Integrating Morphology into the LKB

ACQUILEX-II Working paper No. 17

Bernie Jones
Computer Laboratory, University of Cambridge
Bernie.Jones@cl.cam.ac.uk

August 1993

1 Introduction

The similarity between lexical processes, such as sense extension, and morphological processes, such as nominalisation, has been noticed and described in the literature for some time now [Copestake, 1992, Ch. 4] [Briscoe & Copestake, 1991]. Indeed, in relation to the ESPRIT ACQUILEX projects much has been made of this similarity, and the LKB software is ideally suited to an implementation of these phenomena through its lexical rule structure [Copestake & Briscoe, 1991]. However, until now a major problem has been the lack of a mechanism within the software to handle any form of morphographemic or morphophonemic alternations. This has also been a problem from the point of view of parsing, and associated with this the size of the lexicon, since all morphological variants of every word need a separate lexical entry if parsing is to succeed. This leads not only to reduced performance due to an over-sized lexicon, but also failure to capture the similarity in the variations between words, and the extent to which morphological variants of the same word are related.

Of course the chief difficulty in designing a morphological component to a system such as the LKB is that it is designed around unification. It would be impossible to account for any modification to the string value of the orthography feature of a feature structure within a unification-based framework, so any system that is to perform such a modification must work externally to the main LKB algorithms. In uses of the LKB to date the problem has been tackled in a rather unsatisfactory manner, due to the lack of any such external morphology system. When orthographic modifications were desired, a complex orthographic structure was created within the unification mechanism, in which the components of the complex structure corresponded to the individual morphemes. See for example the two feature structures, and the lexical rules which created them, in figures 1 and 2. These illustrate both the derivational and inflectional applications of morphology.

This paper describes the integration of an existing morphological processing system with the current LKB, enabling proper orthographic modification to take place in the full range of functions the LKB offers.

$$\left[\begin{array}{l} \textbf{lex-noun-sign} \\ \text{ORTH} = \left[\begin{array}{l}\textbf{complex-orth}\\ \text{ORTH1} = \textbf{teach}\\ \text{ORTH2} = \textbf{er}\end{array}\right] \\ \text{CAT} = \boxed{\textbf{noun-cat}} \\ \text{SEM} = \boxed{\textbf{obj-noun-formula}} \\ \text{QUALIA} = \boxed{\textbf{human}} \end{array}\right]$$

```
agentive
   lexical-rule
   < 1 > = strict-trans-sign
   < 0 > = lex-noun-sign
   < 0 : orth : orth1 > = < 1  : orth >
   < 0 : orth : orth2 > = "er"
   < 0 : qualia : telic > = < 1 : sem >
   < 0 : qualia : telic : arg2 : arg1 : arg2 > =
     < 0 : qualia : object-index >.
```

Figure 1: Application of AGENTIVE rule to *teach*

2 Morphology System

The morphological analysis and generation system used here is a much improved version of that first described in [Jones, 1991]. The system works with a data structure created by compiling a set of morphological rules based upon those in [Sanfilippo, 1990]. The structure of the rules includes features such as name, and position (suffix, prefix etc.), but most importantly a set of subrules in the form of matched simple partial regular expressions. The first of each pair of expressions will match a set of letters at the crucial site of affix attachment in the stem (word-end, word-beginning, word-middle) and the second expression describes the set of letters to replace them with to form the morphologically complex form. Thus the following might form a part of the past-tense rule in English:

```
past-tense  suffix  (* ed) (e ed) (y ied) ...
```

In this rule, the asterisk represents a null character, so that the first subrule is the default (*e*d is added at the word-final position). The rest of the sub-rules represent exceptions, so that if e.g. the root form ends with *e* then that *e* is replaced by *ed*. Since the final rules (see section 5) are structured in such a manner that the exception sub-rules override the default, this prevents the formation of morphological complexes such as *hateed*, or (since the ultimate example of an exception sub-rule is one where a particular stem has a totally non-regular complex form) *gived*.

