The *CatLog3* Technical Manual

Glyn V. Morrill

Departament de Ciències de la Computació
Universitat Politècnica de Catalunya

29th August, 2018

Contents

Chapter 1

Preamble

Logical categorial grammar is a version of categorial grammar in which the syntactic calculus is a universal categorial logic, i.e. the syntactic component is invariant from language to language, and a language is described entirely by its lexicon, and a grammar is purely lexical. *CatLog* is a series of programs for automatic logical categorial grammar semantic parsing (that is, the computer translation of human language into logic) developed by the present author; hitherto the programs have been written in Prolog. *CatLog3* is a system for such semantic parsing based on the particular program *CatLog* version l2.pl.

For execution, the engine *CatLog3* requires a logical categorial lexicon, and some linguistic examples. Running *CatLog3* on a lexicon and some examples supports textual listing of a lexicon and generation of linguistic analyses in the Prolog console: the linguistic expressions, associated sequent calculus derivations/syntactic structures, and logical forms. Such execution also creates files containing LaTeX code for the lexicon, and LaTeX code for outputs for the examples resulting from *CatLog3* linguistic analysis (the linguistic expressions, associated sequent calculus derivations/syntactic structures, and logical forms).

The structure of this technical manual is as follows. In Chapter 2 the parser/theorem-prover *CatLog3* is introduced. Chapter 3 illustrates execution outputting to the Prolog console. Chapter 4 lists the standard, predefined, linguistic examples of the file `examples.pl` of *CatLog3*, and Chapter 5 lists the standard, predefined, English lexicon of the file `lexicon.pl` of *CatLog3*. In Chapter 6 the LaTeX output is exemplified. The file `l2.pl` annexing this technical manual contains the Prolog source code of version l2.pl of *CatLog*, the program implementing parsing/theorem-proving, on the basis of what we call 'hedge'-sequent calculus, for the fragment of categorial type logic of *CatLog3* (Morrill 2017[11]); the files `example.pl` and `lexicon.pl` also annexing this technical manual contain the Prolog source code of the English linguistic examples and lexicon respectively.

Chapter 2

Introduction

The *CatLog* programs work by reducing grammar to logic: an expression α is of the grammatical type A if and only if an associated logical statement $Lex(\alpha) \Rightarrow A$, where $Lex(\alpha)$ is a lexical selection for α, is a theorem of a universal categorial type calculus. This type calculus is an intuitionistic sublinear logic. It is *sublinear* in that *theoremhood is not fully preserved under structural rules such as contraction or permutation* (word multiplicity and word order matter in grammar). It is *intuitionistic* in that its proofs are *constructive*, and the reading of a derived expression is a function of the meanings of its lexical expressions and of the constructive content of the proof by which it is deemed to be grammatical. I.e. the design entails *compositionality*, the principle which is generally formulated as stating that the meaning of a sentence is a function of the meanings of its words and of their mode of composition; this principle is attributed to the founder of modern logic G. Frege, although he never stated it as such; it's dual would appear to be *contextuality*: the principle according to which the meaning of a sentence is also a function of its 'context' (idea communicated to me by H. Hendriks).

This overall architectural design is purely lexical: a grammar is just a lexicon (dictionary) which is a set of ternary lexical items (signs) of the form:

lex(Prosodic form, Type, Logical form).

The Prosodic form is a string of words which may be partially bracketed; the brackets represent prosodic or syntactical domains such as intonational phrases or extraction islands. The Prosodic form is the phonosyntactical representation of the lexical item. The Type is a formula of the categorial type logic; the type represents the grammatical category of the lexical item. The Logical form is a closed term of higher-order logic representing the logical semantics of the lexical item. The language model defined by a set of lexical signs is the set of ternary signs derivable from/generated by this lexicon *Lex* according to the logic of the categorial grammatical types:

$$\{(\alpha, A, \phi) | \vdash Lex(\alpha) \Rightarrow A : \phi\}$$

As well as being called logical categorial grammar, grammar on this design has been called, for example, type logical categorial grammar, type logical grammar, or logical grammar; see, for example, Moortgat 1988[6], 1997[8]; Morrill 1994[14], 2011[15]; Carpenter 1997[1]; Jäger 2005[3]; Oehrle 2011[17]; or Moot and Retoré 2012[9]).

The task of semantic parsing is that of computing, for a given phonosyntactical expression α, and grammatical category A, the set of semantic readings (logical forms) of the expression α in type A according to a grammar.

The finite reading property (van Benthem 1991[18]) is the property according to which in each type an expression is associated with at most a finite number of logical forms. This seems to be an empirical property of human language: there appear to be no infinitely ambiguous expressions or, at least, no expressions which are infinitely ambiguous within a single grammatical category.

Clearly, in the absence of the finite reading property explicit semantic parsing is not computable since printing the members of an infinite set can never terminate.[1]

CatLog3 runs under SWI-Prolog version 7.6.4. There are three Prolog source files: the categorial calculus parser/theorem-proving engine `l2.pl`, the dictionary `lexicon.pl` of the substantial English fragment, and the small English corpus `examples.pl`. *CatLog3* computes, given an index of the identifiers of linguistic items in `examples.pl`, the expressions' associated logical forms according to the categorial dictionary of `lexicon.pl` and the categorial calculus of `l2.pl`. As well as pretty printing dictionaries and analyses in the Prolog console, *CatLog3* produces dictionary and analysis output files in the format of LaTeX typesetting code (based on Donald Knuth's TeX).

The complete list of files used in *CatLog3* is:

- `l2.pl`: the Prolog source coding the parser/theorem-prover for the categorial type logic of *CatLog3*;
- `lexicon.pl`: the Prolog source coding the dictionary of the logical categorial grammar of English of *CatLog3*, in which each lexical entry is of the form:

  `lex(`*Prosodic form*`,` *Type*`,` *Logical form*`)`

  where *Prosodic form* is a string of words which may be partially bracketed, *type* is a formula of the categorial type logic of *CatLog3*, and *Logical form* is a closed term of higher-order logic;
- `examples.pl`: the Prolog source coding the linguistic corpus of English examples of *CatLog3*, in which each linguistic example is of the form:

  `str(`*Identifier*`,` *Prosodic form*`,` *Type*`)`

  where *Identifier* is a Prolog term identifying the example, *Prosodic form* is the expression to be parsed, in the form of a string of words which may be partially bracketed, and *type* is the target category of the parsing, in the form of a formula of the categorial type logic of *CatLog3*;
- `lex.tex`: a derived LaTeX file coding a lexicon, created and generated by loading `l2.pl` and a dictionary/lexicon (such as `lexicon.pl`) into Prolog, and executing the query:

  `?- pplexlatex.`

[1] In the categorial logic of *CatLog3* there are two sources of possible non-compliance with the finite reading property and of possible consequent undecidabiity: the subexponentials $\{!, ?\}$, inspired by the exponentials of Girard (1987[2]), (which interact with the bracket modalities $\{\langle\rangle, [\,]^{-1}\}$ of Morrill 1992[10] and Moortgat 1996[7]), and the continuous product unit I and discontinuous product unit J of Lambek (1988[5]) and Morrill, Valentín, and Fadda (2011[13]) respectively. With respect to the units I and J, consider unit assignments in the lexicon (for 'null elements', for example), a lexical entry such as $0 : S/S : f$ in Lambek grammar, or $1 : S{\uparrow}S : f$ in displacement grammar. Both can make every sentence generated by the grammar infinitely ambiguous since these null functors can reapply to a sentence any number of times, changing it semantically but not phonosyntactically. And with respect to subexponentials !, Kanovich, Kuznetsov and Scedrov 2017[4] show undecidability of a bracket-conditioned contraction subexponential in the context of the Lambek calculus with bracket modalites.

Let us say that an occurrence of a connective in a lexical type is positive if it will (eventually) show up as the principal connective of an succedent type in sequent proof search, and that it is negative if it will (eventually) show up as the principal connective of an antecedent formula. In relation to possible non-computability we propose the following. Regarding the second source of infinite ambiguity above, to guarantee the finite reading property it is sufficient to require that in *CatLog3* lexical types are *bracket non-negative* (Morrill and Valentín 2015[12]) meaning that there must be no negative bracket modality $\langle\rangle$, nor any positive anti-bracket modality $[\,]^{-1}$, within any occurrence of a negative contraction subexponential !. And regarding the first source above, we propose, as in Morrill (2018[16]), that *CatLog3* lexical types be *unit non-negative* meaning that they have no continuous or discontinuous unit in negative position. So far as we are aware, in *CatLog3*, bracket and unit non-negativity, together with the assumption that lexical types have no negative expansion subexponentials ?, are sufficient to ensure the finite reading property and for the fragment of categorial logic.

- `der.tex`: a derived LaTeX file coding the output of *CatLog3* parsing, created and generated by loading into Prolog `l2.pl`, a dictionary/lexicon (such as `lexicon.pl`), and some linguistic examples (such as `examples.pl`), and by executing a query:

  ```
  ?- tt(Index).
  ```

  where *Index* is a Prolog term unifying with identifiers of the linguistic examples wanted;

- `out.tex`: a LaTeX file which includes the files `lex.tex` and `der.tex`, and which when typeset produces the file `out.pdf` containing the final formatted dictionary and analyses resulting from loading into Prolog `l2.pl`, a dictionary/lexicon, and some linguistic examples, and executing the queries:

  ```
  ?- pplexlatex.
  ```

  ```
  ?- tt(Index).
  ```

- `prooftree.sty`: Paul Taylor's macro package for formatting proof trees in LaTeX, used by `out.tex`.

Listouts of the SWI-Prolog source files `l2.pl`, `lexicon.pl` and `examples.pl` of *CatLog3* are given in the files `l2.pl`, `lexicon.pl`, and `examples.pl`, annexed to this technical manual.

Chapter 3

Outputting to the Prolog console

In *CatLog3* the program *CatLog* version l2 implementing parsing/theorem-proving for the categorial type calculus is in the file `l2.pl`, and a logical categorial grammar of English is represented by the dictionary in the file `lexicon.pl`. In the file `examples.pl` there is a corpus of linguistic examples which are expressions (prosodic forms) that are partially bracketed strings of English words.

3.1 Outputting of a dictionary to the Prolog console

In *CatLog3*, once `l2.pl` and `lexicon.pl` are loaded into SWI-Prolog, the query:

```
?- pplex.
```

