ANADOLU ÜNİVERSİTESİ BİLİM VE TEKNOLOJİ DERGİSİ
ANADOLU UNIVERSITY JOURNAL OF SCIENCE AND TECHNOLOGY
Cilt/Vol.:3 - Sayı/No: 1 : 59-76 (2002)

ARAŞTIRMA MAKALESİ/**RESEARCH ARTICLE**

THE ROLE OF MORPHOLOGICAL ANALYSIS IN NATURAL LANGUAGE PROCESSING

Zeynep ALTAN[1]

ABSTRACT

Traditionally, the analysis of word structure (morphology) is divided into two basic fields as inflection and derivation. Therefore, the morphological structure of each word may include elements such as prefix, suffix, infix, or even a separate root, and these elements can modify the meaning of the basic root or stem of the word. If the consequent word is only a paradigmatic application of its base form, this variation of the word is called inflection; but if the resulting word is an entirely different word or a compound, which is formed of two or more roots, it is called derivation. While derivation is a word-creating process, inflection constitutes different forms of any word. The model developed in this study, which analyses the morphology of Turkish verbs, can recognize all of the inflectional categories. The computational tool consists of a Java applet that can run on every machine, and a database that has been extracted from Turkish Dictionary published by Turkish Language Society. The database includes both the verb roots and derived verbs. We utilize Koskenniemi's two-level system to develop the morphological model. The input verb, which precedes the suffixes, is analyzed as an invariant root by querying the database, and the following suffix particles may indicate voice (causative, reciprocal, reflexive, passive), modality (necessitive, abilitative, conditional), negation, tense-aspect mood and person/number.

Key Words: Morphology, Recognition, Generator, Syntax, Finite state transducers, Two-level rules.

DOĞAL DİL İŞLEMLEMEDE BİÇİMBİLİMSEL ANALİZİN ROLÜ

ÖZ

Kelime yapısının analizi (biçimbirim), geleneksel olarak çekim ve türetme şeklinde iki temel kısma ayrılır. Her kelimenin biçimbilimsel yapısı önek, sonek veya bağımsız kök gibi bazı ilave öğeler içerebilir ve bu öğeler temel kök veya gövdenin anlamını değiştirebilir. Eğer sonuçta elde edilen kelime başlangıç halinin sadece çekim örneğini oluşturuyorsa, bu değişime çekim adı verilir; oysa tamamen farklı bir kelime veya iki veya daha fazla kökten oluşan bir bileşen ise, buna da türetme adı verilir. Türetme, bir sözcük-oluşturulması yöntemi iken; çekim, verilen bir sözcüğün farklı bir biçiminin oluşturulmasıdır. Türkçe eylemlerin biçimbilimini analiz eden ve bu çalışmada geliştirilen model tüm çekim ulamlarını tanımaktadır. Berimsel paket her bilgisayarda çalışabilen bir Java applet ve hem kök durumunda hem de türemiş durumdaki eylemlerin yer aldığı bir veri tabanından oluşmaktadır. Biçimbilimsel modeli geliştirmek için Koskenniemi'nin iki-düzeyli sisteminden yararlanılmıştır. Giriş kelimesi, tüm eklerden önce gelen değişmeyen kök olarak veri tabanından sorgulanarak analiz edilmektedir; daha sonra gelen ekler ise çatı (ettirgen, işteş,dönüşlü,edilgen), kip durumu, olumsuzluk, asıl zaman kipi ve kişi ekini simgelemektedir.

Anahtar Kelimeler: Biçimbilim, Tanıma, Üreteç, Sözdizimi, Sonlu durum dönüştürücüleri, İki-düzeyli kurallar.

[1] Istanbul University, Faculty of Engineering, Department of Computer Engineering, 34850, Avcılar-İstanbul.
E-mail: zaltan@istanbul.edu.tr.

Received: 20 November 2000; **Revised:** 21 March 2001; **Accepted:** 11 June 2001.

1. INTRODUCTION

Computer science and linguistics have developed together over several decades. Formal language theory has also triggered the improvement of computational linguistics. Transformational generative grammar definitions of Chomsky were the first applications of formal language theory. These hierarchical definitions with several types help to understand the modeling of natural language syntax. Although finite state languages were the first and simplest types of formal language definitions, Chomsky later rejected them as proper domain of natural language theory. He proposed that the theory of syntax requires different language definitions, which does not include finite state properties. However, morphology and phonology can easily be implemented by using finite state models. Inflectional and derivational morphology play an important role in morphological theory. The distinction between the inflection and derivation has thoroughly been studied with ALE (Attribute- Logic Engine) formalism (Matheson, 1995). Since the distinction between derivational and inflectional morphology is not definite in some languages, it may be impossible to define a straightforward classification of these morphological properties.

Morphological analysis is very meaningful for the determination of part-of-speech structure in syntactic parsing, and for the semantic analysis of a sentence. Information about verbal inflection is especially important for the word order concept. Moreover, a word may define two or more expressions. For example, the verb *açmak* (to open) may associate different phrases as *çiçek açıyor* (flower is blooming) or *hava açıyor* (weather is clear). The morphological ambiguity of an inflectional property can also affect the nouns. For example, the word *elbiselerinin* includes four different meanings as

elbise + ÇE + ST + İE (2.Singular) + Determining (….of your clothes),

elbise + ÇE + İE(3.Singular) + KH + Determining (….of his/her clothes),

elbise + ÇE + Determinated + KH+ Determining (….of …..'s clothes),

elbise + İE(3.Plural) + KH + Determining (….of their clothes).[2]

Different meanings of the verb (also other words) in a sentence cause multiply analysis results. Turkish is an agglutinative language, and the words are generally constituted in the subject-object-verb sequence. Therefore, the sentences may possess many different meanings after the derivation of the verbs with more than one suffix. Verb voice is defined as all varieties of the verb base.[3] The effect of the subject in a sentence can be described after the analysis of voice affixes. The verb classification according to the structure, i.e. whether it is simple, derived, or compound verb, is unnecessary for the analysis of voice. However, the compound verbs are grouped according to their formations. If any compound verb is constituted with an auxiliary verb such as *etmek* (to do) or *olmak* (to be), this formation must be declared in the morphological analysis. This time not only the verb, but also the previous word is required for the semantic analysis.

Because of the contributions of Turkish to computational applications and the language's rich linguistic properties, it is approved in linguistic theory. Oflazer is one of the Turkish researchers who have been studying on natural language processing, and he is also one of the participants of Turkish Natural Language Processing Initiative Project (TNLP), which was a collaborative research for the analysis of Turkish texts (Oflazer and Bozşahin, 1994). It was a big project called TU-Language, and was funded by NATO Science. Some of the other participants of this project were Bozşahin, Göçmen, Yılmaz, Yıldırım, Öztaner and Şehitoğlu. With the support of this project, Turkish syntax has completely been characterized in a technical report according to certain distinctive properties of the language as noun, postposition, adjective, adverb and verb groups (Göçmen et al., 1995). Alternative computational models of morphology that came out from that project provided many contributions to the language studies, especially to the morphological studies of Turkish. The first detailed study on developing finite state parsing systems for Turkish has been carried out by Oflazer. The computational approach was the two-level morphological analysis of inflectional and derivational morphemes (Oflazer, 1993). Another morphological description, which is also about computational morphological analysis and generation of Turkish word forms, has been encoded using two-level morphological model (Öztaner, 1996). The other study about the morphology of Turkish happened as a teaching tool, assisting the computer aided language learning for non-native learners of Turkish and student of linguistics (Pembeci, 1998). Different kinds of lexical rules have also been described for inflections and derivations as the morphology-lexicon-syntax interface of Turkish (Şehitoğlu and Bozşahin, 1999). In this study, the productivity of the lexicon is examined for the agglutinative morphology of Turkish.

[2] These marks symbolize the following meanings:
ÇE - plural suffix, KH - fusion word, ST-sound derivation and İE - possessive suffix. This result has been obtained from the tool, which analyses the Turkish nouns [Altan and Aydın, 2000].

[3] If the word is a root, radical (verb base derived with different derivational affixes from noun or verb) , compound or a borrowed word, it is called as base with its nominative.

Essentially, lexical rules handle the changes in the suffixes, enforce type constraints, and control the mapping of subcategorization frame. Therefore, this study divides the lexical rules into three groups: inflectional morphology (person, number, and case), grammatical function changing affixes (voice affixes) and derivational morphology as category-changing operations. The lexicon design is tested as a part of Head-Driven Phrase Structure Grammar (HPSG) of Turkish.

The remainder of this paper is organized as follows. In Section 2 morphology and grammar relations are summarized as morphology and syntax, and morphology syntax interface. Section 3 represents the difference between transformational grammars and the two-level morphology without transformational grammar, and gives examples of some two-level morphological studies. In Section 4, we define the two-level rules for fundamental Turkish grammar rules, which constitute an alternative approach to the morphological analysis of Turkish verbs, and attend the Java codes corresponding to these rules. The ambiguities at the morphological analysis of Turkish have also been studied in this section. The implementation of Turkish verb analyzer is explained in Section 5. Finally, Section 6 concludes the paper.

2. THE ORGANIZATION OF MORPHOLOGY

There are different types of meanings. While lexical meaning can be expressed in stems or radicals, grammatical meaning tends to be expressed in affixes (Sapir, 1921). Since the studies on most languages concentrate on verbal inflection, the expression of inflectional categories, whether they are suffixes or prefixes and what order of the affixes occurs in, has always been an interest in linguistics. As a result, we cannot separate morphology and grammar interactions while studying a language in detail. These interactions can be defined differently according to morphology and syntax, morphology and lexical semantics, and morphology and pragmatics.