This data is compiled into large, double rooted, double linked letter trees — one for each type of morphological attachment type (suffix, prefix etc.),

$$
\left[\begin{array}{l}
\textbf{lex-noun-sign}\\
\text{ORTH} = \left[\begin{array}{l}\textbf{complex-orth}\\ \text{ORTH1} = \textbf{rabbit}\\ \text{ORTH2} = \textbf{+s}\end{array}\right]\\
\text{CAT} = \boxed{\textbf{noun-cat}}\\
\text{SEM} = \left[\begin{array}{l}\textbf{obj-noun-formula}\\ \text{IND} = \boxed{0}\left[\begin{array}{l}\textbf{obj}\\ \text{AGR} = \left[\begin{array}{l}\textbf{agr}\\ \text{NUM} = \textbf{pl}\end{array}\right]\end{array}\right]\\ \text{PRED} = \textbf{rabbit\_L\_1\_1}\\ \text{ARG1} = \boxed{0}\\ \text{PLMOD} = \textbf{true}\\ \text{QUANT} = \textbf{false}\end{array}\right]\\
\text{QUALIA} = \boxed{\textbf{nomqualia}}
\end{array}\right]
$$

```
plural
    lexical-rule
    < 1 > = lex-count-noun
    < 1 : cat : m-feats : agr : num > = sg
    < 0 : cat : m-feats : agr : num > = pl
    < 0  > = lex-count-noun
    < 0  : orth : orth1 > = < 1  : orth >
    < 0  : orth : orth2 > = "+s"
    < 1 : sem > = < 0  : sem >
    < 1 : qualia > =  < 0 : qualia  >.
```

Figure 2: Feature structure for *rabbits*, and PLURAL rule

$$\odot \rightarrow e \rightarrow n \Longrightarrow \odot$$

$$\odot \rightarrow e \rightarrow m \rightarrow m \Longrightarrow m \rightarrow \odot$$

$$\odot \rightarrow e \rightarrow m \rightarrow b \Longrightarrow b \rightarrow \odot$$

$$\odot \rightarrow e \rightarrow m \rightarrow p \Longrightarrow p \rightarrow \odot$$

Figure 3: Some possible analysis routes through the PREFIX tree.

where the letters mark branches according to the expressions in the rule. One half of the tree stores letter patterns for the stem words, wheras the other half stores them for the morphologically complex words; in effec,t therefore, one half of the tree is for generation, and the other half is for analysis. A node in one half of a tree may mark the position that a rule is triggered at (that is, where the set of branches associated with the letters in a particular sub-rule regular expression end) and contain links to the other half of the tree. This point in the other half of the tree marks the end of the branches corresponding to the other expression of the pair in the morphological sub-rule. Since the two halves of the tree perform similar functions for both the first and last expressions in each sub-rule, starting from either root node corresponds either to generating a morphologically complex form, or analysing one down to a stem. When using this data structure the word to be processed is mapped character by character against the branches of the relevant tree (starting from either one of the root nodes, depending on whether generation or analysis is required). When a node is reached at which a rule is triggered, the remainder of the word to be processed — that portion not matched by the regular expression in its translation to the tree structure — is kept, and characters from the other half of the tree, from a pointer from the triggering node down to the other root, are attached to it. Figure 3 illustrates some routes that might be taken through the prefix tree when removing the $en(em)$ prefix.

To actually perform the morphological analysis or generation, the data trees are used repeatedly. Obviously generation is the easier operation, since it is known which rule is the appropriate one. The stem to be generated from is simply matched against the trees until a node marked with the appropriate rule is found; then the link to the other half of the relevant tree is followed and the morphologically complex form is assembled. Analysis is a slightly more challenging affair, in that it is perfectly possible for there to be not only several different possible analyses of the same word, but also for there to be multiple morphemes to strip from the word in order to obtain the stem. Consider the word *unionise*, for example. There are two possible analyses:

either UN+ION+ISE or UNION+ISE, both of which are perfectly valid[1]. Hence in the analysis software there is the facility to maintain multiple possibilities for fragmentation, and also to repeatedly analyse the resulting stems until no more breakdown is possible[2].

It is fortunate that the English language has relatively simple morphology. Morphological modification seems to involve either prefixation or suffixation rather than a combination of the two, and infixation is very rare indeed. However, other languages are far more rich. In German, for example, a single morphological step can often involve both suffix and prefix addition, and in certain cases there is also an infix change. See for example the past tense of the verb *rennen*.

```
renn-en  --->  ge-rann-t
```

For this reason it is not enough simply to specify single place morphological operations, as the original system described in [Jones, 91] did — there should be the facility to link prefix, infix and suffix operations together. The current system will permit this, and although the rules will still be categorised into one of three position types (prefix, suffix and infix), each rule can have associated modifications for the other positions. This means that the system becomes very modular and if there is some scheme that is difficult to express in the current implementation (vowel harmony might be, for example) then it is very simple indeed to add new modules to the system to account for these phenomena. Hence this morphological system is not only structurally very simple — it is also very powerful and flexible, which makes it ideal for the integration with the LKB.

