Natural Language Engineering **4** (2): 115–134. Printed in the United Kingdom

© 1998 Cambridge University Press

Tactical generation in a free constituent order language

DILEK ZEYNEP HAKKANI and KEMAL OFLAZER

Department of Computer Engineering and Information Science, Faculty of Engineering, Bilkent University, 06533 Bilkent, Ankara, Turkey
e-mail: {hakkani,ko}@cs.bilkent.edu.tr

(*Received 3 March 1998*)

Abstract

This paper describes tactical generation in Turkish, a free constituent order language, in which 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 flow or discourse. We have used a recursively structured finite state machine (much like a Recursive Transition Network (RTN)) for handling the changes in constituent order, implemented as a right-linear grammar backbone. 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.

1 Introduction

Generation of natural language sentences from communicative goals specified in the form of some internal representation is an important component of many natural language processing applications, such as machine translation systems, natural language interfaces to database systems, speech generation systems, text skimming and summarization systems, etc. Natural language generation comprises three main activities (Reiter 1995):

1. The information to be communicated to the user and the way this information should be structured, must be determined. These, usually simultaneous, tasks are called *content determination* and *text planning*, respectively.
2. The splitting of information among individual sentences and paragraphs must be determined (sentence planning). During this process, in order to make a smooth text flow, *cohesion devices* (such as pronouns) to be added, should be dictated.
3. The individual sentences should be generated in a grammatically correct manner (realization).

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

In most natural language generation systems there are two different components (Dale 1992; van Noord 1990):

1. the *strategic generator*, which implements the first two of the activities above, and
2. the *tactical generator*, which implements the last one of the activities above.

In this paper, we present our work on natural language generation: the design and implementation of a tactical generator for Turkish. Our target language, Turkish, can be considered as a *subject-object-verb* (SOV) language, in which constituents can change order rather freely, at almost all sentential phrase levels, depending on the constraints of text flow or discourse. This freeness in constituent order is, to a certain extent, facilitated by morphology. Morphological markings on the constituents express their grammatical roles without relying on their order. A tactical generator for a free word order language, like Turkish, should be able to deal with these word order variations using any additional (usually extra-grammatical) information provided. The word order variations in Turkish are, for most part, dictated by the information structure constraints which capture and encode, to a certain extent, discourse related factors that influence word order. In order to deal with all possible constituent order variations, we have used a grammar based on a recursively structured finite state machine (much like a RTN[1]), whose behaviour is controlled by the information structure (whenever provided), and which falls back on a default order instead of (over-) generating all possible orders. A second reason for this approach is that many constituents, especially the arguments of verbs are typically optional and dealing with such optionality within a CFG rule backbone proved to be rather problematic from an implementation point of view. This is because, the number of rules one has to write to cover word order and optionality variations becomes very large and unmanageable under such an approach.

Our implementation environment is the GenKit generation system (Tomita *et al.* 1988), developed at Carnegie Mellon University, Center for Machine Translation. Morphological realization has been performed using an external morphological analysis/generation component, which was developed using Rank Xerox Two Level Morphology Tools (Oflazer 1993).

We currently plan to use this tactical generator in a prototype transfer-based human-assisted machine translation system from English to Turkish (Turhan 1997), which uses SRI's Core Language Engine (Alshawi 1992) at its source side (see Figure 1 for the architecture of this machine translation system). Although the translation domain of this machine translation system is computer manuals, we have developed this generator bearing in mind that it can also be used for other domains and in other applications. For example, this system is currently being used for generating Turkish sentences from interlingua representations (Mitamura *et al.* 1991) together with a mapping system (Hakkani *et al.* 1997), which produces the case frames from interlingua representations.

[1] Throughout the paper, we have deliberately avoided the use of the term RTN, as we feel that it may conjure up the notion of parsing as these devices have almost exclusively been used in this context; but the the general idea is the same.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

