

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

Michael Gasser

Indiana University / Computer Science Department  
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 structures provide an efficient way of handling the templatic morphology that characterizes Tigrinya verb stems as well as the complex Tigrinya verb morphotactics.

## 1 Introduction

### 1.1 Finite state morphology

Morphological analysis is the segmentation of words into their component morphemes and (usually) the assignment of grammatical morphemes to grammatical categories and the assignment of the lexical morpheme to a particular lexeme or lemma. For example, the English adjective *prettier* could be analyzed as pretty+COMPARATIVE. Morphological generation is the reverse process. Both processes relate a **surface** level to a **lexical** level. The relationship between these levels has been the concern of 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 related 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. That is, if FST  $f_1$  outputs string  $s_2$  for input string  $s_1$ , and FST  $f_2$  outputs string  $s_3$  for input string  $s_2$ , then the FST which is the composition of  $f_1$  and  $f_2$  will output  $s_3$  for input  $s_1$ . A second important advance in computational morphology was the recognition by Karttunen et al. (Karttunen et al., 1992) that a cascade of composed FSTs could implement the two-level model. This opened up the possibility of quite complex finite state systems including ordered **alternation rules** representing context-sensitive variation in the phonological or orthographic shape of morphemes,

---

*Informatics for the Democratization of Knowledge, Occasional Paper #1, Indiana University School of Informatics, Bloomington, IN, USA.*

the **morphotactics** characterizing the possible sequences of morphemes (in canonical form) for a given word class, and one or more **sub-lexicons**. 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. This is illustrated in Figure 1. Figure 2 shows one of the FSTs in the cascade.

## 1.2 Finite state approaches to non-concatenative morphology

These ideas have revolutionized computational morphology, in the process making languages with relatively complex morphology, 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**, which is familiar from 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 within the word; 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.<sup>1</sup> For example, the Tigrinya verb root  $\sqrt{\text{sbr}}$  ‘break’ can be combined with some seven different patterns to form the stems that actually occur within verb wordforms; here are some examples of stems: *seber*<sup>2</sup> ‘broke’, *sib\_er* ‘is broken’, *s\_ebaber* ‘break one another’, *asbir* ‘having caused to break’.

A number of approaches that are still within the finite state framework have been proposed to deal with Semitic template morphology. The most familiar ones are characterized by the explicit separation of roots and patterns into distinct sublexica and devices for combining a root and pattern together during processing. One approach is to make use of separate tapes for root and pattern at the lexical level (Kiraz, 2000). A transition in the FSTs that are compiled from 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

<sup>1</sup>For some languages, the pattern is often viewed as further consisting of a template with positions for consonants and vowels and a separate “vocalism”, consisting of one or more vowels that are inserted into the template. We will not be concerned with this three-way root-template-vocalism distinction in this paper, which is not particularly useful for Ethio-Semitic languages such as Tigrinya in any case.

<sup>2</sup>In this paper I use *i* for the high central vowel of Tigrinya, *e* 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.

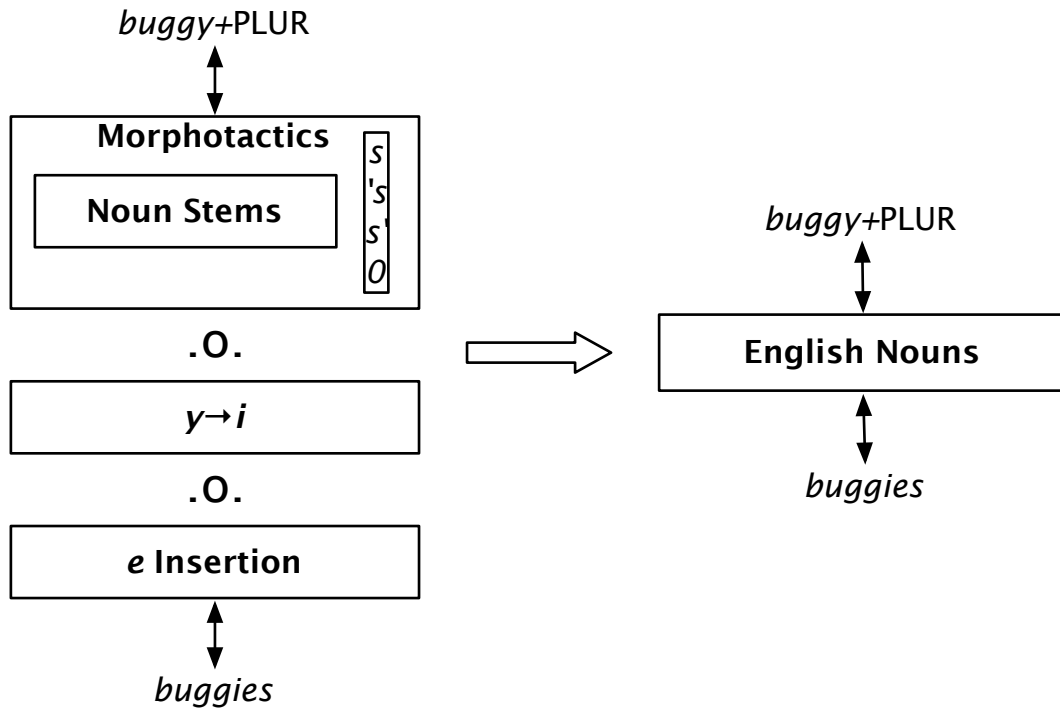


Figure 1: Composition of FSTs for English noun morphology. “.o.” denotes the composition operation.

