Two-level Description of Turkish Morphology

Kemal Oflazer
Department of Computer Engineering and Information Science
Bilkent University
Bilkent, Ankara 06533 TURKEY
E–mail: ko@trbilun.bitnet, Fax: (90-4)266-4127

ABSTRACT: This paper describes a full two-level morphological description [5,9] of Turkish word structures. The description has been implemented using the PC-KIMMO environment [2] and is based on a root word lexicon of about 23,000 roots words. The phonetic rules of contemporary Turkish (spoken in Turkey) have been encoded using 22 two-level rules while the morphotactics of the agglutinative word structures have been encoded as finite-state machines for verbal, nominal paradigms and other categories. Almost all the special cases of, and exceptions to phonological and morphological rules have been taken into account. In this paper, we describe the rules and the finite state machines along with examples and a discussion of how various special cases were handled. We also describe some known limitations and problems with this description. We then briefly describe various natural language processing applications that use this description as the morphological analysis component.

1 Introduction

This paper describes a full two-level morphological description [5,9] of Turkish word structures. The description has been implemented using the PC-KIMMO environment [2] and is based on a root word lexicon of about 23,000 roots words. The phonetic rules of contemporary Turkish have been encoded using 22 two-level rules while the morphotactics of the agglutinative word structures have been encoded as finite-state machines for verbal, nominal paradigms and other categories cases. Almost all the special cases of and exceptions to phonological and morphological rules have been implemented.

Turkish is an agglutinative language with word structures formed by productive affixations of derivational and inflectional suffixes to root words. A popular (and rather exaggerated) example of a Turkish word formation is:

OSMANLILAŞTIRAMAYABİLECEKLERİMİZDENMİŞSİNİZCESİNE

which can be broken down into morphemes as follows:

OSMAN+LI+LAŞ+TIR+AMA+YABİL+ECEK+LER+İMİZ+DEN+MİŞ+SİNİZ+CESİNE

where the +'s indicate morpheme boundaries. This adverb can be translated into English as "as if you were of those whom we might consider not converting into an Ottoman." For the details of Turkish grammar and word formations rules one can refer to a number of books [11,18].[1]

Turkish has finite-state but nevertheless rather complex morphotactics. Morphemes added to a root word or a stem can convert the word from a nominal to a verbal structure or vice-versa, or can create adverbial constructs as above. The surface realizations of morphological constructions are constrained

[1] Familiarity with Turkish morphology, and the two-level paradigm will certainly make this exposition clearer.

and modified by a number of phonetic rules. Vowels in the affixed morpheme have to agree with the preceding vowel in certain aspects to achieve *vowel harmony*, although there are small number of exceptions. Under certain circumstances vowels in the roots and morphemes are deleted. Similarly, consonants in the roots words, or in the affixed morphemes undergo certain modifications, and may sometimes be deleted. We will see these when we discuss the two-level rules later. However, the assimilation of a large number of words into the language from various foreign languages – most notably Arabic and Persian – have resulted in word formations which behave as exceptions to many rules.

Turkish morphology has been investigated from a computational point of view by Köksal [8], by Hankamer [4], and by Solak and Oflazer [14,16,15].

2 Two–level Morphology

Two–level morphology is a general purpose paradigm for morphological description of word structures [2,5,9,17]. It has been used to analyze the morphology of number of languages [1,6,7,10,12].

A two-level description is based on a *lexical* and a *surface* representation of a word structure. The lexical level denotes the structure of the functional components of a word while the surface level denotes the standard orthographic realization of the word with the given lexical structure. The phonetic restrictions and modifications are represented using four different rule types [17]:

1. a:b $\Rightarrow$ LC ___ RC. This is the context restriction rule which states that a lexical a can be realized as a surface b only in the given left context (LC) and right context (RC), but not necessarily always.

2. a:b $\Leftarrow$ LC ___ RC. This is the surface coercion rule which states that a lexical a has to be realized as a surface b always in the given context but not necessarily only in that context.

3. a:b $\Leftrightarrow$ LC ___ RC. This is composite rule which states that a lexical a has to correspond to a surface b in the given context and the correspondence is valid only in that context.

4. a:b / $\Leftarrow$ LC ___ RC. This is the exclusion rule which states that a lexical a can never correspond to a surface b in the given context.

These rules are compiled into finite-state acceptors which check a lexical to surface correspondence in parallel. A lexical to surface correspondence that is not rejected by any of these machines is assumed to be a valid correspondence and hece accepted. Figure 1 shows the basic architecture of a two-level system.

The morphotactics which determine the proper sequencing of the morphemes are encoded as finite state machines using lexicons for roots words and suffixes, and alternations for capturing the sequencing of suffixes. In the following sections we assume that the reader is familiar with the concepts of two-level morphology (refer to Sproat [17] for a good overview.)

3 Two-level description of Turkish morphology

The Turkish language uses an alphabet of 29 letters in its current orthography using Latin characters. There are 8 vowels: a, e, ı, i, o, ö, u, ü, and 21 consonants: b c ç d f g ğ h j k l m n p r s ş t v y z. Tables 1 and 2 from [19] show the phonetic features that correspond to the sounds denoted by these letters.

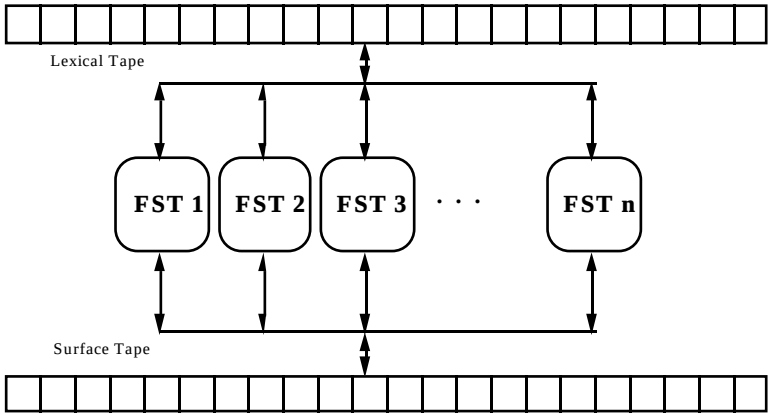

Figure 1: Parallel acceptors for two-level morphology

	+Round		–Round	
	+Front	–Front	+Front	–Front
+High	ü	u	i	ı
–High	ö	o	e	a

Table 1: Turkish Vowels

	Labial	Labio-dental	Dental	Palato-alveolar	Palatal	Velar	Glottal
Voiceless Stop	p		t	ç		k	
Voiced Stop	b		d	c		g	
Voiceless Fricative		f	s	ş			
Voiced Fricative		v	z	j			
Nasal	m		n				
Liquid			l,r				
Approximant					y		h

Table 2: Turkish Consonants

There are however sounds not covered by these. Certain long vowels are mainly used in words borrowed from foreign languages most notably Arabic and Persian. Such vowels are sometimes distinguished in older orthography by various means (such as with a ˆ on top of the vowel). In modern orthography such distinctions are not commonly used. There is also a certain phoneme known as "yumuşak g" (soft g – denoted as ğ in orthography) which creates bisyllabic two-vowel sequences. At the end of a syllable, this phoneme causes the lengthening of the preceding vowel[19].

