Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Speech and Language 20 (2006) 80–106
COMPUTER SPEECH AND LANGUAGE
www.elsevier.com/locate/csl

The architecture and the implementation of a finite state pronunciation lexicon for Turkish ☆

Kemal Oflazer [a,*], Sharon Inkelas [b]

[a] *Faculty of Engineering and Natural Sciences, Sabancı University, 34956 Istanbul, Turkey*
[b] *Department of Linguistics, University of California, Berkeley, CA 94720-2650, USA*

Received 10 November 2003; received in revised form 15 September 2004; accepted 24 January 2005
Available online 26 February 2005

Abstract

This paper describes the architecture and the implementation of a full-scale pronunciation lexicon for Turkish using finite state technology. The system produces at its output, a parallel representation of the pronunciation and the morphological analysis of the word form so that further disambiguation processes can be used to disambiguate pronunciation. The pronunciation representation is based on the SAMPA standard and also encodes the position of the primary stress. The computation of the position of the primary stress depends on an interplay of any exceptional stress in root words and stress properties of certain morphemes, and requires that a full morphological analysis be done. The system has been implemented using XRCE Finite State Toolkit.

1. Introduction

Pronunciation lexicons are computational devices that map the graphemic representation of words to a representation of their pronunciation. They are a valuable resource in annotating

☆ Short versions of this paper were presented as a poster at the Eurospeech 2003 Conference, and the Workshop on Finite State Methods in NLP, held during EACL 2003.

* Corresponding author.
E-mail addresses: oflazer@sabanciuniv.edu (K. Oflazer), inkelas@berkeley.edu (S. Inkelas).

0885-2308/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.csl.2005.01.002

speech data used in training automatic speech recognizers, and in generating accurate speech in text-to-speech systems. In this paper, we present the design and implementation of a full scale finite state pronunciation lexicon of Turkish, an agglutinating language with extremely productive inflectional and derivational morphological and hence an essentially infinite lexicon. The agglutinating nature of the language implies that any corpus based compilation of a word list for use in speech applications will be rather inadequate (Hakkani-Tür et al., 2002). Nouns typically have a few hundred forms, and Hankamer (1989) estimates that a few million forms can be generated from each verbal root. This necessitates that one employ a generative model that is able to recognize all possible words in the language and base the pronunciation lexicon on this to avoid a significant out-of-vocabulary word problem.

The pronunciation lexicon, implemented as a finite state transducer, takes as input a word form and produces all possible pronunciations of the word, paired with the corresponding morphological analyses. The pronunciations are encoded using the SAMPA encoding, and also include the marking of the position of the primary stress.[1] The dependence of the stress computation on the proper identification of morphemes and their morphotactical function requires that the pronunciation lexicon be built on top of a morphological analyzer. Although Turkish superficially seems to have an almost one-to-one mapping between graphemics and pronunciation, there are quite a number of subtle phenomena in loan words, and in morphophonology, such as vowel length alternations. Furthermore, suffixes can induce stress shift. Such phenomena are not distinguished in orthography but have to be handled while representing pronunciation. There are also a number of inter-word phenomena such as word-initial devoicing or word-final voicing that have to be handled at the sentence level, as orthography does not reflect those interactions either.

Finite state transducers are commonly used in building pronunciation lexicons for mapping between orthographic strings and phonological strings, either as an efficient encoding of direct mapping, or a mapping involving some kind of morphological processing (Jurafsky and Martin, 2000). Recently, Gibbon et al. (2000) have described a finite state transducer to act as a pronunciation resource for an inflected language like German. Their approach explicitly models morphologically out-of-vocabulary words by using a morphological parser that can segment and identify suffixes without an explicit root lexicon. Instead, it uses a small finite state transducer that can capture the contextual grapheme-to-phoneme mapping rules for German along with prediction of affix and root boundaries.

Some approaches also model dialectal pronunciation variations using finite state machinery. Hazen et al. (2002) describe a finite state transducer that uses a phonemic base form dictionary of English. Rewrite rules implemented as weighted transducers derive phonological variations. The main goal of the transducer is to generate variations such as contractions, reductions, part-of-speech pronunciation variants, variations that depend on stress and syllable positions, etc.

This paper makes extensive use of finite state methods and two-level morphology to build a full-scale pronunciation lexicon for a language with essentially an infinite vocabulary. Its contributions lie in its use of the morphological structure to compute the position of the lexical stress

[1] See http://www.phon.ucl.ac.uk/home/sampa/turkish.htm. We use the SAMPA notation to represent pronunciations in the text, where necessary.

and phonological changes that are not marked in graphemic representations of words, in addition to being the first such resource for Turkish.

2. Turkish

2.1. Aspects of Turkish morphology

Turkish is an Ural-Altaic language, having agglutinative word structures with productive inflectional and derivational processes. Turkish word forms consist of morphemes concatenated to a root morpheme or to other morphemes, much like "beads on a string" (Sproat, 1992). Except for a very few exceptional cases, the surface realizations of the morphemes are conditioned by various regular morphophonemic processes such as vowel harmony, consonant assimilation and elisions. The morphotactics of word forms can be quite complex when multiple derivations are involved. For instance, the derived modifier `sağlamlaştırdığımızdaki`[2] would be represented as:

```
sağlam+Adj^DB+Verb+Become^DB+Verb+Caus+Pos^DB+Noun+PastPart
  +A3sg+Pnon+Loc^DB+Adj+Rel
```

Starting from an adjectival root *sağlam*, this word form first derives a verbal stem *sağlamlaş*, meaning "to become strong". A second suffix, the causative surface morpheme +*tır* which we treat as a verbal derivation, forms yet another verbal stem meaning "to cause to become strong" or "to make strong (fortify)". The immediately following participle suffix +*dığ*, produces a nominal, which inflects in the normal pattern for nouns (here, for 1st person plural possessor which mark agreement with the subject of the verb, and locative case). The final suffix, +*ki*, is a relativizer, producing a word which functions as a modifier in a sentence, modifying a noun somewhere to the right.

The feature form representation above is generated by a two-level morphological analyzer for Turkish (Oflazer, 1994) that has been built using XRCE finite state tools (Karttunen and Beesley, 1992; Karttunen, 1993; Karttunen et al., 1996). This analyzer first uses a set of morphographemic rules to map from the surface representation to a lexical representation in which the word form is segmented into a series of lexical morphemes. For the word above, this segmented lexical morphographemic representation would be

```
sağlam+lAş+DHr+DHk+HmHz+DA+ki
```

In this representation, lexical morphemes except the lexical root utilize meta-symbols that stand for a set of graphemes which are selected on the surface by a series of morphographemic processes which are rooted in morphophonological processes some of which are discussed below. For instance, `A` stands for back and unrounded vowels *a* and *e*, in orthography, `H`

[2] Literally, "(the thing existing) at the time we caused (something) to become strong". Obviously this is not a word that one would use everyday. Turkish words (excluding noninflecting frequent words such as conjunctions, clitics, etc.) found in typical running text average about 10 letters in length. The average number of bound morphemes in such words is about 2.

stands for high vowels *ı*, *i*, *u* and ü, and D stands for *d* and *t*, representing alveolar consonants. Thus a lexical morpheme represented as +DHr actually represents 8 possible allomorphs, which appear as one of +*dır*, +*dir*, +*dur*, +*dür*, +*tır*, +*tir*, +*tur*, +*tür* depending on the local morphophonemic context.

2.2. Aspects of Turkish phonology

Overviews of Turkish phonology can be found in Clements and Sezer (1982); van der Hulst and van de Weijer (1991) and Kornfilt (1997). Turkish has an 8-vowel inventory which is symmetrical around the axes of backness, roundness, and height: /i, y, e, 2, a, o, 1, u/ which correspond to *i*, *ü*, *e*, *ö*, *a*, *o*, *ı*, and *u* in Turkish orthography. Suffix vowels typically harmonize in backness, and (if high) in roundness to the preceding stem vowel (compare, e.g., *ev* + *ler* /evler/ 'house-plural' to *at* + *lar* /atlar/ 'horse-plural'), there are several suffixes, e.g., the relativizer +*ki* seen above, whose vowels do not harmonize, as well as others, e.g., progressive suffix +*Hyor*, in which the first vowel harmonizes but the second does not.[3] Many roots are internally harmonic but many others are not; these include loan words (e.g., *kitap* /kitap/ 'book', from Arabic) as well as some native words (e.g., *anne* /anne/ 'mother'). Further, vowel harmony does not apply between the two components of compounds (e.g., *denizaltı* 'submarine').[4]

Turkish has 26 consonants: /p, t, tS, k, c, b, d, dZ, g, gj, f, s, S, v, w, z, Z, m, n, N, l, 5, r, j, h, G/. On the other hand, orthography uses only 21 letters for consonants: /g/ and its palatal counterpart /gj/ are written as *g*, while /k/ and and its palatal counterpart /c/ are written as *k*, /5/ and its palatal counterpart /l/ are written as *l*, /v, w/ are written as *v* and /n/ and its nasal counterpart /N/ are written as *n*. Palatalized segments (/gj, c, l/) contrast with their nonpalatalized counterparts only in the vicinity of back vowels (thus *sol* is pronounced /so5/ when used to mean 'left' vs. /sol/ when used to mean 'note in scale'). In the neighborhood of front vowels, palatality is predictable (*lig* /ligj/ 'league').[5] /G/, written as ğ, represents the velar fricative or glide corresponding to the historical voiced velar fricative that was lost in Standard Turkish. When it is syllable-final, some speakers pronounce it as a glide and others just lengthen the preceding vowel. We treat it as a consonant for the purposes of this work and explicitly represent it.

