ISTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

LFG FOR TURKISH POINT-IN-TIME EXPRESSIONS

M.Sc. Thesis by
Tuba GÜMÜŞ, B.Sc.

Department : Computer Engineering
Programme : Computer Engineering

Supervisor : Prof. Dr. Eşref ADALI

JANUARY 2007

Acknowledgements

This thesis emerged from my research at IMS (Institut für Maschinelle Sprachverarbeitung), University of Stuttgart, in the context of ERASMUS exchange programme. First, I would like to thank ITU (Istanbul Technical University) for supporting me during my stay in Germany. I would also like to thank IMS, particularly Prof. Dr. Christian Rohrer, head of the department of applied computational linguistics and HD Dr. Ulrich Heid for providing me a nice working environment.

I wish to extend my thanks to Martin Forst, responsible for grammar development at IMS, for helpful discussions and suggestions on the representation of the grammar.

I am especially grateful to HD Dr. Ulrich Heid for reading and commenting on draft versions of this thesis.

I want to express my gratitude to Prof. Dr. Kemal Oflazer for sharing his knowledge and experience.

I owe special thanks to my supervisor, Prof. Dr. Eşref Adalı for his special interest in Turkish language. There would not be an NLP research team in ITU without him.

Finally, I want to thank my family and my dear friends Tuba and Canan for always being there for me.

December, 2006 Tuba Gümüş

Table of Contents

List of Tables

List of Figures

LFG FOR TURKISH POINT-IN-TIME EXPRESSIONS

SUMMARY

In computational linguistics, many applications (e.g. machine translation) depend on an exact syntactic analysis of language. Towards a high quality Machine Translation, researchers in industrial and academic institutions around the world joint to produce collaboratively written wide coverage computational grammars for Arabic, Chinese, English, French, German, Hungarian, Japanese, Malagasy, Norwegian, Turkish, Urdu, Vietnamese, and Welsh. These are written within the linguistic framework of LFG with a commonly-agreed-upon set of grammatical features. The Parallel Grammar (ParGram) project which uses the XLE (Xerox Linguistic Environment) parser and the grammar development platform aims to produce parallel analyses for similar constructions in each language, where possible.

This work presents the analysis and the implementation of a date-time grammar for Turkish Lexical Functional Grammar (LFG) which may be considered as a contribution to the development of the large scale Turkish LFG grammar in the context of the Parallel Grammar Project (ParGram). The scope of the date-time grammar is restricted to the answer of “when” questions, i.e. points in time expressions, in particular to the clock-time, days of the week, calendar dates and seasons. Some general phrases are also addressed.

When analysing Turkish sentences, there are cases where the type of the adjunct (e.g. temporal, locational etc.) has to be distinguished for syntactic reasons. Besides, temporal information of an event or a state may specifically need to be extracted from a given text (e.g. question answering). These require such information to be marked as “temporal”.

The goal of this study is to produce the syntactic analyses of Turkish point-in-time expressions with rather more details of semantic distinction. Along with a corpus examination, a large number of sample phrases have been collected and a separate set of phrase structure rules have been written where point-in-time expressions are marked.

TÜRKÇE ZAMAN İFADELERİ İÇİN SÖZCÜKSEL İŞLEVSEL GRAMER GELİŞTİRİLMESİ

ÖZET

Bilişimsel dilbiliminde, makine çevirisi gibi uygulamalar dilin söz diziminin doğru bir şekilde çözümlenmesini öngörmektedir. Yüksek kaliteli bir makine çevirisini hedefleyen Paralel Gramerler (ParGram) projesi için farklı ülkelerin akademik ve endüstriyel kuruluşlarındaki araştırmacılar Almanca, Arapça, Çince, Fransızca, Ingilizce, Japonca, Macarca, Norveççe, Türkçe, Urduca ve Vietnamca için geniş kapsamlı gramerler yazmaktadır. Bu gramerler üzerinde hemfikir olunan dilbilgisel özellikler ile Sözcüksel Işlevsel Gramer (LFG) biçiminde yazılmaktadırlar. Xerox tarafından geliştirilmiş XLE ayrıştırıcısı ve geliştirme ortamını kullanan ParGram projesi mümkün olan durumlarda benzer yapılar için paralel çözümler üretmeyi amaçlamaktadır.

ParGram projesi dahilinde geliştirilmekte olan Türkçe için geniş kapsamlı sözcüksel işlevsel gramere katkı sağlaması amacıyla yapılan bu çalışmada “ne zaman?” sorusunu cevaplayan ifadelerin çözümlenmesi anlatılmaktadır. Gerçeklenen gramerde saat ifadeleri, haftanın günleri, tarih ve mevsim ifadeleri ile diğer bazı genel ifadeler ele alınmıştır.

Türkçe cümlelerin sözdizimsel olarak doğru çözümlenmesi için yapılan eylemi tamamlayan tümlecin türünün (zaman, yer vs.) belirlenmesini gerektiren haller vardır. Bunun yanısıra soru-cevaplama gibi bazı uygulamalarda, verilen metinden durum ya da eylemin gerçekleştiği zaman bilgisinin çıkarılması gerekebilir. Bunlar, ancak ilgili bilginin "zaman" ifadesi olduğunun önceden işaretlenmesi ile mümkündür.

Bu çalışma, Türkçede "zamanda yer" bildiren ifadelerin sentaktik çözümünü Sözcüksel Işlevsel Gramer üzerinden anlamsal ayırt edicilerle işaretleyerek üretmektedir. Türkçe derlem üzerinde yapılan detaylı bir çalışma ile yukarıda sınıflanan ifadeler toplanmış ve bu ifadelerin işaretlendiği ayrı bir gramer geliştirilmiştir.

1 INTRODUCTION

In computational linguistics, grammar development is one of the most interesting research areas. Once implemented, those grammars can be used to verify theoretic assumptions about natural languages. Furthermore, many applications, e.g. machine translation in computational linguistics, depend on an exact syntactic analysis of language.

Research in NLP has established high level syntactic formalisms which allow for specification and processing of declarative, reversible grammars that assign rich structures to natural language sentences. The LFG formalism assumes two levels of representation: c(onstituent)-structure encodes phrasal dominance and precedence relations; f(unctional)-structure encodes syntactic information in terms of morphosyntactic features (NUM, TENSE etc.) and functional relations between predicates and their arguments (OBJ, SUBJ etc.). The separation between the surface oriented c-structure and the more abstract representation of functional syntactic properties makes it possible to provide syntactic descriptions for typologically diverse languages which differ radically in terms of their c-structure properties, while relating them to the level of functional representation which encodes functional syntactic properties that are shared across typologically distinct languages. This makes the f-structure representation provided by LFG-based analysis attractive for multilingual NLP tasks, such as Machine Translation [Frank, 1999].

Towards a high quality Machine Translation, researchers in industrial and academic institutions around the world joint to produce collaboratively written wide coverage computational grammars for Arabic, Chinese, English, French, German, Hungarian, Japanese, Malagasy, Norwegian, Turkish, Urdu, Vietnamese, and Welsh. These are written within the linguistic framework of LFG with a commonly-agreed-upon set of grammatical features [King, 2006, `http://www2.parc.com/isl/groups/nltt/pargram`]. The Parallel Grammar (ParGram) project which uses the XLE (Xerox Linguistic Environment) parser and the grammar development platform aims to produce parallel analyses for similar constructions in each language, where possible [Butt et al., 2002].

This thesis presents the analysis and the implementation of a date-time grammar for Turkish LFG which may be considered as a contribution to the ongoing development of a large scale Turkish Grammar in the context of ParGram [Çetinoğlu and Oflazer, 2006a]. The scope of our date-time grammar is restricted to the answer of "when" questions, i.e. points in time expressions. Duration(how long?) and frequency(how often?) expressions are left out.

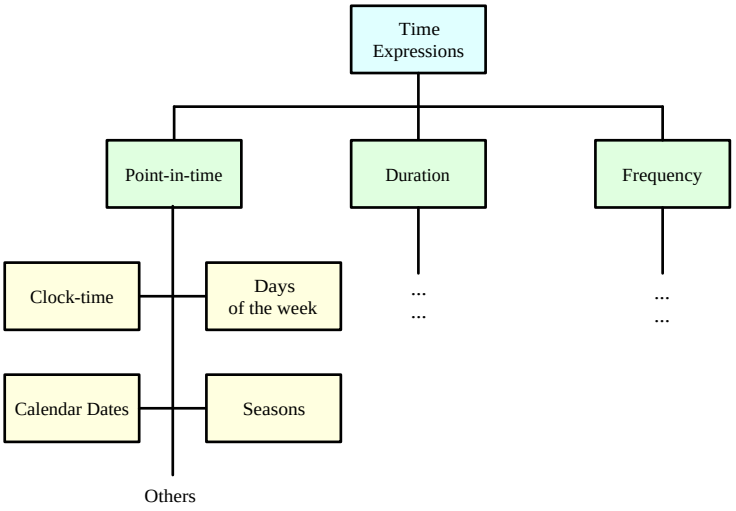


Figure 1.1: Classification of temporal expressions

When analyzing Turkish sentences there are cases where the type of an adjunct (e.g. temporal, locational etc.) has to be distinguished for syntactic reasons. Consider the cases in (1.1):

(1.1) a. *Saat* *ikide* *ulaştık.*
O'clock two-LOC arrive-PAST-A1pl
'We arrived at two o'clock.'

b. **Kapının* *önünde* *ulaştık.*
Door-GEN front-P3sg-LOC arrive-PAST-A1pl
*'We arrived in front of the door.'

Both adjuncts in (1.1) are of locative case. But the sentence is grammatical only if the type of the adjunct is "temporal".

Besides, temporal information of an event or a state may specifically need to be extracted from a given text for applications such as question answering (not really for ParGram).These require such information to be marked as “temporal”. The goal of this study is to produce syntactic analyses of Turkish point-in-time expressions with rather more details of semantic distinction (e.g. clock-time, days of the week, dates, seasons etc.). Along with a corpus examination, a large number of sample phrases have been collected and a separate set of phrase structure rules have been written where point-in-time expressions are marked up.

The outline of the thesis is as follows:

Chapter 2 of this thesis examines the underlying infrastructure. First basics of the LFG formalism are introduced. Then the grammar development environment, XLE, is presented. Recent work on Turkish LFG and the corpus work are also provided in Chapter 2.

Chapter 3 adresses the phenomenon. Turkish point in time expressions are explained and examplified in this chapter.

Chapter 4 of the thesis contains the grammar for Turkish point in time expressions presented in Chapter 3. The relevant grammar code as well as the c- and f-structures are given within the discussion of the analyses.

Chapter 5 provides some discussions and a conclusion.

2 INFRASTRUCTURE

This chapter is devoted to an overview of Lexical Functional Grammar (LFG), an overview of Xerox Linguistic Environment (XLE), an overview of the work on Turkish LFG and the corpus work.

2.1 Framework: LFG

Lexical Functional Grammar (henceforth LFG) is a unification based grammar formalism which was originally developed in the late 1970s. It was first described in detail by [Bresnan and Kaplan, 1982]. In this section readers may find the relevant features of LFG that are applied in the implementation of the Turkish date/time grammar. For a more detailed introduction to LFG, the reader is referred to Bresnan [2001]; Dalrymple [2001]; Falk [2001]; Dalrymple et al. [1995] and Sells [1985, chap. 4]. The remainder of this section is based on these resources.

LFG defines different levels of abstraction to encode syntactic and semantic information. We will represent the basic levels of representation that we make use of.

2.1.1 Levels of Representation

LFG assigns two levels of representation for every sentence of a language to encode syntactic information:

i. constituent structure

ii. functional structure

A constituent structure (c-structure), exists simultaneously with a functional structure (f-structure) representation that integrates the information from c-structure and from the lexicon.

2.1.1.1 C-Structure

C-structure is an annotated tree representation which encodes the order and the syntactic categories of the constituents of which the sentence is composed.

C-structures are generated by a generalized context-free grammar. The right-hand side of the rule is a regular expression constructing a hierarchical structure of phrasal constituents.

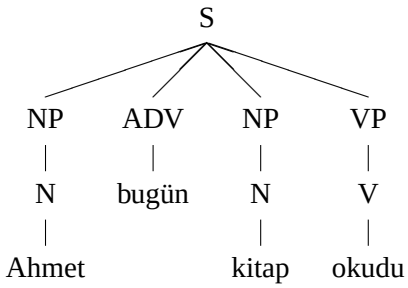
S → (NP) ADV* NP VP

According to the rule given above a sentence S can expand into an optional noun phrase (NP) followed by an adverbial (ADV). The Kleene-star denotes that zero or more adverbials can follow the optinal noun phrase. Finally, the obligatory noun phrase and verb phrase (VP) follow the adverb respectively.

The sentence given in (2.1) which can be generated by the following rules may be analyzed as an example:

(2.1) *Ahmet bugün kitap okudu.*
Ahmet today book read-PAST
'Ahmet read (a) book today'

S	→	NP	ADV	NP	VP	N	→	Ahmet
VP	→	V				ADV	→	bugün
NP	→	N				N	→	kitap
						V	→	okudu



2.1.1.2 F-Structure

F-structure is an attribute-value representation which integrates and unifies the information from the c-structure and the lexicon. Grammatical functions of the constituents, such as subject, object, adjunct etc. or features such as tense and case are encoded on this level.

The notation of f-structures consists of two columns of entries enclosed in large square brackets. The left hand column is for the attributes whereas the right hand column is for the values. Attributes and values are paired, and the members of a pair are written on the same horizontal line.

Attributes are always simple symbols, like SUBJ, PRED etc. Values, on the other hand, may be simple symbols, subordinate f-structures, or semantic forms. Semantic forms are recognizable by their single quoted values.

Below is an f-structure containing the attributes PRED with the value 'Ahmet' (a semantic form) and CASE with the value `nom` (nominative) :

$$\begin{bmatrix} \text{PRED} & \text{'Ahmet'} \\ \text{CASE} & \text{nom} \end{bmatrix}$$

PRED, being a special attribute, also encodes the subcategorization requirements. For instance, a verb like *oku* 'read' subcategorizes for two arguments: subject and object. LFG calls these "thematic arguments". In the f-structure representation, they are enclosed in angle brackets within the PRED value.

$$\begin{bmatrix} \text{PRED} & \text{'oku} \langle \text{ SUBJ, OBJ} \rangle \text{'} \end{bmatrix}$$

As mentioned, the value of an attribute may be an another f-structure. In the following example the f-structure above serves as the value of the attribute SUBJ.

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Ahmet'} \\ \text{CASE} & \text{nom} \end{bmatrix} \end{bmatrix}$$

Features with complex values are usually referred to as functions. The name “f-structure” thus can be thought as the combination of simple features and functions.

For a more complex example, the f-structure for (2.1) given in (2.2) may be considered. (cf. the c-structure example)

(2.2)
$$\begin{bmatrix} \text{PRED} & \text{‘oku } \langle \text{ SUBJ, OBJ } \rangle\text{’} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{‘Ahmet’} \\ \text{CASE} & \text{nom} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{‘kitap’} \\ \text{CASE} & \text{nom} \end{bmatrix} \\ \text{ADJUNCT} & \left\{ \begin{bmatrix} \text{PRED} & \text{‘bugün’} \end{bmatrix} \right\} \\ \text{TENSE} & \text{past} \end{bmatrix}$$

The f-structure can be read as follows: The main predicate of the sentence is *oku* ’read’ which subcategorizes for two argument functions; subject and object. *Ahmet* functions as the subject, *kitap* ’book’ as the object of the sentence. *bugün* ’today’ is an adverb functioning as an adjunct of the sentence.

In a given f-structure, a particular attribute may take place at most once (functional uniqueness will be discussed in (Section 2.1.3.1). Now, consider the sentence “*Ahmet bugün sabahleyin kitap okudu*”. *Sabahleyin* ’in the morning’ is an other adjunct modifying the main predicate. In order to represent such multiple attributes they are designed and analyzed by means of sets. The value of the ADJUNCT function is a set that may contain many f-structures as illustrated in (2.3) (also note the curly brackets in the above f-structure).

(2.3)
$$\begin{bmatrix} \text{ADJUNCT} & \left\{ \begin{bmatrix} \text{PRED} & \text{‘abc’} \end{bmatrix} \begin{bmatrix} \text{PRED} & \text{‘xyz’} \end{bmatrix} \right\} \end{bmatrix}$$

The ordering of the attributes in the f-structure is irrelevant since it is an unordered set.

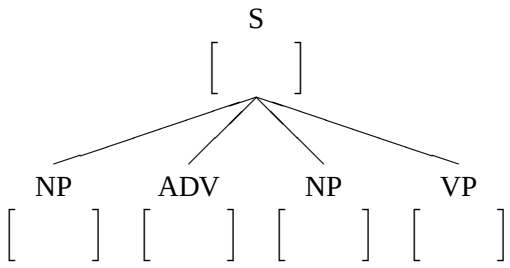
2.1.2 Mapping from c-structure to f-structure

LFG assumes that there is some f-structure associated with each node in the c-structure tree. That means, once a c-structure is built, the corresponding f-structures are built in parallel.

The mapping relation between c-structure trees and the corresponding f-structures is defined by a functional projection called "phi–(ϕ)". This projection is performed thanks to the f-structure annotations added to the c-structure rules. In other words, categories on the right hand side of the rules are associated with expressions that are known as functional schemata.

Consider the sentence 2.1, "*Ahmet bugün kitap okudu*" [1] analyzed by the rule (2.4):

(2.4) S → NP ADV NP VP



The rule 2.4 is annotated by f-structure constraints so that a link can be established between the f-structures of the nodes, cf. (2.5).

(2.5) S →	NP	ADV	NP	VP
	(↑ SUBJ)=↓	↓∈(↑ADJUNCT)	(↑ OBJ)=↓	↑=↓
	(↓ CASE)=nom		(↓ CASE)=nom	

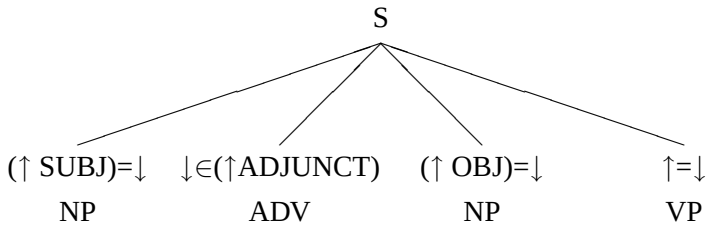
VP → V
NP → N

The ↑ and ↓ arrows are called "metavariables" and they correspond to the f-structures of the c-structure nodes. The ↑-arrow refers to the f-structure of the mother node and the ↓-arrow refers to the f-structure of the node itself.

[1] Please note that the word *kitap* 'book' is nominative. The verb *oku* 'read' can also take objects of case nominative. In accusative use of the object, it expresses definiteness; *kitabı* 'the book'.

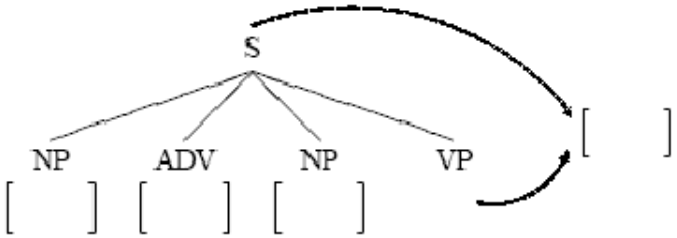
The tree in (2.6) makes it easier for the reader to find the referents for ↑ and ↓. The arrows point to the node whose f-structure they refer to.

(2.6)



The interpretation of the annotations in the tree is as follows: ↑=↓ annotation on the VP node indicates that the S node corresponds to an f-structure which is the same as the f-structure for the VP daughter. That is, the f-structure of S and VP are unified (unification).

Modifying the figure above accordingly, the pointers of both VP and S now point to the same f-structure. VP is the head of the sentence, sharing all its features:



(↑ SUBJ)=↓ annotation on the first NP node indicates that the f-structure for the S has a SUBJ attribute whose value is the f-structure for the first NP daughter.

↓∈(↑ADJUNCT) annotation on the ADV node denotes that the ADV's f-structure will be embedded under ADJUNCT as an element of a set, hence the ∈ relation.

(↑ OBJ)=↓ annotation on the second NP node says that the f-structure for the S has an OBJ attribute whose value is the f-structure for the node.

(↓ CASE)=nom annotations on NP nodes require the case of the NPs' f-structures be nominative.

Note that there is no functional annotation on the N and V nodes in (2.5). This is because there is a general convention that all preterminals are associated with ↑=↓ unless indicated otherwise. So in the NP rule N will be associated with ↑=↓ and so will be the V in the VP rule.

Finally, we reach the terminal nodes of the tree. The teminal nodes of the tree are lexical items.

In LFG notation, the syntactic features and semantic content of lexical items are determined by the schemata in lexicon entries.

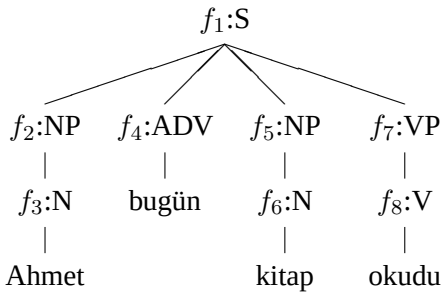
Lexical entries also use the metavariable ↑ to encode information about the f-structures of the preterminal nodes that immediately dominate them. This information is percolated up from the lexicon via ↑=↓.

The lexicon entries for the sentence 2.1 are listed in (2.7).

(2.7)	okudu	V	(↑ PRED)='oku⟨(↑ SUBJ)(↑ OBJ)⟩'
			(↑ SUBJ CASE)=nom
			(↑ SUBJ NUM)=sg
			(↑ SUBJ PERS)=3
			(↑ TENSE)=past
	Ahmet	N	(↑ PRED)='Ahmet'
			(↑ NUM)=sg
			(↑ PERS)=3
	kitap	N	(↑ PRED)='kitap'
			(↑ NUM)=sg
			(↑ PERS)=3
	bugün	ADV	(↑ PRED)='bugün'

When the lexical items that occupy the terminal nodes of the tree are inserted into an f-structure, the information contained in the lexical entry (including relevant equations) is retrieved and included in the f-structure. That is how the lexical information is combined with structural information which comes from the c-structure tree.

We now present the complete c and f-structure for the sentence 2.1. For the sake of an easier representation of the ϕ projection, f-structures will be labelled. It should be noted that the choice of a label variable for a given f-structure is completely arbitrary. For example, the root node may be labeled by f_1:S which means that this node is of category S and projects the f-structure labelled by f_1. As a consequence of the unifications some variables will be labeling the same f-structure.



$$
\begin{bmatrix}
\text{PRED} & \text{`oku} \langle\ (f_1\ \text{SUBJ})(f_1\ \text{OBJ})\ \rangle\text{'} \\
\text{SUBJ} & \underset{f_2,\,f_3}{\begin{bmatrix} \text{PRED} & \text{`Ahmet'} \\ \text{CASE} & \text{nom} \\ \text{NUM sg, PERS 3} & \end{bmatrix}} \\
\text{OBJ} & \underset{f_5,\,f_6}{\begin{bmatrix} \text{PRED} & \text{`kitap'} \\ \text{CASE} & \text{nom} \\ \text{NUM sg, PERS 3} & \end{bmatrix}} \\
\text{ADJUNCT} & \left\{ \underset{f_4}{\begin{bmatrix} \text{PRED} & \text{`bugün'} \end{bmatrix}} \right\}
\end{bmatrix}_{f_1,\,f_7,\,f_8}
$$

2.1.3 The Wellformedness Conditions on the f-structure

There are some restrictions on f-structures to achieve grammatically built sentences.

2.1.3.1 Consistency/Functional Uniqueness

In a given f-structure a particular attribute may have at most one value. Recall that this condition was stated in Section 2.1.1.2.

When f-structures are unified they share their features in such a manner that the resulting f-structure is the union of their individual features. This implies that the features of the unifying f-structures should be compatible.

Take the sentence "*Ahmet kitap okudular*''. The SUBJ NUM feature introduced by *okudular* (SUBJ NUM pl) and the NUM feature introduced by *Ahmet* (NUM sg) are not compatible where they are unified. Thus, this ungrammatical sentence is excluded violating the consistency condition.

$$\begin{bmatrix} \text{PRED} & \text{'oku} \langle \text{ SUBJ, OBJ} \rangle\text{'} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Ahmet'} \\ \text{NUM} & \textbf{sg/pl} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{'kitap'} \end{bmatrix} \end{bmatrix}$$

2.1.3.2 Completeness

An f-structure is locally complete if and only if it contains all the governable grammatical functions that its predicate governs. An f-structure is complete if and only if it and all its subsidiary f-structures are locally complete.

As an example, the incomplete sentence "*Ahmet okudu.*" 'Ahmet read.' may be considered. The predicate of the main f-structure subcategorizes for the grammatical functions SUBJ and OBJ. However, OBJ is missing from the f-structure.

$$\begin{bmatrix} \text{PRED} & \text{'oku} \langle \text{ SUBJ, } \textbf{OBJ} \rangle\text{'} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Ahmet'} \end{bmatrix} \end{bmatrix}$$

2.1.3.3 Coherence

An f-structure is locally coherent if and only if all the governable grammatical functions that it contains are governed by a local predicate. An f-structure is coherent if and only if it and all its subsidiary f-structures are locally coherent.

The sentence "*Ahmet kitap uyudu*" 'Ahmet slept (a) book' is incoherent since the OBJ *kitap* 'book' is not governed by the PRED *uyu* 'sleep'. In other words, there is a governable function in the f-structure which is not subcategorized by the local PRED *uyu*.

$$\begin{bmatrix} \text{PRED} & \text{‘uyu } \langle \text{ SUBJ } \rangle\text{’} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{‘Ahmet’} \end{bmatrix} \\ \textbf{OBJ} & \begin{bmatrix} \text{PRED} & \text{‘kitap’} \end{bmatrix} \end{bmatrix}$$

The question “What is a governable function?” might pop up at this point. The answer may only be an assumption according to the languages being processed in this framework. The governable functions usually considered in LFG are SUBJ, OBJ, OBJ2 (or OBJtheta, indirect object), OBL (oblique argument), COMP (complement), XCOMP(COMP without an overt SUBJ).

2.1.3.4 Constraining Equations vs. Existential Constraints

Equations we have been using for the annotation process are called “defining equations” and they introduce an attribute-value pair to the f-structure. There is another kind of equation called “constraining equation” that checks whether a particular value is assigned for an attribute.

$$(\downarrow \text{CASE}) =_c \text{loc}$$

The $=_c$ notation in the equation above indicates that the value for the attribute CASE that is already defined by a defining equation has to be `loc` (locative). Constraining eqution can also require an f-structure not to have a certain attribute-value pair:

$$(\uparrow \text{OBJ CASE}) \sim=_c \text{loc}$$

According to this equation, the relevant f-structure can not have an OBJ attribute having the feature CASE with the `loc` value.

Existential constraints on the other hand, check for the presence or the absence of a feature.
$(\uparrow \text{MOD})$ is satisfied only if a MOD shows up in the f-structure of the relevant predicate paired with any value.

This constraint, in fact, is already used without mentioning the name. Recall the thematic arguments (i.e. governable functions) enclosed in angle brackets for a subcategorizing verb.

Those arguments may be considered as existential constraints on the f-structure. For the transitive verb *oku* ‘read’ OBJ is an existential constraint, as well as the SUBJ.

$$\left[\text{PRED} \quad \text{‘oku}\langle\text{SUBJ, OBJ}\rangle\text{’}\right]$$

2.2 Grammar Development Environment: XLE

XLE, Xerox Linguistic Environment, is a grammar development platform for large scale grammars. It is implemented in C and uses Tcl/Tk for the user interface. It currently runs on Solaris Unix, Linux, and Mac OS X.

We do not present the user interface here nor we discuss the details of the underlying parsing techniques. For a more detailed information about XLE the reader is directed to Butt et al. [1999, chap. 11]. An XLE user manual documenting its various features including grammatical notations can also be found under Crouch et al. [2006] `http://www2.parc.com/isl/groups/nltt/xle/doc/xle_toc.html`.

We, instead, will give some information about the general architecture of the implementation which will enable the reader to see the whole picture while examining the coming sections.

2.2.1 General Notes on XLE

The grammar writer enters rules and lexicon entries in an editor (emacs has an lfg-mode, that makes some aspects of grammar writing and interacting with XLE easier).

XLE parses input sentences regarding the rules written and the lexicon entries and displays their c-structure and f-structure analyses.[2] Figure 2.1 shows the c- and f-structure displays of XLE for the sentence “*Ahmet kitap okudu*” ‘Ahmet read (a) book’.[3]

[2]Within XLE, it is also possible to generate strings from given f-structures. The generation component basically reverses the parsing process.

[3]There are two more XLE display windows; the logical choice window and the f-structure chart for the packed representation of the possible f-structures.

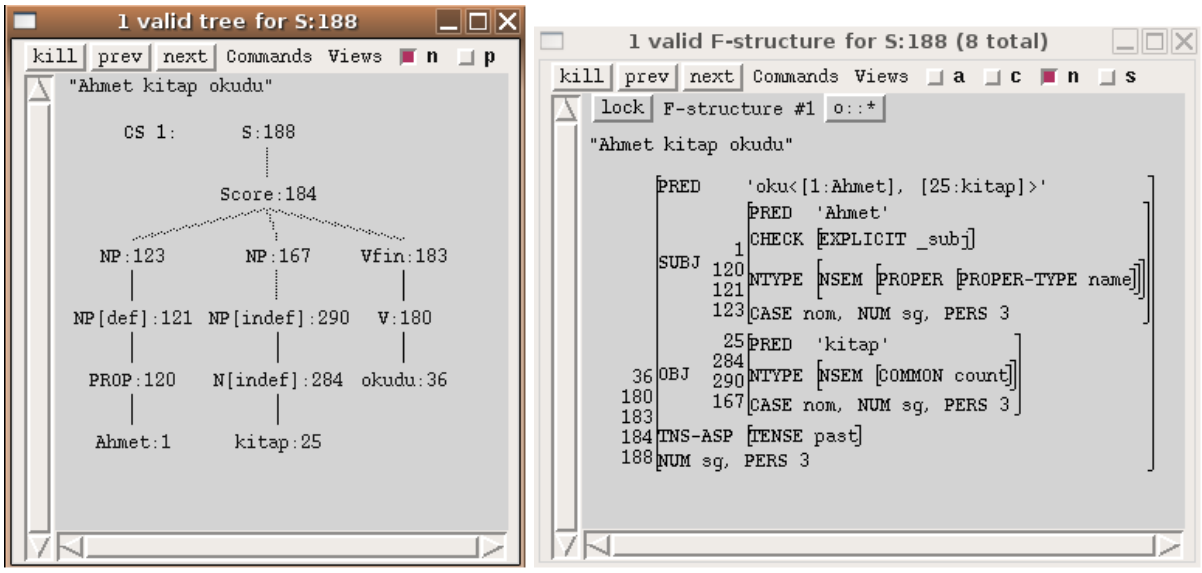


Figure 2.1: C- and f-structure displays of XLE

XLE provides different means of abstraction such as macros and templates which allows a modular grammar implementation. While writing rules, we will quite often make use of these modules hierarchically structured to encode common properties.

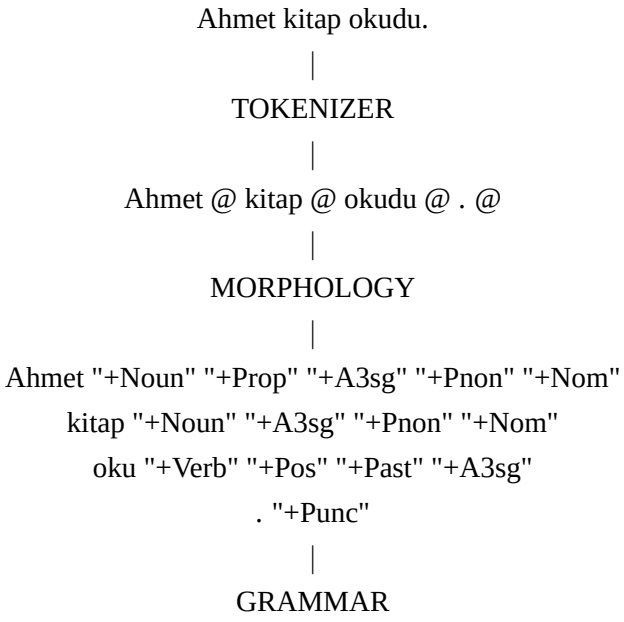
XLE enables the integration of some external modules(e.g. tokenizers, morphological analyzers and some other finite-state transducers) to perform the pre-processing steps. This integration simplifies the task of grammar development in many terms (see multiple transducers in Section 4.1.4).

Beside all, what makes XLE a suitable software for the development of large-scale grammars is the reasonable amount of time even for parsing complex sentences.

Having many advantages by the means of maintainability, modularity and time efficiency Parallel Grammar Project grammar development efforts are realized on XLE.

2.2.2 Components of XLE

Before giving the input string to the grammar to parse, some pre-processing steps are required. In order to illustrate the pre-processing steps we again take the sentence *"Ahmet kitap okudu"*:



We now analyze the flow of the process in the components.

2.2.2.1 Tokenizer

First of all, the tokenizer component segments the input stream into an ordered sequence of delimited tokens. A cascade of tokenizers and staplers may be used together to achieve the required alternation of words within a sentence regarding multiword expressions.

We currently use a single English tokenizer for the tokenization process for Turkish.

2.2.2.2 Morphological Analyzer

Tokens are the input to the morphological analyzer. The morphological analyzer associates the given token with a canonical form and a set of tags that encode relevant morphogical features.The output of the morphological analyzer is then passed to the grammar.

In the development of a large scale LFG for Turkish the Xerox two-level finite-state morphological analyzer is employed [Oflazer, 1994]. In this two level representation, the first level presents the surface form of the word, while the second contains the lemma with a list of morphemes (tags). Tags encode the part of speeches, derivational information (if there is any) and the inflectional information.

Consider the word *kitaplar* 'books'. The full-form lexicon entry for this surface word would be given by the c-structure category N and the associated f-structure constraints as in (2.8):

(2.8) kitaplar N (↑ PRED)='kitap'
(↑ NUM)=pl
(↑ PERS)=A3
(↑ CASE)=nom
(↑ POSS)=none

The morphological output of *kitaplar* 'books' on the other hand is given in (2.9):

(2.9) kitap+Noun+A3pl+Pnon+Nom

where kitap is the lemma, +Noun is the major part-of-speech, and the rest is the inflectional information: +A3pl (plural), +Pnon (non-possesive) +Nom (nominative).

Comparing (2.8) and (2.9) we can conclude that c- and f-structure constraints are given as tags by the morphogical analyzer.

The integration of the finite-state morphological analyzer into the grammar allows there to be a single lexical entry for the stem form of the word, avoiding having to list all the surface forms. The association between surface forms and stems plus tags is exteremely useful even in a language like English which has relatively impoverished morphology. The more inflectional morphology that a language has, the more useful such morphologies are. Agglutinative languages like Turkish make use of the morphological analyzer not only for the inflectional information, but also for derivational information. The representation of the derived word forms will be discussed in Section 2.3.1.

2.2.2.3 Grammar

The lexical entries represent the interface between the words and the preterminal categories of the grammar (e.g N). Since we now make use of the finite-state morphology, the ouput of the morphology has to be integrated into the grammar. It is only after then the grammar can build the relevant phrases and assignes the required f-structures.

The tags produced by the morphology are treated like any other lexical item: they have lexical entries and are placed in the structure via the c-structure rules.

So, for the surface word *kitaplar* 'books' the lemma *kitap* 'book' and the morphological tags are introduced to the lexicon in the same fashion; the lemma (or tag), its category, morphcode (items recognized by the XLE morphology will have an XLE morphcode, others *) and the relevant schema (equations or template calls).

kitap	N	XLE	@(NOUN kitap) @(COMMON count).
+Noun	N_SFX	XLE.	
+A3pl	NUM_PERS_SFX	XLE	@(NUM pl) @(PERS 3).
+Pnon	POSS_SFX	XLE.	
+Nom	CASE_SFX	XLE	@(CASE nom).

Given the sequence of stems and tags, the grammar starts the parsing process firstly by using **sublexical rules**.

Sublexical rules are special types of rules that are similar to ordinary context-free phrase structure rules in LFG, but expand only to sublexical categories (e.g N, N_SFX).

The sublexical rules build the interface between the tags like +Noun and the preterminals like N with the existence of lexicon entries for morphological tags [Kaplan and Newman, 1997].

The sublexical rule for parsing the nouns in Turkish is given in (2.10):

(2.10) N → N_BASE
N_SFX_BASE
NUM_PERS_SFX_BASE
POSS_SFX_BASE
CASE_SFX_BASE.

As seen in (2.10), sublexical rules append the suffix '_BASE ' to the sublexical categories. That enables XLE to distinguish morphological and syntactic categories. Otherwise the sublexical rules would turn into recursive, which is not intended [Butt et al., 1999].

XLE default is not displaying sublexical information. Thus, terminals of the tree are not the stem and tags, but the derived/inflected tokens. The c-structure display with sublexical information for the word *kitaplar* 'books' and the corresponding f-structure are given in Figure 2.2 and Figure 2.3 respectively. Note that full-form entry of kitaplar projects the same f-structure.

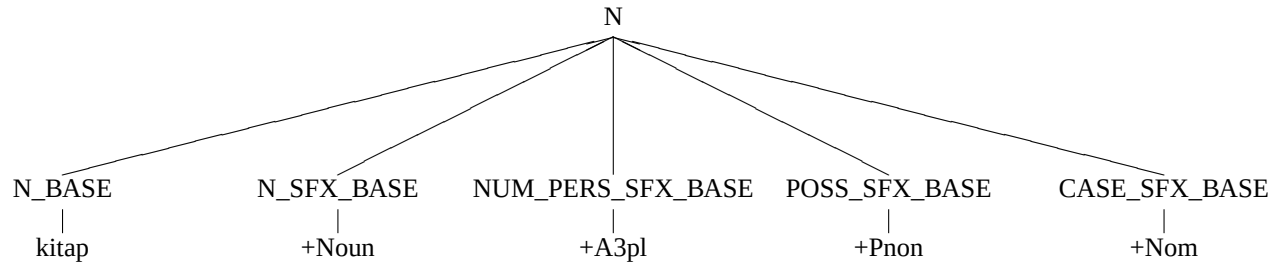


Figure 2.2: C-structure of the noun *kitaplar*

$$\begin{bmatrix} \text{PRED} & \text{'kitap'} \\ \text{NUM} & \text{pl} \\ \text{PERS} & 3 \\ \text{CASE} & \text{nom} \end{bmatrix}$$

Figure 2.3: F-structure of the noun *kitaplar*

Finally, the phrase structure rules of the grammar are entered to the XLE. The format for a simple rule is as the following:

Category --→ Category1: Schema1_1 Schema1_2 etc ;
Category2: Schema2_1 Schema2_2 etc ;
Etc.

Annotated c-structure rules are transferred to the grammar regarding this format. Below is an example:

S →	NP	ADV	NP	VP
	(↑ SUBJ)=↓	↓∈(↑ADJUNCT)	(↑ OBJ)=↓	↑=↓
	(↓ CASE)=nom		(↓ CASE)=nom	

S ⇢ NP : (^ SUBJ)=! (! CASE)=nom;
ADV : ! $ (^ ADJUNCT);
NP : (^ OBJ)= ! (! CASE)=nom;
VP : ^ = ! .

Note that "^" is used for the LFG up-arrow (↑) and "!" is used for the LFG down-arrow (↓).

2.3 LFG Work for Turkish: State of the Art

Turkish has two important aspects from linguistic point of view. First, the constituent order at the syntax level is rather free. Although the unmarked constituent order is Subject-Object-Verb, the order of the constituents may vary freely within the sentence according to discourse requirements. Second, being an agglutinative language Turkish has a complex inflectional and derivational morphology.

Turkish word forms consist of morphemes affixed to the root or to other morphemes [Oflazer, 1994]. In other words, Turkish words may involve multiple derivations as a result of productive derivational phenomena. The derived adjective *sağlamlaştırdığımızdaki* 'existing) at the time we caused (something) to become strong' is given in (2.11) as a sequence of the root word and the morphemes:

(2.11) sağlam+lAş+DHr+DHk+HmHz+DA+ki[4]

sağlam	Adjective	'strong'
sağlam-laş	Verb	'to become strong'
sağlamlaş-tırdığımızda	Noun	'at the time we caused (something) to become strong'
sağlamlaştırdığımızda-ki	Adjective	'(existing) at the time we caused (something) to become strong'

Ignoring the derivations within the word and simply marking it as an adjective would also ignore the fact that the stem is an adjective which probably has syntactic relations with preceeding words such as an adverbial modifier, or that there is an intermediate causative verb which may have an object NP to its left [Oflazer et al., 2003].

LFGs for languages like English can handle derivational processes in the lexicon by single entries since the number for such derived forms would be reasonable. However, the number of the words one can derive from a Turkish root is said to be in the millions [Hankamer, 1989]. Thus, an efficient mechanism for lexical representations of productive derivations is needed.

2.3.1 Recent Work for Turkish LFG

Güngördü and Oflazer [1994] present the first effort for parsing Turkish using the LFG formalism. This work is based on pseudo-unification and covers structurally simple and complex Turkish sentences with respect to the free constituent order phenomena. However, the way it deals with multiple nested derivations makes it quite complicated to access the relevant form of the word while writing the grammar rules. In order to provide an easier access to any derived form within a word, a Turkish word is assumed to be a sequence of inflectional groups (IGs hereafter), which in fact, determine the syntactic relations between words [Oflazer, 2003]. The general form of the representation is given in (2.12):

(2.12) root+IG_1^DB+IG_2^DB+...^DB+IG_n

[4]The surface realizations of the morphemes are conditioned by morphophonemic processes such as vowel harmony, vowel and consonant elisions. For a uniform representation A ={a,e}, H = {ı, i, u, ü}, D = {d, t} [Oflazer, 1994] (cf. *gerçek-leş-tir-diğimizde-ki*: gerçek+lAş+DHr+DHk+HmHz+DA+ki)

In (2.12) each IG_i denotes relevant inflectional features including the part-of-speech for the root while ^DBs denote the derivation boundaries. The derived adjective given in (2.11) now can be represented as in (2.13)[5]:

(2.13)

sağlam('strong')+Adj	root+IG_1
^DB+Verb+Become	^DB+IG_2
^DB+Verb+Caus+Pos	^DB+IG_3
^DB+Noun+PastPart+A3sg+P3sg+Loc	^DB+IG_4
^DB+Adj+Rel	^DB+IG_5

IGs in LFG

The ongoing work for Turkish LFG in the context of ParGram uses these inflectional groups to represent the sublexical units of a word [Çetinoğlu and Oflazer, 2006a]. In this way, grammar rules deal neither with complex derived words nor with every single inflectional morpheme, but with IGs.

