, Oslo)[8] and Paul Henriksen (VISL). Lars Nygaard (Tekstlaboratoriet, Oslo) has developed a MySQL-based search tool on Danish treebank-data, within the Nordic PaNoLa[9] framework. Finally, VISL-to-XML transformers, allowing the use of general XML-search tools and tree-manipulators, have been written by Paulo Quaresma (University of Évora) and Ane Dybro Johansen (Atilf), in a Portuguese and French context, respectively.

Fig. 7 shows a search result for a complex predicator (P:vp) followed by an object clause (Od:fcl).

Fig. 7



5. Challenges: Optimizing CG-PSG-interplay

In principle, the hybrid approach should delegate robustness to CG and structural finesse to PSG (as discussed in chapter 2). However, there are two kinds of obvious interference problems, where the PSG either gets too little or too much disambiguation from its CG-input[10].

First, CG disambiguation errors - automatic or manual - will propagate to the PSG-level, interfering with generative rules, that *might* have worked on ambiguous (i.e. less disambiguated) input. One possible solution - besides better manual revision - is to trade precision for recall by inactivating the most heuristic CG-rules, thus delegating at least some disambiguation of syntactic function tags to the PSG-level. In table 1, results are shown for

---

[7] Corpus Query Processor (IMS Corpus Workbench), developed at the Institut für Maschinelle Sprachverarbeitung, Stuttgart.

[8] This work targeted PALAVRAS-annotated Portuguese text, and includes a tree-searcher for the Floresta Sintá(c)tica treebank, Águia, at http://www.linguateca.pt

[9] A two-year, Norfa-funded project aiming at the integration of Nordic CG-research.

[10] Constraint Grammar, is in its essence, a disambiguation technique. Ambiguity is created routinely for almost every word as part of its methodology, on the one hand by using a full lexico-morphological analyser, on the other hand by *mapping* ambiguous syntactic functions and dependencies onto word classes. Robustness arises from the fact that contextual disambiguation progresses in a reductionist fashion, leaving the last surviving analysis untouched.

different levels of grammar-heuristicity, "syn0" referring to a run with only the safest rules, "syn0,1" to syn0-rules plus the less safe syn1-rules, etc.

Second, where the PSG *is* being used for disambiguation, i.e. structurally, as in the areas of attachment underspecification and coordination resolution, long and complex sentences sometimes create space & time-problems even on a linux system, and will in any case produce large parse forests instead of a single tree - thus burdening manual revision. Therefore, an intermediate *attachment CG* has been introduced, which - based on valency and semantic context - inserts so-called long- and close-attachment markers, as well as topological markers for adverbials and coordinator-based tags specifying the scope of a given coordinator. Apart from minimising ambiguity, this attachment grammar also specifies whether prepositions have fronted or elliptic arguments, a common structural problem in Danish.

However, if no manual revision is performed between CG-analysis and PSG, the attachment-grammar comes at a price - it increases the chance for incomplete "partial" PSG-parses (cp. corresponding rows in table 1 and 2), as does the other time-and-space-saving measure, heuristic CG-rules. Thus, without attachment-CG (table 1), the number of incomplete parses increases gradually from 24.3% (with only the safest rules) to 28.3% (all heuristic rules used).

Table 1

n = 400 ÷ attachCG	sentences with partial tree	average forest-size	max. forest	% sentences with 1 tree	% sentences with 2 trees
syn0	97 (24.3 %)	243.1 trees	26896	19.4	12.1
syn0,1	106 (26.5 %)	75.5 trees	3602	23.5	18.7
syn0,1,2	106 (26.5 %)	27.8 trees	1806	27.0	20.8
syn0,1,2,3	112 (28.0 %)	17.8 trees	1442	29.6	21.6
all/normal	**113** (28.3 %)	11.6 trees	728	**31.1**	22.4

Table 2

n = 400 + attachCG	sentences with partial tree	average forest-size	max. forest	% sentences with 1 tree	% sentences with 2 trees
syn0	118 (29.5 %)	159.5 trees	19680	25.1	15.1
syn0,1	114 (28.5 %)	22.0 trees	882	30.5	20.0
syn0,1,2	113 (28.3 %)	10.1 trees	456	36.0	22.0
syn0,1,2,3	117 (29.3 %)	7.1 trees	456	38.7	24.5
all/normal	**116** (29.0 %)	5.5 trees	376	**40.6**	24.7

Note that the effects of using attachment rules and heuristic rules are interdependent - at least with the current architecture - since less disambiguation of function tags means poor context for the attachment rules, which accept (ambiguous) EXIST-contexts rather than ask for safe, so-called C-contexts with one reading only. Thus, the number of partial trees with

attachment-CG (table 2) actually increases again - to 118 - if all heuristic rules are inactivated (syn0 and syn1).