Root-final plosives (/b, d, g/) typically devoice syllable-finally (thus *kitab* + *a* /ci-ta-ba/ 'book-dative' but *kitap* /ci-tap/ 'book', *kitap* + *lar* /ci-tap-lar/ 'book-plural'). Suffix-initial plosives assimilate in voice to the preceding segment (thus *kitap* + it ta /ci-tap-ta/ 'book-locative' but *araba* + da /a-ra-ba-da/ 'car-locative').

Velar consonants (/g/ and /k/) reduce to /G/ at most root-suffix boundaries; thus *sokak* /so-kak/ 'street', *sokak* + *ta* /so-kak-ta/ 'street-locative' but *sokağ-a* /so-ka-Ga/ 'street-dative'. In certain dialects, a syllable-final /G/ may manifest itself as the lengthening of the preceding vowel.

[3] We use – to denote syllable boundaries and + to denote morpheme boundaries wherever appropriate.

[4] Turkish does not have productive compounding. It has a relatively small set of compound nouns comprising mostly noun-noun and adjective-noun combinations whose lexical semantics is mostly non-compositional/idiomatic.

[5] In conservative spellings of some words, contrastive velar or lateral palatality is indicated with a circumflex on the adjacent vowel, though this convention actually ambiguous and because circumflexes are also used in some words, equally sporadically, to indicate vowel length.

Turkish syllable structure allows open and closed syllables but no onset clusters.[6] Only a subset of consonant clusters are permitted in coda position. Vowel length is phonemic, and long vowels are in complementary distribution with coda consonants; short lexical vowels will lengthen when forced into an open syllable (thus /za-man/ 'time' but /za-ma:-na/ 'time-dative').

2.3. Stress in Turkish words

Turkish has lexical stress: each word has exactly one primary-stressed syllable.[7] Some roots are lexically stressed.[8] Certain morphemes are prestressing, that is, if not overridden, they will stress the preceding *syllable*.[9] A word composed of only unstressed morphemes exhibits the default stress pattern in which the last syllable is stressed. In a word with stressed root and/or prestressing suffixes, only one stress surfaces. In the Istanbul dialect described in most of the literature on Turkish stress, it is the lexical stress associated with the leftmost stress-related morpheme which prevails (see Inkelas (1999) and Inkelas and Orgun (2003) and citations therein for a comprehensive review. Kiparsky (1973) discusses the inflectional accent properties in Indo-European including the principle of "Leftmost Wins".) Other dialects (or perhaps idiolects) of Turkish, including that spoken by the first author, follow different rules for adjudicating cases of competing lexical stresses; in particular, it appears that for many speakers, the prestressing verbal negation suffix *+mA* will prevail even over a lexically stressed morpheme to its left.[10] Further research is surely needed on this pattern.

In place names and foreign names used in Turkish, a different default pattern is used, termed here the "Sezer" stress pattern, in tribute to its description in Sezer (1981). For such words, the antepenultimate syllable is stressed when it is heavy (meaning containing a long vowel or ending in a consonant) and the penultimate is light (meaning ending in a short vowel), e.g., /'an-ka-ra/; otherwise stress is penultimate (e.g., /is-'tan-bul/, /a-'da-na/). The Sezer pattern can be imposed on any word when used as a place name: *kandil* + *li* /kan-dil-'li/ 'oil lamp-with', but *Kandilli* /kan-'dil-li/ (same word used as place name). However, just as the word-final pattern is the default for non-place names, the Sezer stress pattern is a default for place names; it is blocked from applying to place names that happen to contain any lexically stressed morphemes (as may be the case in a place name which is a lexicalized derivation involving a normally prestressing morpheme).

The Turkish stress system is of considerable computational interest because of the potentially complex interplay between morphological structure and stress; *of significance is that the system is generally reducible to the single principle*, mentioned above, according to which the properties of the leftmost stress-perturbing morpheme dictate where stress falls in the word. This same principle also applies to compounds, such as *acemborusu* (/a-'dZem-bo-ru-su/ 'a flower', literally

[6] Except in a few loan words such as *angstrom* /angs-trom/.

[7] The existence of secondary stress is controversial.

[8] We call this *exceptional stress*.

[9] Several disyllabic suffixes systematically surface with stress on their first syllable (unless deleted by a stressed root or prestressing suffix to the left). For formal simplicity we treat these as prestressing as well; the second syllable bears the prestressing marker.

[10] A minor exception is that when the negation morpheme is followed by the aorist marker, the rightward suppression effect is neutralized!

persian pipe) in which it is usually the case that the first member (*acem* (/a-'dZem/ 'persian')) bears the stress it would have as an independent word, while the second member (*borusu* (/bo-ru-'su/ 'pipe')) is stressless in the compound, though it has stress on the final syllable as an independent word.

3. Computational considerations

The problem of grapheme-to-morpheme mapping for Turkish is considerably simpler than for languages such as English or French. Orthography more or less maps one-to-one to pronunciation. While all homophones are homographs, homographs can be morphologically interpreted in different ways may give rise to multiple pronunciations. So orthography can be ambiguous with respect to pronunciation. Such cases usually stem from the fact that a loan word (usually from Arabic, Persian or French) is a homograph of another Turkish word but has a different pronunciation. The once used accent marks to mark the distinctions are no longer consistently and unambiguously used, if at all, and one is left to rely on the context for inferring the correct pronunciation. The other major source of pronunciation ambiguity is the position of primary stress in the word. As we saw, stress is determined by an interplay of any Sezer/exceptional stress in root words and the stress marking properties of morphemes. Certain morphemes which are homographs and homophones except for stress-marking properties may appear in verbal morphotactics. For example, there are two *+mA* suffixes one which marks negative verb polarity and one which derives short infinitive. They are homophonous but their contribution to the position of the final stress are different. So, a word form like *okuma*, would either mean the imperative 'don't read' or the infinitive 'to read/reading'.[11] In the imperative reading, the stress will be on the *syllable* preceding the *+ma* suffix, while in the other reading the suffix is neutral and stress is on the last syllable. Thus, morphological analysis is necessary to determine the morpheme structure which, along with any stress markers in the root morpheme, then determines the position of the primary stress.

In an application context such as text-to-speech, the appropriate pronunciation of a word has to determined by a morphological disambiguation and/or word sense disambiguation process. For instance, morphological disambiguation of the different morphosyntactic interpretations of the word *okuma* would be necessary to select the appropriate pronunciation in a context, while a process akin to word sense disambiguation would be necessary to disambiguate the appropriate pronunciation of the word *sol* in Section 2.2.[12] Application level disambiguation requires the availability of the morphosyntactic features so that morphological interpretations, and hence the appropriate pronunciation can be selected using contextual information with statistical and/or symbolic means (Hakkani-Tür et al., 2002).

[11] In addition to a nominal reading *ok + um + a*, meaning 'to my arrow' which has the same pronunciation as the infinitive reading.

[12] Though, the two are not senses of a word in the lexicographic sense.

4. The architecture of the pronunciation lexicon

The word pronunciation lexicon transducer is the composition of a series of transducers that transform a surface form into all possible and ambiguous parallel representations of its pronunciation and morphological features.[13]

The overall internal structure of the transducer is shown in Fig. 1. All the boxes in this figure are finite state transducers, and in implementation, they are all composed at compile-time to give one (very large) transducer.

Before we discuss the internal details and the functions of each transducer in detail we present below a discussion of rationale for this sequence of transducers.

4.1. The rationale for the architecture

The structure of the sequence of transducer in Fig. 1 reflects the two major components of the computation involved:

(1) The extraction from the input word, of the information that will be needed to compute the pronunciation, and
(2) The actually computation of the representation of the pronunciation from the information extracted.

The bottom three transducers implement the extraction process and produce as the intermediate output, almost all the information necessary, while the remaining transducers compute the pronunciation from this intermediate output.[14] Below we discuss this separation of concerns in more detail and the look at each transducer and its functionality in detail.

As we have discussed before, the production of the correct pronunciation, including the correct placement of the primary stress, requires that root and the bound morphemes and their morphosyntactic functions be identified. Thus before we do anything, we need to

(1) analyze the input token into constituent free and bound morphemes and
(2) associate these with the morphosyntactic features they encode

The reason for this parallel generation of pairs of morpheme sequences and morphosyntactic feature sequences is that we do not want to lose the association of which morphological parse goes with which encoding of the pronunciation at the final output, since, for any further disambiguation of the contextually correct pronunciation will, most likely, make use of the morphosyntactic

[13] We assume that the reader is familiar with the basics of finite state devices and especially with finite state transducers which map between regular sets of strings. As we will not be needing any of the details of such devices in the rest of the paper except that they can be composed, we suggest that, for overviews of transducers and regular expressions used to define them, the reader refer to Karttunen (2001) and Karttunen et al. (1996), and to a recent book (Beesley and Karttunen, 2003).

