Abstract

This paper describes tactical generation in a free constituent order language, Turkish. In Turkish, the order of the constituents may change according to the information structure of the sentences to be generated. In the absence of any information regarding the information structure of a sentence (i.e., topic, focus, background, etc.), the constituents of the sentence obey a default order, but the order is almost freely changeable, depending on the constraints of the text or discourse. We have used a recursively structured finite state machine for handling the changes in constituent order, implemented as a right-linear grammar backed on. Our implementation environment is the GenKit system, developed at Carnegie Mellon University (Center for Machine Translation). Morphological realization has been implemented using an external morphological analysis/generation component which performs concrete morpheme selection and handles morphographemic processes.

Introduction

Natural Language Generation is the operation of producing natural language sentences using specified communicative goals. This process consists of three main kinds of activities (McDonald, 1987):

1. The goals the utterance is to obtain must be determined.
2. The way the goals may be obtained must be planned.
3. The plans should be realized as text.

Tactical generation is the realization, as linear text, of the contents specified usually using some kind of a feature structure that is generated by a higher level process such as text planning, or transfer in machine translation applications. In this process, a generation grammar and a generation lexicon are used.

As a component of a large scale project on natural language processing for Turkish, we have undertaken the development of a generator for Turkish sentences. In order to handle the variations in the constituent order dictated by various information structure constraints, we have used a recursively structured finite state machine instead of enumerating grammar rules for all possible word orders. A second reason for this approach is that many constituents, especially the arguments of verbs, are optional and dealing with such optionality within rules proved to be rather problematic. Our implementation is based on the GenKit environment developed at Carnegie Mellon University (Center for Machine Translation). GenKit provides writing a context-free back-end grammar along with feature structure constraints on the non-terminals.

The paper is organized as follows: The next section presents relevant aspects of constituent order in Turkish sentences and factors that determine it. We then present an overview of the feature structures for representing the contents and the information structure of these sentences, along with the recursive finite state machine that generates the proper order required by the grammatical and information structure constraints. Later, we give the highlights of the generation grammar architecture along with some example rules and sample outputs. We then present a discussion comparing our approach with similar work on Turkish generation and conclude with some final comments.
In terms of word order, Turkish can be characterized as a subordinating language in which constituents at certain phrase levels can change order rather freely, depending on the constraints of text or discourse. The morphology of Turkish enables morphological markings on the constituents to signal their grammatical roles without relying on their order. However, this does not mean that word order is immaterial. Sentences with different word orders reflect different pragmatic conditions, in that, topic, focus and background information conveyed by such sentences differ.

Information conveyed through intonation, stress and/or clefting in fixed word order languages such as English, is expressed in Turkish by changing the order of the constituents. Obviously, there are certain constraints on constituent order, especially inside noun and postpositional phrases. There are also certain constraints at sentence level when explicit case marking is not used (e.g., with indefinite direct objects).

In Turkish, the information which links the sentence to the previous context, the topic, is in the first position. The information which is new or emphasized, the focus, is in the immediately preverbal position, and the extra information which may be given to help the hearer understand the sentence, the background, is in the postverbal position (Erguvanli, 1979). The topic, focus and background information, when available, alter the order of constituents of Turkish sentences. In the absence of any such control information, the constituents of Turkish sentences have the default order: subject, expression of time, expression of place, direct object, beneficiary, source, goal, location, instrument, value determinant, verb. All of these constituents except the verb are optional unless the verb obligatorily subcategorizes for a specific lexical item in order to convey a certain (usually idiomatic) sense. The definiteness of the direct object adds a minor twist to the default order. If the direct object is an indefinite noun phrase, it has to be immediately preverbal. This is due to the fact that, both the subject and the indefinite are nominative case-marked. The accusative case-marked direct object triggers the productive use of the locative case (Erguvanli, 1979).
The generation process gets as input a feature structure that encodes the contents of a sentence. The feature structure is then processed by a finite state machine to produce the surface form of the sentence.

The finite state machine uses a set of transitions to generate the surface form. Each transition is labeled with a label that corresponds to a specific grammatical operation. The machine follows a set of rules to determine the order of the labels and generate the surface form.

The feature structures for sentences are represented using a case-frame representation. The case-frames correspond to the different roles that words can play in a sentence. The case-frames include information about the grammatical role of the word, such as subject, object, or prepositional phrase.

The generation process is recursive, and the machine can generate sentences of any length by recursively applying the transitions. The machine can also generate sentences with embedded clauses, which can be nested to any depth.