Each IG is considered as a separate node in the c-structure. The node containing the IG with the root of the word is labeled as the relevant syntactic category. Other nodes that contain only IGs after DBs are given the name DS, indicating derivational suffix. A similar approach is used by Butt and King [2005] to handle morphological causative in Urdu. They treat the causative morpheme as a separate constituent in c-structure rules. However, Çetinoğlu and Oflazer [2006b] generalize this approach to all derived suffixes, instead of giving the node the name of a particular morpheme.

Consider the noun phrase given in (2.14), taken from [Çetinoğlu and Oflazer, 2006b]:

(2.14)

mavi	*renkli*	*elbiselideki*	*kitap*
blue	color-WITH	dress-WITH-LOC-REL	book

'the book on the one with the blue colored dress'

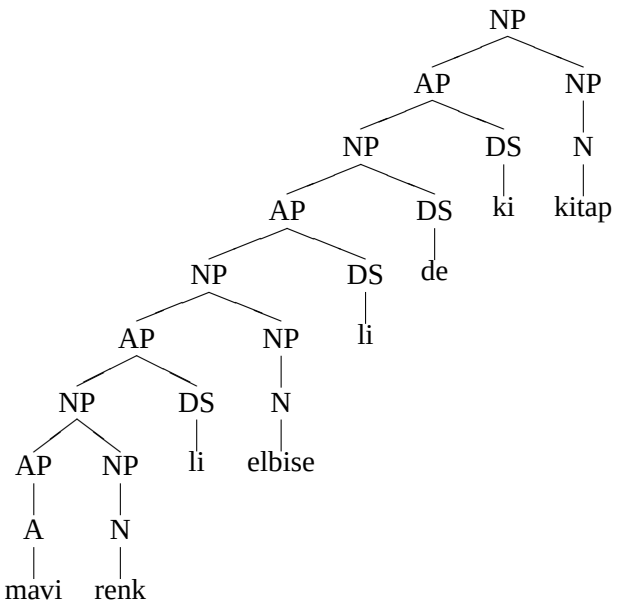
Morphogical analyses of the words composing the phrase 2.14 is given in (2.15).

[5]+Become: become verb,+Caus: causative verb, PastPart: derived past participle, P3sg: third-person singular possessive agreement, A3sg: third-person singular number-person agreement, +Zero: zero derivation with no overt morpheme, +Pnon: no possessive agreement, +Loc: locative case, +Pos: positive polarity.

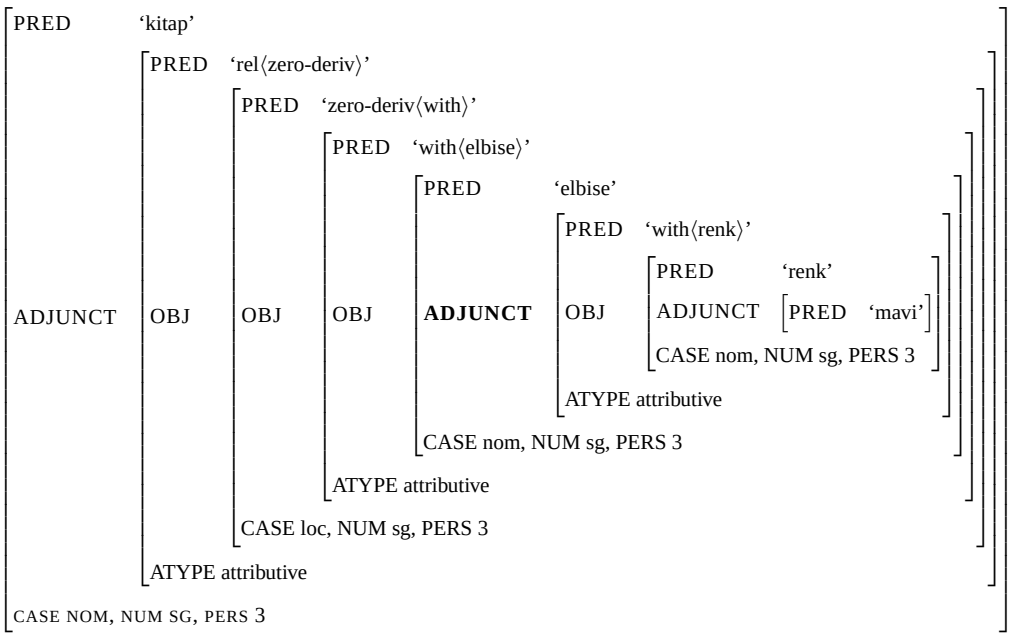
(2.15) mavi : mavi+Adj
renkli : renk+Noun+A3sg+Pnon+Nom^**DB+Adj+With**
elbiselideki : elbise+Noun+A3sg+Pnon+Nom^**DB+Adj+With^DB+Noun+Zero** +A3sg+Pnon+Loc^**DB+ Adj+ Rel**
kitap :kitap+Noun+A3sg+Pnon+Nom

C-structure of the noun phrase 2.14 and the corresponding f-structure are given in (2.16) and (2.17) respectively.

(2.16)



(2.17)



As one can easily see from the c-structure, derived phrases are represented as binary trees where the left daughter is the stem of the phrase and the right daughter is the DS[6]. Since the adjective *mavi* 'blue' modifies the noun *renk* 'color' the derivational suffix *-li* belongs to the phrase *mavi renk* 'blue color' even though it seems to be attached to *renk* 'color'. Hence the required meaning is "with blue color" instead of "blue with color". Employing IGs enables building these dependency relations within words. The first rule analyzes the phrase *mavi renk* 'blue color' as a noun phrase. A second rule can append the suffix *-li* only if the stem is a noun phrase. That is how the phrase *mavi renkli elbise* could be constructed as 'blue colored dress'. The leftmost branch in (2.16) and the innermost level in (2.17) show this dependency relation.

Whenever a phrase is derived (i.e ^DB is encountered) the DS of the derived phrase becomes the head predicate of the f-structure constructed on the fly and it subcategorizes for an OBJ which is the stem of the derived phrase.

Being a PRED value a DS is expected to have a semantic representation. The suffix *-li* in the above example 2.14 is represented by the semantic form 'with <stem>'. That is, the derivational suffix *-li* gives its additional meaning 'with' to the stem. (Similarly, the suffix *-ki* is given in the semantic form 'rel <stem>' where rel stands for relative). However, some suffixes may bring in different meanings according to the stem they are attached to. This representation turns out to be inconsistent in those cases. As an example take the suffix *-lik* (+Ness):

(2.18)

tembel / tembel-lik	lazy / laziness	habit of being . . .
kalem / kalem-lik	pencil / pencil box	place to keep . . .
anne / anne-lik	mother /motherhood	. . . ship / . . . hood
bakan / bakan-lık	minister / ministry	where . . . works

tembellik	: tembel**+Adj**^DB+Noun**+Ness**+A3sg+Pnon+Nom
kalemlik	: kalem**+Noun**+A3sg+Pnon+Nom^DB+Noun**+Ness**+A3sg+Pnon+Nom
annelik	: anne**+Noun**+A3sg+Pnon+Nom^DB+Noun**+Ness**+A3sg+Pnon+Nom
bakanlık	: bakan**+Noun**+A3sg+Pnon+Nom^DB+Noun**+Ness**+A3sg+Pnon+Nom

[6]Note that derivational morphemes are posited in separate nodes in c-structures which is not consistent with the lexical integrity principle of LFG [Dalrymple, 2001]. The issue is currently being discussed within the LFG community.

The same suffix (*-lik*) may also derive adjectives labeled with +FitFor tag by the morphological analyzer:

santimetre / 20 santimetre-lik / (20 santimetre)lik cetvel	cm / 20 cm / 20 cm ruler
gün / gün-lük / günlük gazete	day / daily / daily newspaper
ben / ben-lik / benlik iş	I / fits for me / job (which) fits for me

günlük : gün+Noun+A3sg+Pnon+Nom^DB**+Adj+FitFor**
benlik : ben+Noun+A3sg+Pnon+Nom^DB**+Adj+FitFor**

It is obvious from (2.18) that the assignment of a suffix to a semantic form is not only a matter of determining the part-of-speech of the stem (cf. laziness–motherhood).

Another issue to be solved in the scope of semantic representation of derivational suffixes involves zero derivation. As the name of the derivation already tells, the +Zero suffix is used for the derivations induced by a covert suffix. In (2.17), the f-structure with the PRED value 'zero-deriv<stem>' indicates that the subcategorized OBJ changes part-of-speech due to the locative case suffix *-de*. Even though zero derivation does not add any semantics to the stem, it is still required since an adjective can not be inflected. Therefore, the adjective *elbiseli* 'the one with the dress' first becomes a noun by +Zero then gets inflected by +Loc ; *elbiselide* 'at the one with the dress'. Due to the fact that there is no explicit change in the word, the effect of +Zero might be reflected by a feature-value pair of the relevant f-structure. Nevertheless, this assignment is also complicated for two reasons: Firstly, the +Zero suffix is not always attached to the the same part-of-speech. Secondly, the +Zero suffix does not derive the same part-of-speech every time:

sıralar / sıralar / sıralarında	lists / listing (algorithm) / around
üs / üstü	up / was up

sıralarında : sırala+Verb+Pos+Aor^DB+Adj+Zero^DB+Noun+Zero+A3sg+P3sg+Loc
üstü : üs+Noun+A3sg+Pnon+Nom^DB+Verb+Zero+Past+A3sg

In [Çetinoğlu and Oflazer, 2006b] the semantic represenation of a derivational suffix is hence not determined in detail.

2.3.2 Joint Work for Turkish LFG

Since this work is intended to be a contribution to the development of a large-scale LFG for Turkish, the existing grammar by Çetinoğlu and Oflazer [2006a] is examined, already defined features and templates are employed for the sake of consistency. In our implementation of point-in-time expressions for Turkish LFG we also make use of some NP rules. The reader is informed that we will not put an explicit reference every time we refer to these NP rules.

2.4 Corpus Work

In principle, any collection of more than one text can be called a corpus [McEnery and Wilson, 2001]. But, the term "corpus" (plural: corpora) when used in the context of modern linguistics, may be defined as a large collection of naturally occuring language text chosen to characterize a state or a variety of a language [Sinclair, 1991].

Corpus research has become a key element of all language studies. The importance of corpora to language study is aligned to the importance of empirical data. Empirical data enable the linguist to make objective statements, rather than those which are based on the individual's own internalised cognitive perception of language [McEnery and Wilson, 2001].

The availability of large corpora has made it easier to study the language as it is actually performed by people. Corpora can be used as a repository of examples to expond, test or exemplify given theoretical statements. Tognini-Bonelli [2001] and Sinclair [1991] call this procedure "corpus-based". They build a dichotomy between the corpus based and the corpus driven methodology, where the latter would consist in deriving theoretical statements automatically from the presence of corpus evidence.

In the development of the Turkish date/time grammar we also make use of corpus data. Instead of first devising the grammar by reference to introspection and then modifying where it fails to give a proper analysis, we design our date-time grammar along with an observation on a reference corpus (Section 2.4.1). We write specific rules for the date-time expressions which are additions to the existing rules. In this sense our work may be considered as "corpus-driven".

In the following sections, we will first introduce the corpus we use (METU Turkish) and then the search tool (IMS CWB) with which we extract data from the corpus.

2.4.1 Reference Corpus: METU Turkish Corpus

METU (Middle East Technical University) Turkish Corpus [Say et al., 2002] is a 1.200.000 word collection of 520 samples from 291 written texts from various genres (novels, interview, biography, articles, etc.). Sample size is set to 2000 words and no author is allowed to have more than five publications in the corpus. So, it may be considered as a general balanced corpus. It is synchronic (built of present, post-1990, Turkish). Although it lacks a spoken languge component (consequently, a few constructions typical of spoken Turkish are not represented) the METU corpus is sufficient to generalize the common use of date/time expressions.

The METU Turkish Corpus is freely available for academic research and comes with a query workbench. However, in this query processing tool, boolean queries are realized only over surface words at paragraph or the whole sample level. For instance, to retrieve information where *the names of the days of the week preceed the word gün 'day' in any inflected form within a sentence* is not possible. Hence we needed a more advanced query processing tool.

2.4.2 Corpus Query Processing: IMS Corpus Workbench

The IMS Corpus Workbench [Christ, 1994] is a collection of tools for encoding, indexing, compression, decoding, and querying large text corpora (100 M words and more) with linguistic annotations. It provides fast access to the corpus by binary encoding and fast look-up of word forms and annotations by full indexing.

The CWB input format is one-token-per-line, with annotations (pos, lemma) given as additional tab-separated columns. XML tags must appear on separate lines. (2.19) is an example input to CWB [Evert, 2002].

(2.19)

$< s >$		
It	PP	it
was	VBD	be
an	DT	an
elephant	NN	elephant
.	SENT	.
$< /s >$		

Our reference corpus is not annotated according to the lemmas or parts of speech of the words. So we only have “word” as the “positional attribute”. But we have several structural attributes (meta-data and constituency information) since XML based XCES (Corpus Encoding Standard) is employed while building the corpus. We write two short perl scripts to prepare the corpus as an input to CWB. First script removes the tags which will not necessarily be used for our purpose. The new attribute “s” (sentence) is added so that we can query within a sentence as well as within a paragraph. The second script adds the lowercase versions of the words as a positional attribute. This will enable us to run queries regardless of the writings of the initals of the words (by the use of %c flag in the CQP query). (2.20) shows a sample input to CWB from our tagged corpus file.

(2.20)

$< s >$	
Buraya	buraya
neden	neden
geldim	geldim
?	?
$< /s >$	

After converting our corpus data into the CWB binary format we now can run queries on CQP, the query processor component of CWB [Evert, 2005].

With the regular expression syntax it is easy to search for word combinations in CQP and the results are displayed in a keyword-in-context (KWIC) index. KWIC is the most common form of concordance where the word is centered in a fixed length field. In CQP, the user can define the size of the right anf left context in terms of characters, tokens or sentences. Figure 2.4.2 is the display for the query given in (2.4) with the context size 20 characters on the right and left.

(2.21)
```
METU> set Context 20;
METU> [word="Pazartesi|Salı|Çarşamba|Perşembe|Cuma
      |Cumartesi|Pazar" %c] "gün.*" ;
```

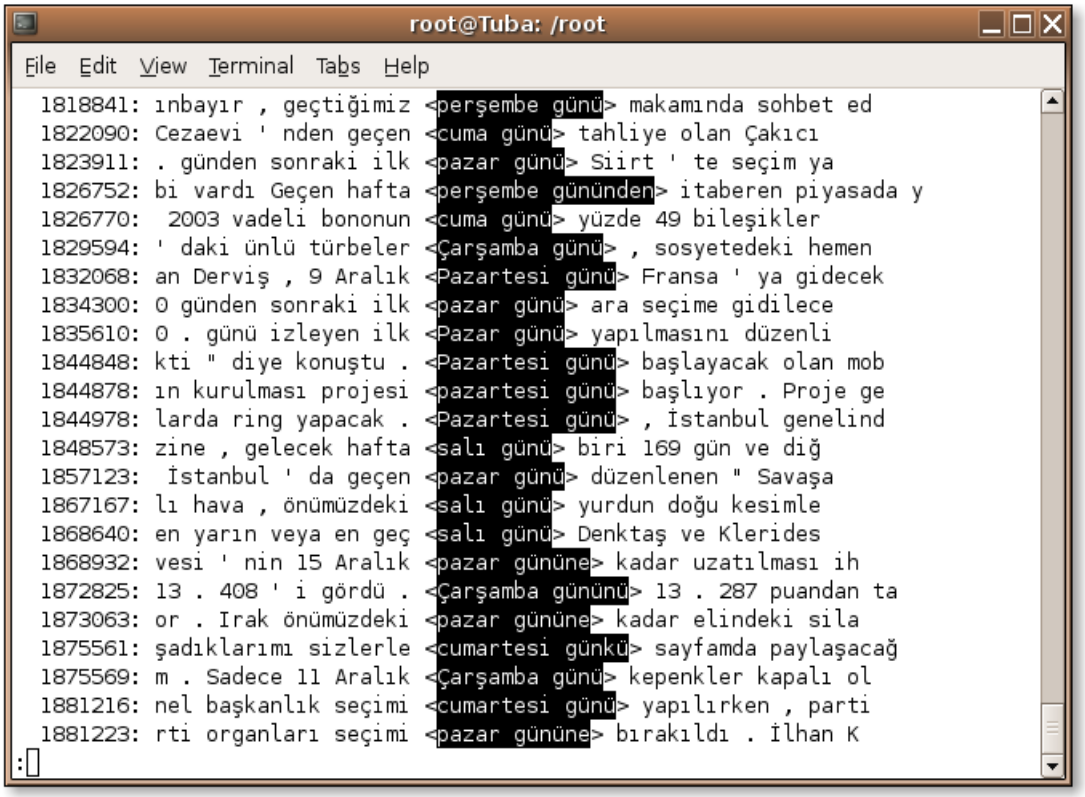


Figure 2.4: KWIC display in CQP

One can also compute frequency distribution of matching word sequences. Table 2.1 is the distribution for the query 2.22. The first coloumn shows the number of matches of the sequence given in the second coloumn. The last coloumn specifies the positions of the results in the corpus.

(2.22)
```
METU>[word="Çarşamba" %c] "gün.*";
METU>count by word;
```

Table 2.1: Frequencies of the query (2.22)

Frequency	*Expression*	*Line*
14	çarşamba günü	[#19-#32]
9	Çarşamba günü	[#3-#11]
3	çarşamba günleri	[#16-#18]
3	çarşamba gününden	[#34-#36]
2	Çarşamba günkü	[#1-#2]
2	çarşamba günkü	[#14-#15]
1	ÇARŞAMBA günü	[#0]
1	Çarşamba gününü	[#12]
1	Çarşamba gününün	[#13]
1	çarşamba günüdür	[#33]
1	çarşamba gününe	[#37]

3 TURKISH POINT-IN-TIME EXPRESSIONS

Point in time expressions are the answer of “when” questions. Clock time (i.e times-of-day), calendar dates, days of the weeks, seasons and some other general expressions fall into this class. In this chapter, we will introduce a number of these expressions that function as adverbials within a sentence. This chapter is mainly based on [Göksel and Kerslake, 2005].

3.1 Clock time

The use of the word *saat* ‘hour’ is optional in all expressions of clock time and if used must precede the numeral indicating the hour. Clock-time points are expressed with the locative suffix on the numeral:

(3.1) *(saat)* *iki-de*
(o’clock) two-LOC
‘at two (o’clock)’

For the half-hours *buçuk* ‘and a half’ follows the numeral and it is to this word that the locative suffix is attached:

(3.2) *(saat)* *iki* *buçuk-ta*
(o’clock) two half-LOC
‘at half-past two (o’clock)’

“Half past twelve” may be considered as an exceptional clock-time in Turkish since the word *yarım* ‘half’ is used instead (elliptic use):

(3.3) *(saat)* *yarımda*
(o’clock) half-LOC
‘half-past twelve (o’clock)’

One also can only use *buçukta* ‘at half’ referring to the hour of the speech. In this deictic use, the word *saat* ‘hour’ is not used at all:

(3.4) *Buçuk-ta orada olurum.*
half-LOC there be-AOR-A1sg
‘I’ll be there at half.’

Time locations less then thirty minutes after the hour are expressed with the word *geçe* ‘past’ following the number of minutes which in turn follow the accusative-marked form of the hour numeral:

(3.5) *(saat) iki-yi on geçe*
(o’clock) two-ACC ten past
‘at ten past two (o’clock)’

Similarly, time locations less then thirty minutes before the hour are expressed with the word *kala* ‘to’ following the number of minutes which in this case follow the dativ-marked form of the hour numeral:

(3.6) *(saat) iki-ye on kala*
(o’clock) two-DAT ten to
‘at ten to two (o’clock)’

For 15 minutes (a quarter) the word *çeyrek* ‘quarter’ is used:

(3.7) *(saat) iki-yi çeyrek geçe*
(o’clock) two-ACC quarter past
‘at quarter past two (o’clock)’

(saat) iki-ye çeyrek kala
(o’clock) two-DAT quarter to
‘at quarter to two (o’clock)’

As an alternative to the expressions given above, the twenty-four hour clock representation may be used with the locative suffix attached to the minute numeral:

(3.8) *(saat) yirmi bir elli-de*
(o’clock) twenty one fifty-LOC
‘at 21:50 (o’clock)’

Clock-time expressions may be used with several postpositions. Below are given some of those postpositional phrases:

(3.9) *saat* *iki-ye* *doğru*
o'clock two-DAT towards
'towards two o'clock'

saat *iki-ye* *kadar*
o'clock two-DAT till
'till two o'clock'

saat *iki-den* *beri*
o'clock two-ABL since
'since two o'clock'

saat *iki-den* *itibaren*
o'clock two-ABL from-on
'from two o'clock on'

saat *iki-den* *önce*
o'clock two-ABL before
'before two o'clock'

saat *iki-den* *sonra*
o'clock two-ABL after
'after two o'clock'

Besides, there are some nouns that are used in approximate time expressions. They function as postpositions giving the meaning "around". These nouns are mostly used in plural and locative case:

(3.10) *saat* *iki* *sularında/sıralarında/dolaylarında/civar(lar)ında*
(o'clock) two around
'around two o'clock'

The noun *arasında* 'between' is also commonly used in approximate time expressions. It functions as a postposition similar to the nouns stated in (3.10):

(3.11) *saat iki ile saat üç arasında*
o'clock two and o'clock three between
'between two o'clock and three o'clock'

saat iki ile üç arasında
o'clock two and three between
'between two and three o'clock'

iki ile üç arasında
two and three between
'between two and three'

Another approximate clock-time phrase is often built with the postposition *gibi* 'about' although it is not a recommended Turkish time expression. It is stylistically criticized.

(3.12) *saat iki gibi*
(o'clock) two about
'about two o'clock'

Clock-time information may be given in a compound with the part of the day:

(3.13) *sabahın altısında*
morning-GEN six-P3sg-LOC
'at six (o'clock) in the morning'

gecenin üçünde
night-GEN three-P3sg-LOC
'at three (o'clock) at night'

Apart from all the expressions given above, a clock-time expression may be the sentence itself instead of being a complement of it:

(3.14) *Saat ikiye geliyor.*
Hour two-DAT come-PROG-A3sg
'It is almost two o'clock.'

Saat ikiyi çeyrek geçiyor.
Hour two-ACC quarter pass-PROG-A3sg
'It is quarter past two o'clock.'

Saat iki bile değildi.
Hour two even be_not-PAST-A3sg
'It was even not two o'clock.'

Analyses of these expressions, however, are beyond our scope.

3.2 Days of the Week

A time point in terms of the days of the week may point to a past or a future day(deictic use). In both cases the name of the day can simply be used either on its own or in a noun compound (indefinite noun phrase) with the noun *gün* 'day' in nominative case.

(3.15) ***Salı*** ***(günü)*** *Stuttgart'a* *geldim.*
Tuesday (day-P3sg) Stuttgart-DAT come-PAST-A1sg.
'I came to Stuttgart on Tuesday.'

(3.16) ***Salı*** ***(günü)*** *Stuttgart'a* *gidiyorum.*
Tuesday (day-P3sg) Stuttgart-DAT go-PROG-A1sg.
'I am going to Stuttgart on Tuesday.'

Note that, time location on the day is obtained without an explicit locative-case marking on the phrase *Sali günü* 'on Tuesday'. The sentence 3.17 would be ungrammatical:

(3.17) ****Salı günü-(n)de*** *Stuttgart'a gidiyorum.*

In fact, one may find some examples in corpora where the locative suffix is attached to the modified word *günü*. These expressions are preceeded by some other modifiers:

(3.18) ***Bu güzel pazar gününde*** *sizlerle birlikte olmaktan mutluyuz.*
'We are pleased to be with you on this beatiful Sunday.'

Yağmurlu bir cumartesi gününde *tanışmıştık onunla.*
'We had met on a rainy Saturday.'

However, our corpus query ($weekday [word='gününde'])for such phrases did not yield result.

When referring to (relative) future time, as an alternative to (3.16), dative-case marked form of the day may be used:

(3.19) ***Salıya*** *Stuttgart'a* *gidiyorum.*
Tuesday-DAT Stuttgart-DAT go-PROG-A1sg.
'I am going to Stuttgart on Tuesday.'

The earliness of the time expressed in terms of the days may be emphasized by the ablative case suffix added to the day:

(3.20) ***Salıdan*** *gidiyorum.*
Tuesday-ABL go-PROG-A1sg.
'I am going already on Tuesday.'

3.3 Calendar Dates

Time location involving dates usually requires the locative suffix except when the date expressing phrase is compounded with the days of the week or parts of the day. [1]

In the expression of time location in a certain year, either the numeral expressing the year gets the locative suffix or it is compounded with the word *yıl* (or the synonymous word *sene*) 'year'. The year numeral may be preceeded by *MÖ* or *MS* (*İÖ* or *İS*) which stand for *milattan önce* or *milattan sonra* (*İsadan önce* or *İsadan sonra*) 'before Christ' or 'after Christ' respectively:

(3.21) *1982'de*
1982-LOC
'in 1982'

MÖ 1982 yıl-ı-(n)da
B.C. 1982 year-P3sg-LOC
'in the year 1982 B.C.'

[1] *9 Eylül 1982 Pazar günü* 'on Sunday, September the 9th, 1982',
9 Eylül sabahı 'in the morning of September the 9th' etc.

Similarly, in the expression of location in a certain month the name of the month may be compounded with *ay* ‘month’:

(3.22) *Eylül-de*
September-LOC
‘in September’

Eylül *ay-ı-(n)da*
September month-P3sg-LOC
‘in the month of March’

Instead of giving the name of the month, the corresponding ordinal number of the month may also be compounded with *ay* ‘month’:

(3.23) *Dokuzuncu* *ay-da*
ninth month-LOC
‘in the ninth month’

Time location on an exact date is expressed by placing the cardinal number before the name of the month and adding the locative suffix to the latter:

(3.24) *9* *Eylülde*
9 September-LOC
‘on the 9th of September’

If the year is expressed as well, this follows the name of the month and the locative suffix appears on the year:

(3.25) *9* *Eylül* *1982’de*
9 September 1982’-LOC
‘on September the 9th, 1982’

The date may be expressed only in terms of the year and the month. Then, the day numeral is simply removed from the expression 3.25.

(3.26) *Eylül* *1982’de*
September 1982’-LOC
‘in September, 1982”

When the month information is not written by its name but by its reference number, "/" or "." is used to separate the day, month and year numerals in Turkish [TDK, 2006]:

(3.27) *9/9/1982'de*
9/9/1982'-LOC
'on 9/9/1982'

To express location on an exact date, date expressing phrase may be compounded with the word *gün* 'day'. Here, the location in time is provided implicitly by the word *günü*'day-P3sg':

(3.28) *9 Eylül 1982 günü*
9 September 1982'-LOC day-P3sg
'on the day 9th of September 1982'

The date expression may involve the days of the week information. Again, the location in time is provided by the word *günü* which the day name is compounded with (see also Section 3.2):

(3.29) *9.9.1982 Pazar günü*
9.9.1982 Sunday day-P3sg
'on Sunday, 9.9.1982'

The day-part of an exact date may be expressed compounding the date with the nouns *sabah* 'morning', *akşam* 'evening', *gece* 'night'. The locative suffix to these nouns is not necessarily used.

(3.30) *9 Eylül Pazar akşamı*
9 September Sunday evening-P3sg
'in the evening on Sunday, September the 9th'

A more informal way of expressing the date of an event is to use a genitive-possessive construction:

(3.31) *Eylülün dokuz-u-(n)da*
September-GEN nine-P3sg-LOC
'on the ninth of September'

Eylül ay-ı-(n)ın dokuz-u-(n)da
September month-P3sg-GEN nine-P3sg-LOC
'on the ninth of the month of September'

The date expressions may be deictic, i.e. bound to the month introduced by the context.

(3.32) *Ay-ın* *dokuz-u-(n)da*
month-GEN nine-P3sg-LOC
'on the ninth of the month'

dokuz-u-(n)da
nine-P3sg-LOC
'on the ninth'

Some other expressions are given in (3.33):

(3.33) *Eylülün* *ilk* *haftası*
September-GEN first week-P3sg
'in the first week of September'

Eylül *ayının* *ikinci* *Pazarı*
September month-P3sg-GEN second Sunday-P3sg
'on the second Sunday of the month September'

Dokuzuncu *ayın* *dokuzuncu* *günü*
ninth month-GEN ninth day-P3sg
'on the ninth day of the ninth month'

The year can be combined with the name of a month in one of two ways:

1. as a noun compound:

1982	*Eylül-ü-(n)de*	in September of 1982
1982	*Eylül ay-ı-(n)da*	in the month of September of 1982
1982 yılı	*Eylülünde*	in September of the year 1982
1982 yılı	*Eylül ayında*	in the month of September of the year 1982

2. in a genitive-possesive construction:

1982'nin	*Eylülünde*	in September of 1982
1982'nin	*Eylül ayında*	in the month of September of 1982

1982 yılının	*Eylülünde*	in September of the year 1982
1982 yılının	*Eylül ayında*	in the month of September of the year 1982

The exact date may be specified in an expression of the type exemplified above. The possible expressions are listed below:

(3.34)	*1982*	*Eylül-ü-nün dokuz-u-(n)da*	on the ninth of September of 1982
	1982'nin	*Eylülünün dokuzunda*	
	1982 yılı	*Eylülünün dokuzunda*	on the ninth of September of the year 1982
	1982 yılının	*Eylülünün dokuzunda*	
	1982	*Eylül ay-ı-nın dokuz-u-(n)da*	on the ninth of the month September of 1982
	1982'nin	*Eylül ayının dokuzunda*	
	1982 yılı	*Eylül ayının dokuzunda*	on the ninth of the month September of the year 1982
	1982 yılının	*Eylül ayının dokuzunda*	
	1982,	*dokuz Eylül-ü-(n)de*	on September the ninth of 1982
	1982'nin	*dokuz Eylülünde*	
	1982 yılı	*dokuz Eylülünde*	on September the ninth of the year 1982
	1982 yılının	*dokuz Eylülünde*	
	1982,	*dokuz Eylül Pazar (günü)*	on Sunday, September the ninth of 1982
	1982'nin	*dokuz Eylül Pazar (günü)*	
	1982 yılı	*dokuz Eylül Pazar (günü)*	on Sunday, September the ninth of the year 1982
	1982 yılının	*dokuz Eylül Pazar (günü)*	

The century information may also be considered as a calendar date expression. In the expression of time location in a certain century, the numeral expressing the century is compounded with the word *yüzyıl*(or the synonymous word *asır*) 'century'. Similar to year numerals, century numerals may be preceeded by *MÖ* (*İÖ*) 'before Christ' or *MS* (*İS*) 'after Christ'.

(3.35) *MÖ 7. yüzyılda*
B.C. 7th century-LOC
'in the 7th century B.C.'

Stating the phase of a date expression is frequently used.

(3.36) *1982 sonunda*
1982 end-P3sg-LOC
'at the end of 1982'

1980'lerin hemen başında
1980'-PL-GEN right beginning-P3sg-LOC
'right at the beginning of 1980s'

1982 yılının ortalarında
1982 year-P3sg-GEN middle-PL-P3sg-LOC
'in the midst of the year 1982 (in mid-1982)'

Eylül ayının sonlarında
September month-P3sg-GEN end-PL-P3sg-LOC
'at the end of the month September'

1982 Eylülünün sonunda
1982 September-P3sg-GEN end-P3sg-LOC
'at the end of September of 1982'

1982 yılı Eylül ayı başında
1982 year-P3sg September month-P3sg beginning-P3sg-LOC
'at the beginning of the month September of the year 1982'

İÖ dokuzuncu yüzyılın ilk yarısında
B.C. ninth century-GEN first half-P3sg-LOC
'in the first half of the ninth century B.C.'

Date-time expressions may be used with post-positions as well:

(3.37) *9 Eylül 1982'den önce*
9 September 1982'-ABL before
'before September the 9th, 1982'

1982 yılına dek
1982 year-P3sg-DAT till
'till the year 1982'

9 Eylül Pazar gününden itibaren
9 September Sunday day-P3sg-ABL from-on
‘from Sunday, September the 9th on’

Eylülün dokuzundan beri
September-GEN nine-P3sg-ABL since
‘since the ninth of September’

1982 yılı Eylül ayı sonuna kadar
1982 year-P3sg September month-P3sg end-P3sg-DAT until
‘until the end of the month September of the year 1982’

3.4 Seasons

To express temporal location *ilkbahar* (or *bahar*) ‘spring’ and *sonbahar* ‘autumn’ gets the locative case suffix whereas *yaz* ‘summer’ and *kısın* ‘winter’ have special forms: *yazın* ‘in summer’ and *kısın* ‘in winter’.

i. ***Yazın*** *öde* ***kışın*** *ısın.*
‘Pay **in summer**, get warm **in winter**.’

ii. *Düzenli olarak* ***sonbaharda*** *başlayan bu depresyon türü,* ***ilkbaharda*** *kendiliğinden kayboluyor.*
‘This type of depression regularly starts **in autumn** and naturally disappears **in spring**.’

Güz is the synonym for *sonbahar* ‘autumn’. But location on this noun is provided by the word *güzün* ‘in autumn’.

(3.38) ***Güzün*** *herkese ikişer çuval buğday dağıttılar.*
‘They gave two bags of wheat to everybody **in autumn**.’

Seasons of the year may be compounded with the word *mevsim* ‘season’:

(3.39) *Menatolin maddesinin insanda uzun süre etkili olduğu ve* ***kış mevsiminde*** *doğanların*

yaz mevsiminde *doğanlara oranla intihar etme risklerinin daha fazla olduğu bilimsel olarak kanıtlanmıştır.*
'It is proved that the substance Menatolin is effective for long term and the people who are born **in winter** are more prone to suicide than the people who are born **in summer**.'

3.5 Other Expressions

In this section we will present some other point in time expressions.

For the actual moment of the speech the adverbials *şu anda* 'at the moment' or *şimdi* 'now' can be used.

(3.40) ***Şimdi/şu anda*** *uyuyor.*
'She is sleeping **now/at the moment**.'

Şimdi 'now' can also refer to very near future or to immediate past:

(3.41) ***Şimdi*** *gelirim.*
'I'll be back **in a moment**.' *Selim Bey* ***şimdi*** *çıktı.*

'Selim Bey has just left.'

The expressions *şimdilerde*, *şu sıralar(da)* and *bu günlerde* 'nowadays/recently/lately' extend both backwards and forwards from the moment of speech and refer to a much broader period of days.

(3.42) *Tuba* ***şu sıralar(da)/bu günlerde/şimdilerde*** *tezini yazmaya çalışıyor.*
'Tuba is trying to write her thesis **nowadays**.'

When used on their own *önce* means 'first' or 'at first' and *sonra* 'then' or 'later':

(3.43) ***Önce/Evvela*** *bütün kitabı gözden geçir,* ***sonra*** *(önemli gördüğün) bölümlere geri dön.*
'**First** look over the whole book, **then** go back to the sections (you consider important).'

Both *önce* and *sonra* can be preceded by a phrase indicating an amount of time. Such

expressions are deictic expressions whose meanings therefore depend on the context in which they are used.

(3.44) ***Yirmi yıl önce/evvel*** *burası tarlaydı.*
‘**Twenty years ago** this was a field.’

Yalnız yaşıyordu. Kocası ***yirmi yıl önce*** *ölmüştü.*
‘She lived alone. Her husband had died **twenty years before**.’

Beş gün sonra *sınavım var.*
‘I have got an exam **in five days’ time/five days from now**.’

Beş gün sonra *sınava girdim.*
‘**Five days later** I took the exam.’

If used in singular and without a determiner the parts of the day nouns *sabah* ‘morning’, *öğlen* ‘afternoon’, *akşamüstü* ‘nightfall’, *akşam* ‘evening’, *gece* ‘night’, *geceyarısı* ‘midnight’ refer to the time point on which attention is already focused.

(3.45) ***Gece*** *yağmur yağmıştı, yerler ıslaktı.*
‘It had rained **during the night**, (so) the ground was wet.’

The nouns *sabah* ‘morning’, *öğle* ‘afternoon’, *akşam* ‘evening’, *gece* ‘night’ have also their adverbial forms: *sabahleyin* ‘in the morning’, *öğleyin* ‘in the afternoon’, *akşamleyin* ‘in the evening’ and *geceleyin* ‘at night’.

Dün ‘yesterday’ *bugün* ‘today’ *yarın* ‘tomorrow’ may either be used on their own or in a compound with the parts of the day:

(3.46) ***Dün*** *bir arkadaşımın doğum günüydü.*
‘Yesterday was the birthday of a friend of mine.’

Dün akşam *bir arkadaşıma mektup yazdım.*
‘I wrote a mail to a friend yesterday evening.’

The nouns *sabah* ‘morning’, *akşam* ‘evening’, *gece* ‘night’, *gün* ‘day’, *hafta* ‘week’, *ay*

'month', *yıl/sene* 'year' may be preceded by one of the following determiners/adjectives : *bir* 'a(n)', *bu* 'this', *o* 'that', *geçen* 'last', *gelecek* 'next', *önceki/evvelki/evvelsi* 'previous/last', *sonraki/önümüzdeki/ertesi* 'the next', *ilk* 'the first', *son* 'the last'.

(3.47) ***Bu sabah*** *geç uyandım.*
'I woke up late **this morning**.'

Gelecek hafta *boş vaktin var mı?*
'Do you have any free time next week?'

As mentioned for the days of the week in Section 3.2, dative case marking produces (relative) future reference for temporal expressions.

(3.48) ***Akşama*** *misafirlerimiz olacağı için yemek yapmakla meşguldük.*
'As we were expecting guests **that evening** we were busy cooking.'

Noun phrases of which the head is the plural-marked form of the time nouns such as *gün* 'day', *hafta* 'week', *ay* 'month', *yıl/sene* 'year' etc. get the locative case marking where location in time is concerned.

(3.49) *son yıllarda*
last year-PL-LOC
'in the last (few) years'

o günlerde
that day-PL-LOC
'in those years'

Locative case marking also occurs in noun phrases of which the head is a noun or noun compound denoting a period of time, e.g. *Osmanlı döneminde* 'in the Ottoman period', *çocukluğumda 'in my childhood'*.

If expressions of time location that include ablative case marking are not followed by a postposition this usually emphasize the earliness of the time referred to:

(3.50) *Ben* ***şimdi-den*** *üşüyorum.*
'I am **already** feeling cold.'

*Bavulunu **akşamdan** hazırlamıştı.*
‘She had **already** packed her suitcase **the evening before**.’

Some other fixed expressions with the ablative case marking include *erkenden* ‘early’, *önceden* ‘beforehand’, *çoktan* ‘long ago/long since’.

As for the adverbial *sonradan* ‘(only) afterwards’ the ablative case marking has the opposite effect of drawing attention to the lateness of the ocurrence of an event:

(3.51) *Onun evli olduğunu **sonra(dan)** öğrendim.*
‘It was **(only) later** that I found out he was married.’

To express the regular occurrence of an event on a certain day of the week or at a certain time of day, the suffix *-lAr-I* is added to the name of the day or to the part of the day.

(3.52) ***Çarşambaları** Almanca kursuna gidiyorum.*
‘**On Wednesdays** I am taking a German course.’

***Sabahları** mektuplarını okumadan asla güne başlamaz.*
‘He never starts the day before he reads his mails **in the mornings**.’

(3.53) lists some other adverbials expressing point-in-time:

(3.53)*demin*	‘a few moments ago, just now’
geçende	‘the other day’
geçenlerde	‘a few days ago’
yakında	‘soon’
eskiden/zamanında/vaktiyle	‘in the past, formerly’
hemen/derhal	‘immediately’
artık	‘now/no longer’
henüz	‘yet/just’
yeni	‘just’
daha	‘still/yet/just’
neredeyse	‘almost’

4 LFG FOR TURKISH POINT-IN-TIME EXPRESSIONS

4.1 General Structure

In section 2.2 we presented the development environment that we use for parsing: XLE. For a more concrete representation we give the system architecture of XLE as a chart [Butt et al., 1999] here.

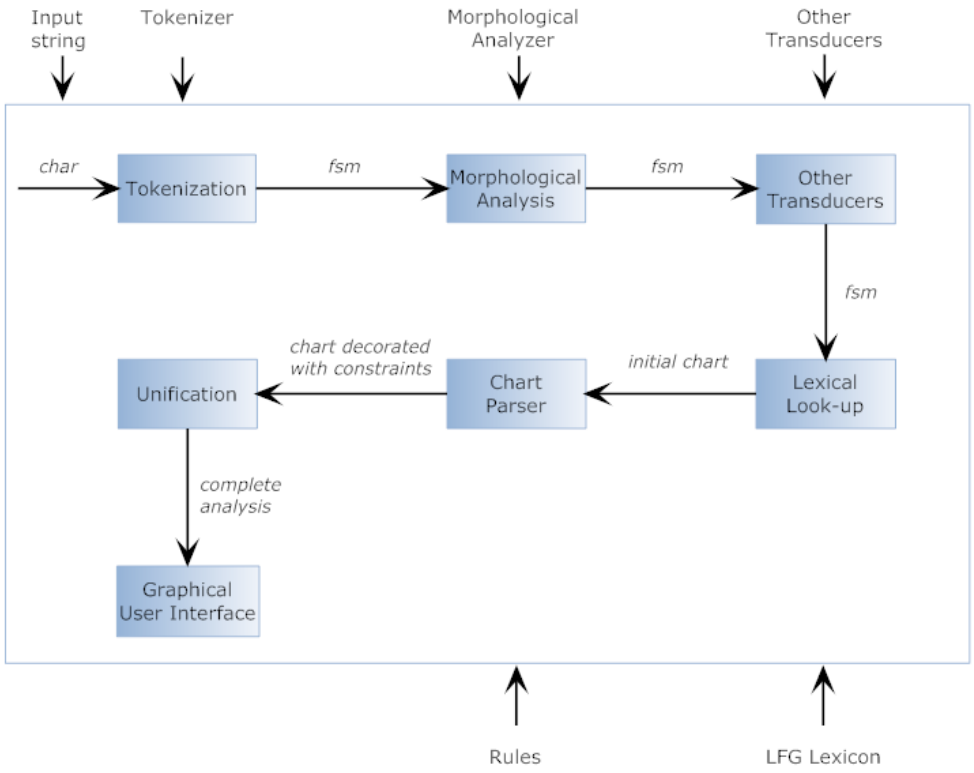


Figure 4.1: The system architecture

We will not give the details but the organization of the basic sections in the grammar. For further information the reader is referred to the XLE documentation [Crouch et al., 2006].

4.1.1 Configuration of the Relevant Files

The interaction between the components of XLE is provided through a configuration section of the grammar. The top level grammar file "turkish.lfg" contains the CONFIG section. 4.1 shows the CONFIG section of the Turkish grammar version we are developing.

