From CG-2/vislcg to CG-3

New developments in the Constraint Grammar formalism

Eckhard Bick & Tino Didriksen

University of Southern Denmark
VISL Project, ISK
GrammarSoft / GramTrans

Constraint Grammar – what is it?

- (1) a methodological paradigm for handling token-linked information in a contextual, rule-based fashion (Karlsson 1990, 1995)
- (2) a descriptive convention within the dependency camp, supporting a lexical approach with a clear form-function distinction
- reductionist, focus on disambiguation, robust, fast, “non-chomskyan” ..
- (A) a formal language to express context grammars
- (B) a number of specific compiler implementations to support different dialects of this formal language

Adding full **numbered dependency**

- Integrated formalism: FDG (Tapanainen)
- Add-on attachment rules: PALAVRAS, DanGram ...(Bick)
- Constraint Grammar-internal: CG3 (Bick & Didriksen)

```
O <artd> DET M S                    @>N       #1->3
último     ADJ M S                  @>N       #2->3
diagnóstico N M S                   @SUBJ>    #3->9
elaborado V PCP2 M S                @ICL-N<   #4->3
por          PRP                    @<PASS    #5->4
a <artd>  DET F S                   @>N       #6->7
Comissão=Nacional PROP F S          @P<       #7->5
não         ADV                     @ADVL>    #8->9
deixa       V PR 3S                 @FMV      #9->0
dúvidas   N F P                     @<ACC     #10->9
$.                                            #11->0
```

CG rules

- rules add, remove or select morphological, syntactic, semantic or other readings
- rules use context conditions of arbitrary distance and complexity (i.e. other words and tags in the sentence)
- rules are applied in a deterministic and sequential way, so removed information can't be recovered (though I t can be traced). Robust because:
  - rules in batches, usually safe rules first
  - last remaining reading can't be removed
  - will assign readings even to very unconventional language input (“non-chomskyan”)

some simple rule examples

-
  ```
  REMOVE VFIN
      IF   (*-1C VFIN BARRIER CLB OR KC)
  ```

  exploits the uniqueness principle: only one finite verb per clause

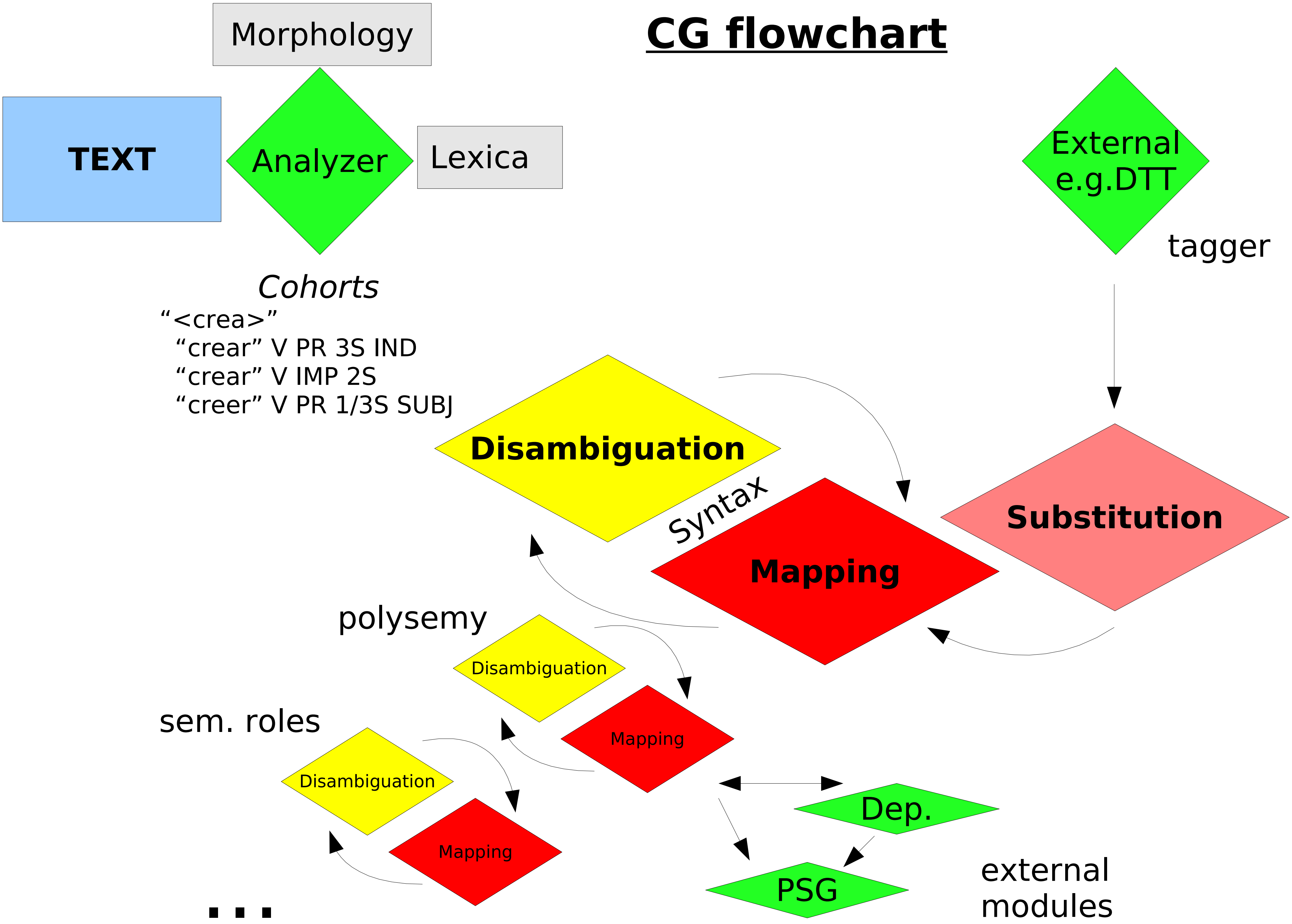
-
  ```
  MAP (@SUBJ> @<SUBJ @<SC) TARGET (PROP)
      IF   (NOT -1 PRP)
  ```

  syntactic potential of proper nouns

-
  ```
  SELECT (@SUBJ>)
      IF   (*-1 >>> OR KS BARRIER NON-PRE-N/ADV)
           (*1 VFIN BARRIER NON-ATTR)
  ```

  clause-initial np's, followed by a finite verb, are likely to be subjects

CG flowchart



```
“<crea>”
  “crear” V PR 3S IND
  “crear” V IMP 2S
  “creer” V PR 1/3S SUBJ
```

coches “coche” <Vground> N M P @ACC §TH #4->2

CG languages (VISL/GS)

Language	Parser	Lexicon	Analyzer	Grammar	Levels
da	DanGram	100.000 lexemes, 40.000 names	Full	8.000 rules	morph., syntax, dep., psg, case roles
pt	PALAVRAS	70.000 lexemes, 15.000 names	Full	7.500 rules	morph., syntax, dep., psg
es	HISPAL	73.000 lexemes	Full	4.500 rules	morph., syntax, dep., psg
en	EngCG	160.000 sem	Full	4.400 rules	morph., syntax, dep., psg
fr	FrAG	57.000 lexemes	DTT + analysis	1.400 rules	morph.-correction, syntax, dep., psg
de	GerGram	25.000 val/sem	(Full)	LS+1.300 rules	morph. (Lingsoft), syntax, dep., psg
eo	EspGram	30.000 lexemes	Full	2.600 rules	morph., syntax, dep.
it	ItaGram	30.600 lexemes	DTT + analysis	1.600 rules	morph., syntax, dep.
se	SveGram	63.000 lexemes	Full	adapted da	morph., syntax, dep.

VISL languages (others)

- Basque
- Catalan
- English ENGCG (CG-1, CG-2, FDG)
- Estonian (local)
- Finnish (CG-1?)
- Irish (Vislcg)
- Norwegian (CG-1)
- Sami (CG-3)
- Swedish (CG1, CG-2?)
- Swahili (Vislcg)

Performance and uses

- Published performance for system-internal evaluations is astonishingly high across languages, with F-scores for mature systems around
- 99% for POS
- 95% for syntactic function (shallow dependency)
- Relative performance in open joint evaluation:
  - e.g. HAREM (Portuguese NER & classification)
- Supports a wide variety of applications
  - Grammar checking (Norwegian, Swedish, Danish ...), e.g. OrdRet (better at weighting suggestions than Word)
  - Corpus annotation (e.g. treebanks) and teaching
  - IR, NER and QA
  - MT and other semantic stuff
  - Anaphora resolution

Some history and comparisons: CG “dialects”

- Common to all CG systems:
  - the context-dependent manipulation of tag-encoded linguistic information at the token level (formally, akin to regular expression substitutions)
  - implemented as REMOVE, SELECT, MAP, ADD, REPLACE, SUBSTITUTE ...
- Differences at the implementational level:
  - programming language: Lisp, C/C++, finite state
  - speed, e.g. cg2 (Tapanainen 1996) = 6 x vislcg (Martin Carlsen)
  - proprietory (cg1, fdg/conexor), academic (cg2), project-bound (Müürisep 2005), commercial (FDG conexor.com), open source (vislcg, cg3)
  - cross compiler compatibility?
[cg1] <-> [cg2 > vislcg > cg3]

