A Computational Lexeme-Based Treatment of Arabic Morphology

Abdelhadi Soudi
CLC and CS Dept. Ecole Nationale de L'Industrie Minérale (ENIM), Av. Hadj Ahmed Cherkaoui, B.P:753 Agdal, Rabat,Morocco
asoudi@enim.ac.ma

Violetta Cavalli-Sforza
Dept. of Computer Science
San Francisco State Univ.
1600 Holloway Avenue
San Francisco, CA, USA
vcs@sfsu.edu

Abderrahim Jamari
Institut Universitaire de La Recherche Scientifique
Av. Allal Fassi, B-P 6287,
Rabat-Instituts, Morocco
jamari@mail.iihem.ac.ma

Abstract

Arabic morphology represents a special type of morphological systems. It is generally considered to be of the non-concatenative type which depends on manipulating root letters in a non-concatenative manner. In addition to prefixation and suffixation, inflectional and derivational processes may cause stems to undergo infixational modification in the presence of different syntactic features as well as certain stem consonants. The basic problem, then, is the large number of variants that must be analyzed or generated. In this paper, we seek to reduce the complexity of Arabic morphology using the lexeme-based morphology theory to represent the linguistic resources and MORPHE as a computational tool to implement them. We show that the space of rules can be kept small if we consider the stem as the phonological domain of realisation rules. The reduction in the number of rules keeps the system small and also increases its understandability and maintainability. We primarily focus on generation of verbs and broken plurals.

1 Lexeme-based Morphology

As Lexeme-Based Morphology (LBM) plays an important role in the background of our claim, a brief review is in order. In a LBM model as in (Aronoff, 1994), only lexemes and free morphemes are minimal grammatical elements. Inflectional or derivational morphemes-suffixes, prefixes, infixes and reduplication are not themselves grammatical elements. Instead, these are merely the phonological expression of operations that apply to basic grammatical elements. This approach, thus, differs from the previous computational analyses of Arabic (Beesley, 1996; Kiraz, 1994) which have, essentially, granted equal status to all the constituents of a word (the root, the pattern and the vocalism) by placing them in separate lexicons.

1.1 The lexeme concept

Lexemes are vocabulary items belonging to the major lexical categories of verb, noun, adjective and adverb. For example, given the forms *cat* and *cats*, we would say that there is a lexeme CAT[1] which has two word forms *cat* and *cats* and that the description 'the singular/plural of CAT' is a grammatical word. A lexeme then is a complex representation linking a meaning with a set of word forms or grammatical words which are associated with corresponding word forms. From the point of view of the lexicon, the lexeme is a lexical entry.

1.2 The stem and the root

While a lexeme is an assembly of an item's sound form, syntax and meaning, a stem and a root correspond only to the sound-form part of

[1] We write lexemes in capital letters.

this assembly. That is, both roots and stems are sound forms of a lexeme, the difference between them is that a root is defined with respect to a lexeme, while a stem is always defined with respect to realization rules. Only the stem is morphologically relevant in that realization rules act on it. A root is what is left when all morphologically added structure has been wrung out.

2 The Arabic Verbal System

One of the most puzzling problems in the study of Arabic is its verbal system which is very rich in forms. An Arabic verb can be conjugated according to one of the traditionally recognized patterns (or conjugations). There are 15 triliteral forms, of which at least 9 are in common use. Within each conjugation/Form, an entire paradigm of word forms is found: two tenses (perfect and imperfect), two voices (active and passive) and five moods (indicative, subjunctive, jussive, imperative and energetic). The paradigms in each conjugation are highly regular. The attested irregularities are due to the phonological peculiarities of certain root consonants. The paradigms of weak roots, however, follow automatically if the nature and position of the weak consonants are known, as will be shown below.

In this section, we argue for the derivational process of Arabic inflected verbs shown in Figure 1. We consider the Arabic binyanim to be simultaneously derivational and inflectional.[2] The choice of a binyan for a particular root is part of lexeme-formation. The net effect of lexeme-formation, however, is to place the root into an inflectional class which determines its overall morphological realization (i.e., the stems a verb lexeme can take). Stem formation is morphomic, where a morphome is defined as a mapping function using purely morphological rules. Arabic verb stems can then be realized by a single morphological function F(fn), where fn ranges over the traditionally recognized patterns/Forms (form1, form2,……fn). The function F(f1), for example, will map the verb lexeme KATAB "write" onto form1 perfective stem *katab* and imperfective stem *ktub*. Inflectional rules then act on the resulting stems to yield fully inflected forms.