Pragmatics reflects the appropriate use of the language according to various goals at the phonology, morphology, syntax and semantics levels; thus linguistic structure is related to contextual phenomena (Verschueren, 1987). Related contextual phenomena include regional, cultural and functional differences of any variation. Word structure with clitics and different affix properties may indicate pragmatic information. Although inflected words have to conform to the requirements of the syntax, different inflectional categories including different affixes can express the same inflectional category, which may be determined by pragmatic factors. In this manner, inflectional processes in the words may also be identified as pragmatics. On the other hand, derivational processes are affected by syntactic structure, but they don't have a direct relation to pragmatics. Thus, pragmatic relations including derivational morphology can only be realized according to the syntactic structure of derived words (verbs or deverbal nouns). Moreover, a complex word, which is lexically analyzed, can carry pragmatic information as lexical pragmatics, and syntactic structure of a word including case and plural markings may cause to syntactic pragmatics.

2.1. Morphology and Syntax

Syntactic expressions of the different semantic elements are expressed as separate and independent words. While morphological, phonological and syntactic modules are defined autonomously during the 1970s and 1980s; other works done in syntax showed that the syntactic systems could handle the morphology in a more restricted way. Thus, it is difficult to decide whether word formation has to be built as an independent module or not. If word formation is an independent module with its own restrictions, it will be difficult to characterize the interaction between this component and syntax. The researchers using independent word formation component have to show that their form includes operations and constraints. Besides, these operations and constraints must not be reduced to independent syntactic conditions. On the other hand, the researchers using syntax, in which syntactic and morphological structures can be derived from semantic representations, must show the word formation without using syntactic processes.

Pre-syntactic and lexically derived models are the examples of word formation components. Their words are defined atomically at the phrasal syntax and semantic levels, including unstructured features. These features cannot be related to the word in syntax (Sciullo and Williams, 1987). Since word formation component is ordered with respect to syntax, the output of word formation is the input to syntax, constituting the interaction only in one fixed point. This component generally includes all the properties of the formal operations characterizing both derivational and inflectional morphology. Orderings can be different according to word formation component. While one possible ordering is prior to any syntactic operation (D-structure), another one can separate the lexical and word formation components as the syntax precedes the morphophonological component. A pre-syntactic independent word formation component example has been illustrated in Figure1. The phonological string *koştu* (s/he ran) is generated applying *[+past]* feature to the root of the word. Thus,

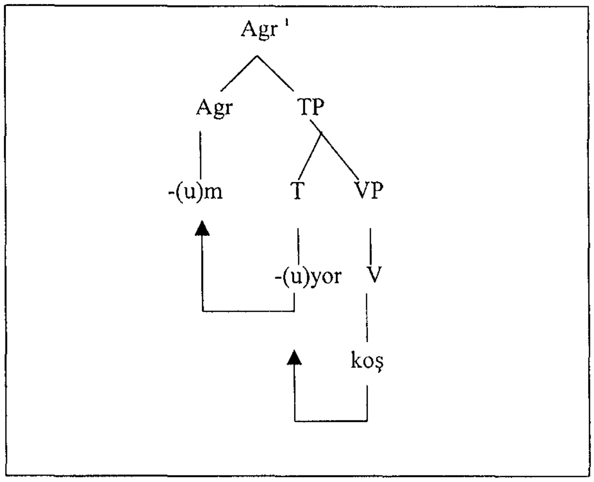


Figure 1. Atomicity Thesis: The Verb V, Moving from Its Original Position to the Functional Heads, Becomes Inflected for Tense and Agreement, Representing the Lexical Approcah to Syntax.

the word becomes syntactically visible by the lexicon at D-structure. To obtain the same property for more than one verb, *[V+past]* feature has to be defined by the word formation component at the morphophonological level. While *[KOŞ+past]* results phonological representation *koştu*, *[GÖR+past]* follows phonological representation *gördü* (s/he saw). In other words, past tense of any word must be associated with the entire word rather than any internal segment of it. Thereby, it becomes morphologically opaque (Borer, 1998).

Since component structure in morphology is similar to the generation process, complex words, especially derived and compound words, have a hierarchical component structure, which represents the derivational properties by tree-diagrams. However, word formation process does not reflect the internal operations. The agglutinative property of Turkish may cause some problems with the hierarchical component structure of the complex words. For example, Turkish word okutturulmamalıymış (he ought not to be made to educate) can hierarchically be analyzed as in Figure 2-a instead of the flat structure in Figure 2-b and 2-c.

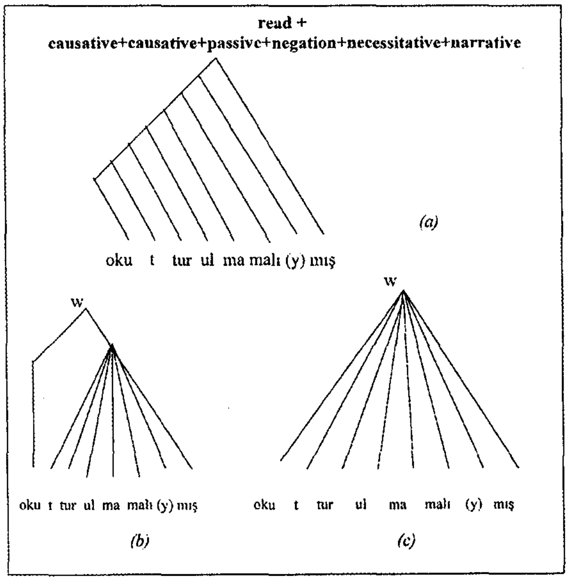

read +
causative+causative+passive+negation+necessitative+narrative
(a)
oku t tur ul ma malı (y) mış
w
w
oku t tur ul ma malı (y) mış
(b)
oku t tur ul ma malı (y) mış
(c)

Figure 2. Three Different Constituent Structures in Morphology. (a) is More Appropriate for Agglutinating Morphologies Than Other Representations (b) and (c).

2.2. Syntactic Models

Syntactic models, which design word formation and syntactic constraints together, are opposite to the independent word formation model and to the limited interaction of word formation component with syntax. These models derive internal word structure syntactically. For example, morphological derivations directly reflect the syntactic derivations (and vice versa) as the Mirror Principle (Baker, 1985). In general, inflectional and derivational morphology is constituted to reduce the morphological representations and syntactic configurations. According to head-to-head movement, the order of morphemes in a derived form has to reflect the syntactic structure. The representation in *Figure 1* is not a complete representation of syntax. However, morphophonological strings change to the morphophonological word with head-to-head movement. Since the order of morphemes must reflect the syntactic structure, the morpheme *-(i)yor*, which corresponds to the future tense in Turkish, is closer to the stem than the morpheme *-(u)m* representing the first person. Figure 3 shows that the tense marker as a syntactic feature remains at the lower level of the tree than the agreement marker. On the other hand, some languages such as Arabic display the opposite order of affixes, and their tense markings are outside argument markings. Therefore, it is necessary to parameterize every inflectional part of morphophonology. Such systems are called as language-specific, or affix- specific, and the ordering may differ from one language to the other (Laka, 1990).

2.3. Morphology - Syntax Interface

If two or more words, which are morphologically associated, are contrast in their lexical semantics, it will be possible to distinguish the lexical meaning of words with the morphology-syntax interface. Following Turkish sentences give an example for different lexical meaning of predicates:

(i) *Çocuklar bu tür değerli şeyleri kolayca kırar*
(Children easily break such valuables).

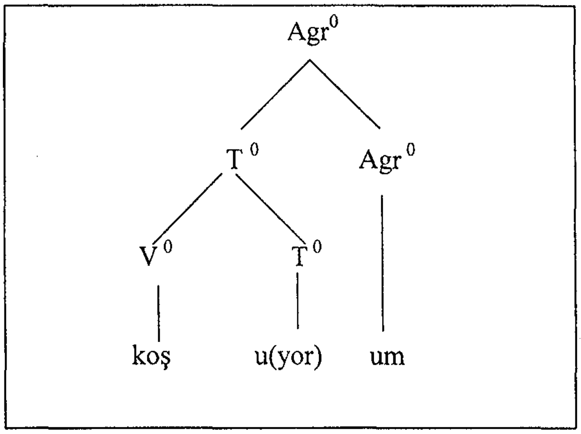


Figure 3. Syntactic Model: Morphemes are Added to the Verb with Head-To-Head Movement Representing the Syntactic Structure of a Morphophonological Word.

(ii) *Bu tür değerli şeyler çocuklar tarafından kolayca kırılır* (Such valuables are easily broken by children).

(iii) *Bu tür değerli şeyler kolayca kırılır* (Such valuables break easily).