3 Into the LKB

Bearing in mind the similarity between the LKB's lexical rules and the proposed morphological rules, it is likely that the morphological variant will have to be included in all the places that the lexical rules are used. The key functional locations are the following:

3.1 Parsing

It should be possible for words that are outside the LKB lexicon due to morphological complexity to be included in a phrase for parsing, and have feature structures built for these words from the lexical entries of their stem morphemes. Thus the flexibility of the parser is much enhanced, and the lexicon can be permitted to stay smaller. The effect on the final parse should be much as the effect of the application of lexical rules. That is that there should be an edge in the chart for the result and argument of each rule that

[1] outside any context, that is — with context there will be a preferred interpretation (pragmatically).

[2] since only one morpheme at a time can be removed from a word.

is applied (morphological rules included). Thus if the phrase "John sleeps" is to be parsed, both *sleep* and *sleeps* should appear in the chart.

One problem that is raised by this is the interaction between morphological and lexical rules. It is not possible, when analysing a morphologically complex word, to first apply all the morphological operations, and then apply all the lexical ones, or vice versa. Consider the following examples:

```
sandwiches       (transfer of reference - human)
    |
sandwiches       (plural)
    |
sandwich         (root)
```

```
beefs       (plural)
  |
beef        (portioning)
  |
beef        (root)
```

In the first example, the morphological process of pluralisation is applied to *sandwich*, followed by the transfer of reference rule considered by [Nunberg, 1978]. He postulated that in the context of waiters or waitresses talking in a restaurant, it would seem quite natural to refer to a customer who has ordered a ham sandwich as *the ham sandwich*. Therefore in this example the reference would be to a customer who has ordered, say, a plate of sandwiches. In the second example a lexical rule has transformed the mass term *beef* into a count noun denoting a portion of that substance; this is then pluralised.

It is therefore clear that lexical and morphological rule applications must have the potential to be totally interleaved, so that they can be applied in whichever order suits the unification constraints. This may require the lexical rules to be formulated in a rather more strict fashion to avoid overgeneration, of course.

3.2 Generation

When the menu options to apply lexical rules to psorts are chosen in the LKB menu, they should also have the ability to apply morphological rules. Thus the option to apply a particular rule should check whether that rule is morphological, and apply the appropriate orthographic transformation on the resultant feature-structure given by the unification procedure, and the function of applying all lexical rules to a psort should exhibit similar behaviour. Obviously with the latter possibility it should be possible to interleave application of morphological and lexical rules when there is the possibility of applying more than one rule to a psort.

3.3 Translation

In the same way that lexical rules are translated through the tlink rule mechanism, morphological rules should have translation equivalents in other languages so that correct translations can be obtained if a morphologically complex form is to be translated — that is, the appropriate morphological rule is applied to the translation of the stem-form of the complex word in the source language.

Therefore it seems that the best way to implement morphology in the LKB is as a form of lexical rule. Indeed, the lexical rule mechanism is admirably suited to that facet of the morphological manipulation requiring default unification — namely the specification of various features in the syntax and semantics of the stem and final forms[3]. A morphological rule must how ever be distinctively different to a lexical rule, so that it can be clearly distinguished as one. This is obviously necessary so that the extra process of orthographic modification can be implemented correctly.

4 Implementation of LKB morphology

Morphological rules are implemented in this system as a subtype of lexical rule. Thus they are similar enough to lexical rules for standard LKB unification operations to work, and yet distinctive enough for the orthographic modification to be able to operate. The morphological rules (whose structure is described in a later section) are read in by the LKB twice: once by the lexical rules mechanism, which deals with the information suitable for unification processing, and once by a mechanism that uses a different section of the rules to build the double rooted tree, described earlier, for use in the generation/analysis stages.

Operations involving generation of morphologically complex forms are accomplished in a fairly simple manner. The code is modified to include a check for a morphological rule. If such a one is found, then the unification function is called with an additional value for the new orthography; if unification fails nothing happens, but if it is successful the new orthography is inserted into the new feature structure before it is returned to the calling function.

