Semitic Morphological Analysis and Generation Using Finite State Transducers with Feature Structures

Michael Gasser
Indiana University, School of Informatics
Bloomington, Indiana, USA
gasser@indiana.edu

Abstract

This paper presents an application of finite state transducers weighted with feature structure descriptions, following Amtrup (2003), to the morphology of the Semitic language Tigrinya. It is shown that feature-structure weights provide an efficient way of handling the templatic morphology that characterizes Semitic verb stems as well as the long-distance dependencies characterizing the complex Tigrinya verb morphotactics. A relatively complete computational implementation of Tigrinya verb morphology is described.

1 Introduction

1.1 Finite state morphology

Morphological analysis is the segmentation of words into their component morphemes and the assignment of grammatical morphemes to grammatical categories and lexical morphemes to lexemes. For example, the English noun *parties* could be analyzed as *party*+PLURAL. **Morphological generation** is the reverse process. Both processes relate a **surface** level to a **lexical** level. The relationship between these levels has concerned many phonologists and morphologists over the years, and traditional descriptions, since the pioneering work of Chomsky and Halle (1968), have characterized it in terms of a series of ordered content-sensitive rewrite rules, which apply in the generation, but not the analysis, direction.

Within computational morphology, a very significant advance came with the demonstration that phonological rules could be implemented as **finite state transducers** (Johnson, 1972; Kaplan and Kay, 1994) (FSTs) and that the rule ordering could be dispensed with using FSTs that relate the surface and lexical levels directly (Koskenniemi, 1983). Because of the invertibility of FSTs, “two-level” phonology and morphology permitted the creation of systems of FSTs that implemented both analysis (surface input, lexical output) and generation (lexical input, surface output).

In addition to inversion, FSTs are closed under **composition**. A second important advance in computational morphology was the recognition by Karttunen et al. (1992) that a cascade of composed FSTs could implement the two-level model. This made possible quite complex finite state systems, including ordered **alternation rules** representing context-sensitive variation in the phonological or orthographic shape of morphemes, the **morphotactics** characterizing the possible sequences of morphemes (in canonical form) for a given word class, and one or more **sublexicons**. For example, to handle written English nouns, we could create a cascade of FSTs covering the rules that insert an *e* in words like *bushes* and *parties* and relate lexical *y* to surface *i* in words like *buggies* and *parties* and an FST that represents the possible sequences of morphemes in English nouns, including all of the noun stems in the English lexicon. The key feature of such systems is that, even though the FSTs making up the cascade must be composed in a particular order, the result of composition is a single FST relating surface and lexical levels directly, as in two-level morphology.

1.2 FSTs for non-concatenative morphology

These ideas have revolutionized computational morphology, making languages with complex word structure, such as Finnish and Turkish, far more amenable to analysis by traditional computational techniques. However, finite state morphology is inherently biased to view morphemes as sequences of characters or phones and words as concatenations of morphemes. This presents problems in the case of **non-concatenative morphology**: discontinuous morphemes (circumfixation); infixation, which breaks up a morpheme by inserting another within it; reduplication, by which part or all of some morpheme is copied; and the **template morphology** (also called stem-pattern morphology, intercalation, and interdigitation) that characterizes Semitic languages, and which is the focus of much of this paper. The stem of a Semitic verb consists of a **root**, essentially a sequence of consonants, and a **pattern**, a sort of template which inserts other segments between the root consonants and possibly copies certain of them (see Tigrinya examples in the next section).

Researchers within the finite state framework have proposed a number of ways to deal with Semitic template morphology. One approach is to make use of separate tapes for root and pattern at the lexical level (Kiraz, 2000). A transition in such a system relates a single surface character to multiple lexical characters, one for each of the distinct sublexica.

Another approach is to have the transducers at the lexical level relate an upper abstract characterization of a stem to a lower string that directly represents the merging of a particular root and pattern. This lower string can then be compiled into an FST that yields a surface expression (Beesley and Karttunen, 2003). Given the extra **compile-and-replace** operation, this resulting system maps directly between abstract lexical expressions and surface strings. In addition to Arabic, this approach has been applied to a portion of the verb morphology system of the Ethio-Semitic language Amharic (Amsalu and Demeke, 2006), which is characterized by all of the same sorts of complexity as Tigrinya.