pretty prints in the Prolog console the categorial dictionary in `lexicon.pl`. This is as follows:

```
MacBook-Air-de-Glyn:CatLog3 morrill$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

Cannot read termcap database;
using dumb terminal settings.
?- [12].
This is CatLog3. Copyright 2018 (C) Glyn V. Morrill.
CatLog3 comes with ABSOLUTELY NO WARRENTY. This is free software.
true.

?- [lexicon].
true.

?- pplex.
007: iLAgNt(s(g)): 007
a: iLAg(Af((SfciiLNt(s(g)))inSf)/CNs(g)): LALBEC[[A C) & (B C)]
admire: iL(<>ÆaNa-EgNt(s(g)))\Sf)/EaNa): ^LALB(Pres ((vadmire A) B))
And: iLAf(Sf/Sf): iAA
and: iLAaAf((eeiL(Sf/ueNa)\[]-1[]-1(Sf/ueNa)): [mapPhin,[app,s,0],and]
and: iLAaAf((eeiL(<>Na\Sf))\[]-1[]-1(<>Na\Sf)): [mapPhin,[app,s,0],and]
and: iLAf((eeiLSf\[]-1[]-1Sf)/iLSf): [mapPhin,0,and]
and: iLAf((eeiL(Sf/EaNa)\[]-1[]-1(Sf/EaNa))/iL(Sf/EaNa)): [mapPhin,[app,s,0],and]
and: iLAwAaAbAf((iL((SfciiL(((<>Na\Sf)o-Wo)/Nb))iiCWw)\[]-1[]-1((Sfci(((<>Na\Sf)o-Ww)/Nb))iiCWw)/^^iL((Sfci(((<>Na\Sf)o-Ww)/Nb))iiCWw)): LALBLC[[B C) & (A C)]
and: iLAfAa((eeiL((<>Na\Sf)/EbNb)\[]-1[]-1((<>Na\Sf)/EbNb))/iL((<>Na\Sf)/EbNb)): [mapPhin,[app,s,0],and]
and: iLAfAa((eeiL(((<>Na\Sf)/EbNb)\(<>Na\Sf))\[]-1[]-1(((<>Na\Sf)/EbNb)\(<>Na\Sf)))/iL(((<>Na\Sf)/EbNb)\(<>Na\Sf))): [mapPhin,[app,s,[app,s,0],and]
and: iLAfAa(iL(((<>Na\Sf)/(EbNb*Eg((CNg/CNg)|+|(CNg\CNg))))\(<>Na\Sf))\[]-1[]-1(((<>Na\Sf)/(EbNb*Eg((CNg/CNg)|+|(CNg\CNg))))\(<>Na\Sf)))/iL(((<>Na\Sf)/(EbNb*Eg((CNg/CNg)|+|(CNg\CNg))))\(<>Na\Sf))): [mapPhin,[app,s,[app,s,0],and]
and: iLAaAbAf((eeiL(((<>Na\Sf)/PPb)\[]-1[]-1(((<>Na\Sf)/PPb))/iL(((<>Na\Sf)/(EcNc*CPb))\(<>Na\Sf))): [mapPhin,[app,s,[app,s,0],and]
and: iLAaAbAf((eeiL(((<>Na\Sf)/(EcNc*PPb))\(<>Na\Sf))\[]-1[]-1(((<>Na\Sf)/(EcNc*PPb))\(<>Na\Sf))/iL(((<>Na\Sf)/(EcNc*PPb))\(<>Na\Sf))): [mapPhin,[app,s,[app,s,0],and]
and: iLAaAbAf((eeiL((<>Na\Sf)/ueNb)\[]-1[]-1((<>Na\Sf)/ueNb))/iL(((<>Na\Sf)/ueNb)): [mapPhin,[app,s,[app,s,0],and]
ate: nL((<>EaNa\Sf)/EaNa): ^LALB(Past ((veat A) B))
bagels: nL(Nt(p(n))&CNp(n)): ^((gen vbagels), vbagels)
barn: nLCNs(n): barn
be: nL((<>W[there]-oSb)/EaNa): ^LA(vbe A)
before: iL(AaAf((<>Na\Sf)\(<>Na\Sf))/Sf): LALBLC((before A) (B C))
beginning: nLCNs(n): beginning
believes: nL(((<>EgNt(s(g)))\Sf)/(CPthat|+|nLSf)): ^LALB(Pres ((vbelieve A) B))
bill: iLNt(s(m)): b
bond: iLNt(s(m)): b
book: nLCNs(n): book
bought: nL((<>EaNa\Sf)/(EaNa*EaNa)): ^LALB(Past (((vbuy piA) pi2A) B))
bought: nL((<>EaNa\Sf)/EaNa): ^LALB(Past ((vbuy A) B))
by: iLAa(((<>Na\S-)\(<>Na\S-))/Na): LALBLC[[C = A] & (B C)]
by: nL(An(CNn\CNn)/EaNa): ^LALB((vby A) B)
buys: nL((<>EgNt(s(g)))\Sf)/(EaNa*EaNa)): ^LALB(Pres (((vbuy piA) pi2A) B))
calls: nL((<>EgNt(s(g)))\Sf)/Ea((W[up].oNa)|+|(Nao.W[up]))): ^LALB(vphone A) B)
catch: nL((<>EaNa\Sb)/EaNa): ^LALB((vcatch A) B)
cezanne: iLNt(s(m)): c
cd: nLCNs(n): cd
charles: iLNt(s(m)): c
clark: iLAgNt(s(g)): c
coffee: nL(Nt(s(n))&CNs(n)): ^((gen vcoffee), vcoffee)
created: nL((<>EaNa\Sf)/EaNa): ^LALB(Past ((vcreate A) B))
darkness: nLCNs(n)&Nt(s(n))): ^(vdarkness, (gen vdarkness))
deep: nLCNs(n): deep
did: iLAaAgAbAh((((<>Na\Sg)ci(<>Nb\Sb))/(Ec<>Nc\Sf))\((<>Na\Sg)ci(<>Nb\Sb))): LALB((A B) B)
did+too: ((((<>NA\SB)ci(<>NC\SD))/(<>NE\SF))\(((<>NG\SH)ci(NI\SJ))): LKLL((K L) L)
doesnt: iLAgAa((Sgci(((<>Na\Sf)/(<>Na\Sb)))inSg): LA-(A LBLC(B C))
dog: nLCNs(n): dog
donuts: nL(Nt(p(n))&CNp(n)): ^((gen vdonuts), vdonuts)
earth: nLCNs(n): earth
eat: nL((<>EaNa\Sb)/EaNa): ^LALB((veat A) B)
edinburgh: iLNt(s(n)): e
editor: nL(AgCNs(g)/PPoF): editor
every: iLAg(Af((SfciNt(s(g)))inSf)/CNs(g)): LALBAC[[A C) -> (B C)]
```

```
everyone: nLAf((SfciAgNt(g))inSf): ~LAAB[(vperson B) -> (A B)]
face: nLCNs(n): face
fell: nL(Ea<>Na\Sf): ~lA(Past (vfall A))
filed: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Past ((vfile A) B))
finds: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vfind A) B))
fish: nLCNs(n): fish
for: iL(PPfor/EaNa): LAA
form: nL(CNs(n)&Nt(s(n))): ^(vform, (gen vform))
fortunately: nLAf(vSfinSf): fortunately
friends: nL(CNp/PPoF): friends
from: nL((AaAf((<>Na\Sf)\(<>Na\Sf))&An(CNn\CNn)/EbNb): ~lA((vfromadv A), (vfromadn A))
gave: nL((<>EaNa\Sf)/(EbNb*PPto)): ~LALB(Past (((vgive piZA) piIA) B))
gave: nL((<>EgNt(s(g))\Sf)/(EaNao.W[the,cold,shoulder])): ~LALB(Past ((vshun A) B))
gave: nL(((<>EaNa\Sf)/EaNa)/EaNa): ~LALBLC(Past (((vgive A) B) C))
girl: nLCNs(f): girl
gives: nL((<>EgNt(s(g))\Sf)/(EaNao.W[the,cold,shoulder])): ~LALB(Pres ((vshun A) B))
God: iLNt(s(m)): God
good: nLAn(CNn/CNn): good
has: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vhave A) B))
he: iL[]-1Ag((iLSg lca iLNt(s(m)))/(<>Nt(s(m))\Sg)): LAA
heaven: nLCNs(n): heaven
her: iLAgAa(((<>Na\Sg)ciiLNt(s(f)))in(iL(<>Na\Sg) lca iLNt(s(f)))): LAA
himself: iLAf(((<>Nt(s(m))\Sf)ciNt(s(m)))in(<>Nt(s(m))\Sf)): LALB((A B) B)
horse: nLCNs(n): horse
in: nL(AaAf((<>Na\Sf)\(<>Na\Sf))/EaNa): ~LALBLC((vin A) (B C))
in: nL(Af(SfndSf)/EaNa): in
is: iL(((<>EgNt(s(g))\Sf)/(EaNa+(Eg((CNg/CNg)-I)))\(CNg\CNg)-I))): LALB(Pres (A->C.[B = C]; D.(CD LE[E = B]) B)))
it: iLW[it]: 0
it: iLAfAa(((<>Na\Sg)ciiLNt(s(f)))in(iL(<>Na\Sg) lca iLNt(s(f)))): LAA
it: iL[]-1Af(((iLSf lca iLNt(s(n)))/(<>Nt(s(n))\Sf)): LAA
jogs: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vjog A))
john: iLNt(s(m)): j
laughs: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vlaugh A))
left: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vleave A))
let: nL(Sim/Sb): let
light: nLCNs(n)&Nt(s(n))): ^(vlight, (gen vlight))
likes: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vlike A) B))
logic: nLCNt(s(n))&CNs(n)): ^((gen vlogic), vlogic)
london: iLNt(s(n)): l
loses: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vlose A) B))
love: nL((<>EaNa\Sb)/EaNa): ~LALB(vlove A) B)
loved: nLAaAb(((<>Na\S->ciNb)dp(((<>Na\S->ciNb)inAg(CNg\CNg))): ^(vlove, LALBLC[(B C) & ED((A C) D)])
loves: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vlove A) B))
man: nLCNs(m): man
mary: iLNt(s(f)): m
met: nL((<>EaNa\Sf)/EaNa): ~LALB(Past ((vmeet A) B))
more: iLAhAgAf((Sfci(((ShciNt(p(g)))inSh)/CNp(g)))in(Sf/^(CPthanciiL(((ShciNt(p(g)))inSh)/CNp(g))))): LALB[|LC(A LDLE[(D C) & (E C)])| > |LFv(B LGLH[(G F) & (H F)])|]
mountain: nLCNs(n): mountain
moved: nL(<>EaNa\Sf): ~lA(Past (vmove A))
necessarily: iL(SA/nLSA): Nec
of: nL((An(CNn\CNn)/iLEbNb)&(PPof/EaNa)): ^(vof, LAA)
or: iLAf((eeiLSf\[]-1[]-1SF)/iLSF): [mapPhin,0,or]
or: iLAaAf((eeiL(<>Na\Sf)\[]-1[]-1(<>Na\Sf))/iL(<>Na\Sf)): [mapPhin,[app,s,0],or]
or: iLAf((eeiLSf/(<>EgNt(s(g))\Sf))\[]-1[]-1(Sf/(<>EgNt(s(g))\Sf)))/iL(Sf/(<>EgNt(s(g))\Sf))): [mapPhin,[app,s,0],or]
or: iLAaAf((eeiL(((<>Na\Sf)/EbNb)/EbNb)\[]-1[]-1(((<>Na\Sf)/EbNb)/EbNb)))/iL(((<>Na\Sf)/EbNb)/EbNb)): [mapPhin,[app,s,[app,s,[app,s,0]]],or]
painting: nLCNs(n)/PPoF): ~lA((vof A) vpainting)
paper: nLCNs(n): paper
park: nLCNs(n): park
past: nLAaAf(((<>Na\Sf)\(<>Na\Sf))/EbNb): ~lALBLC((vpast A) (B C))
perseverance: nL(Nt(s(n))&CNs(n)): ^((gen vperseverance), vperseverance)
peter: iLNt(s(m)): p
phonetics: nL(Nt(s(n))&CNs(n)): ^((gen vphonetics), vphonetics)
praises: nL((<>EgNt(s(g))\Sf)/EaNa): ~LALB(Pres ((vpraise A) B))
raced: nL(<>EaNa\Sf): ~lA(Past (vrace A))
raced: nLAaAb(((<>Na\S->ciNb)dp(((<>Na\S->ciNb)inAg(CNg\CNg))): ^(vrace2, LALBLC[(B C) & ED((A C) D)])
rains: nL(<>W[it]-oSf): ^(Pres vitrains)
reading: nL((<>EaNa\Spsp)/EaNa): ~LALB((vread A) B)
robin: iLAgNt(s(g)): r
said: nL((<>EaNa\Sf)/(EaNa+CPthat)): ~LALB(Past ((vsay A) B))
saw: nL((<>EaNa\Sf)/(EaNa+CPthat)): ~LALB(Past ((A->C.(vseee C); D.(vseet D)) B))
```

```
seeks: nL((<>EgNt(s(g))\Sf)/nLAaAf(((Na\Sf)/EbNb)\(Na\Sf))): ~lALB((vtries ^((vA vfind) B)) B)
sees: nL((<>EgNt(s(g))\Sf)/EaNa): ~lALB(Pres ((vsee A) B))
sent: nL((<>EaNa\Sf)/(EbNb*PPto)): ~lALB(Past (((vsent pi2A) pi1A) B))
sent: nL(((<>EaNa\Sf)/EaNa)/EaNa): ~lALBLC(Past (((vsend A) B) C))
she: iL[]-1Ag((iLSg 1ca iLNt(s(f)))/(<>Nt(s(f))\Sg)): LAA
sings: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vsing A))
slept: nL(<>EgNt(s(g))\Sf): ~lA(Past (vsleep A))
slowly: nLAaAf(nL(<>Na\Sf)\(<>nLNa\Sf)): ~lALB(vslowly ^(vA vB))
sneezed: nL(<>EgNt(s(g))\Sf): ~lA(Past (vsneeze A))
sold: nL((<>EaNa\Sf)/(EbNb*PPfor)): ~lALB(Past (((vsell pi2A) pi1A) B))
someone: nLAf((SfciiLAgNt(g)inSf): ~lAEB[(vperson B) & (A B)]
Spirit: iLCNs(m): Spirit
studies: nL(<>EgNt(s(g))\Sf)/EaNa): ~lALB(Pres ((vstudy A) B))
such+that: iLAn((CNn\CNn)/(Sf 1ca iLNt(n))): LALBLC[(B C) & (A C)]
suzy: iLNt(s(f)): s
talks: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vtalk A))
tall: nLAg(CNg/CNg): tall
teetotal: nLAn(CNn/CNn): ~lALB[(A B) & (vteetotal B)]
tenmilliondollars: iLNt(s(n)): tenmilliondollars
than: iL(CPthan/iLSf): LAA
that: iLAn([]-1[]-1(CNn\CNn)/iL(<>Nt(n)|&|ueiLNt(n))\Sf)): LALBLC[(B C) & (A C)]
that: iL(CPthat/iLSf): LAA
the: iLAn(Nt(n)/CNn): iota
the+cold+shoulder: iLW[the,cold,shoulder]: 0
there: iLW[there]: 0
thinks: nL((<>EgNt(s(g))\Sf)/(CPthat|+|nLSf)): ~lALB(Pres ((vthink A) B))
to: iL(PPto/EaNa)|&|An(<>Nn\Si)/(<>Nn\Sb))): LAA
today: nLAaAf(<>Na\Sf)\(<>Na\Sf)): ~lALB(vtoday (A B))
tries: nL((<>EgNt(s(g))\Sf)/nL(<>EgNt(s(g))\Si)): ~lALB((vtries ^(vA B)) B)
unicorn: nLCNs(n): unicorn
up: iLW[up]: 0
upon: nL(AbAf((<>Nb\Sf)\(<>Nb\Sf))&Ag(CNg\CNg))/EaNa): ~lA((vuponadv A), (vuponadn A))
void: nLAg(CNg/CNg): void
walk: nL(<>(EaNa-EgNt(s(g)))\Sf): ~lA(Pres (vwalk A))
walk: nL(<>EaNa\Sb): ~lA(vwalk A)
walks: nL(<>EgNt(s(g))\Sf): ~lA(Pres (vwalk A))
was: iL((<>EgNt(s(g))\Sf)/(EaNa+(Eg((CNg/CNg)|+|(CNg\CNg))-I))): LALB(Past (A->C_[B = C]; D.((D LE[E = B]) B)))
was: nL((<>W[there]-oSf)/EaNa): ~lA(Past (vbe A))
waters: nLCNp(n): waters
which: iLAnAm(Nt(n)\iNt(m))in([]-1[]-1(CNm\CNm)/iL((<>Nt(n)|&|ueiLNt(n))\Sf))): LALBLCLD[(C D) & (B (A D))]
who: iLAhAn([]-1[]-1(Nt(n)\(SheiNt(n))inSh))/iL((<>Nt(n)|&|ueiLNt(n))\Sf)): LALBLC[(A B) & (C B)]
will: iLAa((<>Na\Sf)/(<>Na\Sb)): LALB(Fut (A B))
without: nL(Ag(CNg\CNg)/EaNa): ~lALBLC[(B C) & -((vwith A) C)]
without: iLAaAf([]-1((<>Na\Sf)\(<>Na\Sf))/(<>Na\Spsp)): LALBLC[(B C) & -(A C)]
woman: nLCNs(f): woman
yesterday: nLAaAf((<>Na\Sf)\(<>Na\Sf)): ~lALB(vyesterday (A B))
true.

?-
```

3.2 Outputting of analyses to the Prolog console

The query

?- tt(*N*).

parses all the exemples the index of which unifies with the Prolog term *N*, and times the task. For example, a session with ?- tt(1) is as follows, with `examples.pl`, `lexicon.pl`, and `l2.pl`.as in the annexes to this technical manual already consulted.

```
MacBook-Air-de-Glyn:CatLog3 morrill$ swipl
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

Cannot read termcap database;
using dumb terminal settings.
?- [l2].
This is CatLog3. Copyright 2018 (C) Glyn V. Morrill.
CatLog3 comes with ABSOLUTELY NO WARRENTY. This is free software.
true.

?- [examples].
true.

?- [lexicon].
true.

?- tt(1).


(1) [john]+walks Sf


[iLNt(s(m)): j], nL(<>EgNt(s(g))\Sf): ^LA(Pres (vwalk A)) => Sf


[iLNt(s(m))], nL(<>EgNt(s(g))\Sf) => Sf [LiL]
    [iLNt(s(m))], <>EgNt(s(g))\Sf => Sf [\iL]
        [iLNt(s(m))] => <>EgNt(s(g)) [<>R]
            iLNt(s(m)) => EgNt(s(g)) [ER]
                iLNt(s(m)) => Nt(s(m)) [iLiL]
                    Nt(s(m)) => Nt(s(m))
        Sf => Sf

(Pres (vwalk j))
% 4,942 inferences, 0.003 CPU in 0.004 seconds (86% CPU, 1609772 Lips)
true
```

We see first the prosodic form (bracketed string) and the type in which it is to be analysed. Then there is the associated logic statement a proof of which would show grammaticality, including the

lexical semantics of the basic expressions. Then there is such a proof. Finally there is the logical translation that results from all this.