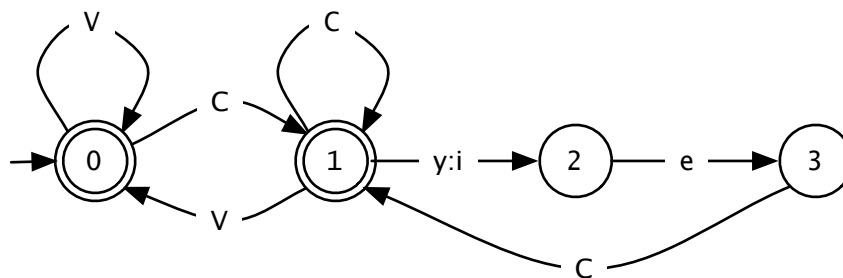


Figure 2: FST which converts surface *y* to lexical *i* following a consonant and before an *e*. On each arc the input character appears to the left of the colon, the output character to the right. The arc label “a:b” means that input character “a” corresponds to output character “b”. The labels “C” and “V” mean that an input consonant corresponds to the same output consonant and an input vowel corresponds to the same output vowel, respectively.

language Amharic by Saba and Girma (2006). Amharic is characterized by all of the same sorts of complexity as Tigrinya; this complexity is the topic of the next section.

This paper proposes a different approach, one that starts with another 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

### 2.1 Dimensions of variation

Tigrinya is an Ethio-Semitic language spoken by 4.5–7 million people in northern Ethiopia and central Eritrea. Given the number of speakers (about the same as that for Finnish or Danish) and its role as the most significant language in Eritrea, Tigrinya clearly belongs to the set of severely under-resourced languages. There has been almost no computational work on the language, and there are effectively no corpora or usable digitized dictionaries. 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.

The morphology of the Tigrinya verb (Leslau, 1941 is a standard reference) can be described in terms of six orthogonal dimensions: root (the only purely lexical component), tense-aspect-mood, derivational category, subject agreement, object agreement, and polarity. With respect to the dimension of TENSE-ASPECT-MOOD (TAM), there are four possibilities: the three categories familiar from other Semitic languages, traditionally referred to as PERFECTIVE (PRF), IMPERFECTIVE (IMF), and JUSSIVE-IMPERATIVE (J-I), and a fourth common to several other Ethio-Semitic languages, conventionally called GERUNDIVE (GER). Within each of

these four categories, the realizations of each of the other dimensions differ significantly, so it is useful (and conventional) to treat the categories separately. The distinctions are illustrated in the following example, for which the values on the other dimensions are identical: PRF: *fələta* ‘he knew her’, IMF: *yifəta* ‘he knows her’, J-I: *yifləta* ‘that he know her’, GER: *fəlitəw.a* ‘he knowing her’. Wordforms within these basic four TAM categories combine with auxiliaries to yield the full range of possibilities in the complex Tigrinya tense-aspect-mood system. Since these auxiliaries are written as separate words, or in the case of deleted initial vowels, separated from the main verbs by an apostrophe, they will not be discussed further.

In this paper, we will only be concerned with the Tigrinya imperfective, the most complex of the four TAM categories. This form, alone or in combination with auxiliaries, conveys various present and future tense possibilities. An imperfective verb consists of a verb stem, one or more prefixes and zero or more suffixes. The stem, as in other Semitic languages, consists of a verb root appearing in one or another of a set of patterns. A verb ROOT in Tigrinya, the second dimension of variation in the verb, is made up of a sequence of three, four, or five consonants. In addition, as in other Ethio-Semitic languages, certain roots have inherent vowels and/or gemination (lengthening) of particular consonants. Thus among the three-consonant roots, there are three subtypes: *CCC*, *CaCC*, *CC.C*. Roots are described in more detail in Section 4.2 below.

As we have seen, the combining of a root and a pattern can be viewed as the “intercalation” of pattern vowels and the copying of root consonants in particular positions. In Tigrinya, and other Ethio-Semitic languages, these patterns represent different DERIVATIONAL possibilities, the third dimension along which verbs vary. A Tigrinya verb

root can appear in up to eight different derivational patterns within each of the TAM categories. The patterns can be characterized in terms of four binary features, which I will refer to as PASSIVE-REFLEXIVE (ps), TRANSITIVE/CAUSATIVE (tr), ITERATIVE (it), and RECIPROCAL (rc). The eight possible combinations, along with their rough semantic correlates and the corresponding imperfective stem of the root  $\sqrt{\text{ft}}$  are as follows. 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 verbs.