A third approach makes use of a finite set of registers that the FST can write to and read from (Cohen-Sygal and Wintner, 2006). Because it can remember relevant previous states, a "finite-state registered transducer" for template morphology can keep the root and pattern separate as it processes a stem.

This paper proposes an approach which is closest to this last framework, one that starts with familiar extension to FSTs, weights on the transitions. The next section gives an overview of Tigrinya verb morphology. The following section discusses weighted FSTs, in particular, with weights consisting of feature structure descriptions. Then I describe a system that applies this approach to Tigrinya verb morphology.

2 Tigrinya Verb Morphology

Tigrinya is an Ethio-Semitic language spoken by 5-6 million people in northern Ethiopia and central Eritrea. There has been almost no computational work on the language, and there are effectively no corpora or digitized dictionaries containing roots. For a language with the morphological complexity of Tigrinya, a crucial early step in computational linguistic work must be the development of morphological analyzers and generators.

2.1 The stem

A Tigrinya verb (Leslau, 1941 is a standard reference for Tigrinya grammar) consists of a stem and one or more prefixes and suffixes. Most of the complexity resides in the stem, which can be described in terms of three dimensions: **root** (the only strictly lexical component of the verb), **tense-aspect-mood** (TAM), and **derivational category**. Table 1 illustrates the possible combinations of TAM and derivational category for a single root.[1]

A Tigrinya verb root consists of a sequence of three, four, or five consonants. In addition, as in other Ethio-Semitic languages, certain roots include inherent vowels and/or gemination (lengthening) of particular consonants. Thus among the three-consonant roots, there are three subclasses: *CCC*, *CaCC*, *CC_C*. As we have seen, the stem of a Semitic verb can be viewed as the result of the insertion of pattern vowels between root consonants and the copying of root consonants in particular positions. For Tigrinya, each combination of root class, TAM, and derivational category is characterized by a particular pattern.

With respect to TAM, there are four possibilities, as shown in Table 1, conventionally referred to in English as PERFECTIVE, IMPERFECTIVE, JUSSIVE-IMPERATIVE, and GERUNDIVE. Word-forms within these four TAM categories combine with auxiliaries to yield the full range of possbilities in the complex Tigrinya tense-aspect-mood system. Since auxiliaries are written as separate words or separated from the main verbs by an apostrophe, they will not be discussed further.

Within each of the TAM categories, a Tigrinya verb root can appear in up to eight different deriva-

[1] I use *ɨ* for the high central vowel of Tigrinya, *ɛ* for the mid central vowel, *q* for the velar ejective, a dot under a character to represent other ejectives, a right quote to represent a glottal stop, a left quote to represent the voiced pharyngeal fricative, and _ to represent gemination. Other symbols are conventional International Phonetic Alphabet.

	simple	**pas/refl**	**caus**	**freqv**	**recip1**	**caus-rec1**	**recip2**	**caus-rec2**
perf	*fɛlɛṭ*	*tɛfɛl(ɛ)ṭ*	*aflɛṭ*	*fɛlalɛṭ*	*tɛfalɛṭ*	*af_alɛṭ*	*tɛfɛlalɛṭ*	*af_ɛlalɛṭ*
imprf	*fɛl(_ɨ)ṭ*	*fɨl_ɛṭ*	*af(ɨ)l(_)ɨṭ*	*fɛlalṭ*	*f_alɛṭ*	*af_alṭ*	*f_ɛlalɛṭ*	*af_ɛlalṭ*
jus/impv	*flɛṭ*	*tɛfɛlɛṭ*	*aflɨṭ*	*fɛlalṭ*	*tɛfalɛṭ*	*af_alṭ*	*tɛfɛlalɛṭ*	*af_ɛlalṭ*
ger	*fɛliṭ*	*tɛfɛliṭ*	*afliṭ*	*fɛlaliṭ*	*tɛfaliṭ*	*af_aliṭ*	*tɛfɛlaliṭ*	*af_ɛlaliṭ*

Table 1: Stems based on the Tigrinya root $\sqrt{}$*flṭ*.

tional categories, which can can be characterized in terms of four binary features, each with particular morphological consequences. These features will be referred to in this paper as "ps" ("passive"), "tr" ("transitive"), "it" ("iterative"), and "rc" ("reciprocal"). The eight possible combinations of these features (see Table 1 for examples) are SIMPLE [-ps,-tr,-it,-rc], PASSIVE/REFLEXIVE [+ps,-tr,-it,-rc], TRANSITIVE/CAUSATIVE: [-ps,+tr,-it,-rc], FREQUENTATIVE [-ps,-tr,+it,-rc], RECIPROCAL 1 [+ps,-tr,-it,+rc], CAUSATIVE RECIPROCAL 1 [-ps,+tr,-it,+rc], RECIPROCAL 2 [+ps,-tr,+it,-rc], CAUSATIVE RECIPROCAL 2 [-ps,+tr,+it,-rc]. Notice that the [+ps,+it] and [+tr,+it] combinations are roughly equivalent semantically to the [+ps,+rc] and [+tr,+rc] combinations, though this is not true for all verb roots.