Chapter 4

examples.pl

```
str(dwp('7-7'), [b([john]), walks], s(f)).
str(dwp('7-16'), [b([every, man]), talks], s(f)).
str(dwp('7-19'), [b([the, fish]), walks], s(f)).
str(dwp('7-32'), [b([every, man]), b([b([walks, or, talks])])], s(f)).
str(dwp('7-34'), [b([b([b([every, man]), walks, or, b([every, man]), talks])])], s(f)).
str(dwp('7-39'), [b([b([b([a, woman]), walks, and, b([she]), talks])])], s(f)).
str(dwp('7-43, 45'), [b([john]), believes, that, b([a, fish]), walks], s(f)).
str(dwp('7-48, 49, 52'), [b([every, man]), believes, that, b([a, fish]), walks], s(f)).
str(dwp('7-57'), [b([every, fish, such, that, b([it]), walks]), talks], s(f)).
str(dwp('7-60, 62'), [b([john]), seeks, a, unicorn], s(f)).
str(dwp('7-73'), [b([john]), is, bill], s(f)).
str(dwp('7-76'), [b([john]), is, a, man], s(f)).
str(dwp('7-83'), [necessarily, b([john]), walks], s(f)).
str(dwp('7-86'), [b([john]), walks, slowly], s(f)).
str(dwp('7-91'), [b([john]), tries, to, walk], s(f)).
%str(dwp('7-94'), [b([john]), tries, to, b([b([catch, a, fish, and, eat, it])])], s(f)).
str(dwp('7-98'), [b([john]), finds, a, unicorn], s(f)).
str(dwp('7-105'), [b([every, man, such, that, b([he]), loves, a, woman]), loses, her], s(f)).
str(dwp('7-110'), [b([john]), walks, in, a, park], s(f)).
str(dwp('7-116, 118'), [b([every, man]), doesnt, walk], s(f)).

str(1, [b([john]), walks], s(f)).
str(2, [b([john]), loves, mary], s(f)).
str(3, [b([john]), thinks, b([mary]), walks], s(f)).
str(4, [b([mary]), believes, b([john]), thinks, b([mary]), walks], s(f)).
str(5, [b([mary]), buys, john, coffee], s(f)).
str(6, [man, loved], cn(s(m))).
str(7, [b([john]), is, loved], s(f)).
str(8, [b([john]), is, loved, by, mary], s(f)).
str(9, [b([the, man]), walks], s(f)).
str(10, [b([the, man]), loves, the, woman], s(f)).
str(11, [man, b([b([that, walks])])], cn(s(m))).
str(12, [b([the, man, b([b([that, walks])])]), talks], s(f)).
str(13, [b([the, man, b([b([that, loves, mary])])]), walks], s(f)).
str(14, [b([the, man, b([b([that, b([mary]), loves])])]), walks], s(f)).
str(15, [b([the, man, b([b([that, b([john]), thinks, b([mary]), loves])])]), walks], s(f)).
str(16, [b([the, horse]), raced, past, the, barn], s(f)).
str(17, [b([the, horse, raced, past, the, barn]), fell], s(f)).

str(18, [b([john]), walks, from, edinburgh], s(f)).
str(19, [b([the, man, from, edinburgh]), walks], s(f)).

str(20, [b([bond]), is, '007'], s(f)).
str(21, [b([bond]), is, teetotal], s(f)).
str(22, [b([bond]), is, b([b(['007', and, teetotal])])], s(f)).
str(23, [b([bond]), is, b([b([teetotal, and, '007'])])], s(f)).
str(24, [b([bond]), is, b([b([tall, and, '007'])])], s(f)).

str(tdc(43), [b([mary]), gave, the, man, the, cold, shoulder], s(f)).
str(tdc(4343), [b([mary]), gave, john, the, cold, shoulder], s(f)).
str(tdc(47), [b([john]), gave, every, book, to, mary], s(f)).
str(tdc(50), [b([mary]), thinks, b([someone]), left], s(f)).
str(tdc(53), [b([everyone]), loves, someone], s(f)).
str(tdc(54), [dog, b([b([that, b([mary]), saw, today])])], cn(s(n))).
str(tdc('58a'), [b([john]), slept, before, b([mary]), did], s(f)).
%str(tdc('58b'), [john, slept, and, mary, did, too], s(f)).
str(tdc(64), [mountain, b([b([the, painting, of, which, by, cezanne, b([john]), sold, for, tenmilliondollars])])], cn(s(n))).
%str(56, [thesis, the, height, of, the, lettering, on, the, first, line, of, the, second, page, of, the, third, chapter, of, which, is, '0.5cm'], cn(n)).
str(tdc(67), [b([john, b([b([who, jogs])])]), sneezed], s(f)).
str(tdc('70a'), [fortunately, b([john]), has, perseverance], s(f)).
str(tdc('70b'), [b([john]), fortunately, has, perseverance], s(f)).
str(tdc('70c'), [b([john]), has, fortunately, perseverance], s(f)).
str(tdc('70d'), [b([john]), has, perseverance, fortunately], s(f)).
%str(tdc(73), [b([b([b([john]), studies, logic, and, b([charles])), phonetics])])], s(f)).
str(tdc(75), [b([john]), ate, more, donuts, than, b([mary]), bought, bagels], s(f)).
str(tdc('86a'), [b([john]), bought, himself, coffee], s(f)).
%str(65, [dorothy, bet, the, straw, man, half, of, himself, that, she, would, reach, emerald, city, first], s).

str(w(1), [b([it]), rains], s(f)).
```

```
str(w(2), [b([mary]), gives, peter, the, cold, shoulder], s(f)).
str(w('3a'), [b([mary]), calls, peter, up], s(f)).
str(w('3b'), [b([mary]), calls, up, peter], s(f)).
%str(w('4a'), [b([b([b([peter]), sees, mary, and, b([robin]), clark])])], s(f)).
%str(w('4b'), [b([b([b([peter]), calls, up, mary, and, b([robin]), clark])])], s(f)).
%str(w('4c'), [b([b([b([peter]), calls, mary, up, and, b([robin]), clark])])], s(f)).
%str(w('4d'), [b([b([b([peter]), gives, mary, the, cold, shoulder, and, b([robin]), clark])])], s(f)).

str(eac(rel(6)), [man, b([b([that, walks])])], cn(s(m))).
str(eac(rel(7)), [man,b([b([that,b([mary]),likes])])], cn(s(m))).
str(eac(rel(eight(a))), [b([john]), likes, the, man], s(f)).
str(eac(rel(eight(b))), [b([mary]), thinks, that, b([john]), likes, the, man], s(f)).
str(eac(rel(eight(c))), [b([suzy]), believes, that, b([mary]), thinks, that, b([john]), likes, the, man], s(f)).
str(eac(rel(eight(d))), [man, b([b([that, b([john]), likes])])], cn(s(m))).
str(eac(rel(eight(e))), [man, b([b([that, b([mary]), thinks, that, b([john]), likes])])], cn(s(m))).
str(eac(rel(eight(f))), [man, b([b([that, b([suzy]), believes, that, b([mary]), thinks, that, b([john]), likes])])], cn(s(m))).
str(eac(rel(eight(g))), [b([b([b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(eight(h))), [b([b([b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(eight(i))), [b([b([b([suzy]), laughs, b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(eight(j))), [b([b([b([bill]), walks, b([suzy]), laughs, b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(eight(k))), [b([b([b([suzy]), talks, b([bill]), walks, b([suzy]), laughs, b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(eight(l))), [b([b([b([john]), sings, b([suzy]), talks, b([bill]), walks, b([suzy]), laughs, b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(rel(9)), [man,b([b([that,b([mary]),likes,today])])], cn(s(m))).
str(eac(rel(10)), [man, that, b([the, friends, of]), walk], cn(s(m))).
str(eac(rel(11)), [man, b([b([that, the, friends, of, admire])])], cn(s(m))).
str(eac(rel(12)), [paper, b([b([that, b([john]), filed, without, reading])])], cn(s(n))).
%str(rel(13), [paper, b([b([that, the, editor, of, filed, without, reading])])], cn(s(n))).

str(eac(crd(14)), [b([b([b([john]), praises, mary, and, b([john]), laughs])])], s(f)).
str(eac(crd(15)), [b([john]), b([b([praises, mary, and, laughs])])], s(f)).
str(eac(crd(16)), [b([john]), b([b([likes, and, will, love])]), london], s(f)).
str(eac(crd(17)), [b([john]), b([b([gave, or, sent])]), mary, the, book], s(f)).
str(eac(crd(18)), [b([b([b([john]), or, b([mary])])])], sings], s(f)).
str(eac(crd(19)), [b([john]), loves, b([b([mary, and, himself])])], s(f)).
str(eac(crd(20)), [b([b([b([john]), likes, and, b([mary]), loves])]), london], s(f)).
str(eac(crd(21)), [b([john]), b([b([gave, the, book, and, sent, the, cd])]), to, mary], s(f)).
str(eac(crd(22)), [b([john]), gave, b([b([the, book, to, mary, and, the, cd, to, suzy])])], s(f)).
str(eac(crd(23)), [b([john]), saw, b([b([mary, today, and, bill, yesterday])])], s(f)).
str(crd(24), [man, b([b([that, b([b([b([john]), saw, yesterday, and, b([bill]), saw, today])])])])], cn(s(m))).
%str(crd(25), [man, b([b([that, b([john]), b([b([saw, yesterday, and, met, today])])])])], cn(s(m))).
str(eac(crd(26)), [b([b([b([john]), walks, b([mary]), talks, and, b([bill]), sings])])], s(f)).
str(eac(crd(27)), [b([john]), b([b([walks, talks, and, sings])])], s(f)).
str(eac(crd(28)), [b([john]), b([b([praises, likes, and, will, love])]), london], s(f)).

str(atb(1), [man, b([b([that, b([b([b([john]), likes, and, b([mary]), loves])])])])], cn(s(m))).

str(atb(2), [man, b([b([that, b([mary]), b([b([loves, and, praises])])])])], cn(s(m))).

str(gen(one(1)), [in, the, beginning, b(['God']), created, b([b([the, heaven, and, the,
earth])])], s(f)).

%str(gen(one('2'(a))), ['And', b([b([b([the, earth]), was, b([b([without, form, and, void])]), and, b([darkness]), was, upon, the, face, of, the, deep])])], s(f)).

str(gen(one('2'(a(i)))), [b([the, earth]), was, b([b([without, form, and, void])])], s(f)).

str(gen(one('2'(a(ii)))), [b([darkness]), was, upon, the, face, of, the, deep], s(f)).

str(gen(one('2'(b))), ['And', b([the, 'Spirit', of, 'God']), moved, upon,
the, face, of, the, waters], s(f)).

str(gen(one(3)), ['And', b([b([b(['God']), said, let, b([there]), be, light,
and, b([there]), was, light])])], s(f)).

%str(gen(one('4'(a))), ['And', b(['God']), saw, b([b([the, light, and, that, b([it]), was, good])])], s(f)).

%str(gen(one('4'(aa))), ['And', b(['God']), saw, b([b([the, light,
%and, that, b([the, light]), was, good])])], s(f)).

% str(gen(one('4'(b))), [b(['God']), divided, the, light, from, the, darkness], s(f)).
```

```
% str(gen(one('S'(a))), ['And', b([b([b(['God']), called, the, light, day, and, the, darkness, he, called, night])])], s(f)).
```