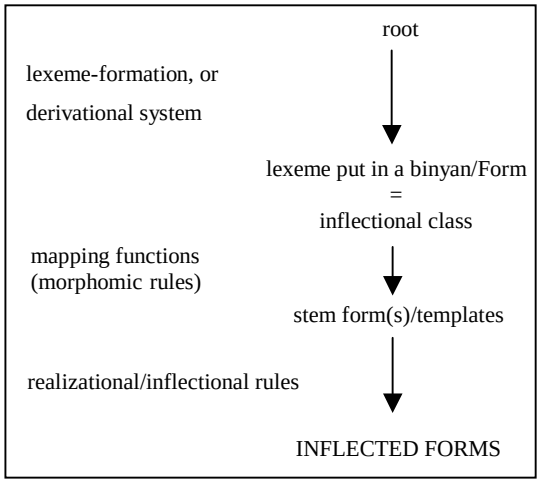

Figure 1. The Derivational Process of Arabic Inflected Verbs

In order to develop the analysis further, it will be useful to make illustrative reference to an Arabic verb's inflectional paradigm. We will draw upon the indicative perfective/imperfective form I conjugation of a strong and a weak triliteral verb. Tables 1 and 2 set out the perfective and imperfective word forms for the lexeme KATAB, respectively.

Given the paradigms in Tables 1 and 2 below, we note the following:

(i) Form 1 perfective and imperfective stems have the templates CVCVC and CCVC (C=consonant, V=vowel), respectively, throughout the paradigm;

(ii) There are cases of syncretism in both paradigms; that is certain combinations of features have the same realization as certain others: certain inflected verbs have the same word form as certain others.

An examination of Arabic strong verbs reveals that Form 1 perfective and imperfective stem templates are valid for all verbs. Accordingly, we can associate form 1 conjugations with two stem templates: CVCVC for the perfective and CCVC for the imperfective.

There is, however, a complication with the postulation of the two prosodic stem templates provided above. The complication relates to

[2] The terms binyan, conjugation, pattern and Form are used interchangeably. The term "word form" or "grammatical word" refers to an inflectional form.

suppletive verbs, namely hollow and weak verbs. Hollow verbs are those with a weak middle radical. Weak verbs are those with a weak final radical. Hollow verbs, for example, are characterized by two perfective stems, one short (CVC) and one long (CaaC). The nature of the stem vowel depends on the lexeme's penultimate consonant (the glide y or w) and the ultimate vowel, as is shown below.

Person	Number	Masculine	Feminine
1st	singular	katab-tu	katab-tu
	dual	katab-naa	katab-naa
	plural	katab-naa	katab-naa
2nd	singular	katab-ta	katab-ti
	dual	katab-tumaa	katab-tumaa
	plural	katab-tum	katab-tunna
3rd	singular	katab-a	katab-at
	dual	katab-aa	katab-ataa
	plural	katab-uu	katab-na

Table 1. Perfective word forms of the strong verb lexeme KATAB (to write)

Person	Number	Masculine	Feminine
1st	singular	?a-ktub-u	ta-ktub-u
	dual	na-ktub-u	na-ktub-u
	plural	na-ktub-u	na-ktub-u
2nd	singular	ta-ktub-u	ta-ktub-u
	dual	ta-ktub-aani	ta-ktub-aani
	plural	ta-ktub-uuna	ta-ktub-na
3rd	singular	ya-ktub-u	ya-ktub-u
	dual	ya-ktub-aani	ya-ktub-aani
	plural	ya-ktub-uuna	ya-ktub-na

Table 2. Imperfective word forms of the strong verb lexeme KATAB

2.1 Hollow Verb Classes

(i) Verbs of the pattern CawuC or CawaC have the perfective stem patterns CuC and CaaC and the imperfective stem patterns CuC and CuuC . For example, *zaara* (from *zawara*) 'he visited' has perfective stems *zur* and *zaar* and imperfective stems *zur* and *zuur*.