The expressions as passive alternation construct morphosyntactic operations (Sadler and Spencer, 1998). Basic semantics of the sentences does not change with the operations. For example, the sentences *şişe rafta duruyor* (the bottle is on the shelf) and *rafta şişe duruyor* (a battle is on the shelf) explain the same thing with different word orderings, because the ordering is controlled pragmatically. This example also constitutes a morphosyntactic operation. Other voice affixes (reflexives, reciprocals and causatives) may conditionally define morphosyntactic operations for certain languages. The semantic effects of these operations are generally different from language to language. For example, the sentence çocuk *defter-i müdür-e imzala-t-tı* (the boy made the director sign the notebook) indicates the working of causatives in Turkish. The word notebook (*defter*) is the object of the causative verb, and the subject of the basic verb director (*müdür*), appears as a dative case marking. Lexical semantics has really an important effect in determining morphosyntactic structures as different uses of verbs. The words *süslenmek* [something is decorated and someone is dolled up] and *yıkanmak* [something is washed and someone is bathed]) are two samples which use passive voice to the active (two uses of the verbs).[4]

The verb of a sentence has some syntactic dependency as obligatory or conditional, and the grammar of any sentence must also include information about these valence requirements. This information may differently be expressed using various models. For example;

- The properties of a word or a sentence can be characterized by associating the string with lexical and syntactic schemata using Lexical Functional Grammar (LFG) formalism (Kaplan and Bresnan, 1982). LFG Grammar-Writers Workbench is also a computational application of this formalism (Kaplan and Maxell, 1996). This system, which has been implemented in the Medley Lisp programming environment, provides various facilities such as writing syntactic rules, lexical entries, and simple morphological rules, and the system interface both defines and manipulates the linguistic rules. Thus, information of this formalism is expressed by using grammatical functions directly such as subject and object.

- Principles and Parameters Theory (P&P) is interested in the syntactic representation of the used grammar, and also draws an explicit link between competence and acquisition (Cook and Newson, 1996). Thereby, information used with this theory has been expressed in terms of syntactic configurations. The structure of the grammar reflects the requirements of acquisition, and constitutes the fundamental design of P&P theory. The definition of Universal Grammar (UG), which specifies the possible values of variations, explains theory of knowledge as a system of principles, conditions and rules. Humans are born with a set of principles that can be applied to all languages, and this knowledge cannot be separated from the problem of how it is acquired (Epstein and Hornstein, 1999). It is assumed that the system in human brain enables meaning comprehension, and the Cognitive System (CS) deals with the sentence formation; information about the system is called Lexical Resources (LR). The CS combines the LR to derive a sentence (Williams and Kalita, 2000). Even though all the languages start with the same initial configuration for the word orders, they have different surface order characteristics in the spoken sentences. Formal features including categorical (nominal and verbal), ϕ (person, number, and gender), and case (nominative, accusative, and genitive) features for lexical items are used to check two items. Derivation of a sentence in the Minimalist Program starts with a set of lexical items. Many checks that test the possible scenarios are performed during the derivation processes until two items match for one or more features, and the derivation converging with the least effort results as grammatical.

- Head-Driven Phrase Structure Grammar (HPSG) expresses information as the combination of grammati-

[4] The effect of the subject turns into itself for the verb that can behave both active and passive, and these are called reflexive (Gencan, 1992).

cal functions and category labels (Pollard and Sag 1994). Instead of transformational derivations as the sequential manipulation of complete sentential structures, HPSG is formulated in terms of order-independent constraints. Phrase structure concept used in this theory is built around the vision of a lexical-head as a single word. This word is a dictionary entry, and specifies information to determine the grammatical properties of the phrase. This structure includes part-of-speech information. In other words, nouns can plan noun phrases, and verbs can plan sentences. Semantic information can also be displayed by sharing the phrasal aspects.

ALE, which was designed to provide an environment for running HPSG and LFG, is an integrated phrase structure parsing system. Type feature structures of the system are the generalization of the common feature structure system. Both grammars and definite clauses are mapped into abstract machine rules, which are interpreted by an emulator. ALE as a definite clause logic programming system is implemented in Prolog. While ALE system provides lexical rules to express the general definitions in lexicon, ALE-RA system is an extension of this defining word formation rules.

Table 1. The Difference between the Generation Rule and Two-Level Phonological Description Involving Raising and Palatalization Example.

An example from phonology defined with "Generative Rules" [a]	An example defined with "Two-Level Rules" for the morphological analysis [b]
Vowel Raising e→i / ___C_0 i (1)	Vowel Raising e⇔i ___C : C* @ : i
Palatalization t→c / ___i (2)	Palatalization t : c ⇔ ___ @ : i
After the application of rules in order UR: t e m i (1) t i m i (2) c i m i SR: c i m i	After the application of rules in parallel: UR: t e m i | | | | 1 2 | | | | | | SR: c i m i

[a] C_0 represents zero or more consonants, UR means Underlying Representation, SR means Surface Representation.

[b] @ is a symbol representing any phonological segment

3. TWO-LEVEL MORPHOLOGY

Any two-level model integrates the components of the transformational approach as a metagrammar and a grammatical form, which obtain all information from the lexicon (Krulee, 1991). Although there is no consensus about the representation of morphological and syntactic structures of complex natural language systems, two-level morphology approach can develop psychologically more realistic, computationally learnable, and effectively parsed models. Two-level representation of grammars came into use by the early 1970's as an addition to transformational grammars. In 1983, Kimmo Koskenniemi as his dissertation developed a declarative system to describe the grammatical morphology, which is called two-level system. This system is different from rewrite rules, which was firstly introduced by Chomsky in 1960s. Table 1 shows a two-level phonological example for these two different derivation rules. We can summarize the differences between Chomsky's grammar structure (i.e. generative rules) and Koskenniemi's two-level rules system as follows (Koskenniemi, 1997):

According to Chomsky's transformational generative grammar structure,

(i) Rewrite rules can only be applied from lexical representation to surface representation as a → b / c_ , changing one symbol into another symbol in the environment following *c*. This relation between the two symbols is interpreted as: *a* is rewritten or turned to symbol *b*. Any lexical symbol is written as a symbol in the surface representation, so this lexical symbol cannot be used for any other rules.

(ii) Although generative rules are also known as process rules like two-level rules, they are applied sequentially. They create a new intermediate level as an output of each rule. This intermediate level is an input to the next rule. Thus, subsequent rules cannot access to the first lexical symbol. In other words, generative rules are ordered, and the application sequences of the rules are important. Incorrect result may be formed for any different order of the rules.

(iii) At each step, generative rules can only access to the current intermediate form of the derivational process, and the rules can only operate from a lexical to surface representation because of the unidirectional property.

(iv) This presentation of rules is dynamic and the programming structure is procedural.

According to Koskenniemi's proposal,

(i) The rules are not ordered, but applied simultaneously (in parallel); so a lexical symbol corresponds to a surface symbol; in other words, the first symbol does not change into the second symbol. We can symbolize this definition as a: b ⇔ c - : -d; where c- : -d expression indicates the constraint as the underlying and surface environment, respectively.

(ii) We can compile each rule with a finite–state transducer, which directly establishes relations with each other.

(iii) Two-level rules do not perform operations on segments, the rule representation is static during the correspondence between underlying and surface forms, and the programming structure is declarative.

(iv) Since the rules are bi-directional, we can obtain both the generation and recognition of the words by using two-level rules.

On the other hand, left-to-right decomposition model takes the words at the phrase level, phrases at the sentence level, morphemes at the word level, and letters at the morpheme level; morphemes are also targets for the recognition (Hudson and Buijs, 1991). This model is contrasted with parallel processing models. Therefore, the elements at the lower levels are configured to determine the elements at the next level. Syntactic and lexical constraints as morphemes are similar at the underlying side. The value counting the branches of recursive transition network determines the alternative transitions, and helps to determine the stem. When the stem has been recognized, it is required to distinguish the correct suffixes. Consequently, morphologically complex words will be processed more rapidly than simple words with the same length.

3.1. Two- Level Modeling Applications

We can not only study the word structure, but also the sound structure in a language by using two-level rules. Koskenniemi explained the morphological structure of the words using two-level phonological rules. KIMMO parser, developed by Karttunen, (1983) was the LISP implementation of Koskenniemi's two-level model. The system has two analytical components as rules and lexical. While rules component consists of two-level rules for phonological and orthographic alternations, lexicon component lists all affixes and stems in the lexical form. These two components including data are used for the generation and recognition processes. PC-KIMMO system, implemented in C, was developed as an example of two-level phonology, and very similar to KIMMO system (Antworth, 1995). This system consists of two parts named phonotactic and morphotactic structures of word-forms. During the process, two-finite automata work in parallel, and a single finite automaton represents morphotactic rules. But the system can only tokenize the input word into its morphemes, and it cannot determine the main property of the word such as plural or singular. PC-KIMMO Version 2 was developed in 1993 by adding a word grammar, an unification-based chart parser, as a third analytical component of the previous system. The word parser tokenizes the word into morphemes producing a parse tree. This property of the new system looks like to a morphological parser that also tokenizes a word into morphemes, and then parses the morphemes by using a unification-based grammar (Ritchie et al., 1992). However, the implementation ways of two parsers are different. The grammar used by the word grammar component in PC-KIMMO Version 2 consists of context-free rules and feature constraints. There are three parts of a grammar: the first contains feature abbreviations, the second contains category templates, and the third contains word grammar rules. Each rule is associated with feature constraints. This system can be used with Englex lexicon, which defines a two-level description of English morphology.