Differences at the Grammar level

- Differences in expressive power
  - scope: global context (standard, most systems) vs. local context (Lager's templates, Padró's local rules, Freeling ...)
  - templates, implicit vs. explicit barriers, sets in targets or not, replace (cg2: reading lines) vs. substitute (vislcg: individual tags)
  - topological vs. relational
- Differences of applicational focus
  - focus on disambiguation: classical morphological CG
  - focus on selection: e.g valency instantiation
  - focus on mapping: e.g. grammar checkers, dependency relations
  - focus on substitutions: e.g. morphological feature propagation, correction of probabilistic modules

The CG3 project

- 3+ year project (University of Southern Denmark & GrammarSoft)
- some external or indirect funding (Nordic Council of Ministries, ESF) or external contributions (e.g. Apertium)
- programmer: Tino Didriksen
- design: Eckhard Bick (+ user wish list, PaNoLa, ...)
- open source, but can compile "non-open", commercial binary grammars (e.g. OrdRet)
- goals: implement a wishlist of features accumulated over the years, and do so in an open source environment
- enabling hybridisation of methodologies: CG, dependency grammar, probabilistic methods, ...
- support for specific tasks: MT, spell checking, anaphora ...

The CG3 project -2

- working version downloadable at http://beta.visl.sdu.dk
- compiles on linux, windows, mac
- speed: equals vislcg in spite of the new complex features, faster for mapping rules, but still considerably slower than Tapanainen's cg2 (working on it).
- documentation available online
- sandbox for designing small grammars on top of existing parsers: The cg lab

A rules file 1 (definitions)

```
DELIMITERS = "<.> "<!>" "<?>" ; # sentence window

SETS

LIST NOMINAL = N PROP ADJ PCP ; # nominals, i.e. potentieal nominal heads
LIST PRE-N = DET ADJ PCP ; # prenominals
LIST P = P S/P ; # plural
SET PRE-N-P = PRE-N + P ; # plural prenominals, equivalent to (DET P) (DET S/P) (ADJ P) (ADJ S/P) (PCP P) (PCP S/P)
LIST CLB = "<,>" KS (ADV <rel>) (ADV <interr>) ; # clause boundaries
LIST ALL = N PROP ADJ DET PERS SPEC ADV V PRP KS KC IN ; # all word classes
LIST V-SPEAK = "dizer" "falar" "propor" ; # speech verbs
LIST @MV = @FMV @IMV ; # main verbs
```

A rules file 2
(morphological disambiguation)

CONSTRAINTS

REMOVE (N S) IF (-1C PRE-N-P) ; # remove a singular noun reading if there is a safe plural prenominal directly to the left.

REMOVE NOMINAL IF (NOT 0 P) (-1C (DET) + P) ; # remove a nominal if it isn't plural but preceded by a safe plural determiner.

REMOVE (VFIN) IF (*1 VFIN BARRIER CLB OR (KC) LINK *1 VFIN BARRIER CLB OR (KC)) ; # remove a finite verb reading if there are to more finite verbs to the right none of them barred by a clause boundary (CLB) and coordinating conjunction (KC).

A rules file 3
(syntactic disambiguation)

MAPPINGS

MAP (@SUBJ> @ACC>) TARGET (PROP) IF (*1C VFIN BARRIER ALL - (ADV)) (NOT -1 PROP OR PRP) (NOT *-1 VFIN) ; # a proper noun can be either forward subject or forward direct object, if there follows a finite verb to the right with nothing but adverbs in between, provided there is no proper noun or preposition directly to the left, and a finite verb anywhere to the left.

CONSTRAINTS

REMOVE (@SUBJ>) IF (*1 @MV BARRIER CLB LINK *1C @<SUBJ BARRIER @MV) ; # remove a forward subject if there is a safe backward subject to the right with only one main verb in between

CG Contexts

- **Context conditon**: word form “<...>”, base form “....”, tag A-Z, <[a-z]> @[A-Z], combinations ...
- direction: + (right), - (left)
- Position marker:
  - 0 self
  - local right: 1, 2, 3 ..., local left: -1, -2, -3, ...
- Globality
  - * continue until match is found
  - ** continue also across context match to fulfil further (linked) conditions
  - 0* nearest neighbour: search in both directions
- Careful: C, e.g. *1C (only unambiguous readings)

CG contexts 2

- **NOT**: conditions can be negated
  - (NOT *1 VFIN)
- contexts can be **LINK**ed
  - (*1C xxx LINK 0 yyy LINK *1 zzz)
- searches can have a **BARRIER**
  - (*1 N BARRIER VFIN)
- contexts can be ANDed
  - IF (0 xxx) (*1 yyy) (NOT *-1 zzz)
- contexts can be negated as a whole
  - (NEGATE *1 ART LINK 1 ADJ LINK 1 N)

Mapping (MAP, ADD)

```
MAP (@SUBJ) TARGET (N) IF (NOT *-1 NON-PRE-N)
MAP (@SUBJ) (N) (NOT *-1 NON-PRE-N)
```

- Usually as a special section (MAPPING or BEFORE-SECTIONS), but in cg3 allowed anywhere
- Strictly ordered
- Both MAP and ADD can be used to add tags, but:
  - MAP "closes" a line for further mapping (but not SUBSTITUTE!) even if the mapped tag(s) does not contain the flagged prefix (default @)
  - ADD maps, but allows further mapping
- MAPed tags can be "seen" by later mapping rules, even in the same section

Substitutions (new in vislcg)

```
SUBSTITUTE (UTR) (NEU) TARGET (@<SC)
IF (*-1C @SUBJ> + NEU BARRIER CLB)
```

- Replaces a tag or tag chain with another, useful for:
  - correcting input from other modules, e.g. stochastic taggers
  - correcting lower level CG once higher lever information is available
  - spell or grammar checkers
- Usually as a special section (CORRECTIONS or BEFORE-SECTIONS), but in cg3 allowed anywhere
- 'TARGET' and 'IF' are optional
- Strictly ordered
- SUBSTITUTE does not "close" a line for mapping
- SUBSTITUTEd tags can be "seen" by later SUBSTITUTE or Mapping rules, even in the same section

REPLACE

```
REPLACE (V IMPF AKT) TARGET ("<.*ede>"r)
IF (-1 (PERS NOM) (1 ikke)
```

- REPLACE replaces **all** non-baseform tags with a new tag chain, hence it has one argument less than SUBSTITUTE; used for:
  - corrections where only the baseform is o.k., e.g. verbal tense errors
- REPLACE works like a mapping operator, closing the line for further mapping
- it is less versatile than SUBSTITUTE, but backward compatible with CG2

New CG-3 features

- a) rules and window management
- b) tag operations, positions and contexts
- c) grammar management (flags, call options)
- d) the **big** additions: subsuming - on top of CG's native topological/field-based approach - all other **descriptive** syntactic traditions:
  - 1. Dependency Grammar: c, p, s
  - 2. Constituent Grammar: templates
  - 3. Unification Grammar: $$SET
- e) subsuming competing **methodological** techniques, on top of the native tag manipulation techniques
  - 1. Integrating regular expressions
  - 2. Integrating statistical information

Individual and soft DELIMITers

[wordform] DELIMIT <target> [contextual_tests]

- an on-the-fly sentence (disambiguation window) delimiter
- for cases where the delimiter has to be decided from context

SOFT-DELIMITERS = "<$;>" ;

- used as delimiter if a disambiguation window approaches the soft-limit for window size (default 300), before hard window breaking (default 500)

Rule Management

- Any type of ryle anywhere in the grammar
- Any type of set definition anywhere in the grammar
- Anything changed by a rule can be seen by all subsequent rules, including ADD, MAP and SUBSTITUTE
- Any tag anywere in a cohort reading (order independent)

BEFORE-SECTIONS
AFTER-SECTIONS

- Run only once (cp. VISLCG's MAPPING / CORRECTIONS)
- Especially for
  - adding ambiguity, e.g. Syntactic, semantic roles etc.
  - Post-processing errors from previous modules

Named Rules

- REMOVE:rule_name <target> [contextual tests]
- MAP:rule_name <tags> <target> [contextual tests]
- rule names need not be unique

Rule application order

```
ForEach (Window)
    ForEach (Rule)
        ForEach (Cohort)
            ApplyRule
```

- each rule on all cohorts (VISLCG: each cohort all rules)
- more predictable results, since rule order is not text dependent
- less need for order-forcing via sections, so sections can be used for their primary purpose, task modularity and heuristicity

Tag operations

REMOVE @<ACC (0 @<SUBJ LINK 0 (<H.*>r) OR (".*ist"r)
-> discard object in favor of subjects if the token is +HUM

- Tag Inversion (!-prefix)
  - e.g. !GEN matches every tag **but** GEN
  - make sense only in combination: (N !GEN) ... for German the same as the set union of (N NOM) (N ACC) (N DAT)
- Fail Fast Tag (^-prefix)
  - prevents a set from matching - regardless of other tag-matchings in the set - if the fail-fast tag is present, too, in the relevant cohort line
  - e.g. SET PRE-N = ART DET ADJ ^<nphead> ^@P< ;