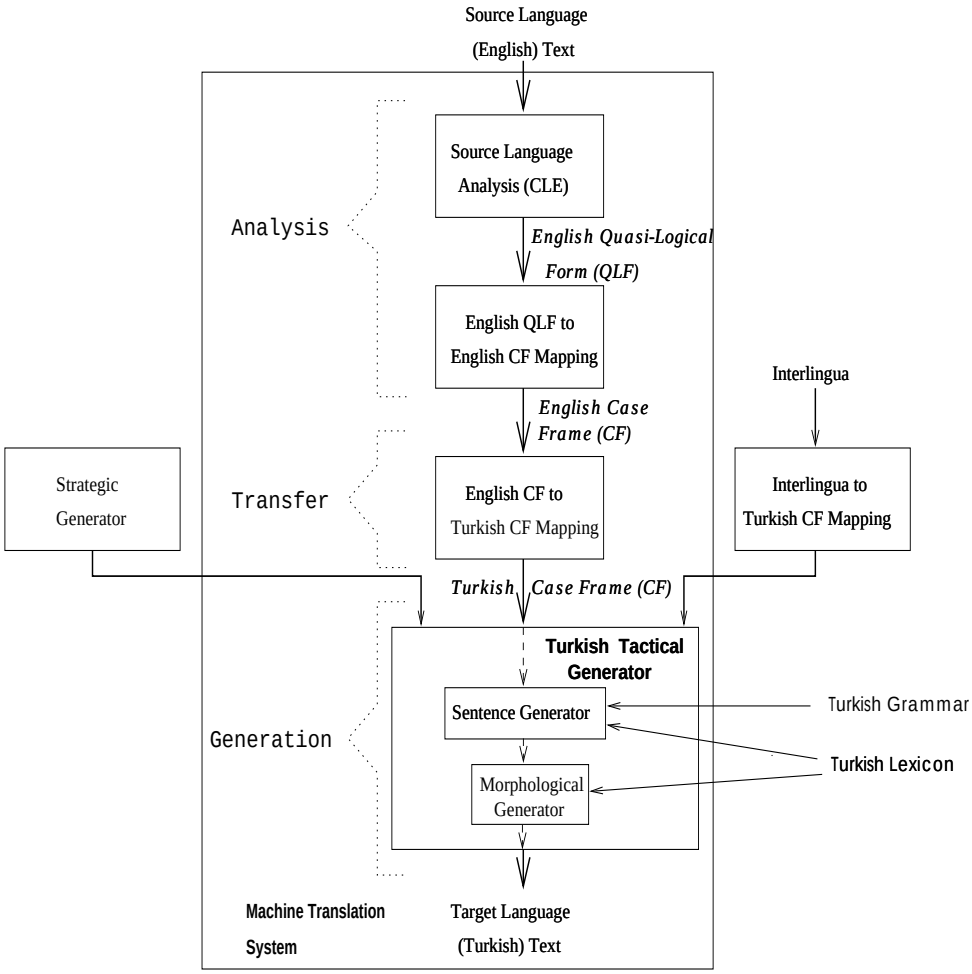


Fig. 1. Outline of the machine translation project.

In section 2, we present some relevant properties of Turkish and then discuss aspects of our system that make it different from other relevant approaches. Following this, we present the input representations for Turkish sentences, and noun phrases, and our approach to surface realization. We conclude after giving some examples from our system.

2 Turkish

In this section, we present relevant information about the word order of Turkish sentences, and the information structure which plays an important role in constituent order changes. We then present some translation examples from English to Turkish, to emphasize the differences among these languages.

2.1 Word order

The constituents of the Turkish sentences obey a default order in the absence of any information regarding the information structure of a sentence. In this respect,

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Turkish can be characterized as a *subject-object-verb (SOV) language*, in which constituents at certain phrase levels can change order rather freely, depending on the constraints of text flow or discourse. Morphologically, Turkish is an agglutinative language with very productive inflectional and derivational suffixation processes. The surface realization of words are further constrained by morphographemic processes like vowel harmony, etc. The grammatical roles of constituents can (usually) be identified without relying on their position in a sentence with explicit morphological case markings on them. For example, the word 'masa' (table) case marked accusative is a definite direct object, however the same word when case marked dative expresses a goal (unless it is accompanied by an idiosyncratic verb which subcategorizes for a dative complement):[2]

masa+yı	(Definite direct object – usually theme)
`table+ACC`	
masa+ya	(Dative object – usually goal)
`table+DAT`	

Turkish is also a *pro-drop* language, which allows the subjects to drop when they can be recovered contextually or via syntactic agreement on the verb. For example, in the second sentence below, the subject is dropped, and both sentences convey the same information, but in the first one the subject is emphasized.

(1) a. Ben uyudum.
`I sleep+PAST+1SG`
'I slept.'

b. Uyudum.
`sleep+PAST+1SG`
'I slept.'

The first sentence can be an answer to the first question below, but not to the second question, and the second sentence can be given as an answer only to the second question, where the person involved is already contextually available.

(2) a. Kim uyudu?
`who sleep+PAST+3SG`
'Who slept?'

b. Ne yaptın?
`what do+PAST+2SG`
'What did you do?'

[2] In the glosses, `3SG` and `1SG` denote third person singular and first person singular verbal agreement, `P1PL` and `P3SG` denote first person plural and third person singular possessive agreement, `WITH` denotes a derivational marker deriving adjectives from nouns, `LOC, ABL, DAT, GEN, ACC` denote locative, ablative, dative, genitive, and accusative case markers, `PAST` denotes past tense, and `INF` denotes a marker that derives an infinitive form from a verb.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

2.2 Information structure

The *information structure* of a sentence captures and encodes the speaker's communicated information relative to his/her beliefs about the hearer's information state (Vallduví 1994). This information is captured by three main indicators:

1. the topic,
2. the focus, and
3. the background.

In free word order languages, these are indicated by order variations, contrary to fixed word order languages, in which intonation, stress, and/or clefting are used (Hoffman 1995-1).

Erguvanlı (1979), has examined the word order variations of Turkish sentences. In Turkish, the information which links the utterance to the previous context, the *topic*, is in the sentence initial position. For example, the topic of sentence (b) below, is the direct object, which is a pronoun:

(3) a. Tarih kitabım ikinci rafta.
`history book+P1SG second shelf+LOC`
'My history book is on the second shelf.'

b. Onu da getirebilir misin?
`it+ACC too bring+ABILITY+QUES+2SG`
'Could you bring it too?'

The information which is new or emphasized, the *focus*, is in the immediately preverbal position. For example, in the answer to the following question, the subject, 'the cat', is the focus:

(4) Q: Vazoyu kim kırdı?
`vase+ACC who break+PAST+3SG`
'Who broke the vase?'

A: Vazoyu kedi kırdı.
`vase+ACC cat break+PAST+3SG`
'The cat broke the vase.'

The additional information which may be given to help the hearer understand the sentence, the *background*, is in the post verbal or sentence final position. For example, in the second sentence below, the subject, 'Ayşe', which is also the subject and the topic of the first sentence, is the background.

(5) a. Ayşe bütün kitaplarını eve götürmek
`Ayşe all book+PLU+P3SG+ACC home+DAT bring+INF`

istedi.
`want+PAST+3SG`
'Ayşe wanted to bring all her books home. '

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

b. Fakat, tarih kitabını okulda unuttu
But history book+P3SG+ACC school+LOC forget+PAST+3SG

Ayşe.
Ayşe
'But, she, Ayşe, forgot her history book at school.'

The information structure of a sentence can be obtained using syntactic clues in the source language in machine translation (Hajičová *et al.* 1993; Steinberger 1994) or using algorithms that determine the topic and focus of the target language sentences using Centering Theory (Styś *et al.* 1995), and given versus new information (Hoffman 1996). Hajičová *et al.* have used the input information on definiteness and lexical semantic properties of words, word order and the systemic ordering of kinds of complementations to automatically identify topic and focus. Similarly, Steinberger suggested a method to automatically recognize the categories *theme, rheme* and *contrastive focus* which determine the word order of constituents. Styś *et al.* tried to identify center information analyzing the discourse to generate communicatively adequate sentences in English–Polish machine translation. Hoffman presents algorithms which use contextual information to determine the topic and focus of Turkish sentences which are translated from English.