In 1996, the Xerox Research Center Europe produced a large morphological analyzer for Modern Standard Arabic words (Beesley, 1996). Although the dictionary of analyzer-generator is based on ALPNET Arabic system, which was also developed by Beesley, he redesigned the system using Xerox finite-state technology.[5] In the Xerox Arabic systems, lexicons are written in the *lexc* language, which is the specific language of the Xerox finite-state technology, and compiled into finite-state transducers. The lexical transducer, which all the components of the grammar are combined, is completely language-independent. Xerox finite-state technology has being used at Xerox Research Centers, Xerox business units, especially by new enterprise companies, and at the universities by research groups with non-commercial licenses. Finite-state morphological analyzers can examine 23 different languages,[6] and Oflazer is the developer of Turkish analyzer, which has been performed by using the two-level transducer software.

An another study developed in Turkish automatically converts the given two-level phonological and morphotactic rules into Prolog program (Çiçekli and Temizsoy, 1997). In the proposed system phonological rules are mapped into new logical representations, and the order of morphemes are mapped into a finite state automaton. This two-level processor as a logic-programming environment for Turkish gives more efficient results according to time than the PC-KIMMO system. Although two systems were developed for the same purpose, authors explain the reason of this delay as the processing of a two-way transducer in PC-KIMMO system. However in logical representation, it is not required to map all phonological rules into transducers to find a correspondence.

[5] George A. Kiraz in his PhD thesis (1994) created a general survey of computational approaches to Arabic.

[6] English, French, Spanish, Portuguese, Italian, Dutch, German, Finnish, Hungarian, Turkish, Danish, Swedish, Norwegian, Czech, Polish, Russian, Japanese, Arabic, Basque, Irish, Korean, Malay, Aymara.

Karlsson (1990) defined Constraint Grammar (CG) to solve the morphological disambiguation of ambiguous word-form tokens, and to study surface-syntactic analysis of these tokens. In this way, morphological and syntactic analyzers have operated the rule-based descriptions. This framework has been implemented with two-level morphology as the language-independent formalism of parsing grammars. English Constraint Grammar Parser (ENGCG), which was primarily designed to analyze the varieties between British and American English, is based on CG.

Dictionaries that analyze large corpora can also be compiled into finite state automata as an application of automata theory to natural language processing. DICOF is an electronic dictionary containing about 700,000 words which are automatically generated from another dictionary DICOS. This dictionary contains simple forms of the words with about 80,000 entries, and these words are called canonical forms. The DICOF transducer generally relates the inflectional properties with the canonic form of the word. Therefore, the dictionary does not include the derived forms of simple verbs. DICOF or any other electronic dictionary cannot analyze this kind of word in a text. But Clemenceau merged the two-level system with the DICOF transducer (Clemenceau, 1992). He used a syntactic approach to define a tree-based representation of derivational operators. The morphological analyzer (MORPHO), which is a finite state transducer, merging the dictionary DICOF and a two-level system creates the morpho-syntactic information for the syntactic analysis of a sentence. Another example is the latest versions of the very large-scale dictionaries of LADL (Laboratoire d'Automatique Documentaire et Linguistique) (Mohri, 1996). The dictionaries including simple, compound or all inflected forms of different languages such as French, English and Italian have fully been implemented and tested by sequential transducers, which also provide reverse representation.

4. MORPHOLOGICAL ANALYSIS OF TURKISH VERBS

If we characterize our morphological analyzer in the simplest form as a black-box module, an input word in the text is accepted by the system, and an output will be the root position of the word, defining all of its affixes (Figure 4). This black box may be implemented in various ways. Since Turkish words include many complex relations with suffixes, we have to divide them into smaller parts to determine their meanings. In this study we analyze the verbs according to voice affixes, which constitute passive, reflexive, reciprocal and causative forms of the verb roots. These words also include simple or compound tense after the suffixes. Turkish language includes 9 different moods, but these moods are grouped into two basic parts as indicatives and subjunctives. A compound tense consists of the combination of any simple tense and substantive (predicative) verb. A substantive verb can also be an imperfect, narrative (dubitative) or conditional. Different analysis examples according to these suffixes have been described in detail in Table 2. We can check the verbal root of the word where simple and derived verbs have been searched from the database. Therefore, it is not possible to examine the other feasible structures of the root such as adjective,[7] noun or verbal stem that can include different meaning with the verbal root. For example, for the word *kalındı* (it was stayed), our system gives the analysis result as *kal(root)* + *(ın)reflexive* + *dı(past indefinite)* + *3rd singular*. We accept that this word as a verb is the last word of the sentence, and the sentence has been arranged in the SOV sequence. But for the determination of part-of-speech of the lexical items in the sentence, different morphological interpretations must be analyzed for each item. In that case, the root of the word *kalındı* may also be *kalın* (thick) as an adjective. The analysis result, *kalın(adjective)* + *dı(past definite)*, coincidences a substantive verb. When we add this kind of adjectives and nouns to the database, nominal words, which carry out verb function, can be seen in the analysis results. Moreover, morphological analysis of the word *alınmış* also includes adjective results (taken and offended), in addition to the part of speech as verb (it was taken and it was a forehead). This kind of morphological interpretation is not in our study scope.

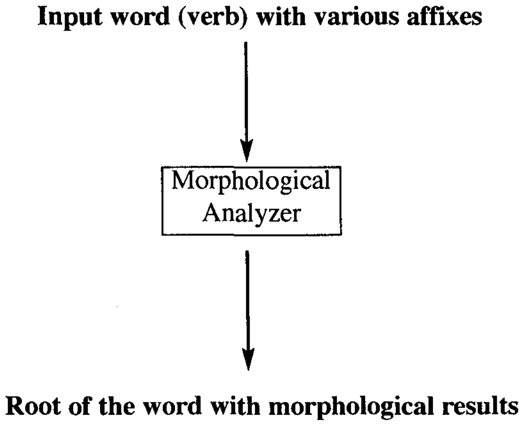

Input word (verb) with various affixes
Morphological Analyzer
Root of the word with morphological results

Figure 4. A Morphological Analyzer Represented as a Black Box.

Followings are some examples in which the meanings and representation of the verbs are different:

[7] In Turkish, adjectives and nouns are accepted that they have the same morphotactics.

Table 2. Recognition Results of Various Examples.

Input word	Magazinleştirilmeseydi [a]
Analysis result	magazinleş (*root*) + tir (*causative voice*) + il (*passive voice*) + me (*negation particle*) + se (desiterative) + y + di (*past definite*) +*3rd singular*
Input word	Koşuşmamalıysanız
Analysis result	koş (*root*) + uş (*reciprocal*) + ma (*negation particle*) + malı (*necessitative*)+ y + sa (*conditional*) + *2nd plural*
Input word	Takıştırıyordu
Analysis result	tak (*root*) + ış (*reflexive*) + tır (*causative voice*) + (ı)yor (*present*) +du (*imperfect*) + *3rd singular*
Input word	Görüştürülecekmiş
Analysis result	gör (*root*) +üş (*reflexive*) + tür (*causative voice*) + ül (*passive voice*) + ecek (*future*) + miş (*narrative*) + *3rd singular*
Input word	Aksattırılmayacaksa
Analysis result	aksa (*root*) + t (*causative voice*) + tır (*causative voice*) + ıl (*passive voice*) + ma (*negation particle*) +y + acak (*future*) + sa (*conditional*) + *3rd singular*
Input word	Oynattıralım
Analysis result	oyna (*root*)+ t (*causative voice*) + tır (*causative voice*)+a (*optative*)+ *1st plural*

[a] Since +*leş* is a suffix which derives verb, the word *magazinleş* is written to the database. Therefore morphological analysis starts after the root word in the database.

(i) *Çocuk yıkandı* (the boy is washed by his mother /he washed himself) sentence describes that the verb can both be passive voice and reflexive.

(ii) *Müdüre göründü* (he showed himself to the director) sentence describes that the verb is reflexive. The passive voice of this word is görüldü (to be seen). There are a few verbs of which their reflexive affix -*n* differs to –*l* for the passive voice. The examples for these words are: *sev(**in**)mek* (to feel happy) _ *sev(**il**)mek* (to be loved) , *döv(**ün**)mek* (to beat oneself) _ *döv(**ül**)mek* (to be beaten) and *tut(**un**)mak* (to grab hold of) _ *tut(**ul**)mak* (to be held).

(iii) The sentences *askerler yavaşça süzüldü* (the soldiers are glided) and *yoğurt süzüldü* (yogurt is filtered) include same verbs with different voices; therefore the analysis result has to represent these two probabilities according to the subject. Therefore, we get two answers for only third singular person as in this example.

(iv) In spite of the verb *süzüldü* above, *adam yoruldu* (the man got tired) and *adam üzüldü* (the man was worried) sentences describe that their verbs are reflexive. But, these verbs have no meaning as passive voice.

4.1.Two-Level Rules in Turkish

In our study, we firstly began to design the tool with the generation process of voice affixes; but our goal was to analyze the verbs in a text. As the suffixes increased, the originality for the generation of the verbs has decreased, and we excluded this construction. But it is possible to execute the generation process again.

The rules defining the generation and recognition process are bi-directional; therefore we can obtain a representation in one level from the representation in other level as given in Figure 5. A two-level rule can be defined as:

Lexical Form: Surface Form ⇔Left Constraint_:_Right Constraint,[8]

where left and right constraints are called as environment. While the left constraint has to be satisfied before the correspondence, right constraint is defined after the correspondence. Since the descriptions in morphology, as in phonology, make use an operation that replaces some symbol or sequence of symbols by other symbol or sequence, the replacement operator : is utilized to model the context of finite state grammars. Some of the rules, such as *a* : *a* or *Se* : *Se* (vowel letters set), has

[8] ⇔ is biconditional rule operator. Different representations of the rule operators are used to symbolize the two-level rules.