Parsing is a little more difficult since the interleaving of lexical and morphological rules and possibility of many different stems must be taken into consideration. Morphological analysis is attempted on every word in the parse and the root morphemes and intermediate forms (through morphological or lexical modification) are added to the chart. Then if the word is also a valid lexical entry it is processed normally; if not, then normal processing is inhibited, so that the 'word unknown' report is not generated. If, however, there are no morphological possibilities for the word, then normal processing is forced, whether the word is lexical or not. This is so that the appropriate error message can be generated if the word does not occur in the lexicon and is not the morphological complex of a lexical stem.

The morphological analysis is processed one step at a time, and between each morpheme addition all possible combinations of lexical rules are tried.

[3] In the case of *plural* for instance, the argument must be a singular count noun, and the result a plural one.

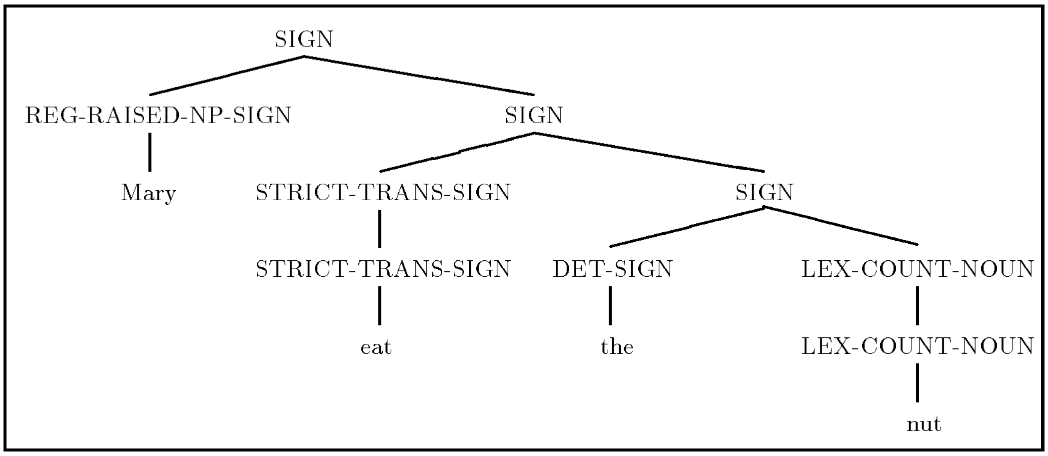

Figure 4: Parse tree for *Mary eats the nuts*

$$\begin{bmatrix} \textbf{sign} \\ \text{ORTH} = \begin{bmatrix} \textbf{complex-orth} \\ \text{ORTH1} = \textbf{mary} \\ \text{ORTH2} = \begin{bmatrix} \textbf{complex-orth} \\ \text{ORTH1} = \textbf{eats} \\ \text{ORTH2} = \begin{bmatrix} \textbf{complex-orth} \\ \text{ORTH1} = \textbf{the} \\ \text{ORTH2} = \textbf{nuts} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \text{CAT} = \boxed{\textbf{sent-cat}} \\ \text{SEM} = \boxed{\textbf{binary-formula}} \end{bmatrix}$$

Figure 5: Feature structure for *Mary eats the nuts*

All the resulting edges are stored in a temporary list, the whole of which is added to the chart if and only if all rule applications unify properly and there is a lexical entry for the stem form. This does of course cause problems, since the subordinate edges to the morphologically complex word in the parse only exist in a theoretical sense within the structure of the parse. We have to make sure that these edges cannot be picked up and spanned by rule applications higher up in the parse. To ensure that this happens, the subordinate edges are all made zero length, which means that they are not considered by the rest of the parsing mechanism, but still appear in the final output of the parse. So, for example, parsing "Mary eats the nuts" results in the parse tree shown in figure 4 and the feature structure shown in figure 5 — note the fact that in the parse tree the orthography of the stem lexeme is given at the leaf nodes. This reinforces the fact that they are a notional component of the real parse.

5 Rule structure

Since, by the description above, the same rule must be read twice by two different input mechanisms, the two different sets of information associated with each rule must each be hidden from the other. This is easily accomplished through the use of a special character, which is treated by the lexical rule input mechanism as a comment, yet indicates to the morphological data compiler that the current line contains information relevant to it. The following rule, for noun pluralisation, illustrates this.

```
plural morph-rule
 %suffix (!s !ss) (!ss !ssses) (ss sses) (!ty !ties) (f ves)
 % (tch tches) (sh shes) (person people) (man men) (formula formulae)
 % (goose geese) (fish fish) (child children) (sheep sheep)
 % (chicken chicken) (mouse mice) (louse lice)
   < 1 > = lex-count-noun
   < 1 : cat : m-feats : agr : num > = sg
   < 0 : cat : m-feats : agr : num > = pl
   < 0 > = lex-count-noun
   < 1 : sem > = < 0  : sem >
   < 1 : qualia > =  < 0 : qualia  >.
```

The portions following the % sign at the start of the line form the sub-rules relevant to the morphological compiler. The name of the rule — *plural* in this case — is extracted from the header of each rule. The first word after the header — *suffix* in this case — indicates which data structure to compile the current rule into. The *!* structure is used to denote special macro symbols internal to the morphological system. Use and justification for these is provided more fully in [Jones, 1991], but briefly they each expand to a set of letters that could possibly replace them. These sets are also defined in the file with the morphological rules, as follows:

```
%(letter-set (!s abcdefghijklmnopqrtuvwxyz))
```

They must, obviously, be defined prior to their first use in the rule file, and only one particular macro symbol may be used in each sub-rule.

Of course, this rule structure is slightly untidy, in that all the exceptions to the productive morphological processes must be listed explicitly. This has only been implemented in such a fashion for the sake of simplicity, and since the testing lexicon in the LKB is rather small. In a larger system one could envisage features for irregular morphology being included in the lexical entries for stems. Hence the presence of an irregular morphology for a particular rule would block the application of that particular rule to the stem, and instead the morphologically complex form would be taken from a feature in the lexical entry.

$$\left[\begin{array}{l} \textbf{strict-trans-sign} \\ \text{ORTH} = \textbf{enters} \\ \text{CAT} = \left[\begin{array}{l} \textbf{strict-trans-cat} \\ \text{RESULT} = \left[\begin{array}{l} \textbf{strict-intrans-cat} \\ \text{RESULT} = \boxed{\textbf{sent-cat}} \\ \text{DIRECTION} = \textbf{forward} \\ \text{ACTIVE} = \left[\begin{array}{l} \textbf{np-sign} \\ \text{ORTH} = \textbf{orth} \\ \text{CAT} = \left[\begin{array}{l} \textbf{np-cat} \\ \text{CAT-TYPE} = \textbf{np} \\ \text{M-FEATS} = \left[\begin{array}{l} \textbf{nominal-m-feats} \\ \text{REG-MORPH} = \textbf{boolean} \\ \text{AGR} = \left[\begin{array}{l} \textbf{agr} \\ \text{PERS} = \textbf{3} \\ \text{NUM} = \textbf{sg} \\ \text{GENDER} = \textbf{gender} \end{array}\right] \\ \text{NOMINAL-FORM} = \textbf{nominal-form} \\ \text{CASE} = \textbf{case} \\ \text{COUNT} = \textbf{boolean} \end{array}\right] \end{array}\right] \\ \text{SEM} = \boxed{0}\ \boxed{\textbf{p-agt-formula}} \end{array}\right] \end{array}\right] \\ \text{DIRECTION} = \textbf{backward} \\ \text{ACTIVE} = \boxed{\textbf{dir-obj-np-sign}} \end{array}\right] \\ \text{SEM} = \boxed{\textbf{strict-trans-sem}} \\ \text{SENSE-ID} = \boxed{\textbf{sem-sense-id}} \end{array}\right]$$

Figure 6: 3rd person present tense form of *enter*

$$\left[\begin{array}{l} \textbf{lex-count-noun} \\ \text{ORTH} = \textbf{lammen} \\ \text{CAT} = \left[\begin{array}{l} \textbf{noun-cat} \\ \text{CAT-TYPE} = \textbf{n} \\ \text{M-FEATS} = \left[\begin{array}{l} \textbf{nominal-m-feats} \\ \text{REG-MORPH} = \textbf{boolean} \\ \text{AGR} = \left[\begin{array}{l} \textbf{agr} \\ \text{PERS} = \textbf{person} \\ \text{NUM} = \textbf{pl} \\ \text{GENDER} = \textbf{gender} \end{array}\right] \\ \text{NOMINAL-FORM} = \textbf{nominal-form} \\ \text{CASE} = \textbf{case} \\ \text{COUNT} = \textbf{true} \end{array}\right] \end{array}\right] \\ \text{SEM} = \boxed{\textbf{unary-formula-entity-arg1}} \\ \text{SENSE-ID} = \left[\begin{array}{l} \textbf{sense-id} \\ \text{FS-ID} = \boxed{0}\ \textbf{lam\_1} \\ \text{LANGUAGE} = \textbf{dutch} \\ \text{DICTIONARY} = \textbf{string} \\ \text{LDB-ENTRY-NO} = \textbf{string} \\ \text{HOMONYM-NO} = \textbf{string} \\ \text{WORD} = \textbf{string} \\ \text{SENSE-NO} = \textbf{string} \end{array}\right] \\ \text{QUALIA} = \boxed{\textbf{animal}} \end{array}\right]$$