[14] Please note that even though we will be describing the information flow from transducer to transducer as if it is happening at "run-time", this structure is, in the final analysis, a "compile time" structure of the computational process, as all transducers get compiled into a single transducer.

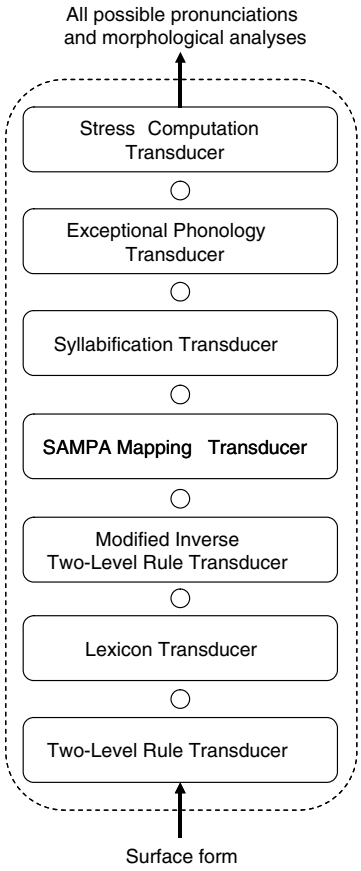

Fig. 1. The internal architecture of the pronunciation lexicon transducer.

features. Generating them separately and independently would not be of much use since then it would not be possible to associate a given pronunciation with an analysis, as this in general an *n*-to-*m* mapping as shown in Fig. 2. For the situation depicted in this figure, for instance, there would be four pronunciation-morphological parse representations: for $(\mathtt{P}_1,\mathtt{M}_1)$ $(\mathtt{P}_1,\mathtt{M}_2)$, $(\mathtt{P}_2,\mathtt{M}_2)$ and $(\mathtt{P}_2,\mathtt{M}_3)$. For example that the first pronunciation is associated with the first and the second morphological analyses.[15]

We should also emphasize that it is not only the morphemes that are relevant to the correct determination of the pronunciations – some of the morphosyntactic (and semantic features that we happen to encode in the morphological analyzer such as whether a root morpheme is a proper name or not) are also relevant in addition to the actual morphemes themselves.

[15] The actual pairing would actually interleave pronunciation and morphological analysis representations as we ill see shortly.

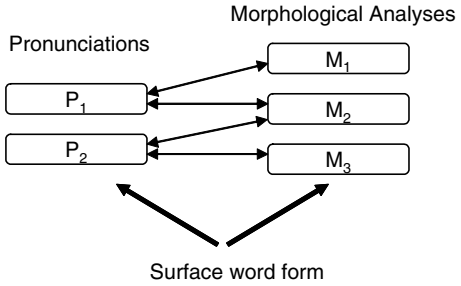

Fig. 2. Relationship between pronunciations and morphological analyses.

These considerations have prompted us to build the information extraction component of pronunciation lexicon (see above), on the scaffolding provided by the wide coverage morphological analyzer for Turkish (Oflazer, 1994). This analyzer is represented by the two (traditional) transducers in two-level morphological analysis setting – the *Two-level Rule Transducer* and the *Lexicon Transducer* – at the bottom of the sequence in Fig. 1.[16]

Since the morphological analyzer "consumes" the surface and the intermediate lexical representation produced by the two-level rule transducer, while it is mapping morphemes to morphosyntactic features, one needs to be able to "recover" the original surface form, from which the pronunciation representation will be produced. To achieve this, the lexical input to the lexicon transducer is replicated out the output, albeit in a format that interleaves morphosyntactic feature symbols with lexical morphemes.

This is the reason that the *Modified Inverse Two-level Rule Transducer* comes third in the sequence of transducers. Its function is to reconstruct the original surface form by applying the two-level morphographemic rules in the reverse direction – from the (replicated) lexical form to the surface representation, doing essentially what the first transducer does but in reverse, and over an interleaved representation, ignoring a bunch of morphographemically irrelevant feature symbols.

At this stage we have access to (almost) all the needed information for the determination of the pronunciation associated with a given morphological analysis. The tasks for the subsequent computation of the pronunciation are the following:

(1) the mapping of the graphemes to phoneme symbols in the SAMPA format,
(2) the syllabification of the phoneme sequence,
(3) any additional processing of the phoneme symbols in the root and (surface) morphemes (e.g., palatalization of the consonants in certain morphemes),
(4) the determination of the final position of the primary stress.

[16] Again this is really a very simplified model, as each of these, especially the *Lexicon Transducer* consists of hundreds of transducers that are combined in various ways.

Tasks (3) and (4) depend on the computation of the SAMPA representation of the pronunciation: The *Stress Computation Transducer* needs to know if the free root morpheme has any exceptional or Sezer stress, since in such cases all prestressing morphemes to the right may have to be suppressed.[17] This injection of the exceptional root stress marker is accomplished during mapping of the root from graphemic to SAMPA representation.[18] Further, for purely representational reasons, the final marker for the primary stress conventionally appears *before* the syllable of the stressed vowel, so the computation of the stress has to be done after the syllabification process.[19]

The *Exceptional Phonology* transducer also relies on the availability of the SAMPA representation, which is the representation it is going the operate on, but crucially it needs to know the syllable structure since the processing it needs to perform is conditional on the syllable structure. Thus syllabification and mapping to SAMPA have to precede the *Exceptional Phonology* transducer.

These considerations imply that both *Stress Computation Transducer* and the *Exceptional Phonology Transducer* follow the *SAMPA Mapping Transducer* and the *Syllabification Transducer*.

Syllabification in Turkish can in principle be done on the graphemic representation of words since standard Turkish orthography maps to pronunciation is a rather straightforward and almost 1-to-1 manner. Importantly, syllabification does not depend on morpheme boundaries, and there is no pressure in Turkish to align syllable and morpheme boundaries. The standard Turkish alphabet does not include any letter whose pronunciation maps to multiple phonemes. However one finds many (usually imported) words in common usage, written in their original orthography and using letters not in the alphabet whose letters map to two phonemes (e.g., *fax* /'faks/) and such phonemes may have to be split during syllabification (e.g., *faxı* /fak-'s1/). Thus the *Syllabification Transducer* needs to work on the output of the *SAMPA Mapping Transducer*.

On the other hand, there is no real precedence relation between the *Stress Computation Transducer* and the *Exceptional Phonology Transducer*s and the functions performed by these transducers can be performed in any order. We have opted to leave the computation of the position of the final stress to the last stage of pronunciation computation component of the overall computation.

We will now describe the function each of these transducers in detail as follows.

4.2. The two-level rule transducer

The *Two-level Rule Transducer* at the bottom implements the morphographemic mapping described by a set of parallel two-level rules (Koskenniemi, 1983). It is the intersection of the transducers for about 35 morphographemic rules that capture the morphographemics of Turkish (Oflazer, 1994). It is these rules that handle vowel harmony across morpheme boundaries and

[17] The markers for prestressing markers will be inserted during morphological analysis, since that is the only component of the system that "knows about" the morphotactics and the semantics of the morphemes.

[18] This could have been conceivably done in the lexicon of the morphological analyzer, but we opted to embed a minimal amount of pronunciation-related information in the morphological analyzer (the first two transducers in Fig. 1).

[19] Note that in general the syllable structure can not be statically determined as it depends on the suffixes added to root word and syllables can span over more than one morpheme and morphemes can be split into two or more syllables.

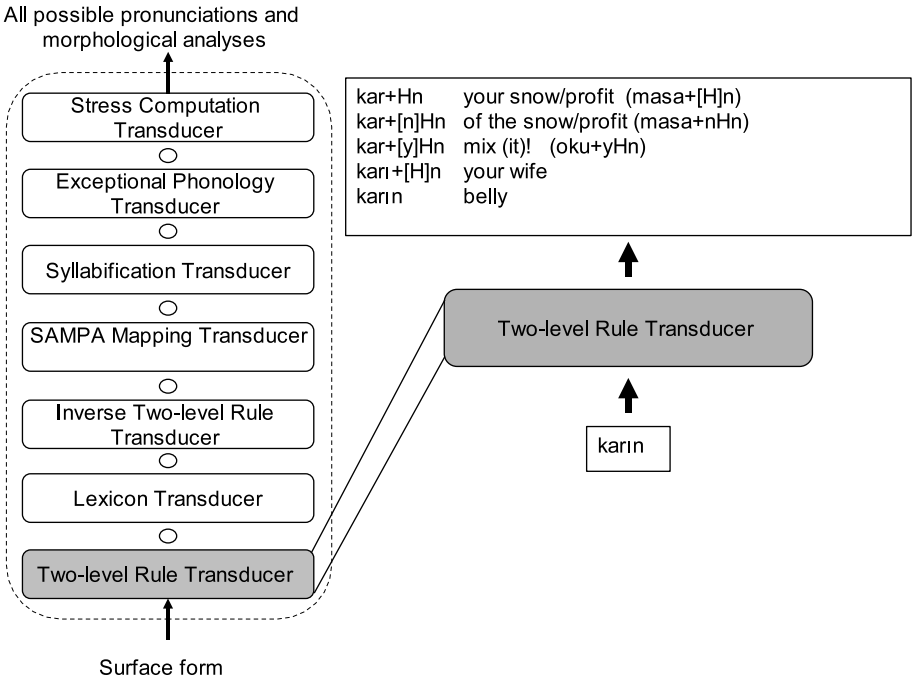

Fig. 3. Input and output of the two-level rule transducer.

consonant changes along morpheme boundaries and many other such processes. This transducer maps the surface representation of a word into possible lexical representations.[20] For example, the surface form `karın` would map to five lexical forms as depicted in Fig. 3, where we also provide English glosses to show the semantics of the morphosyntactic distinctions.[21]

It should be noted that since this transducer does not use any lexicon and is responsible only for morphographemic transformations, it is very overgenerating, but this is tamed when it is composed with the lexicon transducer described next, as this composition will filter out the outputs of the two-level transducer.