1. simple: [-ps,-tr,-it,-rc], *fəlt* ‘know’
2. passive or reflexive: [+ps,-tr,-it,-rc], *fil.ət* ‘be known’
3. causative or transitive: [-ps,+tr,-it,-rc], *afliṭ* ‘cause to know’
4. frequentative: [-ps,-tr,+it,-rc], *fəlaliṭ* ‘know repeatedly’
5. reciprocal 1: [+ps,-tr,+it,-rc], *f.ələləṭ* ‘know one another’
6. causative reciprocal 1: [-ps,+tr,+it,-rc], *af.ələliṭ* ‘cause to know one another’
7. reciprocal 2: [+ps,-tr,-it,+rc], *f.ələṭ* ‘know one another’
8. causative reciprocal 2: [-ps,+tr,-it,+rc], *af.əliṭ* ‘cause to know one another’

All finite Tigrinya verbs must agree with their subjects in person, number, and (for 2nd and 3rd persons) gender; SUBJECT thus represents a fourth dimension along which verbs vary. There are 10 combinations of person, number, and gender in the Tigrinya pronominal and verb-agreement system. For imperfective verbs, as in other Semitic languages, subject agreement takes the form of prefixes and for some person-number-gender combinations (2nd person singular feminine, 2nd person plural, 3rd person plural) also suffixes. The follow illustrate some of the subject affixes the simple form of the root  $\sqrt{\text{ft}}$  ‘know’:

*ifəl.iṭ* ‘I know’, *tifəlṭi* ‘you (sg., fem.) know’, *yifəlṭa* ‘they (fem.) know’.

A transitive Tigrinya verb may also include an object suffix (or object agreement marker), which appears in one of the same set of 10 possible combinations of person, number, and gender as the subject agreement markers. OBJECT thus represents the fifth dimension of variability within Tigrinya verbs. 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. The following illustrate some of the possible object suffixes for the simple form of the root  $\sqrt{\text{ft}}$  ‘know’ with a 3rd person singular masculine subject: *yifəlṭən.i* ‘he knows me’, *yifəlṭəl.ey* ‘he knows for me’, *yifəlṭa* ‘he knows her’, *yifəlṭəl.a* ‘he knows for her’.

The sixth dimension of variation of a Tigrinya verb is its POLARITY; it is either AFFIRMATIVE or NEGATIVE: *yifəlṭən.i* ‘he knows me’; *ay.ifəlṭən.in* ‘he doesn’t know me’.

In addition to the six basic dimensions mentioned so far (TAM, ROOT, DERIVATIONAL PATTERN, SUBJECT, OBJECT, POLARITY), an imperfective Tigrinya verb may take the RELATIVIZING prefix (z)i-, for example, *zifəlṭən.i* ‘(he) who knows me’. The relativizer can in turn be preceded by one of a set of six PREPOSITIONS, for example, *kabzifəlṭən.i* ‘from him who knows me’. Finally, there is the possibility of one of three CONJUNCTIONS at the beginning of the verb (without the relativizer), for example, *kifəlṭən.i* ‘so that he knows me’.

Given the eight possible DERIVATIONAL patterns, the 170 possible combinations of SUBJECT agreement affixes with the two sets of OBJECT agreement suffixes, the two possible values each of POLARITY and RELATIVIZATION, and the nine possible PREPOSITIONAL or CONJUNCTIVE prefixes, each Tigrinya verb root may appear in as many as 31,280 different imperfective wordforms (and

of course many more in the three other TAM categories).

## 2.2 The stem

Most of the complexity of a Tigrinya verb is centered on the stem. We have seen how the stem can be seen as the result of the non-concatenative process of intercalation of a root with a derivational template. Tigrinya verb roots group into about eight conjugation classes, each with its own set of derivational patterns. Three of the eight templates feature reduplication of the penultimate consonant (*-felaliṭ*, *-f.elaleṭ*, *-af.elaliṭ*), and “anti-reduplication” appears in several of root classes (for example, the second and third consonants in the four-consonant roots must be distinct). 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. Finally, as in other Semitic languages, the presence of “laryngeal” (glottal or pharyngeal) consonants or semivowels within a root leads to some changes in the derivational patterns (most of which can be captured in phonological rules).

## 2.3 Dependencies and ambiguity

The morphotactics of the Tigrinya imperfective is replete with dependencies which span the verb stem.

- Negation in non-subordinate verbs is signaled by the circumfix *ay-n*.
- Subordination, which is indicated by the relativizing prefix, with or with preceding prepositions, or the conjunction *ni/nixi*, precludes the negative suffix *-n*.
- The 1st person subject agreement prefixes *i/ni* preclude the 2nd person and 3rd person plural subject agreement suffixes *a/u*.
- The 2nd person singular feminine subject

agreement suffix *i* can only appear with the 2nd person subject agreement prefix *ti*.

- 1st person subject agreement prefixes preclude 1st person object suffixes, and 2nd person subject agreement prefixes preclude 2nd person object suffixes.

There is some ambiguity in the system. For example, the forms for 2nd person masculine singular and 3rd person feminine singular subject agreement are identical (*tifeL-iṭ* ‘you (masc., sing.) know; she knows’) and the 2nd person and 3rd person feminine plural subject suffix is identical to one allomorph of the 3rd person feminine singular object (*yifeṭa* ‘he knows her; they (fem.) know’).