NEGATE

```
(NEGATE *1 (AUX) LINK 1 (@AUX<)) ;
(NEGATE *-1 N LINK -1 DEF) ;
```

- implements aspects of the TEMPLATE idea (being able to refer to - and to negate - chunks of internally linked tokens
- will invert the result of the entire LINK'ed chain that follows
- whereas NOT will only invert the result of the immediately following test
- VISLCG emulated NEGATE with parenthesis-initial NOT

CBARRIER

(**1 N CBARRIER VFIN) ;

- like BARRIER, but only if unambiguous
- i.e. less strict than BARRIER

Nearest Neighbour

(NOT 0* VFIN) ; -> no other finite verb candidates
(0* VFIN BARRIER CLB) ; -> presence of a verb in the same clause

- Magic offset 0, scans for nearest neighbour in **both** directions (-1 -> 1 -> -2 > 2 -> ... -n -> n)
- especially useful to collapse two contexts (1. example) or two rules (2. example) into one

Spanning Window Boundaries

(*1> ("http.*")) ; -> find urls
(*-1< UTR + @SUBJ BARRIER CLB) ;
-> pronoun gender resolution
(*-1W (<Vground>) ; -> text about cars

- Span Left (<): allows to span left boundaries
- Span Right (>): allows to span right boundaries
- Span Both (W): allows to span boundaries right and left
- Default ± windows, otherwise
  - command line flag: --num-windows
- Always allowing **all** spans to cross boundaries
  - command line flag: --always--span

String tag modifiers

REMOVE @<ACC (0 @<SUBJ LINK 0 (<H.*>r) OR (".*ist"r)
-> discard object in favor of subjects if the token is +HUM

- applies to (a) token tags, (b) base form (lexeme) tags, (c) <...> secondary tags
- literal string modifier 'i' = case insensitive
  - "<Wordform>"i, "baseform"i, <secondary>i
- literal string modifier 'r' = regular expression
  - ".*ize"r --- a certain group of transitive verbs in English
  - <[HA].*>r --- semantic prototype tag for *animates, i.e*. *humans* (e.g. <*Hprof*>) and *animals* (e.g. <*Aorn>)*

Creating Dependencies

```
SETPARENT (@>N) (0 (ART DET))  TO (*1 (N)) ;
SETPARENT (@P<) TO (*-1 (PRP)) ;
SETPARENT (@FS-N<) TO (*-1 N LINK NOT p SELF)
```

- create dependencies on the fly
- change existing dependencies
- circularity
  - a rule won't be applied if it introduces circularity
  - however, if there IS circularity further up in the ancestor chain from a previous module, then it will be accepted

Using Dependencies

SELECT (%hum) (0 @SUBJ) (p <Vcog>)

-> assign +HUM to subjects of cognitive verbs

SELECT (@ACC) (NOT s @ACC)

-> uniqueness principle

(*-1 N LINK c DEF)

-> definite np recognized through dependent

ADD (§AG) TARGET @SUBJ (p V-HUM LINK c @ACC LINK 0 N-NON-HUM) ;

- accepts input from other programs in cg-format:
  - ... #n->m
- in a rule, dep-relations (letters) replace positions (numbers), NOT, * and C behave "correspondingly"
  - Parent/Mother (p)
  - Child/Daughter (c)
  - Sibling/Sister (s)

labelled arcs for other purposes

- instead of the default dependency arcs, other relations can be defined:
- SETRELATION (referent) TARGET (<rel>) TO (*-1 N) ;
  (Set a ***”referent”*** *relaton from a relative pronoun to a noun occurring earlier in the sentence.)*
- leads to: ID:n R:identity:m
  - n: arc base (here pronoun) word number
  - identity: relation name introduced by R
  - m: arc head (here the referent noun) word number
- REMRELATION – removes one direction of a relation
  - REMRELATION (name) targetset () TO ()
- SETRELATIONS and REMRELATIONS simultaneously handle 2 names for the two directions of a relation

Parenthesis enclosures

- problem: text within parentheses often has independent syntax, with only a single link to the outside sentence
- problem: CG rules have difficulties in scanning across parentheses, and may wrongly interact with parenthesis content
- solution: Normally, parenthesis content may attach (left) out of the parenthesis, while outside consituents don't attach to inside tokens. Therefore, it helps syntactic cohesion to ignore parentheses in a first pass. With more than one parenthesis, or nested parentheses, this is best done in a layered, iterated fashion.

Parenthesis enclosures 2

- PARENTHESES = ("<$(>" "<$)>") ("<$[>" "<$]>") ("<${>" "<$}>") ("<$«>" "<$»>") ;
- _LEFT_ and _RIGHT_ are magic tags (and sets!) for the left and right parenthesis wordforms of the active enclosure
- MAP (@SUBJ>) TARGET (N NOM) (*1C VFIN BARRIER N OR _RIGHT_)
- Contextual positions L and R, referable only from within a parenthesis (i.e. the active enclosure)
- ADD (@acc) TARGET N/PROP/PRON-NOM
      (-1C N/PROP/PRON + NOM) (*-2 NON-PRE-N/ADV LINK NOT 0 PRP) (*1C VFIN BARRIER NON-ADV) (NOT L ("<[>")) ; # la politikistoj tion volas šanĝi, **not:** [san majkrosistemz]

Probabilistic / statistical tags

REMOVE (<Conf<5>)

-> confidence threshold 5 (%)

REMOVE (<Noun<=10>) (NOT -1 PRE-N)

-> context dependent frequency threshold 10%

- expects input tags with colon-separated numerical values:
  - <Conf:80> (confidence values, e.g. for suggestions of a spell checker
  - <Verb:70> (e.g. monogram PoS-likelihod for a given token)
- all positive integer values are possible, a cohort sum of 100% for confidence is an optional convention, as is the use of relative frequencies

"Magic" sets

- _S_DELIMITERS_ is the standard set of delimiters
- _S_SOFT_DELIMITERS_ refers to the set of soft delimiters
- _LEFT_, _RIGHT_ refer to the active parentheses as tags
- _L_, _R_ refer to their positions
- (*) is the all-set, useful for:
  - negative NON- sets: NON-PRE-N = (*) - PRE-N
  - referencing a position rather than a tag:
    (1 (*) LINK *-1 VFIN BARRIER NON-V) ... finds the heading finite verb of a verb chain even if the target is itself the VFIN

TEMPLATE

- labels for complex contexts conditions, which – once defined – can then be used by many different rules, or even in other templates.

- TEMPLATE np = (ART LINK 1 N) OR (ART LINK 1 ADJ LINK 1 N)

- referenced as **(*1 VFIN LINK *1 (T:np))**.

- Currently, templates still need obligatory positions, so that the above would have to be written as

- TEMPLATE np = (0 ART LINK 1 N) OR (0 ART LINK 1 ADJ LINK 1 N) and then referenced as **(*1 VFIN LINK **1 (*) LINK (T:np))**

- Optimally, the use of positions inside the template should be optional, although that would imply different treatment of templates in rules. It would thus constitute an error to use a positioned template with a position, or a non-positioned one without a position. To make the diffence clearer, we could use T:/TEMPLATE (non-positioned) and PT:/PTEMPLATE (positioned).

Unification

- A way of using tag variables in rule contexts
- LIST CASE = NOM GEN DAT ACC ;
- SELECT $$CASE (1 KC) (2C $$CASE) ;
- SELECT $$NUMBER + ADJ (-1C $$NUMBER + ART/DET)
- unification of reg.ex. strings:
  - allowing $$sets with "....”r base forms and <....>r secondary tags
  - allows only .*, not more complex expression
- SETRELATION (anaphor) TARGET @SUBJ> + $$PROSEM TO (W*-1 @SUBJ> + $$PROSEM BARRIER $$PROSEM) (*1 VFIN LINK 0 @FS-STA LINK NOT 0 <cjt>) ;
  - using: LIST PROSEM = <H.*>r <A.*>r <L.*>r <sem.*>r ;

Dynamic point of origin

- -o “a la “ cg-1, don't cross position 0
- Dynamic switching on of this feature: O

  treats last context as “origin”, if used with linked contexts
- Dynamic switching off of this feature: o

  valid for all further contexts in this rule
- Use of O/o in connection with dependency contexts:
- SETPARENT (@FS-N<) (O*-1 N LINK p (*))

  on-the-fly circularity check against relative clauses linking to their own subject

Regressive linking

- Problem: How to check a negative context or a parent, then continue linking from the original spot
- Solution so far (and only for non-dependency):
  .... LINK *1 X LINK NOT 1 Y LINK -1 ALL LINK ....
- New, general solution:
  - X = dynamic origin (“0”) - if omitted, it defaults to rule target
  - x = dynamic regressive linking
- MAP (§AG) TARGET @SUBJ>
  (**X**c @FS-N< LINK **S**c @ICL-AUX< LINK 0 V-HUM
  LINK **x**c <rel> + @SUBJ>
- equivalent to the longer:
  MAP (§AG) TARGET @SUBJ>
  (c @FS-N< LINK -1 ALL LINK *1 @MV LINK 0 V-HUM)
  (c @FS-N< LINK c <rel> + @SUBJ>

Input Stream Commands

CGCMD:

- an input stream can have some control over the programme pipe, using 1-line commands as part of the input:
- CGCMD:FLUSH ... cuts execution, breaks all windows
  - useful for text type changes, e.g. after head lines etc.
- CGCMD:EXIT ... stops execution (use after FLUSH)
  - useful for trial runs where only the first part of a large corpus is to be analysed
- CGCMD:IGNORE ... causes input to be ignored until RESUME (use after FLUSH)
  - useful to skip binary data, lists, poems, text in the wrong language etc.
- CGCMD:RESUME ... resumes analysis after an IGNORE

Runtime options 1

- --grammar, -g ... the grammar file to use for the run
- --vislcg-compat, -p ... compatible with older VISLCG
- --trace ... adds debug output
- --prefix ... sets mapping prefix, default @
- --sections ... sections to run, default all
    - -- sections 6
    - -- sections 2-5,10-12
- --single-run ... only runs each section once.