(ii) Verbs of the pattern CawiC have the perfective stem patterns CiC and CaaC and the imperfective stem patterns CaC and CaaC. *naama* (from *nawima*) 'he slept', for example, has the stems *nim* and *naam* in the perfective and *nam* and *naam* in the imperfective.

(iii) Verbs of the pattern CayaC have the stem patterns CiC and CaaC in the perfective and CiC and CiiC in the imperfective. For example, *baa'a* (from *baya'a*) 'he sold' has the perfective stems *bi'* and *baa'* and the imperfective stems *bi'* and *bii'*.

(iv) Verbs of the pattern CayiC has the stem patterns CiC and CaaC in the perfective and CaC and CaaC in the imperfective. haaba (from hayiba) 'he feared' has the stems hib and haab in the perfective and hab and haab in the imperfective.

There are two ways of dealing with suppletive verbs. One might list separate stems in the lexicon. Another way is to derive them (the stems) by rules. The last solution is more promising for two main reasons. Firstly, suppletion operates at the stem level and involves only one single root consonant. Secondly, it allows us to capture generalizations by providing a unified treatment of strong and weak verbs.

The number of rules required to capture generalizations depends on how we handle syncretism in the paradigms of strong and hollow verbs: at the whole word form level (= the stem and the affixes) or simply at the stem level. In the perfective paradigm in Table 1 above, there are three evident instances of syncretism: the first person singular masculine and feminine word forms are identical; the first person dual masculine or feminine and first person plural masculine or feminine word forms are identical; the second person dual masculine and second person dual feminine word forms are identical. Given these instances of syncretism, the eighteen word forms resulting from the eighteen number-person-gender combinations can be derived by 13 rules whether we look at syncretism at the whole word form level or simply at the stem level (= 3 rules for cases involving syncretism and 10 rules for the other word forms).

Let us now consider a suppletive verb paradigm, shown in Table 3. As mentioned above, the stem of hollow verbs undergoes suppletion: the middle radical (the glide)

disappears and there is a vowel change. By considering syncretism at the whole word form level, we need 13 more rules (i.e. 26 rules) to account for the stem changes in Table 3 below. That is, we need an additional rule for every person, number and gender combination to get the right stem. However, by handling syncretism simply at the stem level, we reduce the thirteen additional rules to 3. In the paradigm below, the first person and second person word forms have the same stem (i.e. -zur-). To capture this generalization, we postulate a [+/- 3rd person] feature so that the first and second person can form a coherent class. The most economical way to handle the stems in the third person is to postulate a default rule that applies to all third person number, number-gender combinations and another more specific overriding rule to account for the third person plural-feminine combination. (see the computational implementation in section 4).

Person	Number	Masculine	Feminine
	singular	zur+tu	zur+tu
1st	dual	zur+naa	zur+naa
	plural	zur+naa	zur+naa
	singular	zur+ta	zur+ti
2nd	dual	zur+tumaa	zur+tumaa
	plural	zur+tum	zur+tunna
	singular	zaar+a	zaar+at
3rd	dual	zaar+aa	zaar+ataa
	plural	zaar+uu	zur+na

Table 3. Perfective forms of the lexeme ZAWAR (to visit)

It emerges from the facts above that a unified treatment of strong verbs and weak verbs (including hollow verbs) can be captured with fewer rules if we decouple the changes operating at the stem level and those operating at the prefixation/suffixation level.

3 The MORPHE System

We use the MORPHE (Leavitt, 1994) morphological rule compiler, implemented in CommonLisp, to implement the linguistic analysis. MORPHE is a tool that compiles morphological transformation rules into either a word parsing program or word generation program.[3] In this paper we focus on the use of MORPHE in generation.

3.1 Input and Output

The input to MORPHE is a feature structure (FS) which describes the item that must be transformed. An FS is a recursive Lisp list structure for storing morphosyntactic information. Each element of the FS is a feature-value pair (FVP). The value can be atomic or complex. MORPHE's output is simply a string. For example, the FS for generating the verb form nim-tu ' I slept' from the lexeme NAWIM would be:

```
((ROOT "NAWIM") (CAT V) (FORM 1)
 (VOICE ACT) (TENSE PERF) (NUMBER  SG)
 (PERSON 1))
```
[4]

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. Dynamic features, such as TENSE and NUMBER, are determined by MORPHE's caller. In a machine translation system, the input sentence and other linguistic knowledge would determine the tense of the output sentence.