4.3. The lexicon transducer

The next transducer is the *Lexicon Transducer* comprising the root and the affix lexicons. In addition to the proper ordering of the suffix lexicons in the inflectional and derivational paradigms so that only proper combinations of roots and affixes are considered, this lexicon also comprises a couple hundred finite state constraints motivated by morphosyntactic and semantic concerns. These constraints impose fine grained tuning on word structures and significantly tame the overgeneration of the paradigm-based morpheme lexicon ordering, by filtering outputs violating these constraints.

[20] Though such lexical representations do not necessarily make distinctions, such as vowel length, not required by morphographemic processes.

[21] Remember that `H` represents a lexical archiphoneme that denotes a high vowel unresolved for frontness and roundess. Brackets denote segments of lexical morphemes that are deleted on the surface.

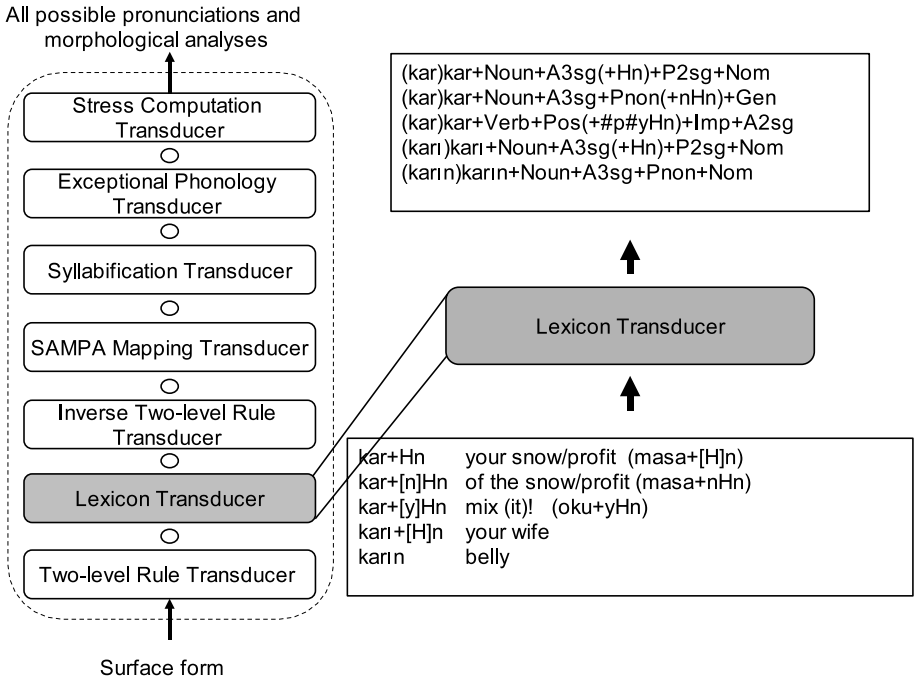

Fig. 4. Input and output of the lexicon transducer.

In the context of the pronunciation lexicon the *Lexicon Transducer* has two main functions:

(1) it produces all possible morphosyntactic feature representations of the free and bound morphemes and
(2) it replicates the lexical form at its output possibly augmented with any stress markers induced by *specific prestressing bound morphemes*.

Continuing the example earlier, the five lexical forms coming out of the *two-level rule transducer* would map to the outputs shown in Fig. 4.

A couple of remarks are in order for the outputs in Fig. 4:

(1) The morphological analysis process is essentially complete. We have an interleaved representation of lexical morphemes (between the parentheses (...)) and the morphosyntactic features they map to. The concatenation of the contents in pairs of parentheses, comprise the original lexical form with possible addition of certain stress markers, while the rest, when concatenated, gives the morphological analysis showing the morphosyntactic features encoded.[22]

[22] The morphological features used in the paper, other than the obvious POSs are: `Imp`: imperative mood, `+P2sg`: 2nd person possessive agreement, `+A1sg`: 1st person singular agreement, `+A2sg`: 2nd person singular agreement, `+A3sg`: 3rd person singular agreement, `+Pnon`: No possessive agreement, `+Nom`: Nominative case, `+Gen`: Genitive case, `+Pos`: Positive Polarity, `+Neg`: Negative Polarity, `+Become`: Become verb, `+Caus`: Causative verb, `+Prog1`: Progressive aspect, `+Past`: Past tense. `^DB` denotes a derivation boundary.

(2) The imperative morpheme +yHn (in the third output in Fig. 4) in morphotactics is a prestressing morpheme (hence the marker #p#), that is, it *may* eventually cause the primary stress to appear on the *syllable* before this marker.

From this point upwards in the structure, we carry the morphological features around, manipulating the lexical representation sandwiched between the parentheses, (...), to generate the representation of the corresponding pronunciation.

4.4. Inverse two-level rule transducer

The *Inverse Two-level Rule Transducer* is essentially (but not exactly) the inverse of the *Two-level Rule Transducer* discussed above. We have the same set of rules and a slightly different set of (inverse) feasible pairs (different for a variety of technical reasons.) The only difference in the rules is that the context regular expressions of the two-level rules are modified to ignore the delimiter symbols (and), the stress markers, and the feature symbols outside the parentheses that were added during the morphological analysis. The function of this transducer is to map the lexical form back to the surface morphemes, the concatenation of which will give the original surface form (plus any stress markers). So, for the five outputs of the lexicon transducer in Fig. 4, we will get the outputs depicted in Fig. 5.

At the output of this transducer, we have recovered the original input surface form with the addition of any stress markers stemming from some of the morphemes. With each output, we also

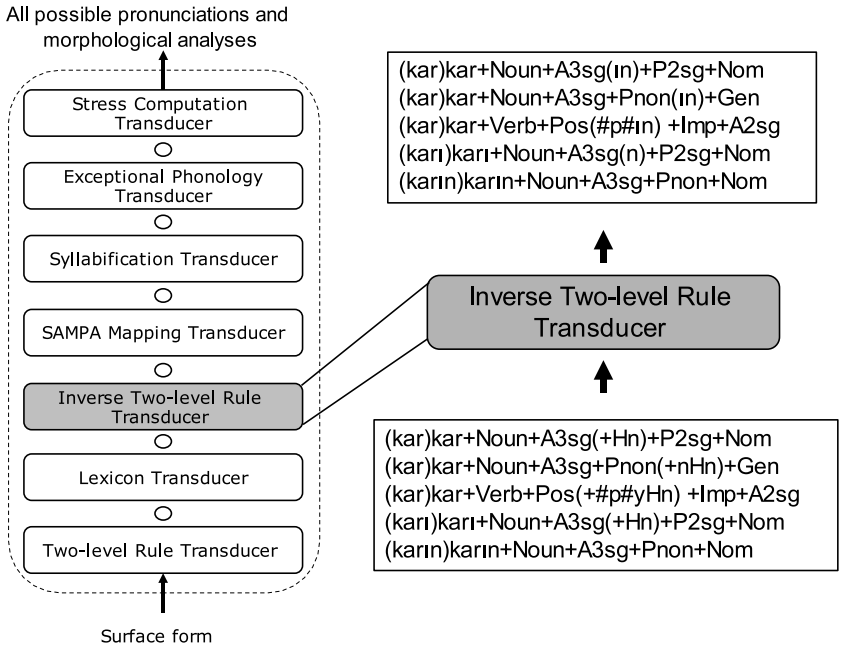

Fig. 5. Input and output of the inverse two-level rule transducer.

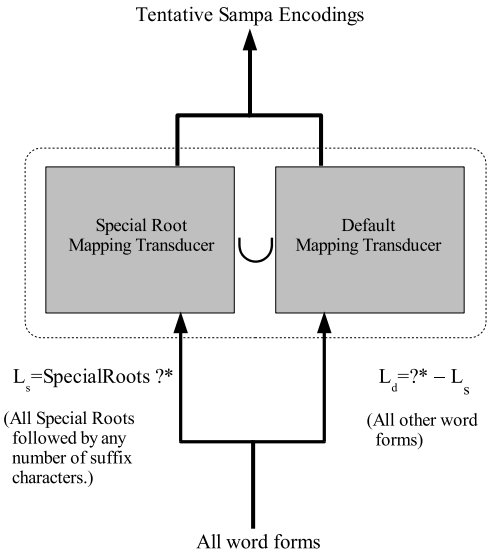

Fig. 6. The internal structure of the *SAMPA Mapping Transducer*.

have the morphosyntactic features encoded in the lexical morphemes which have now been converted to surface morphemes displayed between the (`and`) delimiter symbols.