Runtime options

- --no-mappings ... disables MAP, ADD and REPLACE rules.
- --no-corrections ... disables SUBSTITUTE and APPEND
- --num-windows ... window buffer span, default ±2
- --always-span ... always scan across window boundaries.
- --soft-limit ... token limit for SOFT-DELIMITERS (def. 300)
- --hard-limit ... token limit for hard window breaks (500)
- --o .... target position (origin) will halt a context scans
    - this can be achieved locally by adding upper case 'O' to a context position (which then defines its contextual “parent” as origin
    - if set, the origin-block can be undone by adding a lower case 'o' to a contextual position
    - REMOVE (origin) IF (*-1O (left) LINK *1 (middle) LINK *1o (right)

Input/Output options

- -O or --stdout ... file to print output to instead of stdout.
- -I or --stdin ... file to read input from instead of stdin.
- -E or --stderr ... file to print errors to instead of stderr.
- -C or --codepage-all ... codepage to use for grammar, input, and output streams. Defaults to ISO-8859-1.
- --codepage-grammar ... odepage to use for grammar
- --codepage-input ... codepage to use for input
- --codepage-output ... codepage to use for output
- -L or --locale-all ... locale to use for grammar, input, and output streams. Defaults to en_US_POSIX.
- --locale-grammar ... locale to use for grammar
- --locale-input ... locale to use for input
- --locale-output ... locale to use for output

Optimization by **rule ordering** and/or **context ordering**: Speed vs. heuristicity

- Speed: We have experimented with a cost-benefit analysis for CG rules, and achieved differences in speed around 30% by rules with a high benefit/cost ratio first
  - disambiguation gain: SELECT > REMOVE, frequent rules first (also: avoid "ghost" checking rules), "heavy" sets first, POS targets vs. word targets, target frequency (not used)
  - processing cost:
    - rules length (in number of contexts)
    - global > local contexts
    - NOT/C > simple check

- Heuristicity/safety ordering (inspired by T. Lager):
  - with gold corpus: assign each rule an error likelihood: how often did it remove a CORRECT tag, reiterative reordered runs
  - overriding the last-tag-exception: how often *would* a rule have removed a CORRECT tag if it had been allowed
- Removing unused rules in a corpus-dependent fashion
  - vislcg3 call with a special flag and a text corpus
  - outputs a “lean” grammar, that will run faster
  - safe: unused rules at bottom
  - unsafe: unused rules removed
  - could be used for domain optimization or for grammars by different authors for the same languages, where the second grammar is tuned so as to address only issues the first grammar hasn't handled

- speed-up by removing unused set definitions
  - now implemented in grammar-compilation internally
  - also possible for the rules file proper: clean-cg
- speed-up of grammar start-up (implemented):
  - compile and run binary grammars – also useful commercially
  - vislcg3 –grammar *rules* --grammar-bin *binary*
  - cg3-autobin.pl – creates binary grammar first time and keeps using it from until changes in the original rules files are noted

Evaluation

```
###############################################
##########################
#
# use: eval_cg file1 file2
#
# compares two cg files, either a gold file with a test file, or simply two
different runs on the same input.
#
# (1) Input format can either be niceline or cohort format
#   (a) niceline: word     [base] <...> ... POS MORF ... @FUNC ...
#   (b) cohort: "<word>"
#                  "base" <...> ... POS MORF ... @FUNC
#                  "base" <...> ... POS MORF ... @FUNC
#                  ...
# However, only for syntatic tags (@) is ambiguity allowed - cohort format will
be truncated to one morphological line per cohort (the first). Output format will
be "niceline".
#
# (2) rewrites testfile with difference markers (3a-d), followed by file1 line
number and file1 tag (in parentheses). For syntax, the parenthesis will also
contain a hit-out-of count, e.g. 1/2.
#
```

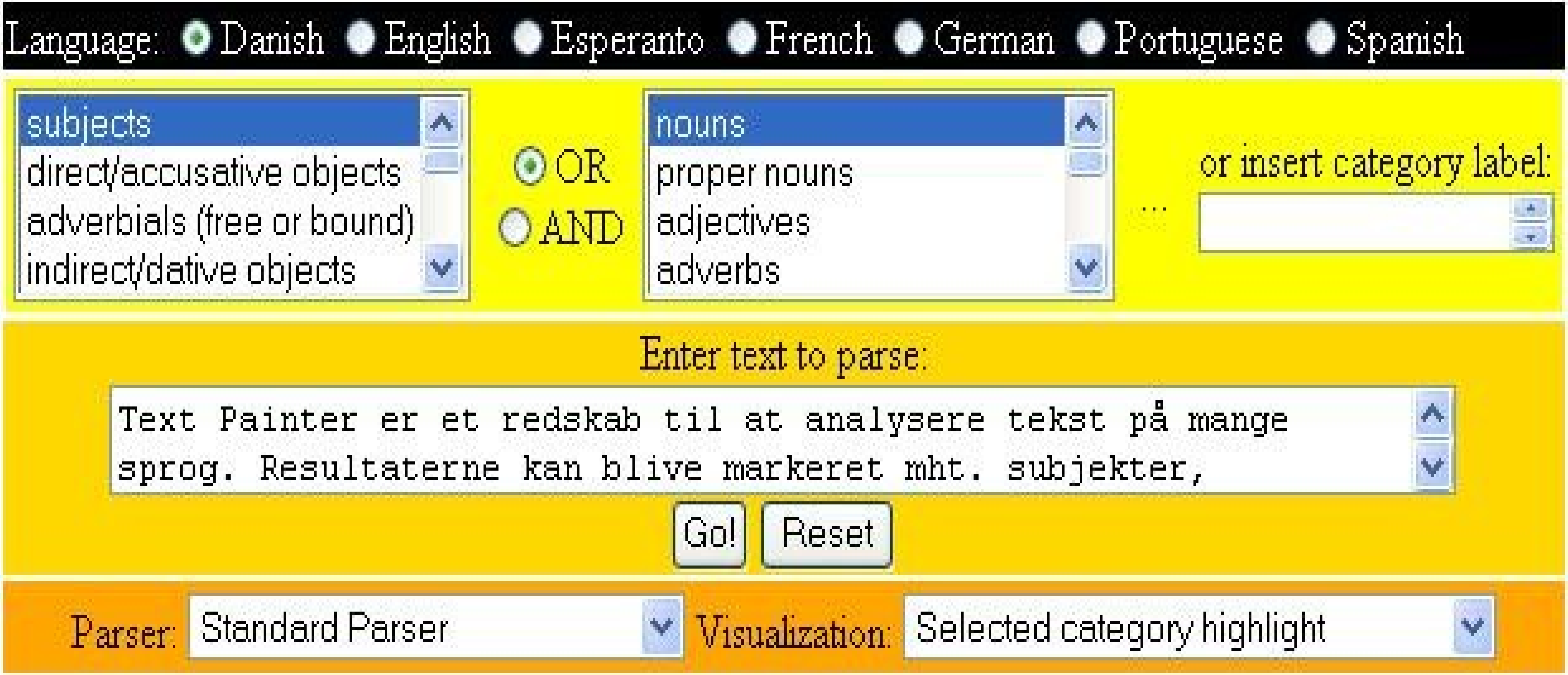
```
###############################################
##########################
#
#
# (3) At the end of the rewritten file2, eval_cg will output file difference as an
evaluation metrics, providing recall, precision and F-score for
#       (a) base form *B
#       (b) part of speech *P
#       (c) morphology *M
#       (d) syntax *S
#
# (4) The program has no special alignment needs - it will tolerate some
tokenisation difference (e.g. regarding polylexicals). Tokenisation mismatches
will be marked in the rewritten file2 as *T_missing, *T_extra and *T_mismatch
followed by the corresponding line number in file1, in the case of many to
many mismatches also with an unti-n/m indicator, where n and m are the
respective line numbers of the point where alignment was reestablished
#
###############################################
##########################
```

still missing from the wish list

- handling of data- and rule-driven meta-variables
  - domain, text type, language recognition
  - SETVARIABLE, REMVARIABLE
- plus many loose ideas ....
- Now's the time for adding more :)