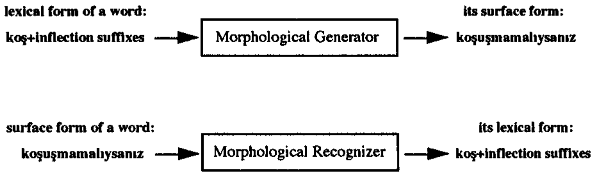


Figure 5. The Basic Structure of Morphological Generation and Recognition Process.
Inflection suffixes = reciprocal + negation + necessitative + conditional + 2nd plural.

only a correspondence between a lexical and surface letter and letter set. In our study, finite-state formalism has been defined as letter-by-letter replacement according to constraints in both sides. Regular expression operators explaining the finite-state network diagrams are defined in Table 3. For example,

$$^{+}SSz_y : {}^{+}YSz_y \Leftrightarrow {}^{?}Se_k^{n-1} \backslash\backslash {}^{?}Se_i^{n-1} : \_ \; {}^{?}Se_k \backslash\backslash {}^{?}Se_i$$

rule presents the softening of consonants as the letters correspond from (p ç t k) to (b c g/ğ d) respectively, and the constraints explain that if the last vowel before the correspondence is any of the back vowel or front vowel letter, the right constraint must respectively be any of the back vowel or front vowel letter from the related set.

For the analysis of the aorist, if the last letter of the verb is a vowel, the affix will only be the letter *r*. If the last letter is a consonant, following rule representation will aid the implementation. The symbol definitions can be seen in Table 4.

$$\phi : (Se_{d1} / e^{re}) \backslash\backslash (Se_{d2} / a^{re}) \Leftrightarrow Se_i^{n-1} \backslash\backslash Se_k^{n-1} : \_ r;$$

the correspondence between the letter sets will be according to the parenthesis, i.e. if last vowel of the word (verb root) is the letter *e*, the letter before *r* may either be *i* or *e* (*gel-ir* (he) comes or *gez-er* (he) strolls around); if the last vowel is the letter *ü*, then the letter before *r* will be the letter *ü* or *e* (*düşün-ür* he thinks or *düş-er* he falls). For the back vowel, the examples may be *kalır* (he stays) or *kazar* (he digs), and *otur-ur* (he sits) or *kur-ar* (he sets up).

Table 3. The Definition of Replace Operators Used for Finite State Network.

?	prefix represents that letters in the letter set can be active disorderly according to the related parentheses
+	prefix represents that symbol pairs (such as a:b) corres ponding to letter sets will be active sequentially
\\	operator represents the corresponding letter sets or active letters according to the rule operator and (or) constraint
/	operator represents that one of the letters or letter set will be active
∪	operator represents that one of the conditions will be active;
re_+	prefix represents the repetition position for the adjacent letter.
re	prefix represents the repeated letters

Table 4. The Definition of Letter Sets Used to Represent Turkish Vowel and Consonant Letters.

Se_d narrow vowels set	$(Se_{d1}\ Se_{d2}) = [(i\ ü)(ı\ u)]$
Se vowels set	[(e i) (ö ü)][(a ı) (o u)]
Se_i frront vowels set	[(e i) (ö ü)]
Se_k back vowels set	[(a ı) (o u)]
SSz_y unvocied strong consonants	(p ç t k)
YSz_y unvocied soft consonants	(b c d g/ğ)

This rule is implemented by the following bold marked Java codes.

```
public void geniscekimle(){
     incekalin(sonsesli);
     if (seslimi){
     geniskok=giris+"r";
     zamancekimli=geniskok;
     }
     else {
        geniskok=giris;
        if (sonsesli= ='e'){ genisek1="i"; genisek2="e";}
        if (sonsesli= ='i'){ genisek1="i"; genisek2="e";}
        if (sonsesli= ='O'){ genisek1="ü"; genisek2="e";}
        if (sonsesli= ='ü'){ genisek1="ü"; genisek2="e";}
        if (sonsesli= ='a'){ genisek1="I"; genisek2="a";}
        if (sonsesli= ='I'){ genisek1="I"; genisek2="a";}
        if (sonsesli= ='o'){ genisek1="u"; genisek2="a";}
        if (sonsesli= ='u'){ genisek1="u"; genisek2="a";}
        genissessiz=true;
           //System.out.println(ince);
        if(ince= =true && eylem[len-1]= ='t')
      geniskok=giris.substring(0,len-1);
        if(sertmi==true && ince==true &&
      eylem[len-1] = ='t') geniskok=geniskok+"d";
          genis1= geniskok+genisek1+"r";
          genis2= geniskok+genisek2+"r";
          //zamancekimli=genis1+"-"+genis2;
          zamancekimli=genis2;

       //fieldControl10.setText(geniskok);
     }
```

We can give another example for the past definite tense with the following rules:

$$\phi : d \backslash\backslash t \Leftrightarrow YSz_y^n \backslash\backslash SSz_y^n \; : \; \_\, Se_d$$

$$\phi : Se_d \Leftrightarrow (Se_i^n \,/\, Se_i^{n-1}) \,//\, (Se_k^n \,/\, Se_k^{n-1}).$$

Past definite of the verbs *git* (go), *yap*(do), *koş* (run) are *gitti* (went), *yaptı*(did) , *koştu* (ran). Since the last letters of the root words are strong consonant, the correspondence will be the letter *t.* But for the roots *gör*(see), *gel* (come) the last letters are soft consonants, so the past definite of these verbs are *gördü* (saw), *geldi* (came). Following is the Java implementation of the second rule and the first rule respectively, which are defined for the past definite tense above.

```
    public void benzesenunlu(char sonses) {
    if (sonses= ='e' || sonsesli= ='i')  benharf="i";
  if (sonses= ='O' || sonsesli= ='ü')  benharf="ü";
    if (sonses= ='a' || sonsesli= ='I')  benharf="I";
  if (sonses= =' o' || sonsesli= ='u')  benharf="u";
          }
  public void dicekimle(){
   benzesen1="d";
     if(sertmi) benzesen1="t";
      zamancekimli=giris+benzesen1+benharf;
     //fieldControl10.setText(cekimli);
  }
```

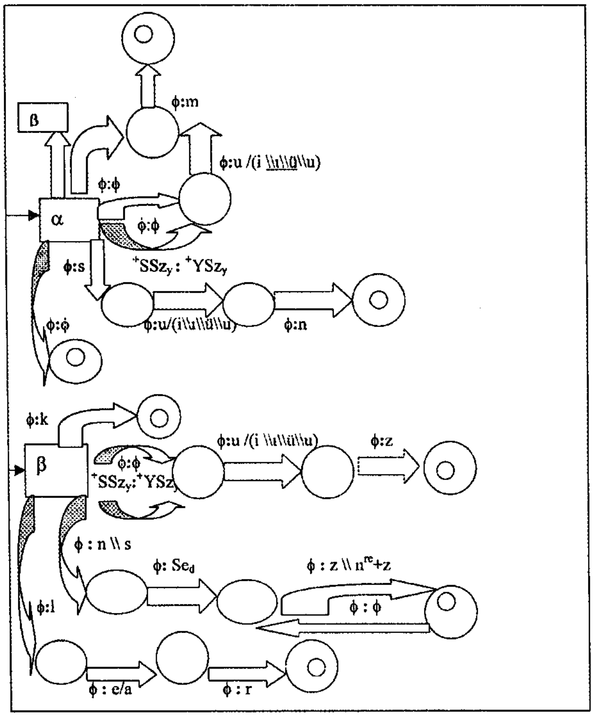


Figure 6. The Representation of Personal Affixes of Turkish Verbs with a Finite State Network Diagram (the Replace Operators are Defined in Table 3 and the Meaning of Symbols as Letter Sets are Given in Table 4).

Turkish letter sets, which we use in our analysis, are classified according to their properties in Table 4. The network diagram in Figure 6 and the constraints in Table 5 display the two-level representation of the personal affixes for Turkish verbs, it is accepted that the rule sets of other mood affixes have been defined by the previous networks.

Following Java implementation only represents the relationships for the first singular person, which has been modeled in Figure 6. Firstly, *ϕ: Se$_d$* rule for aorist, *ϕ: m* rule for past definite are coded. Softening of consonants is then analyzed for the first person of future. Lastly, the rules for remainder tenses *ϕ: u* for present, and *ϕ: i/ı* for past indefinite are implemented. We may not follow easily each step of these rules because of the functions defined formerly.

```
public void sahiscekimle(){
   String cekimler="";
    if(sahis1tek){
    if(genis3){
      if(genissessiz){
      analyze(genis1);
      String tekil1=genis1+benharf+"m";
      analyze(genis2);
      String tekil2=genis2+benharf+"m";
      sahiscekimli=tekil1+"-"+tekil2;
        } else sahiscekimli=zamancekimli+benharf+"m";
          } else{
            analyze(zamancekimli);
             if(seslimi){
```

Table 5. The Rules and Corresponding Constraint to These Rules for Personal Affixes in Figure 6.