2.3 Translation examples from English to Turkish

In this section we will present translation examples from English to Turkish, emphasizing the differences in the word order of these languages. As we mentioned earlier, information conveyed by intonation, stress, and/or clefting in fixed word order languages, is conveyed through changes in word order (along with prosody), in free word order languages like Turkish. For example, in the following question-answer sequences, the first set of Turkish answers (A1s) differ in word order, whereas, the corresponding English sentences are the same, although they will probably have different prosody. The second set of Turkish answers (A2s) can also be given as answers to the respective questions.

(6) Q: Ayşe'ye kim kitap verdi?
Ayşe+DAT who book give+PAST+3SG
'Who gave a book to Ayşe?'

A1: Ayşe'ye Ali kitap verdi.
Ayşe+DAT Ali book give+PAST+3SG
'Ali gave a book to Ayşe.'

A2: Ali verdi.
Ali give+PAST+3SG
'Ali gave (it).'

Q: Ali Ayşe'ye ne verdi?
Ali Ayşe+DAT what give+PAST+3SG
'What did Ali give to Ayşe?'

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

A1: Ali Ayşe'ye kitap verdi.
`Ali Ayşe+DAT book give+PAST+3SG`
'Ali gave a book to Ayşe.'
A2: Kitap verdi.
`book give+PAST+3SG`
'(He) gave a book.'

Another example to difference in word order is (7). In this case, the difference in the word order of answer sentences results from the definiteness of the direct object 'kitap' (book).

(7) Q: Ali kitabı nerede okudu?
`Ali book+ACC where read+PAST+3SG`
'Where did Ali read the book?'
A: Ali kitabı evde okudu.
`Ali book+ACC home+LOC read+PAST+3SG`
'Ali read the book at home.'
Q: Ali nerede kitap okudu?
`Ali where book read+PAST+3SG`
'Where did Ali read a book?'
A: Ali evde kitap okudu.
`Ali home+LOC book read+PAST+3SG`
'Ali read a book at home.'

In English, only the determiner of the direct object changes, however, in Turkish, both the surface case and the position of the direct object change because of its definiteness.

3 Comparison with related work

There are three studies done on generation of Turkish sentences. The first one is the MSc thesis of Dick (1993), done in the Department of Artificial Intelligence, University of Edinburgh. The second one is the PhD thesis of Hoffman (1995-1), at the Computer and Information Science Department of University of Pennsylvania. The third study is Korkmaz's MSc thesis (Korkmaz 1996; Cicekli 1997), which was concurrently done with ours at Bilkent University.

Dick (1993) has worked on a classification-based language generator for Turkish. His goal was to generate Turkish sentences of varying complexity, from input semantic representations in Penman's Sentence Planning Language (SPL). However, his generator is not complete, in that, noun phrase structures in their entirety, postpositional phrases, word order variations, and many morphological phenomena are not implemented. Our generator differs from his in various aspects: We use a case-frame based input representation which we feel is more suitable for languages with free constituent order. Our coverage of the grammar is substantially higher than the coverage presented in his thesis and we also use a full-scale external morphological generator to deal with complex morphological phenomena of Turkish,

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Dick has attempted embedding the morphological phenomena into the sentence generator itself, which we feel is not the right place to deal with morphological generation.

Hoffman, in her thesis (Hoffman 1995-1), has used the Multiset–Combinatory Categorial Grammar formalism (Hoffman 1992), an extension of Combinatory Categorial Grammar (Steedman 1991) to handle free word order languages, to develop a generator for Turkish. Her generator also uses relevant features of the information structure of the input and can handle word order variations within embedded clauses. She can also deal with scrambling out of a clause dictated by information structure constraints, as her formalism allows this in a very convenient manner. The word order information is lexically kept as multisets associated with each verb. She has demonstrated the capabilities of her system as a component of a prototype database query system. We have been influenced by her approach to incorporate information structure in generation, but, since our aim is to build a wide-coverage generator for Turkish for use in a multitude of real world applications, rather than a demonstration system, we have opted to use a simpler formalism and a very robust implementation environment, in the sense that ours generate sentences in a default order when no information structure is available, whereas her generator generates nothing in such a case.

Korkmaz used a functional linguistic theory called Systemic-Functional grammar, and the FUF text generation system (Elhadad 1993) to implement a sentence generator for Turkish. His generator takes semantic descriptions of sentences and then produces a morphological description for each lexical constituent of the sentence. His generator does not handle long-distance scramblings, unbounded dependencies and discontinuous constituents.

4 Representation of Turkish sentences

To generate Turkish sentences of varying complexity, we opted to use a recursively structured finite state machine which can handle the changes in constituent order. Our implementation environment is the GenKit system (Tomita *et al.* 1988), developed at Carnegie Mellon University–Center for Machine Translation.

