VIOLETTA CAVALLI-SFORZA[1]
ABDELHADI SOUDI[2]

[1] **San Francisco State University, Department of Computer Science, San Francisco, California, USA**
[vcs@sfsu.edu]
[2] **Center for Languages and Communication , Ecole Nationale de L'Industrie Minérale, Agdal, Rabat, Morocco**
[asoudi@enim.ac.ma]

Enhancements to a Morphological Generator to Capture Arabic Morphology[1]

ABSTRACT

We describe an enhanced version of the MORPHE tool, a morphological analyzer/generator designed to interface with a knowledge-based machine translation system. MORPHE uses a hierarchy (tree structure) to relate various morphological forms to each other based on common and distinctive features. Transformational rules are attached to the leaf nodes of the hierarchy. In generation, MORPHE takes as input a feature structure and pushes it through the hierarchy, which acts as discrimination net. When a leaf node is reached, MORPHE applies the attached rule. Each rule may contain several mutually exclusive clauses, each of which attempts to match a pattern against the base string contained in the feature structure and, if the match is successful, applies operators to the string to produce a transformed string.

Our enhancements to MORPHE were motivated by attempting to use the tool to generate Arabic morphology. The non-concatenative morphology typical of Semitic languages has spurred the development of sophisticated formalisms and computational engines, as well as produced brute force approaches. In this paper we show how the relatively straightforward formalism used in MORPHE can be extended in simple ways to produce an elegant treatment of Arabic morphology that captures the inflectional regularities of the language. The result is the ability to describe Arabic morphology, as well as the morphology of any language whose word forms undergo stem changes, using a set of rules that contains minimal duplication, is easy to understand and maintain, and is useful for language learning as well as for machine translation applications.

1 INTRODUCTION

The MORPHE system [6] is a morphological analyzer/generator developed as a component of the KANT machine translation technology [7]. Although MORPHE was designed to be used in both analysis and generation, in practice it was used only for generation and only for languages whose morphology is largely concatenative, that is, languages whose morphological variation involves primarily prefixation and suffixation, with only minor modification of stem boundaries. MORPHE was also theoretically capable of handling infixation phenomena, but it had not been put to the test for languages, such as Arabic, that have stem changes occurring in complex combinations with prefixation and suffixation processes.

Our first attempt to use the MORPHE system to generate Arabic verbal morphology resulted in an innovative two-step use of MORPHE [4], decoupling stem changes from prefix and suffix changes in order to reduce the proliferation of rules that the complex Arabic conjugation system required when MORPHE was used in the conventional way. While this approach produced a smaller and less redundant set of morphology rules, there was still significant duplication of rules. Rule duplication is inelegant: it means that the approach has failed to capture the underlying regularities, making the morphology description larger and more complex than necessary. From a practical perspective, duplication also presents a maintenance problem, since changes may need to be repeated in multiple rules.

When we began extending the morphology system to handle Arabic nouns, and in particular the broken plurals of nouns [9], it became apparent that MORPHE's basic representational expressiveness was not adequate for the task. Therefore we extended it to support multiple stems for broken plurals and explicit rules to express the syncretism (similar forms) evident in both verbal stem changes and noun inflection. The Lexeme-based Morphology framework ([1], [2]) provides a theoretical justification for our representation of Arabic morphology, which focuses on stems in contrast to the root+pattern+vocalism approaches followed by other researchers.

In this paper, we begin by sketching the original MORPHE system, but we focus on the current extensions required to accommodate Arabic morphology and on the integration of the morphology system with a prototype interlingua-based English-to-Arabic MT system. We conclude by briefly comparing our approach to other treatments of Arabic morphology and describing future work.

