Morphosyntactic Generation of Turkish from Predicate-Argument Structure

Burcu Karagol-Ayan
Department of Computer Engineering
Middle East Technical University
06531 Ankara, Turkey
burcu@lcsl.metu.edu.tr

Abstract

In Turkish, which is an agglutinative language, it is difficult to divide morphology and syntax, therefore it is reasonable to treat them in the same way. In this paper, we present morphosyntactic generation of Turkish surface forms from a structured meaning representation, predicate-argument structure (PAS). The algorithm uses a categorial framework which integrates inflectional morphology, syntax, and semantics, and in which the basic building blocks are morphemes. The model is based on Combinatory Categorial Grammar (CCG).

1 Introduction

The amount of interaction between morphology and syntax is significant in agglutinative languages. Morphology, as well as syntax, indicates grammatical functions. Therefore, integrating morphology and syntax in one architecture is reasonable for such a language. In this way, the correct semantic bracketing of a phrase, when a morpheme has a phrasal scope, derives straightforwardly without rebracketing[1] which makes the morphology-syntax-semantics interface nontransparent. Bozşahin (1995) proposed a morphosyntactic categorial model which is not word-based, but morpheme-based. This model was used for deriving the predicate-argument structure (PAS) from Turkish surface forms (Bozşahin, 1998). The idea behind my study is doing the reverse process of this, that is generating Turkish surface forms from PAS using the same categorial framework. The term morphosyntax in this study refers to those aspects of morphology and syntax that collectively contribute to grammatical meaning composition. There is no distinction between phrase formation and word inflection.

2 Turkish Data

Turkish is an agglutinative language, that is, grammatical functions can be indicated by adding various inflectional morphemes[2] to words. For example, case-marking or relativization are realized in Turkish morphologically. Nouns and verbs can take several suffixes. The same suffixes that are applied to nouns are applied to proper nouns, pronouns and question words as well. Subject-verb agreement is marked by person and number morphemes on the verb. Some suffixes, such as the nominal case and third person singular, are phonologically null.

Whether or not a noun or a verb can take a specific inflectional morpheme is determined by a number of *morphotactic rules*. For instance, a Turkish noun cannot take a number morpheme if it is already case-marked.

Turkish is a pro-drop language. Since the subject-verb agreement morphology on verbs gives the information concerning the person and number of the sentence, the subject may be omitted, if it is a pronoun. Although Turkish verbs are not marked for object-agreement

[1] Rebracketing is changing the scope relations due to a mismatch, in our case, mismatch with semantics, e.g., PLU(green bus) vs. green(PLU bus)

[2] In Turkish affixes are generally in the form of suffixes, only a few foreign origin prefixes exist.

or with object clitics, objects can also be dropped in Turkish.

Although subject-object-verb (SOV) is the most commonly used word order of a simple transitive Turkish sentence, all six permutations are grammatical and are used under certain discourse situations. All these permutations have the same logical representation, however each one has a different discourse meaning. It is the context that determines the word order in Turkish. Although word order is relatively free in Turkish, there are still some syntactic restrictions on word order. For instance, relativization must be head-final, and embedded clauses and subordinate clauses are strictly verb-final. Turkish direct-objects are normally accusative case-marked. However they can also occur without any case marking. In that case, the OSV word order is not licensed with the non-referential objects.

In Turkish, the surface realization of morphological constructions are done by a number of *morphophonemic rules*. Vowels in the affixed morphemes, or the consonants in the root words or in the affixed morphemes undergo certain modifications, and may sometimes be deleted according to these rules.

3 Categorial Framework

3.1 CCG and PAS

The categorial framework used in this study is based on Combinatory Categorial Grammar (CCG) (Steedman, 1985; Steedman, 1987; Steedman, 1988; Steedman, 1996) which is an extension of Categorial Grammars. CGs are lexicalist formalisms, that is most of the information is put into the lexicon instead of having them in the grammar.

In CCG, grammatical categories are either functors or basic categories. The application rules (1) are used to combine two categories, whereas composition rules are used to combine together two functors, and type-raising turns an argument into a functor. The category $(S\backslash N)$ means that this expression looks for an N on its left to become S.