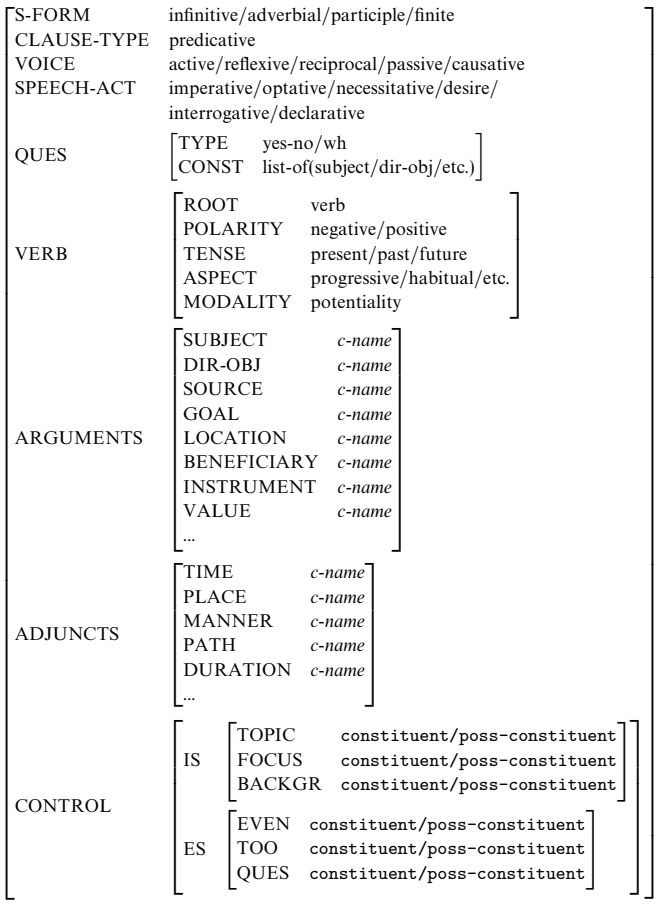
The generation process gets as input a feature structure representing the content of the sentence where all the lexical choices have been made by prior modules (e.g. transfer in machine translation), then produces as output the surface form of the sentence. The feature structures for sentences are represented using a case-frame representation. The fact that in Turkish sentential arguments of verbs adhere to the same morphosyntactic constraints as the nominal arguments enables a nice recursive embedding of case-frames of similar general structure to be used to represent sentential arguments.

In the following, we will present the way we treat Turkish sentences and noun phrases in generation.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

4.1 Sentences

Turkish sentences can be classified into three types, according to their verbs: predicative, existential, and attributive sentences. The case-frame representations for these sentence types are quite similar to each other and so are the the finite state machines we used when generating from them (see Hakkani (1996) for further information). So in this section, we will consider only predicative sentences. We use the following general case-frame feature structure to encode the contents of a predicative sentence:



Although most of the components here are self-explanatory, we think that it may be helpful to explain some of the attributes and values in attribute-value pairs. *c-name* is used to denote a noun phrase or a postpositional phrase which can be the value of the arguments or adjuncts. `constituent` denotes one of the adjuncts or arguments, and `poss-constituent` denotes possessor of one of the nominal constituents (if any). The CONTROL feature embeds the information structure, the IS feature, which guides our grammar in generating the appropriate sentential constituent order, and the emphasis structure, the ES feature encodes the constituent that takes a clitic, or that is subject to a yes-no question. The information in the IS feature is exploited by a right-linear grammar backbone (recursively structured nevertheless)

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

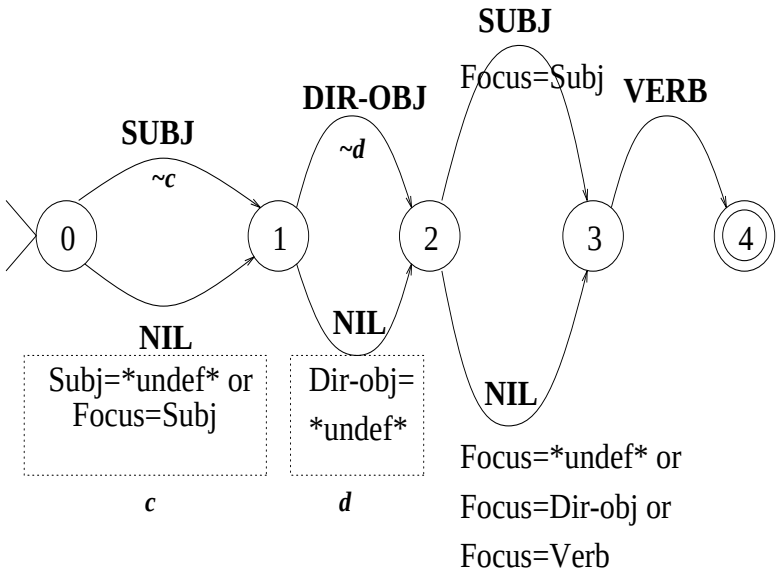


Fig. 2. Finite state machine for giving the outline of a grammar for the simple domain.

to generate the proper order of constituents at every sentential level (including embedded sentential clauses, possibly with their own information structure). A simplified outline of this right linear grammar will be given below as a recursively structured finite state machine. The recursive behavior of this finite state machine comes from the fact that the individual argument or adjunct constituents can also embed sentential clauses. The details of the feature structures for sentential clauses are very similar to the structure for the case-frame. Thus, when an argument or adjunct, which is a sentential clause, is to be realized, the clause is recursively generated by using the same set of transitions.

Before proceeding to the workings of the machine underlying the grammar, we will present an example for a much simpler grammar. In this simple domain, the only arguments of the verb are subject and direct object, there are no adjuncts, and the only constituent of the information structure is the focus. The default word order is *subject, direct object, verb*. The constituent to be emphasized, the focus, moves to the immediately preverbal position. So, in the following example, the first sentence is in the default order, and in the second sentence, the subject, 'Ali', is the focus:

(8) a. Ali topu attı. (Default Order)
`Ali ball+ACC throw+PAST+3SG`
'Ali threw the ball.'

b. Topu Ali attı. ('Ali' is focus)
`ball+ACC Ali throw+PAST+3SG`
'It was Ali, who threw the ball.'

The outline of a grammar to capture this behavior can be given with the finite state machine in Figure 2.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

There, the transition from the initial state to state 1, labeled **Subject** generates the subject, when it is defined and it is not the focus. Otherwise, a **NIL** transition is taken, which generates an empty string. Then, the transition from state 1 to state 2, labeled **Dir-obj**, generates the direct object, if it is defined. The transition from state 2 to state 3, labeled **Subject**, generates the subject if it is the focus. Finally, the verb is generated with the transition from state 3 to the final state, labeled **verb**.