3.2 The Morphological Form Hierarchy

MORPHE is based on the notion of a morphological hierarchy or tree. 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 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 specifying the parent of the node and the conjunction or disjunction of FVPs that define the node.

3.3 Transformational Rules

Each transformational is essentially a set of if-then rules. The 'if' part is a regular expression pattern, which is matched against the ROOT

[3] While MORPHE's computational engine is a general one, the discrimination hierarchy and the transformational rules must be developed for each language.

[4] The choice of feature names and values, other than ROOT, which identifies the lexical item to be transformed is entirely up to the user of MORPHE. That's why, we use the feature name ROOT instead of LEXEME.

FVP. The 'then' part applies operators which include addition, deletion, and replacement of prefixes, infixes and suffixes. The following three rules, for example, would handle the stem changes of all Arabic hollow verbs with the following values: the perfective, persons 1 and 2, singular or plural:

```
(morph-rule v-stem-f1-act-perf-12
   ("^%{cons} (awa)%{cons}$"
      (ri *1* "u")))
   ("^%{cons} (a[wy]i)%{cons}$"
      (ri *1* "i")))
   ("^%{cons} (aya)%{cons}$"
      (ri *1* "i")))
```

The syntax %{var} is used to indicate variables with a given set of values. Enclosing a string in parenthesis associates it with a numbered register, so the replace infix (ri) operator can access it for substitution. The first rule, for example, changes the lexeme/root ZAWAR into zur.

3.4 MORPHE's Process Logic

In generation, the MFH acts as a discrimination network. The specified FS is matched against the features defining each subtree until a leaf node is reached. At that point, MORPHE first checks in the irregular forms lexicon for an entry indexed by the name of the leaf node (i.e., the MF) and the value of the ROOT feature in the FS. If an irregular form is not found, the transformation rule attached to the leaf node is tried. If no rule is found or none of the clauses of the applicable rule match, MORPHE returns the value of ROOT unchanged.

4 Implementation of LBM Treatment of Arabic Verbal Morphology

The linguistic analysis given in section 2 has shown that the most economical way of handling Arabic verbal morphology is (i) to consider the stem as the only morphologically relevant sound form and (ii) to decouple the problem of stem changes from that of affixation changes. It is also claimed that Arabic binyanim is both derivational and inflectional. That is, if we know the conjugation form/binyan of a verb lexeme, we would automatically generate the perfective and imperfective stems automatically, using mapping functions.

These linguistic results suggest that the generation of Arabic verbal morphology be in two steps. Since our lexicon is lexeme-based, the input to MORPHE is a lexeme (ROOT in MORPHE) with a set of morphosyntactic features in the form of FVPs. MORPHE is first called with the feature GEN (=generate) set to stem. The required stem is returned and temporarily substituted for the value of the ROOT (lexeme) feature after MORPHE has traversed the nodes branching from (GEN STEM). The second call to MORPHE with the feature GEN set to INFL (=inflection) adds the required prefixes and/or suffixes, allowing thus inflectional rules to act on the stem. The output is a fully inflected verb. Figure 2 shows a portion of the morphological form hierarchy of Arabic verbs and the perfective stem generation for form 1 (i.e; verbs of the pattern CVCVC) strong and hollow verbs.

The rules effecting the changes are attached to the leaf nodes labelled with the FVPs. They perform the conversion of the ROOT (lexeme) feature value to the short and long stems. In the perfective stem generation subtree, the four classes of hollow verbs provided in section 2 are treated as three separate conditions by matching on the middle radical and the adjacent vowels and replacing them with the appropriate vowel.[5]

By way of illustration, let us consider how the fully inflected verb *nim-tu* "I slept" is generated from the root/lexeme NAWIM. The first call to MORPHE with GEN set to stem returns *nim* after MORPHE traverses the MFH v-stem-f1-act-perf-12 given in Figure 2 and applies the rule meeting this hierarchy: [6]

```
("^%{cons} (a[wy]i)%{cons}$"
   (ri *1* "i")))
```

[5] The syntax of MORPHE allows us to merge classes 2 and 4 which have the same short stem vowel-using one rule:

```
("^%{cons} (a[wy]i)%{cons}$"  (ri *1* "i")))
```

(a[wy]i) denotes an a followed by either a w or y followed by i.