One way to optimise the advantages and disadvantages of the heuristics and attachment grammars is a cascading system, where those and only those sentences that resulted in partial PSG-trees at the full run, would be rerun without attachment grammar and/or excluding the most heuristic level of syntactic CG-rules, thus letting survive more PoS and function tags, and creating more ambiguity for the PSG to work on. The process can be repeated with exclusion of the *second* most heuristic rules (with/without attachment-CG) etc., each time reducing the number of remaining "partial" sentences, while trusting results where CG and PSG in conjunction *did* give one or more analyses. Similarly, PSG-rules could be graded for heuristicity, too, allowing for cascaded reruns of the PSG-module itself.

For every sentence, a heuristic *tree chooser* program creates a priority list for surviving ambiguous trees, drawing on a variety of complexity measures, like embedding depth, coordination flatness and discontinuity[11]. Though with corrected CG-input and a good language-specific PSG, any ambiguous set of well-formed trees should contain the correct analysis/analyses for a given sentence, the likelyhood of the tree-chooser picking out the correct analysis among the others will decrease with growing forest size. Choices dictated by the attachment-CG will provide a better (and smaller) forest, but - where wrong - cannot be unmade by the tree chooser. Revising grammarians are free either to accept and correct the suggested tree or to choose among the full set of ambiguous trees. In one experiment, where 728 sentences in revised cg-format were psg-processed, 707 sentences received well-formed (complete) analyses, while 21 sentences resulted in "fragmented" (partial) trees. Among the completely tree-ified sentences, 40% had only 1 analysis, 28% had 2 analyses, and 4.2% had over 20 analyses. The median was 1.41, and the largest forest contained 864 trees.

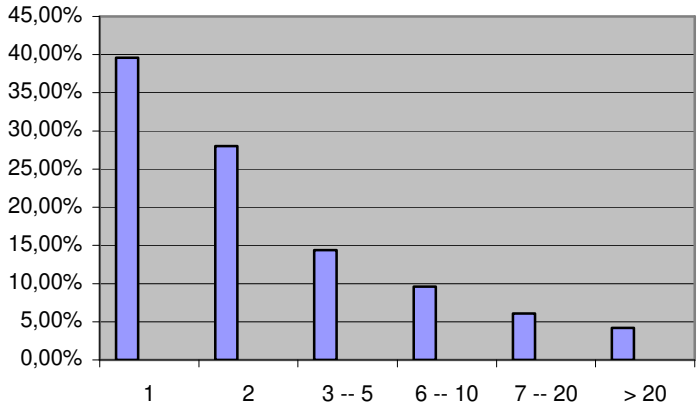


Fig. 8: percentage of forests with n trees

If a human annotator perceives true syntactic ambiguity in a sentences, or for reasons of linguistic theory wishes to maintain two different readings, he can

[11] For "partial" trees, depth optimisation is handled by the PSG rule-compiler itself.

do so by providing more than one tree for a sentence, or - in the case of local ambiguities - by adding alternative function or attachment-information to a given node. DN/fA[-2]:pp, for instance, expresses ambiguity between postnomnal attachment (DN) and free adverbial (fA), where [-2] is an instruction for "lifting" (or less indentation) of the node when read as fA. An ambiguous node will undergo "structural metamorphosis" if clicked upon in the graphical treebank-interface.

6. Outlook

Obviously, revising 10 million words is not a realistic task without dedicated funding, so a "decimal" corpus division is planned, with (1) a fully revised "gardener's" core of a few hundred thousand words, (2) a one million "forester's" reference corpus, with partial revision of major categories, and (3) the unrevised "jungle" corpus with automatic analysis only. (1) will be used to exemplify and document descriptional decisions, underlying linguistic theory and special features of Danish grammar, while (2) could be useful for training purposes and statistics, and (3) would allow research involving rare or lexically conditioned phenomena.

Presently, the garden core of the Arboretum corpus contains roughly 100.000 - 200.000 words (depending on format), which can be accessed and searched at the VISL site (http://corp.hum.sdu.dk/arboretum.html). Revision alone corresponds to over a thousand man hours, not counting ongoing tool and lexicon development. However, since the latter is inspired by the former, one may ultimately hope that a tidy garden will help to catalogue the jungle.

References

Afonso, Susanna & Eckhard Bick & Renato Haber & Diana Santos, "Floresta Sintá(c)tica, a treebank for Portuguese", in Manuel González & Carmen Paz Suárez Araujo (eds.), Proceedings of LREC 2002, Gran Canaria, pp. 1698-1703

Asmussen, Jørg, "Korpus 2000", in *Korpuslingvistik (NyS30),* Akademisk Forlag/Copenhaguen University, 2001

Bick, Eckhard, *The Parsing System 'Palavras' - Automatic Grammatical Analysis of Portuguese in a CG Framework*. Århus: Aarhus Universitetsforlag, 2000

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

Bick, Eckhard, "Morfosyntaktisk opmærkede corpora for Dansk: Korpus90/2000 og Arboretum", in *9. Møde om Udforskningen af Dansk Sprog, Proceedings,* Århus University, 2002

Bick, Eckhard, "A CG & PSG Hybrid Approach to Automatic Corpus Annotation", in *Proceedings of SProLaC2003 (at Corpus Linguistics 2003, Lancaster)*

Brants, Sabine et.al., "The TIGER Treebank", in *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories, 21./22. Sept. 2002, Sozopol*

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