Consonants k,g, and l have palatal and non-palatal allophones. In certain cases the palatalization process has impact on the vowel harmony. Our purpose in this two-level model is not necessarily to account for all phonetic phenomena from a purely phonetic viewpoint, but to have a description which can deal with all observed surface forms for use in practical applications.

For the purposes for two-level specification, we assume an alphabet that includes the letters above and certain additional symbols (that will be described shortly) only used in the two-level description, and not in the orthography.[2] First we define the following subsets:

1. Consonants: $\mathbf{C}$ ={ b, c, ç, d, f, g, ğ, h, j, k, l, m, n, p, r, s, ş, t, v, y, z }
2. Surface vowels subject to ellipsis under certain cases: $\mathbf{V_s}$ = { ı, i, o, ö, u, ü }
3. Lexical vowels: $\mathbf{V}$ ={ a, e, ı, i, o, ö, u, ü, H, A, â, ô, û }. These are the vowels used in the lexical level.
4. Back vowels: $\mathbf{V_b}$ = { a, ı, o, u }
5. Front vowels: $\mathbf{V_f}$ = { e, i, ö, ü }
6. Front unrounded vowels $\mathbf{V_{fu}}$ ={ e, i }
7. Front rounded vowels $\mathbf{V_{fr}}$ ={ö, ü}
8. Back unrounded vowels $\mathbf{V_{bu}}$ ={a, ı }
9. Back rounded vowels $\mathbf{V_{br}}$ ={o, u}
10. Lexical consonants used as the first letter in a suffix but which may disappear on the surface under certain conditions $\mathbf{X}$ = {s, y, n}
11. Lexical consonants used as the first letter in a suffix but which are always realized on the surface $\mathbf{C_r}$ = {S, l, c, D}

Before going any further, it is necessary to point out a number of points about Turkish orthography and our two-level representation:

- In the 3^{rd} item above, H used at the lexical level only, stands for the set of high vowels {ı,i,u ü}, unresolved for the other features, A again used at the lexical level, stands for nonhigh unrounded vowels {a, e}. â, û stand for long vowels which are present lexically but are not used in the orthography. These are realized as a, u on the surface. ô is the vowel in words of foreign origin such as *alkol* (alcohol) or *gol* (goal) where it is always followed by an l. This vowel behaves like an ö in the vowel harmony process.
- Proper nouns are separated from suffixes by an apostrophe ('). All vowel harmony rules and *some* of the consonant change rules are in effect in the orthography of proper nouns.

[2]The PC-KIMMO implementation which is now in public domain uses different characters for the symbols used below as the ASCII character set does not support certain characters in the Turkish alphabet.

- Some roots have vowels which are deleted when certain suffixes are affixed. Such vowels are prefixed with a $ in the lexicon representations.
- D is a lexical symbol denoting dental stop consonants which has default surface realization by the voiced d but may also be realized on the surface as the voiceless t under certain circumstances.
- S represents a lexical s which never gets deleted during affixation. For example in the suffix +sHz, the s never gets deleted, while in the suffix +sH, the s may sometimes be deleted. We represent the s in the former by S, whose surface realization is always s.
- K represents a root-final lexical k which never becomes a surface ğ during any affixation, e.g. in erk (power). Its surface realization is k.
- The rule compiler KGEN heavily used in this development treats a regular expression of the sort a∗b∗c∗ as (a∗b∗c∗)∗. This fact is used in the vowel harmony rules.

3.1 Two–level Rules

The following are the two-level rules for the phonetic component of the description:

1. A:a ⇒ V:V_b ':' ∗ C∗ @:0∗ +:0∗ ___
2. A:e ⇒ [V:V_f | â:a | û:u | ô:o] ':' ∗ @:0∗ +:0∗ ___
3. H:u ⇒ [V:V_{br} | V_{br} :0 +:0] ':'∗ C∗ +:0∗ @:0∗ ___
4. H:ü ⇒ [V:V_{fr} | V_{fr} :0 +:0 | û:u | ô:o] ':'∗ C ∗ +:0∗ @:0∗ ___
5. H:ı ⇒ [V:V_{bu} | V_{bu} :0 +:0] ':'∗ C∗ +:0∗ @:0∗ ___
6. H:i ⇒ [V:V_{fu} | V_{fu} :0 +:0 | â:a] ':'∗ C ∗ +:0∗ @:0 ∗ ___

   The first two rules force the agreement of an A vowel to a preceding vowel in the backness attribute. The last four rules force the agreement of an H vowel to a preceding one in backness and roundedness.

7. H:0 ⇒ V (':') +:0 ___

   An H vowel is deleted if the last phoneme of the stem it is being affixed to is a vowel. For example:

   Lexical: masa+Hm N(table)+1PS-POSS
   Surface: masa00m masam

8. H:0 / ⇐ V:0 +:0 ___ y o r
   However an H vowel is not deleted in the verbal suffix +Hyor denoting the progressive tense (present continuous) if the last phoneme of the stem corresponds to a vowel which gets deleted under this circumstance. For example:

   Lexical: kapa+Hyor V(close)+PR-CON+3PS
   Surface: kap00ıyor kapıyor

9. A:0 ⇔ ___ +:0 H:@ y o r
   The cases covered by this rule are the vowel ellipsis in certain verbal stems when +Hyor denoting the progressive tense is added to a stem ending in a vowel. An example is:

Lexical: ilaç+lA+Hyor N(drug)+NtoV(la)+PR-CON+3PS
Surface: ilaç0l00ıyor ilaçlıyor

10. V_s :0 ⇔ $:0 ___ C +:0 (X:0) [A:@ | H:@] | ___ +:0 H:@ y o r

A vowel in the lexical representation of certain roots will have to be deleted on the surface due to a vowel ellipsis phenomenon. Instead of adding these vowels to the alphabet we have indicated their context with a prefix $ which eventually gets deleted on the surface. An example is:

Lexical bur$un+Hm N(nose)+1PS-POSS
Surface bur00n0um burnum

This rule also deals with the vowel ellipsis for verbals roots ending in a vowel (cf. Rule 9 above). See the example for Rule 8 which also contains an example of the application of this rule.

11. X:0 ⇔ C (':') +:0 ___ (C) V

This rule deletes the beginning s, n, or a y of a suffix when it gets affixed to a stem ending in a consonant.