(4.1)
```
STANDARD TURKISH CONFIG (1.0)
ROOTCAT  S.
FILES
  common.features.lfg
  turkish-features.lfg
  common.templates.lfg
  turkish-templates.lfg
  turkish-rules-sublexical.lfg
  turkish-rules-np.lfg
  turkish-rules-date_time.lfg
  turkish-lex-suffix.lfg
  turkish-lex-noun.lfg
  turkish-lex-verb.lfg
  turkish-lex-adj.lfg
  turkish-lex-adv.lfg
  turkish-lex-others.lfg
  turkish-lex-date.lfg
  turkish-mw-morphconfig.
LEXENTRIES  (all all).
RULES       (STANDARD TURKISH)
            (SUBLEXICAL TURKISH)
            (NP-RULES TURKISH)
            (DATE-TIME-RULES TURKISH).
FEATURES    (STANDARD COMMON)
            (STANDARD TURKISH).
```

```
TEMPLATES   (STANDARD COMMON)
            (STANDARD TURKISH).
MORPHOLOGY  (MW TURKISH).
GOVERNABLERELATIONS  SUBJ OBJ OBJTH OBL OBL-?+ COMP XCOMP.
SEMANTICFUNCTIONS    ADJUNCT  TOPIC.
NONDISTRIBUTIVES     NUM PERS COORD COORD-FORM.
EPSILON              e.
OPTIMALITYORDER      NOGOOD  NP_rule < NPdate_rule >.

----
```

`ROOTCAT S` sets the parsing category to S (sentence) as the default value. Categories other than S should be specified when parsing (e.g. NP: küçük kedi).

`FILES` includes a list of ".lfg" extended files. These files will be discussed in the following relevant sections according to their functions.

`LEXENTRIES`, `RULES`, `FEATURES`, `TEMPLATES` and `MORPHOLOGY` specify the sections to be used. The given sections are to be used regarding priority.

`GOVERNABLERELATIONS` lists the grammatical relations which must be subcategorized for.

`SEMANTICFUNCTIONS` lists the ungovernable grammatical relations. These attributes must have a PRED value for completeness.

4.1.2 Features

Multilingual grammar development requires agreement on a common set of representations. Therefore, the ParGram Project includes a "feature committee" whose job is to find norms for the definition and use of a common multilingual feature space [Butt et al., 2003]. At project meetings held twice a year, analyses of sample sentences are compared and differences are discussed; the goal is to determine whether the differences are justified or whether the analyses should be changed to maintain parallelism [Butt et al., 2002].

common.features.lfg: This is the file where ParGram feature space is defined. If a language has a given feature, one of these values must be used. New features can be added to this space as they become more universal only after the approval by the ParGram feature committee.

The current version [ParGram, 2006] of the common features is given in Appendix A.

In feature declarations, two basic types of feature values are distinguished: atomic values and complex f-structure values [Butt et al., 2003]. For atomic values, the feature declaration specifies that the value of a feature *F* has to be a member of a set of atomic values (4.2):

(4.2) F:→ $ {value1 value2 ... valueN}

For complex f-structure values, the declaration specifies the features that the embedded feature structure may contain, using the subsumption operator ≪ (4.3):

(4.3) FA:→ ≪ [FB1 ... FB2].

turkish.features.lfg: This file alters the common features in terms of adding a new value to a common feature (+), using a subset of a common feature (&), completely replacing the common feature with new values (!), deleting a common feature which is not used in Turkish grammar (-) or adding a new feature. In other words, Turkish spesific features will be defined here.

Some features altered for Turkish are given in (4.4):

(4.4)
```
!CASE:-> $ { nom acc dat loc abl gen inst equ}.
+NUMBER-TYPE:-> $ {time date}.
+NTYPE:-> << [ CHECK ].
CHECK:-> << [ PART NP-MARKER EXPLICIT].
NP-MARKER:-> $ { + }.
!TIME:-> $ {clock-time date day day-part month year century season +}.
!ADV-TYPE:-> << [ ADV-SEM ADV-SYN].
ADV-SEM:-> << [ TEMP DIRECTION QUALITY QUANTITY CONFIRMATION QUESTION ].
ADV-SYN:-> $ {sadv vpadv focus}.
TEMP:-> $ {day-part season before after almost immediate +}.
APP.
```

4.1.3 Templates

A systematic use of templates are used in lexicon entries and grammar rules to provide readability and maintainability. In files **common.templates.lfg** and **turkish.templates.lfg** common and turkish features are projected to templates respectively.

The default version assumes ^ as the path and the _desig version of the template allows to provide a path. (4.5) shows the template entry for the feature *TIME*.

(4.5)
```
TIME(_val)=
        @(TIME_desig ^ _val).
TIME_desig(_path _val)=
        (_path NTYPE NSEM TIME) = _val.
```

4.1.4 Multiple Transducers

The XLE system is organized to make it easy to combine LFG syntactic specifications with externally developed lexical resources embodied in finite-state transducers [Kaplan et al., 2004]. A “morphological configuration” (morph-config) file referenced from the configuration section of the grammar specifies the transducer combining equations [Kaplan and Newman, 1997]. The morph-config file for Turkish grammar **turkish-mw-morphconfig** is given in (4.6).

(4.6)
```
MW    TURKISH   MORPHOLOGY (1.0)

    TOKENIZE:
    english.tok.parse.fst

    ANALYZE USEFIRST:
    turkish-morph-override.txt
    tfeatures.fst

    ANALYZE USEALL:
    turkish-morph-extend.fst

    MULTIWORD:
    time1.fst time2.fst weekday.fst daymonth.fst daymonthyear.fst
      abbr_date_slash.fst

    ----
```

The ANALYZE USEFIRST transducers are applied to a token one by one until an analysis is found. When an analysis is found, transducers specified in subsequent lines in the subsection are not applied to that token any more(overriding). On the other hand, all the lines of the ANALYZE USEALL subsection are applied to each token(extending). Combining externally supplied transducers with locally developed ones, existing morphological analyses may be modified or additional analyses may be specified.

The MULTIWORD subsection of the morphology section is used for transducers that deal with multi-word phenomena. It usually contains only one line of transducers[1]. Multiword transducers in (4.6) will be referenced from the grammar section of this chapter.

4.1.5 Lexicon Files

turkish-lex-suffix.lfg : Lexical entries of the part-of-speeches and tags
turkish-lex-noun.lfg : Noun lexicon
turkish-lex-verb.lfg : Verb lexicon
turkish-lex-adv.lfg : Adverb lexicon
turkish-lex-adj.lfg : Adjective lexicon
turkish-lex-date.lfg : Date-time lexicon
turkish-lex-others.lfg : Lexical entries for numbers, postpositions, pronouns, determiners and some full-forms

Subentries in the LFG lexicons classified above specify additional properties which are not predictable from their morphological decompositions. However, there are large sets of words in each category that have exactly the same properties, such as common nouns. XLE provides for default specifications that permit many redundant individual entries to be removed from all lexicons [Kaplan and Newman, 1997]. The lexical entry for the special headword -unknown contains subentries giving the default syntactic properties for each category.

For instance, when combined by sublexical rules, the entry given in (4.7) will enable parsing sentences with nouns and adjectives which are known to the morphology but not to the lexicon.

[1]Note that the second line in (4.6) is indented to show that it is appended to the preceeding line.

```
(4.7)  -unknown    N       XLE @(NOUN %stem);
                   A       XLE @(ADJ %stem).
```

In (4.7) %stem stands for whatever actual stem matches `-unknown`.

4.1.6 Rule Files

turkish-rules-sublexical.lfg : Sublexical rules for constructing lexical items from their morphemes
turkish-rules-np.lfg : Noun phrase rules
turkish-rules-date_time.lfg : Date/time phrase rules

4.2 The Grammar

In this section we present the grammar that we have developed for a subset of Turkish temporal expressions. Our grammar includes rules for time expressions in terms of clock-time, days of the week, calendar dates and seasons. Some other general point-in-time expressions are also included.

We implement each of these expressions in turn, beginning with clock time.

4.2.1 Clock-Time

Turkish clock-time expressions are given in Section 3.1. Before going through the analyses of all the expressions, we first discuss how to represent the most simple phrase of all: *saat iki* ‘two o’clock’.

As mentioned in Section 3.1, when expressing the clock-time, the noun *saat* ‘o’clock’ may optionally be used before the hour numeral. However, this phrase turns out to be a special noun phrase: Although *saat* complements *iki* in a way, this phrase is not an ordinary noun phrase as in *saat tamircisi* ‘watch/clock repairer’ where *saat* is a countable common noun, nor has *saat* the durational meaning as in *iki saat* ‘two hours’. In other words, the polysemious word *saat* (watch, hour, o’clock) has a special usage for the clock-time expressions which should be distinguished from others. Since all readings of *saat* go under the noun (N) part-of-speech

(POS), either we define a new category for each or we use a disjunction in the lexicon entry 4.8 assigning three different features for the noun *saat*:

(4.8)
```
saat     N     XLE     @(NOUN saat)     { @(TIME clock-time)
                                          | @(TIME +)
                                          | @(COMMON count)}.
```

This entry will enable us to constrain the feature of *saat* to be “clock-time” when it preceeds a numeral. But the question of “in which sense does the noun saat complement the hour” still remains. We have tested three approaches for the answer.

TIMEP approach

First approach is parallel to the one that is opted for in the English grammar. Like the word ‘o’clock’, the word *saat* may be considered as a time adverbial. It functions as the adjunct of the hour numeral. With the lexical entry given in (4.9), Rule 1 parses the phrase *saat iki* ‘two o’clock’.

(4.9)
```
saat     !ADVtime      *      @(PRED %stem).
```

Note that morphcode * is used since *saat* is given an alternative c-structure category other than N, which does not come from the morphology.

```
TIMEP --> "saat iki"

          ADVtime: ! $ (^ ADJUNCT)
          NUM: ^ = !
               (! NUMBER-TYPE) $c {card time}
               @(BARE-TIME clock-time).
```

Rule 1: Rule for TIMEP approach

The c- and f-structures of the Rule 1 are given in Figure 4.2.

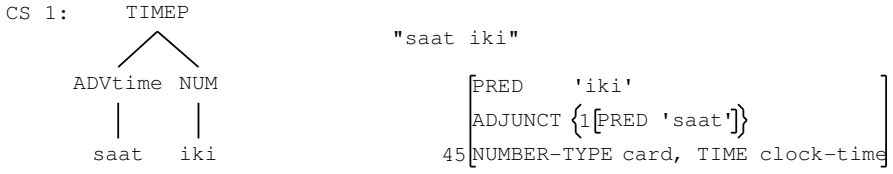


Figure 4.2: C and f-structures for *saat iki* ‘two o’clock’, TIMEP approach

Appositional approach

Second approach to the representation of the phrase *saat iki* ‘two o’clock’ is based on the idea of evaluating the phrase like an appositional noun phrase similar to *bölüm iki* ‘chapter two’, only in this case as if we are labeling every hour. Therefore, the head of the phrase becomes *saat* and the numeral is apposed to it. Here, it is important to select the “o’clock” reading of *saat*. We need to constrain the feature to clock-time, otherwise we would get a common noun sense of *saat* as in ‘watch one, watch two . . . ’ which is not the case [2]. Rule 2 parses the phrase with the head *saat* and the numeral(s) apposed to it. The corresponding c- and f-structures are given in Figure 4.3.

```
NPapposition --> "saat iki"
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c clock-time
                      (^ CASE)=c nom
                      ~(^ SPEC POSS) ~@(NP-MARKER)
            NUM#1#4:  @(APP) "yirmi iki kırk beş"
                      (! NUMBER-TYPE)$c {card time} "2, 14.00"
```

Rule 2: Rule for appositional approach

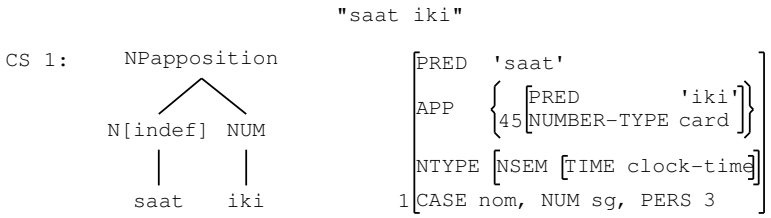


Figure 4.3: C and f-structures for *saat iki* ‘two o’clock’, appositional approach

In Rule 2 note that up to 4 numerals are allowed to be apposed to the noun *saat*. That is because 4 numerals may be used at most for twenty-four hour representation of clock-time expressions. Also note that this notation would result in conflicting heads in the TIMEP approach; an

[2] A general NPapposition rule would be constrained where it is expected to function as a temporal adverbial at the sentence level.

f-structure may only have one head. Nevertheless, we currently do not get an analysis for compound numerals such as *yirmi iki* ‘twenty-two’ from the morphology as a single numeral. The f-structure of *saat yirmi iki kırk beş* ‘twenty-two fourty-five o’clock’ of the appositional approach is given in Figure 4.4:

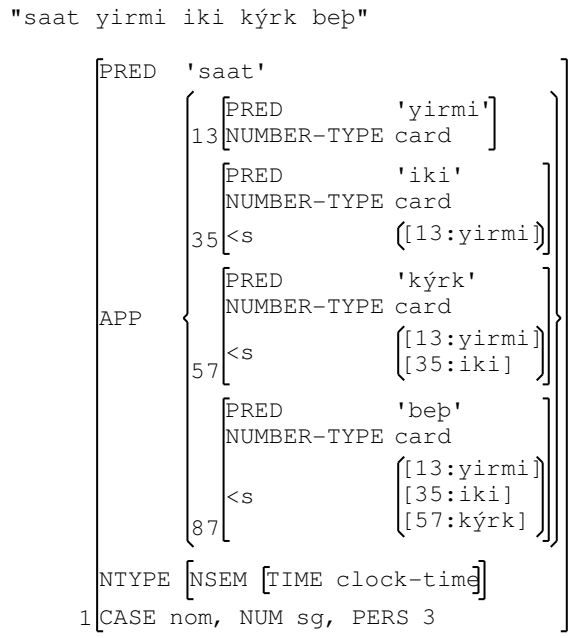


Figure 4.4: F-structure for *saat yirmi iki kırk beş* ‘twenty-two fourty-five o’clock’, Appositional approach

Fortunately, we have the surface scope operator in the definition of the APP template, so that the observer of the f-structure shown in Figure 4.4 can see the order of the numerals. For the sake of simplicity, we will work with single numerals in the rest of the documentation.

In order to function as an adverbial in the sentence, precise clock-time expressions require a locative suffix attached to the numeral; *saat ikide* ‘at two o’clock’. Obviously, the locative case suffix *-DA*[3] in Turkish serves as a preposition (here ‘at’) in English. However, the construction of the Turkish phrase is relatively complicated. The difficulty with this construction is that the numeral changes its POS to N with zero-derivation (i.e. becomes a derived noun) whenever a

[3] *-de, -da, -te, -ta*

case suffix is attached. So, *ikide* ‘at two’ is not a numeral any more, but a noun. (4.10) shows the morphological analyses of *iki* and *ikide*:

(4.10)
```
iki   : iki+Num+Card [4]
ikide : iki+Num+Card^DB+Noun+Zero+A3sg+Pnon+Loc
```

Even though it is only the numeral which gets the suffix, to provide the integrity within the nominative and the derived clock-time phrases, we take the nominative phrase *saat iki* as the stem and attach the derivational suffix to it. Besides, it seems more accurate to attach the suffix to the whole phrase since the word *saat* modifies the hour in such a way that they behave together: at (two o’clock) ? (at two) o’clock.

The use of the word *saat* is optional, so numerals with suffixes should also be handled. We extend the existing NPderiv rule regarding these requirements. The relevant part is given in Rule 3:

```
NPderiv -->
          { "ikide, 20.00’de"
            NUM: (^ OBJ)= !
            "derived noun can be marked by ‘TIME clock-time’ only if
             the stem number is of the time type:20.00"
                 @(IF (! NUMBER-TYPE) =c time
                       @(TIME clock-time))
          | "saat ikide"
            NPapposition:(^ OBJ)= !;
          }
          DS: ^=!.
```

Rule 3: Rule for derived noun phrases

NPderiv is than called by the general NP rule:

```
NP --> NPderiv.
```

Figure 4.5 shows the c- and f-structures.

[4]We also get a derived noun analysis with +Nom for cardinals:

iki : iki+Num+Card^DB+Noun+Zero+A3sg+Pnon+Nom (This noun analysis will be used in Section~4.2.1)

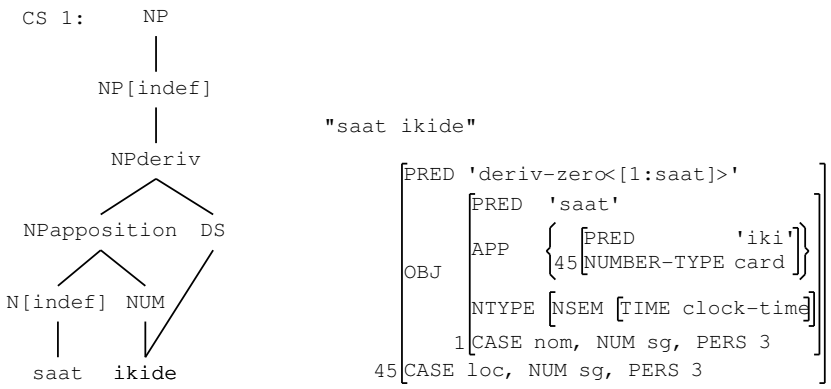


Figure 4.5: C and f-structures for *saat ikide* 'at two o'clock', appositional approach

As already discussed in Section 2.3.1, the stem phrase behaves as the object of the derived one. In the f-structure display in Figure 4.5 *saat* seems to be the one which gets the zero derivation. But this is only because it is the head of the stem phrase *saat iki*. Still, it may be confusing for one who is not familiar with the Turkish morphosyntax. That was the motivation to drive the third and the last approach for the representation of the phrase *saat iki* 'two o'clock'.

Multiword approach

Third approach for the clock-time expression is a preprocessing application of the finite state technology with regard to multiwords. We give the analysis of the main morphological analyzer to an additional transducer that concatenates the noun *saat* with the following numeral (The text file of the transducer is given in Appendix B). There we obtain a new analysis for the multiword *saat iki* with a new tag, namely `+ClockTime`. Then we define this tag as a type of N category (4.11) and include in the sublexical rule, Rule 4, for the category `CLOCK`.

(4.11) `+ClockTime N_TYPE_SFX XLE @(TIME clock-time).`

```
CLOCK -->
          CLOCK_BASE
          N_SFX_BASE
          N_TYPE_SFX_BASE
          NUM_PERS_SFX_BASE
          POSS_SFX_BASE
          CASE_SFX_BASE.
```

Rule 4: Sublexical rule for clock-time

The multiword noun *saat iki* becomes the head predicate together. Since there is no derivation in this case, no further attempt is needed to handle the locative case marked *saat ikide* ‘at two o’clock’. The case value of the noun is simply shown by the CASE feature in the f-structure.

We finally give the c- and f-structures of this approach in Figure 4.6. Note the +ClockTime in the expanded tree representation.

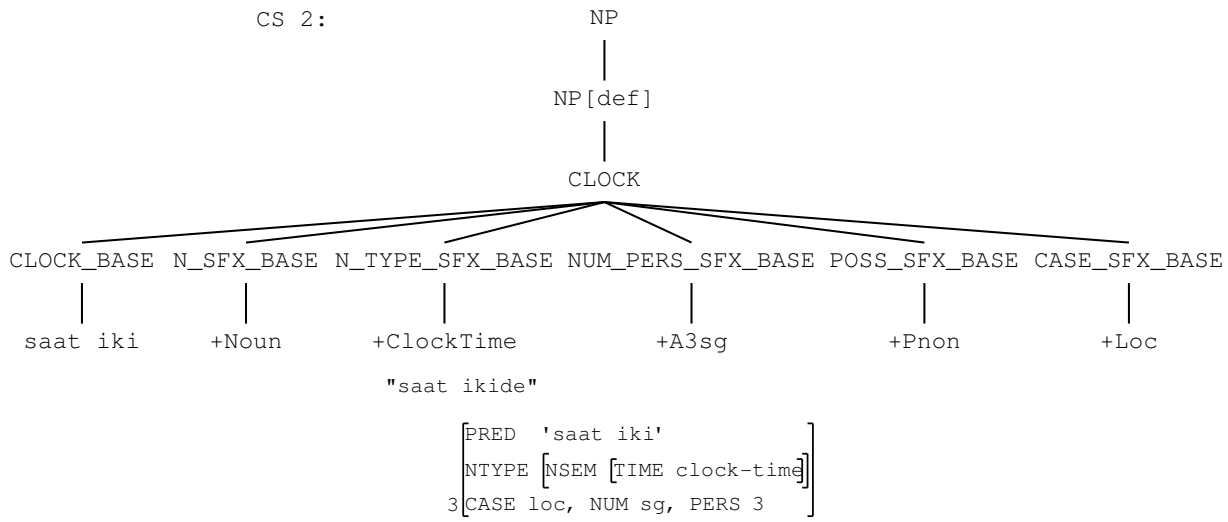


Figure 4.6: C and f-structures for *saat ikide* ‘at two o’clock’, multiword approach

More complex clock-time expressions will be based on this last representation.

We go on with postpositional clock-time phrases. As stated before, the functions of some prepositions are realized by case suffixes in Turkish (recall *-DA* vs. ‘at’). Those of the rest, which can not be handled by suffixes are realized by postpositions which follow the word they govern. A postposition is the head of a postpositional phrase in Turkish just like a preposition is the head of a prepositional phrase in English.

Among the other postpositions, there are four of them that requires special interest in terms of their subcategorizing bahaviour: *geçe* ‘past’, *kala* ‘to’, *önce* ‘before’ and *sonra* ‘after’.

First consider the postpositions[5] *geçe* ‘past’ and *kala* ‘to’ which are commonly used when expressing less than thirty minutes before or after a particular hour time.

(saat) *iki-**yi*** *on/on dakika/çeyrek/biraz* *geçe*
(o’clock) two-ACC ten/ten minutes/quarter/a little bit past
‘at ten past two (o’clock)’

(saat) *iki-**ye*** *on/on dakika/çeyrek/biraz* *kala*
(o’clock) two-DAT ten/ten minutes/quarter/a little bit to
‘at ten to two (o’clock)’

They tend to govern a nominative word which denotes the amount of time past/to the hour numeral. So, one could choose the preceeding nominative words/phrases as the objects of the relevant postpositional phrases. This would be parallel to the analysis we get from the morphology (4.12) where these two postpositions are expected to follow a nominative word:

(4.12) `geçe : geçe+Postp+PCNom`
`kala : kala+Postp+PCNom`

However, there are some cases in which the speaker/writer skips the quantity information for some reasons (does not know exactly, is not sure whether to say or not, does not find it important enough to mention). The following are the examples of this case from the corpus query with *geçe* ‘past’:

*Sabahları **saat 6’yı geçe** ağarıyor gün artık.*
‘The sun now rises **after 6 o’clock** in the mornings.’

*Bütçe açığı zaten **yılın yarısını geçe** 7-8 katrilyonu buldu.*
‘Deficit in budget was already 7-8 quadrillion **as half of the year is past**.’

*Ne zaman **gece yarısını geçe** uyansam, ilkin deprem mi oldu diye bir düşünce geçiyor beynimin içinden.*
‘Whenever I wake up **after midnight** I first come with the thought of an earthquake.’

[5] Even though they are called the converbs of the verbs *geç-* ‘pass’ and *kal-* ‘be left’ in Turkish grammar [Göksel and Kerslake, 2005], as will become clear from the examples discussed, *geçe* and *kala* are rather used like postpositions which is why we classify them as postpositions.

Although this use of *geçe* ‘past’ is rare[6], we still want to parse such cases. Due to the fact that a postposition (POSTP) is a subcategorizing category[7], skipping the object would result in an incomplete f-structure. Thus, a null-object should be allowed. We can then cope with phrases as the ones given in (4.13):

(4.13) *(saat)* *iki-yi* *geçe*
(o’clock) two-ACC past
‘. . . past two (o’clock)’

Another question arises at this point: What is the function of the accusative-marked noun phrase that preceeds the postposition *geçe* ‘past’? Here we realize that it makes more sense when we let the noun (e.g. *birşeyi* in 3a) be the object and the optional quantity information serve as an adjunct in the postpositional phrase.

A generalized phrase structure would be as given in (4.14):

(4.14) a. *birşey-i* *şu* *kadar* *geçe*
something-ACC this much past
‘this much past something’

b. *birşey-e* *şu* *kadar* *kala*
something-DAT this much to
‘this much to something’

So, the required morphological analyses for *geçe/kala* ‘past/to’ turn out to be as in (4.15):

(4.15)
```
geçe   : geçe+Postp+PCAcc
kala   : kala+Postp+PCDat
```

Note that, in (4.15) *geçe* ‘past’ follows an accusative word whereas *kala* ‘to’ follows a dative. We temporarily obtained these analyses by a text-specified transducer which is placed before the main morphological analyzer [Oflazer, 1994] under the “use-first” section of the config file. That is, we overwrote the analyses we got from the main analyzer.

---

[6] ‘after’ could be used instead, but following an ablative form.

[7] postposition template: POSTP(_pred) = (ˆ PRED) = ‘_pred <(ˆ OBJ)>’.

On the other hand, the postpositions *önce* ‘before’ and *sonra* ‘after’ both subcategorize for an ablative case-marked form of a noun. We get multiple analyses for these two words including the postposition part-of-speech:

(4.16)
```
önce    : önce+Postp+PCAbl
          önce+Noun+A3sg+Pnon+Nom
          önce+Adverb

sonra   : sonra+Postp+PCAbl
          sonra+Noun+A3sg+Pnon+Nom
          sonra+Adverb
```

We write the most likely *önce/sonra* ‘before/after’ phrases down:

(4.17) *az* *önce*
a-little before
‘right before’

(4.18) *iki* *dakika* *önce*
two minute before
‘two minutes ago’

(4.19) *2* *saat* *10* *dakika* *sonra*
2 hour 10 minute after
‘after 2 hours 10 minutes’

(4.20) *saat* *iki-den* *sonra*
o’clock two-ABL after
‘after two o’clock’

(4.21) *saat* *iki-den* *on* *dakika* *sonra*
o’clock two-ABL ten minute after
‘ten minutes after two o’clock’

(4.22) *saat* *iki-den* *hemen* *sonra*
o’clock two-ABL right after
‘right after two o’clock’

Consider the phrases (4.17) and (4.18). They are not clock-time expressions but more general temporal expressions. They speak about “how much time?” but not about “before/after what?”. They could also be analyzed as noun-phrases as we also get noun analyses for *önce/sonra* ‘before/after’[8]. But we want parallel analyses for all cases regardless of the explicitly referred event/object. So we apply the same approach as in *geçe/kala* ‘past/to’ and obtain a similar formula in (4.23).

(4.23) *birşey-den* *şu* *kadar* *önce/sonra*
something-ABL this much before/after
‘this much before/after something’

Putting these facts together, a postpositional-phrase rule (POSTPP) is given in Rule 5. It generalizes over cases (4.17) to (4.22).

```
POSTPP -->
          {
            NP: (^OBJ) = !
            ( @(MEASUREP) )
          | e: (^OBJ PRED) = ‘null-pro’
               (^OBJ PRON-FORM) = null
               (^PRED)$c { geçe kala önce sonra };
            @(MEASUREP)
          }
          POSTP.


MEASUREP =
          {
            NP#1#6: @ADJUNCT    "(iki saat) iki dakika"
                (! SPEC NUMBER)
                (! NTYPE NSEM TIME) =c +
                (! CASE) = nom
                (^ PRED)$c { geçe kala önce sonra }
          | NUM: ! $ (^ ADJUNCT)   "on"
                 (^ PRED)$c { geçe kala}
          | ADV: ! $ (^ADJUNCT)    "az"
                 (^ PRED)$c { geçe kala önce sonra }
    }.
```

Rule 5: Rule for postpositional phrases

[8] *az* ‘a-little’ can be both A or ADV. The adverbial analyses of *önce/sonra* ‘before/after’ will be discussed in a later section.

C- and f-structures of some postpositional phrases are given in Figure 4.7, 4.8 and 4.9.

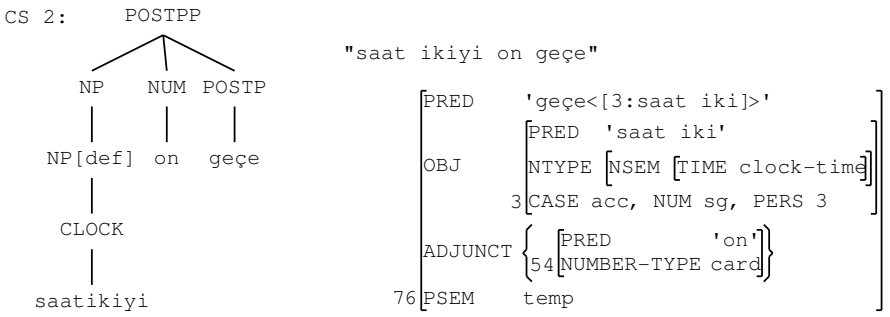


Figure 4.7: C and f-structures for *saat ikiyi on geçe* ‘ten past two o’clock’

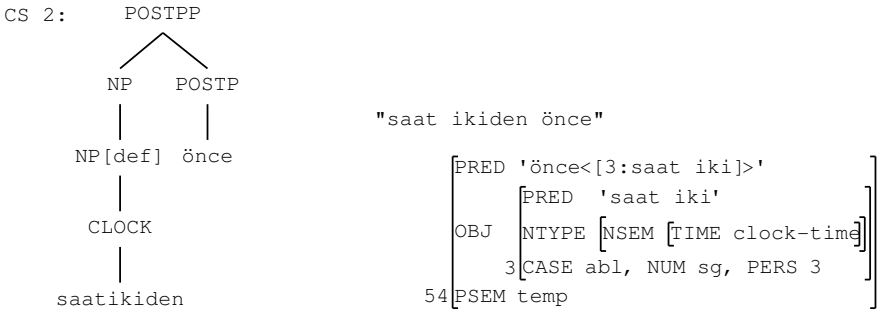


Figure 4.8: C and f-structures for *saat ikiden önce* ‘before two o’clock’

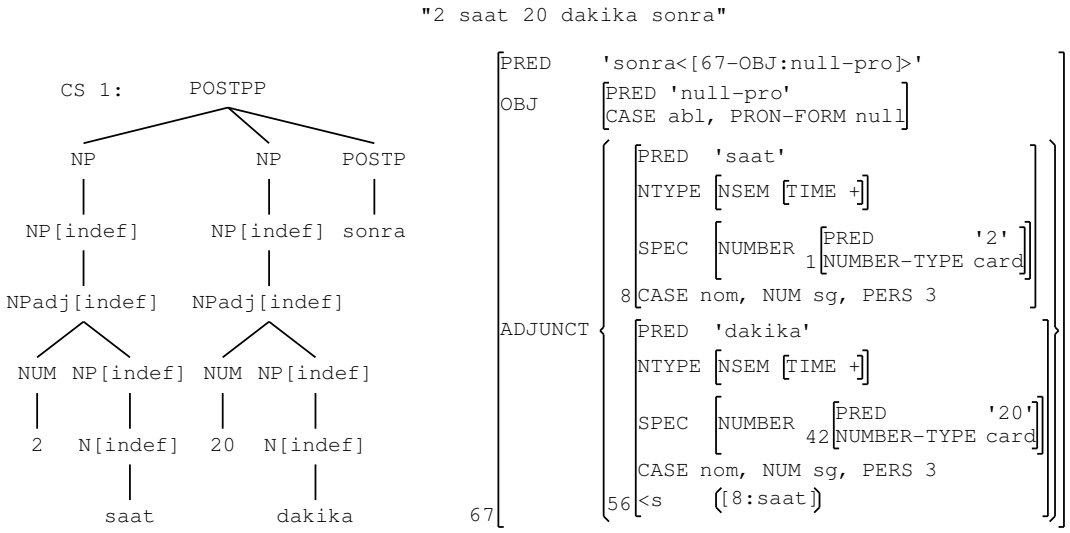


Figure 4.9: C and f-structures for *2 saat 20 dakika sonra* ‘after 2 hours 20 minutes’

By the approach for which we opted, the semantic behaviour of these (so-called) postpositions may be noticed from the f-structures as well. Null-object in Figure 4.9 points to the fact that even though the phrase is missing the object, this information can be extracted from the discourse.

Approximate time expressions headed by the nouns *sularında/sıralarında/dolaylarında/civar(lar)ında* ‘around’ will be parsed by the Rule 9, `TIMEPapprox`. The reason why we treat those nouns as of a new category `POSTPtime` in Rule 9 is illustrated in (4.24).

(4.24)

22.30 sıralarında		vs.	*okul sıralarında*
‘around 22.30’			‘at school desks’
saat 6 sularında		vs.	*Akdeniz sularında*
‘around 6 o’clock’			‘in the waters of the Mediterranean’
saat 22.00 dolaylarında		vs.	*Orta Anadolu dolaylarında*
‘around 22.00 o’clock’			‘in the surrounding of Middle Anatolia’
saat dokuz civarında		vs.	*evimizin civarında*
‘around 9 o’clock’			‘in the surrounding of our house’

They are introduced to the lexicon as given in (4.27) for *sularında* ‘around’.

(4.25) `sularında POSTPtime * (^ PRED)='%stem<(^ OBJ)>' @(PSEM temp).`

Similarly, the noun *arasında* ‘between’ functions as a postposition when it is used in terms of “between x and y”.

(4.26) *saat bir ile iki arasında* vs. *saçları arasında*

The lexicon entry for *arasında* ‘between’ differs from (4.27) in that the object of the phrase has to be coordinated:

(4.27)
```
arasında   POSTPtime  *  (^ PRED)='%stem<(^ OBJ)>'  (^ OBJ COORD-FORM)
                          @(PSEM temp).
```

For the coordination of the same categories in clock time expressions (4.28) we write a macro in (4.29) and extend the sublexical rules NUM and CLOCK in Rule 6 and Rule 7.

(4.28)

[CLOCK: saat iki]	ile	*[CLOCK: saat on]*	*arasında*
[NUM: 14.00]	-	*[NUM: 22.00]*	*arasında*

(4.29)
```
SCCOORD(_CAT ) = "same category coordination"

                "COM{EX RULE PP: iki ile on, saat iki ile saat on, 2 - 10}"

                _CAT: @IN-SET; "one conjunct before the coordinator"
                {CONJ|HYPHEN}
                _CAT: @IN-SET.
```

```
NUM --> {
          NUM_BASE
          NUM_SFX_BASE
          { CARD_TYPE_BASE
          | ORD_TYPE_BASE
          | DIST_TYPE_BASE
          | TIME_TYPE_BASE
          | DATE_TYPE_BASE }
        | @(SCCOORD NUM) }.
```

Rule 6: Sublexical rule for numerals, coordinated

```
CLOCK --> {
            CLOCK_BASE
            N_SFX_BASE
            N_TYPE_SFX_BASE
            NUM_PERS_SFX_BASE
            POSS_SFX_BASE
            CASE_SFX_BASE
          | @(SCCOORD CLOCK) }.
```

Rule 7: Coordinated-Sublexical rule for clock-time

The elements of a coordination may be of different categories (4.30). CLOCKNUMCOORD, Rule 8, parses such coordinations.

(4.30) *[CLOCK: saat iki]* ile *[NUM: on]* *arasında*
[CLOCK: saat 14.00]- *[NUM: 22.00]* *arasında*

```
CLOCKNUMCOORD -->

           "saat iki ile on"

           CLOCK: @IN-SET;
           { CONJ|HYPHEN }
           NUM: @IN-SET.
```

Rule 8: Rule for clock-time noun and number coordination

Below is the TIMEPapprox rule:

```
TIMEPapprox -->

            { "saat iki, saat 20.00, saat iki ile saat on"
              CLOCK: (^ OBJ)= !
                     (! CASE)=c nom
            | "iki, 20.00, iki ile on"
              NUM: (^ OBJ)= !
                   (! NUMBER-TYPE)$c {card time}
            | "saat iki ile on"
              CLOCKNUMCOORD: (^ OBJ)= !
            }
            POSTPtime.
```

Rule 9: Rule for approximate clock-time expressions

Some c- and f-structure representations of approximate time expressions are given in Figures 4.10, 4.11 and 4.12.

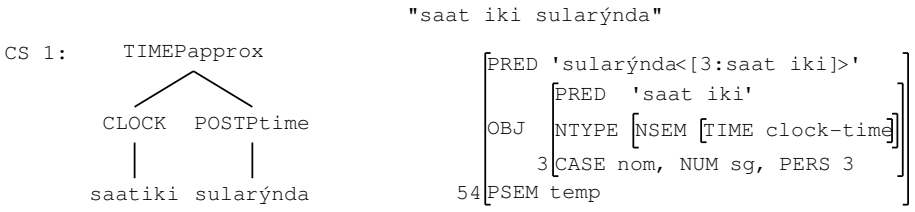


Figure 4.10: C and f-structures for *saat 2 sularında* ‘around 2 o’clock’

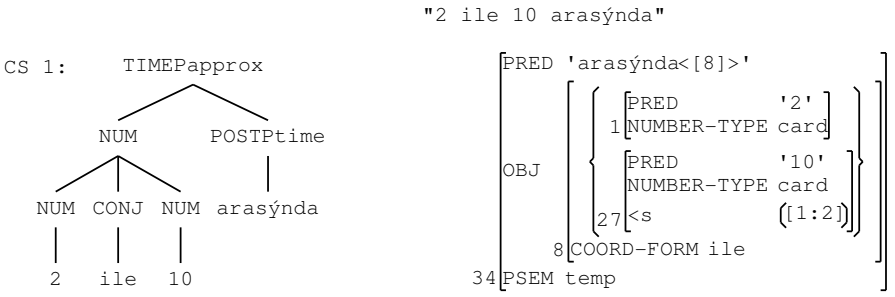


Figure 4.11: C and f-structures for *2 ile 10 arasında* ‘between 2 and 10’

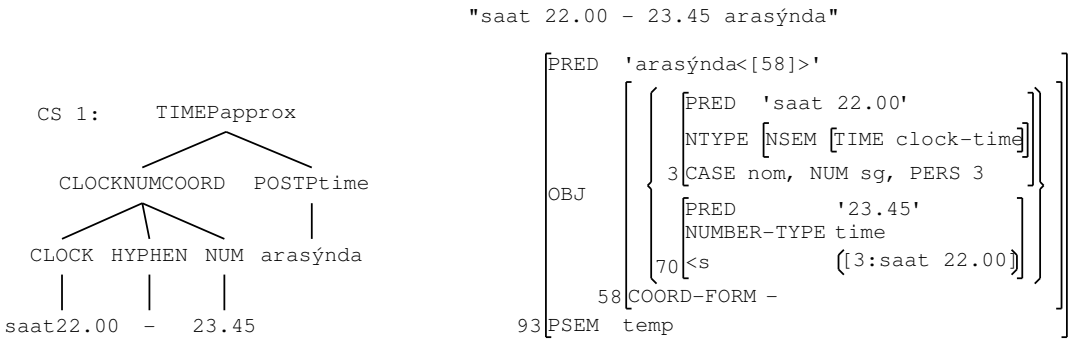


Figure 4.12: C and f-structures for *22.00-23:45 arasında* ‘between 22.00-23.45’

The other approximate time expression style listed in Section 3.1 was formed appending the postposition *gibi* ‘about’ to the clock-time: *saat iki gibi* ‘about two o’clock’. POSTPP rule,

Rule 5 on page 63 already parses such phrases as long as we get an NP analysis for the clock-time time expression. We can make use of the derived noun analyses of the cardinals (zero derivation with +Nom) where the clock-time is expressed only by numerals. Figure 4.13 shows the c- and f-structure of this case.

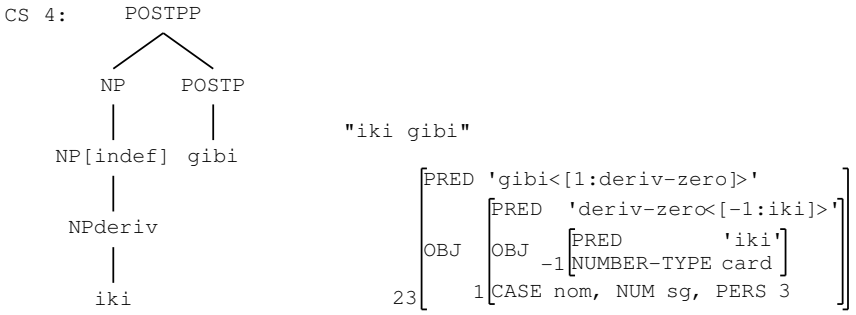


Figure 4.13: C and f-structures for *iki gibi* ‘about two’

Finally, the clock-time compounded with the part of the day information is parsed by the Rule 10, `TIMEPdaypart`. The nouns *sabah* ‘morning’, *öğle* ‘afternoon’, *akşam* ‘evening’, *gece* ‘night’ etc. are introduced to the lexicon with the ‘`TIME day-part`’ feature-value pair as in (4.31).

(4.31) `sabah N XLE @(NOUN sabah) @(TIME day-part).`

C- and f-structures of the phrase *sabahın altısında* ‘six o’clock in the morning’ is given in Figure 4.14.

```
TIMEPdaypart -->

        N[indef]: @MOD
                  (! NTYPE NSEM TIME)=c day-part
                  (! CASE)=c gen
                  ~(! SPEC POSS)
                  (! NTYPE CHECK NP-MARKER) ~=c +;
        NPderiv: ^ = !
                 (^ OBJ NUMBER-TYPE)
                 @(TIME clock-time)
                 (! NTYPE CHECK NP-MARKER)=c +.
```

Rule 10: Rule for noun phrases expressing clock-time with the part of the day information

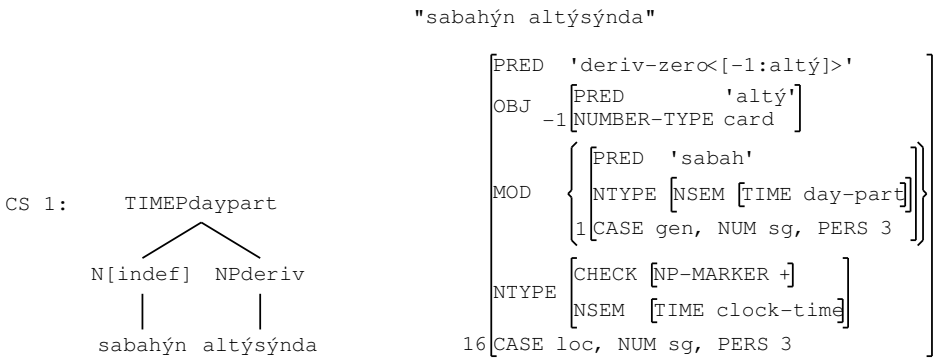


Figure 4.14: C and f-structures for *sabahın altısında* 'six o'clock in the morning'

4.2.2 Days of the Week

Three types of expressions were figured out for location on the days of the week (See Section 3.2):

(4.32) 1. Day in nominative case

Pazar
Sunday
'on Sunday'

2. Day with case marking

a. *Pazar-a*
Sunday-DAT
'on Sunday'

b. *Pazar-dan*
Sunday-ABL
'already on Sunday'

3. Day compounded with "günü"

Pazar günü
Sunday day-P3sg
'on Sunday'

We want these expressions to be marked by the feature `TIME` with the value “day” in the f-structure representations.

