Chapter 7

Parsing Nominal Compounds in Turkish

Jorge Hankamer
University of California, Santa Cruz

The purpose of this paper is to present an outline of a part of a morphological parser which correctly parses nominal compounds in Turkish. Part 7.1 describes the solution to a relatively simple prior problem, that of parsing words formed by affixation. Part 7.2 presents a generative account of nominal compounding. The problem addressed in this paper is to design a parser which yields the same analyses for any well-formed construction as this generative grammar assigns. Part 7.3 outlines the compound parser.

7.1 Affixation

Affixation in Turkish is relatively straightforward from a generative point of view, though there are many questions of detail (for example, regarding the number and nature of the categories that must be distinguished) which can only be tentatively answered at present.

Briefly, an affixation rule is of the form\(^1\)

\(^1\)Since we are interested in parsing, and the method to be developed is a version of bottom-up parsing, I will systematically represent rules of formation in the form \(A + B \rightarrow C\), which is to be read “a unit of category \(A\) concatenated with a unit of category \(B\) counts as a unit of category \(C\)”. Thus a rule of formation corresponds exactly to the step in parsing where a derived stem is identified.
(1) \( XX + XXYY \rightarrow YY \)

where \( XX, YY \) are stem category symbols, and \( XXYY \) is an affix category symbol. This rule says that an item of category \( XX \) combines (by concatenation) with an item of category \( XXYY \) to yield an item of category \( YY \). As a concrete example:

(2) \( N0 + N0V0 \rightarrow V0 \)

\[
\begin{array}{c}
\text{kilit} & \text{+} & \text{le} \\
\text{‘lock’[N0]} & \quad & \text{[N0V0]} \\
\end{array}
\rightarrow \text{kitle}
\]

‘lock’[V0]

The number of stem categories that must be distinguished in a fully explicit morphological description turns out to be rather large. It is for that reason that I have adopted the convention of representing stem categories with two-character names. Each affix category consists of just those affixes that can attach to a stem of a given category and yield a stem of a given category (not necessarily distinct from the first one); hence it is convenient to name affix categories with four-character names representing their input and output stem categories.

An explicit description of word formation for a language, to the extent that it is agglutinative, can be given by defining the stem and affix categories for the language, and saying which of the stem categories count as word categories.

In most cases, the affix categories will be defined by listing. There are, however, some cases of complex affixes, where the affixes are composed of smaller affixes and might be defined by rule (though no such affix-defining rule, to my knowledge, will be recursive). Since affixes are categorized according to what stem category they attach to and what stem category they produce, the affix categories represent possible transitions between stem categories.

The stem categories, on the other hand, will in most cases require definition by recursive rules which refer ultimately to the affix categories and the primitive root categories, the members of which are listed.

The definition of the stem categories begins with a listing of the members of several primitive stem categories. These are the root categories, which include \( N0 \) (nouns), \( V0 \) (verbs), \( A0 \) (adjectives), \( P0 \) (postpositions), \( Q0 \) (predicates), and several minor categories. The full set of stem categories is then recursively defined by the listing of primitive categories and affix categories.

Finally, it is necessary to specify which stem categories count as word categories. To take a small example, the stem \( o \) counts as a word, but the stem \( on \) does not. In the system implemented here, a stem counts as a word only if there is a transition (or a chain of transitions) to a special category named “WW”. Let us consider one example:
(3) onlara  (o-n-lar-a “to them”)

- the root; standing alone, this stem would count as a word, so the grammar must provide transitions to the stem category WW.

- on  o + n yields a new stem, which is in a different category from o, since it cannot stand as a word, and this stem can have the plural affix attached to it, which o cannot. on cannot stand as a word, so the grammar must not provide a transition to the category WW except through further affixes.

- onlar  on + lar yields a new stem, which is in a different category from on. This is necessary because on may have lar attached to it, but onlar may not. onlar may stand as a word, so transitions to WW without further overt affixation should be provided for.

- onlara  onlar + a yields a new stem, which is in a different category from onlar. This is necessary because onlar may have a attached to it, but onlara may not. No further affixation is possible, and onlara is a word, so there must be a transition to WW from this category.

7.1.1 The Affixation Parser

Given a model of agglutinative morphology like that presented above, it is relatively straightforward to design a parser which takes advantage of the fact that the morphology (i.e. the definitions of the stem and affix categories) is representable as a finite state transition network, where the root categories represent jumps from the initial state and the final state is WW.