Lexical: ev+nHn N(house)+GEN
Surface: ev00in evin

Lexical: kalem+sH N(pencil)+3PS-POSS
Surface: kalem00i kalemi

Lexical: ağ$ız+yH N(mouth)+ACC
Surface: ağ00z00ı ağzı

12. D:t ⇔ [h | @:ç | ş | @:k | @:p | @:t | f | s] (':') +:0 (@:0) ___
This rule realizes a D as a t whenever it is preceded by one of the consonants in the option list across a morpheme boundary. D is otherwise realized as a d by default. Some examples are:

Lexical: kitab+DA N(book)+LOC
Surface: kitap0ta kitapta

Lexical: yulaf+DAn N(oat)+ABL
Surface: yulaf0tan yulaftan

Lexical: aç+DHk V(open)+ PERF
Surface: aç0tık açtık

13. {b, d}:{p, t} ⇒ ___ # | ___ +:0 (X:0) [C | c:ç]
This rule realizes voiced obstruents b, d as p, t respectively either when they end a word or when they are followed by a morpheme beginning with a consonant. Some examples are:

Lexical:	kitab+lAr	N(book)+PLU
Surface:	kitap0lar	kitaplar

Lexical:	kitab+cH	N(book)+NtoN(ci)
Surface:	kitap0çı	kitapçı

Lexical:	dolab+nHn	N(closet)+GEN
Surface:	dolab00ın	dolabın

Lexical:	tad+DHk	V(taste)+ PERF
Surface:	tat0tık	tattık

14. c:ç ⇔ [@:ç | ş | @:k | @:p | @:t | f | s] +:0 ___[H:@ | A:@]

15. ç:c ⇔ ___ +:0 X:0 V
    c is another voiced obstruent like those above except that it also appears in certain suffixes as the first letter where it gets modified to a ç.[3] Some examples are:

Lexical:	haraç+cH	N(tribute)+NtoN(ci)
Surface:	haraç0çı	haraççı

Lexical:	yaş+cA	N(age)+NtoAdv(ca)
Surface:	yaş0ça	yaşça

Lexical:	haraç+yA	N(tribute)+DAT
Surface:	harac00a	haraca

16. k:ğ ⇒ ___ +:0 (X:0) V
    The last k sound at the end of a word becomes a ğ when a morpheme starting with a vowel is affixed. Some examples are:

Lexical:	ayak+nHn	N(foot)+GEN
Surface:	ayağ00ın	ayağın

Lexical:	tarak+Hm	N(comb)+1PS-POSS
Surface:	tarağ0ım	tarağım

17. k:g ⇒ n___ +:0 (X:0) V
    However under certain circumstances the same k changes to a g as in:

[3] Normally words in Turkish do not end with c. Our original approach used this in the lexical representation. However we were not able to encode the reciprocal assimilation process that is observed when a suffix beginning with a c is affixed to a root ending in c in which both c's change to ç's by mutual influence.

Lexical: renk+yH N(color)+ACC
Surface: reng00i rengi

Lexical: ahenk+yA N(harmony)+DAT
Surface: aheng00e ahenge

18. g:ğ ⇒ ___ +:0 (X:0) V

    The last g at the end of words (of foreign origin) also becomes a ğ when certain suffixes are added. Examples are:

    Lexical: radyolog+yA N(radiologist)+DAT
    Surface: radyoloğ00a radyoloğa

19. g:ğ / ⇐ [n | r] ___

    However under the circumstances above if the final g is preceded by another consonant (only n and r seem to be such consonants) then the g does not become a ğ when suffixes are added. Examples are:

    Lexical: brifing+Hm N(briefing)+1PS-POSS
    Surface: brifing0im brifingim

    Lexical: aysberg+HnHz N(iceberg)+2PP-POSS
    Surface: aysberg0iniz aysberginiz

20. Y:y ⇔ ___ +:0 [X:0 | H:@]

21. Y:0 ⇔ ___# | ___ +:0 C_r

    These two rules deal with a large number of nominal roots ending with *su* (water) e.g., *akarsu* (river). *Su*, along with *ne* (what) does not obey the standard inflection rules. For example su+sH (water +3PS-POSS) is suyu and not susu and su+nHn (water+GEN) is suyun and not sunun.[4] We have added a lexical Y to such words so that further inflections can proceed properly but the Y is realized as 0 at the end of a word or if followed by a consonant which never drops in affixation. Here are some examples:

    Lexical: akarsuY+yHnHz N(river)+2PP-POSS
    Surface: akarsuy00unuz akarsuyunuz

    Lexical: akarsuY+lar N(river)+PLU
    Surface: akarsu00lar akarsular

22. ':0 ⇔ ___ [# | 1A:@]

    Finally we have a clean-up rule to remove the ' from a lexical proper noun under certain circumstances.

3.2 Root word lexicons

Our word list is based on the word list that we have compiled for our spelling checker [16]. This list is pretty comprehensive and is annotated with categorical and exceptions information. It however

[4] For ne the normal inflections are also valid.

contains a very small number of technical words and other domain specific jargon. This word list has been divided into a number of lexicons for:

1. *Nouns*,
2. *Adjectives* (This contains all nominal roots whose main usage is adjectival. They have the same morphotactics as the nouns. Current implementation has about 18,500 nominal (nouns + adjectives) roots.)
3. *Verbs*, (This lexicon contains about 2,450 verbal roots.)
4. *Compound nouns*,
5. *Proper nouns*, (This lexicon contains the proper nouns. All the roots in this lexicon end with a ' which gets deleted on the surface when necessary).
6. *Pronouns*,
7. *Adverbs*,
8. *Connectives*,
9. *Exclamations*,
10. *Postpositions*,
11. *Acronyms*, (Most of our intended applications that will use this description as a kernel, have to deal with real text which contains many acronyms which nevertheless have their own affixations schemes that we have to deal with.)
12. *Technical jargon*,
13. *Special cases*, (This contains nominal word structures which exhibit a number of special cases.)

3.3 Finite state machines for morphotactics

The morphotactics of a language determine the struture and the ordering of morphemes. In agglutinative languages the morphemes are affixed to a root word like "beads on a string" [17]. The two-level approach to computational morphology has concentrated mainly on the phonetic component, providing very simple finite-state mechanisms for describing the ordering of morphemes. While this is sufficient and convenient for a morphotactically simple language such as English, descriptions of the morphotactics of languages like Turkish may become unwieldly unless the descriptive power (not necessarily the computational power) are significantly enhanced. Nevertheless, our description implements the morphotactics of the Turkish language with the provided mechanisms and we later indicate the limitations of our description.