[6] The use of the feature AGR as a portmanteau morpheme in the MFH makes the tree less bushy and reduces the number of rules. In a previous work (cf. Cavalli-Sforza *et al.*,2000), we decomposed the AGR feature into Number, Person and Gender.

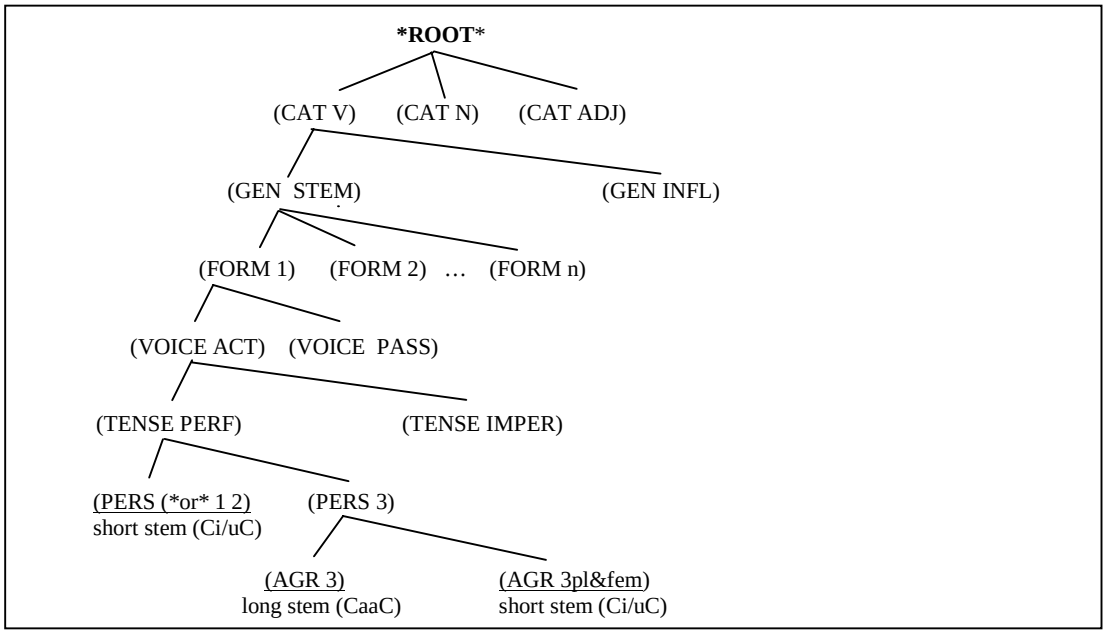

Figure 2. MFH of Arabic verbs and perfective stem generation for form 1 strong and hollow verbs.

In the second call, MORPHE is set to INFL. It traverses a different subtree, namely v-infl-perf-1-sing and applies the rule:

```
("" (+s "tu")
```

This rule adds the inflectional suffix *tu* to the stem and MORPHE returns the fully inflected verb *nimtu*.[7]

5 Handling The Arabic Broken Plural in MORPHE

There are two kinds of plural in Arabic. Firstly, there is the sound plural, the use of which is practically confined to (at least in the masculine) to participles and nouns indicating profession and habit. Secondly, there is the broken plural, which is formed according to many patterns by altering the vowels within or outside the framework of the middle radicals. In this paper, we focus on the broken plural system, which does pose problems for a computational treatment of Arabic morphology. The system is highly allomorphic: for a given singular pattern, two different plural forms may be equally frequent, and there may be no way to predict which of the two a particular singular will take. For some singulars as many as three further statistically minor patterns are also possible. The range of allomorphy is in general from two to five. For example, a singular noun with the pattern CVCC would have one or two of the following plural patterns: CuCuuC, ^aCCaaC, CiCaaC or ^aCCuC. Examples showing the broken plural of the singular pattern CVCC are shown in Table 4.