4.5. The SAMPA mapping transducer

Now that we have the original surface form of the word, the surface morphemes can be mapped to the representation of their pronunciations by the *SAMPA Mapping Transducer*. As we mentioned earlier, the grapheme-to-phoneme mapping for Turkish is almost one-to-one and rather straightforward. But there are many subtle cases which require special handling. Such special cases require that transducer be considered as the union of two transducers as shown in Fig. 6.

(1) The *Special Root Mapping Transducer* maps to SAMPA words whose roots require special handling.
(2) The *Default Mapping Transducer* maps to SAMPA words whose roots do not require any special handling.

The *Special Root Mapping Transducer* handles words whose pronunciations can not be handled using a default mapping for a variety of reasons:

- The word may be a loan word from another language and have long vowels (e.g., *abadi* /a:-ba:-di:/ 'glazed paper').[23] As these long vowels are not necessarily marked in orthography, such words need to be handled specially.

[23] Turkish has long vowels in only loan roots, and for certain speakers syllable-final /G/ lengthens the preceding vowel.

- The root word pronunciation may involve one of the palatal consonants /c, gj, l/ which use the same letters *k*, *g*, and *l* in orthography as their non-palatal counterparts /k, g, 5/. Such root words have to be explicitly handled as their palatality can not be inferred from the nearby vowels. For example *hal* ('state'/'fruit-market') is pronounced /hal/ (in contrast to *bal* /ba5/ 'honey')
- In many cases some of which involve roots words of the two cases above, a homograph will have two different pronunciations usually based on the part-of-speech of the root word. In such cases, one root word will be mapped to more than one pronunciation representations. For example *ama* is /'a-ma/ when it is the conjunction 'but', but /a:-'ma:/ when it is the noun or the adjective 'blind'.
- Many proper nouns and some roots words will have Sezer stress or exceptional stress. For these root words, this mapping will produce a SAMPA representation with the right marker indicating a position of the root word stress. For homographs, for which one of the readings has stress on the root, multiple root mappings need to be produced. For instance *Aydın* as a root is /'aj-d1n/ when it is proper noun and refers to a city in Turkey, but otherwise is neutral with respect to stress as /aj-'d1n/, when it is a proper noun referring to a human, or when it is an adjective.
- The root word may be an unpronouncible acronym (e.g., PTT) whose pronunciation would have be a letter-by-letter pronunciation, /pe-te-'te/ in this case.

To deal with such cases, this transducer employs a separate lexicon (a much smaller version of the root lexica of the morphological analyzer) and maps from the root words and a subset of their morphosyntactic features into a representation of their appropriate pronunciation in the SAMPA standard. This lexicon only contains the mapping of the root words which either have exceptional or Sezer stress and/or a non-default mapping from graphemes to phonemes Any suffixes in the words involving the root words covered by this transducer are handled using a default mapping.

Any word forms not covered by the *Special Root Mapping Transducer* are handled by the *Default Mapping Transducer*. This transducer maps the ambiguous graphemes *g*, *k* and *l* to their non-palatal SAMPA counterparts /g, k, 5/, the grapheme *v* to /v/ and the grapheme *n* to /n/. The remaining ones are not ambiguous and map directly to the appropriate SAMPA symbol.

Although not currently implemented in this version of the pronunciation lexicon, this transducer would be the right place to handle some other systematic pronunciation variations after the mapping, such as the insertion of a short vowel (in pronunciation) to break up onset clusters, e.g., *spor* /sipor/, in some speakers.

Fig. 7 shows the outputs of the SAMPA mapping transducer in response to the inputs from the previous stage. In this example, this transducer produces two pronunciations for morphosyntactic interpretations with root *kar*; the root pronunciation is /car/ when the word is used with meaning 'profit'.

The concatenation of the segments within the parentheses (...) comprise the tentative SAMPA encoding of the word's encoding. A number of minor phenomena have to be fixed in the SAMPA representation. These are handled later by the *Exceptional Phonology Transducer*.

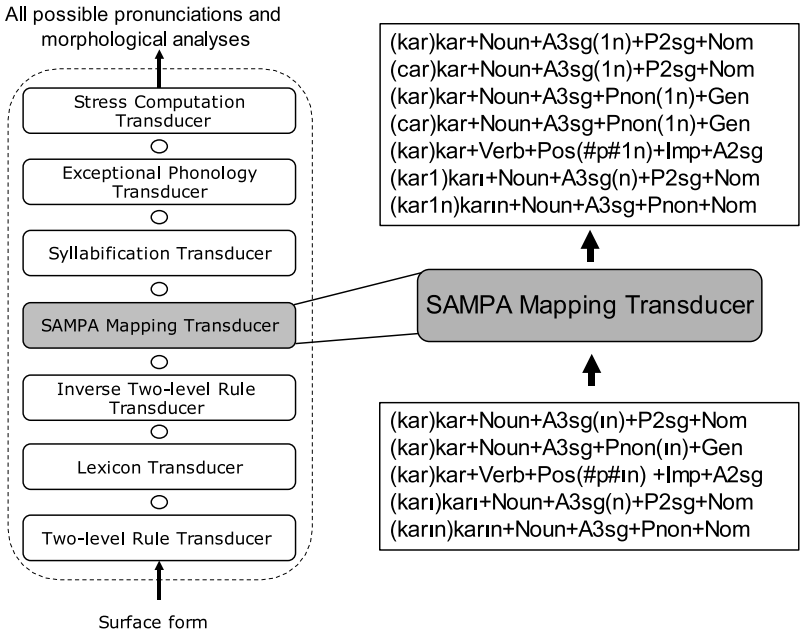

Fig. 7. Input and output of the *SAMPA Mapping Transducer*.

4.6. The syllabification transducer

The *Syllabification Transducer* performs the syllabification of the complete SAMPA encoding. The syllabification of Turkish words is highly regular, and essentially involves inserting syllable boundaries (marked by –) to break up the consonant sequences between two vowels.

VCV sequences syllabify as *V-CV*, while *VCCV* and *VCCCV* sequences syllabify as *VC-CV* and *VCC-CV*, respectively. The generalization followed by Turkish, as in virtually all other languages, is that a prevocalic consonant is always syllabified as a syllable onset. This pattern is captured in Optimality Theory by the universal ONSET constraint (Prince and Smolensky, 1993). The preference for splitting 2- and 3-consonant clusters in the way that Turkish does is also not uncommon cross-linguistically; it is modeled in Optimality Theory by ranking *COMPLEX-ONSET (the ban against syllable-initial consonant clusters) above the general ban against codas (NOCODA) and the more specific ban on complex codas (*COMPLEX-CODA).

The syllabification of longer clusters is more complicated, and can depend on the manner and place of articulation of the consonants involved. 4-consonant clusters split after the 3rd (thus *VCCC-CV*), as in /gangs-ter/, except when the resulting 3-consonant coda cluster would be impossible, as in the case of /ekspres/. /ksp/ is not a possible syllable- or word-final consonant sequence in Turkish, and therefore the syllable break in this word occurs after the second consonant (thus *VCC-CCV*), or /eks-pres/. This situation would be modeled in Optimality Theory by ranking the constraint(s) on what can be a possible complex coda (CODACOND) above the ban on complex onsets: CODACOND $\gg$ *COMPLEX-ONSET. Words with 5 intervocalic consonants pattern similarly. As many consonants as possible are syllabified into the coda; the remainder form a complex onset

to the following vowel. In the case of /angstrom/, the syllable break occurs after the 3rd consonant, since /ngs/ (but not /ngst/) is a possible syllable coda in Turkish; thus /angs-trom/ (*VCCC-CCV*). In the case of /golf-strim/, however, /lf/ is a possible syllable coda but /lfs/ is not; therefore the syllable break occurs after the 2nd consonant, i.e., /golf-strim/ (*VCC-CCCV*).

We summarize below, the possible cases for syllabifications showing how the consonants between two vowels would be split into the coda of the syllable involving the first vowel and the onset of the syllable involving the second vowel:

- *VCCCCCV* may be split as either
  - *VCCC–CCV*, e.g., /a**ngs**–**tr**om/, or as
  - *VCC–CCCV*, e.g., /go**lf**–**str**im/
- *VCCCCV* may be split as either
  - *VCCC-CV*, e.g., /ga**ngs**–**t**er/, or as
  - *VCC-CCV*, e.g., /e**ks**–**pr**es/,

  depending on the nature of the last consonant clusters involved.[24]
- *VCCCV* is split as *VCC-CV*, e.g., /ce**nt**–**t**e/.
- *VCCV* is split as *VC–CV*, e.g., /e**v**–**d**e/.
- *VCV* is split as *V–CV*, e.g., /e-**v**e/.
- *VV* is split as *V–V*, e.g. /ma–a–i-le/

The computational implementation of syllabification is achieved through a cascade of replace transducers ordered in such a way so that cases involving special consonant clusters are handled first as depicted in Fig. 8.

These replace transducers insert a syllable boundary symbol – to break up a consonant cluster provided the context patterns match. Note that when one of these transducers (starting at the bottom) insert boundary symbol within a segment between two vowels, none of the remaining transducers will have matching contexts (which require that no boundary symbol is in the context) so the modified input will just pass through. Again, this sequence of transducers are composed at compile time into a single transducer.[25] In the running example that we have been using to demonstrate the functionality of our transducers, syllabification splits the SAMPA representations after the first vowel as shown in Fig. 9.