2.2 Affixes

The affixes closest to the stem represent **subject agreement**; there are ten combinations of person, number, and gender in the Tigrinya pronominal and verb-agreement system. For imperfective and jussive verbs, as in the corresponding TAM categories in other Semitic languages, subject agreement takes the form of prefixes and sometimes also suffixes, for example, *yɨflɛṭ* 'that he know', *yɨflɛṭu* 'that they (mas.) know'. In the perfective, imperative, and gerundive, subject agreement is expressed by suffixes alone, for example, *fɛlɛṭki* 'you (sg., fem.) knew', *fɛlɛṭu* 'they (mas.) knew!'.

Following the subject agreement suffix (if there is one), a transitive Tigrinya verb may also include an **object suffix** (or object agreement marker), again in one of the same set of ten possible combinations of person, number, and gender. There are two sets of object suffixes, a plain set representing direct objects and a prepositional set representing various sorts of dative, benefactive, locative, and instrumental complements, for example, *yɨfɛlṭɛn_i* 'he knows me', *yɨfɛlṭɛl_ɛy* 'he knows for me'.

Preceding the subject prefix of an imperfective or jussive verb or the stem of a perfective, imperative, or gerundive verb, there may be the prefix indicating **negative polarity**, *ay-*. Non-finite negative verbs also require the suffix *-n*: *yɨfɛlṭɛn_i* 'he knows me'; *ay_ɨfɛlṭɛn_ɨn* 'he doesn't know me'.

Preceding the negative prefix (if there is one), an imperfective or perfective verb may also include the prefix marking **relativization**, *(z)ɨ-*, for example, *zifɛlṭɛn_i* '(he) who knows me'. The relativizer can in turn be preceded by one of a set of seven **prepositions**, for example, *kabzifɛlṭɛn_i* 'from him who knows me'. Finally, in the perfective, imperfective, and gerundive, there is the possibility of one or the other of several **conjunctive prefixes** at the beginning of the verb (without the relativizer), for example, *kifɛlṭɛn_i* 'so that he knows me' and one of several **conjunctive suffixes** at the end of the verb, for example, *yɨfɛlṭɛn_ɨn* 'and he knows me'.

Given up to 32 possible stem templates (combinations of four tense-aspect-mood and eight derivational categories) and the various possible combinations of agreement, polarity, relativization, preposition, and conjunction affixes, a Tigrinya verb root can appear in well over 100,000 different wordforms.

2.3 Complexity

Tigrinya shares with other Semitic languages complex variations in the stem patterns when the root contains glottal or pharyngeal consonants or semivowels. These and a range of other regular language-specific morphophonemic processes can be captured in alternation rules. As in other Semitic languages, reduplication also plays a role in some of the stem patterns (as seen in Table 1). Furthermore, the second consonant of the most important conjugation class, as well as the consonant of most of the object suffixes, geminates in certain environments and not others (Buckley, 2000), a process that depends on syllable weight.

The morphotactics of the Tigrinya verb is replete with dependencies which span the verb stem: (1) the negative circumfix *ay-n*, (2) absence of the negative suffix *-n* following a subordinating prefix, (3) constraints on combinations of subject agreement prefixes and suffixes in the imperfective and jussive, (4) constraints on combinations of subject agreement affixes and object suffixes.

There is also considerable ambiguity in the system. For example, the second person and third person feminine plural imperfective and jussive subject suffix is identical to one allomorph of the third person feminine singular object suffix (*yɨfɛlṭa*) 'he knows her; they (fem.) know'). Tigrinya is written in the Ge'ez (Ethiopic) syllabary, which fails to mark gemination and to distinguish between syllable final consonants and consonants followed by the vowel *ɨ*. This introduces further ambiguity.