Chapter 5

lexicon.pl

```
lex(['007'], iL g iu n(t(s(g))), '007').
lex([a], iL g iu (f iu ((s(f) ci iL n(t(s(g)))) in s(f))/cn(s(g))), [lmd, X, [lmd, Y, [xst, Z, [and, [app, X, Z], [app, Y, Z]]]]]).
lex([admire], iL ((br((a ie n(a)) - (g ie n(t(s(g))))) bs s(f))/(a ie n(a))), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, admire], X], Y]]]]]).
lex(['And'], iL f iu (s(f)/s(f)), [lmd, X, X]).
lex([and], iL a iu f iu((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, 0], and]) :-
      A = s(f)/ue n(a);
      A = br n(a)bs s(f).
lex([and], iL f iu((ee iL s(f)bs ab ab s(f))/iL s(f)), [mapPhin, 0, and]).
lex([and], iL f iu((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, 0], and]) :-
      A = s(f)/a ie n(a).
lex([and], iL w iu a iu b iu f iu ((iL A bs ab ab A)/bg bg iL A), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [app, X, Z]]]]]) :-
      A = (s(f) ci(((br n(a) bs s(f))rio w(w))/n(b)))licn w(w).
lex([and], iL f iu a iu((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, [app, s, 0]], and]) :-
      A = (br n(a) bs s(f))/b ie n(b);
      A = ((br n(a) bs s(f))/b ie n(b))bs(br n(a) bs s(f)).
lex([and], iL f iu a iu((iL A bs ab ab A)/iL A), [lmd, X, [lmd, Y, [lmd, Z, [lmd, W, [and, [app, [app, Y, Z], W], [app, [app, X, Z], W]]]]]]) :-
      A = ((br n(a) bs s(f))/((b ie n(b))+(g ie ((cn(g)/cn(g))iad(cn(g) bs cn(g)))))bs(br n(a) bs s(f)).
lex([and], iL a iu b iu f iu ((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, [app, s, 0]], and]) :-
      A = ((br n(a)bs s(f))/(c ie n(c)+cp(b)))bs(br n(a)bs s(f));
      A = (br n(a) bs s(f))/pp(b);
      A = ((br n(a)bs s(f))/(c ie n(c)*pp(b)))bs(br n(a) bs s(f));
      A = (br n(a)bs s(f))/ue n(b).
lex([ate], nL ((br a ie n(a) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, eat], X], Y]]]]]).
lex([bagels], nL (n(t(p(n)))&cn(p(n))), [up, [pair, [app, gen, [dn, bagels]], [dn, bagels]]]).
lex([barn], nL cn(s(n)), barn).
lex([be], nL ((br w([there]) liu s(b))/a ie n(a)), [up, [lmd, X, [app,
[dn, be], X]]]).
lex([before], iL(a iu f iu ((br n(a) bs s(f))bs (br n(a) bs s(f)))/s(f)), [lmd, X, [lmd, Y, [lmd, Z, [app, [app, before, X], [app, Y, Z]]]]]).
lex([beginning], nL cn(s(n)), beginning).
lex([believes], nL ((br g ie n(t(s(g))) bs s(f))/(cp(that) iad nL s(f))), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, believe], X], Y]]]]]).
lex([bill], iL n(t(s(m))), b).
lex([bond], iL n(t(s(m))), b).
lex([book], nL cn(s(n)), book).
lex([bought], nL ((br a ie n(a) bs s(f))/((a ie n(a))* (a ie n(a)))), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [app, [dn, buy], [fst, X]], [snd, X]], Y]]]]]).
lex([bought], nL ((br a ie n(a) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, buy], X], Y]]]]]).
lex([by], iL(a iu (((br n(a) bs s(-))bs (br n(a) bs s(-)))/n(a))), [lmd, X, [lmd, Y, [lmd, Z, [and, [eq, Z, X], [app, Y, Z]]]]]).
lex([by], nL (n iu (cn(n) bs cn(n))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, [app, [dn, by], X], Y]]]]).
lex([buys], nL ((br g ie n(t(s(g))) bs s(f))/((a ie n(a))* (a ie n(a)))), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [app, [dn, buy], [fst, X]], [snd, X]], Y]]]]]).
lex([calls], nL ((br g ie n(t(s(g))) bs s(f))/a ie ((w([up])lip n(a))iad(n(a)rip w([up])))), [up, [lmd, X, [lmd, Y, [app, [app, [dn, phone], X], Y]]]]).
lex([catch], nL ((br a ie n(a) bs s(b))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, [app, [dn, catch], X], Y]]]]).
lex([cezanne], iL n(t(s(m))), c).
lex([cd], nL cn(s(n)), cd).
lex([charles], iL n(t(s(m))), c).
lex([clark], iL g iu n(t(s(g))), c).
lex([coffee], nL (n(t(s(n)))&cn(s(n))), [up, [pair, [app, gen, [dn, coffee]], [dn, coffee]]]).
lex([created], nL ((br a ie n(a) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, create], X], Y]]]]]).
lex([darkness], nL (cn(s(n))&n(t(s(n)))), [up, [pair, [dn, darkness], [app, gen, [dn, darkness]]]]).
lex([deep], nL cn(s(n)), deep).
lex([did], iL a iu g iu b iu h iu ((((br n(a) bs s(g)) ci (br n(b) bs s(h)))/(c ie br n(c) bs s(f)))bs((br n(a) bs s(g))ci(br n(b) bs s(h)))),
[lmd, X, [lmd, Y, [app, [app, X, Y], Y]]]).
lex([did, too], (((br n(_) bs s(_)) ci (br n(_) bs s(_)))/(br n(_) bs s(_)))bs((br n(_) bs s(_))ci(n(_) bs s(_))),
[lmd, X, [lmd, Y, [app, [app, X, Y], Y]]]).
lex([doesnt], iL g iu a iu ((s(g)ci((br n(a)bs s(f))/(br n(a)bs s(b)))in s(g)),
[lmd, X, [not, [app, X, [lmd, Y, [lmd, Z, [app, Y, Z]]]]]]).
lex([dog], nL cn(s(n)), dog).
lex([donuts], nL (n(t(p(n)))&cn(p(n))), [up, [pair, [app, gen, [dn, donuts]], [dn, donuts]]]).
lex([earth], nL cn(s(n)), earth).
lex([eat], nL ((br a ie n(a) bs s(b))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, [app, [dn, eat], X], Y]]]]).
lex([edinburgh], iL n(t(s(n))), e).
lex([editor], nL (g iu cn(s(g))/pp(of)), editor).
lex([every], iL g iu (f iu ((s(f) ci n(t(s(g)))) in s(f))/cn(s(g))), [lmd, X, [lmd, Y, [all, Z, [imply, [app, X, Z], [app, Y, Z]]]]]).
lex([everyone], nL  f iu ((s(f) ci g iu n(t(g))) in s(f)), [up, [lmd, X, [all, Y, [imply, [app, [dn, person], Y], [app, X, Y]]]]]).
lex([face], nL cn(s(n)), face).
lex([fell], nL (a ie br n(a)bs s(f)), [up, [lmd, X, [app, 'Past', [app, [dn, fall], X]]]]).
lex([filed], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, file], X], Y]]]]]).
lex([finds], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, find], X], Y]]]]]).
lex([fish], nL cn(s(n)), fish).
lex([for], iL(pp(for)/a ie n(a)), [lmd, X, X]).
lex([form], nL (cn(s(n))&n(t(s(n)))), [up, [pair, [dn, form], [app, gen, [dn, form]]]]).
lex([fortunately], nL f iu (sp s(f)in s(f)), fortunately).
```

```
lex([friends], nL (cn(p)/pp(of)), friends).
lex([from], nL ((a iu f iu ((br n(a)bs s(f))bs(br n(a)bs s(f)))&(n iu (cn(n)bs cn(n))))/(b ie n(b))), [up, [lmd, X, [pair, [app, [dn, fromadv], X], [app, [dn, fromadn], X]]]]).
%lex([is], iL((g ie n(t(s(g)))bs s(f))/(g ie (cn(g) bs cn(g)))), [lmd, X, [lmd, Y, [app, [app, X, [lmd, Z, [eq, Z, Y]]], Y]]]).
lex([gave], nL ((br a ie n(a) bs s(f))/(b ie n(b)*pp(to))), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [app, [dn, give], [snd, X]], [fst, X]], Y]]]]]).
lex([gave], nL ((br g ie n(t(s(g))) bs s(f))/(a ie n(a) rip w([the, cold, shoulder]))), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, shun], X], Y]]]]]).
lex([gave], nL (((br (a ie n(a)) bs s(f))/(a ie n(a)))/(a ie n(a))), [up, [lmd, X, [lmd, Y, [lmd, Z, [app, 'Past', [app, [app, [app, [dn, give], X], Y], Z]]]]]]).
%lex([gave, 1, the, cold, shoulder], (n(_) bs s(f)) ci n(_), shun).
lex([girl], nL cn(s(f)), girl).
lex([gives], nL ((br g ie n(t(s(g))) bs s(f))/(a ie n(a) rip w([the, cold, shoulder]))), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, shun], X], Y]]]]]).
lex(['God'], iL n(t(s(m))), 'God').
lex([good], nL n iu (cn(n)/cn(n)), good).
lex([has], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, have], X], Y]]]]]).
lex([he], iL ab g iu ((iL s(g) lca iL n(t(s(m))))/(br n(t(s(m))) bs s(g))), [lmd,X,X]).
lex([heaven], nL cn(s(n)), heaven).
lex([her], iL g iu a iu (((br n(a)bs s(g)) ci iL n(t(s(f)))) in (iL (br n(a)bs s(g)) lca iL n(t(s(f))))), [lmd,X,X]).
lex([himself], iL f iu(((br n(t(s(m))) bs s(f))ci n(t(s(m))))in((br n(t(s(m))) bs s(f)))), [lmd, X, [lmd, Y, [app, [app, X, Y], Y]]]).
lex([horse], nL cn(s(n)), horse).
lex([in], nL (a iu f iu((br n(a) bs s(f)) bs (br n(a) bs s(f)))/a ie n(a)), [up, [lmd, X, [lmd, Y, [lmd, Z, [app, [app, [dn, in], X], [app, Y, Z]]]]]]).
lex([in], nL  (f iu (s(f) nd s(f))/ a ie n(a)), in).
lex([is], iL((br g ie n(t(s(g)))bs s(f))/((a ie n(a))+(g ie ((cn(g)/cn(g))iad(cn(g) bs cn(g))
)-i))), [lmd, X, [lmd, Y, [app, 'Pres', [case, X, Z, [eq, Y, Z], W, [app, [app, W, [lmd, U, [eq, U, Y]]], Y]]]]]).
lex([it], iL w([it]), 0).
%lex([it], iL f iu ((s(f) ci iL n(t(s(n)))) in (iL s(f) lca iL n(t(s(n))))), [lmd,X,X]).
lex([it], iL f iu a iu (((br n(a)bs s(f)) ci iL n(t(s(n)))) in (iL (br n(a)bs s(f)) lca iL n(t(s(n))))), [lmd,X,X]).
lex([it], iL ab f iu ((iL s(f) lca iL n(t(s(n))))/(br n(t(s(n))) bs s(f))),
[lmd,X,X]).
lex([jogs], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, jog], X]]]]).
lex([john], iL n(t(s(m))), j).
lex([laughs], nL (br g ie n(t(s(g))) bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, laugh], X]]]]).
lex([left], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, leave], X]]]]).
lex([let], nL (s(im)/s(b)), let).
lex([light], nL (cn(s(n))&n(t(s(n)))), [up, [pair, [dn, light], [app, gen, [dn, light]]]]).
lex([likes], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, like], X], Y]]]]]).
lex([logic], nL (n(t(s(n)))&cn(s(n))), [up, [pair, [app, gen, [dn, logic]], [dn, logic]]]).
lex([london], iL n(t(s(n))), l).
lex([loses], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, lose], X], Y]]]]]).
lex([love], nL ((br a ie n(a) bs s(b))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, [app, [dn, love], X], Y]]]]).
lex([loved], nL (a iu (b iu (((br n(a) bs s(-))ci n(b))dp(((br n(a) bs s(-))ci n(b))in(g iu (cn(g)bs cn(g)))))))), [up, [pair, [dn, love], [lmd, X, [lmd, Y, [lmd, Z, [and,
[app, Y, Z], [xst, W, [app, [app, X, Z], W]]]]]]]]).
lex([loves], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, love], X], Y]]]]]).
lex([man], nL cn(s(m)), man).
lex([mary], iL n(t(s(f))), m).
lex([met], nL ((br a ie n(a)bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [dn, meet], X], Y]]]]]).
lex([more], iL h iu g iu f iu ((s(f) ci Q) in(s(f)/bg(cp(than) ci iL Q))), [lmd, X, [lmd, Y, [more_than, [card, [lmd, Z, [app, X,
[lmd,P,[lmd,R,[and,[app,P,Z],[app,R,Z]]]]]], [card, [lmd, Z1, [dn, [app, Y, [lmd, P1, [lmd, Q1, [and, [app, P1, Z1],
[app, Q1, Z1]]]]]]]]]]) :- Q=((s(h) ci n(t(p(g)))) in s(h))/cn(p(g)).
lex([mountain], nL cn(s(n)), mountain).
lex([moved], nL (br a ie n(a) bs s(f)), [up, [lmd, X, [app, 'Past', [app, [dn, move], X]]]]).
lex([necessarily], iL(s(F)/nL s(F)), 'Nec').
lex([of], nL (((n iu (cn(n) bs cn(n)))/iL (b ie n(b)))&(pp(of)/(a ie n(a)))), [up, [pair, [dn, of], [lmd, X, X]]]).
lex([or], iL f iu ((ee iL s(f)bs ab ab s(f))/iL s(f)), [mapPhin, 0, or]).
lex([or], iL a iu f iu ((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, 0], or]) :- A = br n(a) bs s(f).
lex([or], iL f iu ((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, 0], or]) :-
      A = s(f)/(br g ie n(t(s(g))) bs s(f)).
lex([or], iL a iu f iu((ee iL A bs ab ab A)/iL A), [mapPhin, [app, s, [app, s, [app, s, 0]]], or]) :-
      A = (((br n(a)) bs s(f))/(b ie n(b)))/(b ie n(b)).
lex([painting], nL (cn(s(n))/pp(of)), [up, [lmd, X, [app, [app, [dn, of], X], [dn, painting]]]]).
lex([paper], nL cn(s(n)), paper).
lex([park], nL cn(s(n)), park).
lex([past], nL a iu f iu(((br n(a)bs s(f))bs (br n(a)bs s(f)))/(b ie n(b))), [up, [lmd, X, [lmd, Y, [lmd, Z, [app, [app, [dn, past], X], [app, Y, Z]]]]]]).
lex([perseverance], nL (n(t(s(n)))&cn(s(n))), [up, [pair, [app, gen, [dn, perseverance]], [dn, perseverance]]]).
lex([peter], iL n(t(s(m))), p).
lex([phonetics], nL (n(t(s(n)))&cn(s(n))), [up, [pair, [app, gen, [dn, phonetics]], [dn, phonetics]]]).
lex([praises], nL ((br (g ie n(t(s(g)))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, praise], X], Y]]]]]).
lex([raced], nL (br(a ie n(a))bs s(f)), [up, [lmd, X, [app, 'Past', [app, [dn, race], X]]]]).
%lex([raced], ((cn(G)bs cn(G))/(n(A)bs(n(B)bs s(-))))*(n(A)bs(n(B)bs s(-))),
%[pair, [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [xst, W, [app, [app, X, Z], %W]]]]]], race2]).
lex([raced], nL (a iu (b iu (((br n(a) bs s(-))ci n(b))dp(((br n(a) bs s(-))ci n(b))in(g iu (cn(g)bs cn(g)))))))), [up, [pair, [dn, race2], [lmd, X, [lmd, Y, [lmd, Z, [and,
[app, Y, Z], [xst, W, [app, [app, X, Z], W]]]]]]]]).
lex([rains], nL (br w([it]) liu s(f)), [up, [app, 'Pres', [dn, itrains]]]).
```

```
lex([reading], nL ((br a ie n(a) bs s(psp))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, [app, [dn, read], X], Y]]]]).
lex([robin], iL g iu n(t(s(g))), r).
lex([said], nL ((br a ie n(a) bs s(f))/s(im)), [up, [lmd, X, [lmd, Y, [app, 'Past',
[app, [app, [dn, say], X], Y]]]]]).
lex([saw], nL ((br a ie n(a) bs s(f))/(a ie n(a)+cp(that))), [up, [lmd, X, [lmd, Y,
[app, 'Past', [app, [case, X, X1, [app, [dn, seee], X1], X2, [app, [dn, seet], X2]], Y]]]]]).
lex([seeks], nL ((br g ie n(t(s(g))) bs s(f))/nL a iu f iu (((n(a) bs s(f))/b ie n(b)) bs (n(a) bs s(f)))), [up, [lmd, X, [lmd, Y, [app, [app, [dn, tries], [up, [app, [app, [dn, X], [dn, find]], Y]]], Y]]]]).
lex([sees], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, see], X], Y]]]]]).
lex([sent], nL ((br a ie n(a) bs s(f))/(b ie n(b)*pp(to))), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [app, [dn, sent], [snd, X]], [fst, X]], Y]]]]]).
lex([sent], nL (((br (a ie n(a)) bs s(f))/(a ie n(a)))/(a ie n(a))), [up, [lmd, X, [lmd, Y, [lmd, Z, [app, 'Past', [app, [app, [app, [dn, send], X], Y], Z]]]]]]).
lex([she], iL ab g iu ((iL s(g) lca iL n(t(s(f))))/(br n(t(s(f))) bs s(g))), [lmd,X,X]).
lex([sings], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, sing], X]]]]).
lex([slept], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Past', [app, [dn, sleep], X]]]]).
lex([slowly], nL a iu f iu(nL (br n(a) bs s(f)) bs (br nL n(a) bs s(f))), [up, [lmd, X, [lmd, Y, [app, [dn, slowly], [up, [app, [dn, X], [dn, Y]]]]]]]).
lex([sneezed], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Past', [app, [dn, sneeze], X]]]]).
lex([sold], nL ((br a ie n(a) bs s(f))/(b ie n(b)*pp(for))), [up, [lmd, X, [lmd, Y, [app, 'Past', [app, [app, [app, [dn, sell], [snd, X]], [fst, X]], Y]]]]]).
lex([someone], nL  f iu ((s(f) ci iL g iu n(t(g))) in s(f)), [up, [lmd, X, [xst, Y, [and, [app, [dn, person], Y], [app, X, Y]]]]]).
lex(['Spirit'], iL cn(s(m)), 'Spirit').
lex([studies], nL ((br g ie n(t(s(g))) bs s(f))/a ie n(a)), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, study], X], Y]]]]]).
lex([such, that], iL n iu ((cn(n) bs cn(n))/(s(f) lca iL n(t(n)))), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [app, X, Z]]]]]).
lex([suzy], iL n(t(s(f))), s).
lex([talks], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, talk], X]]]]).
lex([tall], nL (g iu (cn(g)/cn(g))), tall).
lex([teetotal], nL  n iu (cn(n)/cn(n)), [up, [lmd, X, [lmd, Y, [and, [app, X, Y], [app, [dn, teetotal], Y]]]]]).
lex([tenmilliondollars], iL n(t(s(n))), tenmilliondollars).
lex([than], iL(cp(than)/iL s(f)), [lmd, X, X]).
lex([that], iL(n iu (ab ab (cn(n)bs cn(n))/iL((br n(t(n))iac ue iL n(t(n)))bs s(f)))), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [app, X, Z]]]]]).
lex([that], iL(cp(that)/iL s(f)), [lmd, X, X]).
%lex([that], (cn(G)bs cn(G))/(s(f)/n(t(G))), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [app, X, Z]]]]]).
lex([the], iL(n iu (n(t(n))/cn(n))), iota).
lex([the, cold, shoulder], iL w([the, cold, shoulder]), 0).
lex([there], iL w([there]), 0).
lex([thinks], nL ((br g ie n(t(s(g))) bs s(f))/(cp(that)iad nL s(f))), [up, [lmd, X, [lmd, Y, [app, 'Pres', [app, [app, [dn, think], X], Y]]]]]).
lex([to], iL((pp(to)/a ie n(a))iac(n iu ((br n(n) bs s(i))/(br n(n) bs s(b))))), [lmd, X, X]).
lex([today], nL a iu f iu ((br n(a)bs s(f))bs(br n(a)bs s(f))), [up, [lmd, X, [lmd, Y, [app, [dn, today], [app, X, Y]]]]]).
lex([tries], nL ((br g ie n(t(s(g))) bs s(f))/nL (br g ie n(t(s(g))) bs s(i))), [up, [lmd, X, [lmd, Y, [app, [app, [dn, tries], [up, [app, [dn, X], Y]]], Y]]]]).
lex([unicorn], nL cn(s(n)), unicorn).
lex([up], iL w([up]), 0).
lex([upon], nL ((b iu f iu ((br n(b) bs s(f))bs(br n(b) bs s(f)))&g iu (cn(g) bs cn(g)))/a ie n(a)), [up, [lmd, X, [pair, [app, [dn, uponadv], X], [app, [dn, uponadn], X]]]]).
lex([void], nL g iu (cn(g)/cn(g)), void).
lex([walk], nL (br ((a ie n(a)) - (g ie n(t(s(g))))) bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, walk], X]]]]).
lex([walk], nL (br a ie n(a)bs s(b)), [up, [lmd, X, [app, [dn, walk], X]]]).
lex([walks], nL (br g ie n(t(s(g)))bs s(f)), [up, [lmd, X, [app, 'Pres', [app, [dn, walk], X]]]]).
lex([was], iL((br g ie n(t(s(g)))bs s(f))/((a ie n(a))+(g ie ((cn(g)/cn(g))iad(cn(g) bs cn(g)))-i))), [lmd, X, [lmd, Y, [app, 'Past', [case, X, Z, [eq, Y, Z], W, [app, [app, W, [lmd, U, [eq, U, Y]]], Y]]]]]).
lex([was], nL ((br w([there]) liu s(f))/a ie n(a)), [up, [lmd, X, [app, 'Past', [app, [dn, be], X]]]]).
lex([waters], nL cn(p(n)), waters).
lex([which], iL(n iu m iu ((n(t(n))ci n(t(m)))in (ab ab (cn(m)bs cn(m))/iL((br n(t(n))iac ue iL n(t(n)))bs s(f))))), [lmd, W, [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [app, X, [app, W, Z]]]]]]]).
lex([who], iL (h iu n iu (ab ab (n(t(n))bs((s(h) ci n(t(n)))in s(h)))/iL((br n(t(n))iac ue iL n(t(n)))bs s(f)))), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, X, Y], [app, Z, Y]]]]]).
lex([will], iL a iu((br n(a)bs s(f))/(br n(a)bs s(b))), [lmd, X, [lmd, Y, [app, 'Fut', [app, X, Y]]]]).
lex([without], nL (g iu (cn(g) bs cn(g))/ a ie n(a)), [up, [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [not, [app, [app, [dn, with], X], Z]]]]]]]).
lex([without], iL a iu f iu(ab((br n(a)bs s(f))bs(br n(a)bs s(f)))/(br n(a) bs s(psp))), [lmd, X, [lmd, Y, [lmd, Z, [and, [app, Y, Z], [not, [app, X, Z]]]]]]).
lex([woman], nL cn(s(f)), woman).
lex([yesterday], nL a iu f iu ((br n(a)bs s(f))bs(br n(a)bs s(f))), [up, [lmd, X, [lmd, Y, [app, [dn, yesterday], [app, X, Y]]]]]).

onecoord(A) :- member(A, []).
```