4.7. The exceptional phonology transducer

The *Exceptional Phonology Transducer* handles a set of phenomena for certain exceptional roots and morphemes. The most important of these is the lengthening of the last vowel in selected roots (usually of Arabic origin), when those vowels turn out be in an open syllable when suffixes are added. It turns out that *kar* (/car/) is one of those root words, and in the examples above, the

[24] These first two cases only happen with words of foreign origin.

[25] The details of contexts of the replace rules from which these transducers are compiled, are quite complicated since they have to ignore all the other symbols making up the morphological features, stress markers and the delimiters. See Fig. 9.

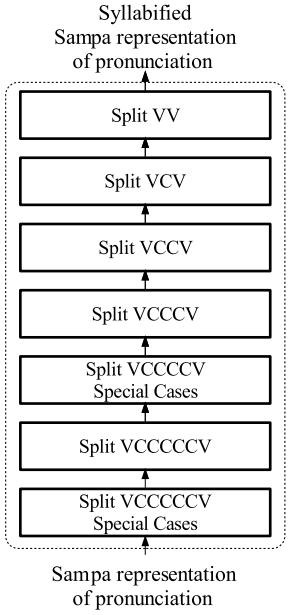

Fig. 8. Internal Structure of the *Syllabification Transducer*.

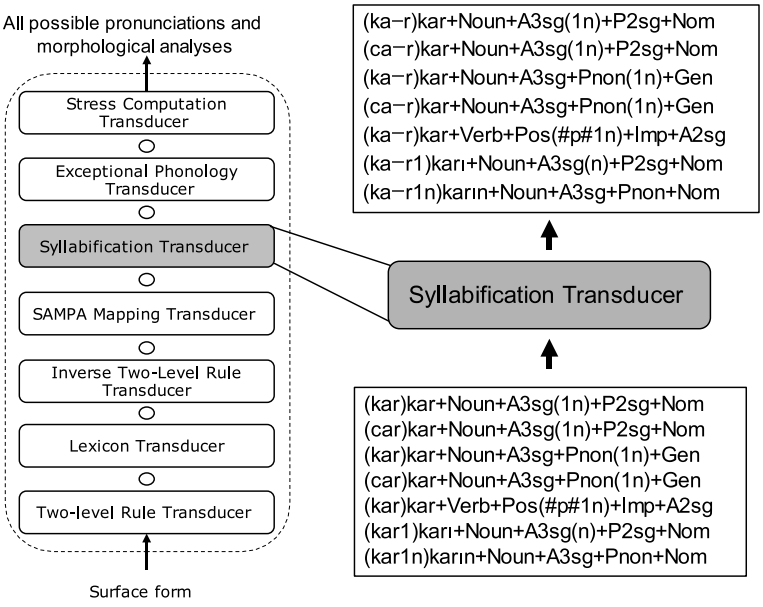

Fig. 9. Input and output of the syllabification transducer.

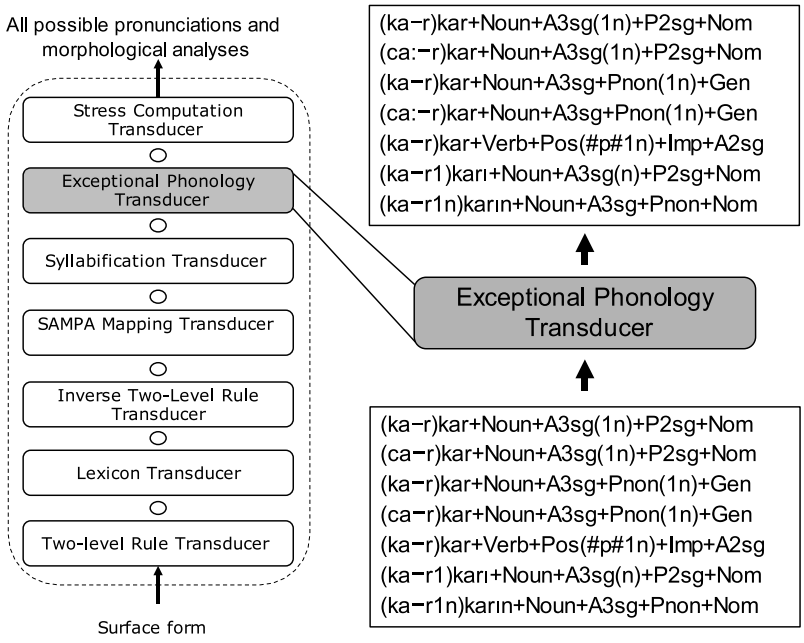

Fig. 10. Input and output of the exceptional phonology transducer.

root is followed by the surface morpheme /1n/ which forces the last (and the only in this case) vowel to an open syllable. So /a/ is lengthened, as shown in Fig. 10.[26]

A second common phenomenon handled by this transducer is the palatalization of /5/ to /l/ and /k/ to /c/ in certain suffixes when they are in the vicinity of the front vowels /e, i, y/ (see Section 2.2).[27,28] Since this is not a morphographemic process, the underlying morphological analyzer is oblivious to this in earlier stages.

The output of this transducer has the correct SAMPA encoding except that the final position of the primary stress is not yet determined.

5. The stress computation transducer

The *Stress Computation Transducer* computes the position of the primary stress by examining any Sezer/exceptional stress markings in the roots and prestressing markers in the morphemes. The position of the final stress is determined in a series of steps:

(1) All prestressing markers (#p#) interspersed in the SAMPA representation that have a prestressing marker or a Sezer/exceptional stress marker somewhere to their left are removed; the exception, for the speakers and/or dialects referred to in Section 2.3, is the

[26] The underlying morphological analyzer did not make vowel length distinctions as they were not needed while handling the morphographemic process.

[27] Remember that the SAMPA mapping of the suffixes is done with the default mapping.

[28] /2/ (ö in orthography) does not appear in any suffixes.

verbal negation marker, whose prestressing status is preserved and causes the deletion of stress markers to its left. For example, the representation at this point of the surface word *taşlaştıramıyorduk*[29] would be

(taS)taş + Noun + A3sg + Pnon + Nom
(-laS)^DB + Verb + Become
(-tl-r)^DB + Verb + Caus
(a-#p#m)^DB + Verb + Able + Neg
(l-j#p#or) + Progl
(-#p#du) + Past
(k) + Alpl

where three morphemes have prestressing markers. This step deletes all such prestressing markers except the first one (in the surface morpheme (a-#p#m)).

(2) Any Sezer or exceptional root stress marker which survives the deletion process described above causes the vowel associated with it to surface with stress.[30] So for example the word *pencerede* ('on the window') whose root *pencere* has exceptional stress but is otherwise composed of stress-neutral morphemes, will have surface root stress:
(pen-dZ″e-re)pencere + Noun + A3sg + Pnon (-de) + Loc.
(3) If there is a surviving prestressing marker, then the vowel of the preceding *syllable* receives the stress. Thus the word above in item 1 would have the representation of its pronunciation taS-laS-tl-r″a-ml-jor-duk.
(4) If there are no stress markers which occurs when no root stress or prestressing stress marker have been inserted, then the stress mark is inserted just before the vowel of the last syllable.

The stress mark is then moved to the preceding syllable boundary in the final representation as a convention. For the examples that we have been tracing all along, the final outputs will be as shown in Fig. 11. We can note that all outputs have word-final stress except for the 5th one, which originally had a surface morpheme tagged with the prestressing morpheme marker.

The only exceptions to these rules are forms of the question clitic (*mi*/*mu*/*mı*/*mü*), the relativizer clitic *ki* the emphasis clitics (*de*/*da*). All these are actually bound morphemes. The written convention is however is to separate these clitics (some of which can be followed, by other bound morphemes) and the preceding morphemes so they appear as different tokens in text. They are, prestressing, so any stress they induce goes on the previous token and they themselves do not bear any primary stress.

5.1. Handling unknown words

For handling unknown words, we use a second pronunciation lexicon which is consulted when the original analyzer fails. The architecture is essentially the same as the architecture in Fig. 1 but

[29] 'we were not being able to petrify (them)'.
[30] Monosyllabic roots with Sezer/exceptional stress have a different behavior. Any prestressing morphemes will override the root stress in such words.