(1) a. Forward Application (>):

$$X{:}\,fx/Y{:}\,x \;\; Y{:}\,a \Rightarrow X{:}\,fa$$

b. Backward Application (<):

$$Y{:}\,a \;\; X{:}\,fx\backslash Y{:}\,x \Rightarrow X{:}\,fa$$

Categories also include semantic interpretations as shown above after colons. The semantic interpretations are represented as *predicate-argument structures* (PAS). PAS is formed from the syntactic derivation. Unification is used to merge the categories. The PAS reflects the order of the arguments between themselves (2) (Bozşahin, 1998). The first element in PAS is the predicate, and the followings are the arguments. The primary term is the last argument. This representation is not equal to the surface order of constituents. For example, the PAS of the sentence '*Poirot solved the mystery*' is (*solve mystery Poirot*), where the subject, *Poirot*, is the primary term. Word order is defined solely by the directional slashes in CCG.

(2) $Predicate \ldots < Secondary\ Term > < Primary\ Term >$

3.2 The Model

This study is based on the categorial framework proposed by Bozşahin (1995; 1998; 1999). In this model morphology and syntax are treated not as separate components, but as a co-extensive domain. The morphological and syntactical processes are integrated, and semantic composition is performed in parallel to these. The idea behind this model is to deal with the difficulty of separating morphology and syntax in agglutinating languages, and the problems caused by processing them separately.

In Turkish, grammatical features can be indicated either by syntactical or morphological means. The grammatical features that are realized by words in a language such as English can be indicated by bound morphemes in Turkish. For instance, case marking is a morphologically marked function, whereas indirect objects are syntactically marked. When this morphosyntactic model is used, the distinction between the morphological and syntactical processes no longer exists. The categorial framework is morpheme-based, that is

the morpheme is put into the lexicon and it has the same lexical representation as the lexeme.

Semantic bracketing mismatches is an important issue. The problem in other languages has been pointed out by e.g. Carpenter (1997), Williams (1981), Moortgart (1988). An example from Turkish is given by Bozşahin(1999)[3].

(3) a. *otobüs bilet-ler-i*
bus ticket-PLU-COMP
'bus tickets' = (PLU(COMP ticket bus))

b. **otobüs bilet-i-ler*

In the example above, the nominal compounding morpheme *-i* should come before the plural morpheme *-ler* in order to get the correct semantics. However, in Turkish plural morpheme must attach to nouns before the other morphemes (3b). Hence, the predicate-argument structure and the surface form conflict in this phrase. Bozşahin (1999) proposes a solution to this inconsistency. Pluralization and compounding are composed into one morpheme as plural compound which has the same properties of a compound morpheme. Therefore, the morpheme *-leri* should be treated as a composite lexical marker of pluralization and compounding, and the phrase in (3a) should be interpreted as (4). In this way, the bracketing problem disappears.

(4) *otobüs bilet-leri*
bus ticket-PLU.COMP
'bus tickets' = (PLU(COMP ticket bus))

A bound morpheme may modify not a single word, but a phrase (or a compound head), as in (5a). Here, the bound morpheme *-lu* scopes over the entire noun phrase *kırmızı panjur*. In (5b), another possible semantic bracketing is shown. Both derivations are reasonable, and both semantic forms are meaningful. When morphemes have the same status with the words, getting the two correct derivations is possible. A word-based approach will need rebracketing to get the semantics in (5a).

(5) a. [[[*kırmızı panjur*]-*lu*] *ev*]
red shutter-ADJ house
'the house with red shutter'

b. [*kırmızı* [[*panjur-lu*] *ev*]]
'the red house with shutter'

A new operator, *directional underspecification* (Bozşahin, 1998), is added to CCG in the model. It gives more flexibility to CCG, and is needed when arguments of a functor can scramble to either side of the functor. The neutral slash, (|), is the lexical operator used for this purpose. It is instantiated to either forward slash (/) or backward slash (\) during derivation. The categories of intransitive, transitive, and ditransitive verbs in Turkish are as follows when neutral slash is used:

(6) a. $IV = S|NP_1$

b. $TV = S|NP_1|NP_2$

c. $DV = S|NP_1|NP_3|NP_2$

4 Generation Using the Categorial Framework

4.1 The Lexicon

Since CCG is a lexicalist formalism, the lexicon used in this study plays an important part. Therefore, first it is necessary to describe the lexicon briefly. Because the idea is to make a generator which uses the same model with the parser implemented by Bozşahin (1998; 1999), the structure of the lexicon is also the same. In this model, morphemes have the same representation as words in terms of syntactic, semantic, phonological, and morphological aspects.