The root and affix categories are specified in two files, a lexicon file and an affix file.

The root lexicon consists of a list of roots with associated category marking (together with a gloss and other grammatical and semantic information, which will not concern us here).

The affixes are listed in categories according to what category of stem they attach to and what category of stem they yield.

The parser tests the word for an initial match with a root; when one is found, the category of the root determines the category in which the next match is sought. If the root is an N0, then the next match must be with a member of a category N0YY, i.e. an affix which can attach to a stem of category N0. If a match with such an affix is found, the process is repeated with the derived stem as base: its category determines the class of affixes to be tested for a match with the next part of the word.
If the end of the word is reached and the final stem is in a category which has a transition to category WW, the parse is deemed a success; otherwise it is a failure. In particular, if at any point there is a remaining substring in which no match with an affix in the required category can be found, the parse is abandoned.

The program incorporates the major phonological rules of Turkish. Roots and affixes are listed in the lexicon and affix files in an abstract underlying representation, and matches with the surface forms are mediated by modifications sensitive to the environment. For example, the past affix is listed as *al*; this form is modified by operations corresponding to the rules of vowel harmony and stop assimilation before the form is matched to a surface string. Thus *alda* will be analyzed as

\[
\text{al-d} \\
\text{take-PST}
\]

but *al* will not be analyzed (wrongly) as

\[
\text{al-t} \\
\text{take-PST}
\]

because the affix-initial consonant would not match the stem-final consonant in voicing.

Similarly, even though the dative affix is listed as *ya*, the string *adanya* will not be parsed. The affix matching routine deletes the buffer from the affix before matching, because the stem *adam* ends in a consonant.\(^2\)

Here is an example of the parsing of a moderately complex word. Given the form *gözlükçülerimizden* the analysis proceeds as follows:

First the root *göz* `eye' is recognized; its lexical entry specifies that it is an N0; then an affix is sought that can attach to N0 and initially matches the string after *göz*. *hg* matches, and is in the category N0N0; its derived category is N0, so *gözük* counts as an N0, Now the program again seeks a match with affixes that can attach to N0. The affix c1 is among them, and matches the string *gü* that follows *gözük*. The category of the new affix is N0N0, so the next affix again has to be in a category N0YY. This time no overt affix is found to match the next part of the word, so a

\(^2\)The system as currently implemented has some imperfections. It does not, for example, have a way of dealing adequately with morphophonemic alternations that are conditioned by the number of preceding syllables, as is the case with the causative and aorist affixes. This causes the parser to accept some false parses (*al-t-t*[take-CAUS-PST] for *altit*, for example). It also has no way of knowing that *gidece* [go-ABL-NEG] is not a likely imperative, but that is probably a semantic rather than a morphological limitation.
jump is taken to category N1.  

From N1 there are jumps to N2 and to N3. If the jump to N2 is taken, the next affix must be the plural affix lar; if the jump to N3 is taken, the next affix must not be the plural. Both paths will be attempted, but the one which commences with a jump to N3 will fail, since there is no way to reach the end of the word along that path. The path commencing with a jump to N2 leads to the correct analysis.

The only affix that can attach to an N2 is the plural affix lar, which is the sole occupant of category N2N3; this matches the ler after gözükçü, so gözükçüler is analyzed as a stem of category N3. Note that a stem of category N3 may or may not have the plural affix.

From N3 there are jumps to N4 (from which there is further progress only by finding a possessive affix), to N7 (from which there is no escape except by finding a case affix); and to N9 (no further affixation required). To find an analysis for the given word, the parser must recognize the possessive affix imiz (category N4N5) as matching imiz, so that gözükçülerimiz is analyzed as a stem of category N5; then a jump is taken to N7, from which the ablative affix dan (category N7N9) can be matched to yield gözükçülerimizden as a stem of category N9. There is a jump from N9 to NN and another jump from NN to WW, so the analysis is a success.

I will not discuss in detail the reasons for all of the category distinctions. Those which figure in the operation of the compound parser outlined in section 7.3 are N0, the category of noun roots and noun stems derived by derivational affixes; N3, the category consisting of noun stems with or without the plural affix but lacking possessive or case affixes; N4, the stem category which must acquire a possessive affix; and N6, which is an N4 with the third person singular possessive affix (affix category N4N6) attached to it.

