CKY Parser for Turkish Grammar

Göktuğ Öcal and Tunga Güngör
Boğaziçi University
Department of Computer Engineering
34342 Bebek, Istanbul, Turkey
{goktug.ocal, gungort}@boun.edu.tr

Abstract

Syntactic parsing is the process of analyzing a sentence or a piece of text and determining its grammatical structure. This includes identifying the constituent phrases and dependencies between the words, as well as determining the roles played by each word in the sentence (such as the subject, verb, and object). In this project, we have developed a Turkish Language CKY Parser which is fed from Chomsky's Normal Form (CNF) grammar rules and a lexicon. The process of parsing and the generation of CNF rules and lexicon are described. All the codes have been accessible on GitHub[1]

1 Introduction

Artificial intelligence (AI) is a field of computer science and engineering that focuses on the creation of intelligent machines that can think and act like humans. It involves developing algorithms and systems that can learn from data, make decisions, and perform tasks that would normally require human intelligence, such as recognizing patterns, solving problems, and adapting to new situations. Natural language processing (NLP) is a subfield of AI that deals with the interactions between computers and human (natural) languages. It involves developing systems and algorithms that can understand, interpret, and generate human language. NLP is closely related to AI because it involves using AI techniques, such as machine learning and deep learning, to analyze and understand natural language data. NLP algorithms are often used to enable AI systems to interact with humans in a more natural way, such as through speech or text.

Syntactic parsing is the process of analyzing a sentence or a piece of text and determining its grammatical structure. This includes identifying the constituent phrases and dependencies between the words, as well as determining the roles played by each word in the sentence (such as the subject, verb, and object). Syntactic parsing is an important task in natural language processing (NLP) because it helps to extract meaning from text and understand the relationships between words and phrases. It is also a key step in many NLP tasks, such as machine translation, text summarization, and information extraction. There are several approaches to syntactic parsing, including rule-based approaches that rely on a set of predefined grammar rules, and statistical approaches that use machine learning algorithms to learn the patterns of a language from annotated corpora. Syntactic parsing can be challenging because natural languages are often ambiguous and have complex grammatical structures. However, advances in NLP have made it possible to develop increasingly accurate syntactic parsers that can handle a wide range of languages and text types.

CKY parsing (also known as Cocke-Kasami-Younger parsing or chart parsing) is a dynamic programming algorithm used to parse natural language sentences and determine their syntactic structure. It was first proposed by John Cocke, Tadao Kasami, and Daniel Younger in 1970 (Cocke and Schwartz, 1970), and it has become a widely used algorithm in natural language processing (NLP). CKY parsing works by constructing a chart (a two-dimensional matrix) that represents the possible parse trees for a given sentence. The chart is filled in bottom-up, starting with the individual words in the sentence and working up to the complete parse tree. To fill in the chart, CKY parsing uses a set of grammar rules that specify

[1] https://github.com/GoktugOcal/turkish-syntactic-parser

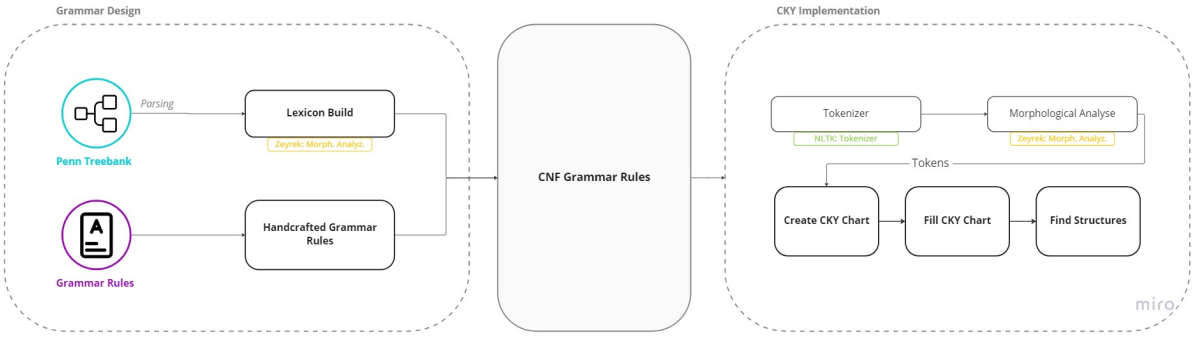


Figure 1: CKY Parser Diagram

the possible combinations of words and phrases in the language being parsed. The algorithm checks the grammar rules against the words in the sentence to see if they can be used to construct a valid parse tree. CKY parsing has several advantages over other parsing algorithms, including its speed and efficiency. It is able to parse sentences very quickly, even for large grammars, and it is able to handle ambiguities in the input by generating multiple possible parse trees.

2 The Project

The project requires to design a context-free grammar (CFG) which is limited with simple types of sentences, but will be able to handle declarative, imperative, and question sentences, and implement a parser for Turkish Language. The grammar rules should be designed with simple SOV sentence types. Then a CKY parser should be implemented which will be able to deal with agreement restrictions. In the whole process, a lexicon with POS tags and suffix types should be defined beforehand in order to not deal with morphological analyses and POS tagging.

3 The Parser

Our approach while building the parser has couple of steps but mainly divided into two steps; grammar design and parser implementation. The grammar design has other steps in itself such as lexicon building with suffix information (such as which number of person and roles such as accusative, dative or imperative etc.), handcrafting grammar rules. The CKY parser implementation is simply algorithm implementation.