Figure 7: *Lambs* translated into Dutch through the TLINK-RULE mechanism

6 Conclusion

The morphological system integrates well with the LKB, and enhances its functionality. Examples with the parsing function have already been shown in the previous section, and examples of use in the other two functions amount to little more than the display of some feature structures. The structures corresponding to the application of the 3rd-person-singular rule to the verb *enter*, and the translation of the plural of *lamb* into Dutch are given in figures 6 and 7 respectively. In the latter example we are linking the English and Dutch plural formation rules trough the translation link mechanism — this seems the logical manner in which to do morphology in translation.

The system performs to specification, allowing a full range of inflectional and derivational morphological phenomena to be experimented with within the LKB. Also, as mentioned previously, the system is not difficult to extend in order to cope with different morphological phenomena that have not been catered for in the current implementation.

In the current system, there is no facility to tackle compounding phenomena, unless a specific rule is written for each compound — one could, for example, envisage the rule in figure 8 being written to cope with the animal grinding phenomenon in Dutch, and the figure also shows a feature structure of the translation of *chicken* (meat) into Dutch using this rule, via a translation rule linking the English ANIMAL-GRINDING rule to the Dutch VLEESCOMPOUNDING one. This is very important since it illustrates just how closely morphological and lexical rules are related, since the translation link rule in this case is between a lexical and morphological rule, rather than between two rules of the same type.

$$
\left[\begin{array}{l}
\textbf{lex-noun-sign}\\
\text{ORTH} = \textbf{kippevlees}\\
\text{CAT} = \boxed{\textbf{noun-cat}}\\
\text{SEM} = \boxed{\textbf{unary-formula-entity-arg1}}\\
\text{SENSE-ID} = \boxed{\textbf{sense-id}}\\
\text{QUALIA} = \left[\begin{array}{l}
\textbf{c\_natural}\\
\text{AGENTIVE} = \boxed{\textbf{agentivestuff}}\\
\text{TELIC} = \boxed{\textbf{verb-sem}}\\
\text{FORM} = \left[\begin{array}{l}\textbf{nomform}\\ \text{RELATIVE} = \textbf{mass}\end{array}\right]\\
\text{CONSTITUENCY} = \boxed{\textbf{nomconst}}\\
\text{OBJECT-INDEX} = \boxed{0}\ \textbf{entity}\\
\text{PROPERTIES} = \boxed{\textbf{phys-properties}}
\end{array}\right]
\end{array}\right]
$$

```
vleescompounding morph-rule
%suffix (p ppevlees) (m mvlees)
     < 1 : sense-id : language > = Dutch
     < 1 : qualia > = animal
     < 1 : cat : m-feats : agr : num > = sg
     < 0 : qualia : form : relative > = mass
     < 0 : qualia > = c_natural .
```

Figure 8: An illustration of the Dutch version of *animal-grinding*

7 References

:

Briscoe, E J & Copestake, A, 1991, *Sense Extensions as Lexical Rules*, Acquilex Working Paper No. 22, in **Sanfilippo, A** (Ed), *The (Other) Cambridge Acquilex Papers*, Cambridge University Computer Laboratory Technical Report No. 253

Copestake, A, 1992, *The Representation of Lexical Semantic Information*, Sussex University Cognitive Science Research Paper No. 280

Copestake, A & Briscoe, E J, 1991, *Lexical Operations in a Unification-based Framework*, Acquilex Working Paper No. 21, in **Sanfilippo, A** (Ed), *The (Other) Cambridge Acquilex Papers*, Cambridge University Computer Laboratory Technical Report No. 253

Jones, B E M, 1991, *A Morphological Analyser*, Cambridge University Computer Laboratory Part II Dissertation

Nunberg, G D, 1978, *The Pragmatics of Reference*, PhD thesis, University of California, reproduced by the Indiana University Linguistics Club

Sanfilippo, A, 1990, *A Morphological Analyser for English and Italian*, Acquilex Working Paper No. 4, in **Sanfilippo, A** (Ed), *The (Other) Cambridge Acquilex Papers*, Cambridge University Computer Laboratory Technical Report No. 253