Every lexical entry has an ordered three-tuple description, (*Category:Type, Phon, Morph*), along with a list of all its possible allomorphs. As an example, a simplified version of the lexical entry for *uyu* 'sleep', whose category is $S|NP_1$, and lexical entry for the accusative case marker, whose category

[3]Constants in the PAS are represented in capital letters.

a. *uyu* := *((s,(Tense,Per,Num)* $\{\leq s-base\})$: *sleep˜A* |
(np,(1,nom,Per1,Num1) $\{\leq phrase\})$: *A*,
phon('uyu', []),
morph('v', (free,concat), [])
).

b. *ı,i,u,ü,yı,yi,yu,yü* := *((np,(Index,Case,Per,Num)* $\{\leq phrase\})$: *A* \
(n,(Index1,Case1,Per1,Num1) $\{\leq n-num\}$): *A*,
phon('(y)I', []),
morph('-ACC', (bound,affix), [])
).

Figure 1: The lexical entries for *uyu* 'sleep' and the accusative case marker.

is *NP\N*, are shown in Figure 1 in pseudo-Prolog notation. ˜ is the juxtaposition operator for the PAS.

The *Category:Type* pairing includes category and the semantics of the lexicon. The part after colon is the PAS. The three basic categories, *N, NP, S*, carry some specific information such as tense, person, number for *S*, and index, case, person, number for *N* and *NP*. The information in curly braces is the *hypocategory* of the category; it is used to provide a finer level of control (Bozşahin, 1999). Hypocategories assist the distinction in form-meaning correspondence. For instance, they help to differentiate a plural-marked *N* from a singular *N* which have in fact the same basic category. They allow a natural treatment of morphosyntactic composition without resorting to nonmonotonic operations.

Phon part contains the phonological form of the entry. Optional segments are put in paranthesis. Meta-phonemes are indicated by upper-case letters. For example, the phonological form of the accusative case marker is *(y)I*, therefore this entry has eight possible surface forms, hence its list of allomorphs consists of *ı, i, u, ü, yı, yi, yu, yü*.

Morph includes part of speech of the lexicon. Whether the morpheme is bound or free (i.e. word), is also indicated in this part. In addition, the type of morphological or syntactic attachment, i.e. if it is affixation, syntactic concatenation, reduplication, or clitic, is pointed out.

4.2 Generation Algorithm

This is an adoptation of a semantic-head-driven bottom-up generation algorithm (Calder et al., 1989; Shieber et al., 1989; Shieber et al., 1990; van Noord, 1990) which takes advantage of top-down input provided by the user as well as the bottom-up lexical information. This algorithm combines aspects of both top-down and bottom-up generation. Although the algorithm is adapted from the semantic-head driven generation algorithm, we do not use the term 'head' in our study, instead we use the term 'anchor'. This is due to the different notion of head in CG (Bouma, 1988).

The generation process gets as input the basic category of the desired output, (*N, NP, S*, or any partial construction derived from these categories), and the semantic representation in the form of PAS, and produces all possible surface forms as output. Since case, tense, person, and number information are not represented in the PAS, but given in the syntactic features of the lexical entries, this information is given as an optional input to the generator. Whether the subject and/or object of the output surface form will be dropped is declared as an optional parameter in the input.