In PC-KIMMO, the morphotactics component is described by root and suffix lexicons which are linked to each other. In the figures describing our morphotactic component (such as Figure 3), the boxes indicate suffixation states, the arrows indicate the next states which can be reached when a suffix matching one of the labels is found. The circles indicate the final states for complete and valid word formations with the labels in parentheses near these states labeled End indicate the class of the word construction when the machine ends up in that final state. The 0 on the transitions indicate that the transition can be taken with *null* input. The states drawn in bold correspond to references to states in other figures. For example, the state labeled Possessive-3 indicates the state of a nominal construction

```
LEXICON NOUNS
    ab          POST-NOUN       "N(ab)"
    abad        POST-NOUN       "N(abat)"
    abadan      POST-NOUN       "N(abadan)"
    abadi       POST-NOUN       "N(abadi)"
    abajur      POST-NOUN       "N(abajur)"
    ...
    ...
    zürriyet    POST-NOUN       "N(zürriyet)"
    züyuf       POST-NOUN       "N(züyuf)"


LEXICON POST-NOUN
    +lHk        POST-NOUN       "+NtoN(lik)"
    +cH         POST-NOUN       "+NtoN(ci)"
    +cHk        POST-NOUN       "+NtoN(cik)"
    +lAş        POST-VERB       "+NtV(laş)"
    +lAr        PLURAL          "+PLU"
    +lArH       POSSESSIVE-3    "+3PP-POSS"
    +lArH       POSSESSIVE-3    "+PLU+3PP-POSS"
    0           PLURAL          ""
```

Figure 2: Example of PC-KIMMO lexicons for Turkish morphotactics

which has been affixed a third person possessive suffix. From that state one can go to a final state indicating a nominal in accusative case with suffix +nH, or to the states labeled Case-1 or Case-2 with the relevant case suffixes, or to another final state with the suffix +cA.

The morphotactic description for PC-KIMMO consists of lexicons of the sort shown in Figure 2. Each lexicon entry consists of a pattern, a pointer (or a set of pointers) to the next lexicon to be processed and the gloss to be output in case of a successful match.

3.3.1 Finite State Machine for Nominal Morphotactics

Figure 3 shows the finite state machine for the nominal paradigm. We would like to point out one caveat: These finite state machines have been designed with the assumption that the input (for recognition) is always legal and the recognizer will then chop it up into the proper sequence of morphemes. This assumption makes the finite state machines a bit simpler.

The morphotactics for the nominal paradigm is relatively simple. There are mainly two parts: The top part corresponds to nominal constructions with plural, possessive, case and relativization suffixes. It is technically possible to go around the loop through the state labeled Relative a number of times though in practice such constructions are rarely used. For example it is possible to have a word structure like:

MASA+LAR+IM+DA+**Kİ**+LER+İN+**Kİ**+NDE

which roughly means "at those (things) which belong to those (other things) at my tables."

The bottom part of the nominal morphotactics state diagram corresponds to the nominal verb and adverbial constructions like:

- *evdeydi* – (S/he/it) was at the house.
- *evdeyse* – If (s/he/it) is at the house.
- *evdeymiş* – (s/he/it) was as the house.(Narrative)
- *evdeyim* – I am at the house.
- *evdedirler* – They are (definitely) at the house.
- *evdeyken* – While (someone) is (was) at the house.
- *evdeymişcesine* – (behaving) as if he is at the house.

The nominal morphotactics are a bit different for compound nouns. The additional states required by these compound nouns are shown in Figure 4. Almost all compound nouns in Turkish have two components, and the second one is always in third person possessive form when the compound noun is used in the nominative case. For example *rengeyiği (reindeer)* (Lexical ren+geyik+sH), is used as both the nominative form and the third person possessive form. However further affixation does not proceed as in other nominals. For example the plural of *rengeyiği* is *rengeyikleri (reindeers)* where the plural suffix is now affixed to the normal nominal form of the second part of the compound and then the third person possessive is added. The same word can also be parsed as *his reindeers/their reindeer/their reindeers*. Similarly in *rengeyiğim* (my reindeer) or *rengeyiğin* (your reindeer/of the reindeer) the affixation is on to the nominal form of the second component and not on to the nominal form of the compound noun. In our lexicon we represent such compound nouns lexically without the third person suffix and the add this suffix on the surface when only the nominal form is used. Otherwise, we treat it as a normal nominal root.

Within the noun lexicon there are a number of roots which are already in plural form. For those cases the plural suffix and/or the possessive suffixes are skipped in the morphotactics. For example:

- *amcamlar* (the family/home of my uncle):[5] This is already in plural form and does not take any possessive suffix either. Hence the suffix lexicon that follows this is the CASE-1 lexicon.
- *bakliyat* (legumes), *baklagiller* (leguminous plants) are already in plural form.

For nouns already in plural form and ending in +lAr, the possessive suffix +sH, can be interpreted as both the third person singular possessive or third person plural possessive.

3.3.2 Finite State Machine for Verbal Morphotactics

Figures 5 and 6 show the finite state machine for the verbal paradigm. The verbal morphotactics is significantly more complicated than the nominal morphotactics. Turkish verbal structures can take a sequence of reflexive[6] or reciprocal, causative and passive suffixes which can then be followed by a compound verb, and then by first tense, a second tense and person suffixes. Verbal structures can also be made into nominal or adverbial structures with the addition of yet other suffixes. When a verbal root takes no reflexive or reciprocal suffix, the causative or the passive suffixes can take a variety of

[5] Note that this looks like it has a possessive suffix (+Hm) followed by the plural suffix (+lAr). However morphotactics puts the possessive after the plural, hence this can not parsed as such within the nominal paradigm.

[6] The reflexive suffix is not very productive.