In the project all the implementations have been implemented with Python 3. For tokenization, NLTK[2] module and for morphological analysis, ZEYREK[3] module have been used. All the grammar files have been stored in JSON and TXT formats. While building the lexicon and other grammar rules, Turkish grammar rules[4] and a simple corpus for test cases have been used.

3.1 Grammar Design

Designing a proper grammar is the most crucial part in CKY parser. The grammar rules are formed in CNF which includes possible sentence formats for given phrase and POS tags. For the Turkish Language, the rules would be more complicated since there are different types of multiple suffixes that indicates the tense of the sentence/verb or the person type of the subject.

Because of the grammar properties of Turkish; verbs, nouns and their suffixes should be investigated and their corresponding POS tags should be included in CNF Grammar Rules. In order to do that, ZEYREK morphology analyzer have been utilized and the rules have been investigated. Also known grammar rules are handcrafted and finally all the rules combined into CNF rule format.

[2] https://www.nltk.org/
[3] https://github.com/obulat/zeyrek
[4] https://en.wikipedia.org/wiki/Turkish_grammar

Table 1: POS tag and their some subcategories

Original POS	Subcategory POS
VP	VPPAST1
	VPPAST2
	VPPAST3
	VPPAST1PL
	VPPAST2PL
	VPPAST3PL
NP	NP1
	NP2
	NP3
	NPACC
	NPACC
PRO	PRO1
	PRO1PL
	PRO2

Morphological Analysis

The ZEYREK morphological analyzer provides couple of feature for a given token such as POS tag and morphemes. We have utilized the analyzer for extracting proper POS tags with our custom grammar defined for Turkish language. The analyzer returns multiple possible POS tags and morphemes, and that increases the possible Parse Trees. A proper procedure should had been implemented in order to handle that problem.

A helper function morphological analyzer have been created in order to create subcategories of POS tags with some predefined rules which have been given in Appendix B. The subcategories created with considering different clauses and number of person properties of most of POSes and clauses/phrases. For each token that were extracted from test corpus, the morphological analyzer have been executed and the desired ones have been selected manually in order to reduce number of possibilities for test cases.

Agreements

The subcategorization of the POS tags helps for time and number of person agreements when the CNF rules for the agreements have been defined. One example is given in Table 1. In these examples, we set CNF rules that would ensure agreements such as only pronouns with first person is going to establish a sentence with verbs with first person. Variety of rules for that kind of agreements have been established in CNF rules and given in Appendix A.

Lexicon Build

Couple of sentences that have been given in the project description have been selected to create lexicon for a test corpus. For that test corpus, tokens from senteces have been extracted and the possible POS tags collected from Morphological Procedure process. From those collected POS tags, the suitable ones were selected for each token as one token has only one POS tag, for the sake of simplicity and given to the Lexicon. One crucial point is since we have manually selected the POS tags for test corpus, the tokens which do not exist in the test corpus may have multiple possible POS tags and that increases the number of possible sentence structures in the end. For that cases, most suitable structure should be selected.

Grammar Rules

As mentioned above the grammar rules have three different components, Lexicon, POS tags collected from Morphological Analyse, and handcrafted ones. The first two were described in the previous sections, handcrafted ones are formed from web resources of grammar rules of Turkish and some rules have been given in Appendix A. When all the grammar rules are merged together (lexicon, morphological and handcrafted) overall grammar rule size became 303. Howewer, the grammar rules are dynamic since we

function CKY-PARSE(*words, grammar*) **returns** *table*

for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  $table[j-1, j] \leftarrow \{A \mid A \rightarrow words[j] \in grammar\}$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      $table[i,j] \leftarrow table[i,j] \cup$
        $\{A \mid A \rightarrow BC \in grammar,$
          $B \in table[i,k],$
          $C \in table[k,j]\}$

Figure 2: CKY Algorithm(Jurafsky and Martin, 2009)

have created a Morphological Analyzer that would return POS tags for tokens that does not exist in the lexicon.

Best Structure Selection

Since the possible structures for a given sentence can be multiple, we have defined a possibility metric that would reveal the most suitable structure. For that purpose, possibilities for each POS tag and non-terminal have been calculated with using the CNF grammar. The simple assumption is that the possibility of a non-terminal is the number of occurance of that non-terminal in the grammar rules divided by the total number of rules;

$$P(nt_i) = -\log\left(\frac{c(nt_i, CNF)}{c(CNF)}\right) \tag{1}$$

where nt_i is a non-terminal and CNF is the all CNF rules. That assumption gives a probability for every non-terminals and in first step only assigned for terminal nodes of the tree. For the rest of the nodes in the tree, the probability is calculated with the multiplication of the left and the right childs of the node and the probability of the non-terminal of the node;

$$P_{node} = \begin{Bmatrix} P(nt_i) & \text{if node is terminal} \\ P_{left} * P_{right} * P(nt_i) & \text{else} \end{Bmatrix} \tag{2}$$

With that assumption, for each of the sentence, we can select best parse by just observing the probabilities of the non-terminals with ”S” and the smallest one would be selected.

3.2 Parser Algorithm

The parser algorithm has simply two main and totally 5 steps as shown in Figure 1. In the first steps, tokenization and morphological analyzer with NLTK and ZEYREK packages are run in order to create useful POS tags for CKY parsing.