In the next sections, we will highlight relevant aspects of our feature structures for Turkish sentences. The feature structures are used to represent the different grammatical roles of words in a sentence. The machine uses these roles to determine the order of the words in the sentence and generate the correct surface form.
In all these cases, the main sentence generator also produces the sentential subjects and objects in addition to generating the main sentence. In this section we will briefly look at important issues in representing noun phrases.

**Issues in Representing Noun Phrases**

- The order of constituents in noun phrases is rather strict at a gross level, i.e., specifiers almost always precede modifiers and modifiers almost always precede the head noun, although there are numerous exceptions.
- Within each group, word order variation is possible due to a number of reasons. Among these groups, the order of quantifiers is close to that of the head noun, although there are numerous exceptions. Also, within each group, word order variation is possible due to a number of reasons.

- The order of quantitatively and qualitatively modifiers may change: the aspect that is emphasized is closer to the head noun. The indefinite singular determiner may follow a classifier in Turkish, which is essentially the equivalent of an English book in forms like book cover in English.

Similarly, the subject of any other constituent of a sentence can also be optionally dropped off. In addition to the sentential subjects and objects, the main sentence generator also produces the sentential subjects and objects in addition to generating the main sentence. In this section we will briefly look at important issues in representing noun phrases.

**Complex Sentences**

Complex sentences are combinations of simple sentences (or complex sentences themselves) which are linked by either conjoining or various relationships like conditional dependence, cause/result, etc. The generator works on a feature structure representing a complex sentence which may be in one of the following forms:

- a simple sentence.
- a series of simple or complex sentences conjoined by coordinating or bracketing conjunctions. Such sentences have the feature structure:

  - `TYPE conj
    - CONJ and
    - ELEMENTS list-of(complex-sentence)`

- sentences linked with a certain relationship.

  - `TYPE link-relation
    - ARG1 complex-sentence
    - ARG2 complex-sentence`
Figure 2: The finite state machine for generating the proper order of constituents in Turkish sentences.
Interfacing with Morphology

As Turkish has complex agglutinative word forms with productive inflectional and derivational morphology, we handle morphological processes outside our system using the generation component of a full-scale morphological analyzer of Turkish (Ozan et al., 1993). Within GenKit, we generate relevant abstract morphological features such as agreement, case markers, and voice, polarity, tense, aspect, mood, and agreement markers for verbal forms. This information is properly ordered at the interface and sent to the morphological generator, which

1. performs concrete morpheme selection, dictated by the morphological context,
2. handles morphographical phenomena such as vowel and consonant ellipses, and
3. produces an agglutinative surface form.

Grammar Architecture and Output

Our generation grammar is written in a formalism called Pseudo Unification Grammar (Bahl and Joshi, 1993). Each rule consists of a context-free phrase structure description and a set of feature constraint equations, which are used to express constraints on feature values. Non-terminals in the phrase structure part of a rule are referenced as X0, X1, etc. in the equations, where X0 corresponds to the non-terminal on the left-hand side, and m is the number of rules. To implement the sentence level generator, we use rules of the form:

S → XP · S

where S and XP denote some state in the intermediate representation and the constituent to be realized while taking this transition. The context-free rules are directly compiled into a finite state machine, and the performance of the system is essentially independent of the complexity of the number of rules, as the context-free rules are directly compiled into the finite state machine. The constraints on feature values are applied after the finite state machine has produced the surface forms. This method is particularly useful in spoken discourse, where the order of lexical and phrasal modifiers (e.g., corresponding to a postpositional phrase on the surface) may change. If the lexical modifier causes unnecessary ambiguity, the original position is preserved. In this case, the phrasal modifier always precedes the lexical modifier. Our sentence level generator performs successive morpheme selection, dictated by the morphological context, and sends the output to the morphological generator, which

1. performs concrete morpheme selection, dictated by the morphological context,
2. handles morphographical phenomena such as vowel and consonant ellipses, and
3. produces an agglutinative surface form.