Chapter 6

LaTeX output

Consider consulting `l2.pl`, `examples.pl` and `lexicon.pl` and then executing the queries:

```
?- pplexlatex, tt(_).
```

Invoking LaTeX on out.tex yields here:

007 : $\blacksquare\forall g N t(s(g)) : 007$
a : $\blacksquare\forall g(\forall f((S f^{\uparrow}\blacksquare N t(s(g)))\backslash S f)/CNs(g)) : \lambda A\lambda B\exists C[(A\ C)\wedge(B\ C)]$
admire : $\square(((\diamond\exists a Na - \exists g N t(s(g)))\backslash S f)/\exists a Na) : \hat{}\lambda A\lambda B(Pres\ (\check{}admire\ A)\ B))$
And : $\blacksquare\forall f(S f/S f) : \lambda A A$
and : $\blacksquare\forall f((?\blacksquare S f)\square^{-1}\square^{-1}S f)/(\blacksquare S f) : (\Phi^{r+}\ 0\ and)$
and : $\blacksquare\forall a\forall f((?\blacksquare(\diamond Na\backslash S f)\backslash\square^{-1}\square^{-1}(\diamond Na\backslash S f))/\blacksquare(\diamond Na\backslash S f)) : (\Phi^{r+}\ (s\ 0)\ and)$
and : $\blacksquare\forall a\forall f((?\blacksquare(S f/!Na))\backslash\square^{-1}\square^{-1}(S f/!Na))/\blacksquare(S f/!Na)) : (\Phi^{r+}\ (s\ 0)\ and)$
and : $\blacksquare\forall f((?\blacksquare(S f/\exists a Na))\backslash\square^{-1}\square^{-1}(S f/\exists a Na))/\blacksquare(S f/\exists a Na)) : (\Phi^{r+}\ (s\ 0)\ and)$
and : $\blacksquare\forall w\forall a\forall b\forall f((\blacksquare((S f^{\uparrow}((((\diamond Na\backslash S f)\circ - Wa v)/Nb))\overset{\wp}{\_}Wa v))\backslash\square^{-1}\square^{-1}(S f^{\uparrow}((((\diamond Na\backslash S f)\circ - Wa v)/Nb))\overset{\wp}{\_}Ww))/\check{}\blacksquare((S f^{\uparrow}((((\diamond Na\backslash S f)\circ - Ww)/Nb))\overset{\wp}{\_}Ww)) : \lambda A\lambda B\lambda C[(B\ C)\wedge(A\ C)]$
and : $\blacksquare\forall f\forall a((?\blacksquare(((\diamond Na\backslash S f)/\exists bNb)\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/\exists bNb))/\blacksquare((\diamond Na\backslash S f)/\exists bNb)) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
and : $\blacksquare\forall f\forall a((?\blacksquare(((\diamond Na\backslash S f)/\exists bNb)\backslash(\diamond Na\backslash S f))\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/\exists bNb)\backslash(\diamond Na\backslash S f)))/\blacksquare(((\diamond Na\backslash S f)/\exists bNb)\backslash(\diamond Na\backslash S f))) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
and : $\blacksquare\forall f\forall a((?\blacksquare(((\diamond Na\backslash S f)/(\exists bNb\oplus\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))))\backslash(\diamond Na\backslash S f))\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/(\exists bNb\oplus\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))))\backslash(\diamond Na\backslash S f)))/\blacksquare(((\diamond Na\backslash S f)/(\exists bNb\oplus\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))))\backslash(\diamond Na\backslash S f))) : \lambda A\lambda B\lambda C\lambda D[((B\ C)\ D)\wedge((A\ C)\ D)]$
and : $\blacksquare\forall a\forall b\forall f((?\blacksquare(((\diamond Na\backslash S f)/(\exists cNc\oplus CPb)))\backslash((\diamond Na\backslash S f)/(\exists cNc\oplus CPb)))\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/(\exists cNc\oplus CPb)))\backslash((\diamond Na\backslash S f)))/\blacksquare(((\diamond Na\backslash S f)/(\exists cNc\oplus CPb)))\backslash((\diamond Na\backslash S f))) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
and : $\blacksquare\forall a\forall b\forall f((?\blacksquare(((\diamond Na\backslash S f)/PPb)\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/PPb))/\blacksquare(((\diamond Na\backslash S f)/PPb)) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
and : $\blacksquare\forall a\forall b\forall f((?\blacksquare(((\diamond Na\backslash S f)/(\exists cNc\bullet PPb)))\backslash((\diamond Na\backslash S f)))\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/(\exists cNc\bullet PPb)))\backslash((\diamond Na\backslash S f)))/\blacksquare(((\diamond Na\backslash S f)/(\exists cNc\bullet PPb)))\backslash((\diamond Na\backslash S f))) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
and : $\blacksquare\forall a\forall b\forall f((?\blacksquare(((\diamond Na\backslash S f)/!Nb)\backslash\square^{-1}\square^{-1}(((\diamond Na\backslash S f)/!Nb))/\blacksquare(((\diamond Na\backslash S f)/!Nb)) : (\Phi^{r+}\ (s\ (s\ 0))\ and)$
ate : $\square(((\diamond\exists aNa\backslash S f)/\exists aNa) : \hat{}\lambda A\lambda B(Past\ (\check{}eat\ A)\ B))$
bagels : $\square(Nt(p(n))\&CNp(n)) : \hat{}(gen\ \check{}bagels),\ \check{}bagels)$
barn : $\square CNs(n) : barn$
be : $\square(((\diamond W[there]\multimap Sb)/\exists aNa) : \hat{}\lambda A(\check{}be\ A)$
before : $\blacksquare(\forall a\forall f(((\diamond Na\backslash S f)\backslash((\diamond Na\backslash S f))/S f) : \lambda A\lambda B\lambda C((before\ A)\ (B\ C))$
beginning : $\square CNs(n) : beginning$
believes : $\square(((\diamond\exists gNt(s(g)))\backslash S f)/(CPthat\sqcup\square S f)) : \lambda A\lambda B(Pres\ (\check{}believe\ A)\ B))$
bill : $\blacksquare Nt(s(m)) : b$
bond : $\blacksquare Nt(s(m)) : b$
book : $\square CNs(n) : book$
bought : $\square(((\diamond\exists aNa\backslash S f)/(\exists aNa\bullet\exists aNa)) : \hat{}\lambda A\lambda B(Past\ (((\check{}buy\ \pi_1 A)\ \pi_2 A)\ B))$
bought : $\square(((\diamond\exists aNa\backslash S f)/\exists aNa) : \hat{}\lambda A\lambda B(Past\ (\check{}buy\ A)\ B))$
by : $\blacksquare\forall a(((\diamond Na\backslash S -))((\diamond Na\backslash S -))/(Na) : \lambda A\lambda B\lambda C[[C = A]\wedge(B\ C)]$
by : $\square(\forall n(CNn\backslash CNn)/\exists aNa) : \hat{}\lambda A\lambda B(\check{}by\ A)\ B)$
buys : $\square(((\diamond\exists gNt(s(g)))\backslash S f)/(\exists aNa\bullet\exists aNa)) : \hat{}\lambda A\lambda B(Pres\ (((\check{}buy\ \pi_1 A)\ \pi_2 A)\ B))$
calls : $\square(((\diamond\exists gNt(s(g)))\backslash S f)/\exists a((W[up]\blacklozenge Na)\sqcup(Na\blacklozenge W[up])))) : \hat{}\lambda A\lambda B(\check{}phone\ A)\ B)$
catch : $\square(((\diamond\exists aNa\backslash Sb)/\exists aNa) : \hat{}\lambda A\lambda B(\check{}catch\ A)\ B)$
cezanne : $\blacksquare Nt(s(m)) : c$
cd : $\square CNs(n) : cd$
charles : $\blacksquare Nt(s(m)) : c$
clark : $\blacksquare\forall gNt(s(g)) : c$
coffee : $\square(Nt(s(n))\&CNs(n)) : \hat{}((gen\ \check{}coffee),\ \check{}coffee)$
created : $\square(((\diamond\exists aNa\backslash S f)/\exists aNa) : \hat{}\lambda A\lambda B(Past\ (\check{}create\ A)\ B))$
darkness : $\square(CNs(n)\&Nt(s(n))) : \hat{}(\check{}darkness,\ (gen\ \check{}darkness))$
deep : $\square CNs(n) : deep$
did : $\blacksquare\forall a\forall b\forall h((((\diamond Na\backslash Sg)\uparrow(\diamond Nb\backslash Sh))/(\exists c(\diamond Nc\backslash S f))\backslash(((\diamond Na\backslash Sg)\uparrow(\diamond Nb\backslash Sh)))) : \lambda A\lambda B((A\ B)\ B)$
did+too : $(((\diamond NA\backslash SB)\uparrow(\diamond NC\backslash SD))/((\diamond NE\backslash SF))\backslash(((\diamond NG\backslash SH)\uparrow(NI\backslash SJ))) : \lambda K\lambda L((K\ L)\ L)$
doesnt : $\blacksquare\forall g\forall a((Sg^{\uparrow}((\diamond Na\backslash S f)/((\diamond Na\backslash Sb))))\backslash Sg) : \lambda A\neg(A\ \lambda B\lambda C(B\ C))$
dog : $\square CNs(n) : dog$
donuts : $\square(Nt(p(n))\&CNp(n)) : \hat{}((gen\ \check{}donuts),\ \check{}donuts)$
earth : $\square CNs(n) : earth$
eat : $\square(((\diamond\exists aNa\backslash Sb)/\exists aNa) : \hat{}\lambda A\lambda B(\check{}eat\ A)\ B)$
edinburgh : $\blacksquare Nt(s(n)) : e$
editor : $\square(\forall gCNs(g)/PPof) : editor$
every : $\blacksquare\forall g(\forall f((S f^{\uparrow}Nt(s(g)))\backslash S f)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C)\rightarrow(B\ C)]$