The parser described in this section is extremely simple, yet it seems to be adequate to deal with the agglutinative aspects of Turkish morphology. What it cannot do, without some sort of radical embellishment, is assign correct analyses to compounds.

7.2 The Structure of Nominal Compounds

The nominal compounds to be treated here are those of the "indefinite ızafer" construction (Lewis (1956), pp. 41–43). As a first approximation, their structure might

---

3Category N1 is distinguished from category N0 because adjectival stems (in particular participles derived from verbs) can function as noun stems, receiving the plural, possessive, and case affixes that affix to noun stems, but cannot receive the N0YY-derivational affixes.

4After this paper was substantially written, I learned that a morphological parser based on similar principles has been developed for Finnish by Keskenniemi (1983).
be described as

(4) N3 + N6 → N0

where N3 is a noun stem with or without a plural affix\(^5\) and N6 is a noun stem with the third person singular possessive affix. This affix is the sole member of the category N4N6; it is glossed POSS in the examples below.

(5) a. \(\text{çocuk} \quad \text{kitab-ı}
\)  
    \begin{align*}
    \text{child} & \quad \text{book-POSS} \\
    \text{‘children’s book’}
    \end{align*}

b. \(\text{ev} \quad \text{kapi-sı}
\)  
    \begin{align*}
    \text{house} & \quad \text{door-POSS} \\
    \text{‘house door’}
    \end{align*}

c. \(\text{kız} \quad \text{okul-u}
\)  
    \begin{align*}
    \text{girl} & \quad \text{school-POSS} \\
    \text{‘girls’ school’}
    \end{align*}

This construction must be distinguished from the “definite izafet” construction, exemplified in (6):

(6) a. \(\text{çocuğun} \quad \text{kitab-ı}
\)  
    \begin{align*}
    \text{child-GEN} & \quad \text{book-POSS} \\
    \text{‘the child’s book’}
    \end{align*}

b. \(\text{ev-in} \quad \text{kapi-sı}
\)  
    \begin{align*}
    \text{house-GEN} & \quad \text{door-POSS} \\
    \text{the door of the house’}
    \end{align*}

c. \(\text{kız-in} \quad \text{okul-u}
\)  
    \begin{align*}
    \text{girl-GEN} & \quad \text{school-POSS} \\
    \text{‘the girl’s school’}
    \end{align*}

In the definite izafet construction, the first member is in the genitive case. The definite izafet construction is in fact quite different from nominal compounding. Its structure is represented by the following rule:

(7) NP[gen] + NP[poss] → NP

---

\(^5\) Lewis (1956) (p. 44, footnote 1) indicates that he thinks the first members of compounds should not have the plural affix. This is probably incorrect, however, for I have seen such things as \(\text{költür okula} \) ‘school for the blind’, and forms like \(\text{uluslararası} \) ‘international’ appear to be well established.
Here “NP” is the phrase category Noun Phrase, not a morphological category. NP[gen] is a NP whose head noun has a genitive affix on it, and NP[poss] is an NP whose head noun has a possessive affix on it.  

This claim is supported by the fact that in the “definite izafet” either the first or the second member may be modified by any of the kinds of complements that are found in NP’s:

(8) küçük çocuğu kitabi
    ‘the small child’s book’
    çocuğu küçük kitabı
    ‘the child’s small book’
    küçük çocuğu küçük kitabı
    ‘the small child’s small book’
    çocuğu bir kitabı
    ‘a book of the child’s’
    adamın çocuğuğun kitabı
    ‘the man’s child’s book’
    senin gördüğün çocuğunun kitabı
    ‘the book belonging to the child that you saw’
    çocuğunun senin gördüğün kitabı
    ‘the book that you saw that belongs to the child’

while in the “indefinite izafet” all modifiers must precede the first member of the construction and semantically apply to the construction as a whole:

(9) küçük çocuk kitabı
    ‘small children’s book’ (small book for children)
    *cocuk küçük kitabı
    bir çocuk kitabı
    ‘a children’s book’
    *cocuk bir kitabı
    senin gördüğün cocuk kitabı
    ‘the children’s book that you saw’
    *cocuk senin gördüğün kitabı

6The possessive affix in this case is any N4YY affix, not necessarily N4N6; it must agree with the genitive affix in NP[gen].
There is one further construction which must be set aside. Turkish grammarians (cf. Lewis (1956), p.42) regarded constructions like those in (10) as compounds:

(10)  
\begin{align*}
\text{tahta kutu} & \quad \text{‘wood box’} \\
\text{demir perde} & \quad \text{‘iron curtain’} \\
\text{bakır tas} & \quad \text{‘copper bowl’}
\end{align*}

In this construction the first member must be a noun denoting a material, the second a noun denoting a type of object, and the whole denotes an object of that type made from that material. Neither member is marked by a suffix.