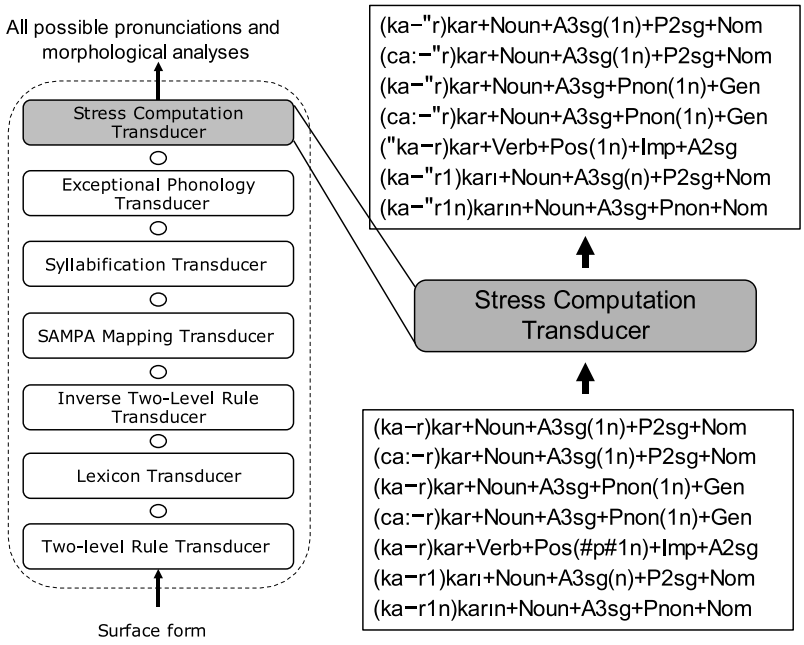

Fig. 11. Input and output of the stress computation transducer.

the morphological analyzer component is replaced by an unknown word analyzer. This analyzer is based on the original morphological analyzer, but with only a simple noun root lexicon that matches an arbitrary sequence of Turkish characters and the full suffix lexicon encoding the morphotactics. This analyzer will hypothesize (noun) interpretations of unknown character sequences subject to Turkish morphographemic and morphotactics constraints and will work properly even if the word has an unknown root but otherwise inflected properly (Oflazer and Tür, 1996). The only handicap of this analyzer is that since the generation of the root pronunciation involves a duplication process in the lexicon transducer, it is only possible to generate the root and its pronunciation in a character-interleaved fashion due to the limitations of the finite state formalism. This would then have to be massaged at the output by some scripts.

5.2. Inter-word interactions

There are also a couple of pronunciation phenomena that occur due to various inter-word interactions within a sentence. These phenomena are not marked in orthography.

(1) If the last consonant of a word with at least two syllables is one of /p, tS, t/ and the next word starts with a vowel, the last consonant is voiced and pronounced as /b, dZ, d/ respectively (except for a limited set exceptional lexical forms). For example *şarap içiyorum* (I am drinking wine) would actually be pronounced as /Sarab/ /itSijorum/. This is the analog of the consonant voicing process that is reflected to orthography when it is on a stem-morpheme boundary within a word.

(2) If the token preceding an emphasis clitic *de/da* ends in a voiceless consonant, then the initial *d* is actually pronounced as /t/. For example, *kitap da* would be pronounced as /citap/ /ta/, while *ev de* would be pronounced as /ev/ /de/.

Although these phenomena can also be implemented by finite state means on top of the word-level pronunciation transducer described above, by extending it so that it can handle a sequence of tokens, there are a number of engineering issues that prevent doing this in a pure finite state approach. Any unknown word would make the whole process fail even if all the other words in the sequence are known.

We have decided thus to implement the inter-word interactions by a non-finite state external process as a post-processor.

5.3. Disambiguation of pronunciation

Although the actual disambiguation of the contextually correct pronunciation is outside of the scope of this paper and is the topic of an ongoing-study (Külekçi, 2004), we would like to present here a discussion of issues involved supported by statistical information that would shed more light into the nature of this problem for Turkish.

We have run the words from a corpus of about 11,600,000 words through the pronunciation lexicon. Counting the outputs from all tokens except for punctuation and tokens involving numerals, but including unknown tokens, we have compiled the aggregate statistics given in Table 1. A detailed look at how these are distributed reveals the data in Tables 2 and 3.

The main point these numbers indicate is that, for uses of this lexicon in an application, such as phonetic transcription of acoustic speech data, that does not necessarily need to know the position of the primary stress, there really is not much of a disambiguation problem – over 98% of the

Table 1
Aggregate statistics over a 11,600,000 word corpus

Average morphological parse-pronunciation pairs/token	1.86
Average distinct morphological parses/token	1.84
Average distinct pronunciations/token	1.11
Average distinct pronunciations (ignoring stress)/token	1.02

Table 2
Distribution of parse-pronunciation pairs and parses

N	% of tokens with *N* parse-pron. pairs	Cumul. % of tokens with *N* parse-pron. pairs	% of tokens with *N* distinct parses	Cumul. % of tokens with *N* distinct parses
1	49.94	49.94	50.17	50.17
2	28.80	78.73	28.88	79.05
3	10.12	88.85	10.01	89.06
4	8.98	97.83	8.91	97.97
5	1.17	99.00	1.11	99.07
>5	0.99	100.00	0.92	100.00

Table 3
Distribution of pronunciation with and without stress marking

N	% of tokens with *N* distinct-prons.	Cumul. % of tokens with *N* distinct-prons.	% of tokens with *N* distinct-prons. (no stress)	Cumul. % of tokens with *N* distinct-prons. (no stress)
1	90.08	90.08	98.32	98.32
2	9.37	99.45	1.68	100.00
3	0.52	99.97	0.00	100.00
4	0.03	100.00	0.00	100.00

tokens have a single pronunciation when the position of the primary stress is ignored. The remaining ambiguities are almost always in differences in vowel length and consonant palatality in the root word portions of the words. The resolution of these, however, is not necessarily a simple problem: it involves applications morphological disambiguation and word sense disambiguation to correctly determine the root word and/or its sense. These however only need to be applied to couple hundred special words so the disambiguation models can be very focused. To give a flavor of the kinds of ambiguities that one encounters here we present the following:

(1) The root words are homographs but have different parts-of-speech: *ama* (/'a-ma/, `ama + Conj`, 'but') vs. *ama* (/a:-'ma:/, `ama + Adj`, 'blind').[31] Such cases can be disambiguated by morphological disambiguation.
(2) The root words are homographs and have the same part-of-speech; and further they inflect in exactly the same way: *kar* (/'kar/ 'snow') vs. *kar* (/'car/ 'profit') or *yar* (/'jar/ 'ravine') vs *yar* (/'ja:r/ 'lover'). This is akin to the disambiguation in English of *bass* ('fish') vs. *bass* ('musical instrument'). Morphological disambiguation would not be of much use here and one would have to resort to techniques of word sense disambiguation.
(3) The root words are homographs and have the same part of speech and pronounced the same, but under certain inflections, the root word with a certain sense undergoes further changes: For example for the word *hal* (/'hal/ 'fruit market' or 'state'), with the dative case marker suffix we get *hale* (/ha-'le/ `hal + Noun ... + Dat`) with the first sense vs. *hale* (/ha:-'le/ `hal + Noun ... + Dat`) with the second sense (and an additional reading *hale* (/ha:-'le/ `hale + Noun ... + Nom` 'halo')). We need to first disambiguate morphology here. If we predict that the word has nominative case, then we know the pronunciation and we are done. However, if we predict that the word has dative case, we now have to kick in a word sense disambiguator to select the appropriate pronunciation which sense of the root *hal* is used.

On the other hand, for an application such as text-to-speech, where the position of the stress is an important consideration, the situation is rather different: about 10% of the tokens have to be

[31] Although we indicated that we ignore stress for the purpose of this discussion, we still indicate the stress as the same examples also apply to the upcoming discussion.

disambiguated for both correct set of phonemes and the position of the primary stress. The issues involved in the disambiguation of the correct set of phonemes have already been discussed above. The differences in the position of the primary stress are due to a number of systematic morphological ambiguities, whether the word is a proper-noun or not, and occasionally, when a word is a proper noun, whether it denotes a location, person or some other named-entity:

(1) The words are homographs but morphological analysis produces multiple segmentations giving rise to free and bound morphemes with different semantics, morphosyntactic functions and stress marking properties: Here are some interesting examples:
   - *ajanda* (/a-'Zan-da/, `ajanda + Noun ... + Nom`, 'agenda') vs. *ajanda* (/a-Zan-'da/, `ajan + Noun ... + Loc` 'on the agent'). Here the first parse has a root word with exceptional root stress.
   - *fazla* (/faz-'5a/ `fazla + Adverb` 'much') vs. *fazla* (/'faz-5a/ `faz + Noun ... + Ins` 'with the phase'). Here, the instrumental case marking morpheme (*-la*) is prestressing, but happens to surface as the last two phonemes of the first root word.
   - *uyardı* (/u-'jar-d1/ `uy + Verb ... + Aor + Past + A3sg` 's/he/it used to fit') vs. *uyardı* (/u-jar-'d1/ `uyar + Verb ... + Past + A3sg` 's/he warned'). In the first interpretation, the morpheme marking past tense is prestressing when preceded by the aorist aspect morpheme, but not otherwise.
   - the *okuma* examples earlier (see page 7): *okuma* (/o-'ku-ma/ `oku + Verb + Neg +Imp+A2sg` 'don't read') vs. *okuma* (/o-ku-'ma/ `oku + Verb + Pos^DB + Noun + Inf...` 'reading')
   - *attı* (/'at-t1/ `at + Noun...^DB + Verb + Past + A3sg` 'it was a horse') vs. *attı* (/at-'t1/`at + Verb ... + Past + A3sg` 'he threw'). Similar to above, in the first interpretation, the morpheme marking past tense is prestressing when applied to a noun or adjective root (through an implicit verbal derivation.)

   Most of these cases can be resolved by morphological disambiguation but to a lesser extent in the last two cases, since most of the relevant morphological features are the same but the roots are different either in form or part-of-speech. Although proper identification of the root can be considered as part of the disambiguation process, it also has aspects closer to the word sense disambiguation problem (Hakkani-Tür et al., 2002).