In sum, the complexity of Tigrinya verbs presents a challenge to any computational morphology framework. In the next section I consider an augmentation to finite state morphology offering clear advantages for this language.

3 FSTs with Feature Structures

A **weighted FST** (Mohri et al., 2000) is a finite state transducer whose transitions are augmented with weights. The weights must be elements of a **semiring**, an algebraic structure with an "addition" operation, a "multiplication" operation, identity elements for each operation, and the constraint that multiplication distributes over addition. Weights on a path of transitions through a transducer are "multiplied", and the weights associated with alternate paths through a transducer are "added". Weighted FSTs are closed under the same operations as unweighted FSTs; in particular, they can be composed. Weighted FSTs are familiar in speech processing, where the semiring elements usually represent probabilities, with "multiplication" and "addition" in their usual senses.

Amtrup (2003) recognized the advantages that would accrue to morphological analyzers and generators if they could accommodate structured representations. One familiar approach to representing linguistic structure is **feature structures** (FSs) (Carpenter, 1992; Copestake, 2002). A feature structure consists of a set of attribute-value pairs, for which values are either atomic properties, such as FALSE or FEMININE, or feature structures. For example, we might represent the morphological structure of the Tigrinya noun *gɛzay* 'my house' as [lex=*gɛza*, num=sing, poss=[pers=1, num=sg]]. The basic operation over FSs is **unification**. Loosely speaking, two FSs unify if their attribute-values pairs are compatible; the resulting unification combines the features of the FSs. For example, the two FSs [lex=*gɛza*, num=sg] and [poss=[pers=1, num=sg]] unify to yield the FS [lex=*gɛza*, num=sg, poss=[pers=1, num=sg]]. The distinguished FS TOP unifies with any other FS.

Amtrup shows that sets of FSs constitute a semiring, with pairwise unification as the multiplication operator, set union as the addition operator, TOP as the identity element for multiplication, and the empty set as the identity element for addition. Thus FSTs can be weighted with FSs. In an FST with FS weights, traversing a path through the network for a given input string yields an FS set, in addition to the usual output string. The FS set is the result of repeated unification of the FS sets on the arcs in the path, starting with an initial input FS set. A path through the network fails not only if the current input character fails to match the input character on the arc, but also if the current accumulated FS set fails to unify with the FS set on an arc.

Using examples from Persian, Amtrup demonstrates two advantages of FSTs weighted with FS sets. First, long-distance dependencies within words present notorious problems for finite state techniques. For generation, the usual approach is to overgenerate and then filter out the illegal strings below, but this may result in a much larger network because of the duplication of state descriptions. Using FSs, enforcing long-distance constraints is straightforward. Weights on the relevant transitions early in the word specify values for features that must agree with similar feature specifications on transitions later in the word (see the Tigrinya examples in the next section). Second, many NLP applications, such a machine translation, work with the sort of structured representations that are elegantly handled by FS descriptions. Thus it is often desirable to have the output of a morphological analyzer exhibit this richness, in contrast to the string representations that are the output of an unweighted finite state analyzer.

4 Weighted FSTs for Tigrinya Verbs

4.1 Long-distance dependencies

As we have seen, Tigrinya verbs exhibit various sorts of long-distance dependencies. The circumfix that marks the negative of non-subordinate verbs, *ay...n*, is one example. Figure 1 shows how this constraint can be handled naturally using an FST weighted with FS sets. In place of the separate negative and affirmative subnetworks that would have to span the entire FST in the absence of weighted arcs, we have simply the negative and affirmative branches at the beginning and end of the weighted FST. In the analysis direction, this FST will accept forms such as *ay_ifɛlṭun* 'they don't know' and *yifɛlṭu* 'they know' and reject forms such as *ay_ifɛlṭu*. In the generation direction, the FST will correctly generate a form such as *ay_ifɛlṭun* given a initial FS that includes the feature [pol=neg].

4.2 Stems: root and derivational pattern

Now consider the source of most of the complexity of the Tigrinya verb, the stem. The stem may be thought of as conveying three types of information: lexical (the root of the verb), derivational, and TAM. However, unlike the former two types, the TAM category of the verb is redundantly coded for by the combination of subject agreement affixes. Thus, analysis of a stem should return at least the root and the derivational category, and generation should start with a root and a derivational category and return a stem. We can represent each root as a sequence of consonants, separated in some cases by the vowel *a* or the gemination character (_). Given a particular derivational pattern and a TAM category, extracting the root from the stem is a straightforward matter with an FST. For example, for the imperfective passive, the *CC_C* root pattern appears in the template *CiC_ɛC*, and the root is what is left if the two vowels in the stem are skipped over.