I deal with these constructions by entering the material nouns twice in the lexicon, once as nouns and once as adjectives. This accounts for the fact that such material nouns, when combined with nominal compounds of the indefinite izafet type, invariably precede the first member of the compound, and cannot intervene between the two members:

(11)  
\begin{align*}
\text{tahta sigara kutusu} & \quad \text{‘wooden cigarette box’} \\
*\text{sigara tahta kutusu} & \quad \text{–}
\end{align*}

Henceforth, then, the term “nominal compound” will mean an instance of the “indefinite izafet” construction.

If rule (4) describes the basic structure of a nominal compound, it is necessary to formulate some transformational rules to modify that structure under certain conditions. The remainder of this section is devoted to the development of a transformational description of nominal compounds proceeding from this assumption. In the following section I will show how to describe the structure of compounds without such transformations.

One reason there must be some transformational modification of compounds if (4) is assumed to be the basic rule of formation is that any time two possessive morphemes (of any person and number) would, according to the rules of combination, appear adjacent to each other, the innermost (leftmost) one is deleted. Thus if the nominal compound is made the right member of a definite izafet construction, we find the following:

(12)  
\begin{align*}
\text{Hasan-}
\text{m ev anahtar-} & \quad \text{‘Hasan’s house key’} \\
*\text{Hasan-}
\text{m ev anahtar-}
\text{\texttt{\textasteriskcentered}si} & \quad \text{–} \\
\text{Hasan-}
\text{m at arab\texttt{\textasteriskcentered}a-} & \quad \text{‘Hasan’s horse cart’} \\
*\text{Hasan-}
\text{m at arab\texttt{\textasteriskcentered}a-}
\text{\texttt{\textasteriskcentered}si-} & \quad \text{–}
\end{align*}
The possessive morpheme never appears immediately preceding another possessive morpheme. When the rules of combination would produce such a sequence, the leftmost of the two adjacent possessive morphemes vanishes:

(13) N4N6 → ⌀/ — N4YY (N4YY is the class of possessive affixes)

We know it is the leftmost possessive morpheme that vanishes because in genitive-possessive constructions the two may represent different persons:

(14) at arabâ-sî ‘horse cart’
    (benîm) at arabâ-m ‘my horse cart’
    *at arabâ-st-m

The 3sg possessive morpheme {s} vanishes when adjacent to a following 1sg possessive morpheme.

This same rule applies no matter what the source of the possessive morphemes. In particular, the nominal compounding rule is recursive, and indeed we find compounds in which one member or the other is itself a compound; see Figure 7.1.

türk dîl-i gramer-i ‘turkish language grammar’
türk dîl kurum-u ‘turkish language society’

\[
\begin{align*}
\text{N} & \quad \text{N[poss]} \\
\text{N} & \quad \text{grameri} \\
\text{N} & \quad \text{tûrk} \\
\text{N} & \quad \text{dîli} \\
\text{N} & \quad \text{kurum-Ø-u}
\end{align*}
\]

Figure 7.1: Recursive compounding

Note that when it is the second member that is a compound, we find not two possessive morphemes but one.

Similarly, the possessive morpheme vanishes when adjacent to a following derivational affix (Lewis (1956), p. 50):

(15) su yol-u ‘water conduit’
    su yol-cu ‘person responsible for the upkeep of water conduits’

(16) N4N6 → ⌀/ — N0YY
These examples are interesting, because they show not only that the domain of poss-deletion is more general, but also that compounds may undergo derivational affixation. In the terms of the model of morphological structure sketched in section 7.1, this means that a compound can count as an N0.

Another transformational rule affecting the appearance of nominal compounds is one which determines the order of plural and possessive suffixes. The relative order of plural and possessive morphemes is fixed: no matter what the origin, a plural morpheme must precede an adjacent possessive morpheme. Morphemes in the order possessive-plural can arise under the current assumptions because rule (4) makes a compound an N0, and N0 (after jumps to N1 and N2) can receive a plural affix; see Figure 7.2.

*ev anahtarilar

```
  N3
  /   \
N2    N2N3
  \
   N1
    N0
```

```
  N1
  /   \
N0    N6
  \
   N3
    N4N6
  |
   N4
  |
   N3
  |
   N1
  |
   N0
```

Figure 7.2: Structure of *ev anahtarilar

Thus we will need a metathesis rule:

(17) N4N6 N2N3 → N2N3 N4N6

Thus we have, as the plural of ev anahtarı “house key”

(18) ev anahtarılar 'house keys'
    *ev anahtarılar

There is yet one more transformation to be formulated. The grammar now generates sequences of two plural affixes. This is because the stem anahtar is eligible to receive
*ev anahtarlarlar

N3
  └── N2
      └── N1
          └── N0

  N3
  └── N6
      └── N1
          └── N0
              └── ev
                  └── N3
                      └── N4
                          └── N4N6

  N2
  └── N2N3
      └── N1
          └── N0
              └── anahtar

Figure 7.3: Structure of *ev anahtarlarlar

a plural affix just as the whole compound is, in which case the structure would be as in Figure 7.3.

Once rule (17) permutes the possessive affix with the outer plural affix, the two plural affixes are adjacent. Since such sequences never appear superficially, there will have to be a transformation deleting one of two adjacent plural affixes.

(19) N2N3 \rightarrow \emptyset / N2N3

The construct *ev anahtarlarlar is in fact several ways ambiguous, because of the conflation of the forms corresponding to "house keys", "his/her/its house keys", "their house keys", "their house key".

Rule (17) may feed both the possessive deletion rule (13) and the plural deletion rule (19).

Together with the affixation rules described in section 7.1, rules (4), (13), (16), (17), and (19) account for the structure of nominal compounds. This description is a formal version of the informal description given by Lewis (1956).

Some examples are shown in Figures 7.4, 7.5, and 7.6.

In Figure 7.4, rule (17) permutes the inner poss (N4N6) morpheme with the plural (N2N3) morpheme; rule (13) then deletes the inner poss morpheme, and the resulting string is shown in Figure 7.4. The rules I have written are not explicit
about certain details of the intermediate structures, in particular what the structural effects of the permutation rule are; this is because those details do not appear to be empirically determinable, and as will be shown in the next section are irrelevant to the parsing analysis.

In Figure 7.5, nothing happens but the deletion of the poss morpheme by rule (16).

In Figure 7.6, rule (13) deletes the innermost poss and rule (16) deletes the next one, assuming a cyclic application of these rules. It doesn’t matter whether this is assumed or not, since any other order of application will have the same net effect.

The description sketched in this section is transformational, in that it assumes that the construction rules produce basic structures which are then modified by permutation and deletion. While this model describes the desired forms in a generative fashion, it is not immediately amenable to a practical parsing interpretation. It is, I believe, possible to construct a parser corresponding to this grammar, i.e. one which accepts the same strings and assigns them the same underlying structures. Such a parser, however, would have to undo much of the work of the affixation parser in unravelling the word structure and replacing the missing N4N6 morphemes.
This is not just a mechanical inconvenience. There is something fundamentally wrong with this analysis, whether we want a parser or not. The basic word formation rules, as represented in the affix transition network, stipulate that a word may have at most one possessive affix and at most one plural affix, that no possessive affix precedes a derivational affix, and that if there are both a plural and a possessive affix, the possessive must precede the plural. The transformational rules (13), (16), (17), and (19) have the effect of restoring the second word in a compound to the state that the basic rules say it must have, in effect undoing the work of rule (4).

We must suspect that something is wrong, and in particular that rule (4) is wrong. Note that similar observations formed the basis for Chomsky's (1969) attack on the transformational analysis of derived nominals, for the situation is the same: the transformational account is suspect because the transformations are required to derive structures which are already described by basic rules.

In the next section, a non-transformational analysis will be outlined which accepts exactly the same strings as are generated by these rules, while assigning them correct structures. The analysis will be presented as a parser, but (like the analysis of agglutination outlined in section 7.1) it can equally well be regarded as a generator.
7.3 Parsing Nominal Compounds

The generative grammar developed in section 7.2 admits the correct class of nominal compounds.

The problem now is to devise a parser which will accept as input a string of words and determine whether the string constitutes a well-formed nominal compound or not, assigning it a correct analysis if it does.

The course adopted here is to allow the affixation parser described in section 7.1 to provide parses for each word, and then to pass the resulting information to a secondary parser which assigns an appropriate structure to the compound, or rejects it if no structure can be assigned. The structures assigned will differ in certain details from those assigned by the grammar of section 7.2, but the major
structural relations (the relations between the nominal members) are the same.

The secondary parser proceeds by identifying possible first and second members of compounds in the bracketed string output by the suffixation parser. It then assigns new bracketing by an "annexation" operation, which makes the left member a sister of the right member, leaving other structural relations intact. The creation of a compound may produce new possibilities for combination, since compounds themselves may be candidates for membership in larger compounds.

The first task, then, is to identify possible left and right members of simple compounds. The grammar of section 7.2 says that compounds are composed of \([N3 + N6]\); but the transformations (13) and (16) may strip the \(N4N6\) suffix off, so that the affixation parser cannot analyze the form as an \(N6\). If we look at the output of the affixation parser as applied to a compound, we find that simple compounds may be identified as \([N3 + N0]\) followed immediately by a derivational affix or by a possessive affix with at most a plural affix intervening.

This suggests that we may begin by identifying the sequence \([N3 + N0]\) as a compound if the environment immediately following is one of the two mentioned above. Once simple compounds are identified, we may observe that an \(N3\) may combine with a following compound to form a compound of the \(türk \text{ dil kurumu}\) type; and that a compound plus its possessive affix (with an optional plural affix in between) can combine with a following right member to form a compound of the \(türk \text{ dili grameri}\) type.

All this is summarized in the following two rules:

\[
\begin{align*}
(20) \quad N3 & \rightarrow N0 \quad \left\{ \begin{array}{l}
N0N0 \\
(N2N3) N4YY
\end{array} \right.
\end{align*}
\]

\[
(21) \quad N3 \quad N0 \quad (N2N3) \quad N4N6 \rightarrow N3
\]

The first rule identifies a sequence \(N3 \ N0\) as an \(N0\) if it is in one of the contexts where compounds can occur; the second rule says that a compound plus its plural (if present) and 3sg possessive affixes counts as an \(N3\) for purposes of further compounding. These rules are strictly cyclic.

The rules are cyclic because units created by them may be components of larger units also created by them. They are "strictly" cyclic because it would not do to allow either rule to iterate on the same domain, yielding multiple analyses like the following:
(22) su yolcu

\[
{\begin{array}{ccc}
[su] & [yol] & -cu \\
N & N & N \\
3 & 0 & 0 \\
\end{array}}
\]

The operation represented by rule (20) may be regarded as chomsky-adjunction of the left member to the right member.

Here is an example:

(23) türk dili grameri

The affixation parser produces the following output: 7

\[
{\begin{array}{ccc}
türk & dil & -i \\
N & N & N \\
3 & 6 & 0 \\
\end{array}}
{\begin{array}{cc}
gramer & -i \\
N & N \\
N & N \\
3 & 6 & 0 \\
\end{array}}
\]

Rule (20) annexes türk to dil:

\[
{\begin{array}{cccc}
türk & dil & -i & gramer \\
N & N & N & N \\
3 & 6 & 0 & 6 \\
\end{array}}
\]

Rule (21) says türk dili counts as an N3:

\[
{\begin{array}{cc}
N & 0 \\
\end{array}}
\]

7 I am omitting null transitions which are ignored by the secondary parser.
Rule (20) combines türk dili with gramer:

\[
türk \quad dīl \quad -i \quad gramer \quad -i \\
N \quad N \quad N \quad N \quad N \\
3 \quad 0 \quad 6 \quad 0 \quad 6
\]

\[
N \\
0 \\
N \\
3
\]

Rule (21) now says that the whole expression counts as an N3.

\[
türk \quad dīl \quad -i \quad gramer \quad -i \\
N \quad N \quad N \quad N \quad N \\
3 \quad 0 \quad 6 \quad 0 \quad 6
\]

\[
N \\
0 \\
N \\
3 \\
N \\
0 \\
N \\
3
\]

The effect of these reanalyses is to extend the categories N0 and N3. But because the compounding analysis is a second stage operation, presuming that affixation has already been dealt with by the affixation parser, there is no possibility of the system accepting a possessive morpheme affixed to one of these secondarily derived N3's. Sequences of adjacent possessive morphemes are ruled out once and for all by the affixation parser.

The analysis of an example like türk dīl kurumu will proceed as follows:
Finally, the analysis of *su yolcu dergisi*:

<table>
<thead>
<tr>
<th>su</th>
<th>yol</th>
<th>-cu</th>
<th>dergi</th>
<th>-si</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

The structures assigned by these rules correspond in all major respects to those assigned by the generative rules of section 7.2. The structural relations assigned in the *su yolcu* type are clearly correct, as shown in Figure 7.7.

![Diagram](image)

**Figure 7.7: Structure of *su yolcu***

It may seem bizarre at first glance that the possessive suffix in *su yol* is not treated as part of the compound but rather as a sister to it; see Figure 7.8.

This appears to be empirically harmless, however. The structure assigned by
the affixation parser already says that the possessive morpheme is part of the last word in the compound, and that fact is not changed. The compounding parser may be regarded as concerned only with assigning structural relations among the N0s in the compound, so it treats the possessive morpheme as if it were as high up as possible just to have it out of the way.

I will conclude this section with a discussion of one interesting example which demonstrates a slightly more complex interaction of the compounding rules. The following example is, in the first place, ambiguous:

(24) türk yolculuk dergisi

It can be analyzed several ways, depending on the order in which the elementary compound parts are combined:

<table>
<thead>
<tr>
<th>türk</th>
<th>yol</th>
<th>-cu</th>
<th>-luk</th>
<th>dergi</th>
<th>-si</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

One possible initial combination, that of yolculuk with dergi, leads to a compound of the same structure as türk dil kurumu, discussed above. It would mean 'travel journal of Turkey'.

If instead türk is first combined with yolcu, we get the following analysis:
(26) ‘journal of Turkish travel’

\[
\begin{array}{cccccc}
\text{türk} & \text{yol} & -\text{cu} & -\text{luk} & \text{dergi} & -\text{si} \\
N & N & N & N & N & N \\
3 & 0 & 0 & 3 & 0 & 6 \\
\end{array}
\]

Rule (20)

Note that the application of Rule (20) that makes \textit{türk yolcu} an N0 also makes \textit{türk yolculuk} an N0, and of course also an N3.

If \textit{türk} is first combined with \textit{yol}, we get the following analysis:

(27) ‘journal of Turkish road work’

\[
\begin{array}{cccccc}
\text{türk} & \text{yol} & -\text{cu} & -\text{luk} & \text{dergi} & -\text{si} \\
N & N & N & N & N & N \\
3 & 0 & 0 & 3 & 0 & 6 \\
\end{array}
\]

Rule (20)

Here the first application of rule (20) has made \textit{türk yol} an N0, which has the consequence that \textit{türk yolcu} and \textit{türk yolculuk} are also N0. \textit{türk yolculuk} is also an N3.
I have presented the analysis as a two-stage parser. It can equally well be viewed as a two-stage generator. The affixation rules can be viewed as generative rules which build stems of various types: N0, N3, N6, etc. The two rules (20) and (21) can be viewed as generative rules which, like the generalized transformations of Syntactic Structures, put the prefabricated stems together in specified ways. Under this view, rule (20) is an embedding transformation.

7.4 Conclusion

The algorithm outlined in section 7.3 recognizes exactly the compound nouns generated by the rules presented in section 7.2. The structures assigned to these compounds differ from those assigned by the rules of section 7.2 only in the question of the structural relation between the possessive morpheme and the rest of the compound.

The virtue of this analysis, besides the fact that it yields a correct and rather straightforward parser, is that the restrictions on affix occurrence and order, which are properties of simple words, are stated once and for all in a component of the grammar which defines simple word structure. None of the patch-up transformations which are required under the analysis sketched in section 7.2 are needed under this analysis.

On the other hand, it might be regarded as a virtue of the transformational analysis that it provides multiple derivations for ambiguous expressions such as ev anaktarlari. The treatment of section 7.3 gives this expression only one analysis. If a correct interpretation of the expression is to be provided as well as a correct parse, it will be necessary to provide for some rules of semantic interpretation which specify the possible contributions of the possessive and plural morphemes to the interpretation of the expression.