The generator first finds the *anchor* of the output surface form in terms of unification using the input information. This part of the process is top-down. Then, the arguments of the matched lexical functor (if it is a

$$
\begin{array}{lcl}
uyu & := & S : sleep\tilde{}(PLU\tilde{}(tired\tilde{}cat)) \mid NP_{1,nom}\text{: } PLU\tilde{}(tired\tilde{}cat) \\
DI & := & (S : sleep\tilde{}(PLU\tilde{}(tired\tilde{}cat)) \mid NP_{1,nom}\text{: } PLU\tilde{}(tired\tilde{}cat) \backslash \\
 & & (S : sleep\tilde{}(PLU\tilde{}(tired\tilde{}cat)) \mid NP_{1,nom}\text{: } PLU\tilde{}(tired\tilde{}cat) \\
\text{-}lAr & := & N : PLU\tilde{}(tired\tilde{}cat) \backslash N : tired\tilde{}cat \\
yorgun & := & N : tired\tilde{}cat \,/ N\text{: } cat \\
kedi & := & N : cat
\end{array}
$$

Figure 2: Lexical entries for the example in the order of their generation.

functor) is generated in a bottom-up fashion. The categorial operators reveal the surface ordering of the functor and the arguments. For instance, in $X\backslash Y$, Y preceeds X, whereas in X/Y, X preceeds Y in the surface form. The function that generates the arguments is called recursively until it has found all of the arguments of the anchor. This anchor-driven generation algorithm uses syntactic, semantic, and morphological information in the lexicon.

Let us consider a sample running of the algorithm with the following input:

(7) `s-past:sleep~(PLU~(tired~cat))`

The generator first tries to find an anchor whose category is S and whose PAS is unifiable with the given PAS. The anchor is the verb *uyu* 'sleep' (Figure 1). Since there is tense information in the input and this is the verb of the sentence, the appropriate tense morpheme, that is the past tense morpheme, is found using the category and tense information and is added to the verb.

After this the generation of the argument(s) begins. The intransitive verb *uyu* has only one argument: its subject. The generation algorithm is called using this argument. The input is the part after the categorial operator. Since the nominative case marker in Turkish has no surface realization, it does not have a lexical entry. Therefore, we do what we can call '*syntactic type-lowering*', and the generator looks for an N instead of an $NP_{1,nom}$. It finds the plural marker morpheme whose category is $N_{pl}\backslash N_{sg}$, therefore is a functor itself. After this, the arguments of the plural marker is generated. N_{sg}[4] is sent to the generator as input and the generator attempts to find a lexical entry that unifies with this input. The adjective *yorgun* 'tired' is found. Since this lexical entry is also a functor, its argument (*N:cat*) is sent to generator, and the noun *adam* 'man' returns as output.

At this point, the plural marker morpheme *-ler* is attached to the phrase *yorgun kedi* 'tired cat' generating *yorgun kedi-ler* 'tired cats'. This is the argument of the anchor *uyu*. The category of the anchor was $S|NP_{1,nom}$. The neutral slash is first substituted with backward slash, and (8a) is produced as output. Then, the generator attempts to find if there is another possible surface form for the input. This time the neutral slash in $S|NP_{1,nom}$ is substituted with forward slash and the argument $NP_{1,nom}$ is put to the right of its functor producing the output in (8b). The generator fails to find any more answers, therefore the generation process ends. Two surface forms (8) have been generated for the input (7).

(8) a. *yorgun kedi-ler uyu-du*
tired cat-PLU sleep-TENSE
'the tired cats slept.'

b. *uyu-du yorgun kedi-ler*

All the lexical entries in this example are shown in Figure 2 in a simplified form. They are given in the order of their generation. The left-hand side shows the phonological form of the entry which is used during .

[4] All the information in *Category:Type* pairing is used during this process.

5 Discussion

Type-raised morphemes are not put in the lexicon and are not used during generation. During the generation process, only application rules of CCG (forward and backward application rules) are used, hence we can say that this is an 'application-based' approach. The consequence of this is that not all word orders of a sentence can be generated. Although all of the two possible orders of an intransitive sentence can be generated successfully, only four out of six possible permutations of a transitive sentence (SOV, OVS, SVO, and VOS), and only eight out of twenty-four possible permutations of a ditransitive sentence can be generated. The order of generation after the anchor always begins from the last argument. As a result of this, the verb and the direct object of a transitive sentence are always adjacent, since the category of a transitive verb is $S|NP_1|NP_2$.

The pro-drop characteristic of Turkish is also reflected; if the subject or the direct-object of the sentence is a pronoun, it can be dropped according to the input.