(2) Proper nouns especially those denoting place names that are homographs with common nouns (inflected or otherwise) usually have non-final stress in the root affecting the stress properties of their inflected versions: e.g., *Ordu* (/'or-du/ 'name of a city') vs *Ordu* (/or-'d u/ 'army'). Although it may be possible to disambiguate whether a noun is a proper noun or not using orthographical cues such as initial capitalization and/or suffix separation characters, this may not always be possible and one may have to use again techniques akin to word sense disambiguation.

(3) The problem above is further complicated in cases where a proper noun is stressed differently when it denotes a place than when it denotes person, e.g., *Aydın* (/'aj-d1n/ 'city') vs. *Aydın* (/aj-'d1n/ 'person'). To disambiguate such cases one would have to resort to a named-entity recognition technique.

It is clear from the preceding analysis that a single technique is not necessarily sufficient to disambiguate the correct pronunciation for the ambiguous cases – one needs to employ an array of

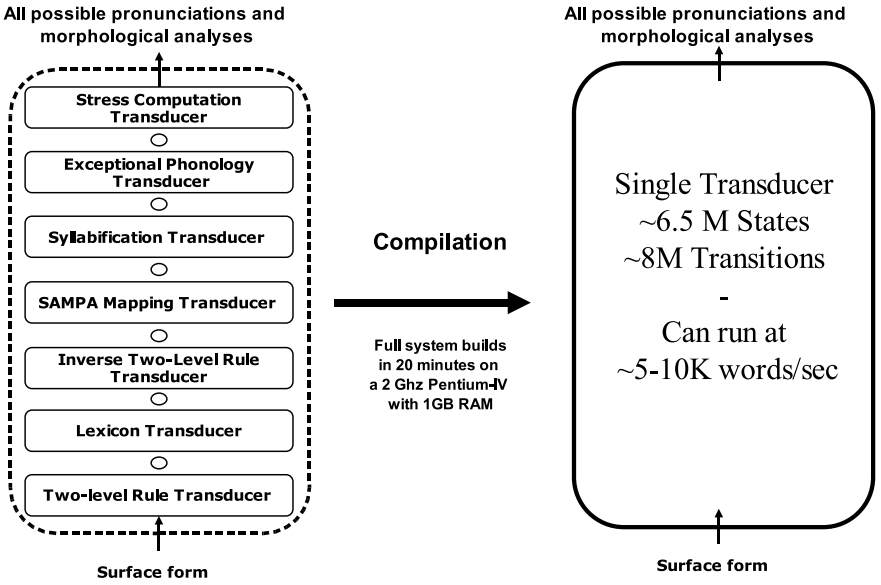

Fig. 12. All transducers are compiled into a single pronunciation lexicon transducer.

techniques. Even in the disambiguation of a certain word one may have to apply these techniques in sequential and incremental way to a single word and its context.

The statistics in Tables 2 and 3 also indicate for example that *full* morphological disambiguation may be an overkill for the purposes of disambiguating pronunciation, as there are more morphological analyses per token than there are distinct pronunciations. Thus, one needs to disambiguate enough of the morphological features so that the correct pronunciation can be deduced from just that much information, and a full-fledged morphological disambiguator which perhaps may have worse error rates need not be used. Further such disambiguation may need to be applied to a very limited set of systematic morphological ambiguity cases.

5.4. Some implementation details

The complete sentence pronunciation model is implemented using the Xerox Research Centre Europe (XRCE) finite state tools, *xfst*, *lexc*, and *twolc*.[32] Apart from various lexicons, the complete system is described by about 500 regular expressions in the XRCE Regular Expression Languages. As depicted in Fig. 12, all the components of the pronunciation lexicon described by these regular expressions can be compiled into one single transducer. The resulting transducer has about 6.5 million states and about 9 million transitions, a large finite state machine by any standard. This compiled transducer maps directly from surface word forms to outputs of the form shown in Fig. 11 and can run at about 5000–10,000 words per second depending on the platform and the length of the word list. If needed, it possible to directly extract from this transducer, transducers that just produce the morphological analysis or the pronunciations only.

[32] See http://www.xrce.xerox.com/.

The pronunciation lexicon is also of very high coverage and implements all word formation processes of Turkish. It has a noun root lexicon of about 25 K entries, a verb root lexicon of about 5 K entries and a proper noun lexicon of about 70 K entries.

6. Discussion and conclusions

We have presented the design and implementation of a wide-coverage finite state pronunciation lexicon for Turkish, an agglutinating language with essentially an infinite lexicon.

The lexicon produces a representation that encodes in parallel, the SAMPA representation of the all possible pronunciations of the word along with the corresponding morphological analyses. The correct computation of the pronunciation and the position of the stress requires that a full morphological analysis be done; consequently the pronunciation has been built on top of a two-level morphological analyzer with additional components for handling syllabification, various exceptional pehnomena and final stress computation. We have also presented a detailed discussion on issues in disambiguation of pronunciation, concluding that in order to correctly perform disambiguation, a single technique is not sufficient, and one would have to employ techniques from morphological disambiguation, word sense disambiguation, named-entity extraction and these techniques would have to be very focussed on specific morphological patterns and specific words.

The system has been implemented using the XRCE finite state tools and is available for experimentation at http://www.hlst.sabanciuniv.edu.

Acknowledgments

This work was supported in part by a joint National Science Foundation and TÜBİTAK (Turkish Scientific and Technological Research Foundation) project "A Unified Electronic Lexicon of Turkish". We also thank XRCE for making the finite state tools available, and the anonymous reviewers for helpful comments.

References

Beesley, K.R., Karttunen, L., 2003. Finite State Morphology. CSLI Publications, Stanford University.

Clements, G.N., Sezer, E., 1982. Vowel and consonant disharmony in Turkish. In: van der Hulst, H., Smith, N. (Eds.), The Structure of Phonological Representations, Part II. Foris, Dordrecht, pp. 213–255.

Gibbon, D., Simoes, A.P.Q., Matthiesen, M., 2000. An optimised FS pronunciation resource generator for highly inflecting languages. In: Proceedings of LREC 2000. Athens, Greece.

Hakkani-Tür, D., Oflazer, K., Tür, G., 2002. Statistical morphological disambiguation for agglutinative languages. Computers and the Humanities 36 (4).

Hankamer, J., 1989. Morphological parsing and the lexicon. In: Marslen-Wilson, W. (Ed.), Lexical Representation and Process. MIT Press.

Hazen, T.J., Hetherington, I.L., Shu, H., Livescu, K., September 2002. Pronunciation modeling using a finite-state transducer representation. In: Proceedings of the ISCA Workshop on Pronunciation Modeling and Lexicon Adaptation. Estes Park, Colorado.

Inkelas, S., 1999. Exceptional stress-attracting suffixes in Turkish: representations vs. the grammar. In: Kager, R., vanderHulst, H., Zonneveld, W. (Eds.), The Prosody–morphology Interface. Cambridge University Press, Cambridge, pp. 134–187.

Inkelas, S., Orgun, C.O., 2003. Turkish stress: a review. Phonology 20 (1).

Jurafsky, D., Martin, J.H., 2000. Speech and Language Processing. Prentice Hall, Upper Saddle River, NJ.

Karttunen, L., 1993. Finite-state lexicon compiler. XEROX, Palo Alto Research Center– Technical Report.

Karttunen, L., 2001. Applications of finite-state transducers in natural language processing. In: Yu, S., Paun, A. (Eds.), Implementation and Application of Automata. No. 2008 in Lecture Notes in Computer Science. Springer Verlag, Heidelberg, pp. 34–46.

Karttunen, L., Beesley, K.R., 1992. Two-level rule compiler. Technical Report, XEROX Palo Alto Research Center.

Karttunen, L., Chanod, J.-P., Grefenstette, G., Schiller, A., 1996. Regular expressions for language engineering. Natural Language Engineering 2 (4), 305–328.

Kiparsky, P., 1973. The inflectional accent in Indo-European. Language 49, 794–849.

Kornfilt, J., 1997. Turkish. Routledge, London.

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

Külekçi, O., 2004. Morphological disambiguation with distinguishing tags and its application to disambiguation of pronunciation, Ph.D Thesis Proposal, Sabancı University, Istanbul, Turkey.

Oflazer, K., 1994. Two-level description of Turkish morphology. Literary and Linguistic Computing 9 (2), 137–148.

Oflazer, K., Tür, G., 1996. Combining hand-crafted rules and unsupervised learning in constraint-based morphological disambiguation. In: Brill, E., Church, K. (Eds.), Proceedings of the ACL-SIGDAT Conference on Empirical Methods in Natural Language Processing.

Prince, A., Smolensky, P., 1993. Optimality theory: constraint interaction in generative grammar. Tech. Rep. RuCCS-TR-2, Rutgers University, ROA-537.

Sezer, E., 1981. On non-final stress in Turkish. Journal of Turkish Studies 5, 61–69.

Sproat, R., 1992. Morphology and Computation. MIT Press, Cambridge, MA.

van der Hulst, H., van de Weijer, J., 1991. Topics in Turkish phonology. In: Boeschoten, H., Verhoeven, L. (Eds.), Turkish Linguistics Today. E.J. Brill.