However, we want to extract both the derivational pattern and the root, and the problem for finite state methods, as discussed in Section 1.2, is that both are spread throughout the stem. The analyzer needs to alternate between recording elements of the root and clues about the derivational pattern as it traverses the stem, and the generator needs to alternate between outputting characters that represent root elements and characters that depend on the derivational pattern as it produces the stem. The process is complicated further because some stem characters, such as the gemination character, may be either lexical (that is, a root element) or derivational, and others may provide information about both components. For example, a stem with four consonants and *a* separating the second and third consonants represents the frequentative of a three-consonant root if the third and fourth consonants are identical (e.g., *fɛlalɛṭ* 'knew repeatedly', root: *flṭ*) and a four-consonant root (*CCaCC* root pattern) in the simple derivational category if they are not (e.g., *kɛlakɛl* 'prevented', root *klakl*).

As discussed in Section 1.2, one of the familiar approaches to this problem, that of Beesley and Karttunen (2003), precompiles all of the combinations of roots and derivational patterns into stems. The problem with this approach for Tigrinya is that we do not have anything like a complete list of roots; that is, we expect many stems to be novel and will need to be able to analyze them on the fly. The other two approaches discussed in 1.2, that of Kiraz (2000) and that of Cohen-Sygal & Wintner (2006), are closer to what is proposed here. Each has an explicit mechanism for keeping the root and pattern distinct: separate tapes in the case of Kiraz (2000) and separate memory registers in the case of Cohen-Sygal & Wintner (2006).

The present approach also divides the work of processing the root and the derivational patterns between two components of the system. However, instead of the additional overhead required for implementing a multi-tape system or registers, this system makes use of the FSTs weighted with FSs that are already motivated for other aspects of morphology, as argued above. In this approach, the lexical aspects of morphology are handled by the ordinary input-output character correspondences, and the grammatical aspects of morphology, in particular the derivational patterns, are handled by the FS weights on the FST arcs and the unification that takes place as accumulated weights are matched against the weights on FST arcs.

As explained in Section 2, we can represent the eight possible derivational categories for a Tigrinya verb stem in terms of four binary features (ps, tr, rc, it). Each of these features is reflected more or less directly in the stem form (though differently for different root classes and for different TAM categories). However, they are sometimes distributed across the stem: different parts of a stem may be constrained by the presence of a particular feature. For example, the feature +ps (abbreviating [ps=True]) causes the gemination of the stem-initial consonant under various circum-

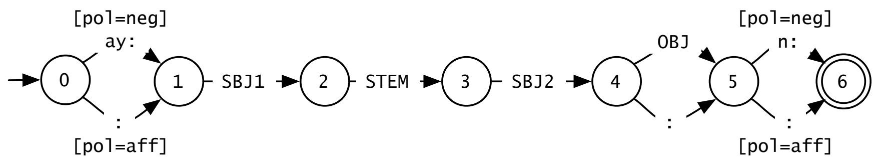

Figure 1: Handling Tigrinya (non-subordinate, imperfective) negation using feature structure weights. Arcs with uppercase labels represents subnetworks that are not spelled out in the figure.

stances and also controls the final vowel in the stem in the imperfective, and the feature +tr is marked by the vowel *a* before the first root consonant and, in the imperfective, by the nature of the vowel that follows the first root consonant (*ε* where we would otherwise expect *ɨ*, *ɨ* where we would otherwise expect *ε*.) That is, as with the verb affixes, there are long-distance dependencies within the verb stem.

Figure 2 illustrates this division of labor for the portion of the stem FST that covers the *CC_C* root pattern for the imperfective. This FST (including the subnetwork not shown that is responsible for the reduplicated portion of the +it patterns) handles all eight possible derivational categories. For the root √*fṣm* 'finish', the stems are [-ps,-tr,-rc,-it]: *fɨṣ_ɨm*, [+ps,-tr,-rc,-it]: *fɨṣ_εm*, [-ps,+tr,-rc,-it]: *afεṣ_ɨm*, [-ps,-tr,-rc,+it]: *fεṣaṣ_ɨm*, [+ps,-tr,+rc,-it]: *f_aṣ_εm*, [-ps,+tr,+rc,-it]: *af_aṣ_ɨm*, [+ps,-tr,-rc,+it]: *f_εṣaṣ_εm*, [-ps,+tr,-rc,+it]: *af_εṣaṣ_ɨm*. What is notable is the relatively small number of states that are required; among the consonant and vowel positions in the stems, all but the first are shared among the various derivational categories.