The *Phon* part and the attachment information in *Morph* are used during the surface realization of the output. The metaphonemes are replaced according to the morphophonemic rules such as vowel harmony, consonant-drop, etc.

Some sample runs from the generator are given in Figure 3. The first line for each entry is the input to the generator. The second line is the output. The third line is the gloss, and the fourth line is the English gloss (not part of the output). If there are more than one output, these are also given. Each output of the generator was given to the parser as input in order to check the consistency of two systems. In all examples so far they turn out to be consistent.

Although there are other studies about generation of Turkish, these are word-based approaches. Morphology is handled outside the generation system (Hakkani et al., 1996; Hoffman, 1994). The main difference in this study is the integration of morphology and syntax, and the use of a morpheme-based lexicon. There is no distinction between syntax and morphology, and morphemes are the main building blocks. Hence there is no need for rebracketing due to semantics either in parsing or generation.

For a more detailed discussion of this study and further explanation of how the generator works, refer to (Karagol-Ayan, 2000)

6 Conclusion

The distinction between morphology and syntax in agglutinative languages is difficult. Morphology plays an important role in marking grammatical functions in these languages. This study is about the generation of such a language, namely Turkish, from PAS using a morpheme-based categorial model. This categorial framework claims that inflectional morphology is not different from syntax, so that they should be treated as one. The model is based on CCG, so it is a lexicalist formalism. In the lexicon, morphemes and words have the same representation in terms of syntactic, semantic, phonological, and morphological aspects. The model correlates syntax, inflectional morphology, and semantics transparently. Generation of morphemes that have phrasal scope are not different from the generation of other morphemes since this is a morpheme-based generation.

The main drawback of this study is that not every word order in Turkish can be generated. The future work will concantrate on this subject. Baldridge (2000) argues that CCG formalism must be augmented in order to give a principal account of local scrambling (word order variation within a clause). Addressing the issue of local scrambling, he proposes Set-CCG, an augmentation of CCG which handles local scrambling straightforwardly. In the formalism, if Set-CCG is used instead of CCG, local scrambling may be handled straightforwardly without type-raising and lexical ambiguity.

References

Jason M. Baldridge. 2000. Strong equivalence of

CCG and set-CCG. ms., Edinburgh University.

Gosse Bouma. 1988. Modifiers and specifiers in categorial unification grammar. *Linguistics*, 26:21–46.

Cem Bozşahin and Elvan Göçmen. 1995. A categorial framework for composition in multiple linguistic domains. In *Proceedings of the 4th International Conference on Cognitive Science of Natural Language Processing*, Dublin.

Cem Bozşahin. 1998. Deriving the predicate-argument structure for a free word order language. In *Proceedings of COLING-ACL*, pages 167–173, Montreal.

Cem Bozşahin. 1999. Categorial morphosyntax: Transparency of the morphology-syntax-semantics interface. ms., METU.

Jonathan Calder, Mike Reape, and Henk Zeevat. 1989. An algorithm for generation in unification categorial grammar. In *Proceedings of the 4th Conference of the European Chapter of the Association for Computational Linguistics*, pages 233–240.

Bob Carpenter. 1997. *Type-Logical Semantics*. MIT Press, Cambridge, MA.

Dilek Zeynep Hakkani, Kemal Oflazer, and İlyas Çiçekli. 1996. Tactical generation in a free constituent order language. In *Proceedings of the 8th International Workshop on Natural Language Generation*, Sussex, UK.

Beryl Hoffman. 1994. Generating context-appropriate word orders in Turkish. In *Proceedings of the International Workshop on Natural Language Generation*.

Burcu Karagol-Ayan. 2000. Generation of turkish surface form from a morphemic lexicon. Master's thesis, Middle East Technical University, Turkey.

Michael Moortgart. 1988. Mixed composition and discontinuous dependencies. In Richard T. Oehrle, Emmon Bach, and Deirdre Wheeler, editors, *Categorial Grammars and Natural Language Structures*. D. Reidel, Dodrecht.

Stuart M. Shieber, Gertjan van Noord, Robert C. Moore, and Fernando C. N. Pereira. 1989. A semantic-head-driven generation algorithm for unification-based formalisms. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*.