Extending this structure to cover all the arguments of the verb, adjuncts, and information structure constituents, we get the finite state machine given in Figure 3 for generating Turkish predicative sentences.[3]

In this finite state machine, transitions are labeled by constraints and constituents (shown in bold face along a transition arc) generated when those constraints are satisfied. If any transition has a **NIL** label, then no surface form is generated for that transition. For example, the transitions from state 0 to state 1 generate the constituent which is the topic. If the topic is not defined, then the **NIL** transition is taken, which generates an empty string. The other transitions, from state 1 to state 14 generate the constituents in the default order. The transitions from state 14 to 15 generate the constituent which is the focus, and the constituent which is the background is generated by the transitions from state 17 to the final state.

4.2 Noun phrases

We use the following feature structure (simplified by leaving out irrelevant details) to describe the structure of a noun phrase:

$$
\begin{bmatrix}
\text{REF} & \begin{bmatrix} \text{ARG} & \textit{basic-concept} \\ \text{CONTROL} & \begin{bmatrix} \text{DROP} & +/- \text{ (default } -) \end{bmatrix} \end{bmatrix} \\
\text{CLASS} & \text{classifier} \\
\text{ROLES} & \textit{role-type} \\
\text{MODF} & \begin{bmatrix} \text{MOD-REL} & \text{list-of}(\textit{mod. relation}) \\ \text{ORDINAL} & \begin{bmatrix} \text{POSITION} & \textit{pos.} \\ \text{INTENSIFIER} & +/- \end{bmatrix} \\ \text{QUANT-MOD} & \textit{quantifier} \\ \text{QUALY-MOD} & \text{list-of}(\textit{simple-property}) \\ \text{CONTROL} & \begin{bmatrix} \text{EMPHASIS} & \text{quant./} \\ & \text{qual.} \end{bmatrix} \end{bmatrix} \\
\text{SPEC} & \begin{bmatrix} \text{DET} & \begin{bmatrix} \text{QUANTIFIER} & \textit{quant.} \\ \text{DEFINITE} & +/- \\ \text{REFERENTIAL} & +/- \\ \text{SPECIFIC} & +/- \end{bmatrix} \\ \text{SET-SPEC} & \text{list-of}(\textit{c-name}) \\ \text{SPEC-REL} & \text{list-of}(\textit{spec. relation}) \\ \text{DEMONS} & \textit{demonstrative} \end{bmatrix} \\
\text{POSS} & \begin{bmatrix} \text{ARGUMENT} & \textit{c-name} \\ \text{CONTROL} & \begin{bmatrix} \text{DROP} & +/- \\ \text{MOVE} & +/- \end{bmatrix} \end{bmatrix}
\end{bmatrix}
$$

[3] The reader can refer to a much more detailed version of this figure in the first author's M.Sc. thesis, available online from `http://www.nlp.cs.bilkent.edu.tr`.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

The outline of the grammar for generating Turkish noun phrases can also be given by a recursively structured finite state machine, which is very similar to the one for the sentences.

5 Grammar architecture

In the following sections, we will give some information about GenKit, the generation system that we have used, and then some example rules of our grammar.

5.1 GenKit

The CMU-CMT Generation Kit (GenKit) is a system which compiles a grammar into a sentence generation program. The grammar formalism used by GenKit is called *Pseudo Unification Grammar* (Tomita *et al.* 1988). Each rule of the grammar 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,...,xn` in the equations, where `x0` corresponds to the non-terminal in the left hand side, and `xn` is the n^{th} non-terminal on the right-hand side.

To implement the sentence level generator (described by the recursively structured finite state machine presented earlier), we use rules of the form:

$S_i \rightarrow$ XP S_j

where the S_i and S_j denote some state in the finite state machine and the XP denotes the constituent to be realized while taking the transition between states S_i and S_j, labeled XP. Using feature constraint equations, the relevant portions of the feature structure of S_i, are assigned to XP, and S_j. If this XP corresponds to a sentential clause, the same set of rules are recursively applied. This is a variation of the method suggested by Takeda (Takeda *et al.* 1991). Thus, there is no need to write a separate rule for each possible constituent order.

In generation, non-determinism, producing multiple surface forms for a given input, is a serious problem. If there is no information structure in the input, our generator generates the sentence in a default order, instead of generating all possible variations as is done, for example, in Karttunen and Kay (1985).

Since the context-free rules are directly compiled into tables, the performance of the system is essentially independent of the number of rules, but depends on the complexity of the feature constraint equations (which are compiled into LISP code). Currently, our grammar has 273 rules (excluding lexical rules), each with very simple and principled constraint checks. Of these 273 rules, 133 are for sentences and 107 are for noun phrases, and the remaining are for adverbials and verbs.

5.2 Example rules

The following are the simplified forms of our rules for the top three transitions from state 0 to state 1 in Figure 3:

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

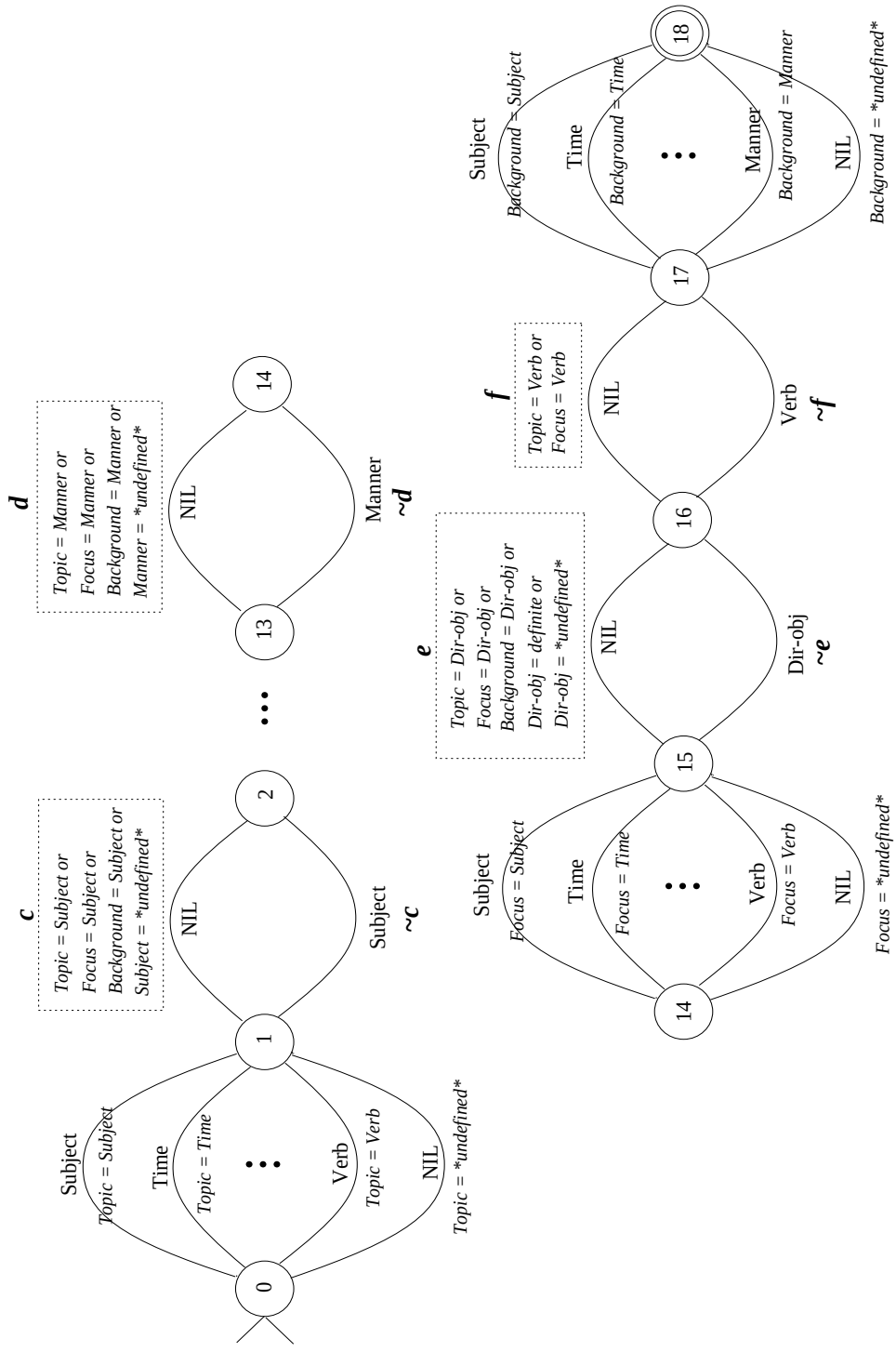


Fig. 3. Finite state machine for predicative sentences.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

```
(<S> <==> (<S1>)
    (
     ((x0 control topic) =c *undefined*)
     (x1 = x0))
    )

(<S> <==> (<Subject> <S1>)
    (
     ((x0 control topic) =c subject)
     (x2 = x0)
     ((x2 arguments subject) = *remove*)
     (x1 = (x0 arguments subject)))
    )

(<S> <==> (<Time> <S1>)
    (
     ((x0 control topic) =c time)
     (x2 = x0)
     ((x2 adjuncts time) = *remove*)
     (x1 = (x0 adjuncts time)))
    )
```

The first rule above is for the **NIL** transition, this transition is done if the topic is not defined. The second rule is for the transition labeled **Subject**, if the topic is subject, then this transition is done. In the feature constraint equations, it is checked whether the subject is the topic, and if so, the part of the feature structure for subject is assigned to `<Subject>`, and the remaining is assigned to `<S1>`. The third rule is for the transition labeled **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 like above for traversing the finite state machine from state 1 to the final state.

5.3 Interfacing with morphology

As Turkish has complex agglutinative word forms with productive inflectional and derivational morphological processes, we handle morphology outside our system using the generation component of a full-scale morphological analyzer of Turkish (Oflazer 1993). The separation of morphological generation from syntactic generation is founded on two main concerns: (1) From a language engineering aspect, the two processes are cleanly separable with a clean interface, hence it makes sense to make this separation and reuse the available morphological analysis/generation system; (2) In a language like Turkish morphological generation involves many processes (e.g. concrete morpheme selection, vowel harmony, etc.) which are of no concern at the syntactic generation level, as these processes are localized to lexical units.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Within GenKit, we generate relevant abstract morphological features such as: *agreement, possessive,* and *case* markers for nominals and *voice, polarity, tense, aspect, mood,* and *agreement* markers for verbal forms, *in addition to markers for all productive derivations*. This information is properly ordered at the interface and sent to the morphological generator, which then:

1. performs concrete morpheme selection, dictated by the morphotactic constraints and morphophonological context,
2. handles morphographemic phenomena such as vowel harmony, and vowel and consonant ellipsis, and
3. produces an agglutinative surface form.

For example, the following feature structures are the outputs of our generator for nominal, verbal, and adjectival forms, respectively. These are sent to the morphological generator, which then performs morpheme selections and converts them into the intermediate forms below (where H represents a high vowel, resolved with vowel harmony), and produces the agglutinative surface forms from these intermediate forms:

$$\begin{bmatrix} \text{CAT} & \text{NOUN} \\ \text{ROOT} & \text{kalem} \\ \text{AGR} & \text{3SG} \\ \text{POSS} & \text{1SG} \\ \text{CASE} & \text{GEN} \end{bmatrix}$$

$$\downarrow$$

kalem+∅+Hm+Hn

$$\downarrow$$

kalemimin

$$\begin{bmatrix} \text{CAT} & \text{VERB} \\ \text{ROOT} & \text{gel} \\ \text{SENSE} & \text{POS} \\ \text{TAM1} & \text{PROG1} \\ \text{AGR} & \text{1SG} \end{bmatrix}$$

$$\downarrow$$

gel+∅+Hyor+Hm

$$\downarrow$$

geliyorum

$$\begin{bmatrix} \text{CAT} & \text{ADJ} & \\ \text{STEM} & \begin{bmatrix} \text{CAT} & \text{NOUN} \\ \text{ROOT} & \text{boya} \\ \text{AGR} & \text{3SG} \\ \text{POSS} & \text{NONE} \\ \text{CASE} & \text{NOM} \end{bmatrix} \\ \text{SUFFIX} & \text{WITH} & \end{bmatrix}$$

$$\downarrow$$

boya+∅+∅+∅+lH

$$\downarrow$$

boyalı

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

6 Examples

In this section, we will present some example outputs of our generator, when run on a Sun Sparc Station 4. [4]