Teaching: e.g. VISL tools

1. TextPainter



Language: ● Danish ○ English ○ Esperanto ○ French ○ German ○ Portuguese ○ Spanish

subjects ● OR ○ AND nouns

Enter text to parse:

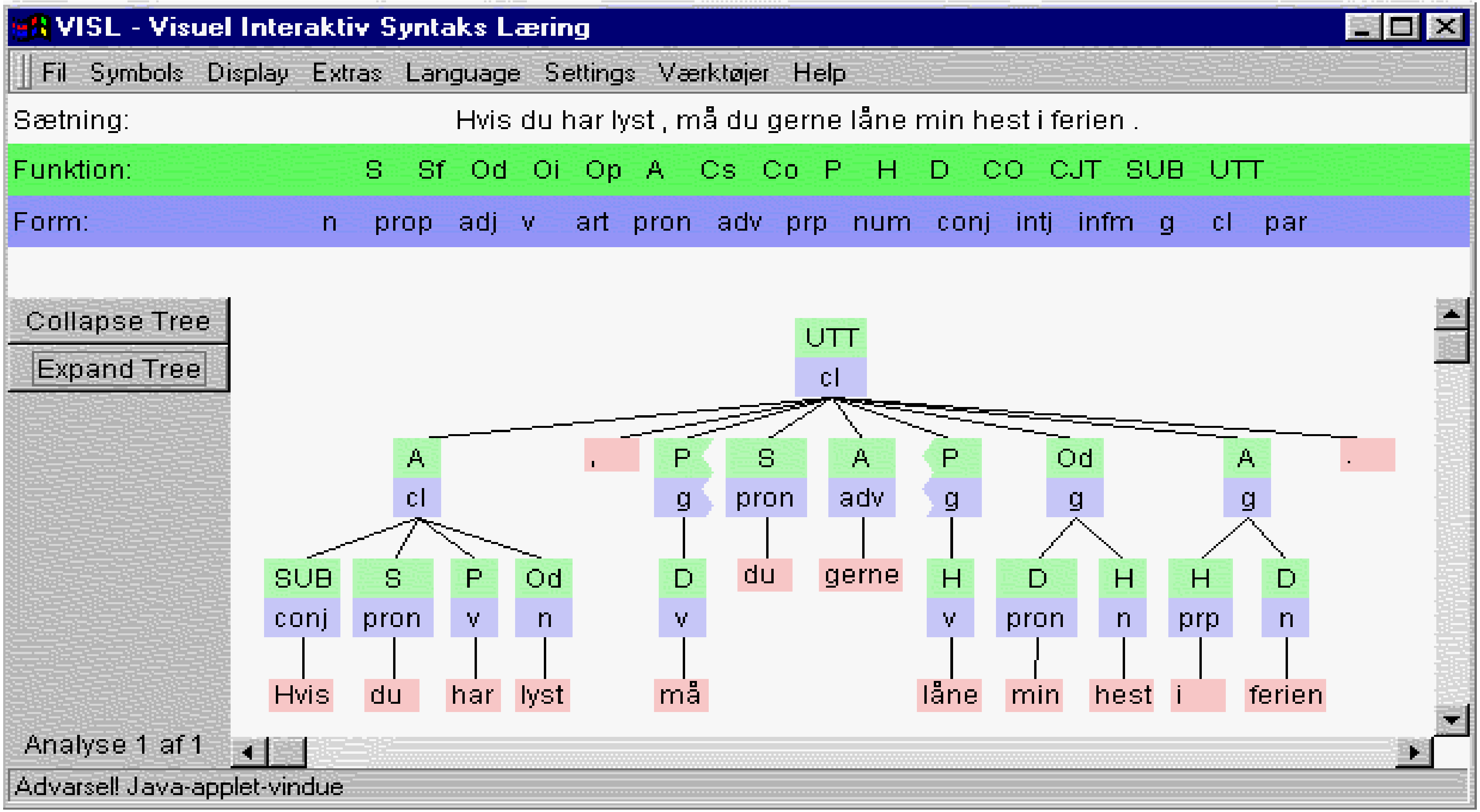
Text Painter er et redskab til at analysere tekst på mange sprog. Resultaterne kan blive markeret mht. subjekter,

Parser: Standard Parser

Visualization: Selected category highlight

direct/accusative objects
adverbials (free or bound)
indirect/dative objects
proper nouns
adjectives
adverbs
...
or insert category label:
Go!
Reset

2. Interactive syntactic tree building



VISL - Visuel Interaktiv Syntaks Læring

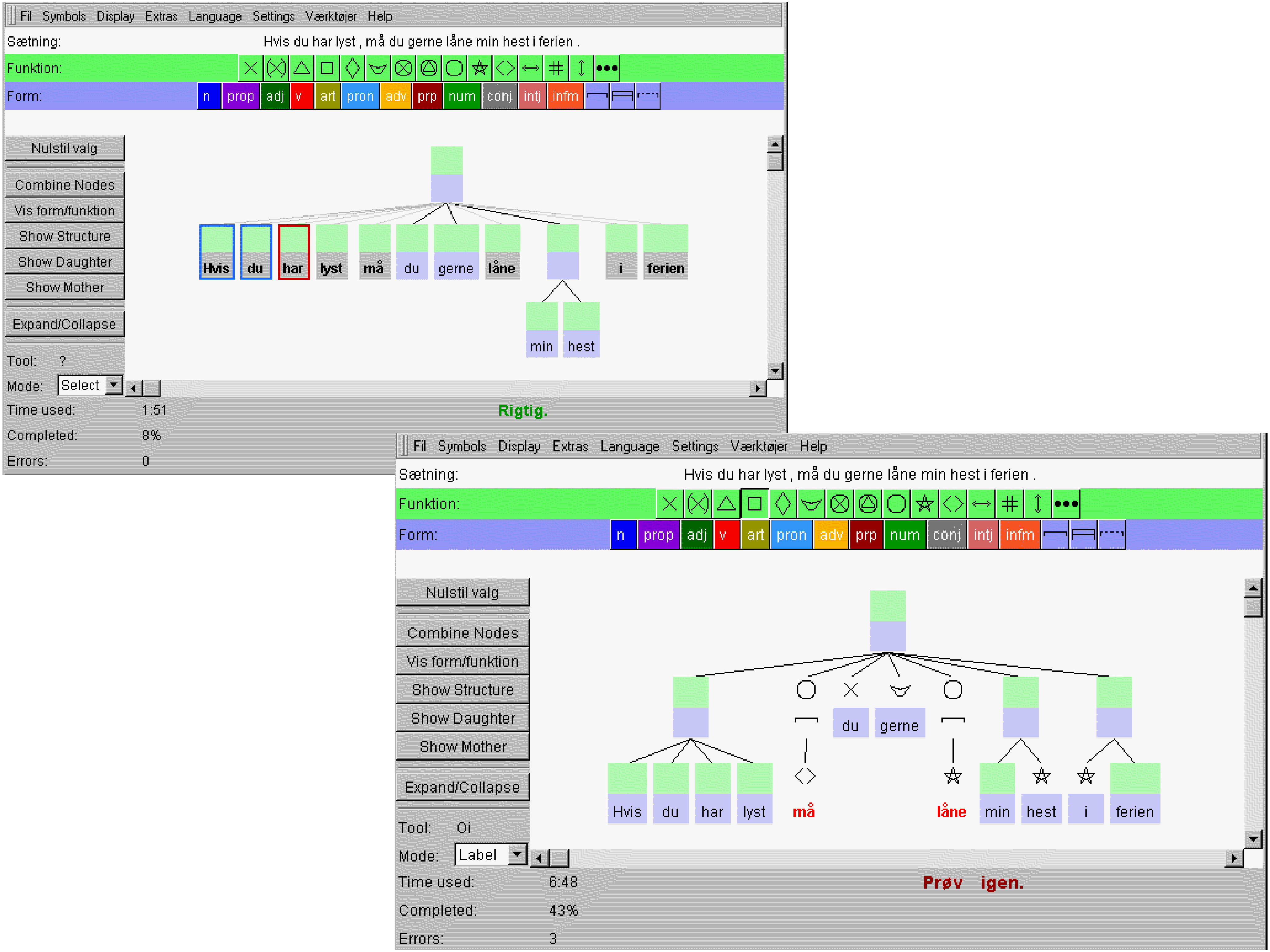
Sætning: Hvis du har lyst , må du gerne låne min hest i ferien .

Funktion: S Sf Od Oi Op A Cs Co P H D CO CJT SUB UTT

Form: n prop adj v art pron adv prp num conj intj infm g cl par

Analyse 1 af 1

Advarsel! Java-applet-vindue



Sætning: Hvis du har lyst , må du gerne låne min hest i ferien .

Tool: ?

Mode: Select

Time used: 1:51

Completed: 8%

Errors: 0

Rigtig.