everyone : $\Box\forall f((Sf\uparrow\forall gNt(g))\downarrow Sf)$: $\hat{}\lambda A\forall B[(\check{}person\ B) \rightarrow (A\ B)]$
face : $\Box CNs(n)$: *face*
fell : $\Box(\exists a\langle\rangle Na\backslash Sf)$: $\hat{}\lambda A(Past\ (\check{}fall\ A))$
filed : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Past\ ((\check{}file\ A)\ B))$
finds : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}find\ A)\ B))$
fish : $\Box CNs(n)$: *fish*
for : $\blacksquare(PPfor/\exists aNa)$: λAA
form : $\Box(CNs(n)\&Nt(s(n)))$: $\hat{}(\check{}form, (gen\ \check{}form))$
fortunately : $\Box\forall f(\check{}Sf\downarrow Sf)$: *fortunately*
friends : $\Box(CNp/PPof)$: *friends*
from : $\Box((\forall a\forall f((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf))\&\forall n(CNn\backslash CNn))/\exists bNb)$: $\hat{}\lambda A((\check{}fromadv\ A), (\check{}fromadn\ A))$
gave : $\Box((\langle\rangle\exists aNa\backslash Sf)/(\exists bNb\bullet PPto))$: $\hat{}\lambda A\lambda B(Past\ (((\check{}give\ \pi_2 A)\ \pi_1 A)\ B))$
gave : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/(\exists aNa\blacktriangleright W[the, cold, shoulder]))$: $\hat{}\lambda A\lambda B(Past\ ((\check{}shun\ A)\ B))$
gave : $\Box(((\langle\rangle\exists aNa\backslash Sf)/\exists aNa)/\exists aNa)$: $\hat{}\lambda A\lambda B\lambda C(Past\ (((\check{}give\ A)\ B)\ C))$
girl : $\Box CNs(f)$: *girl*
gives : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/(\exists aNa\blacktriangleright W[the, cold, shoulder]))$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}shun\ A)\ B))$
God : $\blacksquare Nt(s(m))$: *God*
good : $\Box\forall n(CNn/CNn)$: *good*
has : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}have\ A)\ B))$
he : $\blacksquare[]^{-1}\forall g((\blacksquare Sg|\blacksquare Nt(s(m)))/(\langle\rangle Nt(s(m))\backslash Sg))$: λAA
heaven : $\Box CNs(n)$: *heaven*
her : $\blacksquare\forall g\forall a(((\langle\rangle Na\backslash Sg)\uparrow\blacksquare Nt(s(f)))\downarrow(\blacksquare(\langle\rangle Na\backslash Sg)|\blacksquare Nt(s(f))))$: λAA
himself : $\blacksquare\forall f((((\langle\rangle Nt(s(m))\backslash Sf)\uparrow Nt(s(m)))\downarrow(\langle\rangle Nt(s(m))\backslash Sf))$: $\lambda A\lambda B((A\ B)\ B)$
horse : $\Box CNs(n)$: *horse*
in : $\Box(\forall a\forall f((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf))/\exists aNa)$: $\hat{}\lambda A\lambda B\lambda C((\check{}in\ A)\ (B\ C))$
in : $\Box(\forall f(Sf\div Sf)/\exists aNa)$: *in*
is : $\blacksquare((\langle\rangle\exists gNt(s(g))\backslash Sf)/(\exists aNa\oplus(\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))-I)))$: $\lambda A\lambda B(Pres\ (A \rightarrow C.[B = C]; D.((D\ \lambda E[E = B])\ B)))$
it : $\blacksquare W[it]$: *0*
it : $\blacksquare\forall f\forall a((((\langle\rangle Na\backslash Sf)\uparrow\blacksquare Nt(s(n)))\downarrow(\blacksquare(\langle\rangle Na\backslash Sf)|\blacksquare Nt(s(n))))$: λAA
it : $\blacksquare[]^{-1}\forall f((\blacksquare Sf|\blacksquare Nt(s(n)))/(\langle\rangle Nt(s(n))\backslash Sf))$: λAA
jogs : $\Box(\langle\rangle\exists gNt(s(g))\backslash Sf)$: $\hat{}\lambda A(Pres\ (\check{}jog\ A))$
john : $\blacksquare Nt(s(m))$: *j*
laughs : $\Box(\langle\rangle\exists gNt(s(g))\backslash Sf)$: $\hat{}\lambda A(Pres\ (\check{}laugh\ A))$
left : $\Box(\langle\rangle\exists gNt(s(g))\backslash Sf)$: $\hat{}\lambda A(Pres\ (\check{}leave\ A))$
let : $\Box(Sim/Sb)$: *let*
light : $\Box(CNs(n)\&Nt(s(n)))$: $\hat{}(\check{}light, (gen\ \check{}light))$
likes : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}like\ A)\ B))$
logic : $\Box(Nt(s(n))\&CNs(n))$: $\hat{}((gen\ \check{}logic), \check{}logic)$
london : $\blacksquare Nt(s(n))$: *l*
loses : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}lose\ A)\ B))$
love : $\Box((\langle\rangle\exists aNa\backslash Sb)/\exists aNa)$: $\hat{}\lambda A\lambda B((\check{}love\ A)\ B)$
loved : $\Box\forall a\forall b((((\langle\rangle Na\backslash S-)\uparrow Nb)\circledcirc((((\langle\rangle Na\backslash S-)\uparrow Nb)\downarrow\forall g(CNg\backslash CNg)))$: $\hat{}(\check{}love, \lambda A\lambda B\lambda C[(B\ C) \wedge \exists D((A\ C)\ D)])$
loves : $\Box((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Pres\ ((\check{}love\ A)\ B))$
man : $\Box CNs(m)$: *man*
mary : $\blacksquare Nt(s(f))$: *m*
met : $\Box((\langle\rangle\exists aNa\backslash Sf)/\exists aNa)$: $\hat{}\lambda A\lambda B(Past\ ((\check{}meet\ A)\ B))$
more : $\blacksquare\forall h\forall g\forall f((Sf\uparrow(((Sh\uparrow Nt(p(g)))\downarrow Sh)/CNp(g)))\downarrow(Sf/\hat{}(CPthan\uparrow\blacksquare(((Sh\uparrow Nt(p(g)))\downarrow Sh)/CNp(g)))))$: $\lambda A\lambda B[|\lambda C(A\ \lambda D\lambda E[(D\ C) \wedge (E\ C)])| > |\lambda F\check{}(B\ \lambda G\lambda H[(G\ F) \wedge (H\ F)])|]$
mountain : $\Box CNs(n)$: *mountain*
moved : $\Box(\langle\rangle\exists aNa\backslash Sf)$: $\hat{}\lambda A(Past\ (\check{}move\ A))$
necessarily : $\blacksquare(SA/\Box SA)$: *Nec*
of : $\Box((\forall n(CNn\backslash CNn)/\blacksquare\exists bNb)\&(PPof/\exists aNa))$: $\hat{}(\check{}of, \lambda AA)$
or : $\blacksquare\forall f((?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf)$: $(\Phi^{n+}\ 0\ or)$