Example 1

```
INPUT:
  ; Adam elmayI kadIna verdi.
  ; The man gave the apple to the woman.
  ; This sentence is in the default order, so there will not be
  ; a CONTROL feature.

  ((s-form finite)
   (clause-type predicative)
   (voice active)
   (speech-act declarative)
   (verb
    ((root "ver")
     (sense positive)
     (tense past)
     (aspect perfect)))
   (arguments
    ((subject
      ((referent
        ((arg
          ((concept "adam")))
         (agr ((number singular)
               (person 3)))))))
     (dir-obj
      ((referent
        ((arg
          ((concept "elma")))
         (agr
          ((number singular)
           (person 3)))))
       (specifier
        ((quan
          ((definite +)))))))
     (goal
      ((referent
        ((arg
          ((concept "kadIn")))
         (agr
          ((number singular)
           (person 3)))))))))))

OUTPUT:
  "Turkish Sentence Generated is:"
    "[[CAT=NOUN][ROOT=adam][AGR=3SG][POSS=NONE][CASE=NOM]]
     [[CAT=NOUN][ROOT=elma][AGR=3SG][POSS=NONE][CASE=ACC]]
     [[CAT=NOUN][ROOT=kadIn][AGR=3SG][POSS=NONE][CASE=DAT]]
     [[CAT=VERB][ROOT=ver][SENSE=POS][TAM1=PAST][AGR=3SG]][PERIOD]"

    "adam elmayI kadIna verdi ."
  Evaluation took:
    0.7 seconds of real time
    0.0 seconds of user run time
    0.0 seconds of system run time
    0 page faults and
    588144 bytes consed.
```

[4] In our system, special characters of the Turkish alphabet are represented with upper-case ASCII characters, for example, ü is represented with U, ğ with G, etc.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Example 2

```
INPUT:
  ; KadIna adam verdi elmayI.
  ; It was the man who gave the woman the apple.
  ; This sentence is not in the default order, the destination/recipient,
  ; ‘‘kadIn’’, is the topic, the subject, ‘‘adam’’, is the focus,
  ; the direct object, ‘‘elma’’, is the background.

  ((s-form finite)
   (clause-type predicative)
   (voice active)
   (speech-act declarative)
   (verb
    ((root "ver")
     (sense positive)
     (tense past)
     (aspect perfect)))
   (arguments
    ((subject
      ((referent
        ((arg
          ((concept "adam")))
         (agr ((number singular)
               (person 3)))))))
     (dir-obj
      ((referent
        ((arg
          ((concept "elma")))
         (agr
          ((number singular)
           (person 3)))))
       (specifier
        ((quan
          ((definite +)))))))
     (goal
      ((referent
        ((arg
          ((concept "kadIn")))
         (agr
          ((number singular)
           (person 3)))))))))
   (control
    ((is
      ((topic goal)
       (focus subject)
       (background dir-obj))))))

OUTPUT:
  "Turkish Sentence Generated is:"
    "[[CAT=NOUN][ROOT=kadIn][AGR=3SG][POSS=NONE][CASE=DAT]]
     [[CAT=NOUN][ROOT=adam][AGR=3SG][POSS=NONE][CASE=NOM]]
     [[CAT=VERB][ROOT=ver][SENSE=POS][TAM1=PAST][AGR=3SG]]
     [[CAT=NOUN][ROOT=elma][AGR=3SG][POSS=NONE][CASE=ACC]][PERIOD]"

    "kadIna adam verdi elmayI ."
  Evaluation took:
    0.81 seconds of real time
    0.0 seconds of user run time
    0.0 seconds of system run time
    0 page faults and
    594568 bytes consed.
```

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Example 3

```
INPUT:
  ; Adam kadInIn okuduGu kitabI istedi.
  ; The man wanted the book that the woman read.
  ; The direct object has a sentential modifier.

  ((s-form finite)
   (clause-type predicative)
   (voice active)
   (speech-act declarative)
   (verb
    ((root "iste")
     (sense positive)
     (tense past)
     (aspect perfect)))
   (arguments
    ((subject
      ((referent
        ((arg
          ((concept "adam")))
         (agr
          ((number singular)
           (person 3)))))))
     (dir-obj
      ((roles
        ((role theme)
         (arg
          ((s-form participle)
           (clause-type predicative)
           (voice active)
           (speech-act declarative)
           (verb
            ((root "oku")
             (sense positive)
             (tense past)))
           (arguments
            ((dir-obj
              ((referent
                ((arg
                  ((concept "kitap")))
                 (agr
                  ((number singular)
                   (person 3)))))
               (specifier
                ((quan
                  ((definite  +)))))))
             (subject
              ((referent
                ((arg
                  ((concept "kadIn")))
                 (agr
                  ((number singular)
                   (person 3))))))))))))))))))

OUTPUT:
  "Turkish Sentence Generated is:"
    "[[CAT=NOUN][ROOT=adam][AGR=3SG][POSS=NONE][CASE=NOM]]
     [[CAT=NOUN][ROOT=kadIn][AGR=3SG][POSS=NONE][CASE=GEN]]
     [[CAT=VERB][ROOT=oku][SENSE=POS][CONV=ADJ=DIK][POSS=3SG]]
     [[CAT=NOUN][ROOT=kitab][AGR=3SG][POSS=NONE][CASE=ACC]]
     [[CAT=VERB][ROOT=iste][SENSE=POS][TAM1=PAST][AGR=3SG]][PERIOD]"
    "adam kadInIn okuduGu kitabI istedi ."
```