Of course the full stem FST, applying to all combinations of the eight root classes, the eight derivational categories, and the four TAM categories, is much larger, but the FS weights still permit a good deal of sharing, including sharing across the root classes and across the TAM categories.

4.3 Architecture

The full verb morphology processing system (see Figure 3) consists of analysis and generation FSTs for both orthographic and phonemically represented words, four FSTs in all. Eleven FSTs are composed to yield the phonemic analysis FST (denoted by the dashed border in Figure 3), and two additional FSTs are composed onto this FST to yield the orthographic FST (denoted by the large solid rectangle). The generation FSTs are created by inverting the analysis FSTs. Only the orthographic FSTs are discussed in the remainder of this paper.

At the most abstract (lexical) end is the heart of the system, the morphotactic FST, and the heart of this FST is the stem FST described above. The stem FST is composed from six FSTs, including three that handle the morphotactics of the stem, one that handles root constraints, and two that handle phonological processes that apply only to the stem. A prefix FST and a suffix FST are then concatenated onto the composed stem FST to create the full verb morphotactic FST. Within the whole FST, it is only the morphotactic FSTs (the yellow rectangles in Figure 3) that have FS weights.[2]

In the analysis direction, the morphotactic FST takes as input words in an abstract canonical form and an initial weight of TOP; that is, at this point in analysis, no grammatical information has been extracted. The output of the morphotactic FST is either the empty list if the form is unanalyzable, or one or more analyses, each consisting of a root string and a fully specified grammatical description in the form of an FS. For example, given the form *'aytɨfɨl_eṭun*, the morphotactic FST would output the root *flṭ* and the FS [tam=imprf, der=[+ps,-tr,-rc,-it], sbj=[+2p,+plr,-fem], +neg, obj=nil, -rel] (see Figure 3). That is, this word represents the imperfective, negative, non-relativized passive of the verb root √*flṭ* ('know') with second person plural masculine subject and no object: 'you (plr., mas.) are not known'. The system has no actual lexicon, so it outputs all roots that are compatible with the input, even if such roots do not exist in the language. In the generation direction, the opposite happens. In this case, the input root can be any legal sequence of characters that matches one of the eight

[2]The reduplication that characterizes [+it] stems and the "anti-reduplication" that prevents sequences of identical root consonants in some positions are handled with separate transitions for each consonant pair.

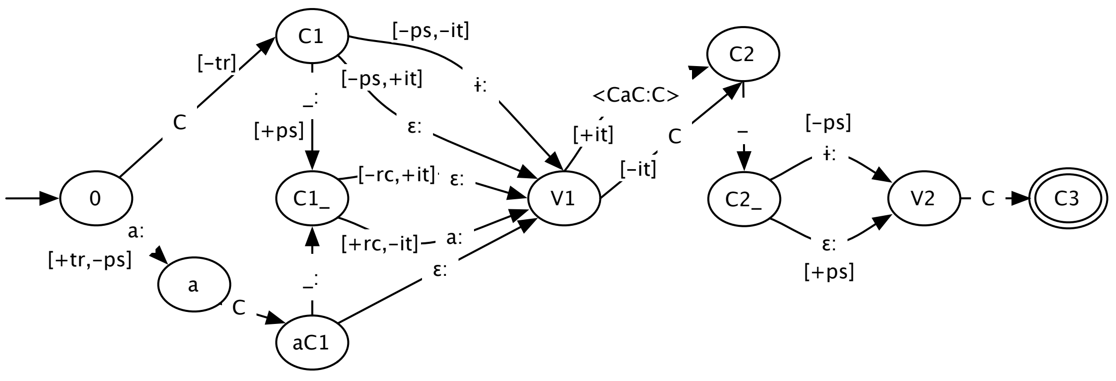

Figure 2: FST for imperfective verb stems of root type *CC_C*. <CaC:C> indicates a subnetwork, not shown, which handles the reduplicated portion of +it stems, for example, *feṣaṣ_im*.

root patterns (there are some constraints on what can constitute a root), though not necessarily an actual root in the language.