Grammar Architecture and Output

Our generation grammar is written in a formalism called Pseudo Unification Grammar (Bahl and Joshi, 1993). Each rule consists of a context-free phrase structure description and a set of feature constraint equations, which are used to express constraints on feature values. Non-terminals in the phrase structure part of a rule are referenced as X0, X1, etc. in the equations, where X0 corresponds to the non-terminal on the left-hand side, and m is the number of rules. To implement the sentence level generator, we use rules of the form:

S → XP · S

where S and XP denote some state in the intermediate representation and the constituent to be realized while taking this transition. The context-free rules are directly compiled into a finite state machine, and the performance of the system is essentially independent of the complexity of the number of rules, as the context-free rules are directly compiled into the finite state machine. The constraints on feature values are applied after the finite state machine has produced the surface forms. This method is particularly useful in spoken discourse, where the order of lexical and phrasal modifiers (e.g., corresponding to a postpositional phrase on the surface) may change. If the lexical modifier causes unnecessary ambiguity, the original position is preserved. In this case, the phrasal modifier always precedes the lexical modifier. Our sentence level generator performs successive morpheme selection, dictated by the morphological context, and sends the output to the morphological generator, which

1. performs concrete morpheme selection, dictated by the morphological context,
2. handles morphographical phenomena such as vowel and consonant ellipses, and
3. produces an agglutinative surface form.

Grammar Architecture and Output

Our generation grammar is written in a formalism called Pseudo Unification Grammar (Bahl and Joshi, 1993). Each rule consists of a context-free phrase structure description and a set of feature constraint equations, which are used to express constraints on feature values. Non-terminals in the phrase structure part of a rule are referenced as X0, X1, etc. in the equations, where X0 corresponds to the non-terminal on the left-hand side, and m is the number of rules. To implement the sentence level generator, we use rules of the form:

S → XP · S

where S and XP denote some state in the intermediate representation and the constituent to be realized while taking this transition. The context-free rules are directly compiled into a finite state machine, and the performance of the system is essentially independent of the complexity of the number of rules, as the context-free rules are directly compiled into the finite state machine. The constraints on feature values are applied after the finite state machine has produced the surface forms. This method is particularly useful in spoken discourse, where the order of lexical and phrasal modifiers (e.g., corresponding to a postpositional phrase on the surface) may change. If the lexical modifier causes unnecessary ambiguity, the original position is preserved. In this case, the phrasal modifier always precedes the lexical modifier. Our sentence level generator performs successive morpheme selection, dictated by the morphological context, and sends the output to the morphological generator, which

1. performs concrete morpheme selection, dictated by the morphological context,
2. handles morphographical phenomena such as vowel and consonant ellipses, and
3. produces an agglutinative surface form.
The following are rule examples that implement some of the transitions from state 0 to state 1:

```
(/<S/> </=/=/> (/<S/>1/))
(/(x/0 control topic/) /=c /*undefined*/ (/)
(x/1 /= x/0/))
```

```
(/<S/> </=/=/> (/<Subject/> /<S/>1/))
(/(x/0 control topic/) /=c subject/ (/)
(x/2 /= x/0/)
(x/2 arguments subject/) /= /*remove*/ (/)
(x/1 /= /(x/0 arguments subject/) (/))
```

```
(/<S/> </=/=/> (/<Time/> /<S/>1/))
(/(x/0 control topic/) /=c time/ (/)
(x/2 /= x/0/)
(x/2 adjuncts time/) /= /*remove*/ (/)
(x/1 /= /(x/0 adjuncts time/) (/))
```

The grammar also has rules for realizing a constituent like `<Subject/>` or `<Time/>` (which may eventually call the same rules if the argument is sentential) and rules above for traversing the finite state machine from state 1 on.

### Examples

In this section, we provide feature structures for three example sentences which only differ in their information structures. Although the following feature structures seem very similar, they correspond to different surface forms.

#### 1

Ahmet d / un kitab/ masada

Ahmet y esterda y boo k /+ ACC table /+LOC

`Ahmet left the book on the table`.

#### 2

It was Ahmet who left the book on the table.

Ahmet /+ PST+3SG

D/ un kitab/ Ahmet

`It was Ahmet who left the book on the table`.  

#### 3

The table was high.

D/ un kitab/ masada

Ahmet /+ ACC table /+LOC

In this section, we provide feature structures for three example sentences which only differ in their information structures. Although the following feature structures seem very similar, they correspond to different surface forms.
The solid arrows represent the transitions of the grammar. We have presented the highlights of our work in the right-linear back one.

**References**

We have presented the highlights of our work in the right-linear back one.

**Conclusions**

We have presented the highlights of our work in the right-linear back one. Since the transitions are consistent with the grammar, the solid arrows represent the transitions of the grammar. We have presented the highlights of our work in the right-linear back one.
Figure 3: The transitions followed for generating sentence 1.

In Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics, Tbilisi, Georgia.

A CCG approach to word order languages. In Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics.:

