Proceedings of The Institute of Acoustics

SYNTAX AND MORPHOLOGY IN TEXT TO SPEECH

Jim Miller and Jim Scobbie

Centre for Speech Technology Research, University of Edinburgh

Since a major component of the text-to-speech system that is being developed in the CSTR/Edinburgh is a morph dictionary, the first task was to work out rules for morphological decomposition: that is, rules that would strip off suffixes and prefixes and allow us to identify and deal with compounds.

One important question that must be faced by anybody working on morphology is: which items are to be recognised as suffixes and prefixes and stripped? A trivial example is the suffix th. This can be recognised in words such as fifth, health, width, but some recent proposals treat these words as unsplittable, on the grounds that th is no longer productive; that is, it is no longer used to form new words in English. (The joke word coolth is excluded precisely because it is a joke word.)

For our purposes it made sense to include th in the list of strippable suffixes. It saves us from storing both cardinal and ordinal numerals such as four and fourth. Of course, fif in fifth is a nuisance, but on the other hand, if a joke word does turn up in a text, the rules can handle it. Prefixes pose nasty problems. Consider deject and de-bug. There is no doubt at all that in the second example "de" is a living prefix, which is actively used in the creation of new words and which carries secondary stress. In the first example, in contrast, the "de" does not carry any stress and is probably not perceived by many native speakers as a prefix. The fact is, however, that "deject" fits a pattern that is to be found in "inject", "abject", "reject", "conject" (as in "conjecture"), "project", "eject." These words can be divided into a prefix plus "ject", and the prefixes occur with other stems such as "ceive", "cur", "duce." Unfortunately, de-bug, e.g., is not always written with a hyphen. When the hyphen is absent, how do we tell which "de" carries secondary stress and which does not? The answer lies in distinguishing stems that are free, i.e. can stand on their own as words, and stems that are bound. i.e. cannot stand on their own as words. In "deject", the stem "ject" does not normally occur as a word, whereas in "debug" the stem "bug" can occur as a word. But what about "delight"? The stem "light" can occur as a word, but the prefix "de" carries no stress. Every language has many exceptions to the rules that even the most ingenious analyst can formulate. The traditional course is to put exceptions into dictionaries, and that is the treatment we adopted. "Delight" is stored in an exceptions dictionary and is not decomposed into a suffix and stem. To our chagrin, not all difficult prefixes can be disposed of so simply.

As a footnote to the preceding section on prefixes, it is worthwhile mentioning that there are other reasons than space-saving which lead to the recognition of unusual prefixes. For example, "af" in "affection" and "ad" in "addiction" are not usually recognised as prefixes but they are so recognised

in the text-to-speech system because this move helps to predict on which syllable stress falls.

Morphological decomposition is not a sufficient basis for an adequate speech output. In order to assign stress correctly in all words the relevant rules require information about word class: contrasting pairs such as the noun convict and the verb con"vict are well-known. Information is further required about the arrangement of words into phrases: this is essential for an acceptable intonation output. It should be noted that information about word class and information about phrase structure are closely connected; indeed, any procedures for assigning word class rest on prior knowledge of possible arrangements into phrases and the correct assignment of word class leads directly to assignment of phrase structure. The rules apply to one sentence at a time. The linguistics market place offers relatively simple and relatively complex analyses of syntax. For our purposes, a simple analysis will suffice. The central categories are the word classes Noun, Verb, Modal, Auxiliary, Adjective, Pronoun, Preposition, Conjunction, Adverb and the phrase classes Noun Phrase, Adjective Phrase, Prepositional Phrase and Verb Phrase. However, in order to write effective word class assignment rules it will be necessary to recognise smaller classes.

For example,in their Contemporary English Grammar, Quirk et al describe Noun Phrases not just with the category Determiner for words such as "this", "the", "a", "some" but with the smaller categories of Pre-, Central and Post-Determiner. Pre-Determiner includes "all", "both", "double", "three times"; Central Determiner includes "the", "my", "whose", "enough"; and Post- Determiner includes "three" etc., "first", etc., "little", "few." In order to assign word class correctly and effectively it may be necessary to set up as many as eight or nine smaller groups of determiners.

The starting point for the word class assignment rules is that morphological clues are made available by the morphological decomposition. For instance, if the word "gregarious" occurs, it is split into "greg", "ar" and "ous." The final suffix "ous" is recognised by the rules and signals that the word it belongs to is an adjective. There are several adjective suffixes, a larger number of noun suffixes and a small number of verb prefixes and suffixes. (That is, suffixes that mark a given word as belonging unequivocally to a particular class.) These morphological markers regularly provide points of orientation in a sentence.

Other points of orientation are provided by those words that belong to only one class: e.g. "theft, concede, ripe". The remaining words in a given sentence - at least, in a given clause, since sentences may consist of more than one clause - are assigned to classes by rules that exploit the points of orientation, knowledge of possible phrases in English, and knowledge of the arrangement of phrases in different types of clause.

It may be necessary to make use of information that will be recorded in dictionary entries. For example, the sentence "She drinks her coffee black" displays the regular but not frequent pattern of a Noun Phrase - "her coffee" - followed by an Adjective Phrase - in this case, the single adjective "black." "Drink" is one of a small number of verbs that allow this pattern. With that information recorded in its dictionary entry, the following "black" can be labelled 'adjective' rather than 'noun'. (It is assumed that the above construction is different from, e.g. "He shouted himself hoarse." Many verbs

can occur in this construction, but typically the object noun is a reflexive pronoun, recognisable by "self".)

Having considered the overall strategies of the morphological and syntactic aspects of our system, we will now discuss our method of syntactic analysis in greater detail, since our approach is a departure from others used in text-to-speech schemes.

In any linguistic computer system there will be at least two competing priorities for the representation of linguistic material: it will have to suit both the purposes of the linguist who writes it and the programme that uses it. Therefore, in the implementation of the syntactic module, comparable time was alotted to the design of the representations, the writing of a compiler to mediate between them, and the form of the parser that uses them. This has resulted in a nice match at all levels, so that the causes of failure can be found more quickly, and an overall understanding of the module's operation is available to all.

The module deals just with word classes, input as a PROLOG list. We interpret this list as a right context stack The operations of the parser can therefore be seen as a transfer of items from this stack to a left context stack; the operation of transferral being the disambiguation of each item.

	At Start	At End	
	[possadj]	[n]	
	[n,a]	[a]	
	[aux,v]	[art]	
	[v,n]	[v]	
	[art]	[aux]	
	[n]	[n]	
[]	[n]	[possadj]	[]
Left	Right	Left	Right

To parse, then, we lift an item from the top of the right stack, disambiguate it, and place it on the top of the left stack. That is, we split our input sequence of candidate categories at some point in the derivation thus:

[a,b,c,d] "Disambiguate here" - "e" [f,g,h,i]

To disambiguate the item "e", we need to look to its left context, say.
For fast list access, this is represented reversed, so that the first item in the left context - the top of the stack - is the head of that list.

Rules for the linguist's use should be in some canonical form:

unknown => Cat Left context____Right context
[within limits]

For the use of the parser, these are translated into a permitted left context stack and a permitted right context stack. A rule which states that an ambiguous auxiliary is to be disambiguated in favour of 'aux' if it occurs

before a verb followed by an article would have the following representations:

```
[aux,v] => 'aux' / __ 'v','art'

[aux,v]:                        'aux'        [v]
                  []                         [art]

          (Left Context)   Zero Item   (Right Context)
```

To disambiguate our candidate category, we then check the permitted right context against the actual right context. In the present case, they would match, and we would have a successful disambiguation. If the actual right context contains an ambiguous element - as with [v,n] in the example candidate string - then this must be disambiguated along with the item being transferred from right to left stack, but only within certain bounds set by the rule. The picture above is complicated by the need to create a new actual right context for the disambiguated element 'aux'. Solving this problem is just a matter of good bookkeeping.

If a rule contains optional elements then it cannot be matched in a simple one-to-one fashion. Optionality comes in two varieties: polar optionality, where either an item of permitted context is present, or wholly absent; and selective optionality, where an item of permitted context is one of a group. In linguistic notation, the first is represented by round brackets, the second by curly brackets.

The compiler takes a linguistic rule with optionality. Rather than spelling out all the various possibilities, it keeps these, and the parser must therefore be able to keep track of all the possibilities. At the moment, no permanent record, in the form of a chart for instance, is used: instead we employ a procedural approach that fits each item of permitted context to the actual context, taking account of the optionality, until success, or failure, making use of PROLOG's backtracking abilities.

Reference

Quirk R. et al. A Comprehensive Grammar of the English Language, Longman 1985.