The highest FST below the morphotactic FST handles one case of allomorphy: the two allomorphs of the relativization prefix. Below this are nine FSTs handling phonology; for example, one of these converts the sequence *aɨ* to *ε*. At the bottom end of the cascade are two orthographic FSTs which are required when the input to analysis or the output of generation is in standard Tigrinya orthography. One of these is responsible for the insertion of the vowel *ɨ* and for consonant gemination (neither of which is indicated in the orthography); the other inserts a glottal stop before a word-initial vowel.

The full orthographic FST consists of 22,313 states and 118,927 arcs. The system handles verbs in all of the root classes discussed by Leslau (1941), including those with laryngeals and semivowels in different root positions and the three common irregular verbs, and all grammatical combinations of subject, object, negation, relativization, preposition, and conjunction affixes.

For the orthographic version of the analyzer, a word is entered in Ge'ez script (UTF-8 encoding). The program romanizes the input using the SERA transcription conventions (Firdyiwek and Yaqob, 1997), which represent Ge'ez characters with the ASCII character set, before handing it to the orthographic analysis FST. For each possible analysis, the output consists of a (romanized) root and a FS set. Where a set contains more than one FS, the interpretation is that any of the FS elements constitutes a possible analysis. Input to the generator consists of a romanized root and a single feature

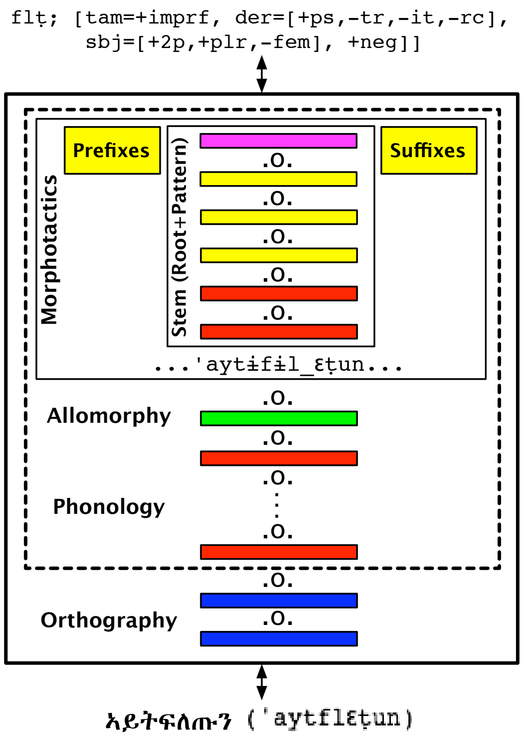

Figure 3: Architecture of the system. Rectangles represent FSTs, ".o."composition.

structure. The output of the orthographic generation FST is an orthographic representation, using SERA conventions, of each possible form that is compatible with the input root and FS. These forms are then converted to Ge'ez orthography.

The analyzer and generator are publicly accessible on the Internet at `www.cs.indiana.edu/cgi-pub/gasser/L3/morpho/Ti/v`.

4.4 Evaluation

Systematic evaluation of the system is difficult since no Tigrinya corpora are currently available. One resource that is useful, however, is the Tigrinya word list compiled by Biniam Gebremichael, available on the Internet at www.cs.ru.nl/ biniam/geez/crawl.php. Biniam extracted 227,984 distinct wordforms from Tigrinya texts by crawling the Internet. As a first step toward evaluating the morphological analyzer, the orthographic analyzer was run on 400 wordforms selected randomly from the list compiled by Biniam, and the results were evaluated by a human reader.

Of the 400 wordforms, 329 were unambiguously verbs. The program correctly analyzed 308 of these. The 21 errors included irregular verbs and orthographic/phonological variants that had not been built into the FST; these will be straightforward to add. Fifty other words were not verbs. The program again responded appropriately, given its knowledge, either rejecting the word or analyzing it as a verb based on a non-existent root. Thirteen other words appeared to be verb forms containing a simple typographical error, and I was unable to identify the remaining eight words. For the latter two categories, the program again responded by rejecting the word or treating it as a verb based on a non-existent root.