$^+SSz_y : {}^+YSz_y \Leftrightarrow {}^?Se_k^{n-1} \backslash\backslash \; {}^?Se_i^{n-1} : \_ \; {}^?Se_k \backslash\backslash {}^?Se_i$
$\phi : u \Leftrightarrow o^{n-1} : \phi \cup \_ (m/z) \cup s \_ n \cup s \_ n^{re}+z$ (right side includes all the constraints for related personal affixes of the present continuous)
ϕ: i\\ü\\ı\\u $\Leftrightarrow (e^{n-1} / i^{n-1}) \backslash\backslash ü^{n-1} \backslash\backslash (ı^{n-1}/a^{n-1}) \backslash\backslash u^{n-1}: \phi \cup \_ (m/z)$ (right side includes all the constraints for first, third singular and plural personal affixes of the related rule)
$\phi : Sed \Leftrightarrow (Se_d^n : n \_ z) \cup (Se^{n-1} : s \_ n^{re} +z)$ (right sides include the constraints both the past tense and other tenses for second plural person) $\phi : e/a \Leftrightarrow (Se_d^n : l \_ r) \cup (Se^{n-1} : l \_ r)$ (right sides include the constraints both the past tense and other tenses for third plural person)

```
            sahiscekimli=zamancekimli+"m";
            } else {
              if(sertmi && gelecek3){
              String yumusayan="";
          zamancekimli=zamancekimli.substring(0, len-1);
          if(sonh.equals("k")) yumusayan="g";
    sahiscekimli=zamancekimli+yumusayan+benharf+"m";
    } else sahiscekimli=zamancekimli+benharf+"m";
    } } }
    .........

    .........
```

Another two examples, which represent the negation positions of the inflected verbs for aorist, and the indicatives including five time affixes have been modeled in Figure 7 and Figure 8. For the negation of aorist, the environments of two rules in Figure 7 are defined as:

ϕ:e\\a ⇔ (Se_i^{n-1}/ Se_i^{n}) \\ (Se_k^{n-1}/ Se_k^{n}) and ϕ:i\\ı ⇔ e\\a.

The first constraint characterizes that if the last vowel in the word is a front vowel, the letter *e* will be added as the correspondence to null; otherwise the letter *a* will be added. The second rule can be interpreted in a similar way.

5. IMPLEMENTATION

In this study as illustrated in the previous section, Java programming language has been preferred as the development platform of our application. Since Java is portable, robust, multithreaded and object-oriented programming language and development environment, this language has major advantages over traditional programming languages. Some of these are excellent reusability, automatic storage management and very low system requirements. In addition, distributed and network computing capabilities, interoperability features with other languages and formats (such as VRML, XML, JavaScript) are advanced properties. Nowadays Java is being driven the information technology markets as an operating system for hand holds, as a new network technology for intelligent networks, and as a platform for independent, fully functional programming tools.

Our tool, which can recognize the complex words, is a Java applet. This means that this tool can be run on almost any platform without requiring recompilation. Only requirement to run the tool is a Java capable web browser. JVMs (Java Virtual Machines) in browsers interpret java byte codes (.class files) as the native machine instructions just in time. Therefore, it is not required to store them anywhere. We use a database to recognize the root of any complex verb. First letters of the verbs organize the database tables. The connection with the database is accomplished via JDBC (Java Database Connectivity), which is an alternative method to access the databases. This type of connection may cause some performance disadvantages according to the traditional database accessing methods. But if you don't have a huge database, the result will generally be acceptable.

The main advantages of using Java in this study are object-oriented structure of Java, minimum system requirements and platform independence. Also other

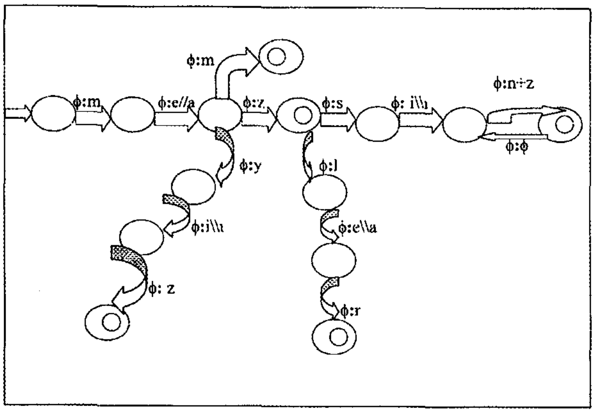


Figure 7. The Representation of Negation and Personal Affixes with Two-Level Morphological Rules for Aorist (gelmem, gelmezsin, gelmez, gelmeyiz, gelmezsiniz, gelmezler or sormam, sormazsın, sormaz, sormayız, sormazsınız, sormazlar. These are inflected forms of aorist negations with personal affixes for the verbs "come" and "ask".)

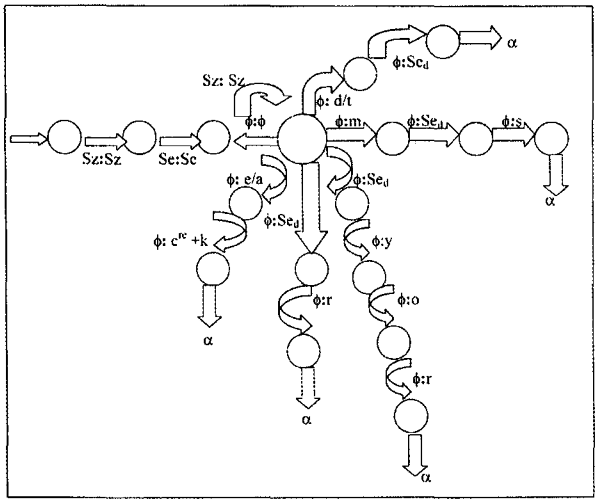


Figure 8. The Representation of the Indicatives of Turkish Words, Including One Syllable, with Two-Level Morhological Rules. Personal Affixes for Each Type of Indicative Can Be Followed from Figure 6. (koştu, koşmuş, koşuyor, koşar, koşacak or gördü, görmüş, görüyor, görür, görecek These are the indicatives of the verbs "run" and "see".)

mentioned features become important for the choice of this program. According to the algorithm, morphological analysis process starts from the end of the verb. First thing which has to be determined is the time clause. Although personal affix is located at the end of the verb, it depends on time clause. Therefore, the first process determines the time, and the second one determines the person. After the algorithm searches whether the subjunctive exists or not, the negation particle is searched. The last process before the determination of the root of the input word is to search the voice affixes. Since the word may include more than one voice affix, the determination process for the voice affixes can be called more than one time. But, all of the other functions are called only one time. Compound tenses, such as imperfect, narrative or conditional are also recognized. Since database is used to recognize the root verb of the word, the algorithm searches the root verb in the database three or four times during the program flow. When the appropriate word is found in the database, program flow is interrupted and the determined suffixes are displayed. It is possible that the verb root contains some syllables similar to voice affixes. Therefore, we have to set up the database carefully to prevent the confusion during the analysis between the root and stem of the word. If the analysis result is still wrong, this will cause of the error in the system.

The database includes all the basic and derived Turkish verbs, which were written from Turkish Language Society Dictionary. The total numbers of the words in the database are 2084. While minimum number is 12 for the verbs beginning with the letter c, the maximum number is 384 for the verbs beginning with the letter *k*. It is expected that the system will run correctly apart from the exceptions explained in Section 4.

6. CONCLUSION

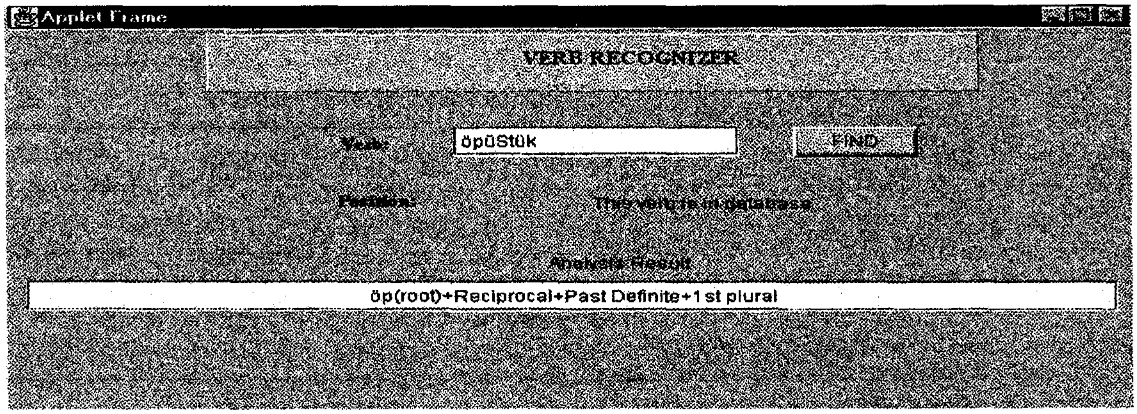
We can accept that morphology is the most influential subdiscipline of linguistics, because words constitute an interface among phonology, syntax, and semantics. Words articulate together to form phrases and sentences, which reflect their syntactic properties; these are phonological properties of the words. Moreover, words establish relationships with each other to form paradigms and lexical groups. Therefore, the interaction of the morphological structure with syntactic and semantic functions must be researched. Prefixed and suffixed words include different morphological characteristics. The languages including syntactic class structure and inflectional information emphasize the suffixes rather than the prefixes.

Prefixes are always derivational, so they may change the meaning of the words and can only be applied to certain syntactic categories. For example *–un* affix in English only precedes adjectives and verbs. Since prefixes have no syntactic affect for the word, they tend to be static part of the word; therefore the prefixes may not be analyzed. Suffixes, on the other hand, may be either derivational or inflectional, and they are both related to the syntactic structures. While inflections are restricted to the same category, the words including derivations have to fit the overall syntactic structure.