Stuart M. Shieber, Gertjan van Noord, Robert C. Moore, and Fernando C. N. Pereira. 1990. Semantic-head-driven generation. *Computational Linguistics*, 16(1), March.

Mark Steedman. 1985. Dependency and coordination in the grammar of Dutch and English. *Language*, 61(3):523–568.

Mark Steedman. 1987. Combinatory grammars and parasitic gaps. *Natural Language and Linguistic Theory*, 5:403–439.

Mark Steedman. 1988. Combinators and grammars. In Richard T. Oehrle, Emmon Bach, and Deirdre Wheeler, editors, *Categorial Grammars and Natural Language Structures*. D. Reidel, Dordrecht.

Mark Steedman. 1996. *Surface Structures and Interpretation*. MIT Press, Cambridge, MA.

Gertjan van Noord. 1990. An overview of head-driven bottom-up generation. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*, Cognitive Science Series, pages 141–166. Academic Press, New York.

Edwin Williams. 1981. On the nouns 'lexically related' and 'head of a word'. *Linguistic Inquiry*, 12(2):245–274.

- s-past-1-sg-dropsub-dropobj : forget˜she˜i
  Unut-tu-m.
  forget-TENSE-PER
  '(I) forgot (her).'

- s-past : read˜(COMP˜book˜(REL˜(at˜PRO˜house)˜PRO))˜mehmet
  Mehmet ev-de-ki kitab-ı-nı oku-du.
  mehmet.NOM house-LOC-NREL book-COMP-ACC read-TENSE
  'Mehmet read his book that is at the house.'
  Mehmet oku-du ev-de-ki kitab-ı-nı
  Ev-de-ki kitab-ı-nı oku-du Mehmet
  Oku-du ev-de-ki kitab-ı-nı Mehmet

- n : REL˜(at˜(ANA˜ball)˜house)˜ball
  ev-de-ki top
  house-LOC-NREL ball
  'the ball that is at the house'

- n : POSS˜(COMP˜book˜(REL˜(at˜PRO˜table)˜PRO))˜mehmet
  mehmed-in masa-da-ki kitab-ı
  mehmet-AGR table-LOC-NREL book-COMP.POSS.3s
  'mehmet's book that is on the table'

- n : REL˜(see˜(REL˜(read˜(ANA˜book)˜child)˜book)˜(ANA˜man))˜man
  çocuğ-un oku-duğu kitab-ı gör-en adam
  child-AGR read-REL.OP book-ACC see-REL.SP man
  'the man that saw the book that the child read'

- n : green˜(PLU˜(COMP˜ticket˜bus))
  yeşil otobüs bilet-leri
  green bus ticket-PLU.COMP
  'green bus tickets'

- n : COMP˜(COMP˜rate˜interest)˜(COMP˜card˜(annual˜credit))
  yıllık kredi kart-ı faiz oran-ı
  credit card-COMP annual interest rate-COMP.POSS.3s
  'annual credit card interest rate' (credit is annual)

- n : annual˜(COMP˜(COMP˜rate˜interest)˜(COMP˜card˜credit))
  yıllık kredi kart-ı faiz oran-ı
  annual credit card-COMP interest rate-COMP.POSS.3s
  'annual credit card interest rate' (credit card interest rate is annual)

- s-past : see˜(POSS˜(COMP˜ticket˜car)˜girl)˜(REL˜(look˜(POSS˜child˜mehmet)˜(ANA˜woman))˜woman)
  mehmed-in çocuğ-u-na bak-an kadın kız-ın araba bilet-i-ni gör-dü
  mehmet-GEN.3 child-POSS.3s-DAT look-REL.SP woman girl-GEN.3 car ticket-COMP.POSS.3s-ACC see-TENSE
  'the woman who looks mehmet's child saw girl's car ticket'
  Kız-ın ev-de-ki araba bilet-i-ni gör-dü mehmed-in çocuğ-u-na bak-an kadın
  Mehmed-in çocuğ-u-na bak-an kadın gör-dü kız-ın ev-de-ki araba bilet-i-ni
  Gör-dü kız-ın ev-de-ki araba bilet-i-ni mehmed-in çocuğ-u-na bak-an kadın

Figure 3: Sample runs from the generator.