To test the morphological generator, the program was run on roots belonging to all 21 of the major classes discussed by Leslau (1941), including those with glottal or pharyngeal consonants or semivowels in different positions within the roots. For each of these classes, the program was asked to generate all possible derivational patterns (in the third person singular masculine form). In addition, for a smaller set of four root classes in the simple derivational pattern, the program was tested on all relevant combinations of the subject and object affixes[3] and, for the imperfective and perfective, on 13 combinations of the relativization, negation, prepositional, and conjunctive affixes. For each of the 272 tests, the generation FST succeeded in outputting the correct form (and in some cases a phonemic and/or orthographic alternative).

In conclusion, the orthographic morphological analyzer and generator provide good coverage of Tigrinya verbs. One weakness of the present system results from its lack of a root dictionary. The analyzer produces as many as 15 different analyses of words, when in many cases only one contains a root that exists in the language. The number could be reduced somewhat by a more extensive filter on possible root segment sequences; however, root internal phonotactics is an area that has not been extensively studied for Tigrinya. In any case, once a Tigrinya root dictionary becomes available, it will be straightforward to compose a lexical FST onto the existing FSTs that will reject all but acceptable roots. Even a relatively small root dictionary should also permit inferences about possible root segment sequences in the language, enabling the construction of a stricter filter for roots that are not yet contained in the dictionary.

5 Conclusion

Progress in all applications for a language such as Tigrinya is held back when verb morphology is not dealt with adequately. Tigrinya morphology is complex in two senses. First, like other Semitic languages, it relies on template morphology, presenting unusual challenges to any computational framework. This paper presents a new answer to these challenges, one which has the potential to integrate morphological processing into other knowledge-based applications through the inclusion of the powerful and flexible feature structure framework. This approach should extend to other Semitic languages, such as Arabic, Hebrew, and Amharic. Second, Tigrinya verbs are simply very elaborate. In addition to the stems resulting from the intercalation of eight root classes, eight derivational patterns and four TAM categories, there are up to four prefix slots and four suffix slots; various sorts of prefix-suffix dependencies; and a range of interacting phonological processes, including those sensitive to syllable structure, as well as segmental context. Just putting together all of these constraints in a way that works is significant. Since the motivation for this project is primarily practical rather than theoretical, the main achievement of the paper is the demonstration that, with some effort, a system can be built that actually handles Tigrinya verbs in great detail. Future work will focus on fine-tuning the verb FST, developing an FST for nouns, and applying this same approach to other Semitic languages.

[3]With respect to their morphophonological behavior, the subject affixes and object suffixes each group into four categories.

References

Saba Amsalu and Girma A. Demeke. 2006. Non-concatenative finite-state morphotactics of Amharic simple verbs. *ELRC Working Papers*, 2(3).

Jan Amtrup. 2003. Morphology in machine translation systems: Efficient integration of finite state transducers and feature structure descriptions. *Machine Translation*, 18:213–235.

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford, CA, USA.

Eugene Buckley. 2000. Alignment and weight in the Tigrinya verb stem. In Vicki Carstens and Frederick Parkinson, editors, *Advances in African Linguistics*, pages 165–176. Africa World Press, Lawrenceville, NJ, USA.

Bob Carpenter. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press, Cambridge.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English*. Harper and Row, New York.

Yael Cohen-Sygal and Shuly Wintner. 2006. Finite-state registered automata for non-concatenative morphology. *Computational Linguistics*, 32:49–82.

Ann Copestake. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford, CA, USA.

Yitna Firdyiwek and Daniel Yaqob. 1997. The system for Ethiopic representation in ascii. URL: citeseer.ist.psu.edu/56365.html.

C. Douglas Johnson. 1972. *Formal Aspects of Phonological Description*. Mouton, The Hague.

Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–378.

Lauri Karttunen, Ronald M. Kaplan, and Annie Zaenen. 1992. Two-level morphology with composition. In *Proceedings of the International Conference on Computational Linguistics*, volume 14, pages 141–148.

George A. Kiraz. 2000. Multitiered nonlinear morphology using multitape finite automata: a case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105.

Kimmo Koskenniemi. 1983. Two-level morphology: a general computational model for word-form recognition and production. Technical Report Publication No. 11, Department of General Linguistics, University of Helsinki.

Wolf Leslau. 1941. *Documents Tigrigna: Grammaire et Textes*. Libraire C. Klincksieck, Paris.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2000. Weighted finite-state transducers in speech recognition. In *Proceedings of ISCA ITRW on Automatic Speech Recognition: Challenges for the Millenium*, pages 97–106, Paris.