Sætning: Hvis du har lyst , må du gerne låne min hest i ferien .

Tool: Oi

Mode: Label

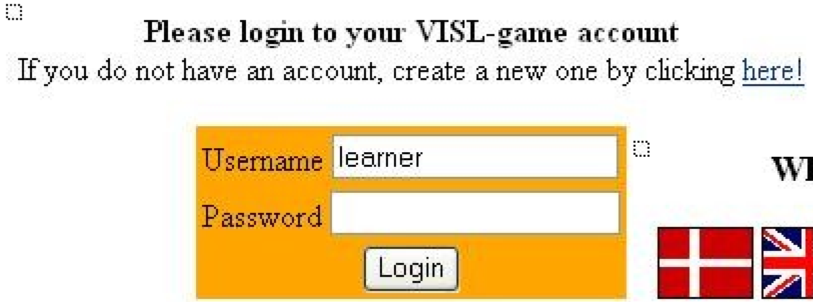
Time used: 6:48

Completed: 43%

Errors: 3

Prøv igen.

3. KillerFiller:
Automatic corpus-based slot-filler exercises

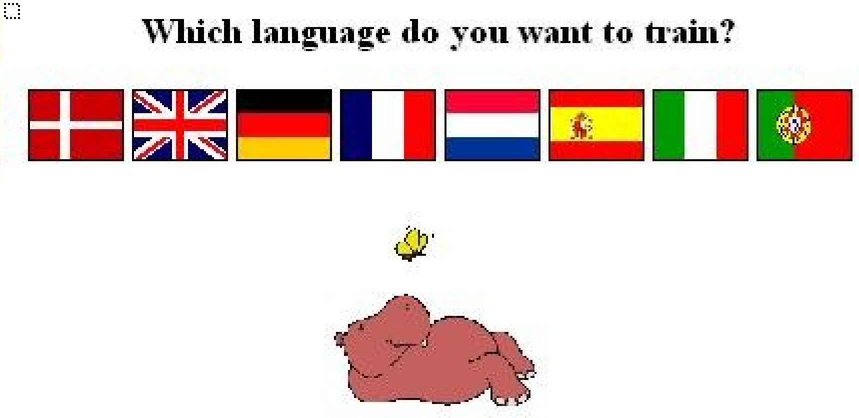


Please login to your VISL-game account

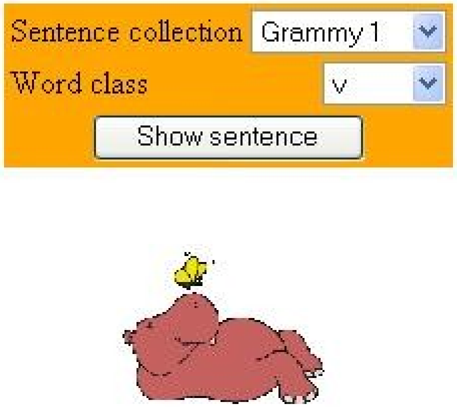
If you do not have an account, create a new one by clicking here!

Username: learner

Password:

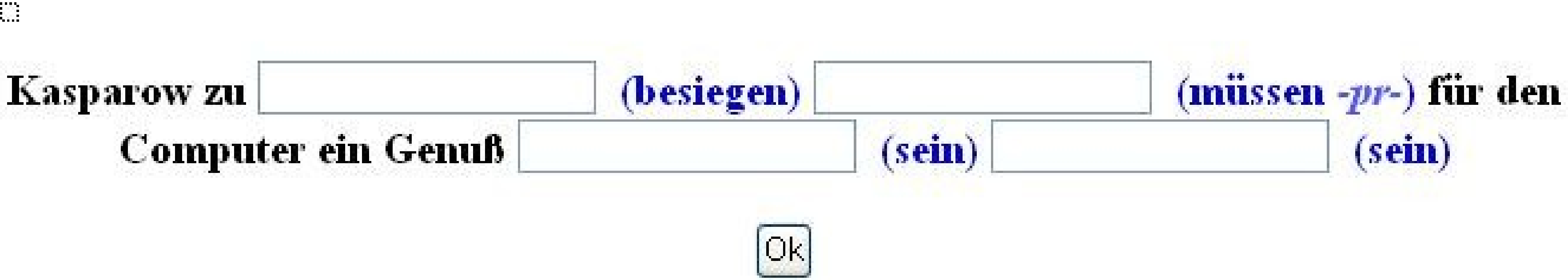


Which language do you want to train?



Sentence collection: Grammy 1

Word class: v



Kasparow zu ______ (besiegen) ______ (müssen *-pr-*) für den Computer ein Genuß ______ (sein) ______ (sein)

Question-answering systems (EPIA2003): better question-typing

QUE:fcl

=ADVL:adv('quando' <interr>) **Quando**
[=FOC:adv('é=que')   **foi=que**]

=P:v-fin('nascer' PS 3S IND)   **nasceu**

=SUBJ:prop('Balladur' <hum> M/F S) **Balladur**
=?

From this information the system fills in a number of variables:

question pattern (Atemp-PS)

interrogative constituent: Q-word ("quando"), Q-function ("ADVL")

predicator information: P-base ("nascer"), P-tense ("PS")

search point constituent: S-string ("Balladur"), S-function ("SUBJ"), S-head ("Balladur")

Hit sentence: *Balladur nasceu em Esmirna (Turquia), em 1929, e formou-se na Escola Nacional de Administração, de onde saiu a elite da função pública francesa.*

```
STA:cu
CJT:fcl
=SUBJ:prop('Balladur' <hum> M/F S)     Balladur
=P:v-fin('nascer' PS 3S IND)    nasceu
=ADVL:pp
==H:prp('em')    em
==P<:np
===H:prop('Esmirna' <civ> M S)  Esmirna
===(
====N<PRED:prop('Turquia' <civ> F S)     Turquia
===)
=,
=ADVL:pp
==H:prp('em')    em
==P<:num('1929' <date> <card> M S)   1929
=,
```

syntactic analysis permits to extract more implicit knowledge, e.g. ISA relations from appositions, predicatives and relative clauses:

1. Onde é/fica Smirna

2. Quando Rakhmonov derrubou o governo?

A guerra civil no Tadjiquistão, que fez mais de 50 mortos, começou em 1992, quando as ***forças*** *do neo-comunista Rakhmonov derrubaram o governo dos islamistas ...*

```
SUBJ:np
=H:n(<HH>)   forças
=N<:pp
==H:prp   de
==P<:np
===>N:art     o
===H:n(<hum>)  neo-comunista
===N<:prop(<hum>)      Rakhmonov
```

- (a) name-np-flattening: post-nominal or appositive names are substituted for the np, whose head they are dependent of: O neo-comunista Rakhmonov -> Rakhmonov
- (b) toto-pro-pars: semantic heads of postnominal de-pp's are substituted for the pp: as forças de Rakhmonov -> as forças Rakhmonov

Apply a - b - a

Machine Translation: Polysemy resolution, Lexical transfer

udsætte_V

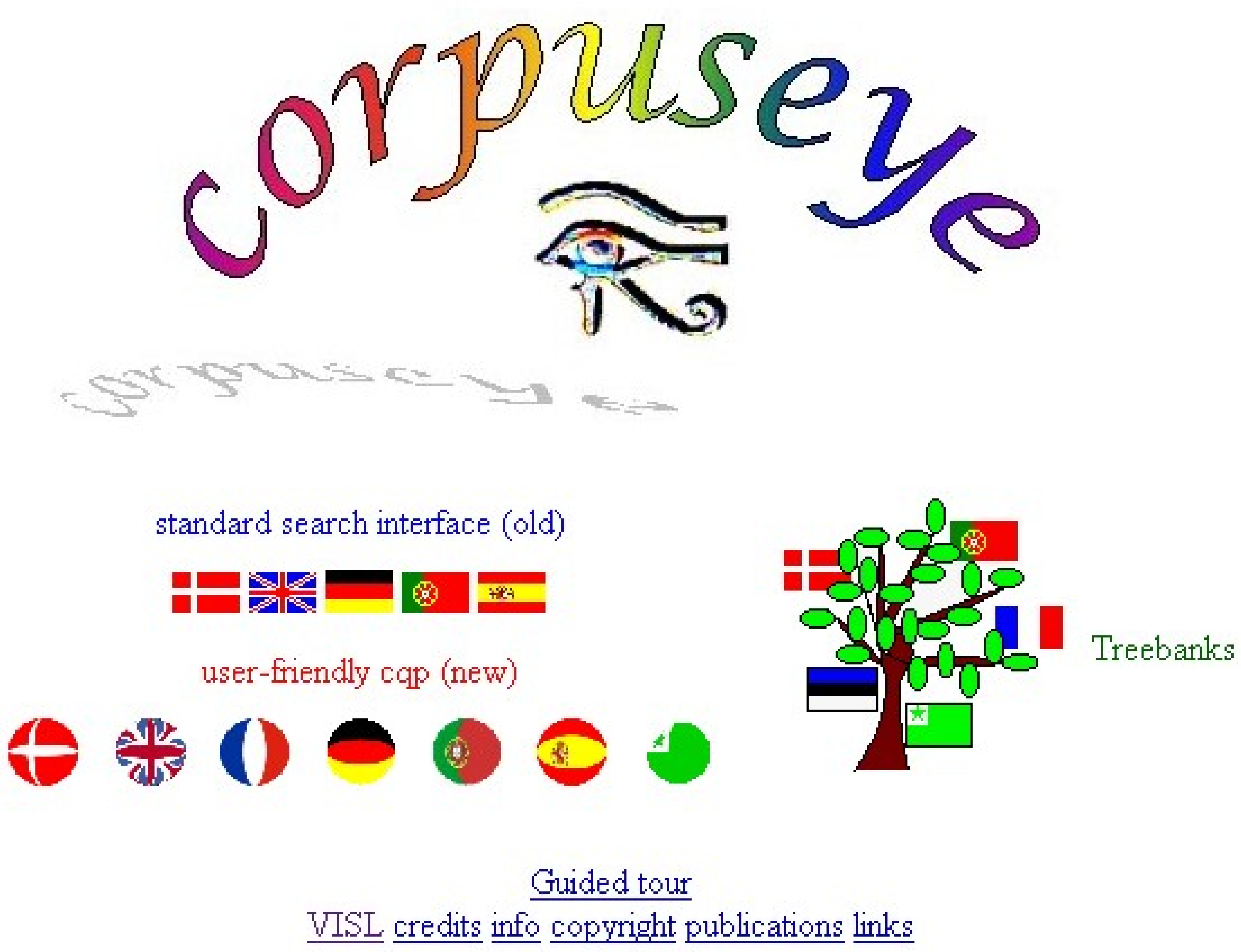
- {opsætte} :postpone, :put=off;
- D=(@ACC) D=("for")_to :expose;
- D=(<prize> @ACC) :offer;
- S=(INF) M=(<quant>) :criticize;
- D=("vagt")_sentry :post;
- D=(<Vwater> @ACC) :put=out;
- D=("lejer" @ACC) :evict;