2 THE BASIC MORPHE SYSTEM

MORPHE [6] is a morphological rule compiler written in Common Lisp. In MORPHE, the morphology of a language is described using two types of structures: 1) a morphological form hierarchy and 2) a set of transformational rules attached to the leaf nodes of the hierarchy. MORPHE compiles this description into either a word generation program or a word parsing program. The morphological generator takes as input a feature structure (FS), a list structure

[1] This research has been supported in part by a grant from the National Science Foundation.

whose elements are feature-value pairs (FVP). The value part of an FVP can be atomic or itself a FS, making the FS a recursive structure. For example, the FS for generating the Arabic zurtu 'I visited' would be:[2]

```
((root "zawar")(cat v)(form 1)(vow hol) (tense perf)(mood ind)(voice act)(number sg)(person 1))
```

The choice of feature names and values, other than root, which identifies the lexical item to be transformed, is entirely up to the user. The FVPs in a FS come from one of two sources. Static features, such as cat (part of speech) and root, come from the syntactic lexicon, which, in addition to the base form of words, can contain morphological and syntactic features. Dynamic features, such as tense and number, are set by MORPHE's caller which, in the context of a machine translation system, is the syntactic generation component. The output of the morphological generator is simply a string.

2.1 The Morphological Form Hierarchy (MFH)

The MFH organizes the transformational rules depending on the values of the features in a FS and acts as a discrimination network for retrieving the rules appropriate to a given FS. Each internal node of the tree specifies a piece of the FS that is common to that entire subtree. The root of the tree is a special node `*root*` that simply binds all subtrees together. The leaf nodes of the tree correspond to distinct morphological forms in the language. Each node in the tree below the root is built by a morphological form that specifies the parent of the node and the logical combination of FVPs that distinguish the node from its parent and siblings. For example the morphological form

```
(morph-form  v-stem-f1-act-perf-1/2  v-stem-f1-act-perf (person (*or* 1 2)))
```

says that the node `v-stem-f1-act-perf-1/2` is a child of node `v-stem-f1-act-perf` and adds the information that the `person` feature must have value `1` or `2`.

2.2 Transformational Rules

A rule attached to each leaf node of the MFH effects the desired morphological transformations for that node. A rule consists of one or more mutually exclusive clauses. The 'if' part of a clause is a regular expression pattern, which is matched against the value of the feature `root` (a string). The 'then' part includes one or more operators, applied in the given order. Operators include addition, deletion, and replacement of prefixes, infixes, and suffixes. The output of the transformation is the transformed `root` string. The transformational rule that produces the '*zur*' part of the Arabic *zurtu* 'I visited' is:

```
(morph-rule v-stem-f1-act-perf-1/2
   ("^%{cons}(awa)%{cons}$"   (ri *1* "u"))
   ("^%{cons}(a[wy]i)%{cons}$"       (ri *1* "i"))
   ("^%{cons}(aya)%{cons}$"   (ri *1* "i")))
```

2.3 Process Logic and Irregular Forms

On the assumption that not all morphology is regular, MORPHE supports irregular forms as part of its process logic. In generation, the FS is matched against the features defining each subtree in the MFH until a leaf node is reached. The generator then checks the irregular form lexicon for an entry indexed by the value of the `root` feature and the name of the node and returns it if there is one. Otherwise it attempts to apply the transformational rule attached to the leaf node. If there is no rule or none of the clauses match, the value of the `root` feature is returned unchanged.

3 USING MORPHE TO GENERATE ARABIC VERBAL MORPHOLOGY

A conventional use of MORPHE assumes that all the necessary morphological transformations to the value of the `root` feature can be specified by a single rule attached to the leaf node of the hierarchy. While this assumption holds for Arabic, it gives rise to a very bushy MFH and a rule set where the same transformational operations are repeated in several rules. The source of the repetition is the Arabic verbal system itself.[3] Verb stems may differ in the active and passive voices, in the perfect and the imperfect, in the imperfect moods, and these changes depend on the verb form. For the so-called hollow verbs (verbs with a weak middle radical), for the same voice mood and tense the stem varies for the 13 possible person-gender-number combinations. In contrast, prefixes and suffixes associated with each person-gender-number combination remain relatively stable across the different types of verbs, showing some boundary friction with weak verbs (verbs with a weak final radical).

Our initial attempt to use MORPHE to represent Arabic verbal morphology showed that we could substantially reduce the amount of repetition in our transformational rules by decoupling stem transformations and prefix/suffix changes. Portions of the Arabic MFH are shown in Figure 1. Since MORPHE supports only one MFH, we created two subtrees under the verb subtree, whose root node is indicated in Figure 1 by the FVP `(cat v)`. One subtree, indicated by the FVP `(chg stem)`, contains rules for stem changes; the other, indicated by the FVP `(chg psfix)`, contains rules for

---

[2] The rationale for choosing '*w*' as the glide (weak consonant) in '*zawar*' is given in [11].