The application part of this paper presents a tool, which analyzes the verbs including various suffixes, especially voice affixes that have widely been used in Turkish verbs (Figure 9). The distinction between morphology and syntax in agglutinative languages is more difficult than relatively more isolated languages. For instance, Turkish includes significant amount of interaction between morphology and syntax as explained in Section 4. Causative suffixes may change the valence of the verb, and the reciprocal suffix may subcategorize the verb for a noun phrase. But we only analyze the verbal words. If we analyze three different classifications of the verbal words, i.e. infinitive, participle and gerund; it will be possible to perform syntactic composition of a subordinate clause according to this morphological composition. Therefore, inflectional suffixes reflect syntactic properties, and derivational suffixes reflect both lexical choices and syntactic properties. In fact, it is difficult to decide between the morphologic and syntactic preferences. But Turkish studies use more morphology than word order. Different languages may consider the opposite of this opinion.

In Section 2, we explained the importance of morphological structure of the words for the syntactic analysis. Then, finite-state morphology has been studied in detail as a language-independent model in Section 3. But our application represents a language-specific approach in order to easily integrate the tool to our other researches.[9] This application is not a comprehensive study as other studies, which we explained in Section 3. The reason is the assimilation facilities of the tool, which we developed in the Java environment to our other studies in similar environments. Therefore, the advantage of this Turkish specific approach is the using convenience for our studies, and the independency from platform according to other comprehensive examples.

[9] In these days, we are carrying on a study to develop a prototype by using supervised disambiguation method (Bayesian classification approach) about the word sense disambiguation of some Turkish verbs. In this study, we use two tools to obtain verbal and nominal morphological analysis results. The other tool analyzes Turkish nouns (Altan and Aydın, 2000). Thus, the number of occurrences of the ambiguous word and other related words in the training corpus could be calculated correctly and easily. Moreover, this tool will be used in another study, which we have already begun to study about the extraction of information from documents.



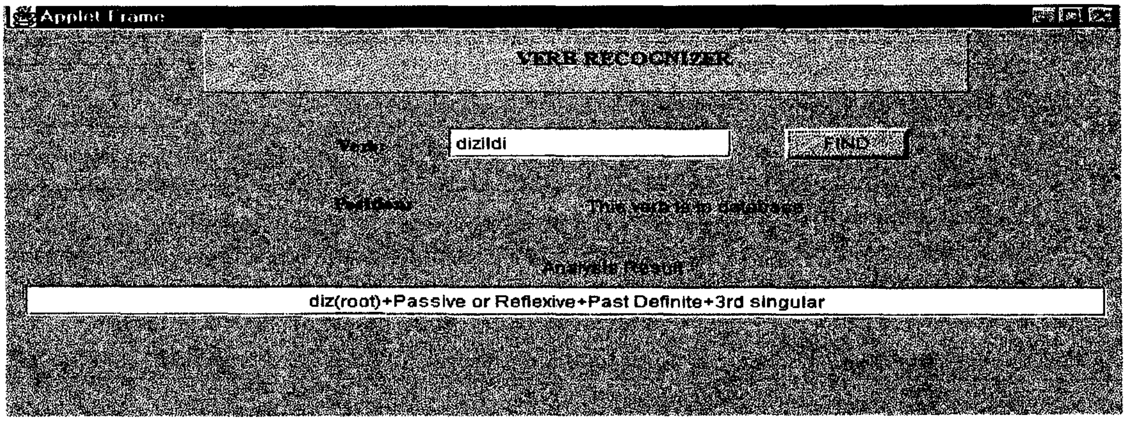
Verb: öpüStük

Position: This verb is in database

Analysis Result

öp(root)+Reciprocal+Past Definite+1 st plural

(a) We Kissed Each Other.



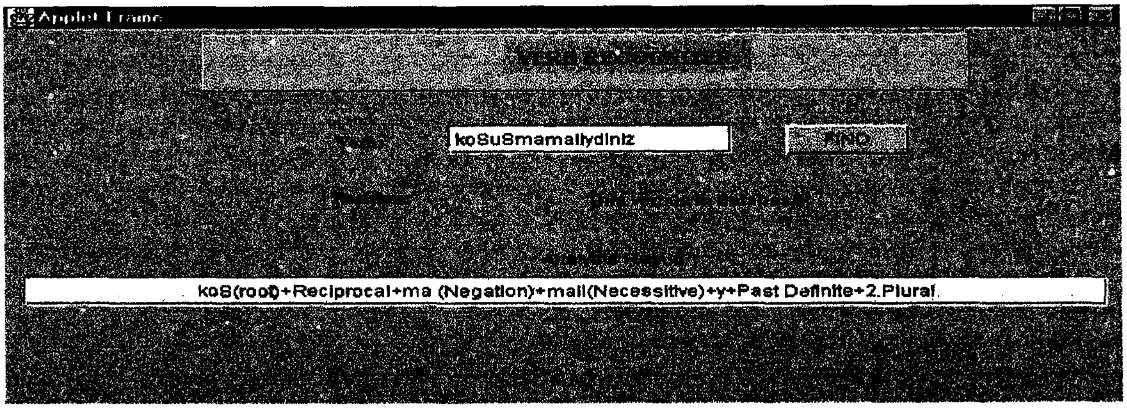
Verb: dizildi

Position: This verb is in database

Analysis Result

diz(root)+Passive or Reflexive+Past Definite+3rd singular

(b) The Beads Were Strung On A Rope Or The Soldiers Were Drawn Up Troops.

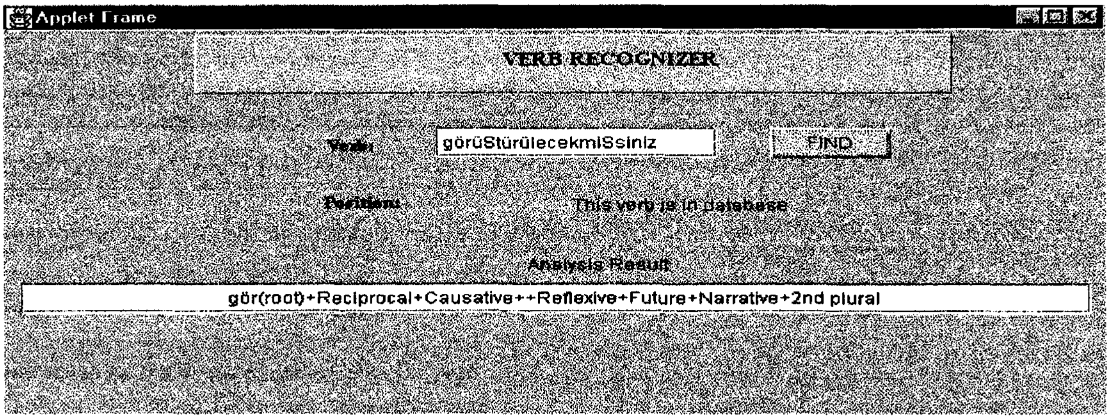


koSuSmamallydlnlz

koS(root)+Reciprocal+ma (Negation)+mall(Necessitive)+y+Past Definite+2.Plural.

(c) You Ought Not To Be Made To Run.

Figure 9. Various Screen Outputs (Continued on the next page).



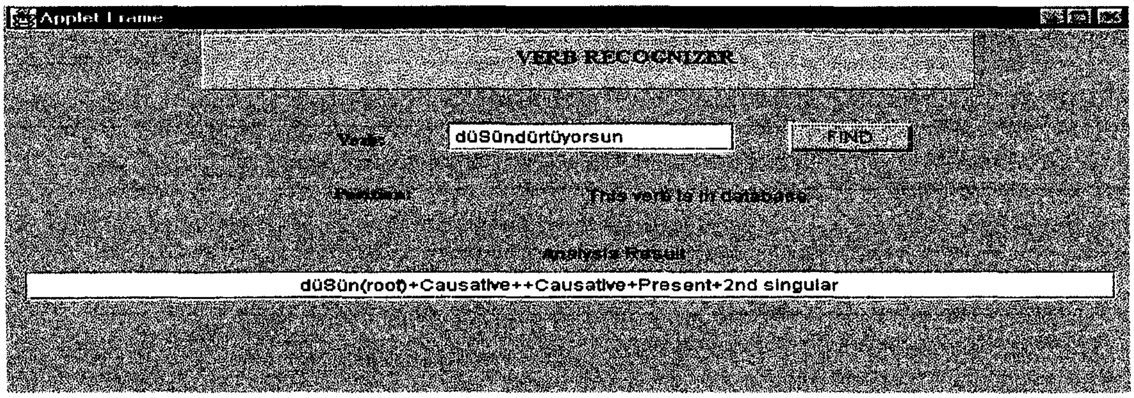
Verb: görüStürülecekmiSsiniz

Position: This verb is in database

Analysis Result

gör(root)+Reciprocal+Causative++Reflexive+Future+Narrative+2nd plural

(d) You Would Have Been Allowed To Meet Them.