or : $\blacksquare\forall a\forall f((?\blacksquare(\langle\rangle Na\backslash Sf)\backslash\square^{-1}\square^{-1}(\langle\rangle Na\backslash Sf))/\blacksquare(\langle\rangle Na\backslash Sf)) : (\Phi^{n+}\ (s\ 0)\ or)$
or : $\blacksquare\forall f((?\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))\backslash\square^{-1}\square^{-1}(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf)))/\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))) : (\Phi^{n+}\ (s\ 0)\ or)$
or : $\blacksquare\forall a\forall f((?\blacksquare(((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)\backslash\square^{-1}\square^{-1}(((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb))/\blacksquare(((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)) : (\Phi^{n+}\ (s\ (s\ (s\ 0)))\ or)$
painting : $\square(CNs(n)/PPof) : \hat{}\lambda A(\check{}of\ A)\ \check{}painting)$
paper : $\square CNs(n) : paper$
park : $\square CNs(n) : park$
past : $\square\forall a\forall f(((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf))/\exists bNb) : \hat{}\lambda A\lambda B\lambda C(\check{}past\ A)\ (B\ C))$
perseverance : $\square(Nt(s(n))\&CNs(n)) : \hat{}(\check{}gen\ \check{}perseverance)\ \check{}perseverance)$
peter : $\blacksquare Nt(s(m)) : p$
phonetics : $\square(Nt(s(n))\&CNs(n)) : \hat{}(\check{}gen\ \check{}phonetics)\ \check{}phonetics)$
praises : $\square((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ (\check{}praise\ A)\ B))$
raced : $\square(\langle\rangle\exists aNa\backslash Sf) : \hat{}\lambda A(Past\ (\check{}race\ A))$
raced : $\square\forall a\forall b(((\langle\rangle Na\backslash S-)\uparrow Nb)\circledcirc(((\langle\rangle Na\backslash S-)\uparrow Nb)\downarrow\forall g(CNg\backslash CNg))) : \hat{}(\check{}race2, \lambda A\lambda B\lambda C[(B\ C) \wedge \exists D((A\ C)\ D)])$
rains : $\square(\langle\rangle W[it]\multimap Sf) : \hat{}(Pres\ \check{}itrains)$
reading : $\square(((\exists aNa\backslash Spsp)/\exists aNa) : \hat{}\lambda A\lambda B(\check{}read\ A)\ B)$
robin : $\blacksquare\forall gNt(s(g)) : r$
said : $\square((\langle\rangle\exists aNa\backslash Sf)/Sim) : \hat{}\lambda A\lambda B(Past\ (\check{}say\ A)\ B))$
saw : $\square((\langle\rangle\exists aNa\backslash Sf)/(\exists aNa\oplus CPthat)) : \hat{}\lambda A\lambda B(Past\ ((A \rightarrow C(\check{}see\ C); D, (\check{}see\ D))\ B))$
seeks : $\square(((\langle\rangle\exists gNt(s(g))\backslash Sf)/\square\forall a\forall f(((Na\backslash Sf)/\exists bNb)\backslash(Na\backslash Sf)) : \hat{}\lambda A\lambda B((\check{}tries\ \hat{}((\check{}A\ \check{}find)\ B))\ B)$
sees : $\square(((\exists gNt(s(g))\backslash Sf)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ ((\check{}see\ A)\ B))$
sent : $\square(((\exists aNa\backslash Sf)/(\exists bNb\bullet PPto)) : \hat{}\lambda A\lambda B(Past\ (((\check{}sent\ \pi_2 A)\ \pi_1 A)\ B))$
sent : $\square((((\exists aNa\backslash Sf)/\exists aNa)/\exists aNa) : \hat{}\lambda A\lambda B\lambda C(Past\ (((\check{}send\ A)\ B)\ C))$
she : $\blacksquare[]^{-1}\forall g(\blacksquare Sg\blacksquare Nt(s(f)))/(\langle\rangle Nt(s(f)))\backslash Sg)) : \lambda AA$
sings : $\square(\langle\rangle\exists gNt(s(g)))\backslash Sf) : \hat{}\lambda A(Pres\ (\check{}sing\ A))$
slept : $\square(\langle\rangle\exists gNt(s(g)))\backslash Sf) : \hat{}\lambda A(Past\ (\check{}sleep\ A))$
slowly : $\square\forall a\forall f(\square(\langle\rangle Na\backslash Sf)\backslash(\langle\rangle\square Na\backslash Sf)) : \hat{}\lambda A\lambda B(\check{}slowly\ \hat{}(\check{}A\ \check{}B))$
sneezed : $\square(\langle\rangle\exists gNt(s(g)))\backslash Sf) : \hat{}\lambda A(Past\ (\check{}sneeze\ A))$
sold : $\square(((\exists aNa\backslash Sf)/(\exists bNb\bullet PPfor)) : \hat{}\lambda A\lambda B(Past\ ((((\check{}sell\ \pi_2 A)\ \pi_1 A)\ B))$
someone : $\square\forall f((Sf\uparrow\blacksquare\forall gNt(g))\downarrow Sf) : \hat{}\lambda A\exists B[(\check{}person\ B) \wedge (A\ B)]$
Spirit : $\square CNs(m) : Spirit$
studies : $\square(((\langle\rangle\exists gNt(s(g))\backslash Sf)/\exists aNa) : \hat{}\lambda A\lambda B(Pres\ ((\check{}study\ A)\ B))$
such+that : $\blacksquare\forall n((CNn\backslash CNn)/(Sf|\blacksquare Nt(n))) : \lambda A\lambda B\lambda C[(B\ C) \wedge (A\ C)]$
suzy : $\blacksquare Nt(s(f)) : s$
talks : $\square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda A(Pres\ (\check{}talk\ A))$
tall : $\square\forall g(CNg/CNg) : tall$
teetotal : $\square\forall n(CNn/CNn) : \hat{}\lambda A\lambda B[(A\ B) \wedge (\check{}teetotal\ B)]$
tenmilliondollars : $\square Nt(s(n)) : tenmilliondollars$
than : $\blacksquare(CPthan/\square Sf) : \lambda AA$
that : $\blacksquare(CPthat/\square Sf) : \lambda AA$
that : $\blacksquare\forall n([]^{-1}[]^{-1}(CNn\backslash CNn)/\blacksquare(((\langle\rangle Nt(n)\sqcap\blacksquare Nt(n))\backslash Sf)) : \lambda A\lambda B\lambda C[(B\ C) \wedge (A\ C)]$
the : $\blacksquare\forall n(Nt(n)/CNn) : \iota$
the+cold+shoulder : $\blacksquare W[the, cold, shoulder] : 0$
there : $\blacksquare W[there] : 0$
thinks : $\square(((\rangle\exists gNt(s(g)))\backslash Sf)/(CPthat\sqcup\square Sf)) : \hat{}\lambda A\lambda B(Pres\ (\check{}think\ A)\ B))$
to : $\blacksquare((PPto/\exists aNa)\sqcap\forall n(((\langle\rangle Nn\backslash Si)/(\langle\rangle Nn\backslash Sb))) : \lambda AA$
today : $\square\forall a\forall f(((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf)) : \hat{}\lambda A\lambda B(\check{}today\ (A\ B))$
tries : $\square(((\exists gNt(s(g)))\backslash Sf)/\square(\langle\rangle\exists gNt(s(g))\backslash Si)) : \hat{}\lambda A\lambda B(\check{}tries\ \hat{}(\check{}A\ B))\ B)$
unicorn : $\square CNs(n) : unicorn$
up : $\blacksquare W[up] : 0$
upon : $\square((\forall b\forall f(((\langle\rangle Nb\backslash Sf)\backslash(\langle\rangle Nb\backslash Sf))\&\forall g(CNg\backslash CNg))/\exists aNa) : \hat{}\lambda A((\check{}uponadv\ A), (\check{}uponadn\ A))$
void : $\square\forall g(CNg/CNg) : void$
walk : $\square(\langle\rangle(\exists aNa-\exists gNt(s(g)))\backslash Sf) : \hat{}\lambda A(Pres\ (\check{}walk\ A))$

walk : $\Box(\langle\rangle\exists aNa\backslash Sb) : \hat{}\lambda A(\check{}walk\ A)$
walks : $\Box(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda A(Pres\ (\check{}walk\ A))$
was : $\blacksquare((\langle\rangle\exists gNt(s(g))\backslash Sf)/(\exists aNa\oplus(\exists g((CNg/CNg)\sqcup(CNg\backslash CNg))-I))) : \lambda A\lambda B(Past\ (A \rightarrow C.[B = C]; D.((D\ \lambda E[E = B])\ B)))$
was : $\Box((\langle\rangle W[there]\multimap Sf)/\exists aNa) : \hat{}\lambda A(Past\ (\check{}be\ A))$
waters : $\Box CNp(n) : waters$
which : $\blacksquare\forall n\forall m((Nt(n)\uparrow Nt(m))\downarrow([]^{-1}[]^{-1}(CNm\backslash CNm)/\blacksquare((\langle\rangle Nt(n)\sqcap!\blacksquare Nt(n))\backslash Sf))) : \lambda A\lambda B\lambda C\lambda D[(C\ D) \wedge (B\ (A\ D))]$
who : $\blacksquare\forall h\forall n([]^{-1}[]^{-1}(Nt(n)\backslash((Sh\uparrow Nt(n))\downarrow Sh))/\blacksquare((\langle\rangle Nt(n)\sqcap!\blacksquare Nt(n))\backslash Sf)) : \lambda A\lambda B\lambda C[(A\ B) \wedge (C\ B)]$
will : $\blacksquare\forall a((\langle\rangle Na\backslash Sf)/(\langle\rangle Na\backslash Sb)) : \lambda A\lambda B(Fut\ (A\ B))$
without : $\Box(\forall g(CNg\backslash CNg)/\exists aNa) : \hat{}\lambda A\lambda B\lambda C[(B\ C) \wedge \neg((\check{}with\ A)\ C)]$
without : $\blacksquare\forall a\forall f([]^{-1}((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf))/(\langle\rangle Na\backslash Spsp)) : \lambda A\lambda B\lambda C[(B\ C) \wedge \neg(A\ C)]$
woman : $\Box CNs(f) : woman$
yesterday : $\Box\forall a\forall f((\langle\rangle Na\backslash Sf)\backslash(\langle\rangle Na\backslash Sf)) : \hat{}\lambda A\lambda B(\check{}yesterday\ (A\ B))$

(dwp(7-7))) **[john]+walks** : Sf

$[\blacksquare Nt(s(m)) : j], \Box(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda A(Pres\ (\check{}walk\ A)) \Rightarrow Sf$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\boxed{Nt(s(m))} \Rightarrow Nt(s(m))}}{\boxed{\blacksquare Nt(s(m))} \Rightarrow Nt(s(m))}\blacksquare L}{\blacksquare Nt(s(m)) \Rightarrow \boxed{\exists gNt(s(g))}}\exists R}{[\blacksquare Nt(s(m))] \Rightarrow \boxed{\langle\rangle\exists gNt(s(g))}}\langle\rangle R \qquad \dfrac{}{\boxed{Sf} \Rightarrow Sf}$$

$$\dfrac{}{[\blacksquare Nt(s(m))], \boxed{\langle\rangle\exists gNt(s(g))\backslash Sf} \Rightarrow Sf}\backslash L$$

$$\dfrac{}{[\blacksquare Nt(s(m))], \boxed{\Box(\langle\rangle\exists gNt(s(g))\backslash Sf)} \Rightarrow Sf}\Box L$$

$(Pres\ (\check{}walk\ j))$

(dwp(7-16))) **[every+man]+talks** : Sf

$[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \rightarrow (B\ C)], \Box CNs(m) : man], \Box(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}talk\ D)) \Rightarrow Sf$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Nt(s(m)) \Rightarrow Nt(s(m))}}{Nt(s(m)) \Rightarrow \boxed{\exists g Nt(s(g))}}\exists R}{[Nt(s(m))] \Rightarrow \boxed{\langle\rangle \exists g Nt(s(g))}}\langle\rangle R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}\setminus L}{[Nt(s(m))]\boxed{\langle\rangle \exists g Nt(s(g)) \setminus Sf} \Rightarrow Sf}\Box L}{[Nt(s(m))]\boxed{\Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf)} \Rightarrow Sf}\uparrow R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}\uparrow L}{[1], \Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf) \Rightarrow Sf \uparrow Nt(s(m))}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\boxed{CNs(m)} \Rightarrow CNs(m)}}{\boxed{\Box CNs(m)} \Rightarrow CNs(m)}\Box L}{\boxed{\forall f((Sf\uparrow Nt(s(m))) \downarrow Sf)/CNs(m)}, \Box CNs(m)] \Rightarrow Sf}\forall L}{[\boxed{\forall g(\forall f((Sf\uparrow Nt(s(g))) \downarrow Sf)/CNs(g))}, \Box CNs(m)] \Rightarrow Sf}\forall L}{[\blacksquare \forall g(\forall f((Sf\uparrow Nt(s(g))) \downarrow Sf)/CNs(g))}, \Box CNs(m)] \Rightarrow Sf}\blacksquare L$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[\boxed{(Sf\uparrow Nt(s(m))) \downarrow Sf}] \Rightarrow \Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf)}\Box L}{[\boxed{\forall f((Sf\uparrow Nt(s(m))) \downarrow Sf)}] \Rightarrow \Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf)}\forall L}{\boxed{\forall f((Sf\uparrow Nt(s(m))) \downarrow Sf)/CNs(m)}, \Box CNs(m)], \Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf) \Rightarrow Sf}/L$$

$$\forall C[(\text{"}man\ C) \rightarrow (Pres\ (\text{"}talk\ C))]$$

$$(\mathbf{dvp}(7\text{-}19))\ [\mathbf{the+fish}]\text{+}\mathbf{walks} : Sf$$

$$[\blacksquare \forall n(Nt(n)/CNn) : \iota, \Box CNs(n) : fish], \Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf) : \text{"}\lambda A(Pres\ (\text{"}walk\ A)) \Rightarrow Sf$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\boxed{CNs(n)} \Rightarrow CNs(n)}}{\boxed{\Box CNs(n)} \Rightarrow CNs(n)}\Box L \quad \dfrac{}{\boxed{Nt(s(n))} \Rightarrow Nt(s(n))}}{\boxed{Nt(s(n))/CNs(n)}, \Box CNs(n) \Rightarrow Nt(s(n))}/L}{\boxed{\forall n(Nt(n)/CNn)}, \Box CNs(n) \Rightarrow Nt(s(n))}\forall L}{[\blacksquare \forall n(Nt(n)/CNn)], \Box CNs(n) \Rightarrow Nt(s(n))}\blacksquare L}{\blacksquare \forall n(Nt(n)/CNn), \Box CNs(n) \Rightarrow \boxed{\exists g Nt(s(g))}}\exists R$$

$$\dfrac{[\blacksquare \forall n(Nt(n)/CNn), \Box CNs(n)] \Rightarrow \boxed{\langle\rangle \exists g Nt(s(g))} \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}\langle\rangle R}{[\blacksquare \forall n(Nt(n)/CNn), \Box CNs(n)], \boxed{\langle\rangle \exists g Nt(s(g)) \setminus Sf} \Rightarrow Sf}\setminus L$$

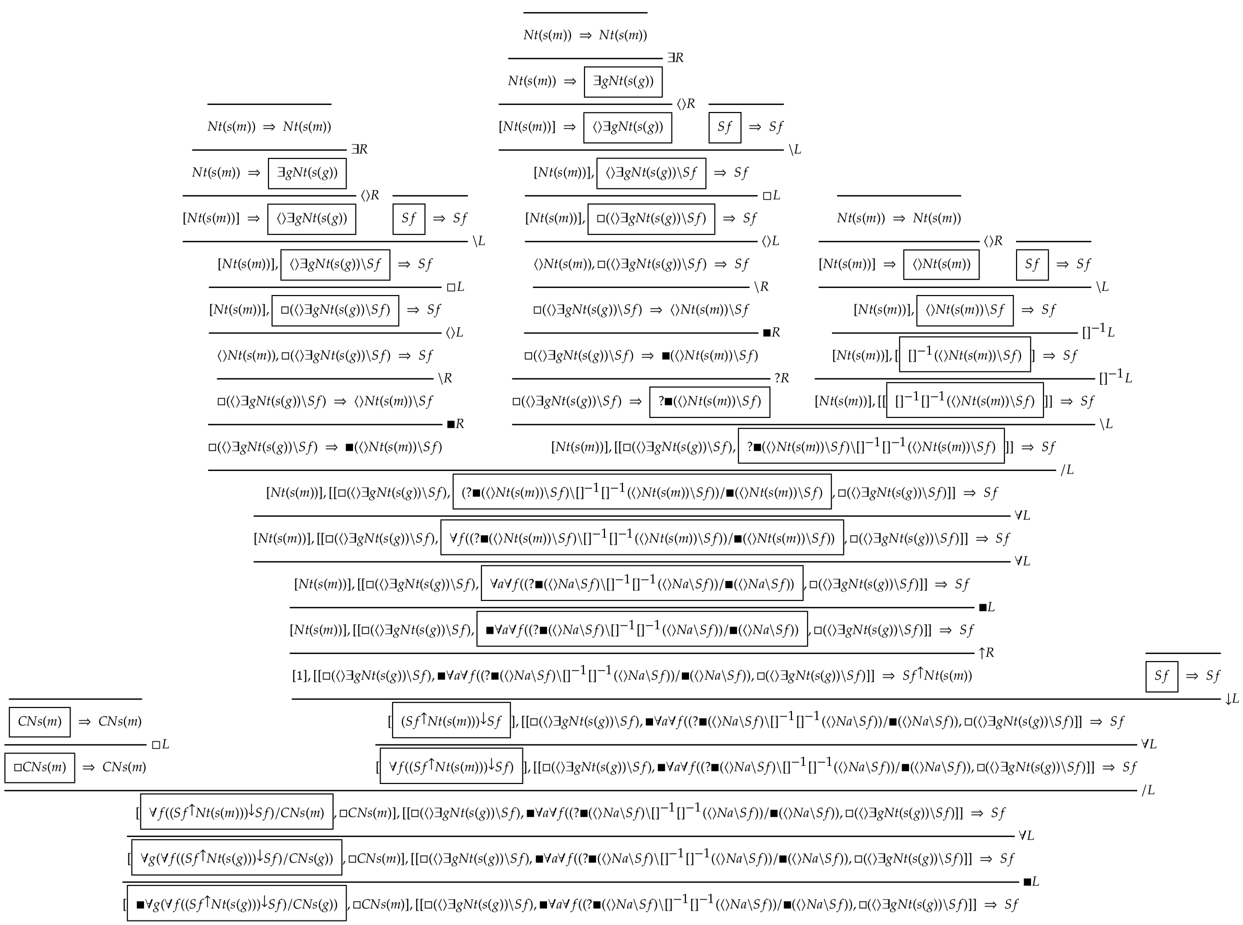
$$\dfrac{[\blacksquare \forall n(Nt(n)/CNn), \Box CNs(n)], \boxed{\Box(\langle\rangle \exists g Nt(s(g)) \setminus Sf)} \Rightarrow Sf}{[\blacksquare \forall n(Nt(n)/CNn), \Box CNs(n)], \boxed{\langle\rangle \exists g Nt(s(g)) \setminus Sf} \Rightarrow Sf}\Box L$$