Machine Translation: Movement rules, Structural transfer

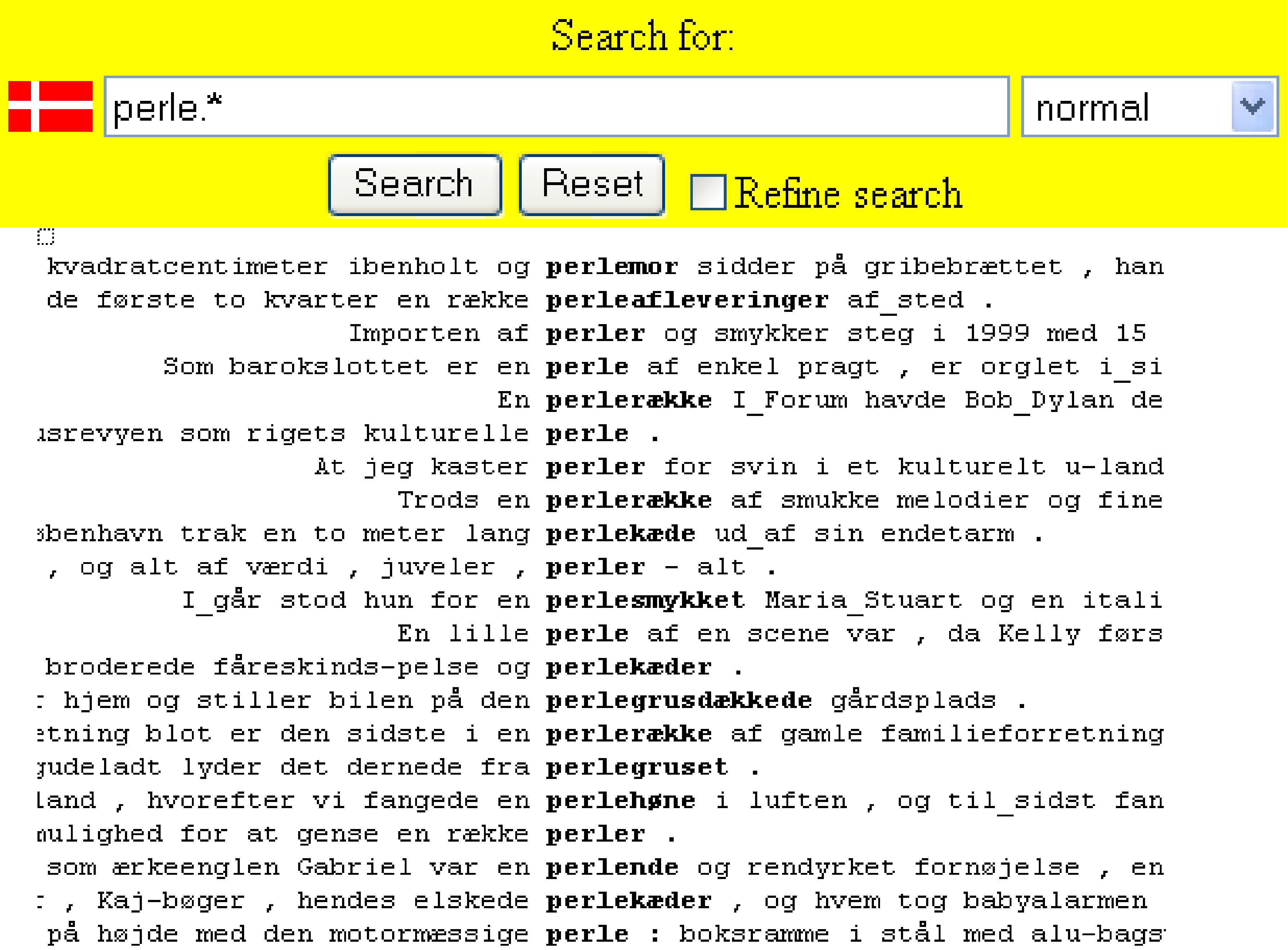
- ***I dag @ADVL*** *drikker @FMV vi @SUBJ vin @ACC* - ***Today*** *we drink wine*

-
```
(@ADVL|@ACC|@FS-ADVL|@FS-ACC|@>>P), I_dag
         w(@FMV|@FAUX|@FS-[^Q]+),drikker
             w(@ICL-AUX<)?,
                 w(@ADVL)?,
                     (@SUBJ|@F-SUBJ|@S-SUBJ)vi
    -> 1, 5, 2, 3, 4
```

A user-friendly Corpus interface


corpuseye
standard search interface (old)
user-friendly cqp (new)
Treebanks
Guided tour
VISL credits info copyright publications links