Tigrinya is written in the Ethiopic (Geez) syllabary, which fails to distinguish between syllable final consonants and consonants followed by the vowel *i* and fails to mark consonant gemination. In the absence of lexical information, this leads to additional ambiguity. For example, the orthographic form that would be romanized as *'nfeṭ* has different interpretations depending on whether it is pronounced *'in-ifil-εṭ* ‘which we  $\sqrt{\text{fl}}$  + PASSIVE’ or *'iniflet* ‘I  $\sqrt{\text{nfl}}$  + PASSIVE’ (though only the former root actually exists).

Finally, a number of alternation rules are needed to specify various phonological and orthographic regularities. For example, the vowel sequence *ia*, which may occur at the boundary between two prefixes or between a prefix and a stem, is realized as  $\epsilon$ : *ti+afil-iṭ*  $\rightarrow$  *tefil-iṭ* ‘she causes to know’.

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

### 3 FSTs with Feature Structures

#### 3.1 Weighted FSTs

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 combined with “addition”. Weighted FSTs are closed under the same operations as unweighted FSTs; in particular for our purposes, 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.

#### 3.2 Feature structures as weights

Amtrup (2003) recognized the advantages that would accrue to morphological analyzers and generators if they could accommodate structured representations. One productive and popular approach to representing linguistic structure is **typed 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 *gezay* ‘my house’ as [lexeme=*geza*, number=singular, possessor=[person=1, number=singular]]. The basic operation over FSs is **unification**. Loosely speaking, two FSs unify if their attribute-values pairs are compatible, and the resulting unification combines the features of the FSs. For example, the two FSs [lexeme=*geza*, number=singular] and [possessor=[person=1, number=singular]] unify to

yield the FS [lexeme=*geza*, number=singular, possessor=[person=1, number=singular]]. The distinguished FS TOP unifies with any other FS. Variables in FSs permit a straightforward implementation of agreement and other constraints within linguistic units (though these will not concern us further in this paper). A **typed** feature structure has a type from which it inherits a set of default feature values.

Amtrup shows that sets of typed 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 the arc. Consider again the simple example of singular Tigrinya nouns. The network shown in Figure 3 will analyze nouns whose stems consist of sequences of consonants and vowels (but without multiple vowels in succession), ending in a vowel, with or without the accusative prefix *ni* and one of the possessive suffixes *y* (first person singular) or *xi* (second person singular feminine).<sup>3</sup> For example, given the input string *nigezay* and TOP as the initial weight, the network yields as output string *geza* and as output weight [case=acc, poss=[prs=1, num=sng]]. Because any FST (including any weighted FST) can be inverted, we can utilize the network in the reverse direction for generation. For example, given the input string *dim.u* and the initial

<sup>3</sup>For all of the examples mentioned in this paper, each FS set weight on an arc is a singleton, so for simplicity the braces indicating the set are omitted from the figures and examples in the text.

weight [case=nom, poss=[prs=2, num=sng, gen=fem]], the inverted network would yield the output string *dim\_xi* and the initial weight unchanged as output.

Using examples from Persian, Amtrup demonstrates two advantages of FSTs weighted with feature structure sets. First, long-distance dependencies within words present notorious problems for finite state techniques. Such constraints are common in languages with complex morphology, including Tigrinya, as we have seen. 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 feature structures, 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 as machine translation, work with the sort of structured representations that are elegantly handled by feature structure descriptions. Thus it is often desirable to have the output of a morphological analyzer have this richness, in contrast to the string representations that are the output of an unweighted finite state analyzer.

## 4 Processing Tigrinya Verbs Using FSTs Weighted with Feature Structures

### 4.1 Long-distance dependencies

As we have seen, imperfective Tigrinya verbs feature various sorts of long-distance dependencies. The circumfix that marks the negative of non-subordinate verbs, *ay...n*, is one example. Figure 4 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\_ifeḷtun* ‘they don’t know’ and *yifeḷtu* ‘they know’ and reject forms such as *ay\_ifeḷtu*. In the generation direction, the FST will correctly generate a form such as *ay\_ifeḷtun* given a initial FS that includes the feature [pol=neg].

### 4.2 Stems: root and derivational pattern

Now consider the heart of the Tigrinya verb and the source of most of its complexity, the stem. The stem conveys two types of information: lexical (the root of the verb) and derivational (one of the eight derivational categories described in Section 2 above). Analysis of a stem should return the root and the derivational category; generation should start with a root and a derivational category and return a stem. It is possible to group Tigrinya roots into eight categories which cover all but a few exceptional cases. We can represent each root as a sequence of consonants, separated in some cases by the vowel *a* and in one case by the gemination character.