The aforementioned problems can be handled in two ways. We can either provide the broken plural pattern in the lexicon and then a series of morphological rules operate on the singular noun to generate the plural noun in the morphological component. These rules would act at the internal level to convert the singular stem to the plural stem and at the external level to add the inflectional affixes (e.g. Case affixes : nominative, genitive or accusative suffixes). The other approach would be to provide the singular and plural

[7] The imperfective paradigms are handled in a similar manner except that the ablaut (stem vowel change) exhibited in Form 1 verbs is accounted for by pre-specifying the imperfective vowel for the unpredictable cases.

stems in the lexicon and then add the required affixes in morphology. The two approaches have been neatly summarized by Zwicky (1986) as a "trade-off between multiple operations and multiple stems". The first approach would obviously involve several rules, since nouns with a broken plural pattern have, in general, complex stem alternants. The multiple-stem approach sounds more promising: nouns with a broken pattern commonly display two major stem alternants: the singular/dual allostem and the plural allostem. To capture the fact that there are two forms and that these forms are systematically distributed, the lexeme is given an inventory of two stems. To ensure that the right stem is used for every morphosyntactic word, the stems are labelled with the number feature such that stems with the singular feature are the ones referred to in inflectional rules acting on the singular stem and those with the plural feature in inflectional rules acting on the plural stem.

Singular	Plural	Gloss
wazn	^awzaan	"measure"
qird	quruud	"monkey"
kalb	kilaab	"dog"
'ayn	'uyuun, ^a'yun	"eye"

Table 4. Examples of broken plural patterns

Adopting a stem-based approach, no stem changes are required in morphology, for both the singular and the plural stems are retrievable from the syntactic lexicon. MORPHE would just add the prefixes (eg. definite article) and the Case suffixes required (the nominative, genitive or accusative suffix). The lexeme *RAJUL* "man", for example, would have the stem inventory: *rajul* (singular stem) and *rijaal* (broken plural stem). In the generation of the indefinite, nominative plural form *rijaalun*, for example, the plural stem is retrieved from the syntactic lexicon and then the nominative suffix *un* is added after MORPHE traverses the relevant morphological form hierarchy describing the definiteness and Case features. The same suffix can be added to singular stem to generate the indefinite, nominative and singular form *rajulun*.

6 Conclusion

We have demonstrated that greater generalization can be located in the derivational system of Arabic by taking a lexeme-based approach where the morphological transformations of lexemes operate at the stem level. In this way, we gain a significant reduction in the number of transformational rules required. This improves the space efficiency of the system and its maintainability by reducing duplication of rules. It also simplifies the rules by isolating different types of changes.

The system has been tested on weak and strong verbs and nouns. Its integration within the KANT (Knowledge-based Accurate Natural-Language Translation) system, a tool developed at the Language Technologies Institute in Carnegie Mellon University for the research and development of large-scale, practical translation systems for technical documentation, has also been successfully tested.

References

Mark Aronoff. 1994. *Morphology by Itself: Stems and Inflectional Classes*. MIT Press, Cambridge, Mass.

Robert Beard. 1995. *Lexeme-Morpheme Base Morphology: A General Theory of Inflection and Word Formation*. State University of New York Press, Albany, NY.

Ken Beesley. 1990. Finite-State Description of Arabic Morphology. In *Proceedings of the Second Cambridge Conference: Bilingual Computing in Arabic and English*.

Ken Beesley. 1996. Arabic Finite-State Morphological Analysis and Generation. In *Proceedings of COLING'96*, volume 1, pp. 89-94.

Cavalli-Sforza, V., Soudi A and Mitamura T. (2000). Arabic Morphology Generation Using a Concatenative Strategy. In *Proceedings of NAACL 2000*. Seattle, WA.

D. Cowan. 1964. *An Introduction to Modern Literary Arabic*. Cambridge University Press, Cambridge.

George Kiraz. 1994. Multi-tape Two-level Morphology: A Case study in Semitic Non-linear Morphology. In *Proceedings of of COLING'94*, volume.1, pp. 180-186.

John R. Leavitt. 1994. *MORPHE: A Morphological Rule Compiler*. Technical Report, CMU-CMT-94-MEMO.

J. McCarthy and A. Prince. 1990. Foot and Word in Prosodic Morphology: The Arabic Broken Plural. *Natural Language and Linguistics Theory*, 8:209-283.

Arnold Zwicky. 1986. The General Case: Basic Form versus Default Form. *Berkeley Linguistics Society* 12:305-314.