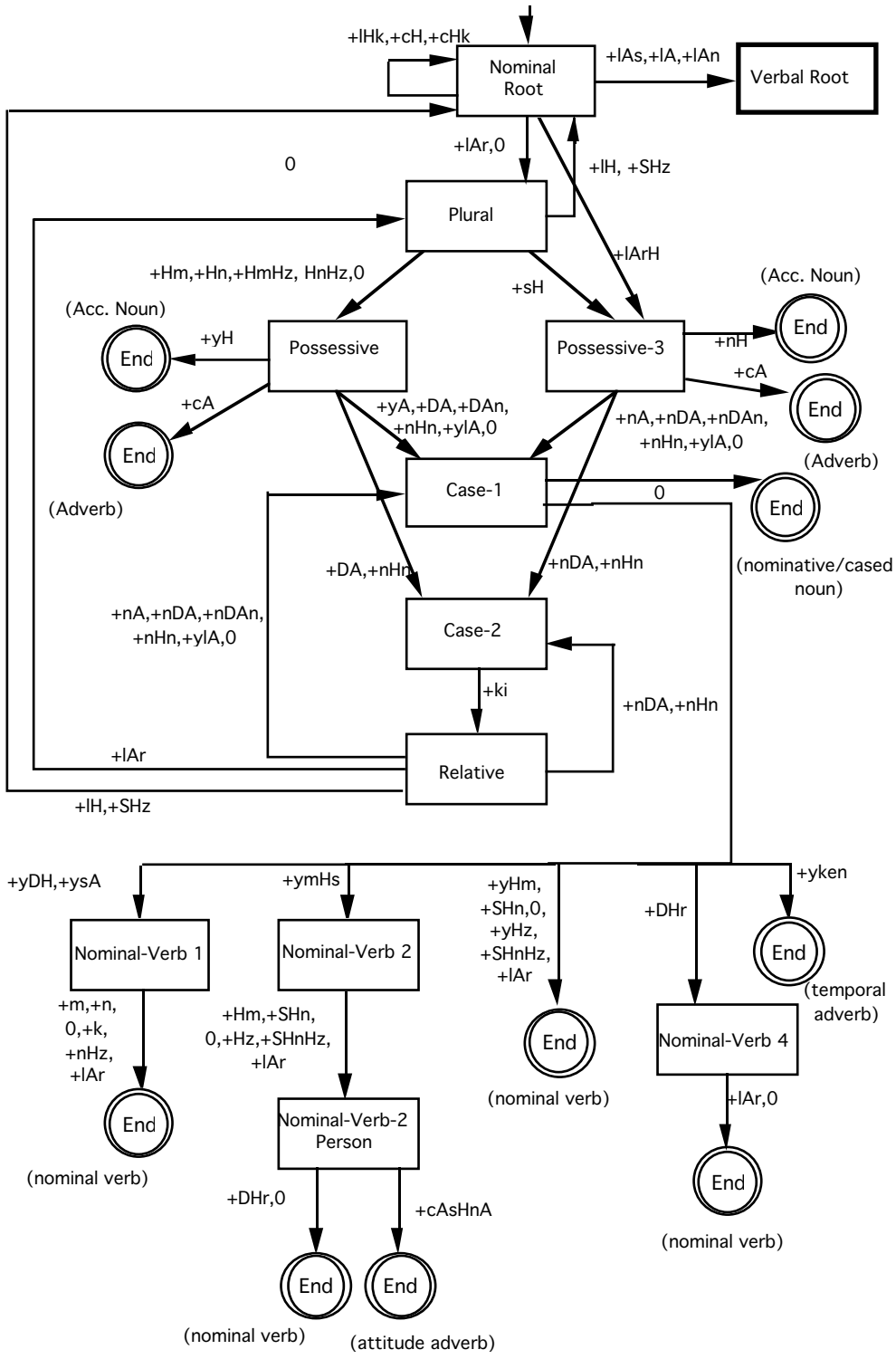

Figure 3: Finite State Machine for Nominal Morphotactics

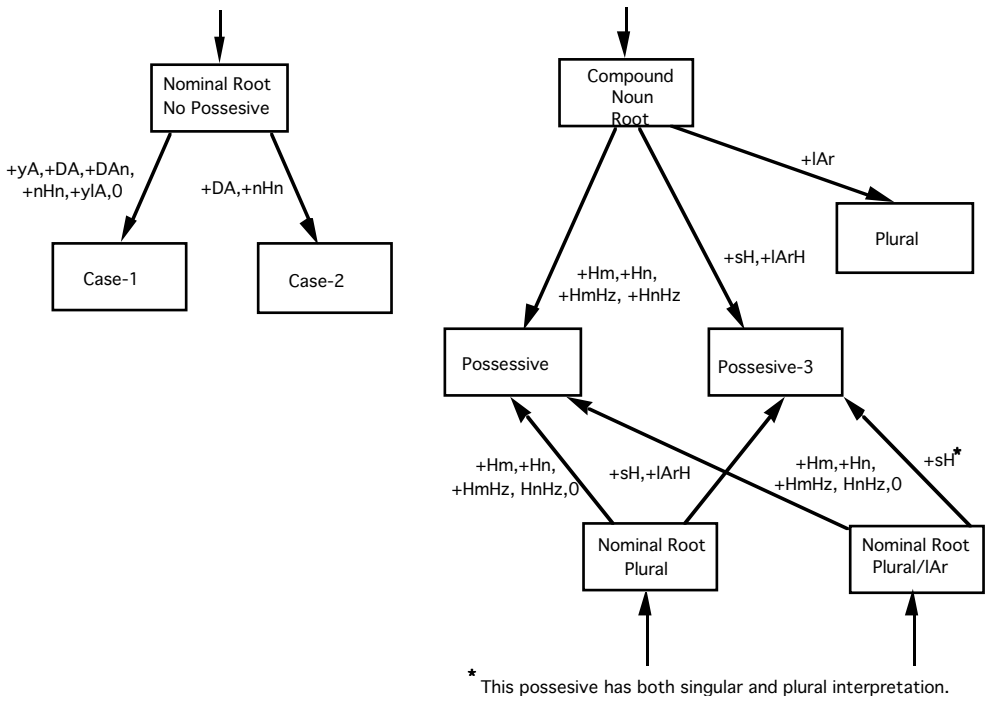

Figure 4: Finite State Machine for Compound Noun Morphotactics

forms depending on a number of criteria on the roots.[7] If, however, they take either of the reciprocal or the reciprocal suffixes (which are mutually exclusive), then the causative and passive formations are very simple as shown on Figure 5. After state labeled Passive Hn which corresponds to a verbal stem with all the reflexive/reciprocal, causative, and passive suffixes are accounted for, we can construct a negative form by +mA and +yAmA or directly go into positive verb construction. In any case, we can possibly add from a small number of auxiliary (or compound) verbs (the most common being +yAbil corresponding to *may* or *can* in English) to get a verbal stem to which we can now add tense and person suffixes, or suffixes which form nominal structures, infinitives and adverbs.

Turkish verbs can have at most two tense suffixes. The first one can be one of *narrative, future, aorist, present continuous, necessitative, optative, imperative, perfect* and *conditional* suffixes. These can take possibly different sets of person suffixes to form a verbal structure, or take a second tense morpheme indicating *perfect, conditional* or *narrative*. There are a number of nonstandard cases especially involving the third plural person and these are accounted for in the state diagrams.

An example will clarify the general idea behind verbal constructions. Consider the verb: *görülemiyormuşum* which can be translated into English as "(it is said that) I was not able to be seen." The morpheme structure is:

gör	+Hl	+yAmA	+Hyor	+ymHş	+yHm
gör	+ül	+0em0	+iyor	+0muş	+0um
see	+PASS	+NEG	+PRES-CONT	+NARR	+1PS

This verbal root *gör* will generate the structure above by going through the states labeled:

1. Verbal Root (root)
2. Passive Hl with +Hl

[7] See [14] for the details of these criteria.