[3] An explanation of the morphology of Arabic, even a partial one, is far beyond the scope of this paper. Briefly, verbal stems are based on triliteral or quadriliteral roots (3- or 4-radicals). Stems are formed by a derivational combination of a root morpheme and a vowel melody; the two are arranged according to canonical patterns or forms. Roots are said to interdigitate with patterns to form stems. For example, the Arabic stem katab 'he wrote' is composed of the morpheme ktb (notion of writing) and the vowel melody morpheme 'a-a'. The two are coordinated according to form 1, which corresponds to pattern CVCVC (C=consonant, V=vowel). There are 15 triliteral patterns, of which at least 9 are in common use, and 4 much rarer quadriliteral patterns. Two consonants 'w' and 'y' are considered weak in the sense that they tend to disappear or appear under different guises. For a more detailed description see [4, 11].

prefix and suffix changes. The morphological generator is called twice. The first call returns a modified stem. This is substituted as the value of the root feature before the second call, which takes care of the prefixes and suffixes. For the example of *zurtu* 'I visited', the first call finds and applies the transformational rule shown in Section 2.2 above. The second call finds and applies the rule

```
(morph-rule v-psfix-perf-1-sg  ("" (+s "otu")))
```

which adds the suffix "*tu*" (the '*o*' indicates absence of vowel).

Figure 1 also shows other aspects of the representation of Arabic morphology. The information we maintain in the lexicon for verbs includes the stem (given as the value of the `root` feature, e.g. `"zawar"`), the pattern of the verb (e.g. `(pat cvcvc)`), and the vowel change that can be expected. For hollow verbs, the middle radical ('*w*' or '*y*') and its vowel in the `root` feature and the FVP `(vow hol)` determine the required stem transformations in the perfect and imperfect. For strong verbs, the value of the vow feature provides information for imperfect stem changes. By decoupling stem changes from prefixation and suffixation, we avoided replicating all of the prefix/suffix rules for each of the different types of stem changes. Nonetheless, the subtree for the perfect tense shows that there are still duplicate rules: the leaf nodes labeled "short stem" have identical rules attached to them, as do the nodes labeled "long stem" (matching the stems '*zur*' and '*zaar*' respectively). The limited logic used to specify the MFH makes it impossible to combine these nodes and, even if we were to extend that logic, the result would be hard to understand. These and other observations motivated the extensions to MORPHE described in the next section.

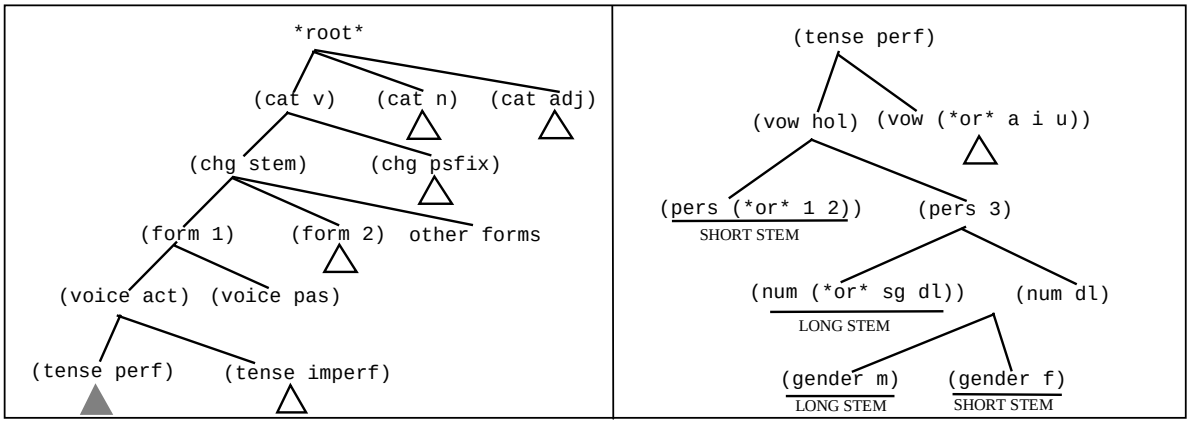

Fig. 1. Portions of the Arabic morphological form hierarchy. The shaded perfective subtree on the left-hand side of the figure is shown expanded on the right hand side of the figure. While in the actual MFH all nodes are given a name (ideally one that reflects the feature-value path used to reach them), the figure above shows only the nodes' distinguishing feature-value pairs.

