VERBA[1], a Multi-word-unit-oriented Feature-unification-based Parser

The recognition and proper analysis of multi-word units (henceforth mwu's) is an excellent test-bed for assessing the degree of integration between lexis and grammar that a parser is able to achieve. A number of reasons can be brought to the fore.

Mwu's are liable to exhibit any type and amount of internal grammatical structure, from full clausal skeleton (THE SHIT HIT THE FAN) down to nil (BY AND LARGE, which is an adverbial phrase, of course, but has no internal structure, since it is not made up of the conjunction of the preposition *by* and the adjective *large*).

Attempting to spot the occurrence of a mwu in running text without parsing it fully is possible thanks to a number of short cuts expressible in terms of regular expressions, but then again since the mwu is not really analysed it cannot be properly connected to the context it occurs in. More specifically, since the boundaries between mwu and free grammatical structure are not clear-cut, we must be able to assess the degree to which the candidate structure is frozen, or, to put it more positively, the extent to which it conforms to the restrictions on lexical variation and syntactic manipulation required by a mwu reading of the string.

Consider the contrast between 1 and 2:

1. She was given her due.

2. She was given his due.

1 can be recognised as a straight exponent of the mwu GIVE SOMEBODY THEIR DUE, whereas 2 is to be recognised as a pun on the mwu : it makes use of the mwu components, but flouts the important restriction expressible in terms of agreement: the indirect object and the possessive must agree along the person and gender dimensions. In order to establish this, we must be able to keep track of the indirect object in spite of the syntactic manipulations the mwu allows it to undergo, in this case promotion to subject on account of passivization. And of course we must have activated a feature check which embodies the relevant agreement pattern.

To put it in a nutshell: stylistic 'creativity' in the mwu world boils down to de-freezing: partial de-freezing, preferably, so that the mwu reading is still available, somewhere in the background for contrast. And in order to measure the degree to which the mwu structure is being de-frozen, we must be able to keep track of all the restrictions that are essential to a full, straight mwu reading of the string.

We must also make sure that we do not close the door to the recognition of exponents of the mwu which do not conform to the string the dictionaries tend to use to sum up, in skeletal form, the backbone of the mwu. Consider 3, which is a genuine example, being extracted from John Le Carré's **The Constant Gardener** (Scribner, New York and London, 2001, p. 67):

3. Spot of shit seems to have hit the proverbial fan.

In order to make sure that we have an exponent of the mwu THE SHIT HIT THE FAN, we must be able to parse *spot of* as a quantifier (along the lines of *some*, *a lot of*, etc.), to recognise the subject-to-subject raising due to *seems*, to be able to parse the remainder of the verbal group so as to recognise it as a licit exponent of the predicate *hit*, and finally to recognise the 'inserted' adjective as belonging to a restricted class of metalinguistic idiom-identifiers (*proverbial* in *the proverbial fan*, *proverbial* or *fatal* in *the proverbial/fatal bucket*). This is quite a lot to ask from a parser, but it would seem that there is no other way of making really sure that we are dealing with an exponent of

1 A first version of VERBA is described in Michiels 2006.

the mwu which dictionaries register under a string such as *the shit hit the fan*, relying on the dictionary user's knowledge of language and rhetoric to enable him to 'recover' the skeletal form.

Mwus are structures that are – partly at least – specifiable in terms of lexical units. These lexical units themselves (the components of the mwu, thus) are likely to need a treatment in which they are granted a certain degree of syntactic autonomy. As a matter of fact, in a fully integrated parser, lexical 'rules' embodying mwu structures are to be distinguished from purely 'grammatical' rules only through the fact that the lexical rules contain pointers to individual lexical units, and not only, as is the case in purely 'grammatical' rules, to broader classes specifiable in terms of their constituency structure and semantic features. In *give somethig to somebody, something* and *somebody* are fillers for noun phrases, the first fully unspecified (*something* as a filler is broader in its reference than the true indefinite *something*, which is restricted to non-humans), and the second restricted to noun phrases whose reference is to one or more human beings. In the mwu GIVE SOMEBODY THEIR DUE, the object must be specified down to the level of its lexical head, namely *due*. There is no point in building up a class that covers *due* and no other item. Of course, in our discussion of *give something to somebody, give* itself is no more than a member of a broader class, that of the verbs featuring the alternation *somebody something/something to somebody* for their argument pair indirect object/direct object. In GIVE SOMEBODY THEIR DUE, we have two lexical anchor points: *give* and *due*. Most parsers will undoubtedly prefer to start from the argument bearer, but this is a question of parsing strategy, not of the representation of the internal structure of the mwu.