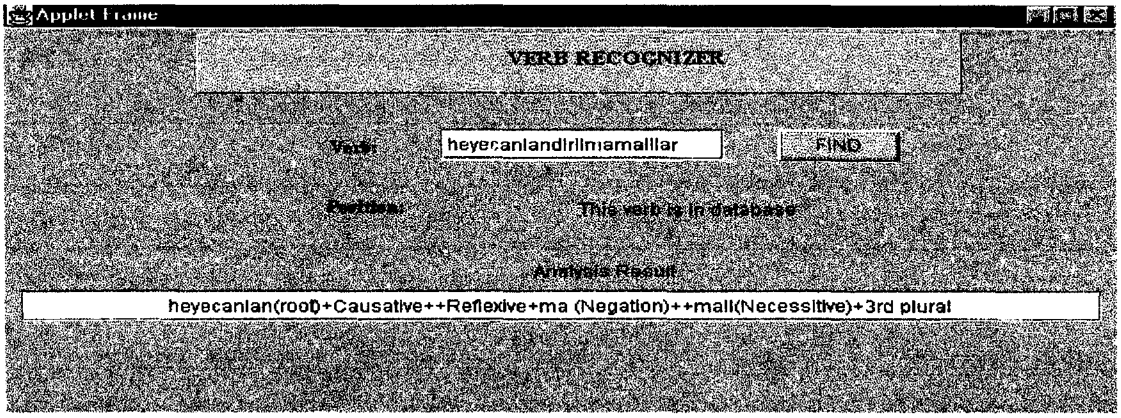
Verb: düSündürtüyorsun

Position: This verb is in database

Analysis Result

düSün(root)+Causative++Causative+Present+2nd singular

(e) You Are Having Me Have Thought.



Verb: heyecanlandirilmamalilar

Position: This verb is in database

Analysis Result

heyecanlan(root)+Causative++Reflexive+ma (Negation)++mali(Necessitive)+3rd plural

(f) They May Not To Be Made To Get Excited.

Figure 9. Various Screen Outputs (Continued).

The explanation for why we utilize from two-level morphological model is: the attributions of the rules are completely suitable to the processing of the programming logic. We displayed this approach in Section 4 by describing the two-level rules and corresponding code fragments. All application fields, which are related to the language processing technology, constitute independent research subjects. They are generally knowledge-based technologies, which the processing style of knowledge is unimportant. However, the query methods planned in the lower level can also be used for special fields. The linguistics and the fundamental natural language processing methods, including morphology, have to be considered with the details for the practicability of all data driven technologies.

REFERENCES

Altan, Z. and Aydın K. (2000). İsimlerde çekim eklerinin oluşturduğu ses olaylarının Visual-Basic ortamında incelenmesi. *Elektrik-Elektronik-Bilgisayar Mühendisliği Sempozyum Bildirileri,* Bursa, Turkey, pp. 307-312.

Aksu-Koç, A. (1988). *The acquisition of aspect of modality–the case of past reference in turkish.* Cambridge University Press.

Antworth, E.L. (1995). *User's Guide to PC-KIMMO Version2. http://www.sil.org/pckimmo/v2/doc/twolphon.html.*

Baker, M. (1985). The mirror principle and morphosyntactic explanation. *Linguistic Inquiry*, 16, 373-416.

Banguoğlu, T. (1998). *Türkçenin Grameri.* Türk Dil Kurumu Yayınları: 528.

Beesley, K.R. (1996). Arabic finite-state morphological analysis and generation. *Proceeding of the 16 th International Conference on Computational Linguistics*, University of Copenhagen, Denmark, V.1, pp.89-94.

Beesley, K.R. and Kartunnen, L. (1998). *DRAFT: Finite State Morphology: Xerox Tools and Techniquies.* Xerox Corporation.

Borer, H. (1998). Morphology and Syntax. *The Hand Book of Morphology,* Eds: A. Spencer and A.M. Zwicky (eds), pp. 151-190, Blackwell Publishers.

Bybee, J.L. (1985). *Morphology: A study of the Relation Between Meaning and Form.* John Benjamins Publishing Company.

Campione, M. and Walrath, K. (1998). *Java Tutorial Object-Oriented Programming for Internet.* Addison Wesley.

Clemenceau, D. (1992). Dictionary completeness and corpus analysis. *Proceedings of the 2nd International Conference on Computational Lexicography*, Budapest, Linguistics Institute, Hungarian Academy of Sciences, pp. 91-100.

Cook, J.V. and Newson, M. (1996). *Chomsky's Universal Grammar: An Introduction, 2nd Edition.* Blacwell Publisher Ltd.

Çicekli, İ. and Korkmaz, T. (1997). Generation of Turkish noun and verbal groups with systemic-functional grammar. *Proceedings of the 6th European Workshop on Natural Language Generation,* Duisburg, Germany, pp. 1-6.

Çiçekli, İ. and Temizsoy, M. (1997). Automatic creation of a morphological processor in logic programming environment. *Proceedings of the 5th International Conference on the Practical Application of Prolog*, London, UK, pp. 165-174.

Epstein, S.D. and Hornstein, N. (1999). *Working Minimalism.* The MIT Press.

Gencan, T.J. (1992). *Dilbilgisi.* Kanaat Yayınları.

Göçmen, E., Şehitoğlu O. and Bohşahin, C. (1995). *An Outline of Turkish Syntax.* Technical Report. *http://www.Lcsl.metu.edu.tr/pubs.html*

Heller, P., Roberts, S., Seymour, P. and McGinn, T. (1997). *Java 1.1 Developer's Handbook.* Sybex Inc. (Turkish language edition).

Hengirmen, M. (1999) *Dilbilgisi ve Dilbilim Terimleri Sözlüğü.* Engin Yayınevi.

Hudson, P.T.W. and Buijs, D. (1991). Left-to-right processing of derivational morphology. *Morphological Aspects of Language Processing*, Ed: L.B. Feldman.

Kaplan, R.M. and Bresnan, J. (1982). Lexical-functional grammar: A formal system for grammatical representation. *The Mental Representation of Grammatical Relations, Ed:* J. Bresnan pp. 173-281, The MIT Press.

Kaplan, R.M. and Maxwell, J.T. (1996). *LFG Grammar Writer's Workbench Version 3.1.* Xerox Corporation.

Karlsson, F. (1990). Constraint grammar as a framework for parsing running text. *Proceedings of the 13th International Conference on Computational Linguistics*, Ed: H. Karlgren, 3, 168-173, Helsinki.

Karttunen, L (1983). KIMMO: A general morphological processor. *Texas Linguistics Forum*, 22, 163-186.

Koskenniemi, K. (1997). Representations and finite –state components in natural language. In *Finite –State Language Processing,* Eds: E. Roche and Y. Schabes, pp. 99-116. MIT Press, London.

Krulee, G.K. (1991). *Computer Processing of Natural Language Processing*. Prentice Hall.

Laka, I. (1990). *Negation in Syntax: On the Nature of Functional Categories and Projections*. Ph.D.Dissertation, Massachussets Institute of Technology, Cambridge, MA, U.S.A.

Matheson, C. (1995). Computational Morphology: An Introduction to ALE-RA. *http://www.ltg.hcrc.ed.ac.uk/projects/ledtools/ale-ra/*

Mohri, M. (1994). *On some applications of finite-state automata theory to natural language processing*. Techinal Report, Institut Gaspard Monge.

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

Oflazer, K. and Bozşahin, H.C. (1994) Turkish natural language processing initiative: An overview. *Proceedings of the Third Turkish Symposium on Artificial Intelligence and Neural Networks*, Ankara, Türkiye, pp. 111-123.

Öztaner, S.M. (1996). *A Word Grammar of Turkish with Morphophonemic Rules*. M.S. Thesis. Department of Computer Engineering, Middle East Technical University, Ankara, Turkey.

Pembeci, İ. (1998). *A Unification-Based Tool for Learning of Turkish Morphology*. M.S. Thesis. Dept. of Computer Engineering, Middle East Technical University, Ankara, Turkey.

Pollard, C. and Sag, I.A. (1994). *Head–Driven Phrase Structure Grammar*. The University of Chicago Press and Stanford: CSLI Publications.

Ritchie, G.D., Graham, J.R., Alan, W.B., and Stephen G.P. (1992). *Computational Morphology: Practical mechanism for the English Lexicon*. The MIT Press, Cambridge.

Sadler, L. and Spencer, A. (1998). Morphology and Argument Structure. *The Handbook of Morphology*, Eds: A. Spencer and A.M.. Zwicky, pp. 206-236. Blackwell Pub.

Sapir, E. (1921). *Language*. Harcourt, Brace and World, New York.

Sciullo, A.D. and Williams, E. (1987). *On the Definition of Word*. The MIT Press.

Şehitoğlu, O. and Bozşahin, C. (1999). Breadth and depth of semantic lexicons. *Lexical Rules and Lexical Organization: Productivity in the Lexicon*. Ed: E. Viegasled. Kluwer Press.

Verschueren, J. (1987). Pragmatics as a Theory of Linguistics Adaptation. *IPrA Research Center Working Document 1*, Antwerp.

Williams, J.S. and Kalita, J.K. (2000). Parsing and interpretation in the minimalist paradigm. *Computational Intelligence*, 16(3), 378-407.

Zeynep Altan attended Istanbul Technical University both as an undergraduate and as a graduate student, receiving a B.A. as Mathematical Engineer in 1980 and a Master of Science Degree in System Analysist Section of Science Institute in 1983. She completed her Ph.D. Degree at the Istanbul University in Numerical Methods Section of Institute of Social Sciences in 1990. Until 1993, she was a research assistant at the Istanbul Technical University, Mathematical Engineering Department. Since then she is an assistant professor of Computer Engineering Department at Istanbul University, Faculty of Engineering. She teaches computer science courses, and she is a mother of a daughter and a son.