(*Pres* ($\check{}$*walk* (ι $\check{}$*fish*)))

(dwp((7-32))) **[every+man]+[[walks+or+talks]]** : Sf

$[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \to (B\ C)], \square CNs(m) : man], [[\square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall f((?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf) : (\Phi^{n+}\ 0\ or), \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda E(Pres\ (\check{}talk\ E))]] \Rightarrow Sf$

$[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \to (B\ C)], \square CNs(m) : man], [[\square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall a\forall f((?\blacksquare(\langle\rangle Na\backslash Sf)\backslash[]^{-1}[]^{-1}(\langle\rangle Na\backslash Sf))/\blacksquare(\langle\rangle Na\backslash Sf)) : (\Phi^{n+}\ (s\ 0)\ or), \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda E(Pres\ (\check{}talk\ E))]] \Rightarrow Sf$



$\forall C[(\check{}man\ C) \to [(Pres\ (\check{}walk\ C)) \vee (Pres\ (\check{}talk\ C))]]$

$[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \to (B\ C)], \square CNs(m) : man], [[\square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall f((?\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))\backslash[]^{-1}[]^{-1}(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf)))/\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))) : (\Phi^{n+}\ (s\ 0)\ or), \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda E(Pres\ (\check{}talk\ E))]] \Rightarrow Sf$

$[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \to (B\ C)], \square CNs(m) : man], [[\square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall a\forall f((?\blacksquare((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)\backslash[]^{-1}[]^{-1}((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb))/\blacksquare((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)) : (\Phi^{n+}\ (s\ (s\ (s\ 0)))\ or), \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda E(Pres\ (\check{}talk\ E))]] \Rightarrow Sf$

(dwp(7-34))) $[[[\textbf{every+man}]+\textbf{walks+or+}[\textbf{every+man}]+\textbf{talks}]] : Sf$

$[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \to (B\ C)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall f((?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf) : (\Phi^{n+}\ 0\ or), [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda E\lambda F\forall G[(E\ G) \to (F\ G)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda H(Pres\ (\check{}talk\ H))]] \Rightarrow Sf$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Nt(s(m)) \Rightarrow Nt(s(m))}\exists R}{Nt(s(m)) \Rightarrow \boxed{\exists gNt(s(g))}}\langle\rangle R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[Nt(s(m))] \Rightarrow \boxed{\langle\rangle\exists gNt(s(g))}}\backslash L}{[Nt(s(m))], \boxed{\langle\rangle\exists gNt(s(g))\backslash Sf} \Rightarrow Sf}\square L}{[Nt(s(m))], \boxed{\square(\langle\rangle\exists gNt(s(g))\backslash Sf)} \Rightarrow Sf}\uparrow R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[1], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf\uparrow Nt(s(m))}\downarrow L}{[\boxed{(Sf\uparrow Nt(s(m)))\downarrow Sf}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}/L \quad \dfrac{}{\boxed{CNs(m)} \Rightarrow CNs(m)}}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)/CNs(m)}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare L}{[\boxed{\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare R}{[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow \blacksquare Sf}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Nt(s(m)) \Rightarrow Nt(s(m))}\exists R}{Nt(s(m)) \Rightarrow \boxed{\exists gNt(s(g))}}\langle\rangle R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[Nt(s(m))] \Rightarrow \boxed{\langle\rangle\exists gNt(s(g))}}\backslash L}{[Nt(s(m))], \boxed{\langle\rangle\exists gNt(s(g))\backslash Sf} \Rightarrow Sf}\square L}{[Nt(s(m))], \boxed{\square(\langle\rangle\exists gNt(s(g))\backslash Sf)} \Rightarrow Sf}\uparrow R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[1], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf\uparrow Nt(s(m))}\downarrow L}{[\boxed{(Sf\uparrow Nt(s(m)))\downarrow Sf}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}/L \quad \dfrac{}{\boxed{CNs(m)} \Rightarrow CNs(m)}}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)/CNs(m)}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare L}{[\boxed{\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare R}{[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow ?\blacksquare Sf}?R \quad \dfrac{\dfrac{}{\boxed{Sf} \Rightarrow Sf}[]^{-1}L}{[\boxed{[]^{-1}Sf}] \Rightarrow Sf}[]^{-1}L}{[[\boxed{[]^{-1}[]^{-1}Sf}]] \Rightarrow Sf}\backslash L$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Nt(s(m)) \Rightarrow Nt(s(m))}\exists R}{Nt(s(m)) \Rightarrow \boxed{\exists gNt(s(g))}}\langle\rangle R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[Nt(s(m))] \Rightarrow \boxed{\langle\rangle\exists gNt(s(g))}}\backslash L}{[Nt(s(m))], \boxed{\langle\rangle\exists gNt(s(g))\backslash Sf} \Rightarrow Sf}\square L}{[Nt(s(m))], \boxed{\square(\langle\rangle\exists gNt(s(g))\backslash Sf)} \Rightarrow Sf}\uparrow R \quad \dfrac{}{\boxed{Sf} \Rightarrow Sf}}{[1], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf\uparrow Nt(s(m))}\downarrow L}{[\boxed{(Sf\uparrow Nt(s(m)))\downarrow Sf}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)}], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}/L \quad \dfrac{}{\boxed{CNs(m)} \Rightarrow CNs(m)}}{[\boxed{\forall f((Sf\uparrow Nt(s(m)))\downarrow Sf)/CNs(m)}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\forall L}{[\boxed{\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare L}{[\boxed{\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g))}, \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow Sf}\blacksquare R}{[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) \Rightarrow \blacksquare Sf}?R \quad \dfrac{\dfrac{}{\boxed{Sf} \Rightarrow Sf}[]^{-1}L}{[\boxed{[]^{-1}Sf}] \Rightarrow Sf}[]^{-1}L}{[[\boxed{[]^{-1}[]^{-1}Sf}]] \Rightarrow Sf}\backslash L}{[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)], \boxed{?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf}]] \Rightarrow Sf}/L$$

$$\dfrac{\dfrac{[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)], \boxed{(?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf}, [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)]] \Rightarrow Sf}\forall L}{[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)], \boxed{\forall f((?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf)}, [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)]] \Rightarrow Sf}\blacksquare L$$

$$[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)], \boxed{\blacksquare\forall f((?\blacksquare Sf\backslash[]^{-1}[]^{-1}Sf)/\blacksquare Sf)}, [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)), \square CNs(m)], \square(\langle\rangle\exists gNt(s(g))\backslash Sf)]] \Rightarrow Sf$$

$[\forall H[(\check{}man\ H) \to (Pres\ (\check{}walk\ H))] \vee \forall C[(\check{}man\ C) \to (Pres\ (\check{}talk\ C))]]$

$[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \rightarrow (B\ C)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall a\forall f((?\blacksquare(\langle\rangle Na\backslash Sf)\backslash[]^{-1}[]^{-1}(\langle\rangle Na\backslash Sf))/\blacksquare(\langle\rangle Na\backslash Sf)) : (\Phi^{n+}\ (s\ 0)\ or), [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda E\lambda F\forall G[(E\ G) \rightarrow (F\ G)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda H(Pres\ (\check{}talk\ H))]] \Rightarrow Sf$

$[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \rightarrow (B\ C)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall f((?\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))\backslash[]^{-1}[]^{-1}(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf)))/\blacksquare(Sf/(\langle\rangle\exists gNt(s(g))\backslash Sf))) : (\Phi^{n+}\ (s\ 0)\ or), [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda E\lambda F\forall G[(E\ G) \rightarrow (F\ G)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda H(Pres\ (\check{}talk\ H))]] \Rightarrow Sf$

$[[[\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda A\lambda B\forall C[(A\ C) \rightarrow (B\ C)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda D(Pres\ (\check{}walk\ D)), \blacksquare\forall a\forall f((?\blacksquare((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)\backslash[]^{-1}[]^{-1}((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb))/\blacksquare((((\langle\rangle Na\backslash Sf)/\exists bNb)/\exists bNb)) : (\Phi^{n+}\ (s\ (s\ (s\ 0)))\ or), [\blacksquare\forall g(\forall f((Sf\uparrow Nt(s(g)))\downarrow Sf)/CNs(g)) : \lambda E\lambda F\forall G[(E\ G) \rightarrow (F\ G)], \square CNs(m) : man], \square(\langle\rangle\exists gNt(s(g))\backslash Sf) : \hat{}\lambda H(Pres\ (\check{}talk\ H))]] \Rightarrow Sf$

(dwp(7-39))) $[[[\mathbf{a}+\mathbf{woman}]+\mathbf{walks}+\mathbf{and}+[\mathbf{she}]+\mathbf{talks}]] : Sf$

.
.
.

Bibliography

[1] Bob Carpenter. *Type-Logical Semantics*. MIT Press, Cambridge, MA, 1997.

[2] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[3] Gerhard Jäger. *Anaphora and Type Logical Grammar*, volume 24 of *Trends in Logic – Studia Logica Library*. Springer, Dordrecht, 2005.

[4] Max Kanovich, Stepan Kuznetsov, and Andre Scedrov. Undecidability of the Lambek calculus with subexponential and bracket modalities. In R. Klasing and M. Zeitoun, editors, *21st International Symposium on Fundamentals of Computation Theory (FCT 2017), Bordeaux, France, September 11-13, 2017*, volume 10472 of *LNCS*, pages 326–340. Springer-Verlag, 2017.

[5] J. Lambek. Categorial and Categorical Grammars. In Richard T. Oehrle, Emmon Bach, and Deidre Wheeler, editors, *Categorial Grammars and Natural Language Structures*, volume 32 of *Studies in Linguistics and Philosophy*, pages 297–317. D. Reidel, Dordrecht, 1988.

[6] Michael Moortgat. *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus*. Foris, Dordrecht, 1988. PhD thesis, Universiteit van Amsterdam.

[7] Michael Moortgat. Multimodal linguistic inference. *Journal of Logic, Language and Information*, 5(3, 4):349–385, 1996. Also in *Bulletin of the IGPL*, 3(2,3):371–401, 1995.

[8] Michael Moortgat. Categorial Type Logics. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Language*, pages 93–177. Elsevier Science B.V. and the MIT Press, Amsterdam and Cambridge, Massachusetts, 1997.

[9] Richard Moot and Christian Retoré. *The Logic of Categorial Grammars: A Deductive Account of Natural Language Syntax and Semantics*. Springer, Heidelberg, 2012.

[10] Glyn Morrill. Categorial Formalisation of Relativisation: Pied Piping, Islands, and Extraction Sites. Technical Report LSI-92-23-R, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, 1992.

[11] Glyn Morrill. Parsing logical grammar: CatLog3. In Roussanka Loukanova and Kristina Liefke, editors, *Proceedings of the Workshop on Logic and Algorithms in Computational Linguistics 2017, LACompLing2017*, pages 107–131, Stockholm University, 2017. DiVA.

[12] Glyn Morrill and Oriol Valentín. Computational Coverage of TLG: Nonlinearity. In M. Kanazawa, L.S. Moss, and V. de Paiva, editors, *Proceedings of NLCS'15. Third Workshop on Natural Language and Computer Science*, volume 32 of *EPiC*, pages 51–63, Kyoto, 2015. Workshop affiliated with Automata, Languages and Programming (ICALP) and Logic in Computer Science (LICS).

[13] Glyn Morrill, Oriol Valentín, and Mario Fadda. The Displacement Calculus. *Journal of Logic, Language and Information*, 20(1):1–48, 2011.

[14] Glyn V. Morrill. *Type Logical Grammar: Categorial Logic of Signs*. Kluwer Academic Publishers, Dordrecht, 1994.

[15] Glyn V. Morrill. *Categorial Grammar: Logical Syntax, Semantics, and Processing*. Oxford University Press, New York and Oxford, 2011.

[16] Glyn V. Morrill. The CatLog2 technical manual. Manuscript, Universitat Politècnica de Catalunya, http://www.cs.upc.edu/~morrill/, 2018.

[17] R. T. Oehrle. Multi-Modal Type-Logical Grammar. In R. D. Borsley and K. Börjars, editors, *Non-transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell, Oxford, UK, 2011. doi 10.1002/9781444395037.ch6.

[18] J. van Benthem. *Language in Action: Categories, Lambdas, and Dynamic Logic*. Number 130 in Studies in Logic and the Foundations of Mathematics. North-Holland, Amsterdam, 1991. Revised student edition printed in 1995 by the MIT Press.