4 SUPPORTING MULTIPLE STEMS, DEFAULT RULES, AND RULES OF REFERRAL

We specify Arabic verb morphology in MORPHE by providing stem information in the lexicon and writing transformational rules that act on that information to produce the final stem, but the Arabic plural noun system imposes different demands on the morphological representation. A minority of nouns form their plural by regular processes of suffixation, but the majority of nouns have one or more 'broken' plural form. An analysis of previous accounts of the Arabic broken plural system [9, 11] shows that while the association between singular and plural forms is not random, there is no way to predict exactly which plural pattern a singular will take. In generation, we need to provide at least one plural stem on which other inflectional rules can act. We could store the plural pattern in the lexicon and write complex rules to generate the plural stem from the root or singular stem, or we could store the plural stem directly in the lexicon. In giving priority to stems in our lexicon, we draw from Lexeme-based Morphology ([1], [2]), explained briefly in Section 6 below. Once we allow the broken plural stem to appear directly in the lexicon, it is easy to use the same two-step approach we used for verbal morphology to generate inflected nouns. The noun subtree of the MFH is again split into two subtrees, one for producing the correct stem and one for adding the necessary suffixes. The noun portion of the MFH is shown in Figure 2 below. The MFH is fully fleshed out only for nominative indefinite noun inflection. Other cases are handled in a parallel way.

The MFH shows three significant additions to MORPHE: 1) multiple stems using **allomorph rules**, 2) **default rules**, and 3) **rule equivalencing**.

4.1 Allomorph Rules

When a morphology specification is compiled in the enhanced MORPHE system, the user can specify which feature in the FS provides the string to be acted upon by the transformational rules (this 'base' feature could only be `root` in the original MORPHE, but we use `stem` for Arabic in the new system). The user can also attach an allomorph rule to a leaf node indicating that a different feature in the FS should be the source of the string for that node. The syntax of an allomorph rule declaration is:

```
(morph-allomorph <node name> <feature name>)
```

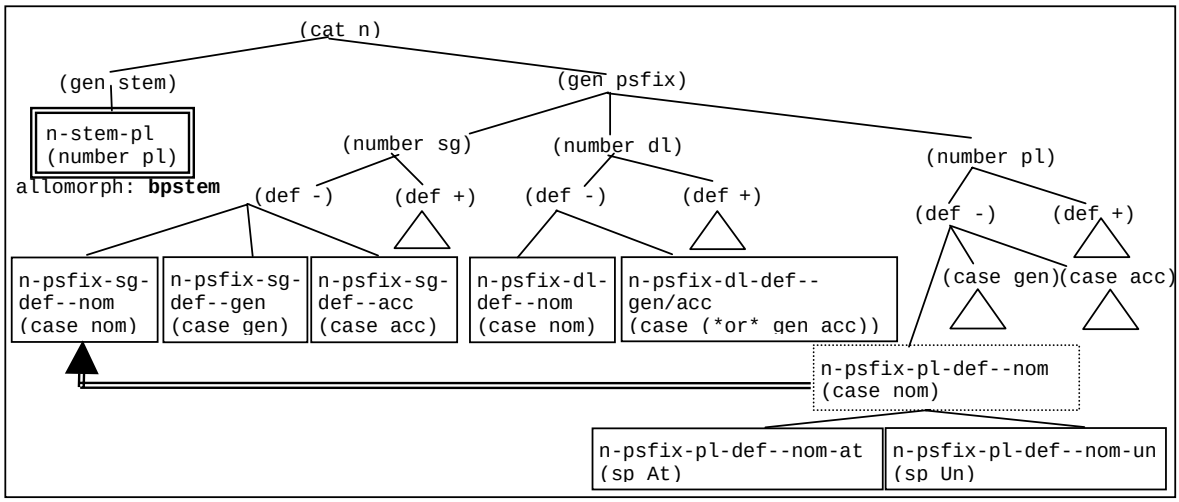

Fig. 2. The Arabic noun morphological form hierarchy (not fully expanded).