These root patterns, and examples of each, are as follows:

1. *CCC*, *sbr* ‘break’
2. *CC\_C*, *dq\_s* ‘sleep’
3. *CaCC*, *bark* ‘bless’
4. *CCCC*, *mskr* ‘testify’
5. *CCaCC*, *klakl* ‘prevent’
6. *CCCCC*, *nqtqt* ‘tremble’
7. *CCCaCC*, *nslalw* ‘be shaded’
8. *CC|aCC*, *ns|aff* ‘float’

The extra character (|) in pattern 8 is necessary to distinguish it from pattern 5, which has the same sequence of segments but interacts differently with the derivational patterns. In pattern 8, the two initial consonants are never separated by a vowel; in pattern 5, they are always separated by the vowel *ε*. Further sub-

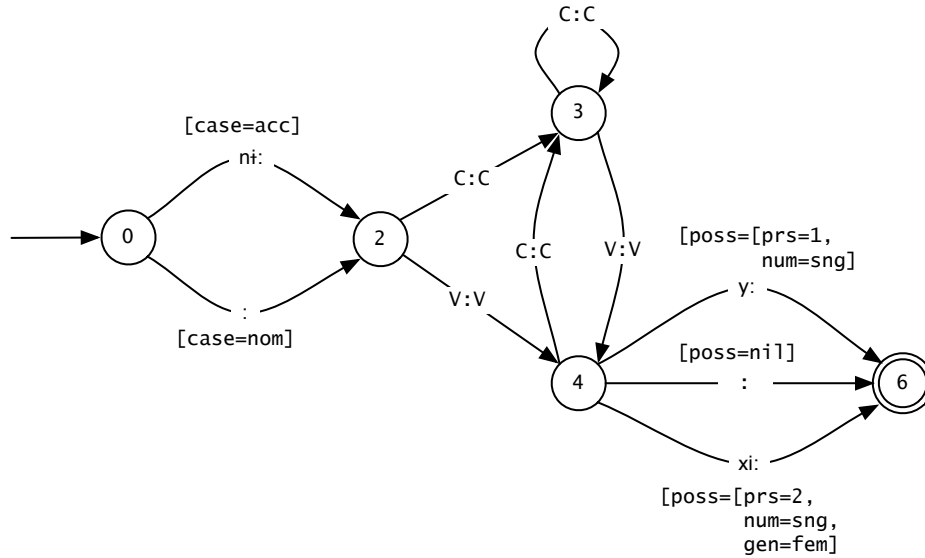


Figure 3: FST weighted with feature structures for analyzing a small subset of Tigrinya nouns. Feature structure weights are indicated in brackets next to the corresponding arcs. Arcs with no weights shown have weight TOP.

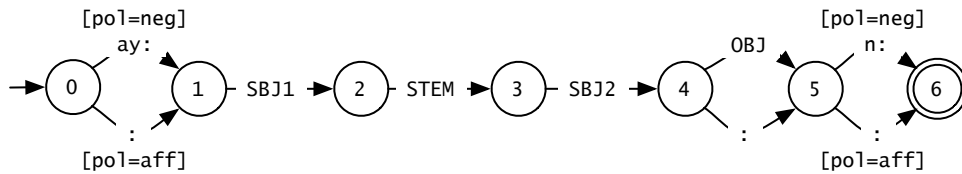


Figure 4: Handling Tigrinya (non-subordinate, imperfective) negation using feature structure weights. Arcs with uppercase labels represents subnetworks that are not spelled out in the figure.

divisions within the eight patterns derive from the presence of laryngeal consonants (*h*, *ħ*, *ʕ*, *ʔ*) and/or semivowels (*w*, *y*) in one or another position. These variations can be handled by phonological/orthographic alternation rules.

Given a particular derivational pattern, extracting the root from the stem is a straightforward matter with an FST. For example, for the passive pattern, root pattern 2 (*CC\_C*) appears in the template *CiC\_εC*, 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 iterative form of a three-consonant root if the third and fourth consonants are identical (*felaliṭ*) and a four-consonant root (*CCaCC* root pattern) in the simple derivational pattern if they are not (*kelakil*).

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 don't 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 or generate them on the

fly. Another approach, that of Kiraz (2000), is closer to what is proposed here. He processes the root and patterns using separate tapes, each working with its own sublexicon.

As in Kiraz's system, the approach used here 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, this system makes use of the existing FSTs weighted with FSs that are already motivated for aspects of morphology, as argued above. In this approach, the lexical aspects of verb morphology are handled by the ordinary input-output character correspondences that are part of all FSTs, and the grammatical aspects of verb morphology, including the derivational patterns with the verb stems, 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. Unlike Kiraz's system, this system also does not require an explicit root lexicon.

As explained in Section 2, we can represent the eight possible derivational categories in 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 in some cases differently for different root categories). However, in some cases, they are 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 circumstances and also controls the final vowel in the stem, and the feature *+tr* is marked by the vowel *a* before the first root consonant and by the nature of the vowel that follows the first root consonant (*ε* where we would otherwise expect *i*, *i* where we would otherwise expect *ε*.) That is, as with the verb prefixes and suffixes, there are long-distance dependencies within the verb stem.

Figure 5 illustrates this division of labor for the portion of the stem FST that is responsible for root pattern 2, *CC\_C*. This FST (including the subnetwork not shown that is responsible for the reduplicated portion of the iterative patterns) handles all eight of the possible derivational patterns. For the root  $\sqrt{f\dot{s}m}$  'finish', the stems are simple ([-ps,-tr,-it,-rc]): *fiṣ\_im*, passive/reflexive: *fiṣ\_εm*, causative/transitive: *afεṣ\_im*, frequentative: *feṣaṣ\_im*, reciprocal 1: *f\_aṣ\_εm*, causative reciprocal 1: *af\_aṣ\_im*, reciprocal 2: *f\_εṣaṣ\_εm*, causative reciprocal 2: *af\_εṣaṣ\_im*. 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 patterns (with the extra consonant and vowel that are part of the the three +it patterns sharing the additional states in the reduplication subnetwork).

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

### 4.3 Architecture

The full morphological processing system consists of four FSTs, two that analyze surface word forms and two that generate surface word forms, the surface forms appearing in orthographic representations in the one case, phonemic representations in the other. The analysis FSTs are the result of the composition of a cascade of 15 FSTs (13 for the phonemic FST and two additional for the orthographic FST). The generation FSTs are created by inverting the analysis FSTs. Figure 6 gives an overview of the entire architecture.

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 with two FSTs representing phonological processes that

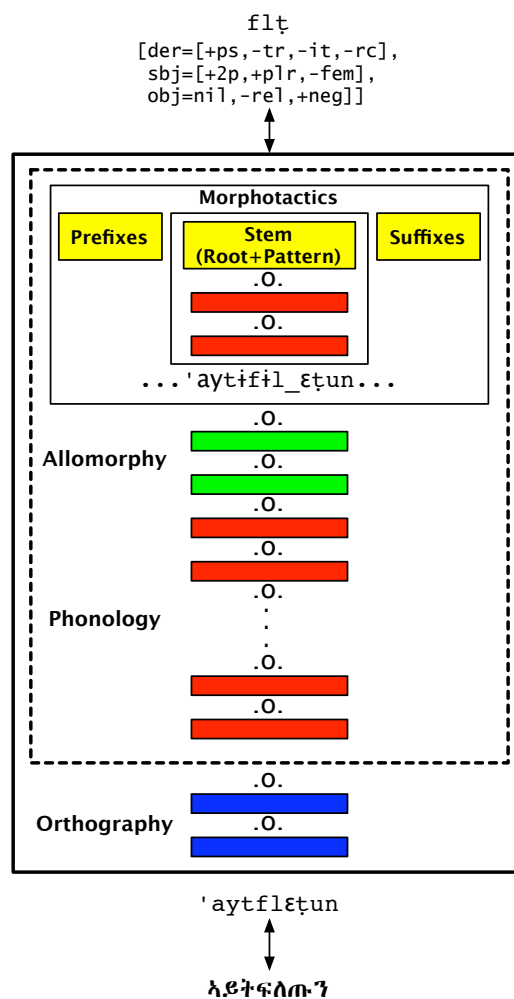


Figure 6: Architecture of the system. Each rectangle represents an FST. The large rectangle with the solid border is the orthographic FST, and the rectangle with the dashed border is the phonemic FST. “.o.” denotes composition. The example shown is the word ‘*aytifil\_εṭun*’ ‘you (plr., mas.) are not known’.

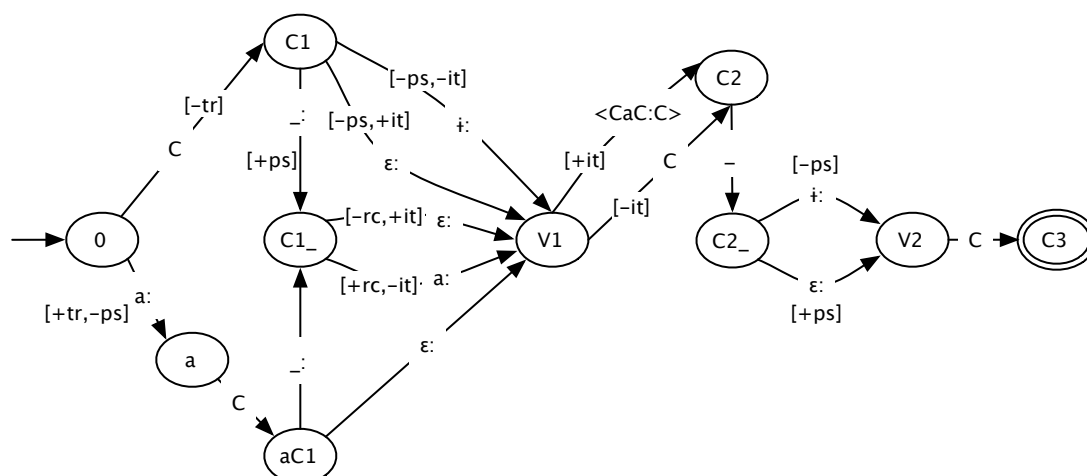


Figure 5: FST for imperfective verb stems of root type  $CC_C$ .  $\langle CaC:C \rangle$  indicates a subnetwork, not shown, which handles the reduplicated portion of  $+it$  stems, for example, *sebābir*,

apply only to the stem. A prefix FST and a suffix FST are then concatenated onto the composed stem FST to create the full morphotactic FST. In the analysis direction, this 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 *'aytifilLeṭun*, the morphotactic FST would output the root *ḥt* and the FS [der=[+ps,-tr,-rc,-it], sbj=[+2p,+plr,-fem], obj=nil, -rel, +neg] (see Figure 6). That is, this word represents the negative, non-relative passive of the verb root *ḥt* ('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. Given the current state of Tigrinya digital lexicography, this seems like a reasonable approach.

In the generation direction, the opposite

happens. In this case, the input root can be any legal sequence of characters (there are some constraints on what can constitute a root), not necessarily an actual root in the language.

The remaining 12 FSTs are responsible for the realization of several morphemes and for phonological or orthographic alternations. The two highest FSTs below the morphotactic FST handle allomorphy; for example, one of these is responsible for the two allomorphs of the relativization prefix. Below this are 10 FSTs handling phonology; for example, one of these converts the sequence *ai* to  $\epsilon$ . The most complex of the phonological FSTs is the metrical FST that governs gemination of certain consonants: the second consonant in root pattern 1 ( $CCC$ , the most common of the root patterns) and the initial consonant in most of the object suffixes. At the bottom end of the cascade are two orthographic FSTs which are not part of the overall phonemic FST but 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 *i*, which is not indicated in the orthography, the other for consonant gemination, also not indicated in the orthography.

The full phonemic FST consists of 7,865 states and 35,172 arcs, and the full orthographic FST consists of 8,097 states and 36,501 arcs. The system handles imperfective verbs in all of the root classes discussed by Leslau (1941), including those with laryngeals and semivowels in different root positions and the three genuinely irregular verbs.<sup>4</sup> It handles all grammatical combinations of subject and object affixes, negation affixes, the relativization prefix, prepositional prefixes, and conjunctive prefixes. It does not accommodate all of the possibilities that Leslau (1941) indicates for the realization of stem vowels for roots that include laryngeals and semivowels. It would be a simple matter to include these alternatives, but it would slow down generation.

#### 4.4 Implementation details

The program is written in Python. The FST module is based on one (author unidentified) that is part of the distribution for the Natural Language Toolkit (NLTK), an open-source suite of Python tools for computational linguistics applications (<http://nltk.sourceforge.net/>). The author corrected some bugs in the NLTK FST module and implemented a facility for handling weights on transitions, including FS weights. Composition of weighted FSTs has been implemented but not determinization and minimization, which could significantly reduce the number of states in the FSTs. Feature structures and unification are implemented as in the NLTK modules; the author added types and inheritance to these.

For the orthographic version of the analyzer, the user enters a word in Ethiopic script (UTF-8 encoding). The program romanizes the in-

<sup>4</sup>The only exception is the small set of verbs whose transitive-causative pattern begins with the root-external sequence *as* and whose passive pattern begins with a root-external *s*. This unproductive category was apparently borrowed from Amharic (Leslau, 1947). Since it is difficult to know how many verbs belong to this category, they have been left out of the system in its current form.

put using the SERA transcription conventions (Firdyiwek and Yaqob, 1997), which represent Ethiopic characters using 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 feature structure set. Where a set contains more than one feature structure, the interpretation is that any of the feature structure elements constitutes a possible analysis.

Input to the phonemic version of the analyzer conforms to SERA conventions, except that the vowel *i*, not normally represented in SERA, is always indicated (by *I*), and gemination, also unmarked in SERA, is always indicated (by the *\_* character). Output is as with the orthographic version.

Input to either version of the generator consists of a romanized root and a single feature structure. The program rejects roots which do not match one of the eight root patterns. 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 Ethiopic orthography. The output of the phonemic generation FST is a phonemic representation of each possible form, using SERA conventions, except as noted for analysis above.

#### 4.5 Evaluation

Systematic evaluation of the system is difficult since no Tigrinya corpora are currently available. Instead the program was tested in both directions on words belonging to all combinations of the root categories discussed by Leslau (1941), including not only the eight categories listed here but categories 1, 2 and 3 with laryngeals in any position, *w* or *y* in second or third position and categories 1 and 4 with labialized consonants in first, second, and third positions. For each of these categories, the program was asked to generate all possible derivational patterns and then to analyze the

resulting surface outputs. The program was also tested on all relevant combinations of the subject and object affixes<sup>5</sup> and on eight combinations of the relativization, negation, prepositional, and conjunctive prefixes. In all cases, the program was tested with both the phonemic and orthographic FSTs.

For each of the 320 tests, the generation FST succeeded in outputting the correct form (and in some cases a phonemic and/or orthographic alternative). In the analysis direction, there was much more ambiguity, with up to four different outputs. This is as it should be given the lack of a lexicon of roots in the system and the fact that for some derivational categories the same form is shared across multiple root categories. For example, the stem form  $C_1\_eC_2aC_2eC_3$  represents the reciprocal 2 form of verbs in root categories 1, 2 and 3. Though many of the roots that are output do not actually exist in the language, all of the results of analysis conform to the morphotactics and morphophonology of Tigrinya imperfective verbs. Figure 7 shows a screen dump with the program generating and analyzing the orthographic form of the word *yit.əhagageza* 'they (fem.) help one another'.

## 5 Conclusion

Progress in all computational linguistics applications for a language such as Tigrinya is held back to the extent that 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.

<sup>5</sup>With respect to their morphophonological behavior, the subject affixes and object suffixes each group into four categories.

Second, Tigrinya verbs are simply very elaborate. In addition to the stems resulting from the intercalation of eight root categories and eight derivational patterns, there are up to four prefix slots and three 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, I consider this the main achievement of this paper; it demonstrates that, with some effort, a system can be built that actually handles Tigrinya verbs in detail. Future work will focus on implementing the remainder of the verb system and noun morphology and on developing a Tigrinya corpus.

## References

- Amsalu, Saba and Girma A. Demeke. 2006. Non-concatenative finite-state morphotactics of Amharic simple verbs. *ELRC Working Papers*, 2(3).
- Amtrup, Jan. 2003. Morphology in machine translation systems: Efficient integration of finite state transducers and feature structure descriptions. *Machine Translation*, 18:213–235.
- Beesley, Kenneth R. and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford, CA, USA.
- Buckley, Eugene. 2000. Alignment and weight in the Tigrinya verb stem. In Carstens, Vicki and Frederick Parkinson, editors, *Advances in African Linguistics*, pages 165–176. Africa World Press, Lawrenceville, NJ, USA.
- Carpenter, Bob. 1992. *The Logic of Typed Feature Structures*. Cambridge University Press, Cambridge.
- Chomsky, Noam and Morris Halle. 1968. *The Sound Pattern of English*. Harper and Row, New York.
- Copestake, Ann. 2002. *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford, CA, USA.

```

>>> from ti_ipf import *
Creating feature structure type hierarchy for verbs
Loading FST for analysis of phonemically represented verbs
Loading FST for analysis of written verbs
Creating inverted FSTs for generation
>>> FS = ipf_features('ti') # Simple imperfect FS for Ti
>>> FS['der']['ps'] = FS['der']['it'] = True # Reciprocal 2
>>> FS['sbj']['plr'] = FS['sbj']['fem'] = True # Subject: 3p plur fem
>>> FS = FSSet(FS) # FS set from this FS
>>> gen_word('Hg_z', FS, written=True) # Generate word in Ti orthography
Generating Hg_z : Features: sbj:3prs,plr,fem; drv:pas/itr
-> Word form: ያተላገዛ || Features: sbj:3prs,plr,fem; drv:pas/itr
[['\xe1\x8b\xad\xe1\x89\xb0\xe1\x88\x93\xe1\x8c\x8b\xe1\x8c\x88\xe1\x8b\x9b', [%ti
_ipf cnj=nil, der=[%der +it, +ps, -rc, -tr], -neg, obj=[%obj -expl, -fem, -p1, -p2
, -plr, -prp], prp=nil, -rel, sbj=[%agr +fem, -p1, -p2, +plr], -sub]]]
>>> anal_word('ያተላገዛ', written=True) # Analyze the output
Analyzing ያተላገዛ
-> Root: Hg_z || Features: sbj:3prs,plr,fem; drv:pas/itr |
      sbj:3prs,sng,mas; obj:3prs,sng,fem; drv:pas/itr
      Root: Hagz || Features: sbj:3prs,plr,fem; drv:pas/itr |
      sbj:3prs,sng,mas; obj:3prs,sng,fem; drv:pas/itr
      Root: Hgz || Features: sbj:3prs,plr,fem; drv:pas/itr |
      sbj:3prs,sng,mas; obj:3prs,sng,fem; drv:pas/itr
[['Hg_z', [cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[-expl], prp=nil, -rel, sb
j=[+fem, -p1, -p2, +plr], -sub];[cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[+ex
pl, +fem, -p1, -p2, -plr, -prp], prp=nil, -rel, sbj=[-fem, -p1, -p2, -plr], -sub]
], ['Hagz', [cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[-expl], prp=nil, -rel, s
bj=[+fem, -p1, -p2, +plr], -sub];[cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[+e
xpl, +fem, -p1, -p2, -plr, -prp], prp=nil, -rel, sbj=[-fem, -p1, -p2, -plr], -sub]
], ['Hgz', [cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[-expl], prp=nil, -rel, s
bj=[+fem, -p1, -p2, +plr], -sub];[cnj=nil, der=[+it, +ps, -rc, -tr], -neg, obj=[+e
xpl, +fem, -p1, -p2, -plr, -prp], prp=nil, -rel, sbj=[-fem, -p1, -p2, -plr], -sub]
]]
>>> []

```

Figure 7: Screen dump of the program generating and analyzing orthographic form of *yit.ehagageza* 'they (fem.) help one another'

- Firdyiwek, Yitna and Daniel Yaqob. 1997. The system for Ethiopic representation in ascii. URL: [citeseer.ist.psu.edu/56365.html](http://citeseer.ist.psu.edu/56365.html).
- Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. Mouton, The Hague.
- Kaplan, Ronald M. and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–378.
- Karttunen, Lauri, 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.
- Kiraz, George A. 2000. Multitiered nonlinear morphology using multitape finite automata: a case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105.
- Koskenniemi, Kimmo. 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.
- Leslau, Wolf. 1941. *Documents Tigrigna: Grammaire et Textes*. Libraire C. Klincksieck, Paris.
- Mohri, Mehryar, 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.