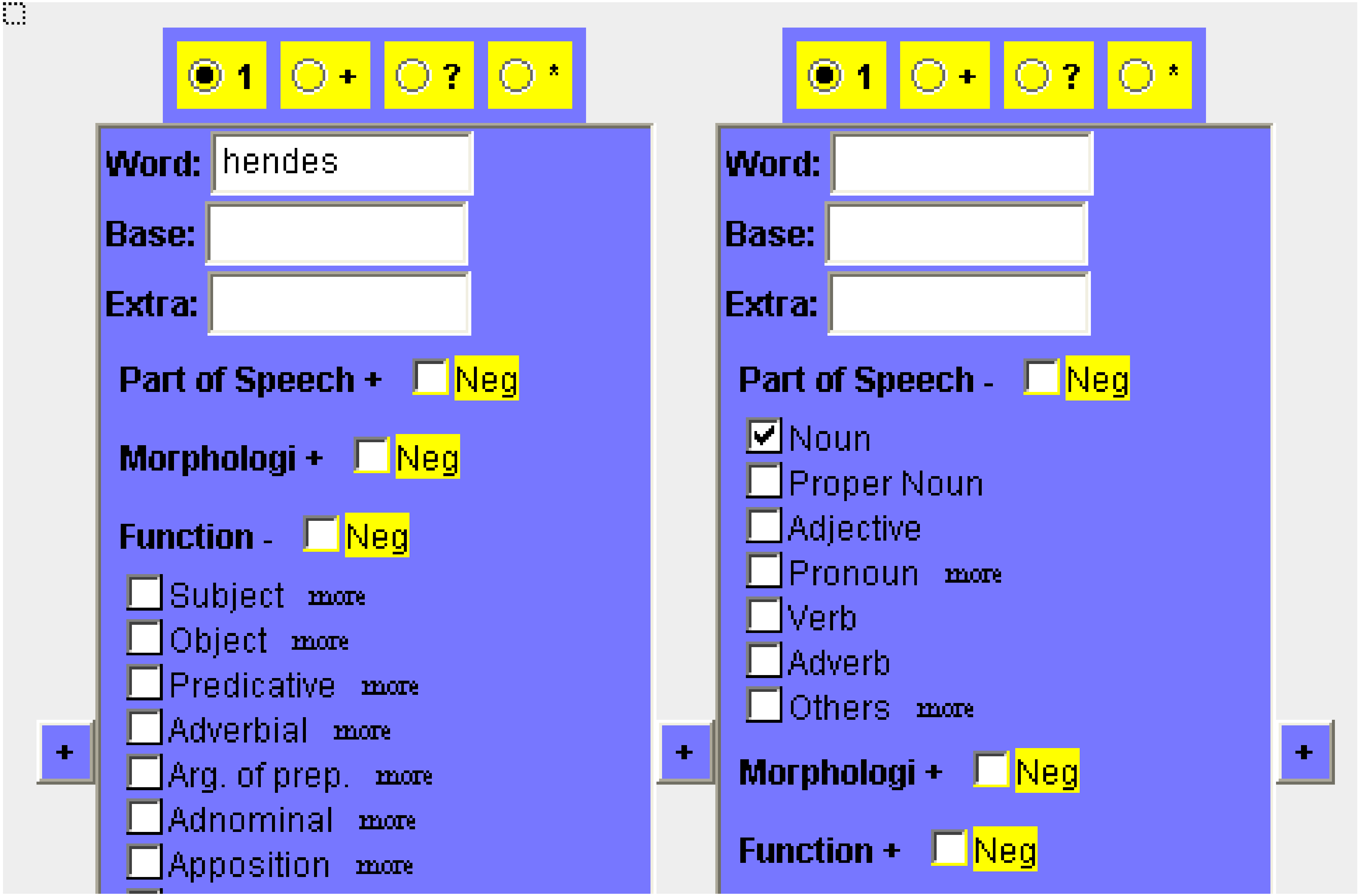
Simple text searches: fx. eg. composita



Search for: perle.*   normal

```
kvadratcentimeter ibenholt og perlemor sidder på gribebrættet , han
de første to kvarter en række perleafleveringer af_sted .
                  Importen af perler og smykker steg i 1999 med 15
       Som barokslottet er en perle af enkel pragt , er orglet i_si
                           En perlerække I_Forum havde Bob_Dylan de
srevyen som rigets kulturelle perle .
                At jeg kaster perler for svin i et kulturelt u-land
                     Trods en perlerække af smukke melodier og fine
benhavn trak en to meter lang perlekæde ud_af sin endetarm .
, og alt af værdi , juveler , perler - alt .
        I_går stod hun for en perlesmykket Maria_Stuart og en itali
                     En lille perle af en scene var , da Kelly førs
broderede fåreskinds-pelse og perlekæder .
 hjem og stiller bilen på den perlegrusdækkede gårdsplads .
tning blot er den sidste i en perlerække af gamle familieforretning
udeladt lyder det dernede fra perlegruset .
and , hvorefter vi fangede en perlehøne i luften , og til_sidst fan
ulighed for at gense en række perler .
som ærkeenglen Gabriel var en perlende og rendyrket fornøjelse , en
 , Kaj-bøger , hendes elskede perlekæder , og hvem tog babyalarmen
på højde med den motormæssige perle : boksramme i stål med alu-bags
```

Menu based category search



● 1 ○ + ○ ? ○ *

Word: hendes

Base:

Extra:

Part of Speech + □ Neg

Morphologi + □ Neg

Function - □ Neg

□ Subject more
□ Object more
□ Predicative more
□ Adverbial more
□ Arg. of prep. more
□ Adnominal more
□ Apposition more

+

● 1 ○ + ○ ? ○ *

Word:

Base:

Extra:

Part of Speech - □ Neg

☑ Noun
□ Proper Noun
□ Adjective
□ Pronoun more
□ Verb
□ Adverb
□ Others more

Morphologi + □ Neg

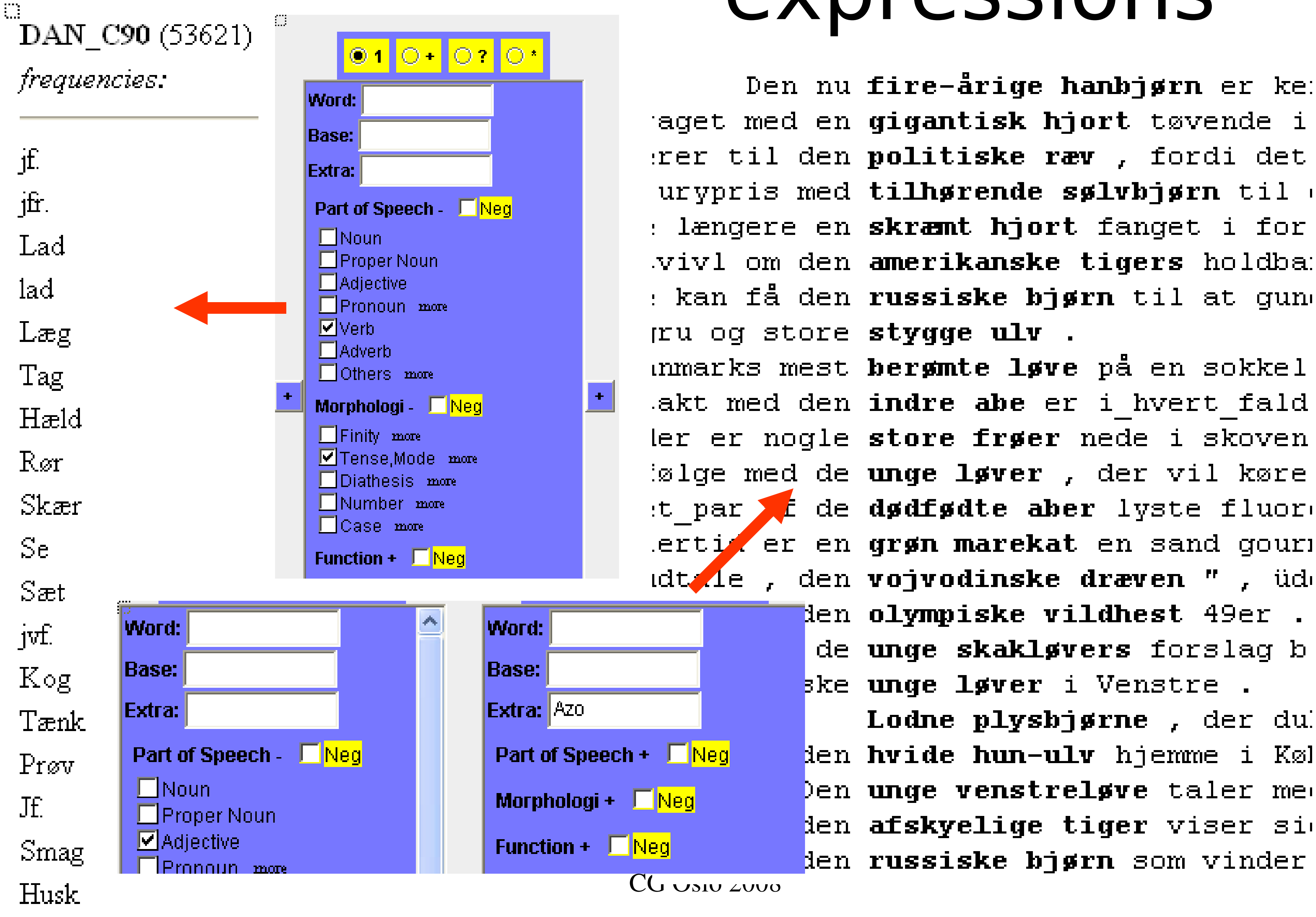
Function + □ Neg

+

+

imperatives

animal expressions



DAN_C90 (53621)

frequencies:

jf.
jfr.
Lad
lad
Læg
Tag
Hæld
Rør
Skær
Se
Sæt
jvf.
Kog
Tænk
Prøv
Jf.
Smag
Husk

● 1 ○ + ○ ? ○ *

Word:
Base:
Extra:

Part of Speech - □ Neg

□ Noun
□ Proper Noun
□ Adjective
□ Pronoun more
☑ Verb
□ Adverb
□ Others more

Morphologi - □ Neg

□ Finity more
☑ Tense,Mode more
□ Diathesis more
□ Number more
□ Case more

Function + □ Neg

Word:
Base:
Extra:

Part of Speech - □ Neg

□ Noun
□ Proper Noun
☑ Adjective
□ Pronoun more

Word:
Base:
Extra: Azo

Part of Speech + □ Neg

Morphologi + □ Neg

Function + □ Neg

```
 Den nu fire-årige hanbjørn er ke
aget med en gigantisk hjort tøvende i
rer til den politiske ræv , fordi det
urypris med tilhørende sølvbjørn til
 længere en skræmt hjort fanget i for
vivl om den amerikanske tigers holdba
 kan få den russiske bjørn til at gun
ru og store stygge ulv .
nmarks mest berømte løve på en sokkel
akt med den indre abe er i_hvert_fald
ler er nogle store frøer nede i skoven
ølge med de unge løver , der vil køre
t_par  f de dødfødte aber lyste fluor
erti  er en grøn marekat en sand gour
dt  le , den vojvodinske dræven " , üd
den olympiske vildhest 49er .
 de unge skakløvers forslag b
ske unge løver i Venstre .
    Lodne plysbjørne , der du
den hvide hun-ulv hjemme i Køl
)en unge venstreløve taler me
den afskyelige tiger viser si
den russiske bjørn som vinder
```