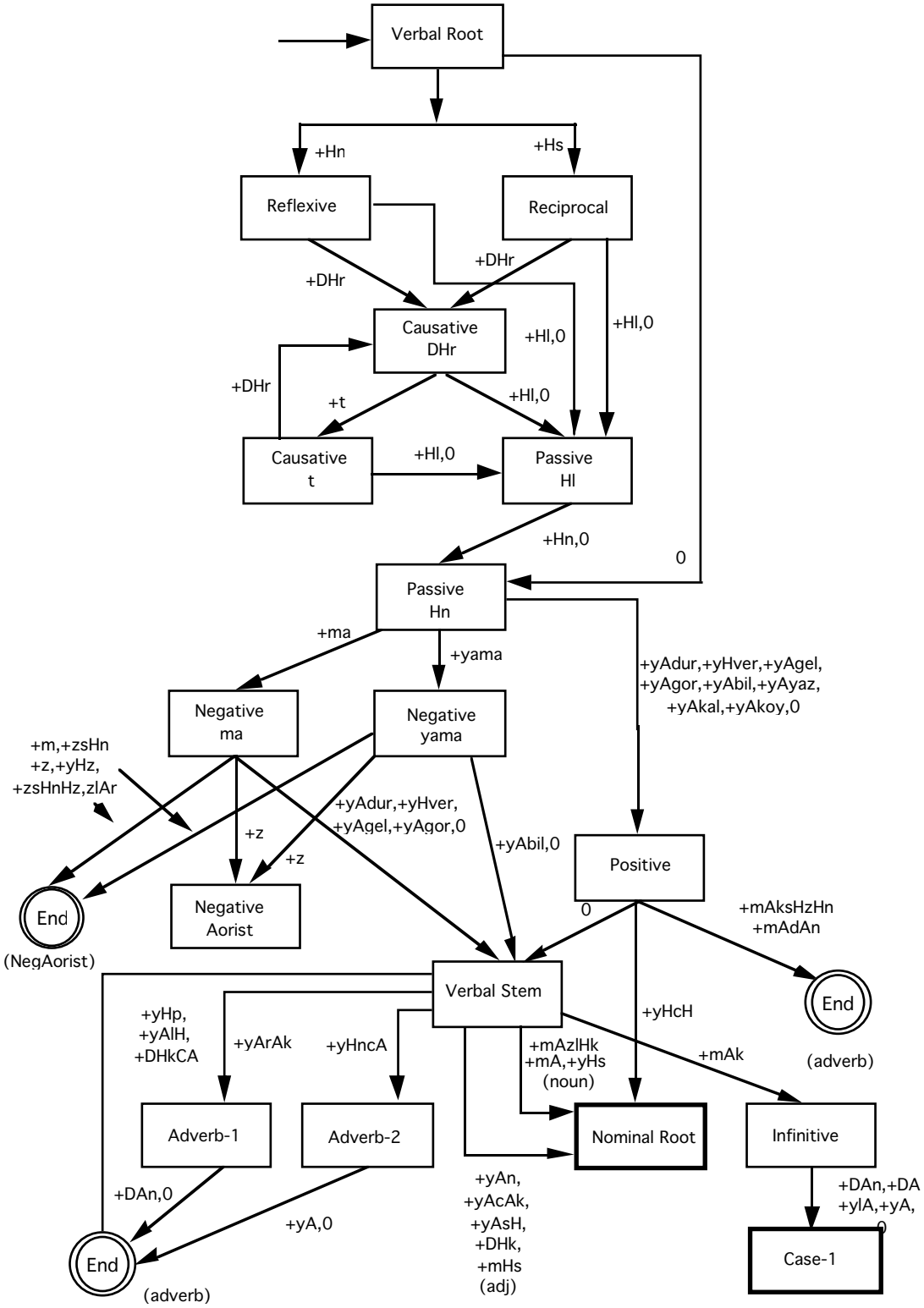

Figure 5: Finite State Machine for Verbal Morphotactics

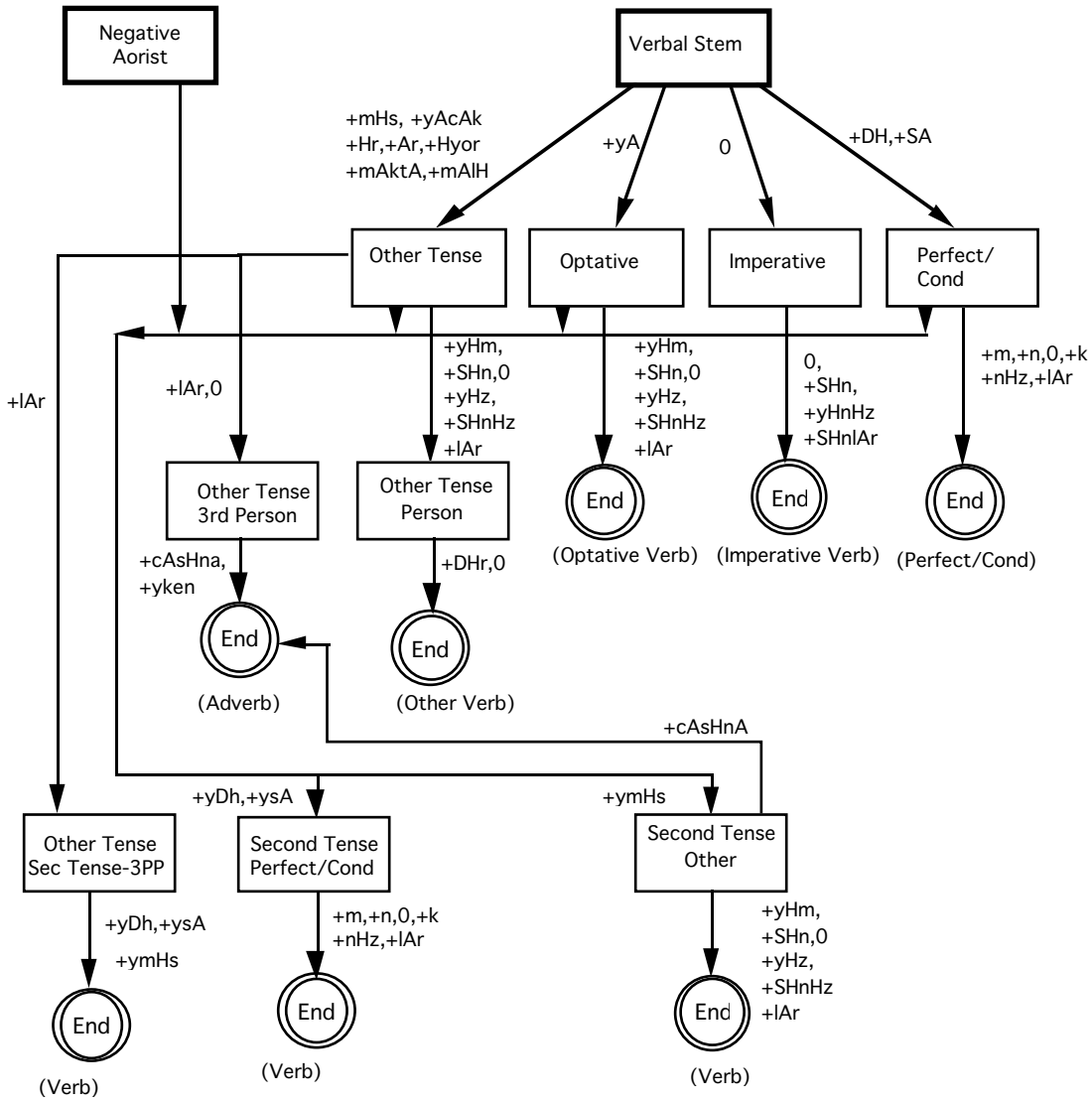

Figure 6: Finite State Machine for Verbal Morphotactics (cont.)