Pazar In (4.32), we intensionally chose the polysemious word *Pazar* ‘Sunday’ so that we can discuss some linguistic points through. Three readings of *Pazar* ‘Sunday’ are given in (4.33):

(4.33) *Pazar/pazar* : Sunday
Pazar : the name of a town
pazar : bazaar

The morphological analysis of *Pazar* (with an initial capital) is given in (4.34):

(4.34)
```
xfst[1]: up Pazar
         pazar+Noun+A3sg+Pnon+Nom
         Pazar+Noun+Prop+A3sg+Pnon+Nom
```

Note that the morphological analyzer also lists analyses beginning with a small letter due to the fact that it might be the first word of a sentence (cf. the query “up pazar” does not give the proper noun analysis).

It is not clear from (4.34) whether the day *Pazar* ‘Sunday’ is evaluated as a proper noun as we already know that there are both a proper name (the town) and a common name (bazaar) reading of the word. Thus, we check for a non-polysemious day name. (4.35) is the analysis for *Pazartesi* ‘Monday’:

(4.35)
```
xfst[1]: up Pazartesi
         pazartesi+Noun+A3sg+Pnon+Nom
```

We now see that our morphology treats the names of the days (of the week) as regular nouns.

Some of the languages (e.g. English) assume the days of the weeks as proper nouns. Turkish, on the other hand, regulates that the names of the days and months should be written with an initial capital only within a precise calendar date expression (as in 4.36a, 4.36b), otherwise not (as in 4.36c) [TDK, 2006]:

(4.36) a. *9 Eylül 1982 Pazar günü doğdu.*
9 September 1982 Sunday day-P3sg was-born
'She was born on Sunday, September the 9th, 1982.'

b. *Lale festivali 25 Haziranda başlayacak.*
Tulip festival-P3sg 25 June-LOC start-FUT-A3sg
'Tulip festival is going to start on June the 25th.'

c. *Okullar genellikle eylülün ikinci haftasında öğretime başlarlar.*
Schools generally September-GEN second week-P3sg-LOC education-DAT start-AOR-A3sg
'Generally, the schools start by the second week of September.'

d. *Yürütme Kurulu toplantılarını perşembe günleri yaparız.*
Executive committee meetings-P3sg-Acc Thursday days-P3sg do-AOR
'We have the executive committee meetings on Thursdays.'

We will deal with date expressions in Section 4.2.3. But, note the locational suffix to the precise date expression *25 Haziran* 'June the 25th in (4.36)a. In a proper noun case, the inflectional suffixes are supposed to be separated by an apostrophe [TDK, 2006]. Shortly, even though day and month names are expected to be written beginning with a capital letter in calendar date expressions, they are not proper nouns in Turkish.

As can ben seen from (4.35), the day nouns do not get a distinctive tag in our morphology system. In order to use the day readings of these common nouns we define a new category by the sublexical rule DAY, Rule 11.

```
DAY -->
        DAY_BASE
        N_SFX_BASE
        NUM_PERS_SFX_BASE
        POSS_SFX_BASE
        CASE_SFX_BASE.
```

Rule 11: Sublexical rule for day nouns

Since there is a closed set of day nouns we enter all of them to the lexicon with the DAY category in the fashion given in (4.37):

(4.37) `pazar DAY XLE @(N-DAY %stem); ETC.`

(4.37) introduces *pazar* ‘Sunday’ to the lexicon as a day noun with the N-DAY template presented in Section 4.1.3. Placing ETC in (4.37) results in retaining all previous entries. That is, we keep the other category analyses (N and PROP) for Pazar as long as (4.37) is placed in a later lexicon in the configuration of the lexicon files. For the days other than *pazar* ‘Sunday’ and *cuma* ‘Friday’ [9] common noun analyses by the morphology are meant to be given for the days of the week. (4.38) deletes the N entry for *pazartesi* ‘Monday’ once DAY is added:

(4.38)
```
pazartesi      +DAY      XLE      @(N-DAY %stem);
               -N        XLE      ; ETC.
```

(4.39) may have been written as an alternative to (4.38). Placing ONLY as the final subentry in a later lexicon will remove all earlier subentries unless they are explicitly retained with the “=” operator [Butt et al., 1999].

(4.39)
```
pazartesi      +DAY      XLE      @(N-DAY %stem); ONLY.
```

The days of the week may be the subject or the object of a sentence. (4.40) is an example of the subject case:

(4.40)

Pazartesi	*haftanın*	*en*	*sevdiğim*	*günüdür.*
Pazartesi	week-GEN	most	like-PASTPART-P1sg	day-PRES-COP-A3sg
Pazartesi	of the week	most	that I like	is the day

‘**Monday** is the day of the week that I like most.’

So, DAY is called by the general NP rule. We do not get an additional N analysis for *Pazartesi* thanks to the operators to manipulate subentries.

The resulting f-structure for *pazar* ‘on Sunday’ is shown in Figure 4.15.

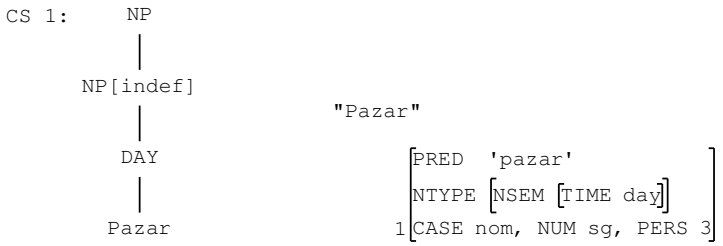


Figure 4.15: C and f-structures for *Pazar* ‘Sunday’

[9] *pazar* ‘Sunday’ in the sense of “bazaar, market” and *cuma* ‘Friday’ in the sense of “Friday prayer” are also common nouns.

Pazara, Pazardan *Pazara* ‘on Sunday’ is the dative marked form of *pazar* and *pazardan* ‘already on Sunday’ is the ablative marked form:

```
xfst[1]: up pazara
         pazar+Noun+A3sg+Pnon+Dat
xfst[1]: up pazardan
         pazar+Noun+A3sg+Pnon+Abl
```

Hence, the c- and f-structures are similar to the ones for *pazar*, but with the ‘`CASE dat`’ for *pazara* and ‘`CASE abl`’ for *pazardan*.

Case marked day adverbials should not be confused with oblique objects or indirect complements:

Pazar-dan ***itibaren*** *çalışmaya* *başlıyorum.*
Sunday-ABL from-on work-DAT start-PROG-A1sg
‘**From Sunday on** I start working.’

Her *şey* ***pazar-a*** ***bağlı***.
Every thing Sunday-DAT dependent.
‘Everything **depends on Sunday**.’

Another approach should be taken into consideration here. One could also consider *pazara* ‘on Sunday’ or *pazardan* ‘already on Sunday’ as an adverb part-of-speech. In that case following analyses have to be added to the morphology:

```
pazara    : pazara+Adverb
pazardan  : pazardan+Adverb
```

That analysis, however, would not be marked by the ‘`TIME day`’ feature since it is not a noun type anymore. There, `TEMP day` (adverbial type feature value) would need to be defined. We opted for a homogeneous analysis as we already get case marked days as nouns.

Pazar günü According to our corpus examination, compounding the name of a day with the word *günü* is the most common way of expressing the time in terms of the days of the week. Table 4.1 shows the distribution of the alternatives [10]:

Table 4.1: Frequencies of the use of days of the week

Expression format	*# of matches*
DAY günü	226
DAY	85
DAY-DAT	7
DAY-ABL	0

Pazar günü 'Sunday' could be analyzed as a noun phrase (noun-noun compound); *günü* being the head of the phrase (cf. *pazar gecesi* 'Sunday night'). In that case, we get the f-structure representation in Figure 4.16 :

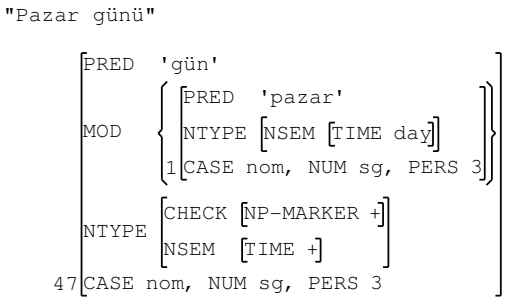


Figure 4.16: F-structure for *Pazar günü* 'on Sunday'

The attribute/value pair '`TIME +`' in Figure 4.16 is employed for temporal words other than the days of the week, the months and the seasons. *gün* 'day' is one of those words.

This representation would be satisfying for noun phrases like *Eylül günü* 'September day', *yaz günü* 'summer day' etc. where the focus is on the "day" word. The two-word expression *pazar*

[10]The very low rate of matches for the case marked days is also a result of lacking a spoken component in the design of the corpus. Because case suffix marking on the days of the week is mostly found in colloquial usage.

günü ‘Sunday’, however, is used to express directly the day of the week. So it seems reasonable to treat it as a single token. We basically take the same steps as we did for the clock time expressions. First we check the analyses provided by the morphology for the words *pazar* and *günü*.

```
xfst[1]: up pazar
         pazar+Noun+A3sg+Pnon+Nom
xfst[1]: up günü
         gün+Noun+A3sg+P3sg+Nom
```

Then we write a multi-word transducer according to the output of the morphology (given in Appendix C and put it after the morphological analyzer. An additional tag, `+WeekDay`, is used to label the new type of noun.

(4.41) `+WeekDay  N_TYPE_SFX  XLE  @(TIME day).`

We rewrite the `DAY` sublexical rule given in Rule 11 adding an optinal line, `N_TYPE_SFX_BASE`, which stands for the `+WeekDay` tag in Rule 12.

```
DAY -->
        DAY_BASE
        N_SFX_BASE
        (N_TYPE_SFX_BASE)
        NUM_PERS_SFX_BASE
        POSS_SFX_BASE
        CASE_SFX_BASE.
```

Rule 12: Modified sublexical rule for days

The lexicon entry of *pazar günü* ‘on Sunday’ is given in (4.42).

(4.42) `pazar‘ günü  DAY  XLE  @(N-DAY %stem); ONLY.`

After all, *pazar günü* ‘on Sunday’ can be parsed just like *pazar* ‘on Sunday’. The expanded c-structure and the relevant f-structure are given in Figure 4.17. Note the `+WeekDay` tag on the tree.

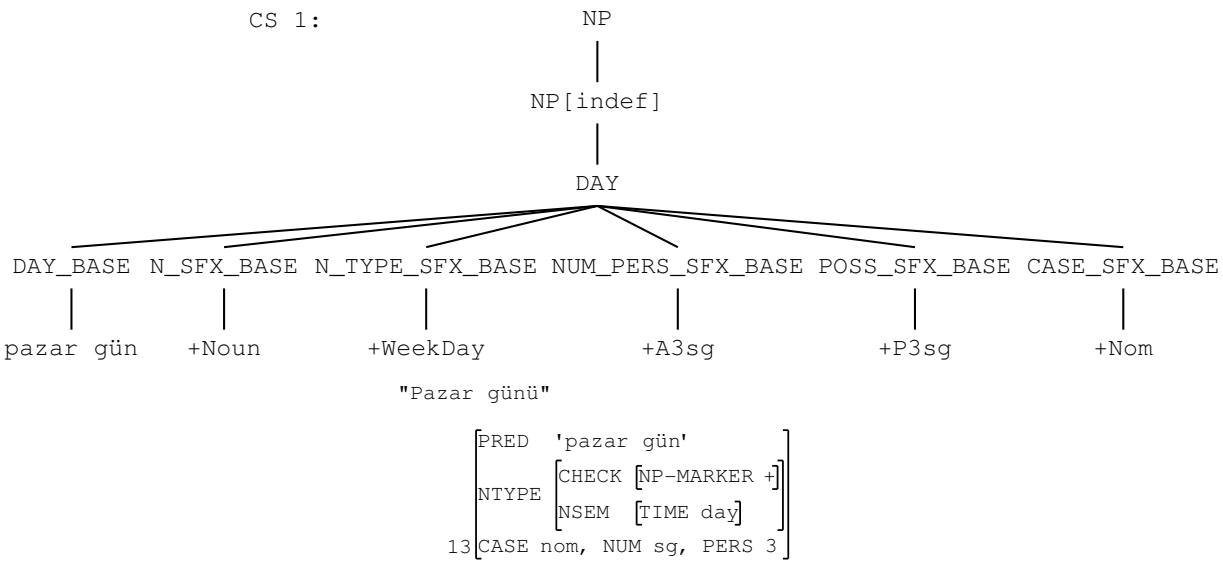


Figure 4.17: C and f-structures for *Pazar günü* 'on Sunday'

4.2.3 Calendar Dates

Date expressions which are discussed in Section 3.3 will be analyzed in this section.

First of all, a sublexical rule for a new category, `MONTH`, will be written so that the names of the months can be introduced to the lexicon with the feature-value pair '`TIME month`' as in (4.43).

(4.43) `eylül MONTH XLE @(N-MONTH %stem); ETC.`

Three facts that will guide us through the design of the `MONTH` sublexical rule are given below:

1. Month names are accepted as common nouns in Turkish. As discussed together with the days of the week names in Section 4.2.2, suffixes to the month names are supposed not to be separated by an apostrophe [TDK, 2006].

2. Our corpus queries show that, unsimilar to day nouns [11], people tend to use an apostrophe before the suffixes of the month names in precise date expressions. Table 4.2 shows the distribution with the locative case suffix, *-DA*.

Table 4.2: Frequencies of the use of month nouns

Expression format	*# of matches*
month-LOC	37
Month-LOC	7
Month'-LOC	524

3. Our morphology analyzes inflected month names as proper nouns when the apostrophe is used and as common nouns when not. (4.44) shows the analyses for *Haziranda* 'in June'.

(4.44)
```
xfst[1]: up Haziranda
         haziran+Noun+A3sg+Pnon+Loc
xfst[1]: up Haziran'da
         Haziran+Noun+Prop+A3sg+Pnon+Loc
```

The large number of matches for the proper noun use of months could be a motivation to change their POSs to `+Prop`. Nevertheless, this solution disables the adjacent use of the suffixes to the months which, in fact, is told to be the appropriate way of writing.

Thus, we look for a normalization to handle both cases. The optional `PROP_SFX_BASE` line in the `MONTH` sublexical rule, Rule 13, enables to parse month nouns when suffixes are attached by an apostrophe; *Haziran'da* 'in June' (proper noun case). *Haziranda* or *haziranda* will be parsed by the same rule (common noun case).

```
MONTH -->
          MONTH_BASE
          N_SFX_BASE
          (PROP_SFX_BASE)
          NUM_PERS_SFX_BASE
          POSS_SFX_BASE
          CASE_SFX_BASE.
```

Rule 13: Sublexical rule for months

[11] Actually that is because the day names get their case markings to the noun "günü" that they are usually compounded with.

One could argue that getting both common and proper noun analyses for months is linguistically not consistent. What could be done alternatively is removing the line that stands for the proper noun analysis in Rule 13 and letting the inflected common month nouns get an apostrophe. This can be done by a text-specified transducer within the section “morphology use-all”. The entry that would be needed for the month *Eylül* ‘September’ is given in (4.46).

(4.45)
```
E y l ü l +Noun +A3sg +Pnon +Acc
E y l ü l ’ ü

E y l ü l +Noun +A3sg +Pnon +Dat
E y l ü l ’ e

E y l ü l +Noun +A3sg +Pnon +Loc
E y l ü l ’ d e

E y l ü l +Noun +A3sg +Pnon +Abl
E y l ü l ’ d e n

E y l ü l +Noun +A3sg +Pnon +Gen
E y l ü l ’ ü n

E y l ü l +Noun +A3sg +P3sg +Nom
E y l ü l ’ ü

E y l ü l +Noun +A3sg +P3sg +Acc
E y l ü l ’ ü n ü

E y l ü l +Noun +A3sg +P3sg +Dat
E y l ü l ’ ü n e

E y l ü l +Noun +A3sg +P3sg +Loc
E y l ü l ’ ü n d e

E y l ü l +Noun +A3sg +P3sg +Abl
E y l ü l ’ ü n d e n

E y l ü l +Noun +A3sg +P3sg +Gen
E y l ü l ’ ü n ü n
```

In our implementation, we will be using Rule 13 being aware of the fact that months which also have proper noun readings will get two analyses for their nominative cases.

Another issue concerns year numerals. Consider the sentences in (23).

(4.46) i. *22'de doğdu.*
22-LOC was-born-A3sg
'S/he was born at/in 22.'

ii. *1982'de doğdu.*
1982-LOC was-born-A3sg
'S/he was born in 1982.'

The only difference between (i) and (ii) is the magnitude of the numbers. Although we have the impression that the former is not a year numeral but a clock-time, our morphology does not distinguish cardinal numbers. In other words, we do not get an additional '`+Num+Year`' analysis for any cardinal number. Choosing a time interval for year numerals (say 1000-2200) and having two analyses ('`+Num +Year`' and '`+Num +Card`') for the numbers in that interval (as English and German morphology do) would avoid the ambiguity in (i). However, this is not a realistic approach; the date of an historical event may fall outside of that interval (4.47).

(4.47) *İskenderiye Kitaplığı, ilk kez, MÖ 47'de Romalılar'ın Julius Cesar komutası altında Mısır'ı işgal etmeleri sırasında yakıldı.*
'The Alexandria Library was destroyed for the first time during the occupation of Egypt by Romans commanded by Julius Cesar in 47 B.C. '

Klasik çağ mimarlarından Vitruvius'un MÖ 25'te yazdığı varsayılan eseri 'De Architectura'dan yola çıkarak..
'Relying on the work of art that Vitruvius, one of the architects of the classical age, wrote in 25 B.C. . . . '

Moreover, having two analyses from the morphology is not the best solution as it may result in other ambiguities.

We define two short rules in Rule 14 and Rule 15. These rules will enable us to see which cardinal numbers stand for the day and year numerals of date expressions in the c-structure representation. They will be used in some other date phrase rules.

```
DAYnum -->
       NUM: ^ = !
            (^ NUMBER-TYPE)=c card
            @(BARE-TIME date).
```

Rule 14: Rule for day numerals

```
YEARnum -->
        (ADVdate: @ADJUNCT)
        NUM: ^ = !
             (^ NUMBER-TYPE)=c card
             @(BARE-TIME year).
```

Rule 15: Rule for year numerals

The ADVdate category that is called optionally before the year numeral in Rule 15 is a new category defined for the abbreviations *MÖ (milattan önce), İÖ (İsadan önce)* 'before Christ' and *MS (milattan sonra), İS (İsadan sonra)* 'after Christ'. (4.48) is the lexicon entry for *MÖ*:

(4.48) `MÖ ADVdate * @(PRED %stem).`

One can also analyze these abbreviations as postpositional phrases, introducing them to the lexicon of POSTPP (not POSTP) category. In that case, the expression *MÖ 1982'de* 'in 1982, B.C.' can be thought of a two-phrase expression: *milattan önce 1982'de* 'in 1982 before Christ'. This would be similar to parsing adjacent temporal expressions:

[öğleden önce] [saat onda]
[before noon] [at ten o'clock]
'at ten o'clock before noon'

But the abbreviated expressions like *MÖ* 'B.C.' are not used on their own. They always precede a year (or century) numeral. So we prefer a grouping like in (4.49)a instead of (4.49)b in our design.

(4.49) a. [in 57 B.C.] [when X became Y]

b. [in 57] [B.C.] [when X became Y]

The expression of time location in the year *1982'de* 'in 1982' is a derived noun. The `NPderiv` rule (Rule 3, Page 57) already parses this noun with the cardinal number stem, but as we already discussed above, it does not get any year distinction. Since we want to mark the year time expression by '`TIME year`', we rewrite the `NPderiv` rule in Rule 16:

```
NPderiv --> "2'de, 22.00'de, 1982'de"

            { "all types of numbers"
               NUM: (^ OBJ) = !
                    @(IF (! NUMBER-TYPE) =c time
                        @(TIME clock-time))
            | "for year numerals,unfortunately covers all cardinal numbers"
               YEARnum: (^ OBJ) = !
                         @(TIME year)
            }
            DS: ^ = !
                @(IF @(TIME year)
                    ~(^ SPEC POSS) ~@(NP-MARKER)).
```

Rule 16: Rule for derived noun phrases

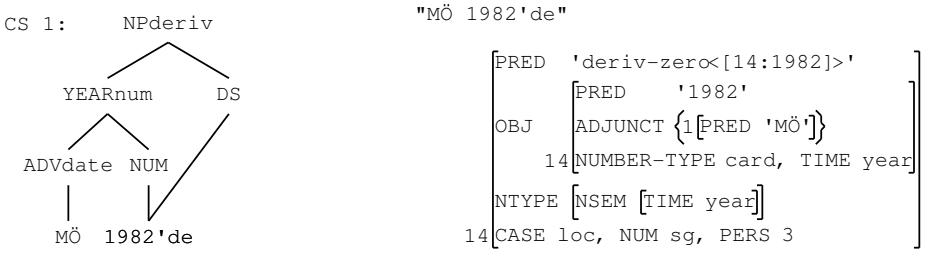


Figure 4.18: C and f-structures for *MÖ 1982'de* 'in 1982 B.C.'

Apparently, Rule 16 will give two solutions for every noun derived from a cardinal number. As year expressions do not get possesive tags, we restrict the numerals which will get the '`TIME year`' mark by an IF control. In this way, phrases like the ones given in (4.50) do not get a 'year' feature value.

(4.50) a. *kedilerden* ***ikisi***
cat-PL-LOC two-P3sg
'two cats out of ..'

b. *gecenin* ***üçü***
night-GEN three-P3sg
'three (o'clock) at night'

c. *nüfusun* ***%47'si***
population-GEN %47-P3sg
'47% of the population'

Besides, the disjunction in Rule 16 may be ignored with the constraint ∼@(TIME year) whenever the type of the cardinal number is known when writing the rules for phrases, for instance, in (4.51).

(4.51) a. ***altıda*** *bir*
six-LOC one
'one out of six'

b. ***on beşte*** *bir*
fifteen-LOC one
'once every fifteen days'

c. ***ikide*** *bir*
two-LOC one
'highly frequently'

In Turkish, the most common date expression including the day, month and the year information is formed putting one after another: *9 Eylül 1982* 'September the 9th, 1982'. When we want to analyze this expression in terms of modifier-modified constituents in a single level, Eylül 'September' seems to be the only possible head of the phrase. It may be considered that the day and year numerals both modify the month noun. Note that, here, the numerals behave as nouns; they do not specify the number of the month (i.e. not "how many Septembers"). Rule 17 parses the phrase *9 Eylül 1982* 'September the 9th, 1982' according to these criteria. Figure 4.19 shows the c- and f-structures of this approach.

```
DATEP -->

       "COM{EX RULE DATEP: 9 Eylül (1982)}"

        DAYnum: @MOD;
        MONTH: ^ = !
               { ~ (^ SPEC POSS) ~@(NP-MARKER)
               | (^ NTYPE CHECK NP-MARKER)=c + }   "3sg: 1982 yılı 9 Eylülü"
        (YEARnum: @MOD).                           "optional year: 9 Eylül"
```

Rule 17: Rule for date phrases

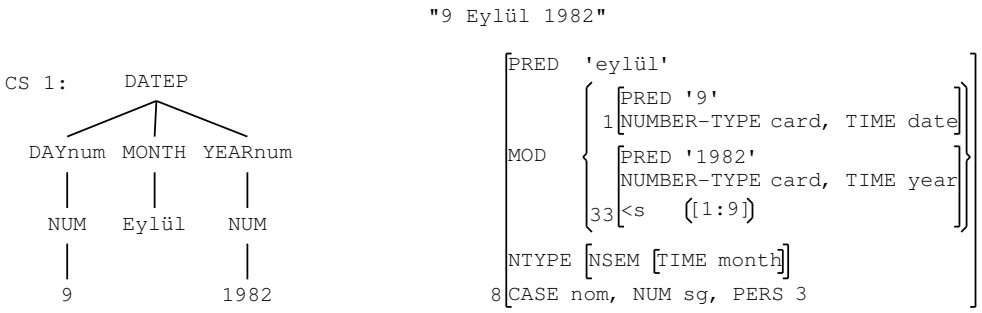


Figure 4.19: C and f-structures for *9 Eylül 1982* ‘September the 9th, 1982’

Although the representation for the nominative case form of the phrase seems reasonably fine, it becomes complicated in locative case where the year numeral changes its POS to noun. Figure 4.20 shows the c- and f-structures of *9 Eylül 1982’de* ‘on September the 9th, 1982’.

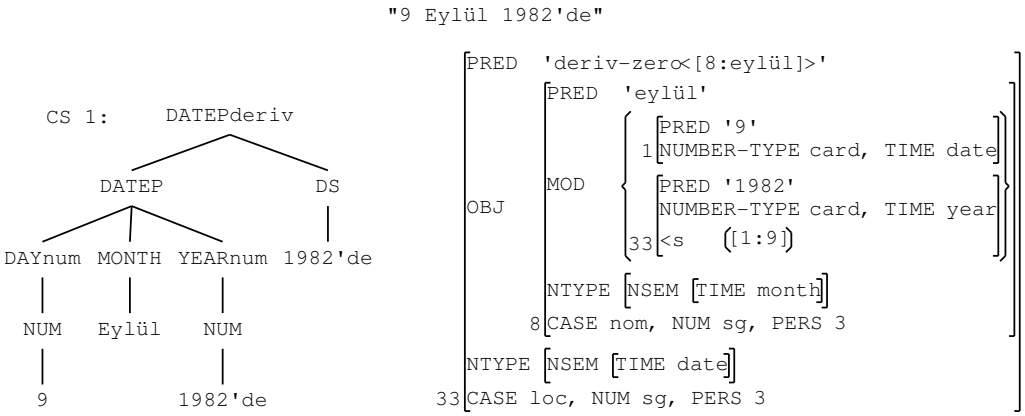


Figure 4.20: C and f-structures for *9 Eylül 1982’de* ‘on September the 9th, 1982’

In Figure 4.20, note that the zero derivation is shown on *Eylül* 'September'. This mismatch between the c- and f-structures is already discussed within the appositional approach for clock-time expressions in Section 4.2.1. This is in itself not a problem. But a two-step solution would lead to c- and f-structures which are parallel with the head marking.

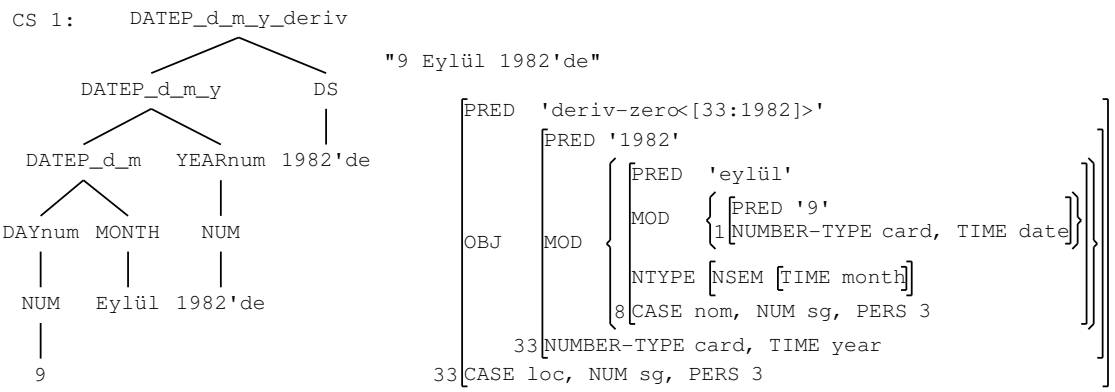


Figure 4.21: C and f-structures for *9 Eylül 1982'de* 'on September the 9th, 1982',2-step

In Figure 4.21, first, the day numeral and the month build a sub-phrase. Then this phrase modifies the year numeral and they together constitute the date phrase. In this case, the year numeral becomes the head of the main-phrase. So, zero derivation appears on the year. However, this representation is deeply nested.

In actual fact, there is no definite linguistic relation between the constituents of this calendar date phrase. Hence, we will assume this 3-word date expression as a single token and treat this pattern as a special case of noun, applying the same approach we used for clock-time (*saat iki*) or days of the week (*pazar günü*). The multiword transducer (Appendix E) labels the valid sequence by `+DateTime` tag. This tag is added to the lexicon as a noun type (4.52).

(4.52) `+DateTime N_TYPE_SFX XLE @(TIME date).`

The date expressions in the form of *9 Eylül 1982* can now be parsed by the `DATE` sublexical rule, Rule 18, which will then be called by the `NP` rule. As the whole expression is considered as a noun, the suffixes do not bother any more, unnecessary derivations do not show up. Figure 4.22 shows the most proper and final representation.

```
DATE -->
        "COM{EX RULE NP: 9 Eylül 1982}"

         DATE_BASE
         N_SFX_BASE
         N_TYPE_SFX_BASE
         NUM_PERS_SFX_BASE
         POSS_SFX_BASE
         CASE_SFX_BASE.
```

Rule 18: Sublexical rule for dates

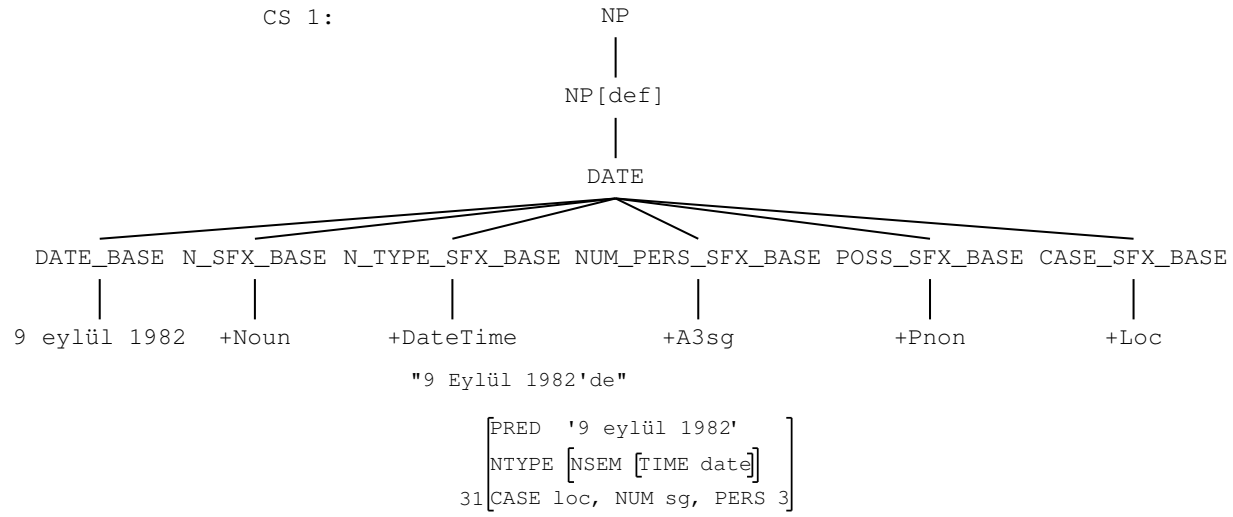


Figure 4.22: C and f-structures for *9 Eylül 1982'de* '9 September 1982', multiword

The expression *Eylül 1982* 'September, 1982' where the day information is skipped may be analyzed in two ways. In the first, the month noun may be assumed to be labeled by the year numeral. There, an appositional DATEP headed by *Eylül* 'September' is to be built. But the locative case marked form of this phrase, *Eylül 1982'de* 'in September, 1982' brings again the discussion of zero derivation[12] appearance on the month noun in the f-structure.

[12]Zero derivation representation may be reconsidered, a non-embedded representation may be employed.

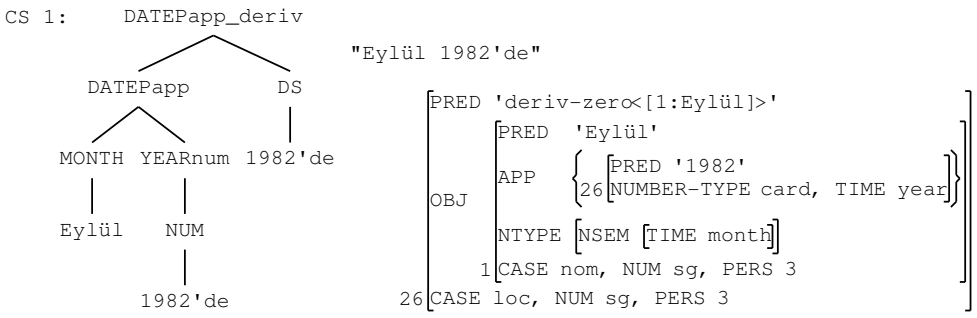


Figure 4.23: C and f-structures for *Eylül 1982'de* 'in September, 1982'

In the second approach, we change the year numeral into optional in the transducer we wrote for the expressions in the form of *9 Eylül 1982* 'September the 9th, 1982' and evaluate *Eylül 1982* 'September, 1982' as a 2-word datetime noun .

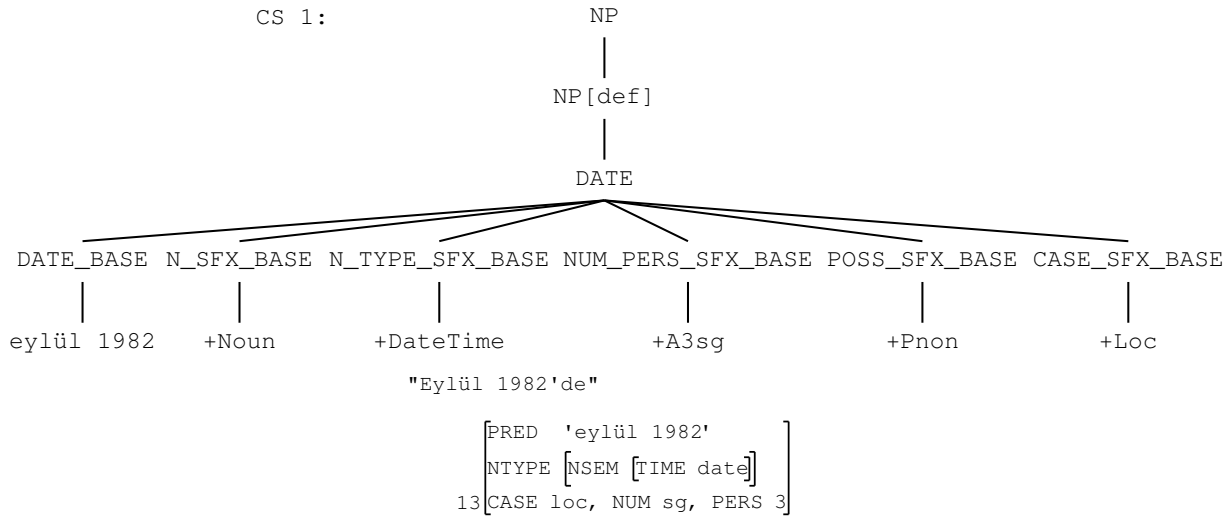


Figure 4.24: C and f-structures for *Eylül 1982'de* 'in September, 1982', multiword

The abbreviated date expression 9/9/1982 is taken as 5 tokens from the tokenizer:

```
% tokens 9/9/1982
9 @ / @ 9 @ / @ 1982 @
```

The punctuation mark "/" gets only a `+Punct` tag from the morphological analyzer. Before we merge those 5 tokens by a multiword transducer, a distinctive tag for "/" is needed. The line given in 4.53 is added to the text-specified transducer which is the first transducer in morphology use-first section (overwrites the existing analysis).

(4.53) `/ : /'// +Punct +Slash`

A new multiword transducer (Appendix F) defines the sequences like 9/9/1982 as a noun of type `DateTime`, consistent with the expanded version, *9 Eylül 1982* 'September the 9th, 1982'.

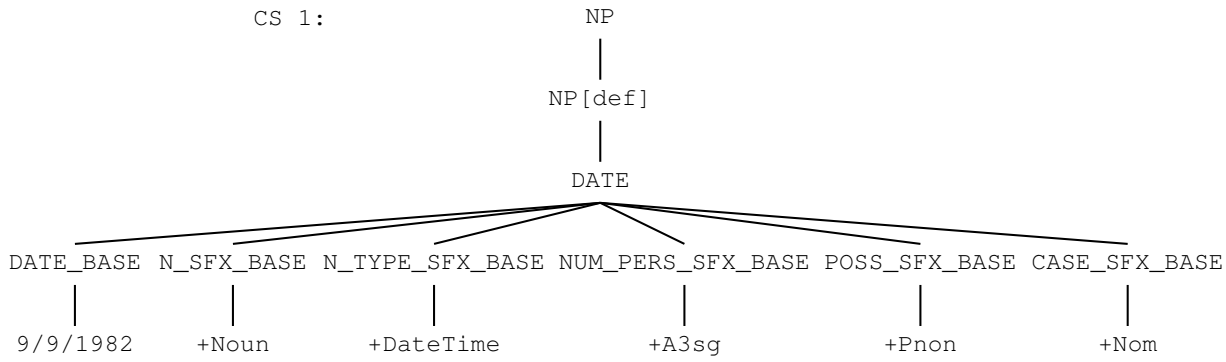


Figure 4.25: C-structure for 9/9/1982, expanded

The c- and f-structure representations of *9/9/1982'de* 'on 9/9/1982' are given in Figure 4.26.

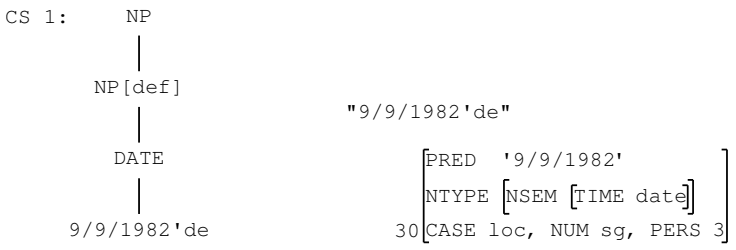


Figure 4.26: C and f-structures for *9/9/1982'de* 'on 9/9/1982'

The numeral expression of *Eylül 1982*, 9/82, did not get any result from the Turkish corpus query. But such patterns may similarly be parsed making the day related part (9/) optional in the design for the solution of patterns like 9/9/1982.

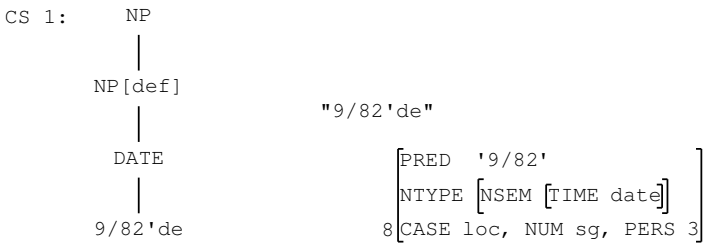


Figure 4.27: C and f-structures for *9/82'de* 'on 9/82'

Other cases where two numbers are joint by "/" (4.54) will require a separate fraction analysis (with +Fract).

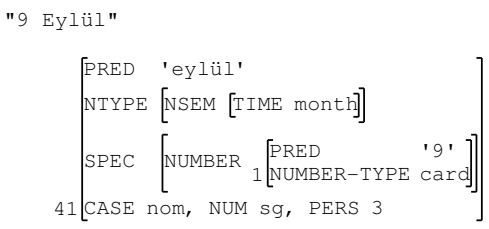
(4.54) a. *Mercedes sürücüsü 8/8 kusurlu bulundu.*
'The Mercedes driver was 8/8 faulty.'

b. *Karar No: 1963/83.*
'Judgement No: 1963/83.'

c. *Hamamın 1/150 boyutlarında maketini çalıştık.*
'We designed a 1/150 scaled model of the Turkish bath.'

Without the year information, the date expression *9 Eylül* 'September the 9th, has two readings:

(4.55) 1. *9 Eylül* '9 Septembers'
Istanbul'da 2 yıl kaldım. 2 Eylülde de sıcaklık 20 derecenin üzerindeydi.
'I stayed 2 years in Istanbul. The temperature was over 20 degrees in both Septembers.'



2. *9 Eylül* 'September the 9th'
9 Eylülde yeniden Istanbul'da olacağım.
'I will be again in Istanbul on the 9th of September.'

For the date reading, either the day numeral (Rule 14) may modify the month (DATEP, Figure 4.28) or the composing words of the phrase may be combined as a single token (see Appendix D) and treated as a date type noun (DATE, Figure 4.29). We will apply the latter.

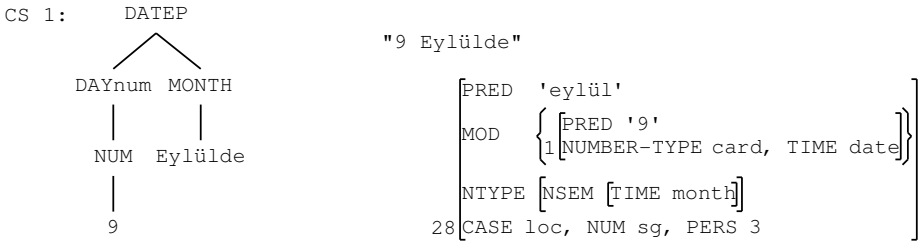


Figure 4.28: C and f-structures for *9 Eylülde* ‘on the 9th of September’, Modifier

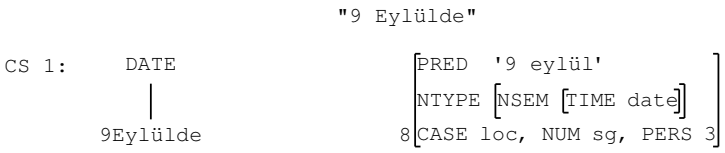


Figure 4.29: C and f-structures for *9 Eylülde* ‘on the 9th of September’, multiword

The date expressing phrase may be used as the modifier of a noun phrase headed by the word *tarih* ‘date’. We write a separate `DATEPdate` rule in Rule 19.

```
DATEPdate -->

        "9 Eylül(1982) 9/9/1982 tarihinde"

         DATE: @MOD;
         N[indef]: ^ = !
                   (^ PRED)=c ‘tarih’
                   (^ NTYPE CHECK NP-MARKER)=c +.
```

Rule 19: Rule for date phrases headed by the noun *tarih* ‘date’

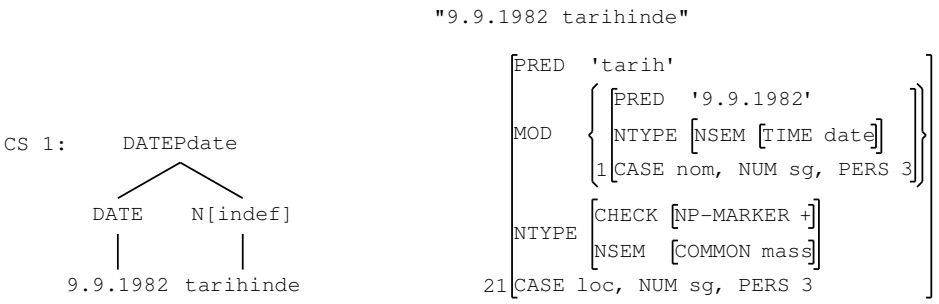


Figure 4.30: C and f-structures for *9.9.1982 tarihinde* 'on the date 9.9.1982'

The date expressions in (4.56) and (4.57) will be parsed by Rule 20 and Rule 21 respectively.