For example, the double-border box in Figure 2 represents the declaration

```
(morph-allomorph n-stem-pl bpstem)
```

and specifies that, if node `n-stem-pl` is reached, MORPHE should look in the FS for a feature named `bpstem`, which, if present, should be used in subsequent processing.

4.2 Default Rules

MORPHE uses a specific FVP or set of FVPs to represent special cases requiring special rules. These FVPs distinguish leaf-nodes from a parent node. It is also occasionally convenient to have a default rule apply when the special cases do not obtain, without requiring a particular FVP to indicate the default case. The dotted node in Figure 2, `n-psfix-pl-def--nom`, is an example of using a default rule. It shows that, if an indefinite plural noun does not have a sound plural (indicated by the FVP `(sp un)` or `(sp at)`, a default rule can be applied. The default rule is attached to the parent of the leaf nodes that represent the special sound plural cases.

4.3 Rule Equivalencing

The original MORPHE system required the MFH to be a tree. If several leaf nodes required the same transformational rule, the rule had to be duplicated. The enhanced MORPHE system avoids duplication by declaring rule equivalence in two ways:

Implicit equivalencing. Nodes reached by different paths in the MFH (i.e., by different FVP sequences) can be given the same name. A rule can then be shared by all nodes bearing the same name as the rule.

Explicit equivalencing. Nodes reached by different paths in the MFH and bearing different names can be explicitly declared to share the same rule by using the declaration syntax :

```
(morph-equivalence <reference node name> <equivalent node list>)
```

where `<reference node name>` can be the name of an actual node in the hierarchy or a virtual name and `<equivalent node list>` is a list of one or more actual node names. The effect of a `morph-equivalence` declaration is to cause all nodes in the equivalence list to share the same rule, which can be attached to an actual or virtual node name.
In Figure 2, the double arrow is standing in for an explicit rule equivalence:

```
(morph-equivalence n-psfix-sg-def--nom (n-psfix-pl-def--nom))
```

It says that the suffix for an indefinite nominative plural noun is the same as that for an indefinite nominative singular noun. It interacts with default rules allowing the default rule to be equivalenced to a rule on a different node. Used in combination with careful design, rule equivalencing eliminates rule duplication, highlights syncretism cases and embodies the rules of referral of the Lexeme-based approach.

The process logic of the enhanced MORPHE system is summarized in Figure 3.

5 INTERFACING MORPHOLOGICAL GENERATION WITH MACHINE TRANSLATION

MORPHE was designed to work within the framework of the KANT MT system [7]. In this approach, a source language sentence is transformed into a language-independent semantic representation called the interlingua representation (IR). To generate the target language sentence, the mapper maps the semantic information contained in the IR into a feature structure (FS) that reflects the syntactic structure of the target language. Target language lexicon entries are also FSs; they are retrieved during mapping and added to the sentence FS under construction.

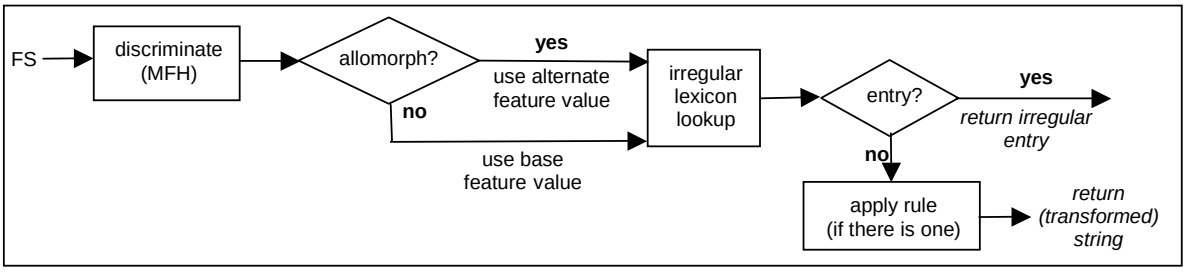

Fig. 3. Overall process logic of the enhanced MORPHE system during generation.