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

```
Evaluation took:
  0.85 seconds of real time
  0.0 seconds of user run time
  0.0 seconds of system run time
  0 page faults and
  863216 bytes consed.
```

7 Conclusions

We have presented the highlights of our work on tactical generation in Turkish, a free constituent order language with agglutinative word structures. Our generator takes as input the information structure of the sentence (topic, focus and background), as well as the content information. The information structure is used to select the appropriate word order. Our grammar uses a right-linear rule backbone which implements a (recursively structured) finite state machine for dealing with alternative word orders. We have also provided for constituent order and stylistic variations within noun phrases based on certain emphasis and formality features. We are currently using this generator in a prototype transfer-based human assisted machine translation system from English to Turkish and also integrating it to an interlingua-based machine translation system.

Acknowledgements

We thank Carnegie–Mellon University Center for Machine Translation for providing the GenKit generation system, and Lauri Karttunen of Rank Xerox Research Centre, Grenoble, for providing us RXRC Finite State Tools, with which the morphological analyzer/generator was developed. This work was in part supported by a NATO Science for Stability Project Grant TU-LANGUAGE.

References

Alshawi, H. (ed). (1992) *The Core Language Engine*. MIT Press.

Cicekli, I. 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 (EWNLG-97)*. Duisburg, Germany.

Dale, R. (1992) *Generating Referring Expressions*. MIT Press.

Dick, C. (1993) Classification Based Language Generation in Turkish. *Master's thesis*, Department of Artificial Intelligence, University of Edinburgh.

Elhadad, M. (1993) *FUF: the Universal Unifier User Manual 5.2*. Department of Computer Science, Ben Gurion University of the Negev.

Erguvanlı, E. E. (1979) The Function of Word Order in Turkish Grammar. *PhD Thesis*, University of California, Los Angeles, CA.

Hajičová, E., Sgall, P. and Skoumalová, H. (1993) Identifying topic and focus by an automatic procedure. *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, pp. 178–182.

Hakkani, D. Z. (1996) Design and Implementation of a tactical generator for Turkish, a free constituent order language. *MSc Thesis*, Bilkent University, Department of Computer Engineering and Information Science, Ankara, Turkey.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press

Hakkani, D. Z., Tür, G., Mitamura, T., Nyberg III, E. H. and Oflazer, K. (1997) Issues in Generating Turkish from Interlingua. *Technical Report CMU-LTI-97-152*, Carnegie Mellon University, Center for Machine Translation, Pittsburgh, PA.

Hoffman, B. (1995) The Computational Analysis of the Syntax and Interpretation of "Free" Word Order in Turkish. *PhD thesis*, Computer and Information Science, University of Pennsylvania.

Hoffman, B. (1995) Integrating "Free" Word Order Syntax and Information Structure. *Proceedings of the 7th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 245–252. Dublin, Ireland.

Hoffman, B. (1992) A CCG Approach to Free Word Order Languages. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pp. 245–561.

Hoffman, B. (1996) Translating into Free Word Order Languages. *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pp. 556–561. Copenhagen, Denmark.

Karttunen, L. and Kay, M. (1985) Parsing in a Free Word Order Language. In *Natural Language Parsing*, Karttunen, L. and Dowty, G (eds.), pp. 279–307. Cambridge University Press.

Korkmaz, T. (1996) Turkish Text Generation with Systemic-Functional Grammar. *Master's thesis*, Bilkent University, Department of Computer Engineering and Information Science.

Mitamura, T., Nyberg III, E. H. and Carbonell, J. (1991) An efficient interlingua translation system for multi-lingual document production. *Proceedings of the MT Summit III*, pp. 55–62. Washington, DC.

Oflazer, K. (1993) Two-level description of Turkish Morphology. *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics.* (A full version appears in *Literary and Linguistic Computing* **9**(2): 1994, 137–148.)

Reiter, E. (1995) Building natural-language generation systems. *Proceedings of AI in Patient Education Workshop*. Glasgow, Scotland.

Steedman, M. (1991) Structure and intonation. *Language* **61**: 523–568.

Steinberger, R. (1994) Treating Free Word Order in Machine Translation. *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pp. 69–75. Kyoto, Japan.

Styś, M. E. and Zemke, S. S. (1995) Incorporating discourse aspects in English-Polish MT: Towards robust implementation. *Proceedings of the Conference on Recent Advances in Natural Language Processing*.

Takeda, K., Uramoto, N., Nasukawa, T. and Tsutsumi, T. (1991) Shalt2 – A Symmetric Machine Translation System with Conceptual Transfer. IBM Research, Tokyo Research Laboratory, Tokyo, Japan.

Tomita, M. and Nyberg III, E. H. (1988) *Generation kit and transformation kit, version 3.2*, user's manual. Carnegie Mellon University, Center for Machine Translation, Pittsburgh, PA.

Turhan, C. K. (1997) An English to Turkish machine translation system using structural mapping. *Proceedings of 5th ACL Conference on Applied Natural Language Processing*, pp. 320–323. Washington, DC.

Vallduví, E. (1994) The dynamics of information packaging. In *Integrating Information Structure into Constraint-based and Categorial Approaches*, Engdahl, E. (ed.), Esprit Basic Research Project 6852, DYANA-2, Report R1.3.B.

VanNoord, G. (1990) An overview of head-driven bottom-up generation. In *Current Research in Natural Language Generation*, Dale, R., Mellish, C. S. and Zock, M. (eds.), pp. 141–165. Academic Press.

https://doi.org/10.1017/S135132499800196X Published online by Cambridge University Press