A parser geared towards the treatment of mwu's must cater for the possible, indeed probable and wished-for, addition of new mwu's to its 'lexicon'. The absence of a neat division between lexis and grammar constrains the design of the parser. It must be built in such a way that new elements involving the syntactic backbone of the parser can be introduced without any component of the parser having to be redesigned.

A formalism must therefore be chosen that is both powerful enough and conspicuous enough for the lexicographer to be able to provide new structural elements (the mwu's) that can immediately be made use of by the parser (allowing for macro-expansion), as if we were dealing, in a simple parser, with the addition of, let's say, a new countable noun such as *table*.

It would seem that a feature apparatus is both sufficiently powerful and flexible, and that we can use feature unification to integrate the information carried by the newly-introduced mwu's into the whole framework.

Standardly, we define a feature as being a pair of **feature name : feature value**, where the feature name is atomic, and the feature value is one of the following:

1. a variable (in Prolog syntax, indicated by the use of an opening capital: *Nb*, *VerbForm*, ...)
2. an atom (e.g. *third*, *2*, *due, masc, ...*)
3. a list of features (e.g. *[person:third, gender:masculine, number:plural]*)

For ease of use (from the lexicographer's point of view), we should come up with a feature-unification algorithm allowing the use of the **or** and **except** operators, as in

lex:or([about,around])

where the **or** operator means that any member of the list fed as argument to **or** is a licit value (here either of the atoms *about* and *around* is a possible value for the *lex* feature).

The **except** operator allows any value for the feature except those which are members of the list it is fed as argument. This operator is not likely to be of use unless the set of possible values for the feature in question is itself restricted. The two operators can interact, as in the following feature unification call:

funify([one:except([a,b]), two:b],[one:or([b,c,a,d]), two:b],S).

which instantiates the variable passed as third argument to the unification of the two feature lists given as first and second argument, i.e. returning

S = [one:or([c,d]), two:b]

We must also, in the case of semantic features, allow the exploration of the thesauric hierarchy they are declared to belong to. Specifically, if a semantic restriction takes the form of a semantic feature, e.g. *sem:document*, feature values that are below the specified value in the hierarchy will be deemed to satisfy the semantic restriction. Therefore, if the hierarchy in which *document* is inserted, has a path in which a node *book* appears somewhere below *document*, the atom *book* will be held to satisfy the restriction set by the semantic feature specification *sem:document*.

Finally, again for greater expressibility, we allow a feature value to be a standard Prolog term, but only within the *ft* field, which is meant to house calls to commands that work across levels: we shall see the need for such calls when we discuss polarity and agreement checks that apply across levels.

Since mwu's can exhibit any amount of syntactic structure, we must be able to deal with their own argument structure. Very often, the argument list will be a mix of lexically, grammatically and semantically restricted syntactic positions, and the parser will have to make sure that all these restrictions are enforced. As an example, here is the 'lexical' rule for the mwu PRIDE ONESELF ON ONE'S X, a rule which is made use of in the parsing of the string *'Anyone who can be expected to pride themselves on their books should be asked not to write them.',* whose parse is given in Appendix B :

```
verb([v(prides, pride, prided, prided, priding, pride_oneself_on)],
mwutrprep,
arglist:[subject:[type:np, canon:0,gappable:yes, oblig:yes,
                  constraints:[sem:[hum], lex:Lex, agr:AgrSubj]],
         object:  [type:np, canon:1, gappable:no, oblig:yes,
                  constraints:[type:refp,agr:AgrRef]],
         pp_arg:[type:pp, canon:2, gappable:yes, oblig:yes,
                  constraints:[prep:on,
                               c_str:[arg_prep:[c_str:[det:[type:poss_adj, agrposs:AgrPoss]]]]]]],
ft:[pc:[agree(Lex,AgrSubj,AgrRef), prolog:constraint(AgrRef,AgrPoss)]]).
```

This lexical entry looks quite daunting at first, and a few words of explanation are in order. Beginning at the top, we first find the various verb forms that are associated with the lexeme *pride*, as well as a conventional name for the mwu (*pride_oneself_on*). We then find an atom indicating the class this mwu belongs to (multi-word-unit, transitive prepositional type), followed by the argument list. *Arglist* is here the feature name, and the value is a three-pronged list of features, one for each argument. We note that the subject is treated as an argument on exactly the same footing as the object and the prepositional phrase. The features for each argument exhibit a *constraints* feature, whose task is to set restrictions on the possible exponents of the argument. We see here the power given to the lexical rule to look down into the constituent structure (*c_structure*, *c_str* feature) of the candidate fillers for the argument position. In fact, it does not seem possible to set a boundary on the level of delicacy that must be reachable by restrictions imposed on a mwu reading of a string. Here we look down into the possessive adjective that accompanies the head of the noun phrase that builds up the argument of preposition *on* to yield the filler of the whole pp_arg slot. The value retrieved is captured in a variable that is passed on to a command-type feature value that will be in charge of checking person/gender agreement between the subject and the possessive in question. We can of course write macros that will take on the bulk of putting together the lexical rule for the entry – we do not suggest that the lexicographer should write entries like the above, but only that he should understand what happens to the entries he submits, and what he can expect the parser to be able to deal with.

Whatever the format selected for lexical rules, it stands to reason that the parser must be able to track the argument slot fillers for all the arguments that can be involved in lexical rules (this simply means all arguments, as we can put no restriction whatsoever on the type of arguments reachable by mwu-imposed constraints). VERBA must therefore prove able to deal with disruptions of the canonical argument order due to such 'transformations' (the word is quoted to avoid any theoretical stance it might still be thought to reflect) as question formation, relativization, passivization, the various types of raising operations, etc. And we must go deeper, also taking into account those transformations that involve lexis as well as grammar. For instance, we must be able to establish that *ritual* is the head of the noun phrase functioning as argument of the preposition *through* in *We knew the daily ritual she was expected to go through*, despite the disruption of word order induced by relativization (the parse is given in appendix B). But we must also be able to retrieve the personal pronoun third person singular masculine subject of the mwu GO THROUGH THE PROPER CHANNELS in *I appreciate his willingness to go through the proper channels* and be able to retrieve the proper noun Mary as subject of the mwu GO THROUGH THE MOTIONS in *I like Mary's refusal to go through the motions*, whose parse is also to be found in appendix B.

Besides, the task of parsing multi-word units is liable to lead us to provide double analyses for mwu's whose behaviour seems to conform sometimes to the first and sometimes to the second of these structural assignments. A case in point is, I think, mwus of the MAKE AN EXAMPLE OF type, whose double passivization pattern is perhaps best accounted for by a double argument analysis, one in which the whole prepositional phrase is recognized as an argument, and one in which the np is extracted and raised to full argument status, candidate for subject promotion in the passive argument structure, in order for VERBA to be able to parse both *An example was made of the teachers* and *The teachers were made an example of*, the latter's parse being given in appendix B. Here are the two VERBA entries:

Analysis A : an example was made of the teachers

```
verb([v(makes,make,made,made,making,make_an_example_of)],
    mwu_trprep,
    arglist: [subject:[type:np, canon:0,gappable:yes, oblig:yes, constraints:[sem:[hum]]],
              object:[type:np, canon:1, gappable:yes, oblig:yes,
                              constraints:[c_str:[det:[txt:an], head:[txt:example]]]],
              pp_arg:[type:pp, canon:2, gappable:yes, oblig:no, constraints:[prep:of]]],
    ft:[]).
```

Analysis B : the teachers were made an example of ; here the dangling prep is analysed as athematic (i.e. playing no role in the argument structure) and the arg_prep inside the pp_arg is raised to top-level arg status and, being gappable, is candidate for promotion to subject in passive clauses

```
verb([v(makes,make,made,made,making,make_an_example_of)],
    mwu_trprep,
    arglist:[subject:[type:np, canon:0,gappable:yes, oblig:yes,  constraints:[sem:[hum]]],
             object:[type:np,canon:1, gappable:yes, oblig:yes,
                              constraints:[c_str:[det:[txt:an],  head:[txt:example]]]],
             athematic:[type:prep, canon:2, gappable:no,  oblig:yes, constraints:[lex:of]],
             arg_prep:[type:np, canon:3, gappable:yes, oblig:yes,  constraints:[]]],
    ft:[]).
```

A mwu-geared parser such as VERBA must also cope with restrictions that at first sight might be regarded as less important than the big structural ones we have been looking at so far, but which nevertheless affect a sizeable number of mwu's, such as the restriction to non-affirmative contexts. Consider a mwu such as NOT MINCE (ONE'S) WORDS. The word *not* in the standard lexemic format is meant to embody such a restriction. In fact, a negation is not necessary at all – what matters is that

the context in which the mwu gets inserted be a non-affirmative one:

He won't mince his words.

?? *He will mince his words.*

He can't be expected to mince words. (neg-transportation)

I doubt whether he will mince words. (neg-switch ; parse in appendix B)

I appreciate his refusal to mince his words. (negation to be retrieved from *refusal*)

I know a teacher unwilling to mince his words. (negation to be retrieved from the *un-* prefix ; parse in appendix B)

It is the non-local character of what counts as context here that is the real rub. A parser such as VERBA is strictly incremental : it repeatedly goes through various passes, using the structures built by previous passes or by the very pass it is going through to build new structures, getting out of the loop only when no new production is possible. In *I doubt whether he will mince words*, *"he will mince words"* will have to be recognized as a full clause, but it will be assigned a *kill* feature that needs to get removed at some higher level (namely when the clause is absorbed as object of *doubt*). The same need to open the possibility of operating from higher levels yet to be constructed can also be exemplified with the particular agreement patterns displayed by mwu's. In the already discussed '*Anyone who can be expected to pride themselves on their books should be asked not to write them.',* the agreement pattern involves *anyone*, *themselves* and *their*. These three elements are not on the same level at all : *themselves* and *their* are within the argument structure of *pride*, and on the same clausal level. But the subject is far removed: we need first to cope with the object to subject promotion induced by passivization in the higher clause built around 'be expected to'. But besides we are within a relative clause, and the subject relative *who* must be co-indexed with the antecedent *anyone*. It is to be noted that we must go this high in order to be able to trigger the particular agreement pattern associated with the indefinite pronouns in *-body* or *-one* : *his*, *her*, *his or her*, ? *her or his*, *their* (similar gamut for *themselves*). We cannot enforce agreement without knowing the value of the lex(eme) feature of the subject.

The need to account for these specific restrictions renders the parser a bit more complex than one would have liked.

Finally, since mwu's can and do have arguments, we need to integrate into the parser a tool for assessing lexical proximity such as LEXDIS (see Michiels 2009). The reason is that very often predicate arguments are lexicographically assigned collocate lists, in order for the user to get a flavour of the type of argument filler he can expect to find in running text. As pointed out in Michiels 2009, such collocate lists are very often, along with indicators, the only type of metalinguistic information available to distinguish between word senses (in a monolingual dictionary) or target translations (in a bilingual dictionary). In *We knew the daily ritual she was expected to go through*, since, as pointed out above, the parser is able to keep track of the argument of the preposition despite the disruption caused by relativization, we can match the lemmatized head of that argument (the lexeme *ritual*) against each and every element of each and every collocate list for that argument in the various entries for GO THROUGH. In VERBA we work with six different entries for GO THROUGH (besides the entries devoted to the larger mwu's GO THROUGH SOMEBODY'S HANDS, GO THROUGH THE PROPER CHANNELS and GO THROUGH THE MOTIONS). Here is the one which will be selected on account of the quality of the match between the textual filler of the arg (*ritual*) and the collocate list, one member of which yields the best proximity factor (namely *ceremony*):

verb([v(goes,go,went,gone,going,go_through_3_perform_rehearse)],
v_mwu_prep,

arglist:[subject:[type:np, canon:0,gappable:yes, oblig:yes, constraints:[sem:[hum]]],
athematic:[type: prep, canon:2, gappable:no, oblig:yes,constraints:[lex:through]],
arg_prep:[type:np, canon:3, gappable:yes, oblig:yes,
constraints:[c_str:[head:[lex:Lex]]]],
ft:[pc:[coll(arg_prep, Lex, [marriage, initiation, scene, lesson, programme, ceremony, formality, procedure])]]).

Here the call to the *coll* procedure, to which the lexeme filling the head of the np arg of the prep is passed in the *Lex* variable, will trigger LEXDIS into action. LEXDIS will come up with two proximity factors reflecting the strength of the lexicographical links between the filler of the arg and the collocates in that particular collocate list (which numbers eight members). The first is the factor for the best match (the match *ritual/ceremony*, weight 25) and the second is the average computed over all eight matches (weight 10). This is the information to be derived from
*weight_coll:**25-ceremony-10***
which appears on the node for the object, the one featuring the np '*the ritual*'.

In the case of the longer entries featuring *go through* (such as GO THROUGH THE MOTIONS), the recognition in the string of lexical material belonging to the mwu ought to be given priority over the match with a collocate list, even if the latter should contain a single element, and the text should match it perfectly (i.e. the lexeme of the arg filler is the collocate itself), the reason being that collocates are to be interpreted not as lexical elements to be recognized as such, but as elements to be lexicographically linked, as strongly as possible, to the lexeme (sometimes the word-form, this is a whole issue not to be dealt with here) of the textual filler of the targeted argument. If it ever should be the case that the collocate list should be restricted to a single item that could not be matched in the text by a synonym or near-synonym, it is the lexicographical description of the arg bearer that should be called into question and submitted to revision, not the decision to give priority to the recognition of lexical material included in the mwu's specification.

Consequently, in *I like Mary's refusal to go through the motions*, the important thing to check is that the lexeme should be recognized as that of the mwu GO THROUGH THE MOTIONS (the *lex* feature has *go_through_the_motions* as feature value – see the parse in Appendix B).

To conclude, we wish to emphasize that the main characteristic feature of the VERBA parser is indeed the intimate mesh between grammar and lexis, which we feel essential to the proper treatment of mwu's. Mwus ARE lexical rules; lexical rules ARE grammatical rules, even if they have the particular property of featuring lexical material. The parsing process is the same for all structures, be they purely 'grammatical' or partly 'lexical'. The parser builds structure as soon as the component elements of the structure have themselves been built. We therefore start with the leaves and work our way up to the roots of the trees that can be regarded as licit parses for the string submitted to the parser. The parser does not destroy structure, but is strictly incremental. It implements a single filter acting on the candidate parses for the whole string (the S's). They should have the following properties:

1. they should be gapless : all gap positions must have found their fillers by the time the parsing process is deemed to have come to an end (recall that this point is reached only when the collection of passes has run out of new productions to record);
2. the top S should be finite;
3. the top S cannot sport a *kill* feature; the *kill* features at lower level must have been 'redeemed' (recall the mechanism for accepting such a clause as *he will mince words* as constituent of the higher *I doubt whether S*).

The parser is implemented in Prolog (SWI-Prolog, available for various platforms), as is the integrated LEXDIS tool (which calls on heavy-weight lexicographical resources – see the

Lexicographical Resources section in the References section).

References

Lexicographical resources

CIDE = *Cambridge International Dictionary of English*. Cambridge University Press, Cambridge, England, 1995
COBUILD = *The Collins COBUILD English Language Dictionary,* Second Edition *e*dited by John McH. Sinclair *et al.* London, 1995
LDOCE = *Longman Dictionary of Contemporary English* edited by Della Summers, Harlow, 1987
ODCIE = *Oxford Dictionary of Current Idiomatic English* (Vol.1: Verbs With Prepositions and Particles), edited by A.P. Cowie and R. Mackin, Oxford University Press, London, 1975
OH = Oxford/Hachette English/French pair (*The Oxford-Hachette French Dictionary French-English English-French* edited by Marie-Hélène Corréard and Valerie Grundy, Oxford University Press, Oxford, Hachette, Paris, 1994)
RC = Le Robert and Collins English/French pair (*Collins Robert French/English, English/French Dictionary*, Unabridged, Third Edition, edited by Beryl T. Atkins, Alain Duval and Rosemary C. Milne, Harper-Collins Publishers, 1993 (First ed. 1978)
WordNet = WordNet 3.0 Prolog files (see Miller 1990)

Other references

Michiels, A. 2006. '*Les lexies en TAL'*, in Bracops (M.), Dalcq (A.-E.), Goffin (I.), Jabé (A.), Louis (V.) et Van Campenhoudt (M.), éd., 2006 : *Des arbres et des mots. Hommage à Daniel Blampain*, Bruxelles, Éditions du Hazard, ISBN : 2-930154-14-4. (http://hdl.handle.net/2268/1884)

Michiels, A. 2009. '*LEXDIS, a tool for measuring lexical proximity*', Unpublished paper, University of Liège, 2009 (available as http://promethee.philo.ulg.ac.be/engdep1/download/prolog/lexdis/lexdis.pdf)

Miller, G. (1990). 'Wordnet: An on-line lexical database.' *International Journal of Lexicography (special issue)*, 3(4):235–312.

APPENDICES

A. List of multi-word units in VERBA

beat/flog a dead horse	
change/swap horses in mid-stream	
[NOT] mince (*one's*) words	non-affirmative contexts; agr with subject
[NOT] budge/move/give an inch	non-affirmative contexts
[NOT] know the first thing about	non-affirmative contexts
dig *one's* own grave	contrast with non-idiomatic dig sby's grave
kick the bucket	inclusion of adjs like *proverbial*, *fatal*
pride *oneself* on *one's* X	agreement with subject
brush aside	mobility of the particle according to end-weight
bear/carry/catch/face/take the brunt	choice of support verb
cause/create/wreak havoc on/...	choice of support verb and preposition
make havoc of	deeper frozen variant of the preceding mwu
play havoc with	id.
cock a snook at	mobility of the prep : at whom v. ....at
have in common (with)	restriction on the subject if shorter form
hold at bay	object as insertable non-idiomatic arg
hold *one's* horses	
spill the beans	
shout/scream the place/the house down	variant string realisation inside the mwu
(the) shit hits the fan	full clause mwu
horse sense, horse's ass, the horses	compounds and nps as idioms
(from) the horse's mouth	idiomatic pp built with idiomatic np
fly in the ointment	
pig in a poke	
by and large	structureless idiom (only the whole has structure)
to and fro	id.
borrow/take a leaf out of/from *someone's* book	filler for genitives and possessives
make an example of	double passive argues for double analysis
give *someone his/her* due	agreement with indirect object
go through + collocate lists for the arg of the prep	integration of the LEXDIS tool

B. S*ample parses*

1. String :

I doubt whether he will mince words.

2. WordList:

[0/i, 1/doubt, 2/whether, 3/he, 4/will, 5/mince, 6/words, endpos(7)]

3. Pretty-printed parse

```
cat:pred
voice:active
weight_coll:0
c_str
   head
      cat:vg
      pos:v
      lex:doubt
      tense:present
      voice:active
   subject
      cat:np
      sem:[hum]
      lex:i
      index:i(0, 1)
      c_str
         head
            lex:i
            sem:[hum]
   object
      cat:pred
      voice:active
      weight_coll:0
      c_str
         head
            auxgroup:[tense:present]
            prop:[mod:[will]]
            pos:v
            lex:not_mince_words
            tense:untensed
            voice:active
         subject
            cat:np
            sem:[hum]
            lex:he
            index:i(3, 4)
            c_str
               head
                  lex:he
                  sem:[hum]
         object
            cat:np
            sem:[thing]
            lex:word
            index:i(6, 7)
```

c_str
  det
    det
    zero
  head
    pos:n
    lex:word
    sem:[thing]

1. String :

Anyone who can be expected to pride themselves on their books should be asked not to write them.

2. WordList:

[0/anyone, 1/who, 2/can, 3/be, 4/expected, 5/to, 6/pride, 7/themselves, 8/on, 9/their, 10/books, 11/should, 12/be, 13/asked, 14/not, 15/to, 16/write, 17/them, endpos(18)]

3. Pretty-printed parse

```
cat:pred
voice:passive
weight_coll:0
c_str
   head
      auxgroup:[tense:past, prop:[mod:[should]]]
      prop:[voice:passive]
      pos:v
      lex:ask
      tense:untensed
      voice:passive
   subject
      cat:np
      weight_coll:0
      index:i(0, 11)
      sem:[hum]
      lex:anyone
      c_str
         head
            cat:np
            sem:[hum]
            lex:anyone
            index:i(0, 1)
            c_str
               head
                  lex:anyone
                  context:nonaff
                  sem:[hum]
         rel_clause
            index:i(0, 1)
            sem:[hum]
            weight_coll:0
            c_str
               head
                  auxgroup:[tense:present, prop:[mod:[can]]]
                  prop:[voice:passive]
                  pos:v
                  lex:expect
                  tense:untensed
                  voice:passive
               subject
                  e:i(0, 1)
               object
                  cat:pred
                  voice:active
                  weight_coll:0
                  c_str
                     head
                        auxgroup:[tense:untensed]
```

```
                pos:v
                lex:pride_oneself_on
                tense:untensed
                voice:active
             subject
                e:i(0, 1)
             object
                cat:np
                lex:themselves
                index:i(7, 8)
                c_str
                   head
                      lex:themselves
             pp_arg
                cat:pp
                prep:on
                c_str
                   head
                      lex:on
                   arg_prep
                      cat:np
                      sem:[document]
                      lex:book
                      index:i(9, 11)
                      c_str
                         det
                            pos:det
                            lex:their
                         head
                            pos:n
                            lex:book
                            sem:[document]
object
   cat:pred
   voice:active
   weight_coll:0
   c_str
      head
         auxgroup:[tense:untensed]
         pol:neg
         pos:v
         lex:write
         tense:untensed
         voice:active
      subject
         e:i(0, 11)
      object
         cat:np
         sem:[document]
         lex:them
         index:i(17, 18)
         c_str
            head
               lex:them
               sem:[document]
```

1. String :

I know a teacher unwilling to mince his words.

2. WordList:

[0/i, 1/know, 2/a, 3/teacher, 4/unwilling, 5/to, 6/mince, 7/his, 8/words, endpos(9)]

3. Pretty-printed parse

```
cat:pred
voice:active
weight_coll:0
c_str
   head
      cat:vg
      pos:v
      lex:know
      tense:present
      voice:active
   subject
      cat:np
      sem:[hum]
      lex:i
      index:i(0, 1)
      c_str
         head
            lex:i
            sem:[hum]
   object
      cat:np
      index:i(2, 9)
      sem:[hum]
      lex:teacher
      c_str
         head
            cat:np
            sem:[hum]
            lex:teacher
            index:i(2, 4)
            c_str
               det
                  pos:det
                  lex:a
               head
                  pos:n
                  lex:teacher
                  sem:[hum]
         post_mod
            subject
               e:i(2, 4)
            c_str
               head
                  cat:adjp
                  c_str
                     head
                        pos:adj
                        lex:unwilling
               pol:pos
               subject
```

```
    e:i(2, 4)
  object
    cat:pred
    voice:active
    weight_coll:0
    c_str
      head
        auxgroup:[tense:untensed]
        pos:v
        lex:not_mince_ones_words
        tense:untensed
        voice:active
      subject
        e:i(2, 4)
      object
        cat:np
        sem:[thing]
        lex:word
        index:i(7, 9)
        c_str
          det
            pos:det
            lex:his
          head
            pos:n
            lex:word
            sem:[thing]
```

1. String :

I like Mary's refusal to go through the motions.

2. WordList:

[0/i, 1/like, 2/mary, 3/'\'', 4/s, 5/refusal, 6/to, 7/go, 8/through, 9/the, 10/motions, endpos(11)]

3. Pretty-printed parse

```
cat:pred
voice:active
weight_coll:0
c_str
   head
      cat:vg
      pos:v
      lex:like
      tense:present
      voice:active
   subject
      cat:np
      sem:[hum]
      lex:i
      index:i(0, 1)
      c_str
         head
            lex:i
            sem:[hum]
   object
      cat:np
      sem:[abstract]
      lex:refusal
      index:i(2, 11)
      c_str
         det
            pos:det
            c_str
               det
                  cat:np
                  sem:[hum]
                  lex:mary
                  index:i(2, 3)
                  c_str
                     head
                        pos:n
                        lex:mary
                        sem:[hum]
         head
            pos:n
            lex:refusal
            sem:[abstract]
         args
            pol:pos
            subject
               index:i(2, 3)
               lex:mary
               sem:[hum]
            object
               cat:pred
```

```
voice:active
weight_coll:0
c_str
   head
      auxgroup:[tense:untensed]
      pos:v
      lex:go_through_the_motions
      tense:untensed
      voice:active
   subject
      e:i(2, 3)
   arg_prep
      cat:np
      sem:[abstract]
      lex:motion
      index:i(9, 11)
      c_str
         det
            pos:det
            lex:the
         head
            pos:n
            lex:motion
            sem:[abstract]
```

1. String :

We knew the daily ritual she was expected to go through.

2. WordList:

[0/we, 1/knew, 2/the, 3/daily, 4/ritual, 5/she, 6/was, 7/expected, 8/to, 9/go, 10/through, endpos(11)]

3. Pretty-printed parse

```
 cat:pred
voice:active
weight_coll:0
c_str
   head
      cat:vg
      pos:v
      lex:know
      tense:past
      voice:active
   subject
      cat:np
      sem:[hum]
      lex:we
      index:i(0, 1)
      c_str
         head
            lex:we
            sem:[hum]
   object
      cat:np
      weight_coll:25-ceremony-10
      index:i(2, 11)
      sem:[abstract]
      lex:ritual
      c_str
         head
            cat:np
            sem:[abstract]
            lex:ritual
            index:i(2, 5)
            c_str
               det
                  pos:det
                  lex:the
               adjp
                  cat:adjp
                  c_str
                     head
                        pos:adj
                        lex:daily
               head
                  pos:n
                  lex:ritual
                  sem:[abstract]
         rel_clause
            index:i(2, 5)
            sem:[abstract]
            weight_coll:0
```

```
c_str
   head
      auxgroup:[tense:past]
      prop:[voice:passive]
      pos:v
      lex:expect
      tense:untensed
      voice:passive
   subject
      cat:np
      sem:[hum]
      lex:she
      index:i(5, 6)
      c_str
         head
            lex:she
            sem:[hum]
   object
      cat:pred
      voice:active
      weight_coll:0
      c_str
         head
            auxgroup:[tense:untensed]
            pos:v
            lex:go_through_3_perform_rehearse
            tense:untensed
            voice:active
         subject
            e:i(5, 6)
         arg_prep
            e:i(2, 5)
```

1. String :

The teachers were made an example of.

2. WordList:

[0/the, 1/teachers, 2/were, 3/made, 4/an, 5/example, 6/of, endpos(7)]

3. Pretty-printed parse

```
cat:pred
voice:passive
weight_coll:0
c_str
   head
      auxgroup:[tense:past]
      prop:[voice:passive]
      pos:v
      lex:make_an_example_of
      tense:untensed
      voice:passive
   subject
      cat:np
      sem:[hum]
      lex:teacher
      index:i(0, 2)
      c_str
         det
            pos:det
            lex:the
         head
            pos:n
            lex:teacher
            sem:[hum]
   object
      cat:np
      sem:[abstract]
      lex:example
      index:i(4, 6)
      c_str
         det
            pos:det
            lex:a
         head
            pos:n
            lex:example
            sem:[abstract]
```