3. Passive Hn with 0
4. Negative yama with +yAmA
5. Verbal Stem with 0
6. Other Tense with +Hyor
7. Second Tense Other with +ymHş
8. End with +yHm

Readers familiar with details of verb formation in Turkish will note that our morphotactic model does not deal with the three groups of a total of 13 verbal roots whose aorist forms are exceptions to the rules.[8] Based on the caveat expressed above, our model can correctly identify the morphemes given correctly formed input but may not occasionally reject ill–formed aorist forms. We have opted to live with this as the alternative would have been to add another orthogonal categorization to the already large number of verb root groups and then replicate the remaining state machine three times to account for the aorist special cases. It is also possible to deal with special entries in the lexicon but this would not have prevented acceptance of ill–formed input. We expect to add this refinement to our model as we start using our two-level in applications like corpus tagging.

3.4 Special cases and exceptions

There are also a number of exceptions to most of the rules for word formation. We have opted to deal with these as special root lexicon entries. These exceptions are typically of the following sorts:

1. Adverbial formations indicating temporal and spatial relative positions can be directly formed with the +ki suffix to a nominal root denoting a time or place. For example:
   - *önceki*: (the one) before,
   - *yarınki*: (the one) tomorrow,
   - *bugünkü*: (the one) today,[9]
   - *karşıki*: (the one) across from here.
2. Third person singular forms of certain roots of arabic origin ending with a vowel do not have a intervening s. For example:
   - *sanayii* as opposed to *sanayisi*: industry+3PS-POSS
   - *mevkii* as opposed to *mevkisi*: place+3PS-POSS
3. There are adverbs formed from roots indicating numbers by the suffixes +şAr or +Ar, which are only applicable to such roots. For example:
   - *birer*: one each,
   - *ikişer*: two each,
   - *ellişer*: fifty each.

[8] This corresponds to the whether +Hr or +Ar is to be selected as the aorist suffix from the state labeled Verbal Stem. See [14] for details.

[9] Note that the relative suffix conforms to vowel harmony to his case

4. There are nominal roots ending with consonant which is repeated (possibly after going through a modification) before certain suffixes. For example:
   - *hakkı* (hak+yH): right+ACC (but *haklar* is hak+PLU)
   - *tıbba* (tıp+yA): medicine+DAT
5. For certain polysemus nominal roots, the vowel ellipsis phenomenon (dealt with $ above) does not occur when usage is for one of the meanings, but occurs for the other usage. For example:
   - metin+yH (text+ACC) is *metni* but
   - metin+yH (strong+ACC) is *metini*.
6. There are certain polysemus roots, where the last consonant is repeated before certain suffixes when used with one of the meanings and is not repeated with the other meaning.
   - şık+yH (chic+ACC) is *şıkı* but
   - şık+yH (item+ACC) is *şıkkı*.

In Turkish question suffixes starting with mH are written as a separate word, but the lexical H has to harmonize with the last vowel of the preceding word. For example:

geldiler mi?: Did they come?
görmüşsün you saw (it), görmüş müsün?: Did you see (it)? but not görmüşsün mü?

Since PC-KIMMO does not deal with sequences of words for recognition, this implementation does not deal with this harmonization nor with whether the prior morphotactics is valid if it is followed by the question part. It may be possible to deal with this by introducing a dummy space character which gets incorporated into the rules and gets inserted between the preceding word and the question part when it is detected.[10] In addition to this construct there are a large number of multi-word constructs in Turkish grammar where the grammatical role of the component words involved do not have much relevance to the grammatical role of the multi-word construct. We are currently working on an intermediate level as part of a project of part-of-speech tagging of Turkish text [13] which sits upon the morphological processor and handles multi-word and idiomatic constructs. This level will both provide a more accurate analysis of text and reduce the load on syntactic and semantic analyzers (e.g., [3]) that now have to do this handling at their level. Examples of cases that can be be handled at that stage are:

- double optative and 3SG verbal constructions like *koşa koşa* (running, as in *he came running*) which actually are used as adverbs,
- question suffix formations, discussed above,
- multiword verb formations with *etmek (to make), olmak (to be), yapmak (to do),* etc.,
- aorist verbal constructions like *yapar yapmaz* (as soon as (..) does (it)) which function as temporal adverbs,
- emphatic adjectival forms involving the question suffix, such as *güzel mi güzel* (very beautiful),
- word sequences with idiomatic usage such as *yanı sıra* (besides),
- various multiple word proper names.

[10] Current PC-KIMMO interface would not let you do this but it is possible to embed the recognition capability in a stand alone program that does the question form checking and combining input strings for a recheck.

4 Examples

In this section we give a number of examples from our PC-KIMMO implementation.[11] The examples follow the following format:

Input	
Morpheme Structure	Gloss
	English meaning
çalışmanın	
çalış+mA+Hn+nHn	[V(çalış)+VtoN(ma)+2PS-POSS+GEN]
	of your work(ing)
çalış+mA+nHn	[V(çalış)+VtoN(ma)+GEN]
	of the work(ing)
çocukları	
çocuk+lAr+sH	[N(çocuk)+PLU+3PS-POSS]
	his/her children
çocuk+lAr+yH	[N(çocuk)+PLU+ACC]
	children (accusative)
çocuk+lArH	[N(çocuk)+3PP-POSS]
	their child
çocuk+lArH	[N(çocuk)+PLU+3PP-POSS]
	their children
ödüllendirilmiş	
ödül+lAn+DHr+Hl+mHş	[N(ödül)+NtoV(lan)+CAUS+PASS+VtoAdj(mis)]
	rewarded (person)
ödül+lAn+DHr+Hl+mHş	[N(ödül)+NtoV(lan)+CAUS+PASS+NARR+3PS]
	(s/he has been) rewarded.
alınmış	
al+Hn+ymHş	[N(al)+2PS-POSS+NtoV()+NARR+3PS]
	(it) was your red (one)
al+nHn+ymHş	[N(al)+GEN+NtoV()+NARR+3PS]
	(it) belongs to the red (one)
al$ın+ymHş	[N(alın)+NtoV()+NARR+3PS]
	(it) was a forehead
al+Hn+mHş	[V(al)+PASS+VtoAdj(mis)]
	(a) taken (object)
al+Hn+mHş	[V(al)+PASS+NARR+3PS]
	it was taken
alın+mHş	[V(alın)+VtoAdj(mis)]
	(an) offended (person)
alın+mHş	[V(alın)+NARR+3PS]
	s/he was offended
boynu	
boy$un+sH	[N(boyun)+3PS-POSS]
	(his/her) neck
boy$un+yH	[N(boyun)+ACC]

[11] These outputs have been edited for proper character orthography, and then annotated.

neck (accusative)

5 Using the two-level description of Turkish in natural language processing applications

Our main reason for developing this description was to use it as a morphological analysis component in a number natural language processing applications in the context of a large scale project on natural language processing in Turkish. We have currently a number of efforts in progress for developing:

- Lexical–functional grammar specification for parsing Turkish sentences,
- Augmented transition networks for parsing Turkish sentences,
- Corpus taggers for Turkish text,

all of which use this description for doing morphological analysis.