The Genkit syntactic analyzer/generator processes the FS and generates a preliminary target sentence string, calling MORPHE when it encounters lexical symbols in the generation grammar. Optional post-processing produces the final target sentence.

6 COMPARISON TO OTHER APPROACHES AND FUTURE WORK

Lexeme-based Morphology (LBM) [1, 2], supports the claim that the stem is the only morphologically relevant form of a lexeme. A lexeme is a vocabulary item belonging to the major lexical categories of verb, noun, adjective and adverb. It is a complex representation linking a meaning with a set of grammatical words that are associated with corresponding word forms. In an LBM model, only lexemes and free morphemes are minimal grammatical elements. Inflectional or derivational morphemes – suffixes, prefixes, infixes and reduplication – are not themselves grammatical elements but merely the phonological expression of operations that apply to basic grammatical elements.

Our computational approach to Arabic morphology using the enhanced MORPHE system adhere to LBM by storing in the lexicon multiple stems or information that allows us to derive stems from other stems. Thus the stem and operations on the stem become the focus of the representation. Our approach differs from the previous computational treatments of Arabic morphology (e.g., [3], [5]), which have essentially granted equal status to all the constituents of an Arabic word (the root, the pattern and the vocalism), by placing them in separate lexicons. These and other approaches to Arabic morphology are described in detail in [11].

The enhanced MORPHE system significantly extends the original tool's ability to represent the morphology of languages such as Arabic. Moreover the extensions are not ad-hoc but theoretically and linguistically motivated. They provide an alternative approach to a computational description of Arabic morphology, one that is significantly more compact and is easier to understand and maintain than the well-known two-level approaches and other finite-state approaches [2], [5]. Our approach is also suitable to a number of applications. For example, a morphological description such as ours, developed with full diacritics, for the purposes of computer-assisted language learning or speech synthesis, can be quickly rewritten to remove diacritics for the purposes of general text generation, which usually omits diacritics. There are also some weaknesses. Both the original and the enhanced MORPHE do not fully support morphological analysis and is almost surely slower than other approaches. Our future plans include supporting analysis, adding rule inheritance so that rules can be attached to internal nodes of the MFH, and evaluating the tool's performance in both analysis and generation from the perspective of speed.

5 REFERENCES

1. Aronoff, M: Morphology by Itself: Stems and Inflectional Classes. MIT Press, Cambridge, Mass. (1994)
2. Beard, R.: Lexeme-Morpheme Base Morphology: A General Theory of Inflection and Word Formation. State University of New York Press (1995)
3. Beesley, K.: Arabic Finite-State Morphological Analysis and Generation. In: Proceedings COLING'96, Vol. 1, (1996) 89-94
4. Cavalli-Sforza, V., Soudi A., Mitamura T.: Arabic Morphology Generation Using a Concatenative Strategy. In: Proceedings of NAACL 2000. Seattle, WA (2000).
5. Kiraz, G.: Multi-tape Two-level Morphology: A Case study in Semitic Non-Linear Morphology. In: Proceedings of COLING'94, Vol. 1 (1994) 180-186.
6. Leavitt, J.R:. MORPHE: A Morphological Rule Compiler. Technical Report, CMU-CMT-94-MEMO (1994)
7. Nyberg, E.H., Mitamura, T.: The KANT System: Fast, Accurate, High Quality Translation in Practical Domains. In: Proceedings of COLING'92 (1992)
8. Soudi, A, Cavalli-Sforza, V. Jamari, A.: A Computational Lexeme-based Treatment of Arabic Morphology. In: Proceedings of the Workshop on Arabic Language Processing: Status and Prospects, ACL 2002, Toulouse (2001)
9. Soudi, A, Cavalli-Sforza, V. Jamari, A.: Arabic Noun System Generation. In: Proceedings of the Arabic Processing Conference, University of Manouba, Tunisia (2002)
10. Soudi, A, Cavalli-Sforza, V. Jamari, A.: A Prototype English-to-Arabic Interlingua-based MT System". In: Proceedings of Workshop on Arabic Language Resources and Evaluation: Status and Prospects, LREC'2002, Las Palmas (forthcoming)
11. Soudi, A.: A Computational Lexeme-based Treatment of Arabic Morphology. Doctorat d'Etat (PhD) Thesis. Mohamed V University, Rabat (2002)