(4.56) *9 Eylül (1982)* *Pazartesi/Pazartesi günü günü/sabahı/akşamı/gecesi*

(4.57) *9 Eylül (1982)Pazartesi günü* *sabahı/akşamı/gecesi*

```
DATEPday -->

        "COM{EX RULE DATEPday: 9 Eylül 1982 Pazartesi}"
        "COM{EX RULE DATEPday: 9/9/1982 Pazartesi günü}"
        "COM{EX RULE DATEPday: 9 Eylül günü/sabahı etc}"

        DATE: @MOD;
        { "Pazartesi, Pazartesi günü"
           DAY
        | "günü/sabahı/akşamı/gecesi"
           N[indef]: ^ = !
                     { (^ PRED FN)=c gün
                     | (^ NTYPE NSEM TIME)=c day-part
                     }
                     (^ NTYPE CHECK NP-MARKER)=c +
        }.
```

Rule 20: Rule for date expressions with date and day information

```
DATEPdaypart -->

        "COM{EX RULE DATEPdaypart: 9 Eylül Pazar akşamı}"
        "COM{EX RULE DATEPdaypart: 9 Eylül 1982 Pazar günü sabahı}"

        DATEPday: @MOD;
        N[indef]: ^ = !
                  (^ NTYPE NSEM TIME)=c day-part
                  (^ NTYPE CHECK NP-MARKER)=c +.
```

Rule 21: Rule for date expressions with date and day-part information

C- and f-structures for the expressions *9 Eylül 1982 Pazar günü* ‘on Sunday, September the 9th, 1982’ and *9 Eylül Pazar sabahı* ‘in the morning on Sunday, September the 9th’ are shown in Figure 4.31 and Figure 4.32.

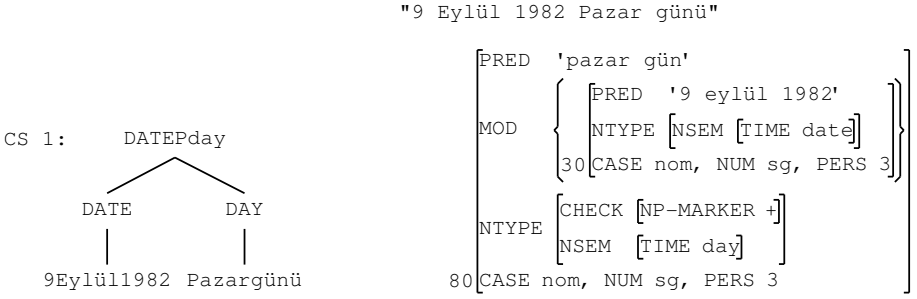


Figure 4.31: C and f-structures for *9 Eylül 1982 Pazar günü* ‘on Sunday, September the 9th, 1982’

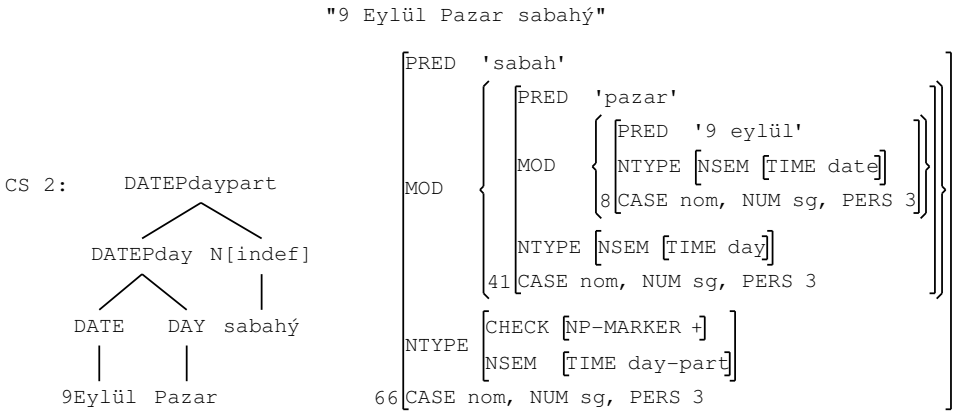


Figure 4.32: C and f-structures for *9 Eylül Pazar sabahı* ‘in the morning on Sunday, September the 9th’

It is clear that some date expressions may be parsed by the existing NP rules. But there we get so many spurious analyses. Constraining those rules regarding the requirements of date expressions results in losing analyses for some other noun phrases.

(4.58) (*9 Eylül*) (*1982 Pazartesi*) *günü*
(9 September) (1982 Monday) day-P3sg

In order not to get an analysis as illustrated in (4.58), one can restrict the relevant rule saying the days of the week can not get a numeral specifier. But then, the parse for the case in (4.59) would have been missed.

(4.59) ***2 Pazartesi*** *günü de benim için uygun değil, Salıları olsa?*
'**Both Mondays** are not ok for me, what about Tuesdays?'

In addition, there are cases like where we want to add a `TIME` mark (4.60i), but this is not possible due to the other reading of the phrase (4.60ii):

(4.60) i. *1982 Eylül ayının dokuzunda* 'in 9 Septembers out of 1982'
ii. *1982 Eylül ayının dokuzunda* 'on the ninth of the month September of 1982'

More complicated date expressions are composed by the phrasal uses of the year and month information within each other. A systematic way with nested rules will be followed to analyze those expressions.

First, Rule 22 is written, telling that the numeral in the expression *1982 yılı* 'the year 1982' is a year numeral. This numeral modifying the noun *yıl* (or *sene*) 'year' has a meaning other than in the phrase *(onun) 1982 yılı* 'his 1982 years'. That is, 1982 is not a `SPEC NUMBER` but a `MOD`.

```
DATEPyear -->

          "COM{EX RULE DATEPyear: (MÖ) 1982 yılı}"

          YEARnum: @MOD;
          N[indef]: ^ = !
                    (^ PRED FN )$c {yıl sene}
                    (^ NTYPE CHECK NP-MARKER)=c +.
```

Rule 22: Rule for year numerals compounded with the noun *yıl* ‘year’

Eylül ayı ‘the month September’ or the *dokuzuncu ay* ‘the ninth month’ are the elements of the `DATEPmonth` rule, Rule 23.

```
DATEPmonth --> "Eylül ayı/dokuzuncu ay"

          { "COM{EX RULE DATEPmonth: Eylül ayı}"

            MONTH: @MOD;
            N[indef]: ^ = !
                      (^ PRED FN)=c ‘ay’
                      (^ NTYPE CHECK NP-MARKER)=c +

          | "COM{EX RULE DATEPmonth:dokuzuncu ay}"

            Aord: @ADJUNCT
                  @(ATYPE_desig ! attributive);
            N[indef]: ^ = !
                      (^ PRED FN)=c ‘ay’
                      { ~ (^ SPEC POSS) ~@(NP-MARKER)
                      | (^ NTYPE CHECK NP-MARKER)=c + }
          }.
```

Rule 23: Rule for months compounded with the noun *ay* ‘month’

Note in the `DATEPmonth` rule that ordinal numbers are given the category `Aord` and end up as ADJUNCT in the f-structures rather than SPEC NUMBER as cardinals would do. As we get NUM analysis for ordinals from the morphology, we write the sublexical rule `Aord` in Rule 24. The unknown entry in (4.61) assigns `Aord` as an attributive adjective.

```
Aord -->
         "COM{EX RULE Aord: ikinci}"

          Aord_BASE
          NUM_SFX_BASE
          ORD_TYPE_BASE.
```

Rule 24: Sublexical rule for ordinal numbers

(4.61) `-unknown`

```
Aord     XLE     @(ADJ %stem)     @(ATYPE attributive); ETC.
```

Now, we can build different combinations of year, month and day phrases. The classification of these phrases is mostly based on the expressions encountered in the METU corpus. Some sample sentences may be found in Appendix G. We do not give the c and f-structures for every expression enclosed by the coming rules. Appendix H presents a test suite with the corresponding c- and f-structure representations of more expressions.

The informal date expressions including the year and month information are summarized in (4.62). `DATEPyearmonth` given in Rule 25 covers these phrases.

(4.62) *1982/1982'nin/1982 yılı/1982 yılının* *Eylülü*
Eylül ayı

The year information modifying the month information is written as a macro (4.63) so that it may then be used also in Rule 27 where the day information is appended.

(4.63) `beforeMonth =`

```
{ "1982 yılı(nın)"
  DATEPyear: @MOD
             (! CASE)$c {nom gen};
 |"1982"
  YEARnum: @MOD;
  (COMMA)  "1982, 9 Eylülünde"
 |"1982'nin"
  NPderiv: @MOD
           (! NTYPE NSEM TIME)=c year
           ~(! SPEC POSS)
           (! NTYPE CHECK NP-MARKER) ~=c +
           (! CASE)=c gen
}
```

```
DATEPyearmonth -->

          "1982(nin)/1982 yılı(nın) ..."

          @(beforeMonth)
          { "Eylülü"
            MONTH: ^ =!
                   { ~ (^ SPEC POSS) ~@(NP-MARKER)
                   | (^ NTYPE CHECK NP-MARKER)=c + }
                   "none:{NPdate: 1982 yılının Eylül(ünün) sonunda}"

           |"Eylül ayı/dokuzuncu ayı"
            DATEPmonth
          }
```

Rule 25: Rule for date expressions with year and month information

Date expressions that do not contain a year information are summarized in (4.64). They will be parsed by Rule 26.

(4.64) *Eylülün/Eylül ayının/dokuzuncu ayın/ayın* *dokuzu*
9. günü
ilk haftası, haftasonu
ikinci Pazarı
ilk Pazartesi günü

```
DATEPmonthday -->

          ({ "Eylül(ün)"
             MONTH: @MOD
                    (! CASE)=c gen
           | "Eylül ayı(nın), dokuzuncu ay(ın)"
             DATEPmonth: @MOD
                         (!CASE)=c gen
           | "ayın"
             N[indef]: @MOD
                       (! NTYPE NSEM TIME)=c +
                       (! PRED FN)=c 'ay'
                       (! CASE)=c gen
           })
           @(afterMonth).
```

Rule 26: Rule for date expressions with month and day information

The day (or week) information coming after the modifying month is given by the macro `afterMonth` in (4.65). This macro will be used in Rule 27 as well.

(4.65) `afterMonth =`

```
{ "dokuzu"
   NPderiv: ^ = !
            (^ OBJ NUMBER-TYPE) =c card
            (^ NTYPE CHECK NP-MARKER)=c +
            @(BARE-TIME_desig ! date)
| "(ilk, ikinci, son) haftası/Pazartesi günü"
  NPadj[indef]: ^ = !
                (^ ADJUNCT ATYPE)=c attributive
                { (^ NTYPE NSEM TIME)=c day
                | (^ PRED FN)$c {hafta haftason gün}}
                (^ NTYPE CHECK NP-MARKER)=c +
}.
```

The deictic expressions (4.66) are also covered by Rule 26 thanks to the optionality of the month information part.

(4.66)

onunda	‘on the tenth’
ikinci Pazar günü	‘on the second Sunday’
ilk haftasonunda	‘on the first weekend’

Finally `DATEPyearmonthday` given in Rule 27 covers the expressions that include the year, month and day information together.

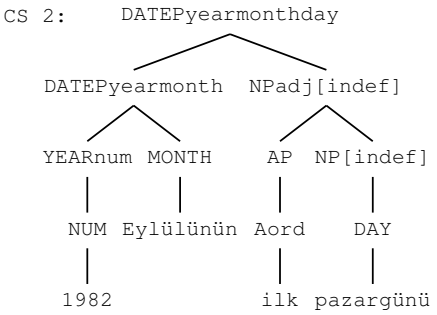
```
DATEPyearmonthday -->

        { "1982(nin)/1982 yılı(nın) Eylülünün/Eylül ayının/dokuzuncu ayın dokuzu"

          DATEPyearmonth: @MOD
                          (! CASE)=c gen;
          @(afterMonth)
        | "1982(nin), 1982 yılı(nın) 9 Eylülü"
          @(beforeMonth)
          { "9 Eylülü"
             DATE: ^ = !
                   (^ NTYPE CHECK NP-MARKER)=c +
          | "9 Eylül Pazartesi (günü)"
             DATEPday
          }
        }.
```

Rule 27: Rule for date expressions with year, month and day information

Some sample c- and f-structures are given in Figure 4.33, Figure 4.34, Figure 4.35 and Figure 4.36.



"1982 Eylülünün ilk pazar günü"

```
[ PRED    'pazar gün'
  ADJUNCT { 38 [ PRED 'ilk'
                 ATYPE attributive, DEGREE positive ] }
  MOD     { [ PRED  'eylül'
              MOD   { 1 [ PRED '1982'
                          NUMBER-TYPE card, TIME year ] }
              NTYPE [ CHECK [NP-MARKER +]
                      NSEM  [TIME month] ]
            28 [ CASE gen, NUM sg, PERS 3 ] ] }
  NTYPE   [ CHECK [NP-MARKER +]
            NSEM  [TIME day] ]
55[ CASE nom, NUM sg, PERS 3 ]
]
```

Figure 4.33: C and f-structures for *1982 Eylülünün ilk pazar günü* ‘on the first Sunday of September, 1982’

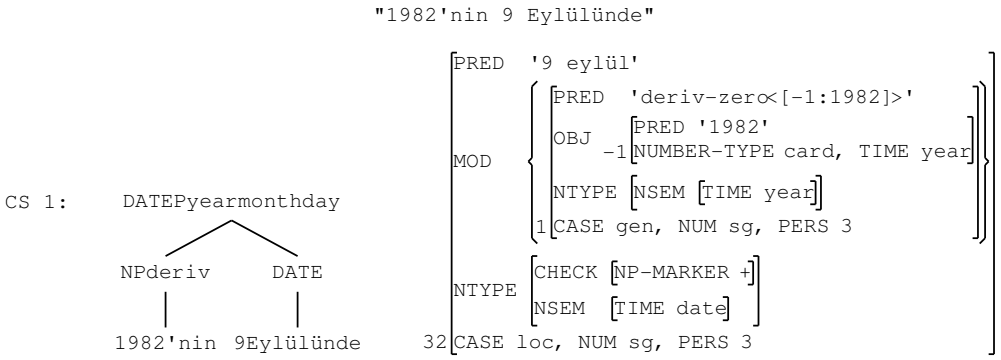


Figure 4.34: C and f-structures for *1982'nin 9 Eylülünde* 'on the 9th of September of 1982'

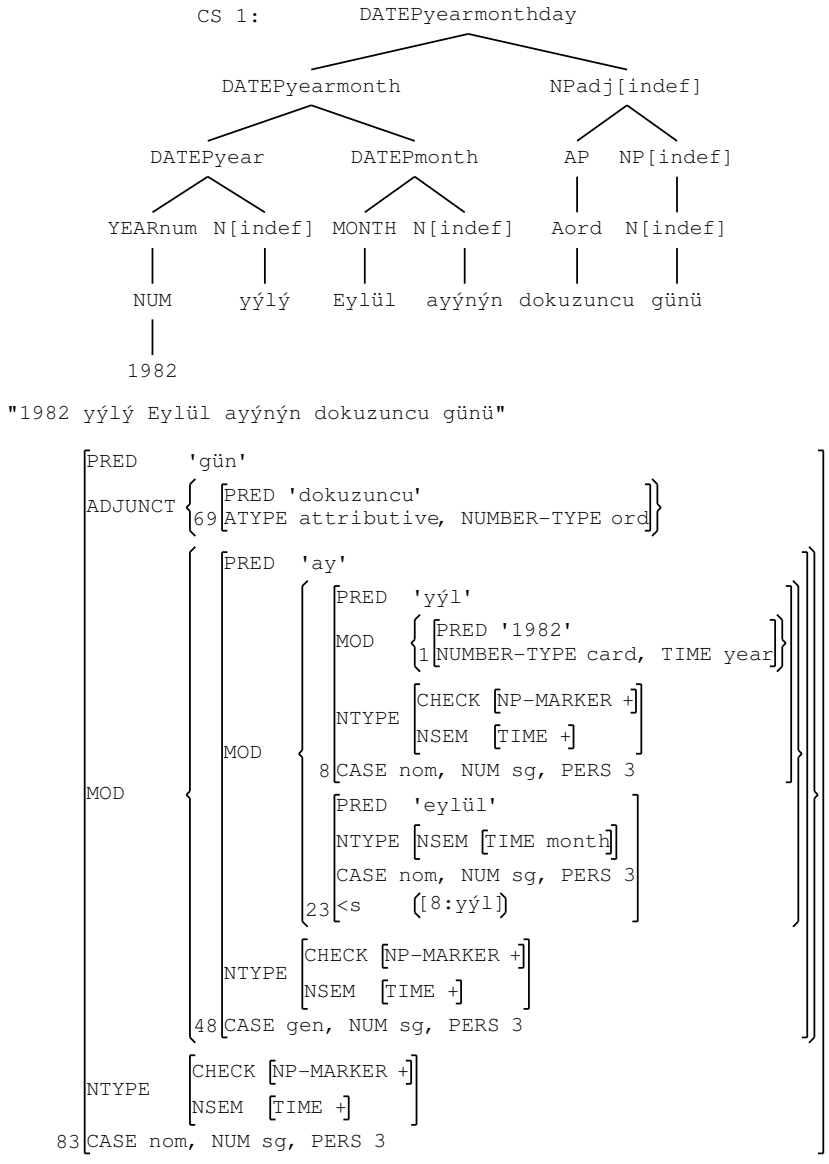


Figure 4.35: C and f-structures for *1982 yılı Eylül ayının dokuzuncu günü* 'on the ninth day of the month September of the year 1982'

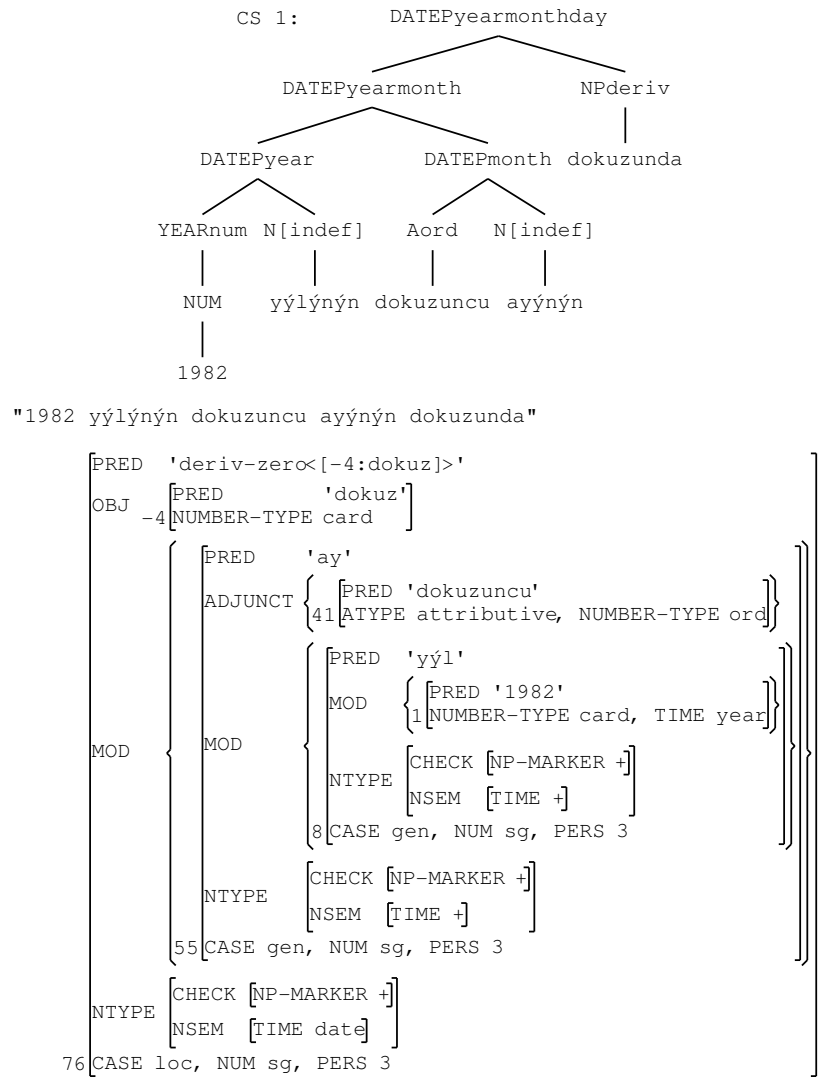


Figure 4.36: C and f-structures for *1982 yılının dokuzuncu ayının dokuzunda* ‘on the ninth of the ninth month of the year 1982’

Rule 29 will be parsing the date expressions including the century information. The century numeral has to be an ordinal number. It is marked as `century` as the value of the `TIME` feature in Rule 28.

```
CENTURY_num -->

            Aord: @ADJUNCT
            @(ATYPE_desig ! attributive)
            @(BARE-TIME century).
```

Rule 28: Rule for century numerals

```
DATEPcentury -->

            "COM{EX RULE DATEPcentury: (Mö) 7. yüzyıl}"

            (ADVdate: @ADJUNCT)
            CENTURY_num
            N[indef]: ^ = !
                      (^ PRED FN )$c {yüzyıl asır}
                      ~(^ SPEC POSS) ~@(NP-MARKER).
```

Rule 29: Rule for date expressions with century information

C- and f-structures of the phrase *MÖ 7. yüzyılda* ‘in the 7th century B.C.’ is given in Figure 4.37.

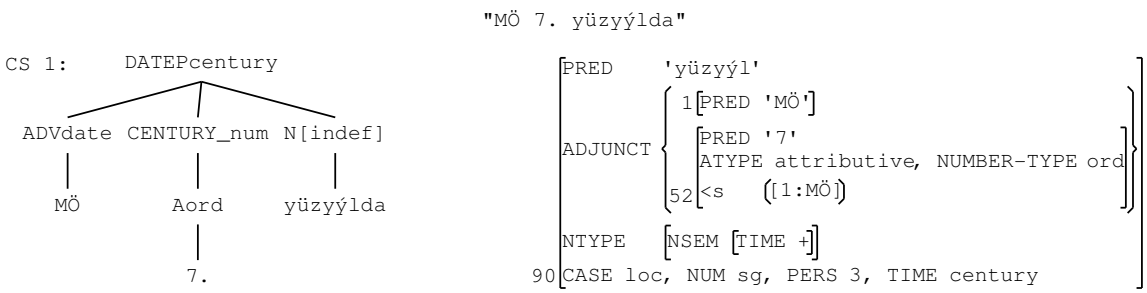


Figure 4.37: C and f-structures for *MÖ 7. yüzyılda* ‘in the 7th century B.C.’

Date expressions including the phase of the time point are sampled in Section 3.3. Such phrases are headed by the nouns given in (4.67).

(4.67)	*baş(lar)ında*	‘at the beginning’
	orta(lar)ında	‘in the midst/mid-something’
	son(lar)ında	‘at the end’
	(ilk/ikinci/son) yarısında	‘in the first/second/last half’

The lexicon entries of the nouns *baş* ‘beginning’, *orta* ‘middle’, *yarı* ‘half’ and *son* ‘end’ do not contain a `TIME` mark as they may be indicating location as well (4.68).

(4.68) *sokağın başında* ‘at the beginning of the street’
yolun ortasında/yarısında ‘in the middle of the road’
koridorun sonunda ‘at the end of the hall’

We mark those nouns with `TIME +` while writing the rule, Rule 30, to analyse the approximate date expressions (see Section 3.3).

```
DATEPapprox -->

          { "1982(nin)/1982 yılı(nın)"
            beforeMonth: @MOD
          | "Eylül"
            MONTH: @MOD
                   ~(! SPEC POSS) ~@(NP-MARKER)
                   (! CASE)$c {nom gen}
          | "Eylül ayı(nın)"
            DATEPmonth: @MOD
                        (! CASE)$c {nom gen}
          | "1982(nin)/1982 yılı(nın) Eylül(ünün)/Eylül ayı(nın)"
            DATEPyearmonth: @MOD
                            (! CASE)$c {nom gen}
          | "MÖ 4. yüzyılın"
            DATEPcentury: @MOD
                          (! CASE)$c {nom gen}
          }
          ( ADVfoc: @ADJUNCT) "hemen, tam"
          ( Aord: @ADJUNCT
                  (! PRED FN)$c {ilk ikinci son} )
          "başi/sonu/ortası/yarısı"
          N[indef]: ^ = !
                    (^ PRED FN)$c {baş orta son yarı}
                    (^ NTYPE CHECK NP-MARKER)=c +
                    @(TIME_desig ! + ).
```

Rule 30: Rule for approximate date expressions

In Rule 30 the optional focus-type adverbial modifying the head noun is introduced to the lexicon as `ADVfoc` category (4.69):

(4.69) `hemen ADVfoc * @(ADV-FOC %stem) @(TEMP immediate).`

Figure 4.38 presents the c and f-structures for the phrase *1980lerin hemen başında* ‘right at the beginning of 1980s’.

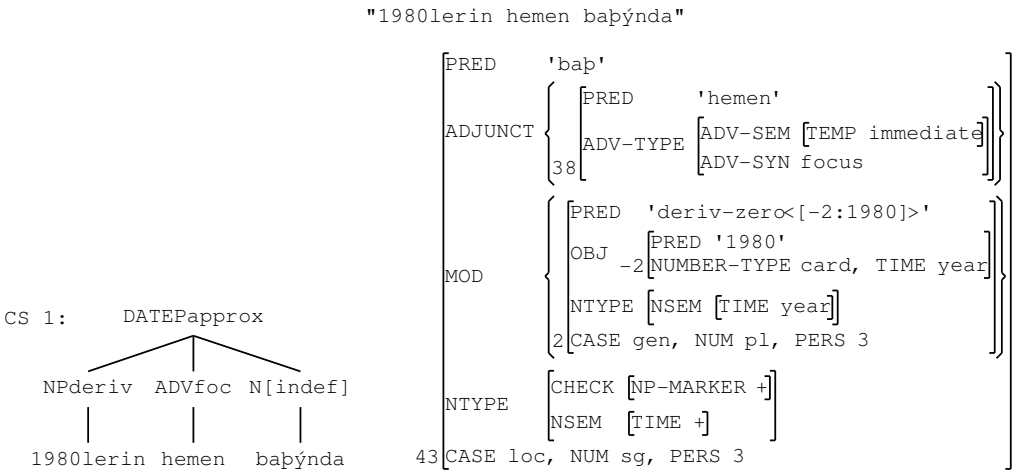


Figure 4.38: C and f-structures for *1980lerin hemen başında* ‘right at the beginning of 1980s’

The date expressions split into separate rules for easier control will now be put under `NPdate` rule, Rule 31.

```
NPdate -->

        { "Eylülde"
          MONTH
        | "9 Eylülde, 9 Eylül 1982'de"
          DATEP: ~(^ SPEC POSS) ~@(NP-MARKER)
        | "9 Eylül 1982 Pazar günü"
          DATEPday
        | "9 Eylül Pazar sabahı"
          DATEPdaypart
        | "Eylül ayı"
          DATEPmonth
        | "Eylülün, Eylül ayının, dokuzuncu ayın dokuzu"
          DATEPmonthday
        | "1982 yılında"
          DATEPyear
        | "1982(nin), 1982 yılı(nın) Eylülü, Eylül ayı"
          DATEPyearmonth: (^ NTYPE CHECK NP-MARKER)=c +
        | "1982(nin)/1982 yılı(nın) Eylülünün/Eylül ayının dokuzu/9 Eylülü"
          DATEPyearmonthday
        | "9 Eylül (1982) tarihinde"
          DATEPdate
        | "MÖ 9. yüzyılda"
          DATEPcentury
        | "Eylül(ün), Eylül ayı(nın), 1982(nin), 1982 yılı(nın),
           1982(nin)(yılı(nın)) Eylül(ü)(ayı(nın))"
          DATEPapprox
        }.
```

Rule 31: Rule for date expressions

Note that, the CASE feature of the expressions are not constrained to the locative value in the NPdate rule. That is because these expressions may also serve as the subject or the object of the sentence (4.70).

(4.70) ***11 Ocak*** *çok önemli.*

'**January the 11th** is very important.'

*Bakanlar Kurulu'nun **28 Kasım Cuma gününü** tatil etmesi durumunda kamu çalışanları, 1 Aralık Pazartesi günü dönecekler.*

'The public workers will return back to work from their holidays on Monday, the 1st of December only if the council of ministers decide the **Friday, the 28th of November** be an off day.'

Apparently, NPdate should be called by the general NP rule. In order not to get the parses that NP rule offers for the expressions we covered by NPdate, we disprefer the NP rule solutions. In other words, if we get more than one analysis for an expression, we prefer the ones that are parsed by the NPdate rule. This is simply done by "OT-Marks (Optimality Theory Marks)". We mark both rules by the OT-MARK template with the arguments NP_rule and NPdate_rule respectively. The precedence of these rules is determined in the OPTIMALITY ORDER section in the configuration file.

(4.71)
```
OPTIMALITYORDER      NP_rule < NPdate_rule >.
```

Recall the POSTPP rule given in Rule 5 on page 63. The object that postpositions subcategorize for was given as the general NP category (or null). As NPdate is to be called by NP, post-positional phrases with date expressions (see Section 3.3) will be enclosed automatically.

We extend the Rule 5 saying that the post-positional phrase is temporal if the object is of TIME type. The POSTPP rule is re-written in Rule 32.

```
POSTPP -->

        "COM{EX RULE POSTPP: 9 Eylül Pazar gününden itibaren}"
        "COM{EX RULE POSTPP: 9 Eylül 1982'den önce}"
        "COM{EX RULE POSTPP: 1982 yılı Eylül ayı sonuna kadar}"
        "COM{EX RULE POSTPP: 9/9/1982 tarihine dek}"

        NP: (^ OBJ) = !
            (^ P-SEM)= temp;
        ( @(MEASUREP) );     "9/9/1982'den birkaç gün önce"
        POSTP.
```

Rule 32: Modified rule for postpositional phrases

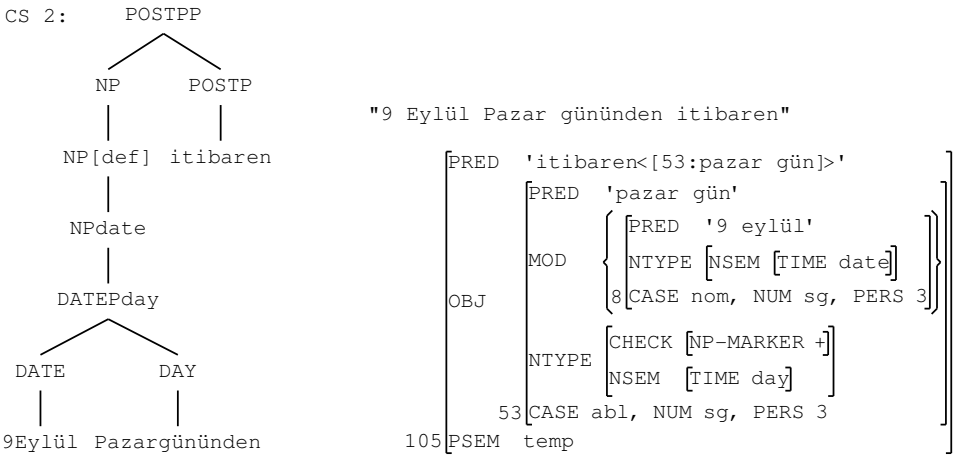


Figure 4.39: C and f-structures for *9 Eylül Pazar gününden itibaren* 'from Sunday, the 9th of September on'

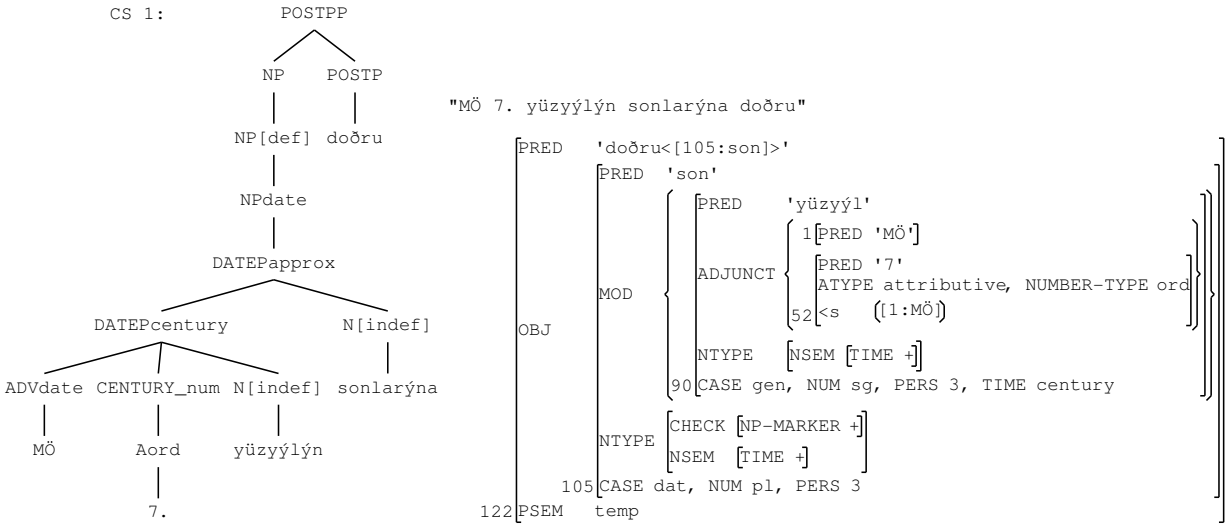


Figure 4.40: C and f-structures for *MÖ 7. yüzyılın sonlarına doğru* 'towards the end of the 7th century B.C.'

In the locative case[13], `NPdate` will be functioning as a temporal adverbial at the sentence level.

[13]except patterns ending with "DAY günü/sabahı.."

4.2.4 Seasons

The names of the seasons, *ilkbahar (bahar)* ‘spring’, *yaz* ‘summer’, *sonbahar (güz)* ‘autumn’ and *kış* ‘winter’ are analyzed as being of the category `N` by the morphology (4.72).

(4.72)
```
xfst[1]: up ilkbahar
         ilkbahar+Noun+A3sg+Pnon+Nom
```

So, we simply introduce them to the lexicon as nouns of type ‘`TIME season`’.

(4.73)
```
ilkbahar      N      XLE      @(NOUN ilkbahar)  @(TIME season).
```

The c- and f-structures for the temporal expression *ilkbaharda* ‘in spring’ is given in Figure 4.41.

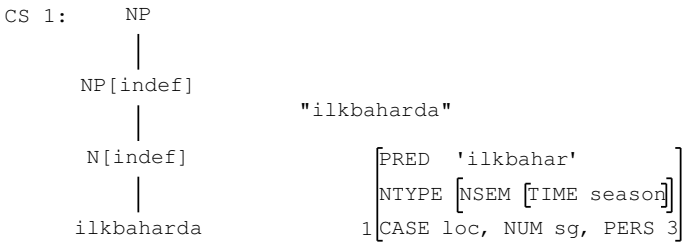


Figure 4.41: C and f-structures for *ilkbaharda* ‘in spring’

As exemplified in Section 3.4, adverbs with particular forms are used when expressing location on *yaz* ‘summer’ and *kış* ‘winter’ (and *güz* if it is used instead of *sonbahar* ‘autumn’):

```
xfst[1]: up yazın
         yazın+Adverb
xfst[1]: up kışın
         kışın+Adverb
xfst[1]: up güzün
         güzün+Adverb
```

Since these expressions are not nouns but adverbs, to ensure consistency with similar temporal expressions we define a temporal feature for adverbials and add a season value in the set (Section 4.1.2). The lexicon entry for *yazın* ‘in summer’ is given in (4.74).

(4.74) `yazın  ADV  XLE  @(ADV yazın)  @(TEMP season).`

Expressions where season names are compounded with the word *mevsim* ‘season’ are already parsed as expected by the existing `NPnn` rule from [Çetinoğlu and Oflazer, 2006a]. When we enter the noun *mevsim* to the lexicon with ‘`TIME +`’ we get the structures given in Figure 4.42.

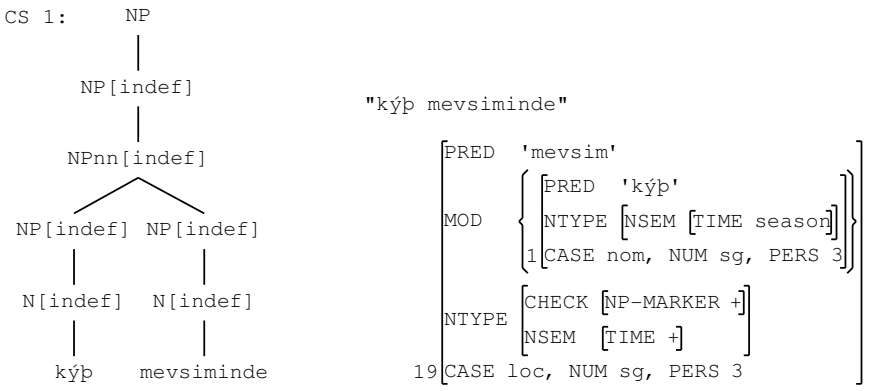


Figure 4.42: C and f-structures for *kış mevsiminde* ‘in the winter season’

If the season information is specified by the year, the rule `DATEPseason` in Rule 33 analyzes the expressions in (4.75).

(4.75) *1982(’nin)/1982yılı(nın)* *yazında*
kışında
baharında/ilkbaharında
sonbaharında/güzünde

```
DATEPseason -->

            "1982('nin)/1982yılı(nın).."
            @(beforeMonth)

            "..sonbaharı"
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c season
                      (^ NTYPE CHECK NP-MARKER)=c +.
```

Rule 33: Rule for expressions with date and season information

DATEPseason is called by the NPdate rule which was already called by the general NP rule (see Section 4.2.3).

The c and f-structures for *1982 yılı baharında* 'in spring of the year 1982' is shown in Figure 4.43.

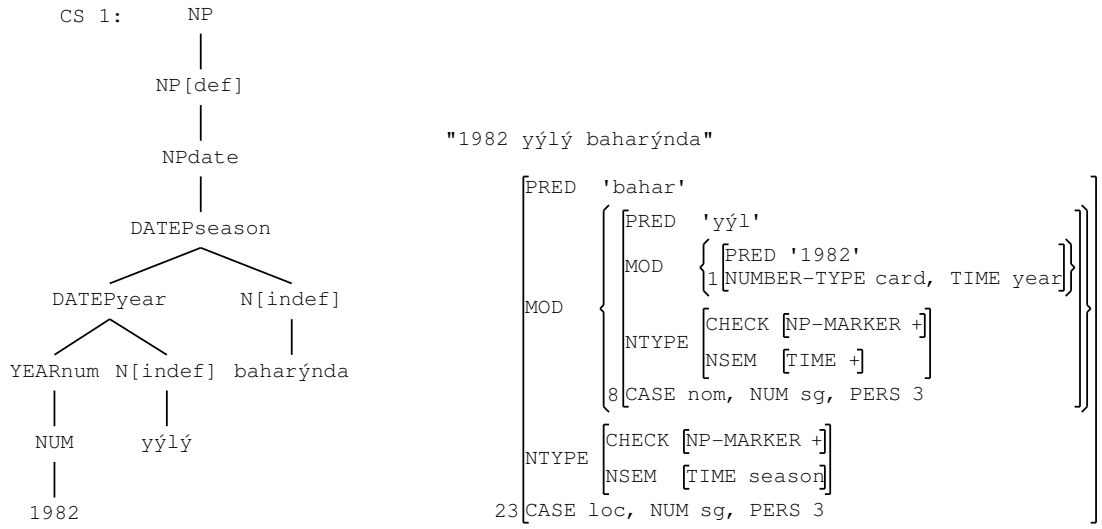


Figure 4.43: C and f-structures for *1982 yılı baharında* 'in spring of the year 1982'

4.2.5 Other Expressions

The following rule, NPadv, will be parsing the adverbial noun phrases.

```
NPadv -->

          { "dün bugün yarın"
            Ndate: ^ = !
                   (^ CASE)= nom
                   ~(^ SPEC POSS) ~@(NP-MARKER)
          | "sabah öğlen akşam gece akşamüstü geceyarısı"
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c day-part
                      { "sabah sabaha sabahtan"
                        (^ PRED FN)$c {sabah öğlen akşam gece}
                        ~(^ SPEC POSS) ~@(NP-MARKER)
                        (^ CASE)$c {nom dat abl}
                      | "akşamüstü"
                        (^ PRED FN)$c {akşamüst akşamüzer geceyarı}
                        (^ NTYPE CHECK NP-MARKER)=c +
                        (^ CASE)= nom
                      | "oğlende"
                        (^ PRED FN)=c öğlen
                        ~(^ SPEC POSS) ~@(NP-MARKER)
                        (^ CASE)= loc
                      }
          | "dün akşam"
            Ndate: @MOD;
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c day-part
                      ~(^ SPEC POSS) ~@(NP-MARKER)
          |
            DAY: ^ = !
                 { "Pazartesi, Pazartesileri"
                   (^ CASE)= nom
                   (^ POSS)$c {none 3pl}
                 | "Pazartesiye"
                   (^ CASE)= dat
                   (^ NTYPE CHECK NP-MARKER)=c +
                   (^ NUM)= pl
                 }
          | "haftaya, seneye"
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c +
                      ~(^ SPEC POSS) ~@(NP-MARKER)
                      (^ CASE)= dat
          |
            { "o bu şu bir"
              D: (^ SPEC)= !
            | "geçen evvelki önceki sonraki önümüzdeki gelecek ertesi ilk son"
              Adate:@ADJUNCT
            }
            N[indef]: ^ = !
                      (^ NTYPE NSEM TIME)=c +
                      ~(^ SPEC POSS) ~@(NP-MARKER)
                      { "gelecek ay, önümüzdeki haftaya"
                        (^ NUM)= sg
                        (^ CASE)$c {nom dat}
                      | "son günlerde, o yıllarda"
                        (^ NUM)= pl
                        (^ CASE)=c loc
                      }
          }.
```

For the adverbial use of the nouns *dün* ‘yesterday’, *bugün* ‘today’, *yarın* ‘tomorrow’ at the sentence level we define these nouns as of `Ndate` category in the lexicon like the one given for *dün* ‘yesterday’ in (4.76).

(4.76) `dün  Ndate  *  @(NOUN %stem)  @(TIME +); ETC.`

The part-of-day nouns may also behave as adverbials in the sentence. The second disjunct of the `NPadv` rule covers such cases. Note that *akşamüstü*/*akşamüzeri* ‘nightfall’ and *geceyarısı* ‘midnight’ may only be used in nominative case whereas *sabah* ‘morning’ *öğlen* ‘afternoon’ *akşam* ‘evening’ and *gece* ‘night’ are still temporal adverbials in dative and ablative cases as explained in Section 3.5.

In Turkish, noun-noun phrases are build in one of two ways:

1. Indefinite noun phrase: the head of the phrase gets the possesive suffix where the modifier remain without any suffix.

   kitap *kapağı*
   book cover-P3sg
   ‘book cover’

2. Definite noun phrase: the head noun gets the possesive suffix where the modifier gets the genitive suffix.

   kitabın *kapağı*
   book-GEN cover-P3sg
   ‘cover of the book’

Expressions like *dün akşam* ‘yesterday evening’, *yarın sabah* ‘tomorrow morning’ are exceptions in this sense. They are constructed as if the part-of-day nouns are adjectives. Neither the head nor the modifier gets any suffix.

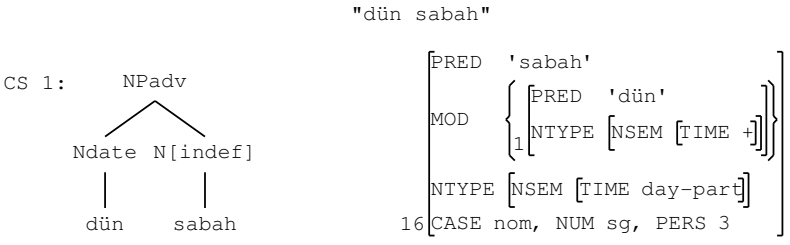


Figure 4.44: C and f-structures for *dün sabah* ‘yesterday morning’

The last disjunct of the `NPadv` rule deals with expressions which are commonly in use to point a time location e.g. *geçen yıl* ‘last year’, *önümüzdeki hafta* ‘next week’. The closed set of the modifiers of these phrases are entered to the lexicon as date adjectives, `Adate`.

(4.77) `geçen  Adate  *  @(ADJ-DATE %stem); ETC.`

C- and f-structures of the phrase *önümüzdeki hafta* ‘next week’ is given in Figure 4.45.

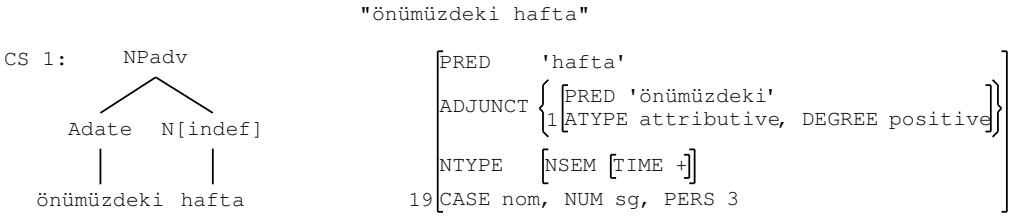


Figure 4.45: C and f-structures for *önümüzdeki hafta* ‘next week’

The phrases where *önce* and *sonra* are preceded by a noun phrase indicating the amount of time are already discussed under the Section 4.2.1. Recall that we prefer to analyze those structures within the postpositional phrases with null objects. Figure 4.46 and Figure 4.47 show the c- and f-structures of the phrases *yirmi gün önce* ‘twenty days ago’ and *az sonra* ‘after a while’ respectively.

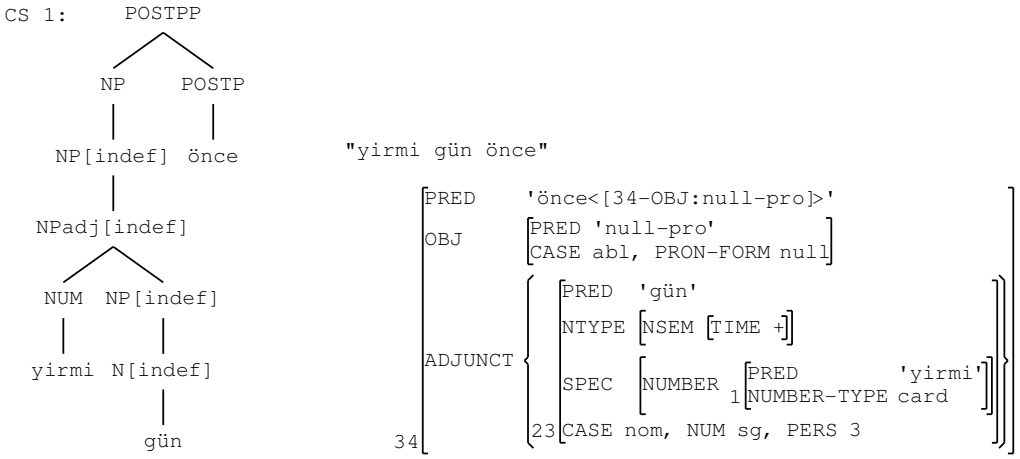


Figure 4.46: C and f-structures for *yirmi gün önce* ‘twenty days ago’

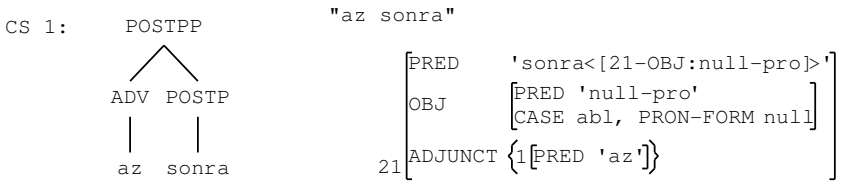


Figure 4.47: C and f-structures for *az sonra* 'after a while'

As for the rest of the adverbials (e.g. *şimdi, demin, sonradan* etc.) introduced in Section 3.5 we already get ADV analyses from the morphology. So they will be recognized thanks to their their POSs.

5 CONCLUSION

In this thesis, we have presented the development of an LFG grammar for a subset of Turkish point-in-time expressions. The implementation of the grammar is based on the Xerox Linguistic Environment (XLE). This work is considered to be a contribution to the ongoing development of a large scale Turkish grammar in the context of the Parallel Grammar Project (ParGram).

We restricted our grammar domain of point-in-time expressions particularly to the phrasal expressions of clock-time, days of the week, calendar dates and seasons. Some general phrases are also adressed.

In order to capture a larger range of expressions in our domain, we considered not only the examples in the grammar literature but also some others from our corpus examination. For this purpose, we ported the Turkish METU corpus data into a form so that they can be interrogated with the Stuttgart Corpus Workbench, IMS CWB (see Section 2.4.2). After all, we achieved a compendium of such expressions which can be used in further computational linguistic works with different formalisms to test the coverage of a grammar.

After having a large collection of sample phrases, we started to the formal modelling of our date-time grammar comparing the solutions needed for Turkish with those available in the formal grammars of German and English, from the ParGram project. While doing this, we paid attention to the possibility to integrate this grammar fragment with the existing LFG grammar of Turkish. Therefore, regular examinations on the current grammar have been needed for a maximal compatibility.

We wrote an idiosyncractic set of phrase structure rules for point-in-time expressions. These rules include noun phrases which would not be parsed by general NP rules (e.g. N-N compound *dün sabah* ‘yesterday morning’, *9 Eylül 1982* ‘September the 9th, 1982’) as well as the phrases which would not get the intended attribute (e.g. 1982 MOD in *1982 yılında ‘in the year 1982’* vs. SPEC NUM) in their functional structure representation.

For date expressions such as *9 Eylül 1982* ‘September the 9th, 1982’, we concluded that there is actually no syntactic relation between the constituents of the phrase. Therefore, we opted for a multi-word analysis treating the whole phrase as a single token. This is provided by a finite-state transducer which is put after the morphological analyzer. This transducer may be optimized in terms of valid-date metrics. Currently, dates which do not exist(e.g. *31 Eylül 2006* ‘September the 31st, 2006’ or *29 Şubat 2006* ‘February the 29th, 2006’ also get parses.

In 4.2.1, we discussed the representation of clock-time expression *saat iki* ‘two o’clock’ through three approaches for the answer of the question: in which sense does the noun *saat* ‘o’clock’ complement the hour? Finally we employed the multiword solution. This solution, however, does not take clock-time phrases into consideration where an adverbial preceeds the numeral as in 5.1.

(5.1) *saat tam ikide* ‘at exactly two o’clock’ vs. *tam saat ikide* ‘exactly at two o’clock’
saat aşağı yukarı ikide ‘at nearly two o’clock’ vs. *aşağı yukarı saat ikide* ‘nearly at two o’clock’

If the representation of the zero derivation is changed into a non-embedded structure later, TIMEP or appositonal approaches may be reconsidered.

Within our grammar, where possible, we marked point-in-time phrases as “temporal”, indeed with more details of semantic distinctions such as clock-time, day, date, season etc. This is necessary both for syntactic and semantic purposes. As we opted for a separate date-time grammar, this was easy to implement by means of templates.

One can argue that the specification for date expressions could be embedded in general NP rules instead. However, our attempts for such a grammar resulted in overparses or lack of temporal marks; constraining NP rules regarding the requirements of date expressions resulted in losing analyses for some other noun phrases (see 4.58, 4.59 and 4.60 on page 93). Hence, we opted for a separate date-time grammar, disprefering the analyses (by OT marks) that NP rules offer for phrases for which we already have analyses.

REFERENCES

Bresnan, Joan, 2001. *Lexical-Functional Syntax*. Oxford: Blackwell.

Bresnan, Joan and Ronald M. Kaplan, 1982. Lexical-functional grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations*. Cambridge, MA: The MIT Press, pp. 173–281.

Butt, Miriam, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi and Christian Rohrer, 2002. The parallel grammar project. In *Proceedings of COLING 2002: Workshop on Grammar Engineering and Evaluation*. Taipei, Taiwan.

Butt, Miriam, Martin Forst, Tracy Holloway King, and Jonas Kuhn, 2003. The feature space in parallel grammar writing. In *Proceedings of ESSLLI 2003: Workshop on Ideas and Strategies in Multilingual Grammar Development*. Vienna, Austria.

Butt, Miriam and Tracy Holloway King, 2005. Restriction for morphological valency alternations: The urdu causative. In *Proceedings of the 10th International LFG Conference*. Bergen, Norway.

Butt, Miriam, Tracy Holloway King, María-Eugenia Niño, and Frédérique Segond, 1999. *A Grammar Writer's Cookbook*. Stanford: CSLI.

Çetinoğlu, Özlem and Kemal Oflazer, 2006a. LFG for Turkish. *Ongoing Lingware Development, latest version of 28/11/2006, internal*. Human Language and Speech Technologies Laboratory, Sabancı University, Türkiye.

Çetinoğlu, Özlem and Kemal Oflazer, 2006b. Morphology-syntax interface for Turkish LFG. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Sydney, Australia.

Christ, Oliver, 1994. A modular and flexible architecture for an integrated corpus query system. In *Proceedings of 3rd Conference on Computational Lexicography and Text Research*. Budapest, Hungary.

Crouch, Dick, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell, and Paula Newman, 2006. *XLE Documentation*. `http://www2.parc.com/isl/groups/nltt/xle/doc/xle_toc.html`. Palo Alto Research Center. Last accessed: 12.10.2006.

Dalrymple, Mary, 2001. *Lexical-Functional Grammar*. San Diego, CA: Academic Press.

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell, and Annie Zaenen (eds.), 1995. *Formal issues in Lexical-Functional Grammar*. Stanford, CA: CSLI.

Evert, Stefan, 2002. *Corpus Encoding Tutorial*. Technical report. IMS, Stuttgart, Germany. `http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/UsersCorner.html`. Last accessed: 12.10.2006.

Evert, Stefan, 2005. *The CQP Query Language Tutorial, CWB version 2.2.b90*. Technical report. IMS, Stuttgart, Germany. `http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/CQPTutorial/html/`. Last accessed: 12.10.2006.

Falk, N. Yehuda, 2001. *Lexical-Functional Grammar: An Introduction to Parallel Constraint-Based Syntax*. Stanford, CA: CSLI Publications.

Frank, Anette, 1999. From parallel grammar development towards machine translation. In *Proceedings of Machine Translation Summit VII: MT in the Great Translation Era*. Kent Ridge Digital Labs, Singapore.

Göksel, Aslı and Celia Kerslake, 2005. *Turkish, A Comprehensive Grammar*. London: Routledge.

Güngördü, Zelal and Kemal Oflazer, 1994. Parsing turkish using the Lexical Functional Grammar formalism. In *Proceedings of the 15th International Conference on Computational Linguistics*. Kyoto, Japan.

Hankamer, Jorge, 1989. Morphological parsing and the lexicon. In William Marslen-Wilson (ed.), *Lexical Representation and Process*. Cambridge, MA: MIT Press, pp. 392–408.

Kaplan, Ronald M., John T. Maxwell, Tracy Holloway King, and Richard Crouch, 2004. Integrating finite-state technology with deep lfg grammars. In *Proceedings of ESSLLI 2004: Workshop on Combining Shallow and Deep Processing for NLP*. Nancy, France.

Kaplan, Ronald M. and Paula S. Newman, 1997. Lexical resource conciliation in the Xerox Linguistic Environment. In *Proceedings of the ACL Workshop on Computational Environments for Grammar Development and Engineering*. Madrid, Spain.

King, Tracy Holloway, 2006. *Parallel Grammar Project*. `http://www2.parc.com/isl/groups/nltt/pargram`. Palo Alto Research Center. Last accessed: 12.12.2006.

McEnery, Tony and Andrew Wilson, 2001. *Corpus Linguistics*. Edinburgh: Edinburgh University Press.

Oflazer, Kemal, 1994. Two-level description of turkish morphology. *Literary and Linguistic Computing*, 9(2):137–148.

Oflazer, Kemal, 2003. Dependency parsing with an extended Finite-State approach. *Computational Linguistics*, 29(4).

Oflazer, Kemal, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür, 2003. Building a turkish treebank. In Anne Abeille (ed.), *Building and Exploiting Syntactically-annotated Corpora*. Cambridge, MA: Kluwer Academic Publishers.

ParGram, 2006. *"common.features.update.lfg"*. Fall Meeting, Oxford.

Say, Bilge, Deniz Zeyrek, Kemal Oflazer, and Umut Özge, 2002. Development of a Corpus and a Treebank for Present-day Written Turkish. In *Proceedings of the 11th International Conference of Turkish Linguistics*. Gazimağusa, TRNC.

Sells, Peter, 1985. *Lectures on Contemporary Syntactic Theories*. Stanford, CA: CSLI.

Sinclair, John M., 1991. *Corpus, Concordance, Collocation*. Oxford: Oxford University Press.

TDK, 2006. *Orthographical Rules*. `http://www.tdk.gov.tr`. Turkish Language Association. Last accessed: 11.11.2006.

Tognini-Bonelli, Elena, 2001. *Corpus linguistics at work*. The Netherlands: Benjamins.

A Common Feature Declarations

```
STANDARD COMMON FEATURES (1.0)

"VERSION DATE: 2006-09-22"

ADJUNCT:-> << [%any].
ADJUNCT-QT:-> << [%any].
ADJUNCT-TYPE:-> $ {conditional degree neg parenthetical relative}.
ADV-TYPE:-> $ {focus loc temp sadv vpadv}.
ANIM:-> $ {+ -}.
AQUANT:-> << [ ADJUNCT PRED QUANT-TYPE DEGREE DEG-DIM ].
ATYPE:-> $ {adverbial attributive predicative}.
CASE:-> $ {abl acc dat erg gen inst loc nom obl part voc}.
CLASSIFIER-FORM:-> $ {%any}.
CLAUSE-TYPE:-> $ {adv cond decl imp int nom pol-int wh-int}.
COMMON:-> $ {count gerund mass measure partitive}.
COMP:-> << [%any].
COMP-FORM:-> $ {%any}.
COORD-FORM:-> $ {%any}.
COORD-LEVEL:-> $ {%any}.
DEG-DIM:-> $ {equative neg pos}.
DEGREE:-> $ {comparative positive superlative}.
DEIXIS:-> $ {distal post-distal proximal}.
DET:-> << [ DEIXIS DET-TYPE PRED ].
DET-TYPE:-> $ {def demon indef int rel}.
FIRST:-> << [%any].
FOCUS:-> << [%any].
FOCUS-INT:-> << [%any].
GEND: {->  $ {fem masc neut} | ->  << [ FEM MASC NEUT ]}.
GEND-SEM:-> $ {female male nonhuman}.
GLOSS:-> << [ ROMAN TRANS ].
HUMAN:-> $ {+ -}.
INCLUSIVE:-> $ {+ -}.
LAST:-> << [%any].
LOCATION-TYPE:-> $ {city country}.
```

```
MOD:-> << [%any].
MOOD:-> $  {imperative indicative subjunctive successive}.
NAME-MOD:-> << [%any].
NAME-TYPE:-> $ {first_name last_name}.
NSEM:-> << [ COMMON NUMBER-TYPE PROPER TIME ].
NSYN:-> $ {common pronoun proper}.
NTYPE:-> << [ NSEM NSYN ].
NUM:-> $ {dual pl sg}.
NUMBER:  -> << [NUMBER-TYPE PRED ADJUNCT CLASSIFIER-FORM MOD].
NUMBER-TYPE:-> $ {card fract ord percent}.
OBJ:-> << [%any].
OBJ-TH:-> << [%any].
OBL:-> << [%any].
OBL-AG:-> << [%any].
OBL-COMPAR:-> << [%any].
OBL-PART:-> << [%any].
PASSIVE:-> $ {+ -}.
PERF:-> $ {+ - +_ -_}.
PERS:-> $ {1 2 3}.
PFORM:-> $ {%any}.
POSS:-> << [%any].
POSTCOORD-FORM:-> $ {%any}.
PRECOORD-FORM:-> $ {%any}.
PROG:-> $ {+ - +_ -_}.
PRON-FORM:-> $ {%any}.
PRON-INT:-> << [%any].
PRON-REL:-> << [%any].
PRON-TYPE:-> $ {demon expl_ free inh-refl_ int locative pers
                quant poss recip refl rel}.
PROPER:-> << [ PROPER-TYPE LOCATION-TYPE NAME-TYPE ].
PROPER-TYPE:-> $ {addr_form location name organization title}.
PRT-FORM:-> $ {%any}.
PSEM:-> $ {ag ben comit compar dir inst loc manner num part
          poss purp temp}.
PTYPE:-> $ {nosem sem}.
QUANT:-> << [ADJUNCT QUANT-TYPE POL PRED DEGREE DEG-DIM ].
QUANT-TYPE:-> $ {comparative equative existential gen negative
                 superlative universal}.
REST:-> << [%any].
ROMAN:-> $ {%any}.
SPEC:-> << [ADJUNCT AQUANT DET NUMBER POSS QUANT].
STMT-TYPE:-> $ {decl excl header imp int}.
SUBJ:-> << [%any].
TENSE:-> $ {fut past pres}.
```

```
TIME:-> $ {date day hour minute month season second week year}.
TNS-ASP:-> << [MOOD PERF PROG TENSE].
TOKEN:-> $ {%any}.
TOPIC:-> << [%any].
TOPIC-REL:-> << [%any].
TRANS:-> $ {%any}.
VTYPE:-> $ {aux copular main modal noncopular predicative raising}.
XCOMP:-> << [%any].

----
```

B Multiword FSTs for Clock-time

```
define OClock [ {bir} | {iki} | {üç} | {dört} | {beş} | {altı}
              | {yedi} | {sekiz} | {dokuz} | {on}
              | ([%0|1]) [%0|1|2|3|4|5|6|7|8|9] | 2 [%0|1|2|3|4] ];

regex [
?*(
s a a t " ":"+Noun" 0:"+A3sg" 0:"+Pnon" 0:"+Nom"
OClock  "+Noun":"+Num"  "+ClockTime":"+Card"
        ("+A3sg":0  "+Pnon":0  "+Nom":0)
        (0:"^DB"  0:"+Noun"  0:"+Zero"  "+A3sg"  "+Pnon"
        [ "+Acc" | "+Dat" | "+Loc" | "+Abl" | "+Gen" ] )
?*)*
];

save time1.fst



define OClock [ ([%0|1|2]) [%0|1|2|3|4|5|6|7|8|9] %. [%0|1|2|3|4|5]
              [ %0|1|2|3|4|5|6|7|8|9] ];

regex [
?*(
s a a t " ":"+Noun" 0:"+A3sg" 0:"+Pnon" 0:"+Nom"
OClock  "+Noun":"+Num"  "+ClockTime":"+Time"
        ("+A3sg":0  "+Pnon":0  "+Nom":0)
        (0:"^DB"  0:"+Noun"  0:"+Zero"  "+A3sg"  "+Pnon"
        [ "+Acc" | "+Dat" | "+Loc" | "+Abl" | "+Gen" ] )
?*)*
];

save time2.fst
```

C Multiword FST for Days of the Week

```
define Week  [ {Pazartesi} | {Salı} | {Çarşamba} | {Perşembe}
             | {Cuma} | {Cumartesi} | {Pazar} | {pazartesi}
             | {salı} | {çarşamba} | {perşembe} | {cuma}
             | {cumartesi} | {pazar} ];

regex [
?*
(Week " ":"+Noun" 0:"+A3sg" 0:"+Pnon" 0:"+Nom"
g ü n  "+Noun" "+WeekDay":0 "+A3sg" "+P3sg"
      [ "+Nom"|"+Acc"|"+Dat"|"+Loc"|"+Abl"|"+Gen" ]
?*)*
];

save weekday.fst
```

D Multiword FST for Dates with Day and Month

```
define 1to9   [ 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ];
define 0to9   [ %0 | 1to9 ];
define Month  [ {ocak} | {Şubat} | {mart} | {nisan} | {mayıs}
              | {haziran} | {temmuz} | {ağustos} | {eylül}
              | {ekim} | {kasım} | {aralık} | {Ocak}
              | {Şubat} | {Mart} | {Nisan} | {Mayıs}
              | {Haziran} | {Temmuz} | {Ağustos} | {Eylül}
              | {Ekim} | {Kasım} | {Aralık}];

regex [
?*
([ (%0) 1to9 | [1|2] 0to9 | 3 [%0|1] ] " ":"+Num" 0:"+Card"
[ [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Nom"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Acc"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Dat"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Loc"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Abl"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+Pnon" "+Gen"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Nom"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Acc"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Dat"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Loc"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Abl"]
| [Month "+Noun" ("+Prop") "+DateTime":0 "+A3sg" "+P3sg" "+Gen"] ]
?*)*
];

save daymonth.fst
```

E Multiword FST for Dates with Day, Month and Year

```
define 1to9   [ 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ];
define 0to9   [ %0 | 1to9 ];
define Year   [ 1to9 ( 0to9 ( 0to9 ( 0to9 ) ) ) ];
define Month  [ {ocak} | {şubat} | {mart} | {nisan} | {mayıs}
              | {haziran} | {temmuz} | {ağustos} | {eylül} | {ekim}
              | {kasım} | {aralık} ];

regex [
?*(
([ (%0) 1to9 | [1|2] 0to9 | 3 [%0|1] ] " ":"+Num" 0:"+Card" )
Month " ":"+Noun" 0:"+A3sg" 0:"+Pnon" 0:"+Nom"
[ [ Year 0:"+Num" 0:"+Card"
         "+Noun":0 "+DateTime":0 "+A3sg":0 "+Pnon":0 "+Nom":0 ]
| [ Year 0:"+Num" 0:"+Card"
         0:"^DB" "+Noun" "+DateTime":"+Zero" "+A3sg" "+Pnon" "+Acc" ]
| [ Year 0:"+Num" 0:"+Card"
         0:"^DB" "+Noun" "+DateTime":"+Zero" "+A3sg" "+Pnon" "+Dat" ]
| [ Year 0:"+Num" 0:"+Card"
         0:"^DB" "+Noun" "+DateTime":"+Zero" "+A3sg" "+Pnon" "+Loc" ]
| [ Year 0:"+Num" 0:"+Card"
         0:"^DB" "+Noun" "+DateTime":"+Zero" "+A3sg" "+Pnon" "+Abl" ]
| [ Year 0:"+Num" 0:"+Card"
         0:"^DB" "+Noun" "+DateTime":"+Zero" "+A3sg" "+Pnon" "+Gen" ] ]
?*)*
];

save daymonthyear.fst
```

F Multiword FST for Abbreviated Dates

```
define 1to9  [ 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 ];
define 0to9  [ %0 | 1to9 ];
define Year  [ 0to9 ( 0to9 ( 0to9 ( 0to9 ) ) ) ];

regex [
?*(
( [(%0) 1to9 | [1|2] 0to9 | 3 [%0|1] ] 0:"+Num" 0:"+Card"
%/ 0:"+Punct" 0:"+Slash" )
[ (%0) 1to9 | 1 0 | 1 1 | 1 2 ] 0:"+Num" 0:"+Card"
%/ 0:"+Punct" 0:"+Slash"
Year "+Noun":"+Num" "+DateTime":"+Card"
     ("+A3sg":0  "+Pnon":0  "+Nom":0)
     (0:"^DB"  0:"+Noun"  0:"+Zero"  "+A3sg"  "+Pnon"
     [ "+Acc" | "+Dat" | "+Loc" | "+Abl" | "+Gen" ] )
?*)*
];

save abbr_date_slash.fst
```

G Sample Sentences from METU Corpus

1. **<Cumartesi>** canlı müzik ve büyük bir orkestra onlara eşlik ediyor, vals yapıyorlar.

2. Annem **<çarşambaya>** trenle gelir artık.

3. Ibrahim Demirhan isimli tütün satıcısı ise **<9 Haziran'da>** öldürülmüştü.

4. Kendisiyle telefonla yapılan ve birkaç gün önce, **<31 Temmuz 1994'te>**, Cumhuriyet'te yayımlanan bir röportaj bu inancımı güçlendirdi.

5. Keza **<16-20 Temmuz 1994'de>** Jüpiter gezegeninin Kuyruklu Yıldız Yağmuru'na uğrayısını televizyonlardan, naklen tıpkı bir Uzay fiovu gibi izlemiştik.

6. **<Kasım 1945'te>** Refik Koraltan partiden çıkarıldı ve **<Aralığın başında>** Celâl Bayar bir parti kuracağını açıkladı.

7. Bu gelişmeyle birlikte **<Aralık ortasında>** 16.000'li seviyeleri görebiliriz.

8. Marsan Marmara Holding A.Ş. **<1981 yılında>** Özer Uçuran Çiller'in annesi Ismet Uçuran ve paravan isimler Atilla Suman, H.Demet Suman, M.Hikmet Azmak ve Ayla Azmak tarafından 10 milyon TL. sermayeyle kurulmuştur. Bilahare **<31.12.1981'de>**, 10 milyon TL. sermayenin 4 milyon 990 bin lirası Özer Uçuran Çiller, 4 milyon 980 bin lirası da eşi Tansu Çiller tarafından devir alınmıştır.

9. Yağışlar, **<18 Ocak Cumartesi günü>** yurdun doğusunda etkisini gösterecek. **<19 Ocak Pazar günü>** ise yurdun doğusundaki yağışlı hava etkisini kaybedecek.

10. Genelkurmay Başkanlığı Genel Sekreterliği'nden konuya ilişkin yapılan açıklama şöyle : "Diyarbakır ili Hani ilçesi kırsalında icrasına başlanılan planlı bir operasyon kapsamında, **<16 Ocak 2003 Perşembe günü sabahı> <06.50'de>** bir grup terörist ile temas sağlanmıştır.

11. Ressam ve şair Bedri Rahmi Eyüboğlu şiir kitapçığını **<Karadut Hanım 'ın ölümünden iki yıl sonra>**, **<1948'de>** kendi olanaklarıyla, can dostu, çocukluk arkadaşı Rüknettin Resuloğlu'nun da aracılığıyla bastırmış, tanesini de yüz kuruştan satmıştı.

12. Türkiye’de **<80’lerde>** gelir dağılımının iyice bozulduğu herkesçe bilinen bir olgu.

13. Baytop **<20 Haziran 2000’de>** 80 yaşını dolduracak; **<1987 yılında>** yaş haddi nedeniyle devlet onu emekli etmiş, ama o kendini çalışmalarından emekli etmiyor.

14. Burada **<MÖ 2000 yıllarında>** Asur’dan gelen tüccarlar bir koloni kurmuşlar ve hemen hemen üç nesil boyunca Asur, Mezopotamya ile Anadolu arasında büyük bir ticaret ağı oluşturmuşlar.

15. **<Haziran ayında>** sokaklar manolya, yasemin, gül değil, egzoz gazı, lağım ve ter kokuyor.

16. Aralarında kaçıncı ayda iletişim kurulsa beğenirsiniz. O da tek tük cümlelerle: **<On dördüncü ayda>**.

17. Bu yandan çarklılardan biri, **<1848 Ekiminde>** New-York limanını iki yüz on yolcuyla terk ettikten sonra ta güneyden, Macellan Boğazından dolaşarak San Francisco’ya doğru yoluna devam etmiş, Panama açıklarında Güney Amerika ’nın çeşitli ülkelerinden gelen yeni yolcuların akınına uğrayarak ancak **<1849 yılının şubat ayında>** bin beş yüz kişiyle San Francisco Körfezine demir atabilmişti.

18. Kit Carson **<1864 Ocak’ının ilk haftasında>** Chelly Boğazını bastı.

19. Geriye doğru baktığımızda, **<2000 yılı ocak ayında>** kötü bir zamanlama ile Nasdaq’a yatırım yapanlar ortalama paralarının dörtte üçünü kaybetmişken, Buffet’ın şirketinin hisse senedini alanlar yaklaşık yüzde 50 kazanmış durumda!

20. **<1935 yılı şubatında>**, Adolf Hitler’in Almanya’da yasak ettiği kitaplardan meydana getirilen bir Amerikan kitaplığı Brooklyn’deki Jewish Center’da Profesör Einstein tarafından açıldı.

21. **<2003 yılının ekiminde>** kimbilir ne tür olaylar çekilecek manşetlere.

22. Çalışanlar, belgelerini, **<2003 Ocak ayında>** teslim edecek ve **<şubattan itibaren>** özel gider indirimi çerçevesinde iade alacaklar.

23. Biz kendi kendimizi **<2003’ün Ekim ayına kadar>** bunları yapacağımıza inandırırsak, yaparız.

24. Bir insanın yatak odasını bile görüntüleyebilecek olanaklara sahip olan NSA’nın, **<98’in Mayıs ayında>** Hindistan’ın gerçekleştirdiği nükleer denemeden haberi bile olmaması; Kuzey Kore’nin Japonya semalarında başlattığı balistik füze denemelerini ve Afrika ’daki Amerikan elçiliklerine yapılan saldırıları önceden tespit edememesi, bu başarısızlıkların örnekleridirler.

25. **<1923 yılı Eylül ayında>** Neue Press muhabirine bir demeç verirken akıl ve yüreğinde taşıdığı Cumhuriyet hedefini tüm dünyaya haykırdı: Egemenlik kayıtsız, koşulsuz ulusundur.

26. **<1996 yılının Aralık ayında, akşam üzeri>**, televizyonda haberleri seyrediyordum.

27. **<1991 yılının ilk ayında>** dışarıdaki hayata adımını atmış: **<Şimdi>** 40 yaşında, Yeni ülke dergisinin Izmir temsilciliğini yapıyor; onu gerçek ve cesur kılan tek bir aşka inanıyor: Toplumsal mücadele aşkı.

28. Milliyet'in diplomatik kaynaklardan edindiği bilgiye göre, henüz resmiyet kazanmayan ziyaret, **<ocağın son haftasında>** gerçekleşecek.

29. Risk almak istemeyen yatırımcılar **<aralık ayının ikinci yarısından itibaren>** pozisyonlarını küçültebilirler.

30. Euro bazında **2002 yılının ilk yarısında>** yatırım fonları toplam tutarı 4.4 oranında geriledi.

31. **<Önce>** 2 yüzyıl kavisli bir soru: **<1902, 12 Eylül'ünde>** neler yazılıp konuşuluyordu; **<2102, 12 Eylül'ünde>** neler konuşulup yazılacak?

32. **<2002 yılının ilk yarısında>** yaşanan artışta Euro'nun ABD dolarına karşı değer kazanması etkili oldu.

33. **2002 Aralık başında>** Irak'la ilgili olarak gerginleşen ortamda euro - dolar karşısında çok sert bir yükselişe başladı.

34. **<1985 yılının şubat sonunda>** bir akşamüstü, Nevizâde Sokak'ta bir meyhanede demlenen Tuğrul'la Altan, bildik ve eskimiş tekrarlarını; sözün gelişi anılacak kişilerin dedikodularını bitirmişler ve eski arkadaşları Cahit hakkında konuşmaya başlamışlardı.

35. Onu en son **<1972 yılının Aralık ayı sonlarında>** Stockholm'de görmüştüm.

36. Bunun için TBMM'nin hemen açılması gerektiğinin altını çizen Çiller, "önümüzde 1-2 hafta var. **<Temmuz ayının sonuna kadar>** AB ile ilgili yasaları çıkaramazsak **<eylülde>** AB tarafından yazımına başlanacak rapora Türkiye bölümü girmeyecek" dedi.

37. **<Her yılın 24 Ocak'ında>** onu anıyor, anısına şapka çıkarıyoruz.

38. Yine onun ısrarı ile **<1957 yazında>** Almanya'nın yeni kurmuş olduğu Idarî Bilimler Merkezine üç ay süre ile Speyer'e konuk asistanlık yapmak üzere Almanya'ya gittim.

39. **<2001 yılı baharında>** Birecik’te kırk iki kelaynak var.

40. Raporun bu bölümünün özeti, **<1994 yılının sonbaharında>** Cumhuriyet Gazetesi’nde yayımlanınca Banker Kastelli telefonla aradı.

41. Işte taç giyme anı: **<IÖ 109 yılının 14 Temmuz akşamı saat 19:37>**.

42. Ömer Deniz, **<1937 yılının 3 Aralık günü>**, Nişantaşı’ndaki evinin kapısını çalar Nâzım Hikmet’in.

43. **<18 Ağustos 1878 günü>** Avusturya-Macaristan ordularına karşı direnen kent halkı bu caminin minaresinden de ateş açmış işgal ordusuna.

44. **Yaklaşık 6 ay önce>**, **7 Aralık 1997 tarihinde>**, bu köşede kaleme aldığımız yazının başlığı **98’de>** Yılın Olayı idi.

45. **<2003’ün hemen başında>** ise trendlere tekrar bakılarak bono piyasasında yeni stratejilerin belirlenmesi bekleniyor.

46. Iletişim aletleri bölümünde sergilenen telgraflar, fonografların replikaları, **<19’uncu yüzyıl sonunda>** kullanılmış değişik gramofonlar ilgi çekicidir.

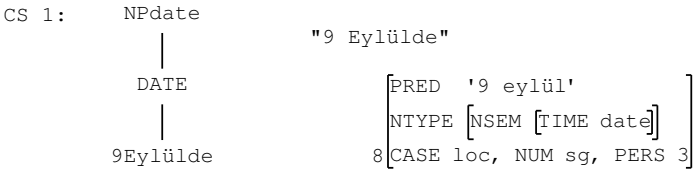
47. Göçebe aşiretler **<IÖ 9. yüzyılın ortalarında>** birleşme ile sonuçlanan bir süreç içine girdiler.

48. ***<16 Mayıs> <sabaha karşı>** tam Ay tutulması için uygun koşullar oluşurken, Ay, **<TSI 05:03’te>** Dünya’nın yarı gölgenin içine girmesiyle başlayacak.

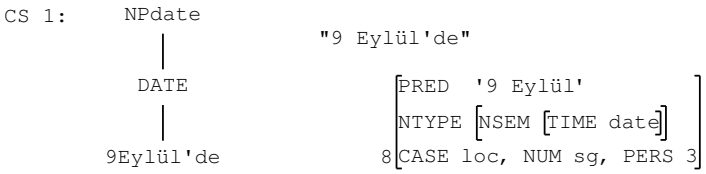
49. Migros, **<21 Kasım-11 Aralık arasında>** 644 üründe özel indirimler yapıyor.

H Test-suite

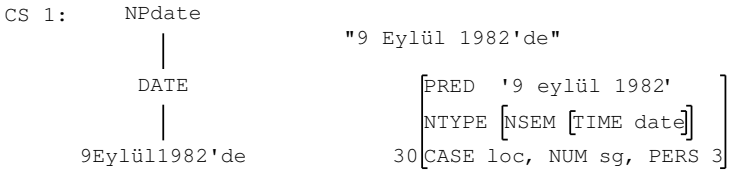
1. *9 Eylülde*
   9 September-LOC
   ‘on September the 9th’



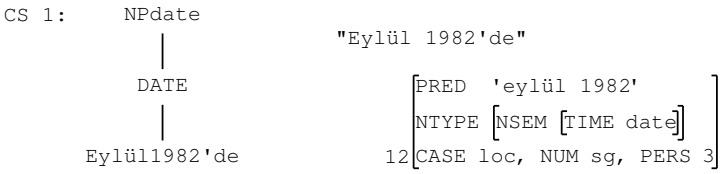
2. *9 Eylül’de*
   9 September-LOC
   ‘on September the 9th’



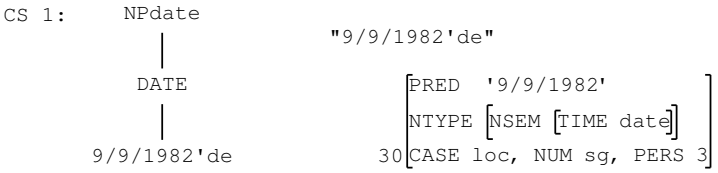
3. *9 Eylül 1982’de*
   9 September 1982-LOC
   ‘on September the 9th, 1982’



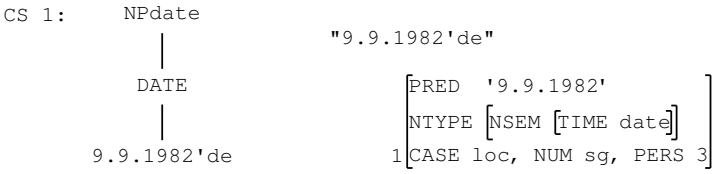
4. *Eylül 1982'de*
   September 1982-LOC
   'in September, 1982'



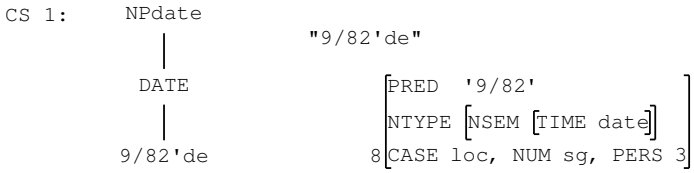
5. *9/9/1982'de*
   9/9/1982-LOC
   'on 9/9/1982'



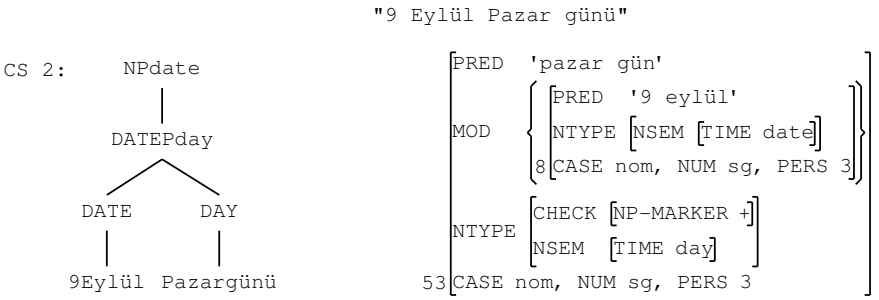
6. *9.9.1982'de*
   9.9.1982-LOC
   'on 9.9.1982'



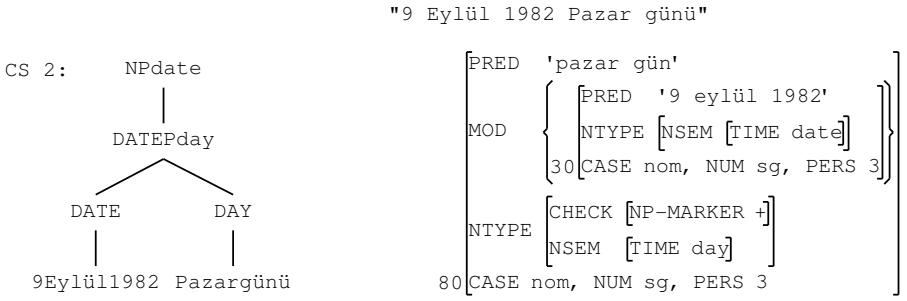
7. *9/82'de*
   9/82-LOC
   'in 9/82'



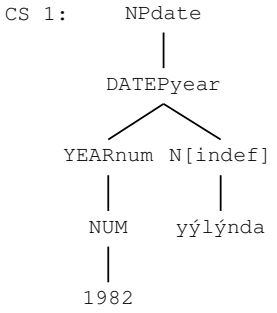
8. *9 Eylül Pazar günü*
   9 September Sunday day-P3sg
   'on Sunday, September the 9th'



9. *9 Eylül 1982 Pazar günü*
   9 September 1982 Sunday day-P3sg
   'on Sunday, September the 9th, 1982'



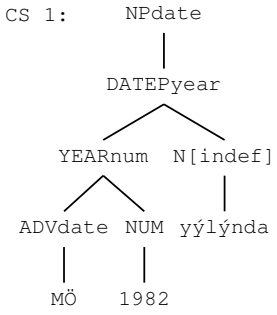
10. *1982 yılında*
1982 year-P3sg-LOC
‘in the year of 1982’



"1982 yýlýnda"

$$\begin{bmatrix} \text{PRED} & \text{'yýl'} \\ \text{MOD} & \left\{ 1 \begin{bmatrix} \text{PRED} & \text{'1982'} \\ \text{NUMBER-TYPE} & \text{card, TIME year} \end{bmatrix} \right\} \\ \text{NTYPE} & \begin{bmatrix} \text{CHECK} & [\text{NP-MARKER} +] \\ \text{NSEM} & [\text{TIME} +] \end{bmatrix} \\ 8 & [\text{CASE loc, NUM sg, PERS 3}] \end{bmatrix}$$

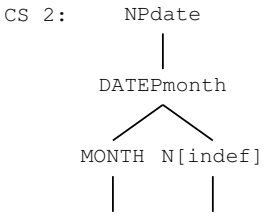
11. *MÖ 1982 yılında*
B.C. 1982 year-P3sg-LOC
‘in the year of 1982 B.C.’



"MÖ 1982 yýlýnda"

$$\begin{bmatrix} \text{PRED} & \text{'yýl'} \\ \text{MOD} & \left\{ \begin{bmatrix} \text{PRED} & \text{'1982'} \\ \text{ADJUNCT} & \{1[\text{PRED 'MÖ'}]\} \end{bmatrix} \right. \\ & \left. 14[\text{NUMBER-TYPE card, TIME year}] \right\} \\ \text{NTYPE} & \begin{bmatrix} \text{CHECK} & [\text{NP-MARKER} +] \\ \text{NSEM} & [\text{TIME} +] \end{bmatrix} \\ 21 & [\text{CASE loc, NUM sg, PERS 3}] \end{bmatrix}$$

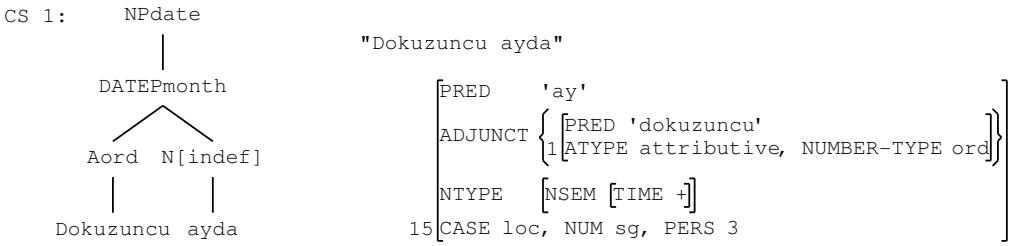
12. *Eylül ayında*
September month-P3sg-LOC
‘in the month of September’



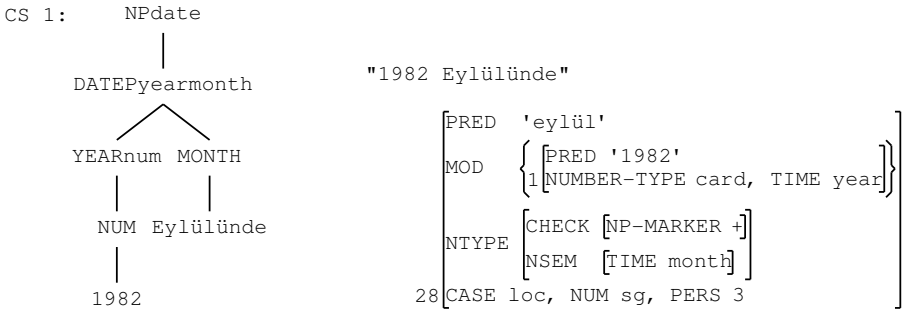
"Eylül ayýnda"

$$\begin{bmatrix} \text{PRED} & \text{'ay'} \\ \text{MOD} & \left\{ 1 \begin{bmatrix} \text{PRED} & \text{'eylül'} \\ \text{NTYPE} & [\text{NSEM} [\text{TIME month}]] \\ \text{CASE nom, NUM sg, PERS 3} & \end{bmatrix} \right\} \\ \text{NTYPE} & \begin{bmatrix} \text{CHECK} & [\text{NP-MARKER} +] \\ \text{NSEM} & [\text{TIME} +] \end{bmatrix} \\ 26 & [\text{CASE loc, NUM sg, PERS 3}] \end{bmatrix}$$

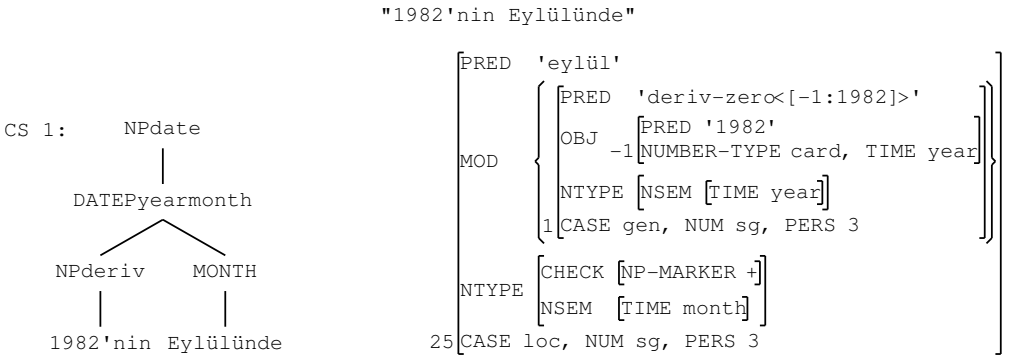
13. *Dokuzuncu ayda*
Ninth month-LOC
'in the ninth month'



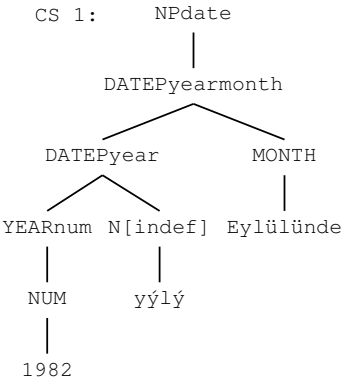
14. *1982 Eylülünde*
1982 September-P3sg-LOC
'in September of 1982'



15. *1982'nin Eylülünde*
1982-GEN September-P3sg-LOC
'in September of 1982'



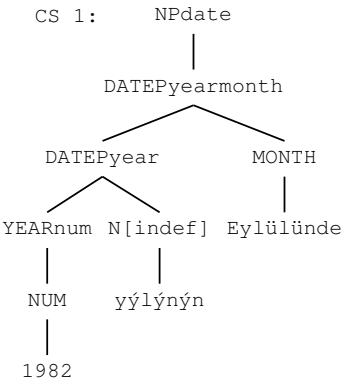
16. *1982 yılı Eylülünde*
1982 year-P3sg September-P3sg-LOC
'in September of the year 1982'



"1982 yýlý Eylülünde"

```
[PRED  'eylül'
 MOD   {1[PRED  'yýl'
          MOD   {1[PRED '1982'
                   NUMBER-TYPE card, TIME year]}
          NTYPE [CHECK [NP-MARKER +]
                 NSEM  [TIME +]]
          8[CASE nom, NUM sg, PERS 3]}
 NTYPE [CHECK [NP-MARKER +]
        NSEM  [TIME month]]
23[CASE loc, NUM sg, PERS 3]]
```

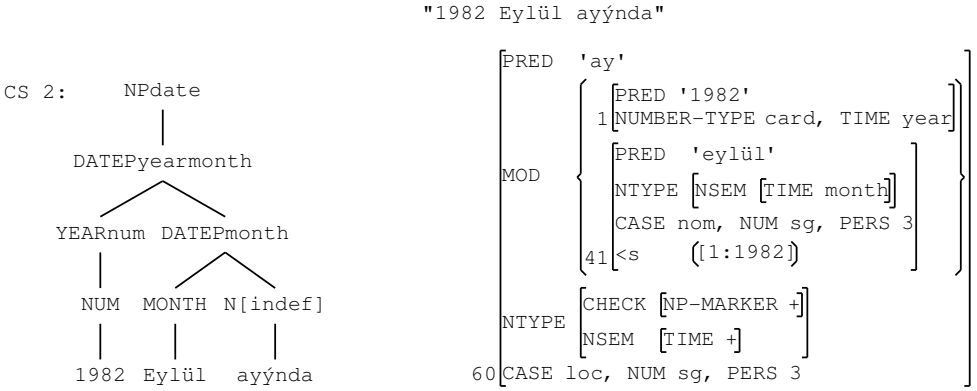
17. *1982 yılının Eylülünde*
1982 year-P3sg-GEN September-P3sg-LOC
'in September of the year 1982'



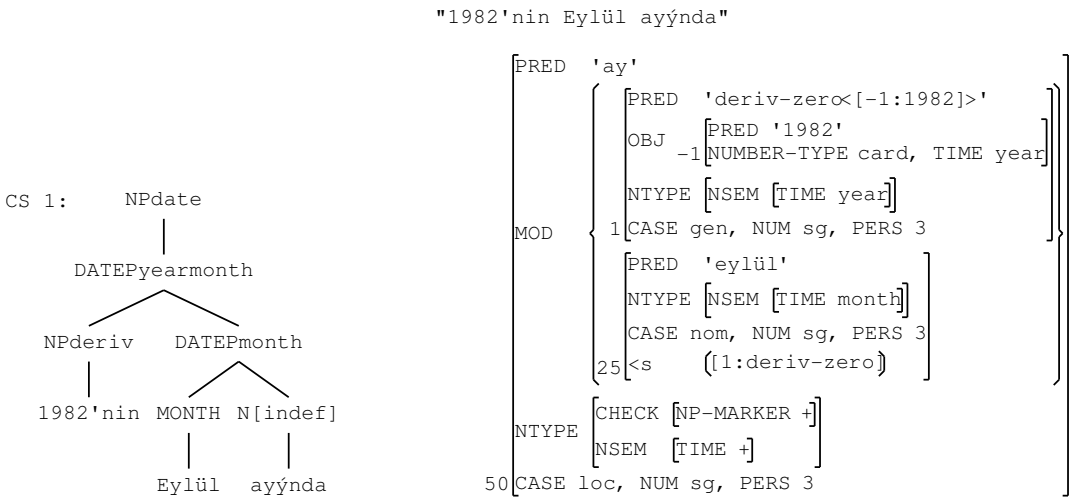
"1982 yýlýnýn Eylülünde"

```
[PRED  'eylül'
 MOD   {1[PRED  'yýl'
          MOD   {1[PRED '1982'
                   NUMBER-TYPE card, TIME year]}
          NTYPE [CHECK [NP-MARKER +]
                 NSEM  [TIME +]]
          8[CASE gen, NUM sg, PERS 3]}
 NTYPE [CHECK [NP-MARKER +]
        NSEM  [TIME month]]
41[CASE loc, NUM sg, PERS 3]]
```

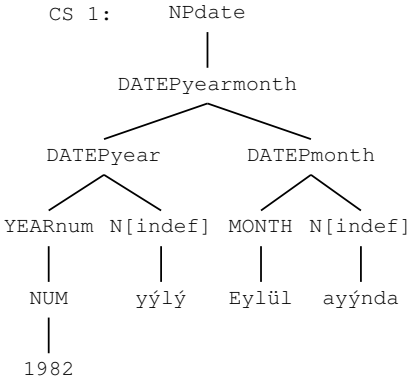
18. *1982 Eylül ayında*
1982 September month-P3sg-LOC
'in the month of September of 1982'

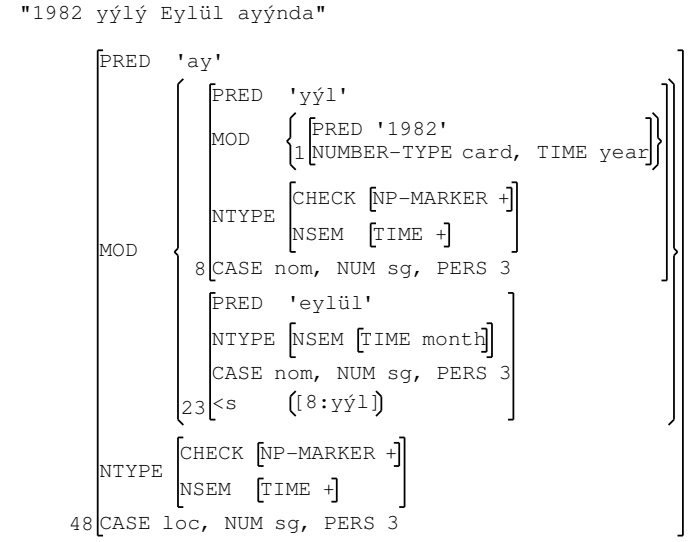


19. *1982'nin Eylül ayında*
1982-GEN September month-P3sg-LOC
'in the month of September of 1982'

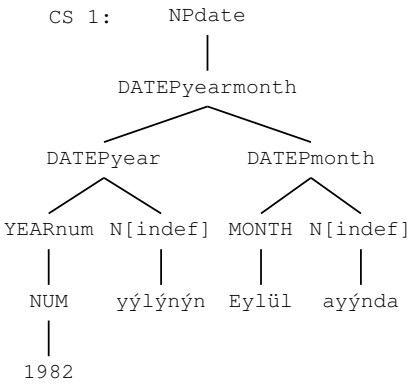


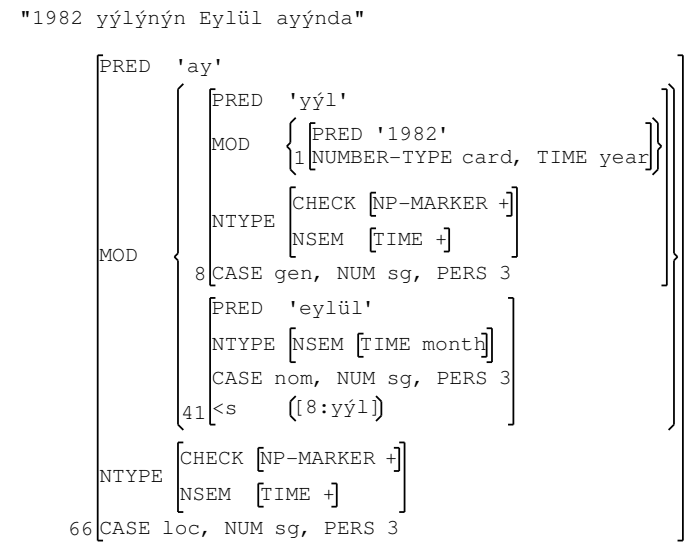
20. *1982 yılı Eylül ayında*
1982 year-P3sg September month-P3sg-LOC
'in the month of September of the year 1982'



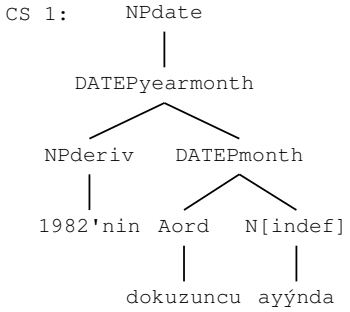


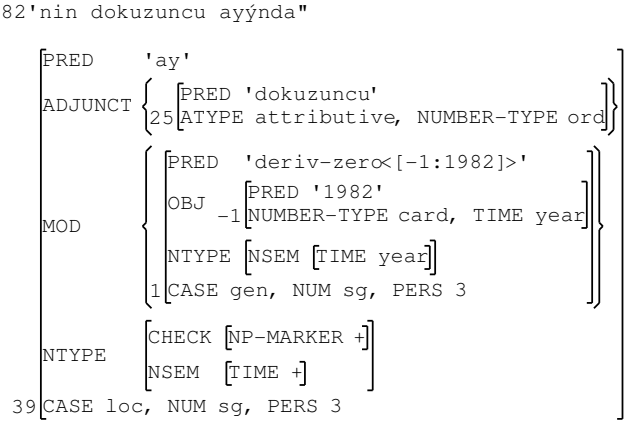
21. *1982 yılının Eylül ayında*
1982 year-P3sg-GEN September month-P3sg-LOC
'in the month of September of the year 1982'



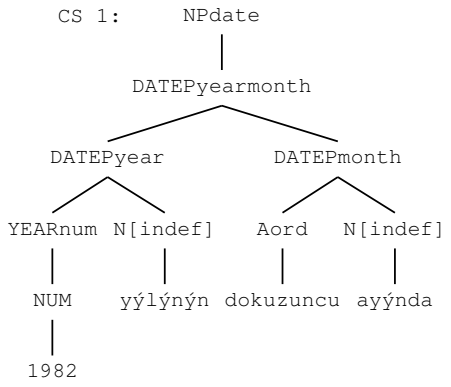


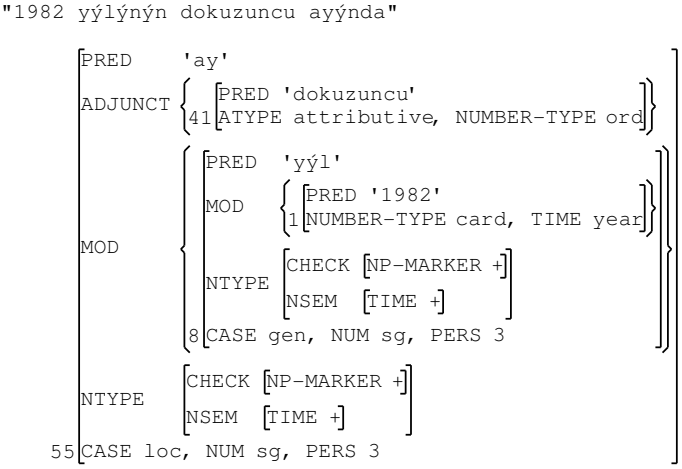
22. *1982'nin dokuzuncu ayında*
1982-GEN ninth month-P3sg-LOC
'in the ninth month of 1982'



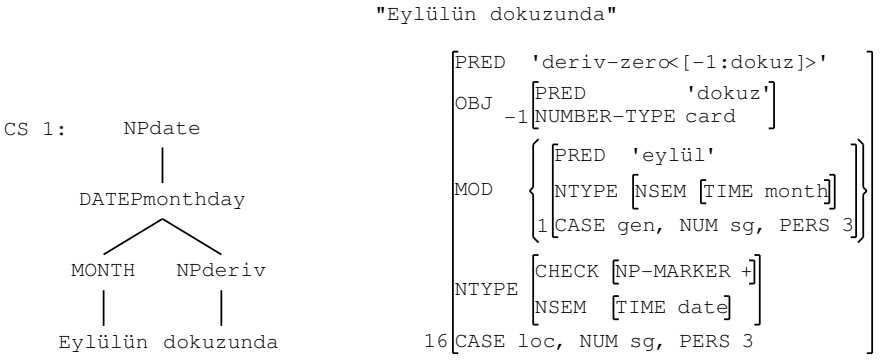


23. *1982 yılının dokuzuncu ayında*
1982 year-P3sg-GEN ninth month-P3sg-LOC
'in the ninth month of the year 1982'

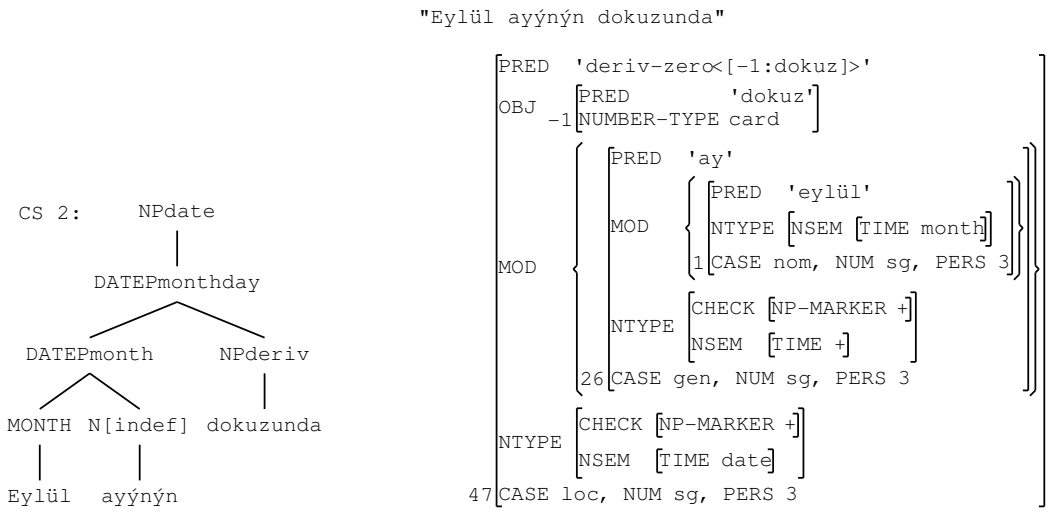




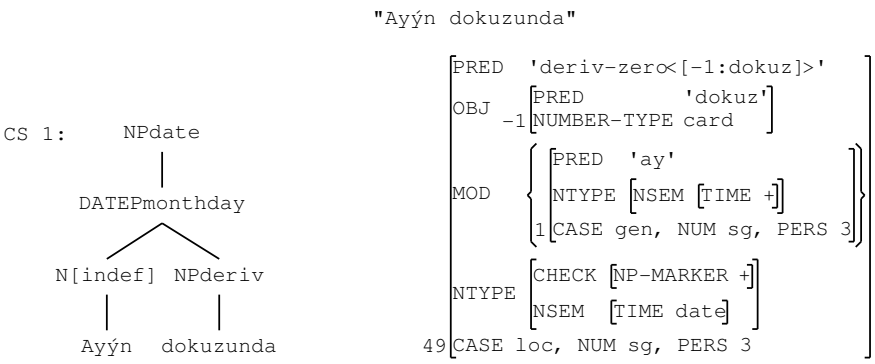
24. *Eylülün* *dokuzunda*
September-GEN nine-P3sg-LOC
'on the ninth of September'



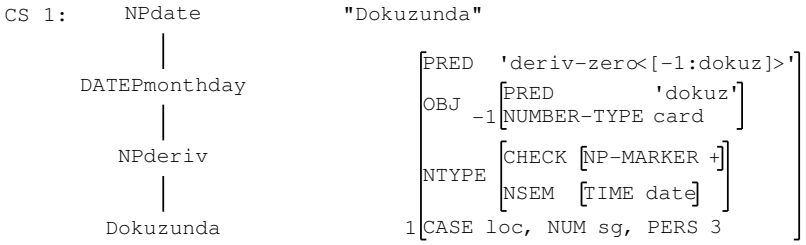
25. *Eylül* *ayının* *dokuzunda*
September month-P3sg-GEN nine-P3sg-LOC
'on the ninth of the month of September'



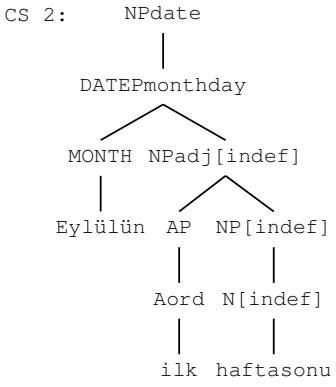
26. *Ayın* *dokuzunda*
month-GEN nine-P3sg-LOC
‘on the ninth of the month’

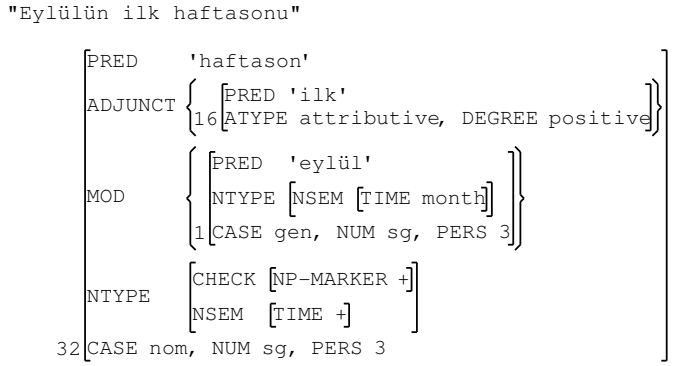


27. *Dokuzunda*
nine-P3sg-LOC
‘on the ninth’

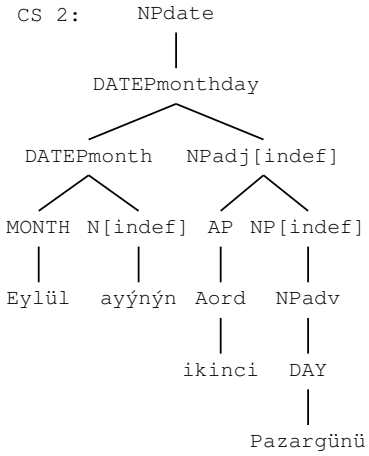


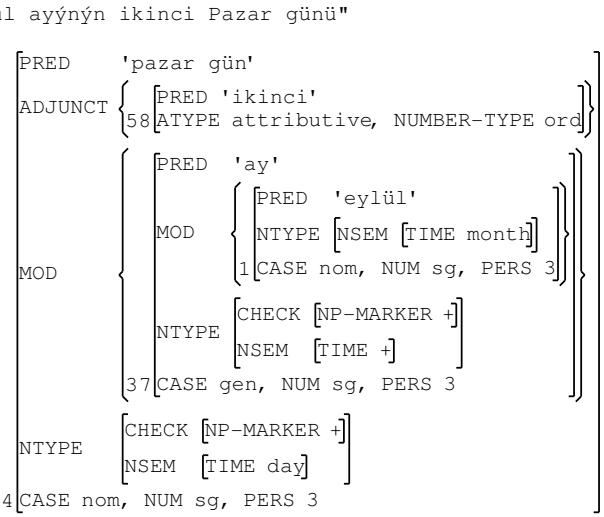
28. *Eylülün ilk haftasonu*
September-GEN first weekend-P3sg
'in the first weekend of September '



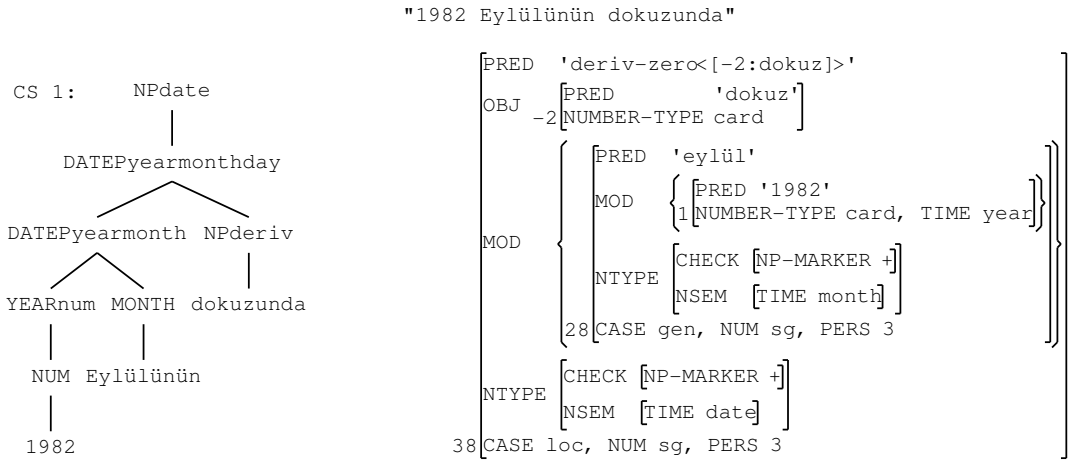


29. *Eylül ayının ikinci Pazar günü*
September month-P3sg-GEN second Sunday day-P3sg
'on the second Sunday of the month of September'

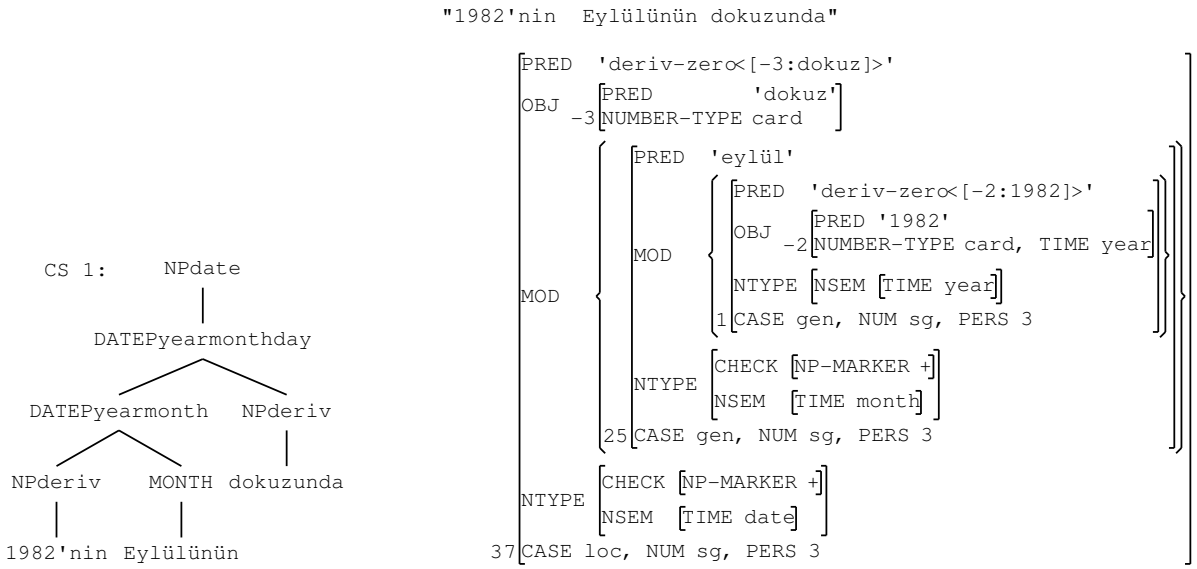




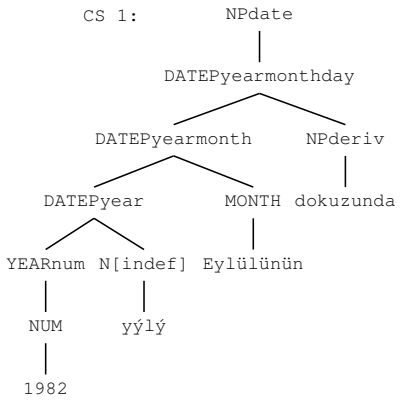
30. *1982 Eylülünün dokuzunda*
1982 September-P3sg-GEN nine-P3sg-LOC
‘on the ninth of September of 1982’

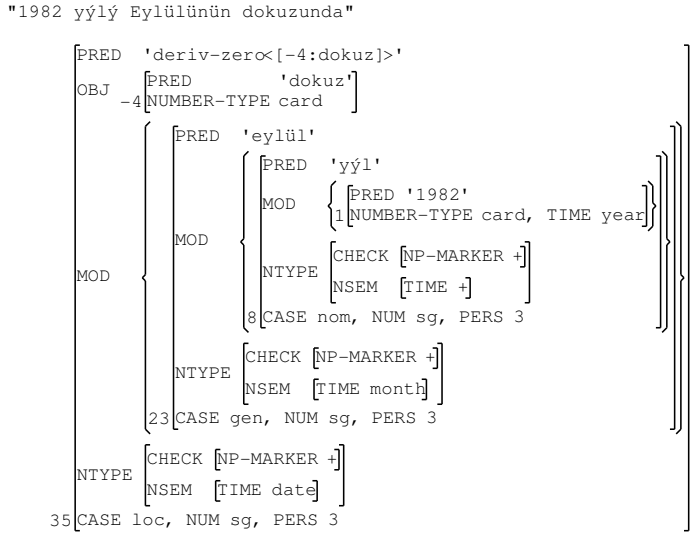


31. *1982nin Eylülünün dokuzunda*
1982-GEN September-P3sg-GEN nine-P3sg-LOC
‘on the ninth of September of 1982 ’

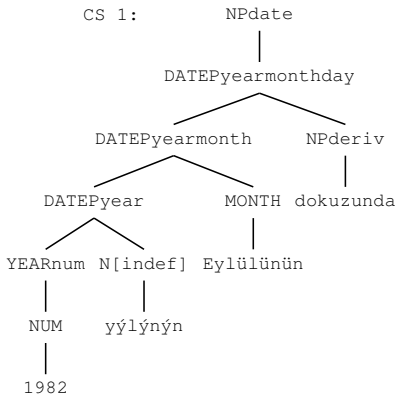


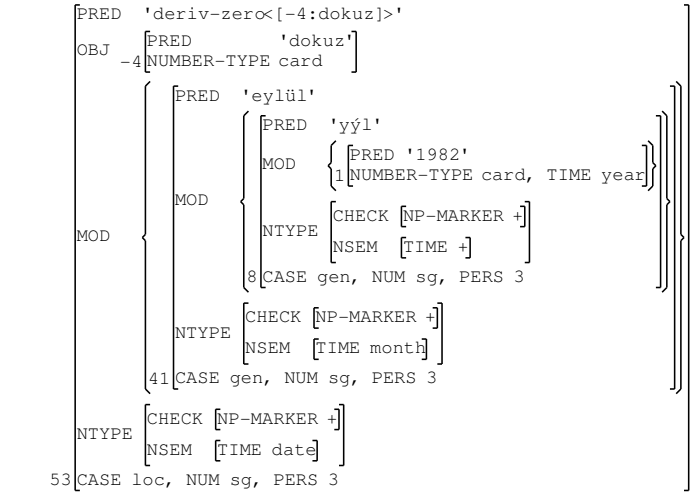
32. *1982 yılı Eylülünün dokuzunda*
1982 year-P3sg September-P3sg-GEN nine-P3sg-LOC
‘on the ninth of September of the year 1982 ’



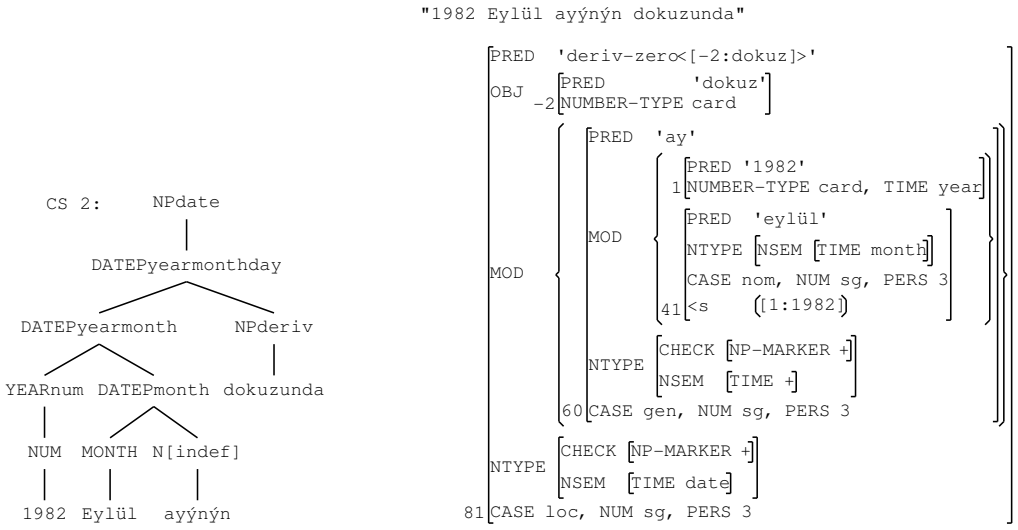


33. *1982 yılının Eylülünün dokuzunda*
1982 year-P3sg-GEN September-P3sg-GEN nine-P3sg-LOC
‘on the ninth of September of the year 1982 ’

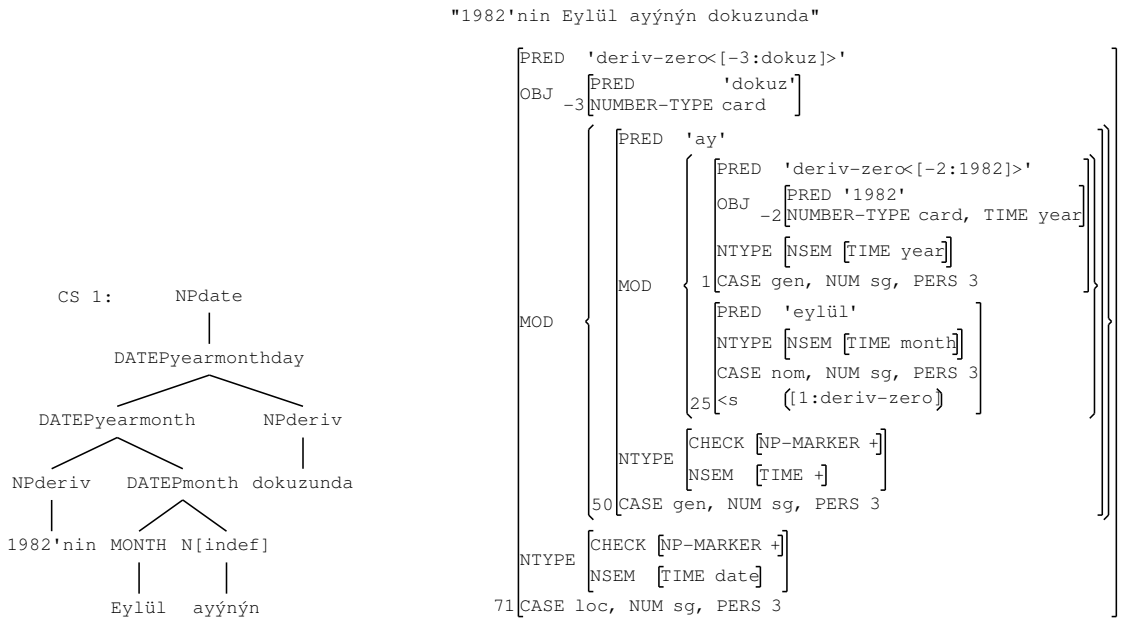




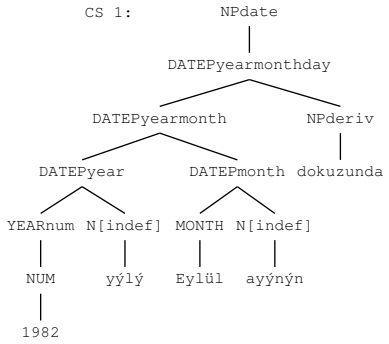
34. *1982 Eylül ayının dokuzunda*
1982 September month-P3sg-GEN nine-P3sg-LOC
‘on the ninth of the month of September of 1982’

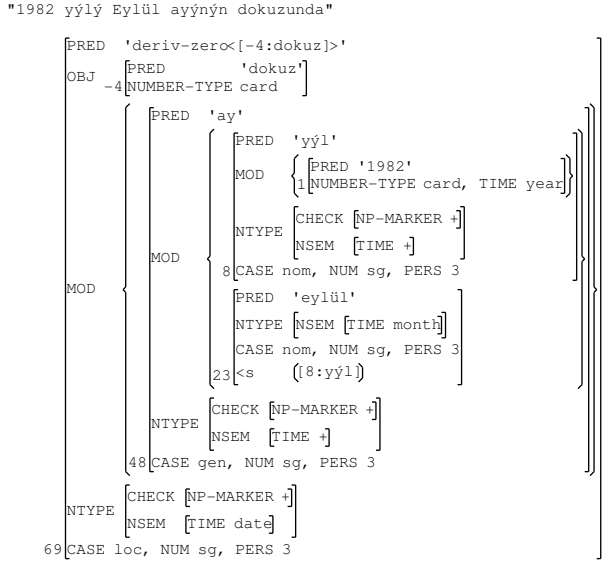


35. *1982nin Eylül ayının dokuzunda*
1982-GEN September month-P3sg-GEN nine-P3sg-LOC
‘on the ninth of the month of September of 1982’

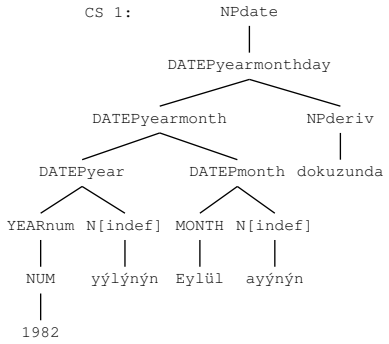


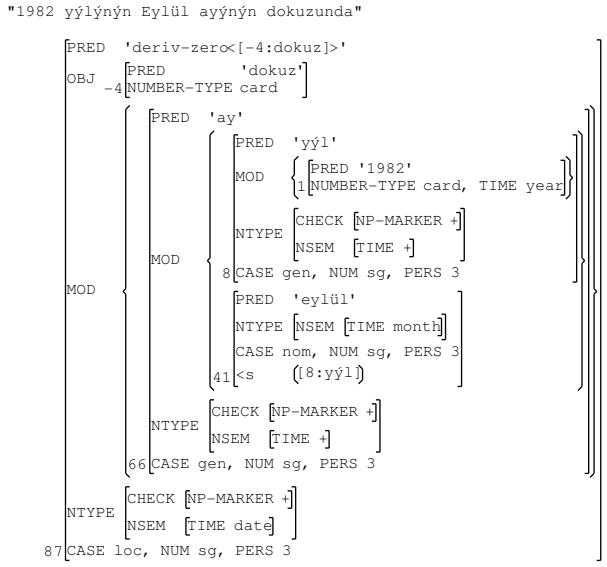
36. *1982 yılı Eylül ayının dokuzunda*
1982 year-P3sg September month-P3sg-GEN nine-P3sg-LOC
‘on the ninth of the month of September of the year 1982’



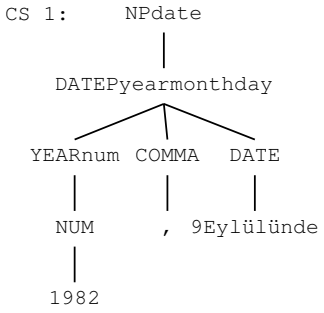


37. *1982 yılının Eylül ayının dokuzunda*
1982 year-P3sg-GEN September month-P3sg-GEN nine-P3sg-LOC
‘on the ninth of the month of September of the year 1982’





38. *1982, 9 Eylülünde*
1982, 9 September-P3sg-LOC
'on September the 9th, 1982'

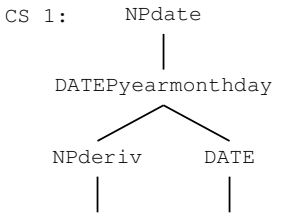


"1982, 9 Eylülünde"

$$\begin{bmatrix} \text{PRED} & \text{'9 eylül'} \\ \text{MOD} & \left\{ 1 \begin{bmatrix} \text{PRED} & \text{'1982'} \\ \text{NUMBER-TYPE} & \text{card, TIME year} \end{bmatrix} \right\} \\ \text{NTYPE} & \begin{bmatrix} \text{CHECK} & [\text{NP-MARKER} +] \\ \text{NSEM} & [\text{TIME date}] \end{bmatrix} \\ 8 & \begin{bmatrix} \text{CASE loc, NUM sg, PERS 3} \end{bmatrix} \end{bmatrix}$$

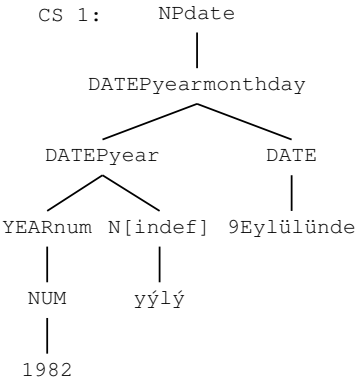
39. *1982'nin 9 Eylülünde*
1982-GEN 9 September-P3sg-LOC
'on September the 9th of 1982'

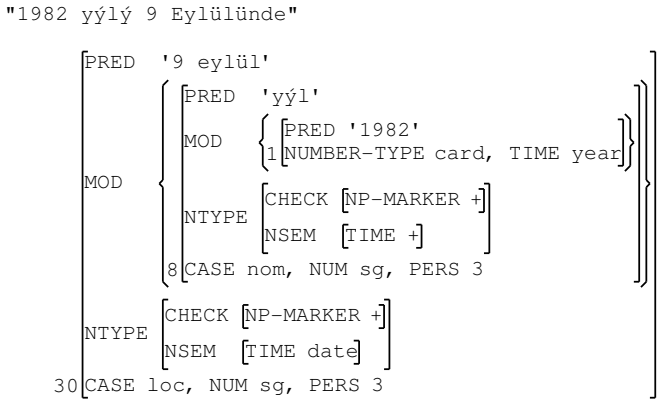
"1982'nin 9 Eylülünde"



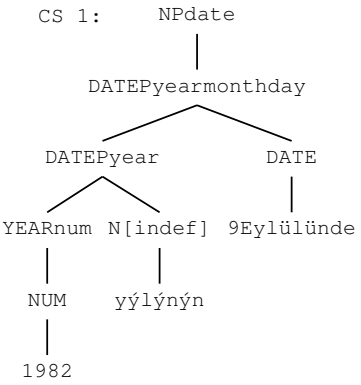
$$\begin{bmatrix} \text{PRED} & \text{'9 eylül'} \\ \text{MOD} & \left\{ 1 \begin{bmatrix} \text{PRED} & \text{'deriv-zero<[-1:1982]>'} \\ \text{OBJ} & -1 \begin{bmatrix} \text{PRED} & \text{'1982'} \\ \text{NUMBER-TYPE} & \text{card, TIME year} \end{bmatrix} \\ \text{NTYPE} & [\text{NSEM} [\text{TIME year}]] \\ & \text{CASE gen, NUM sg, PERS 3} \end{bmatrix} \right\} \\ \text{NTYPE} & \begin{bmatrix} \text{CHECK} & [\text{NP-MARKER} +] \\ \text{NSEM} & [\text{TIME date}] \end{bmatrix} \\ 32 & \begin{bmatrix} \text{CASE loc, NUM sg, PERS 3} \end{bmatrix} \end{bmatrix}$$

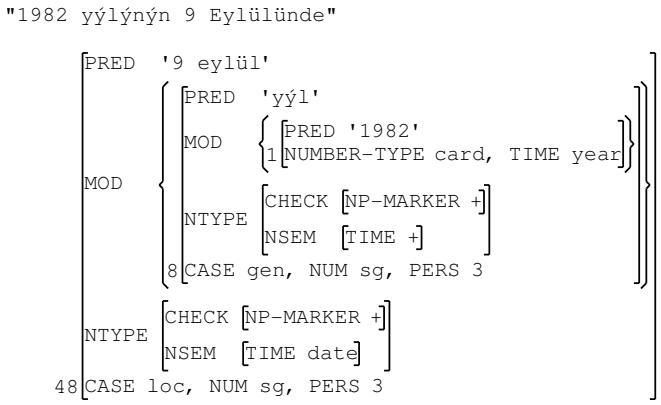
40. *1982 yılı 9 Eylülünde*
1982 year-P3sg 9 September-P3sg-LOC
'on September the 9th of the year 1982'



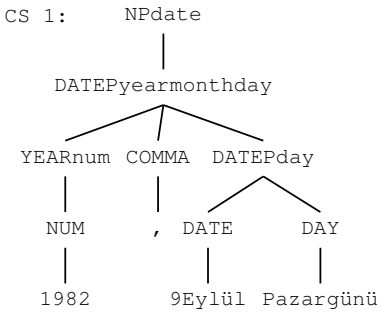


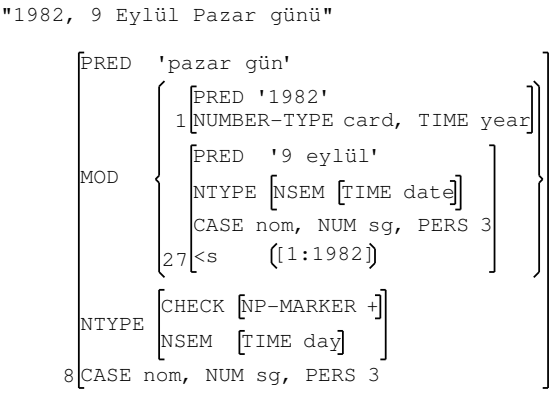
41. *1982 yılının 9 Eylülünde*
1982 year-P3sg-GEN 9 September-P3sg-LOC
'on September the 9th of the year 1982'



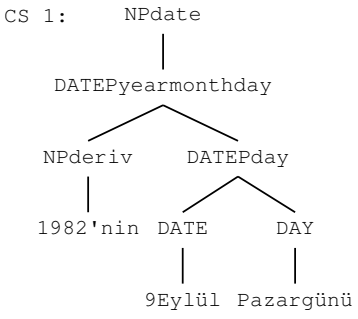


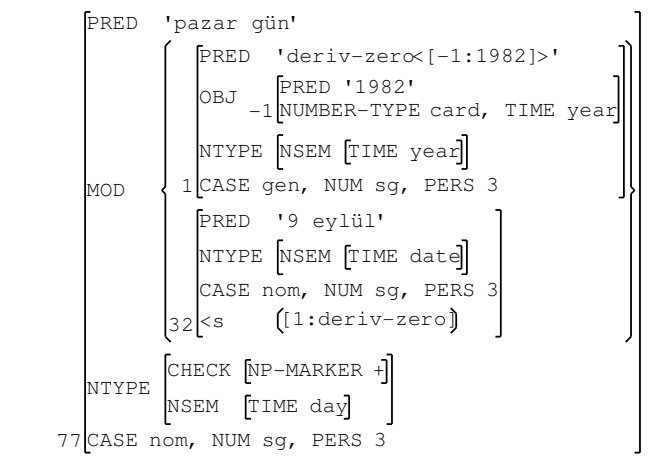
42. *1982, 9 Eylül Pazar günü*
1982, 9 September Sunday day-P3sg
'on Sunday, September the 9th, 1982'



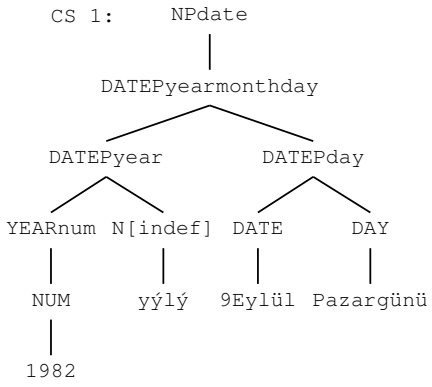


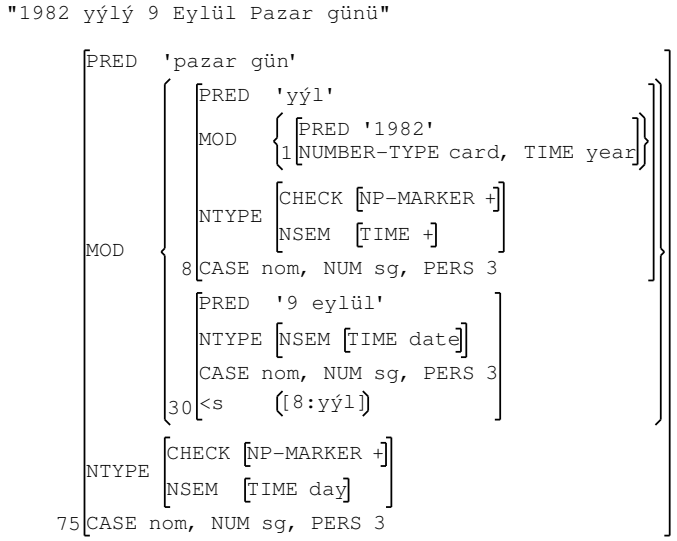
43. *1982'nin 9 Eylül Pazar günü*
1982-GEN 9 September Sunday day-P3sg
'on Sunday, September the 9th of 1982'



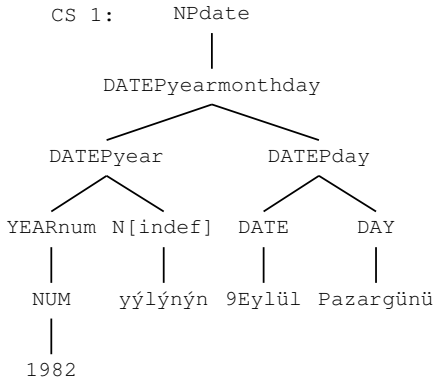


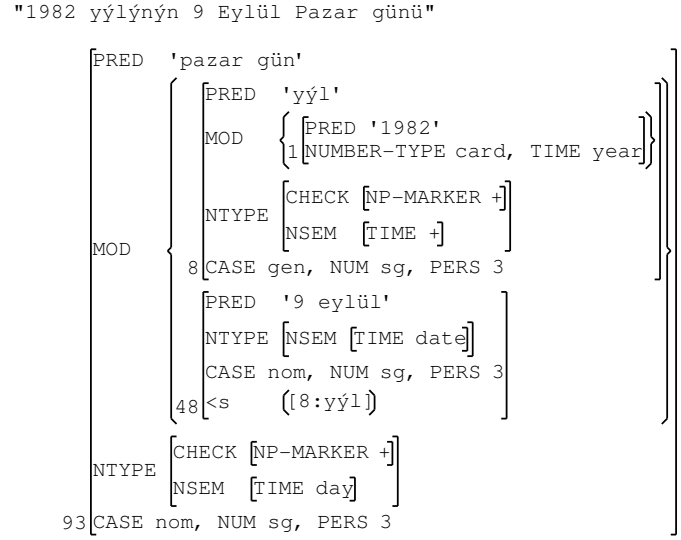
44. *1982 yılı 9 Eylül Pazar günü*
1982 year-P3sg 9 September Sunday day-P3sg
‘on Sunday, September the 9th of the year 1982’



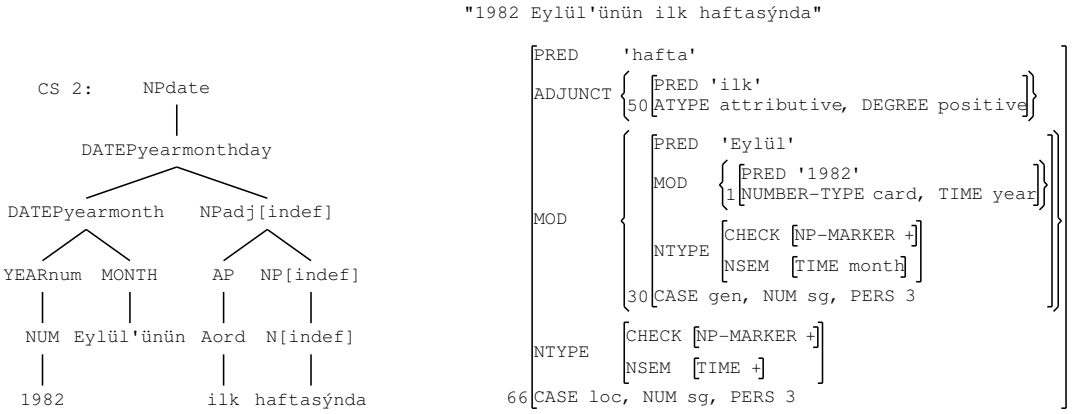


45. *1982 yılının 9 Eylül Pazar günü*
1982 year-P3sg-GEN 9 September Sunday day-P3sg
‘on Sunday, September the 9th of the year 1982’

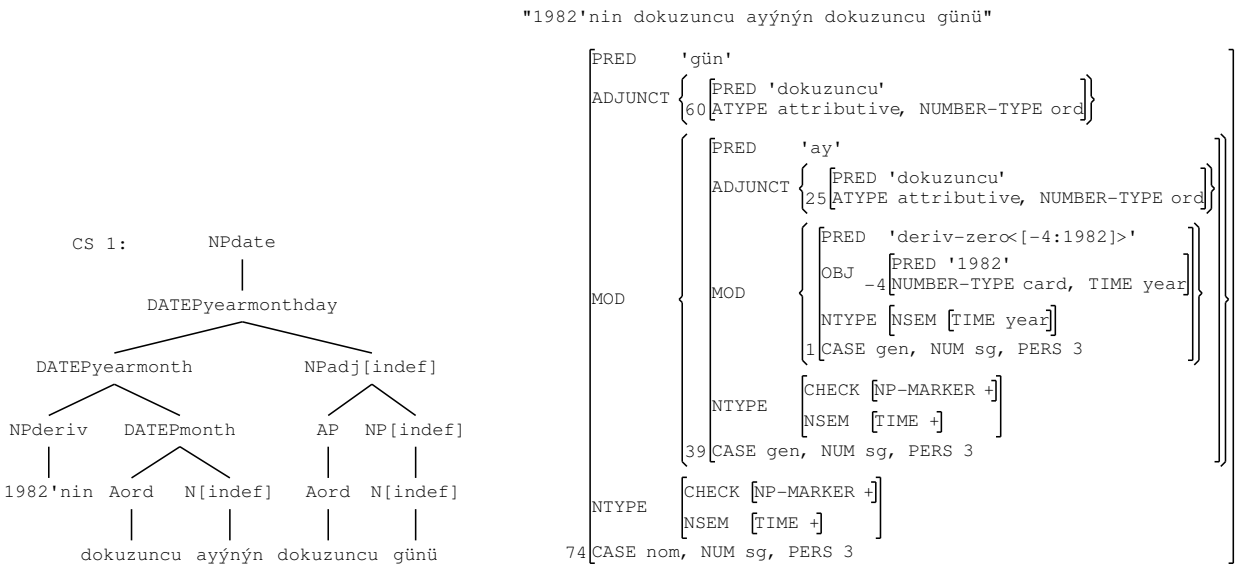




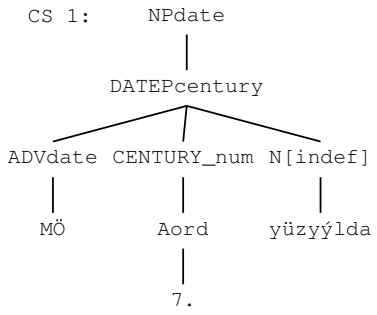
46. *1982 Eylülünün ilk haftasında*
1982 September-P3sg-GEN first week-P3sg-LOC
'in the first week of September of 1982 '

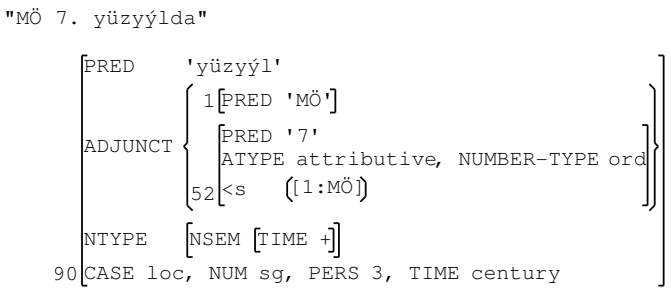


47. *1982'nin dokuzuncu ayının dokuzuncu günü*
1982-GEN ninth month-P3sg-GEN ninth day-P3sg
'on the ninth day of the ninth month of 1982 '

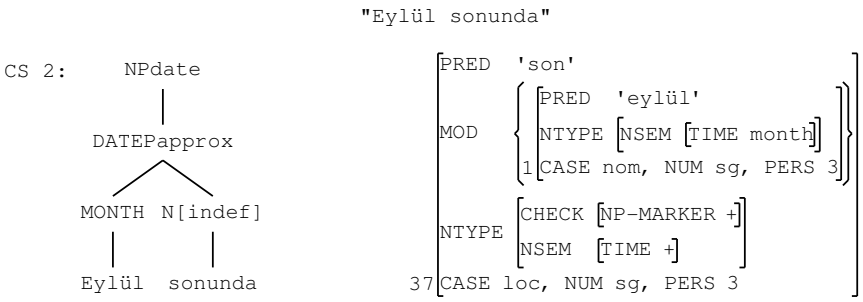


48. *MÖ 7. yüzyılda*
B.C. 9th century-LOC
'in the ninth century B.C.'

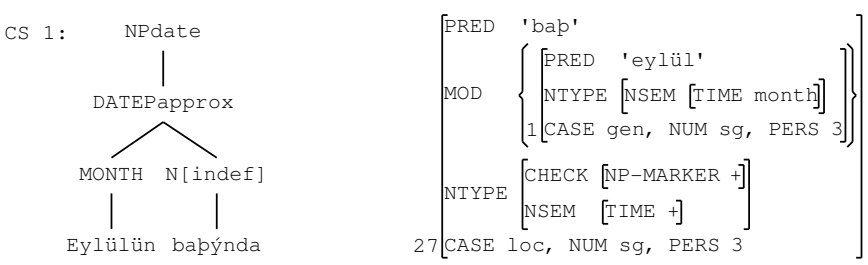




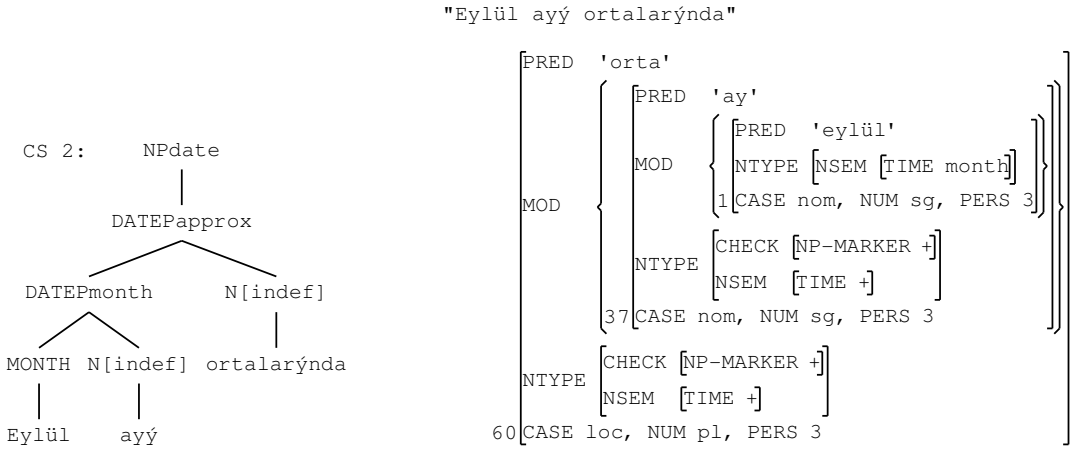
49. *Eylül sonunda*
September end-P3sg-LOC
'at the end of September'



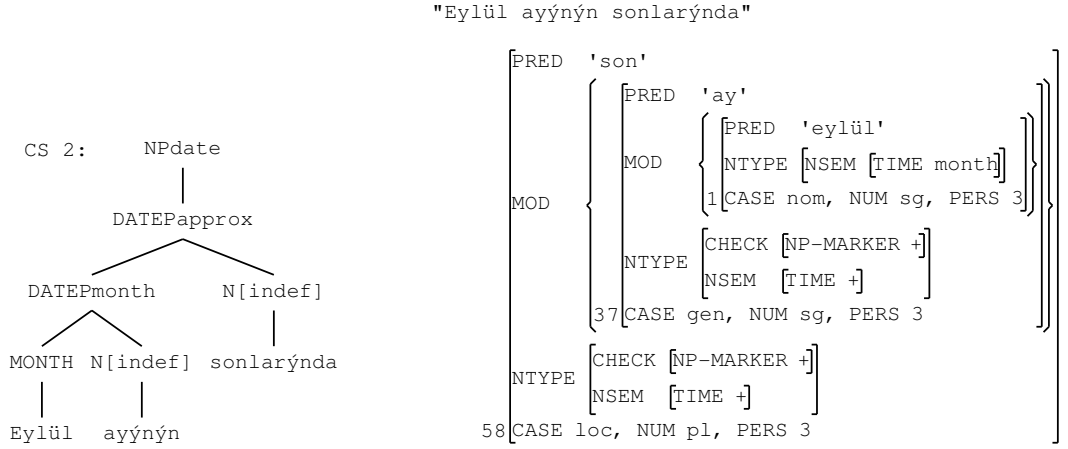
50. *Eylülün başında*
September-GEN beginning-P3sg-LOC
'at the beginning of September'



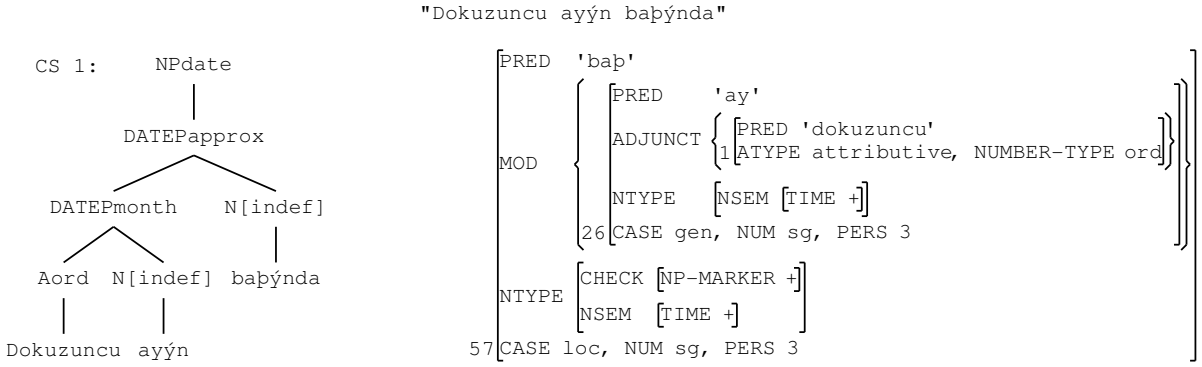
51. *Eylül* *ayı* *ortalarında*
September month-P3sg middle-PL-P3sg-LOC
'in the midst of the month September'



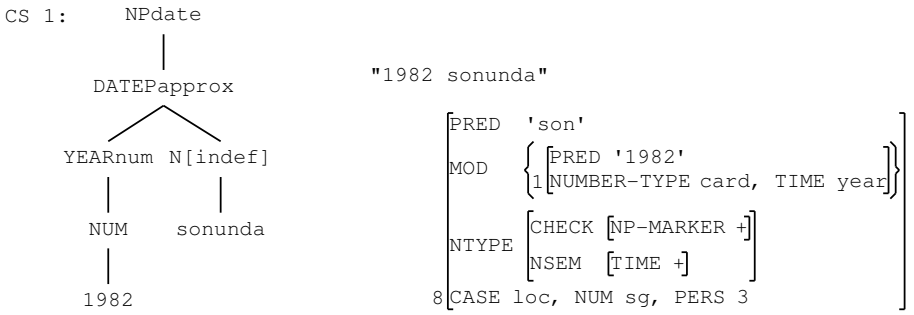
52. *Eylül* *ayının* *sonlarında*
September month-P3sg-GEN end-PL-P3sg-LOC
'at the end of the month September'



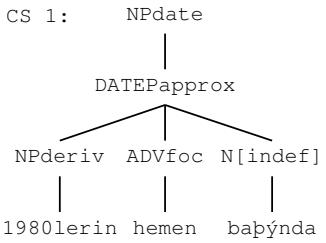
53. *Dokuzuncu* *ayın* *başında*
ninth month-GEN beginning-P3sg-LOC
'at the beginning of the ninth month'



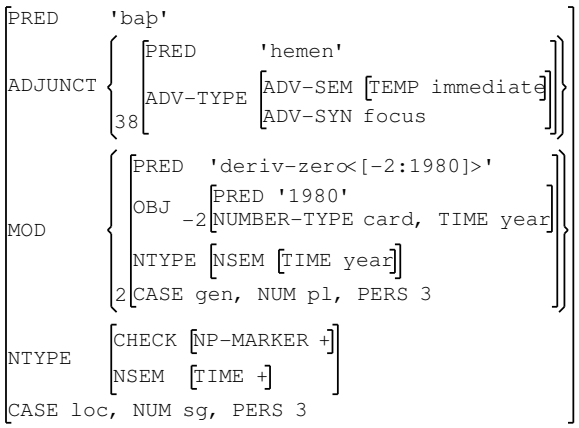
54. *1982* *sonunda*
1982 end-P3sg-LOC
'at the end of 1982'



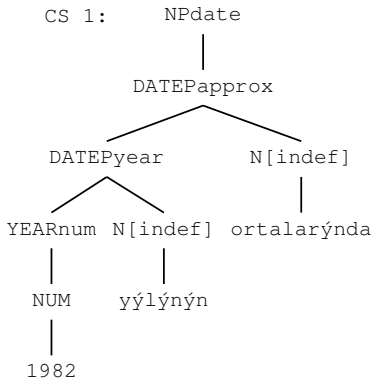
55. *1980'lerin hemen başında*
1980-PL-GEN right beginning-P3sg-LOC
'right at the beginning of 1980s'



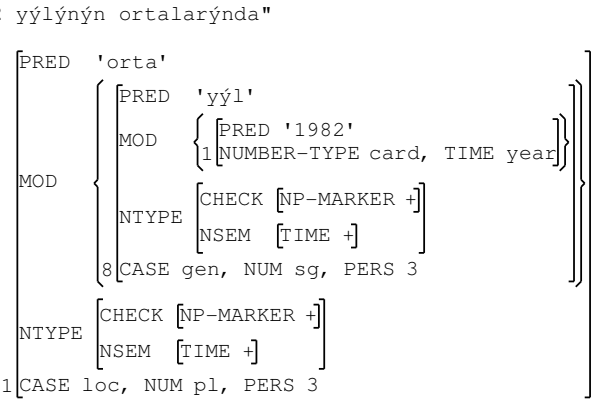
"1980lerin hemen başında"



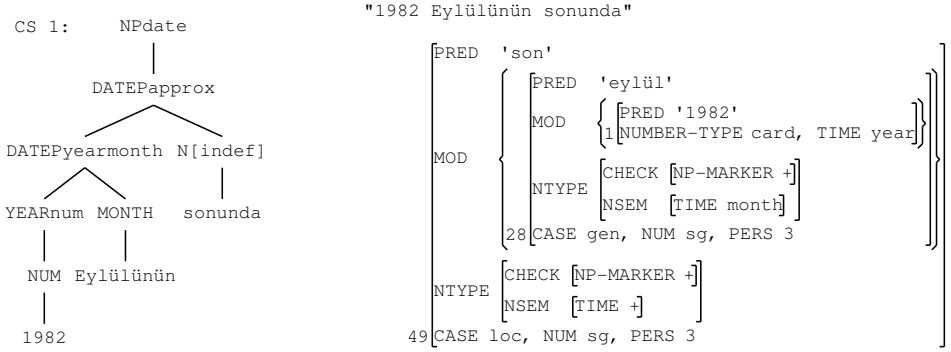
56. *1982 yılının ortalarında*
1982 year-P3sg-GEN middle-PL-P3sg-LOC
'in the midst of the year 1982'



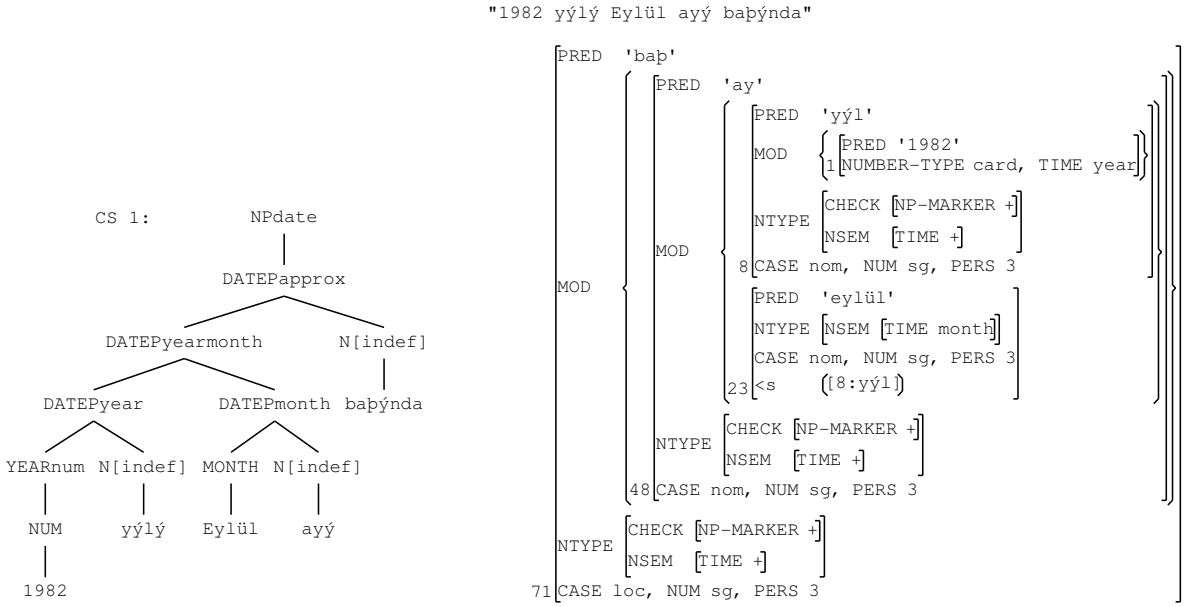
"1982 yılının ortalarında"



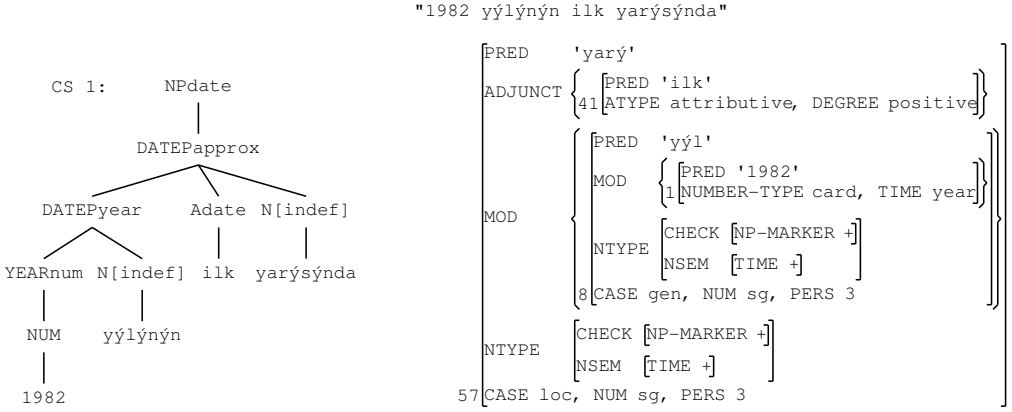
57. *1982 Eylülünün sonunda*
1982 September-P3sg-GEN end-P3sg-LOC
‘at the end of September of 1982’



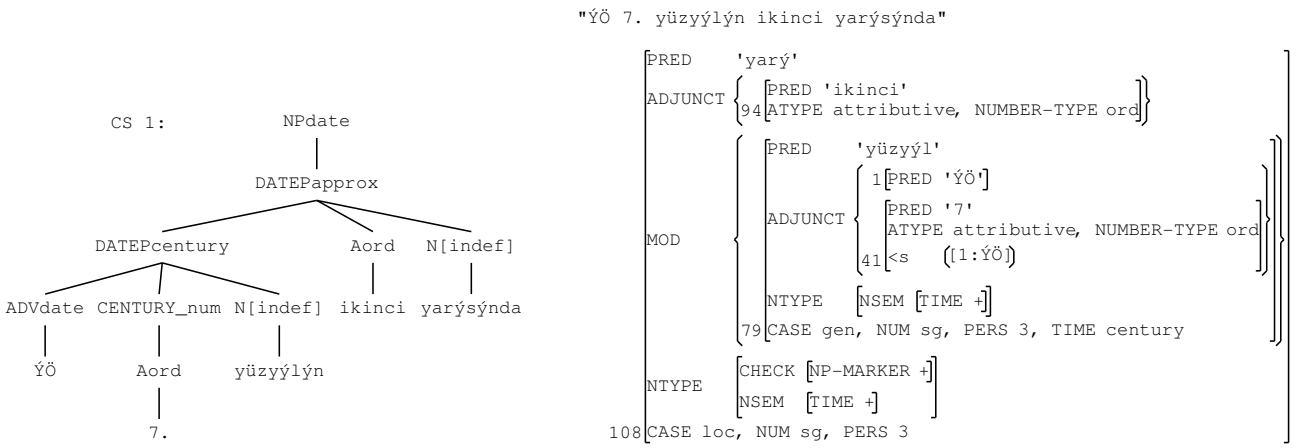
58. *1982 yılı Eylül ayı başında*
1982 year-P3sg September month-P3sg beginning-P3sg-LOC
‘at the beginning of the month September of the year 1982’



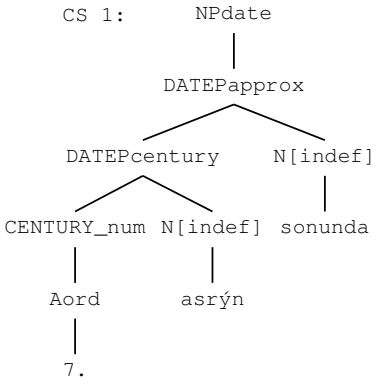
59. *1982 yılının ilk yarısında*
1982 year-P3sg-GEN first half-P3sg-LOC
‘in the first half of the year 1982’

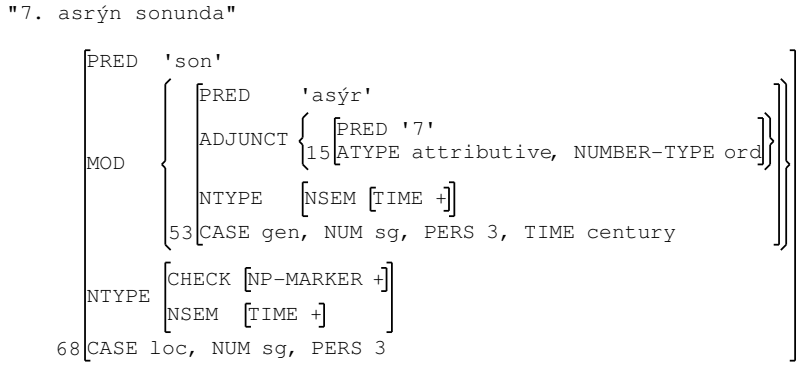


60. *İÖ 7. yüzyılın ikinci yarısında*
B.C. 9th century-GEN second half-P3sg-LOC
‘in the second half of the 9th century B.C.’

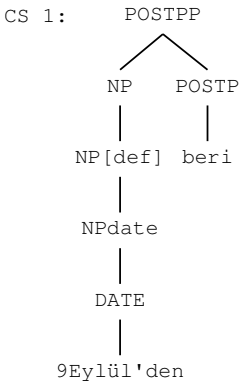


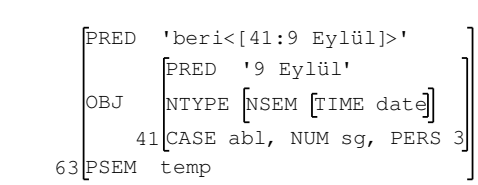
61. *7. asrın sonunda*
9th century-GEN end-P3sg-LOC
'at the end of the 9th century'



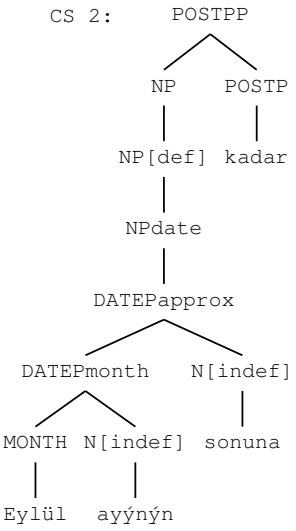


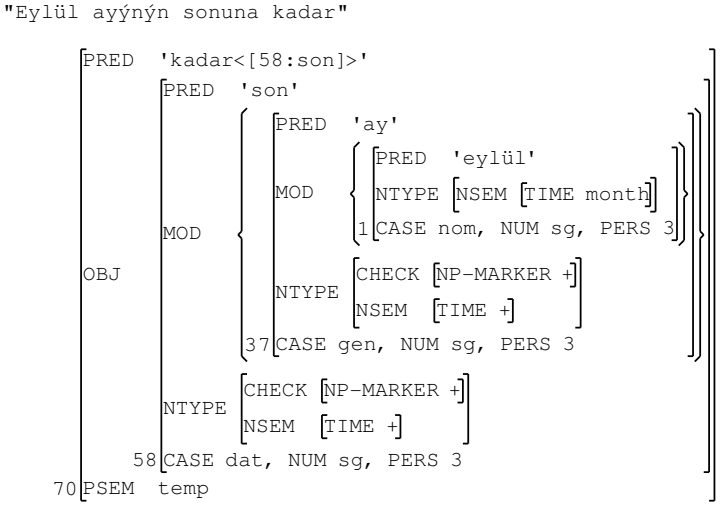
62. *9 Eylülden beri*
9 September-ABL since
'since the 9th of September'



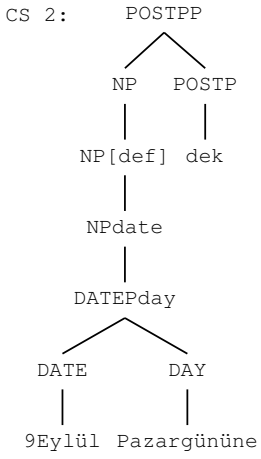


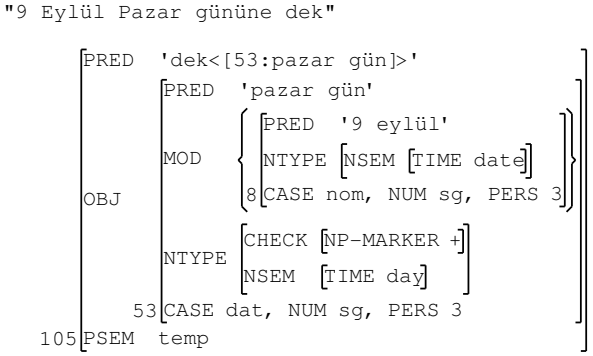
63. *Eylül* *ayının* *sonuna* *kadar*
September month-P3sg-GEN end-P3sg-DAT till
'till the end of the month September'



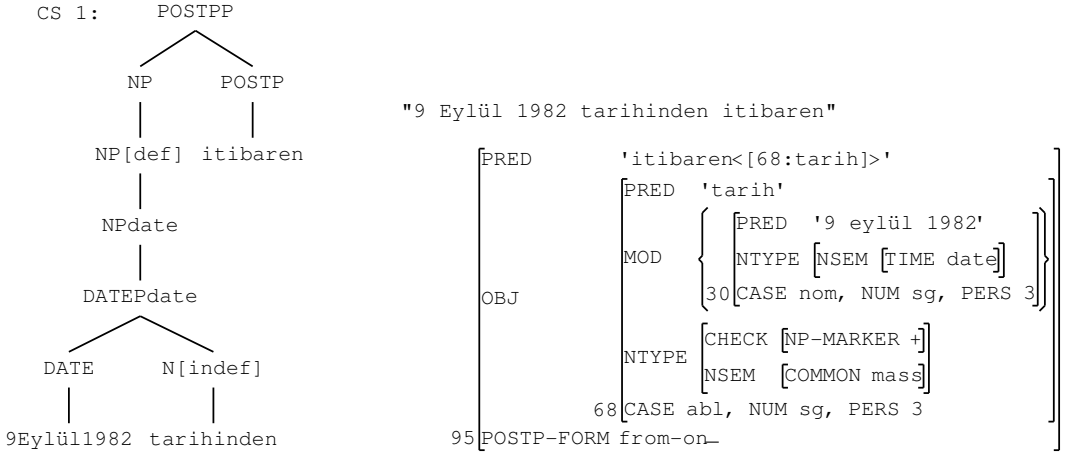


64. *9 Eylül* *Pazar* *gününe* *dek*
9 September Sunday day-P3sg-DAT till
'till Sunday, September the 9th'

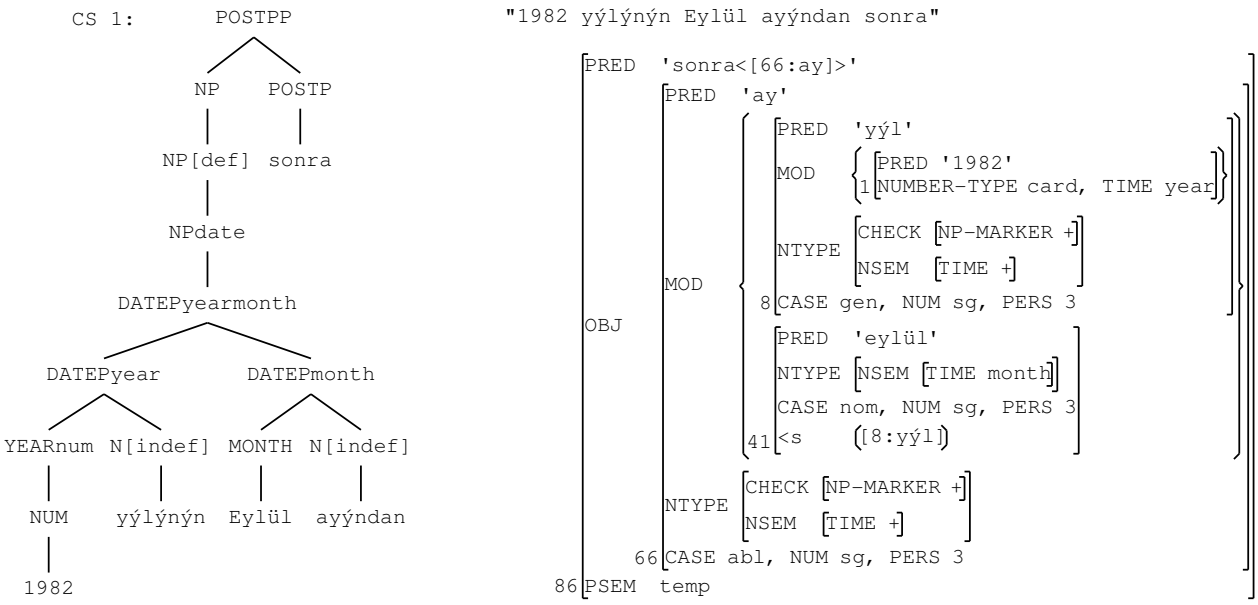




65. *9 Eylül 1982 tarihinden itibaren*
9 September 1982 date-P3sg-ABL from-on
'from the date September the 9th, 1982 on'



66. *1982 yılının Eylül ayından sonra*
1982 year-P3sg-GEN September month-P3sg-ABL after
'after the month of September of the year 1982'



BIOGRAPHY

Tuba Gümüş, having received a B.Sc. degree in computer engineering from Istanbul Technical University in 2004, has chosen to work on issues of computational linguistics for her master studies. In the period September 2005–September 2006, she has been at the Institut für Maschinelle Sprachverarbeitung (Natural Language Processing Institute) of Universitaet Stuttgart, Germany. She has been supported by the ERASMUS programme scholarship during her stay at IMS.