6 Suggestions for a more powerful morphotactic descriptions for two-level morphology

We have noted a certain difficulty in the coding of the morphotactic component of the description. As the mechanisms for describing state transition are rather basic, the size of the finite state machines become unwieldy when portions of the machines have to be duplicated to deal with minor variations. For example, in our verbal morphotactics for the verbal structures, we have split the set of verbal roots into a number of groups depending on the kinds of causative and passive suffixes they can take and we have deliberately opted not to deal with the aorist suffix in this way. The distinctions among these groups are typically based on features like whether the root lexicon entry end with a vowel or not, or ends with a certain consonant or not, or whether the root is polysyllabic, etc. For example, one of the groups corresponds to those verbal roots (taking the causative suffix +DHr) which end with a consonant other than l, so that when they do not take the causative suffix, the passive is formed by Hl. If one could say *bind a variable* to the lexicon names and entry tokens matched as a transition is being taken, and refer to it in a later transition through an additional condition, structure of the finite state machines could be a simpler (though the speed could possibly be affected negatively!) For example, in the state diagrams for the verbal structures, the aorist suffix is dealt in the transition coming out the state Verbal Stem with suffixes +Hr or +Ar. The +Ar should be taken only by those mono-syllabic roots ending with a consonant (except for a small finite set of roots) and those compound verbal roots formed with +et.[12] The +Hr is taken by all multi-syllabic stems ending with a consonant except those formed by +et, and those mono-syllabic ones in that finite set above.[13] It would advantageous to refer to certain properties of the variables bound, while taking a transition. Suppose we have a lexicon entry structure such as

token {COND C } alternation gloss {BIND V}

where {COND C } and {BIND V} are optional *transition condition* and the *variable to bind to the matching token* respectively (cf. Figure 2. Then we could simply write the aorist transition as

[12] These are explicitly listed in the verbal lexicon.

[13] For stems ending in a vowel, the suffix is +r which is obtained when the vowel in either of the aorist suffixes drop.

```
LEXICON AORIST
+Ar     COND Check-AR-Aorist    POST-AORIST    "+AOR"  BIND Aorist-Suffix
+Hr     COND Check-HR-Aorist    POST-AORIST    "+AOR"  BIND Aorist-Suffix
```

For instance we could write the first condition Check-AR-Aorist as (expressed informally here):

> if **Causative–Suffix** is NULL and **Passive–Suffix** is NULL and **Negative-Suffix** is NULL, and
> [**Root** has a single vowel (hence mono-syllabic) and
> **Root** ends with a consonant and
> **Root Lexicon** is not VerbExceptions1] or
> **Root** ends with +et
> then return TRUE else return FALSE

with boldfaces indicating variables bound during recognition. Thus if this condition is checked when a suffix matching +Ar is found, and the check succeeds we would then take the transition. Doing this with the standard mechanisms of morphotactics provided is very cumbersome and ends up in large finite state models in which large subsets of states are duplicated to deal with minor variations.

Although we have not yet investigated the influence this would have on the overall formalism and performance of two-level morphology approach, we have reason to believe that for languages like Turkish such additional functionality would simplify constructions of very accurate morphotactic descriptions with simple machines.

7 Conclusions

In this paper we have presented a two-level description of Turkish morphology that we have implemented in the PC-KIMMO environment. We have used 22 two-level rules and the lexicon is based on a root word list of about 23,000 words. Being the first large scale two-level description of Turkish, this description can be used as inputs to a number of applications like morphosemantic analysis, corpus tagging, sentence level parsing etc. From a computational speed viewpoint, PC-KIMMO is rather slow as it is a general purpose environment. Our PC-KIMMO implementation can process an input word in about one half second on a SparcStation ELC. On the other hand, a highly optimized parser that we have developed for spelling checking Turkish[16] is about 1000 times faster,[14] but then ours is not a general purpose tool. This description of Turkish morphology is publicly available electronically via various archives and via the European Corpus Initiative collection.

8 Acknowledgements

I would like to thank Dr. Lauri Karttunen of XEROX-PARC for encouraging the implementation of this description, and to Dr. Evan Antworth of Summer Institute of Linguistics for providing the PC-KIMMO environment and numerous discussions.

References

[1] Yukiko Sasaki Alam. A two-level morphological analysis of Japanese. *Texas Linguistic Forum*, 22:229–252, 1983.

[14] Although due to the nature of the process, it will return after the first successful parse.

[2] Evan L. Antworth. *PC-KIMMO: A two-level processor for Morphological Analysis.* Summer Institute of Linguistics, Dallas, Texas, 1990.

[3] Zelal Güngördü and Kemal Oflazer. A Lexical-Functional Grammar specification for a subset of Turkish. In *Proceedings of Second Turkish Symposium on Artifical Intelligence and Neural Networks*, 1993.

[4] Jorge Hankamer. Finite state morphology and left to right phonology. In *Proceedings of the West Coast Conference on Formal Linguistics*, volume 5. Stanford University, 1986.

[5] Lauri Karttunen. KIMMO: a general morphological processor. *Texas Linguistic Forum*, 22:163 – 186, 1983.

[6] Lauri Karttunen and K. Wittenburg. A two-level morphological analysis of English. *Texas Linguistics Forum*, 22:217 – 228, 1983.

[7] Robert Khan. A two-level morphological analysis of Rumanian. *Texas Linguistic Forum*, 22:253–270, 1983.

[8] Aydın Köksal. *Automatic Morphological Analysis of Turkish.* PhD thesis, Hacettepe University, Ankara, Turkey, 1975.

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

[10] Kimmo Koskenniemi. An application of of the two-level model to Finnish. In Fred Karlsson, editor, *Computational morphosyntax: a report on reseach 1981 – 1984.* University of Helsinki Department of General Linguistics, 1985.

[11] G. L. Lewis. *Turkish Grammar.* Oxford University Press, 1991.

[12] S. Lun. A two-level morphological analysis of French. *Texas Linguistic Forum*, 22:271–278, 1983.

[13] Kemal Oflazer. A text tagger for Turkish. In *Proceedings of Second Turkish Symposium on Artifical Intelligence and Neural Networks.* Bosphorus University, 1993.

[14] Ayşın Solak. Design and implementation of a spelling checker for Turkish. Master's thesis, Bilkent University, Dept. of Computer Engineering and Information Science, Ankara, Turkey, 1991.

[15] Ayşın Solak and Kemal Oflazer. Parsing agglutinative word structures and its application to spelling checking for Turkish. In *Proceedings of the* 15^{th} *International Conference on Computational Linguistics*, volume 1, pages 39 – 45, Nantes, France, 1992. International Commitee on Computational Linguistics.

[16] Ayşın Solak and Kemal Oflazer. Design and implementation of a spelling checker for Turkish. *Linguistic and Literary Computing*, 1993.

[17] Richard Sproat. *Morphology and Computation.* MIT Press, 1992.

[18] Robert Underhill. *Turkish Grammar.* MIT Press, 1976.

[19] Harry van der Hulst and Jeroen van de Weijer. Topics in Turkish phonology. In Hendrik Boeschoten and Ludo Verhoeven, editors, *Turkish Linguistics Today.* E. J. Brill, 1991.