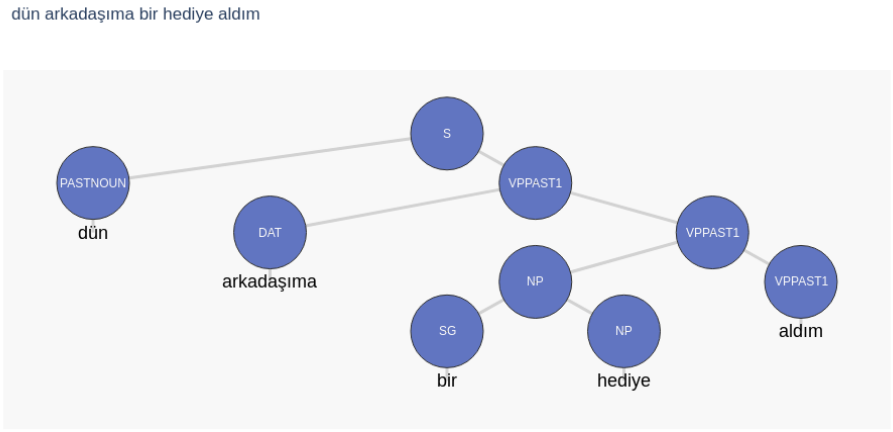
At the following steps, an traditional CKY parsing algorithm (Jurafsky and Martin, 2009) is utilized and the psudocode of the algorithm given in Figure 2. First of all, a CKY chart is utilized and its diagonal filled with the POS tags of the tokens of given and parsed sentence. Then the proposed algorithm applied and the resulting CKY chart has created.

3.3 Results

The test corpus was given in the project description and shown below with parsign results. For each of them the parsing have been executed and the a custom tree and POS tag visualizations have been created with a simple visualization module created by us using Plotly[5] and Spacy[6] modules.

- Dün arkadaşıma bir hediye aldım.

```
Tokens : ['dün', 'arkadaşıma', 'bir', 'hediye', 'aldım']
POS Tags : [['PASTNOUN'], ['DAT'], ['SG'], ['NP'], ['VPPAST1']]
Sentence is grammatically correct.
(S(PASTNOUN dün ) (VPPAST1(DAT arkadaşıma ) (VPPAST1(NP(SG bir ) (NP hediye ) )(VPPAST1 aldım ) ))) Score : 23172.44

######### CKY CHART #########
------------  ----------  ------  ------  ----------------
dün           arkadaşıma  bir     hediye  aldım
['PASTNOUN']  []          []      []      ['S', 'VPPAST1']
[]            ['DAT']     []      []      ['VPPAST1']
[]            []          ['SG']  ['NP']  ['S', 'VPPAST1']
[]            []          []      ['NP']  ['S', 'VPPAST1']
[]            []          []      []      ['VPPAST1']
------------  ----------  ------  ------  ----------------
##### BEST SENTENCE STRUCTURE #####
(S(PASTNOUN dün ) (VPPAST1(DAT arkadaşıma ) (VPPAST1(NP(SG bir ) (NP hediye ) )(VPPAST1 aldım ) )))
```

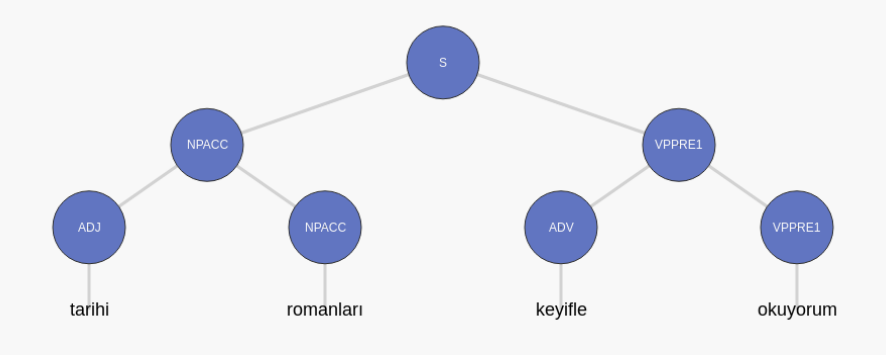


- Tarihi romanları keyifle okuyorum.

```
Tokens : ['tarihi', 'romanları', 'keyifle', 'okuyorum']
POS Tags : [['ADJ'], ['NPACC'], ['ADV'], ['VPPRE1']]
Sentence is grammatically correct.
(S(NPACC(ADJ tarihi ) (NPACC romanları ) )(VPPRE1(ADV keyifle ) (VPPRE1 okuyorum ) )) Score : 3864.13

######### CKY CHART #########
-------  ---------  -------  ----------
tarihi   romanları  keyifle  okuyorum
['ADJ']  ['NPACC']  []       ['S']
[]       ['NPACC']  []       ['S']
[]       []         ['ADV']  ['VPPRE1']
[]       []         []       ['VPPRE1']
-------  ---------  -------  ----------
##### BEST SENTENCE STRUCTURE #####
(S(NPACC(ADJ tarihi ) (NPACC romanları ) )(VPPRE1(ADV keyifle ) (VPPRE1 okuyorum ) ))
```

[5] https://plotly.com/
[6] https://spacy.io/

tarihi romanları keyifle okuyorum

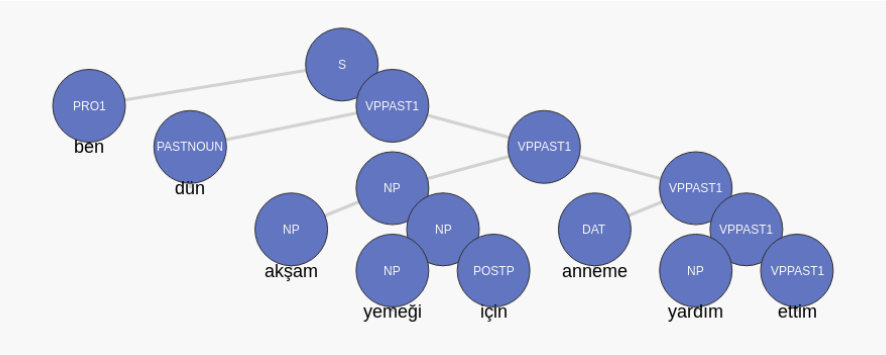


- Ben dün akşam yemeği için anneme yardım ettim.

```
Tokens : ['ben', 'dün', 'akşam', 'yemeği', 'için', 'anneme', 'yardım', 'ettim']
POS Tags : [['PRO1'], ['PASTNOUN'], ['NP'], ['NP'], ['POSTP'], ['DAT'], ['NP'], ['VPPAST1']]
Sentence is grammatically correct.
(S(PRO1 ben ) (VPPAST1(PASTNOUN dün ) (VPPAST1(NP akşam ) (VPPAST1(NP(NP yemeği ) (POSTP için ) )(VPPAST1(DAT anneme ) (VPPAST1(NP yardım )
Score : 29464808.43
(S(PRO1 ben ) (VPPAST1(PASTNOUN dün ) (VPPAST1(NP(NP akşam ) (NP(NP yemeği ) (POSTP için ) ))(VPPAST1(DAT anneme ) (VPPAST1(NP yardım ) (VPP
Score : 22828590.32
(S(PRO1 ben ) (VPPAST1(PASTNOUN dün ) (VPPAST1(NP(NP(NP akşam ) (NP yemeği ) )(POSTP için ) )(VPPAST1(DAT anneme ) (VPPAST1(NP yardım ) (VPP
Score : 22828590.32

######### CKY CHART #########
--------  ------------  ------  ------  ------------  -------  ------  ----------------------------------------------------
ben       dün           akşam   yemeği  için          anneme   yardım  ettim
['PRO1']  []            []      []      []            []       []      ['S', 'S', 'S']
[]        ['PASTNOUN']  []      []      []            []       []      ['S', 'VPPAST1', 'S', 'VPPAST1', 'S', 'VPPAST1']
[]        []            ['NP']  ['NP']  ['NP', 'NP']  []       []      ['S', 'VPPAST1', 'S', 'S', 'VPPAST1', 'S', 'VPPAST1']
[]        []            []      ['NP']  ['NP']        []       []      ['S', 'S', 'VPPAST1']
[]        []            []      []      ['POSTP']     []       []      ['PPPASTCLAUSE1']
[]        []            []      []      []            ['DAT']  []      ['VPPAST1']
[]        []            []      []      []            []       ['NP']  ['S', 'VPPAST1']
[]        []            []      []      []            []       []      ['VPPAST1']
--------  ------------  ------  ------  ------------  -------  ------  ----------------------------------------------------
##### BEST SENTENCE STRUCTURE #####
(S(PRO1 ben ) (VPPAST1(PASTNOUN dün ) (VPPAST1(NP(NP akşam ) (NP(NP yemeği ) (POSTP için ) ))(VPPAST1(DAT anneme ) (VPPAST1(NP yardım ) (VPP
```

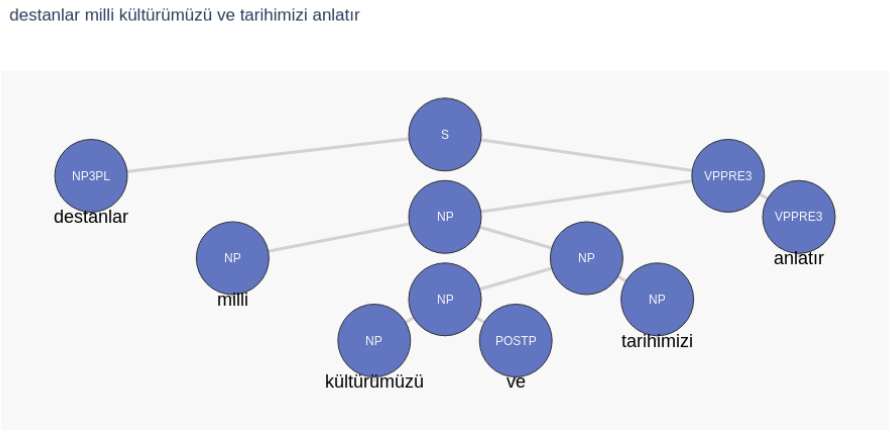
ben PRO1 dün PASTNOUN akşam NP yemeği NP için POSTP anneme DAT yardım NP ettim VPPAST1

ben dün akşam yemeği için anneme yardım ettim



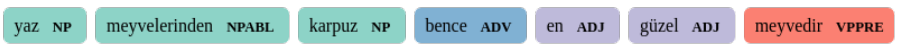
- Destanlar milli kültürümüzü ve tarihimizi anlatır.

```
Tokens : ['destanlar', 'milli', 'kültürümüzü', 've', 'tarihimizi', 'anlatır']
POS Tags : [['NP3PL'], ['NP'], ['NP'], ['POSTP'], ['NP'], ['VPPRE3']]
Sentence is grammatically correct.
(S(NP3PL destanlar ) (VPPRE3(NP milli ) (VPPRE3(NP(NP kültürümüzü ) (POSTP ve ) )(VPPRE3(NP tarihimizi ) (VPPRE3 anlatır ) ))))
Score : 301016.16
(S(NP3PL destanlar ) (VPPRE3(NP milli ) (VPPRE3(NP(NP(NP kültürümüzü ) (POSTP ve ) )(NP tarihimizi ) )(VPPRE3 anlatır ) )))
Score : 193189.47
(S(NP3PL destanlar ) (VPPRE3(NP(NP milli ) (NP(NP kültürümüzü ) (POSTP ve ) ))(VPPRE3(NP tarihimizi ) (VPPRE3 anlatır ) )))
Score : 193189.47
(S(NP3PL destanlar ) (VPPRE3(NP(NP(NP milli ) (NP kültürümüzü ) )(POSTP ve ) )(VPPRE3(NP tarihimizi ) (VPPRE3 anlatır ) )))
Score : 193189.47
(S(NP3PL destanlar ) (VPPRE3(NP(NP milli ) (NP(NP(NP kültürümüzü ) (POSTP ve ) )(NP tarihimizi ) ))(VPPRE3 anlatır ) ))
Score : 123987.27
(S(NP3PL destanlar ) (VPPRE3(NP(NP(NP milli ) (NP(NP kültürümüzü ) (POSTP ve ) ))(NP tarihimizi ) )(VPPRE3 anlatır ) ))
Score : 123987.27
(S(NP3PL destanlar ) (VPPRE3(NP(NP(NP(NP milli ) (NP kültürümüzü ) )(POSTP ve ) )(NP tarihimizi ) )(VPPRE3 anlatır ) ))
Score : 123987.27

######### CKY CHART #########
---------  ------  -----------  ------------  ------------------  ----------------------------------------------------------------------------------
destanlar  milli   kültürümüzü  ve            tarihimizi          anlatır
['NP3PL']  []      []           []            []                  ['S', 'S', 'S', 'S', 'S', 'S', 'S']
[]         ['NP']  ['NP']       ['NP', 'NP']  ['NP', 'NP', 'NP']  ['S', 'VPPRE3', 'S', 'VPPRE3', 'S', 'S', 'VPPRE3', 'S', 'VPPRE3', 'S', 'VP
[]         []      ['NP']       ['NP']        ['NP']              ['S', 'S', 'VPPRE3', 'S', 'VPPRE3']
[]         []      []           ['POSTP']     []                  ['PPPRECLAUSE3']
[]         []      []           []            ['NP']              ['S', 'VPPRE3']
[]         []      []           []            []                  ['VPPRE3']
---------  ------  -----------  ------------  ------------------  ----------------------------------------------------------------------------------
##### BEST SENTENCE STRUCTURE #####
(S(NP3PL destanlar ) (VPPRE3(NP(NP(NP milli ) (NP(NP(NP kültürümüzü ) (POSTP ve ) )(NP tarihimizi ) ))(VPPRE3 anlatır ) ))
```

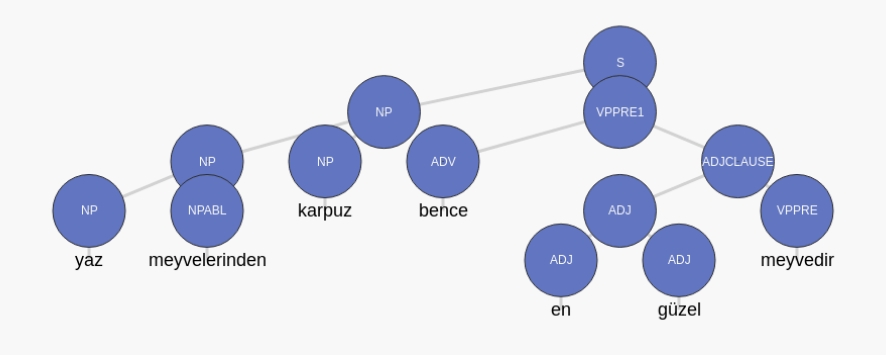


- Yaz meyvelerinden karpuz bence en güzel meyvedir.

```
Tokens : ['yaz', 'meyvelerinden', 'karpuz', 'bence', 'en', 'güzel', 'meyvedir']
POS Tags : [['NP', 'VPIMP'], ['NPABL'], ['NP'], ['ADV'], ['ADJ'], ['ADJ'], ['VPPRE']]
Sentence is grammatically correct.
(S(NP(NP yaz ) (NPABL meyvelerinden ) )(VPPRE1(NP karpuz ) (VPPRE1(ADV bence ) (ADJCLAUSE(ADJ(ADJ en ) (ADJ güzel ) )(VPPRE meyvedir ) ))))
Score : 4301621.89
(S(NP(NP(NP yaz ) (NPABL meyvelerinden ) )(NP karpuz ) )(VPPRE1(ADV bence ) (ADJCLAUSE(ADJ(ADJ en ) (ADJ güzel ) )(VPPRE meyvedir ) )))
Score : 3015811.0

######### CKY CHART #########
---------------  -------------  ------  -------  -------  -------  ------------------------------
yaz              meyvelerinden  karpuz  bence    en       güzel    meyvedir
['NP', 'VPIMP']  ['NP']         ['NP']  []       []       []       ['S', 'VPPRE1', 'S', 'VPPRE1']
[]               ['NPABL']      []      []       []       []       []
[]               []             ['NP']  []       []       []       ['S', 'VPPRE1']
[]               []             []      ['ADV']  []       []       ['VPPRE1']
[]               []             []      []       ['ADJ']  ['ADJ']  ['ADJCLAUSE']
[]               []             []      []       []       ['ADJ']  ['ADJCLAUSE']
[]               []             []      []       []       []       ['VPPRE']
---------------  -------------  ------  -------  -------  -------  ------------------------------
##### BEST SENTENCE STRUCTURE #####
(S(NP(NP(NP yaz ) (NPABL meyvelerinden ) )(NP karpuz ) )(VPPRE1(ADV bence ) (ADJCLAUSE(ADJ(ADJ en ) (ADJ güzel ) )(VPPRE meyvedir ) )))
```



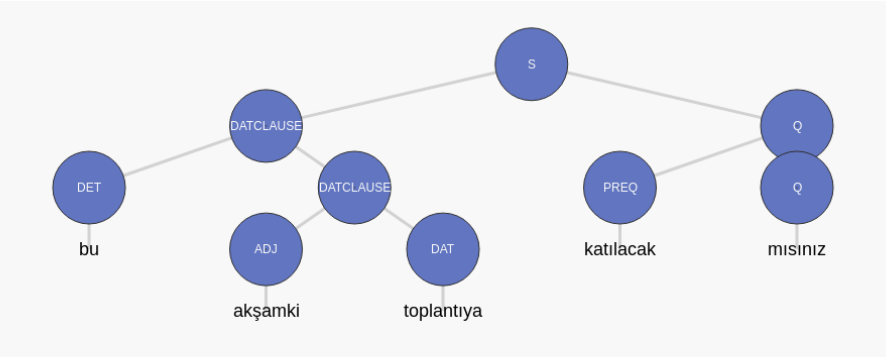
yaz meyvelerinden karpuz bence en güzel meyvedir



- Bu akşamki toplantıya katılacak mısınız?

```
Tokens : ['bu', 'akşamki', 'toplantıya', 'katılacak', 'mısınız']
POS Tags : [['DET'], ['ADJ'], ['DAT'], ['PREQ'], ['Q']]
Sentence is grammatically correct.
(S(DATCLAUSE(DET bu ) (DATCLAUSE(ADJ akşamki ) (DAT toplantıya ) ))(Q(PREQ katılacak ) (Q mısınız ) )) Score : 101271.43

######### CKY CHART #########
-------  -------  -------------  ---------  ----------
bu       akşamki  toplantıya     katılacak  mısınız
['DET']  []       ['DATCLAUSE']  []         ['S', 'Q']
[]       ['ADJ']  ['DATCLAUSE']  []         ['S', 'Q']
[]       []       ['DAT']        []         []
[]       []       []             ['PREQ']   ['Q']
[]       []       []             []         ['Q']
-------  -------  -------------  ---------  ----------
##### BEST SENTENCE STRUCTURE #####
(S(DATCLAUSE(DET bu ) (DATCLAUSE(ADJ akşamki ) (DAT toplantıya ) ))(Q(PREQ katılacak ) (Q mısınız ) ))
```

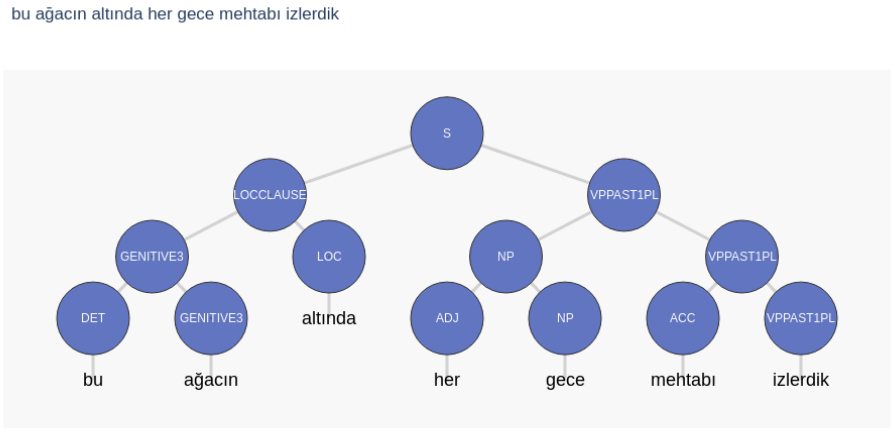
bu akşamki toplantıya katılacak mısınız



- Bu ağacın altında her gece mehtabı izlerdik.

```
Tokens : ['bu', 'ağacın', 'altında', 'her', 'gece', 'mehtabı', 'izlerdik']
POS Tags : [['DET'], ['GENITIVE3'], ['LOC'], ['DET', 'ADJ'], ['NP'], ['ACC'], ['VPPAST1PL']]
Sentence is grammatically correct.
(S(LOCCLAUSE(GENITIVE3(DET bu ) (GENITIVE3 ağacın ) )(LOC altında ) )(VPPAST1PL(NP(ADJ her ) (NP gece ) )(VPPAST1PL(ACC mehtabı ) (VPPAST1PL
Score : 33482049.83
```

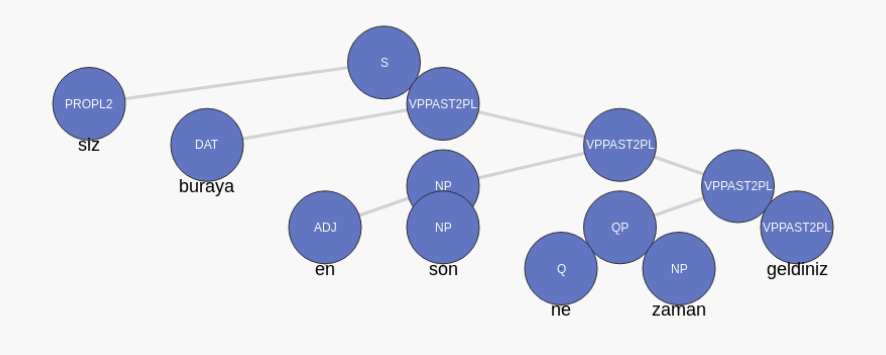
```
######### CKY CHART #########
-------  -------------  -------------  --------------  ------  -------  -------------
bu       ağacın         altında        her             gece    mehtabı  izlerdik
['DET']  ['GENITIVE3']  ['LOCCLAUSE']  []              []      []       ['S']
[]       ['GENITIVE3']  ['LOCCLAUSE']  []              []      []       ['S']
[]       []             ['LOC']        []              []      []       []
[]       []             []             ['DET', 'ADJ']  ['NP']  []       ['VPPAST1PL']
[]       []             []             []              ['NP']  []       ['VPPAST1PL']
[]       []             []             []              []      ['ACC']  ['VPPAST1PL']
[]       []             []             []              []      []       ['VPPAST1PL']
-------  -------------  -------------  --------------  ------  -------  -------------
##### BEST SENTENCE STRUCTURE #####
(S(LOCCLAUSE(GENITIVE3(DET bu ) (GENITIVE3 ağacın ) )(LOC altında ) )(VPPAST1PL(NP(ADJ her ) (NP gece ) )(VPPAST1PL(ACC mehtabı ) (VPPAST1PL
```



- Siz buraya en son ne zaman geldiniz?

```
Tokens : ['siz', 'buraya', 'en', 'son', 'ne', 'zaman', 'geldiniz']
POS Tags : [['PROPL2'], ['DAT'], ['ADJ'], ['NP', 'ADJ'], ['Q'], ['NP'], ['VPPAST2PL']]
Sentence is grammatically correct.
(S(PROPL2 siz ) (VPPAST2PL(DAT buraya ) (VPPAST2PL(NP(ADJ en ) (NP son ) )(VPPAST2PL(QP(Q ne ) (NP zaman ) )(VPPAST2PL geldiniz ) ))))
Score : 6582397.46

######### CKY CHART #########
----------  -------  -------  -------------  -----  ------  -------------
siz         buraya   en       son            ne     zaman   geldiniz
['PROPL2']  []       []       []             []     []      ['S']
[]          ['DAT']  []       []             []     []      ['VPPAST2PL']
[]          []       ['ADJ']  ['NP', 'ADJ']  ['S']  []      ['VPPAST2PL']
[]          []       []       ['NP', 'ADJ']  ['S']  []      ['VPPAST2PL']
[]          []       []       []             ['Q']  ['QP']  ['VPPAST2PL']
[]          []       []       []             []     ['NP']  ['VPPAST2PL']
[]          []       []       []             []     []      ['VPPAST2PL']
----------  -------  -------  -------------  -----  ------  -------------
##### BEST SENTENCE STRUCTURE #####
(S(PROPL2 siz ) (VPPAST2PL(DAT buraya ) (VPPAST2PL(NP(ADJ en ) (NP son ) )(VPPAST2PL(QP(Q ne ) (NP zaman ) )(VPPAST2PL geldiniz ) ))))
```

siz buraya en son ne zaman geldiniz

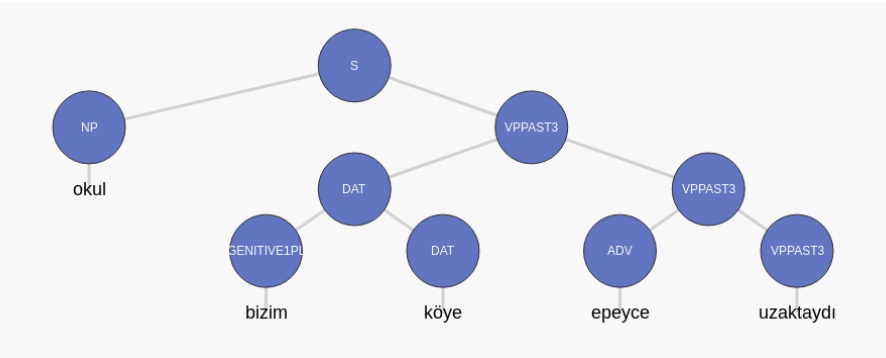


- Okul bizim köye epeyce uzaktaydı.

```
Tokens : ['okul', 'bizim', 'köye', 'epeyce', 'uzaktaydı']
POS Tags : [['NP'], ['GENITIVE1PL'], ['DAT'], ['ADV'], ['VPPAST3']]
Sentence is grammatically correct.
(S(NP okul ) (VPPAST3(DAT(GENITIVE1PL bizim ) (DAT köye ) )(VPPAST3(ADV epeyce ) (VPPAST3 uzaktaydı ) )))
Score : 35722.48

######### CKY CHART #########
------  ---------------  -------  -------  ----------------
okul    bizim            köye     epeyce   uzaktaydı
['NP']  []               []       []       ['S', 'VPPAST3']
[]      ['GENITIVE1PL']  ['DAT']  []       ['VPPAST3']
[]      []               ['DAT']  []       ['VPPAST3']
[]      []               []       ['ADV']  ['VPPAST3']
[]      []               []       []       ['VPPAST3']
------  ---------------  -------  -------  ----------------
##### BEST SENTENCE STRUCTURE #####
(S(NP okul ) (VPPAST3(DAT(GENITIVE1PL bizim ) (DAT köye ) )(VPPAST3(ADV epeyce ) (VPPAST3 uzaktaydı ) )))
```

okul NP bizim GENITIVE1PL köye DAT epeyce ADV uzaktaydı VPPAST3

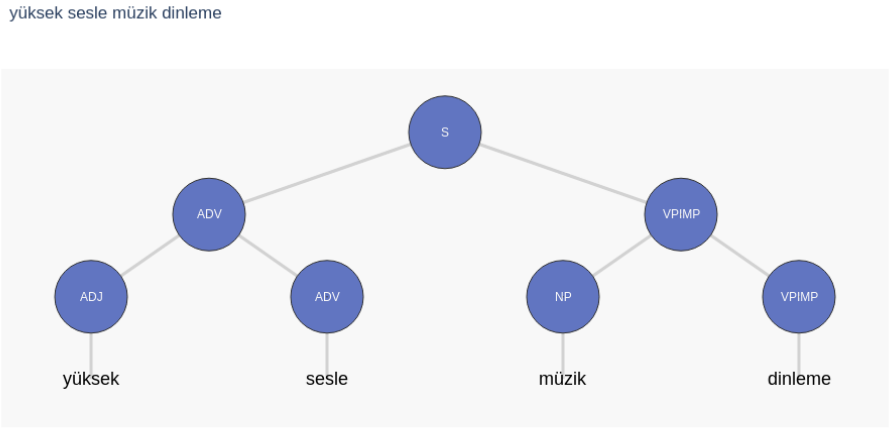
okul bizim köye epeyce uzaktaydı



- Yüksek sesle müzik dinleme.

```
Tokens : ['yüksek', 'sesle', 'müzik', 'dinleme']
POS Tags : [['ADJ'], ['ADV'], ['NP'], ['VPIMP']]
Sentence is grammatically correct.
(S(ADV(ADJ yüksek ) (ADV sesle ) )(VPIMP(NP müzik ) (VPIMP dinleme ) )) Score : 2476.52
```

```
######### CKY CHART #########
-------  -------  ------  --------------
yüksek   sesle    müzik   dinleme
['ADJ']  ['ADV']  []      ['S', 'VPIMP']
[]       ['ADV']  []      ['S', 'VPIMP']
[]       []       ['NP']  ['S', 'VPIMP']
[]       []       []      ['VPIMP']
-------  -------  ------  --------------
##### BEST SENTENCE STRUCTURE #####
(S(ADV(ADJ yüksek ) (ADV sesle ) )(VPIMP(NP müzik ) (VPIMP dinleme ) ))
```



Grammatically False Cases

- Ben arkadaşıma hediye aldın.

```
Tokens : ['ben', 'arkadaşıma', 'hediye', 'aldın']
POS Tags : [['PRO1'], ['DAT'], ['NP'], ['VPPAST2']]
Sentence is not grammatically correct...

######### CKY CHART #########
--------  ----------  ------  -----------------
ben       arkadaşıma  hediye  aldın
['PRO1']  []          []      []
[]        ['DAT']     []      ['VPPAST2']
[]        []          ['NP']  ['S', 'VPPAST2']
[]        []          []      ['VPPAST2']
--------  ----------  ------  -----------------
```

- Tarihi bir romanlar okudum.

```
Tokens : ['tarihi', 'bir', 'romanlar', 'okudum']
POS Tags : [['ADJ'], ['SG'], ['NP3PL'], ['VPPAST1']]
Sentence is not grammatically correct...

######### CKY CHART #########
-------  ------  ---------  -----------
tarihi   bir     romanlar   okudum
['ADJ']  []      []         []
[]       ['SG']  []         []
[]       []      ['NP3PL']  []
[]       []      []         ['VPPAST1']
-------  ------  ---------  -----------
```

- Dün babama yardım edeceğim.

```
Tokens : ['dün', 'babama', 'yardım', 'edeceğim']
POS Tags : [['PASTNOUN'], ['DAT'], ['NP'], ['NP1', 'PREQ', 'VPFUT1', 'ADJ']]
Sentence is not grammatically correct...

######### CKY CHART #########
------------  -------  ------  --------------------------------
```

```
dün           babama   yardım  edeceğim
['PASTNOUN']  []       []      []
[]            ['DAT']  []      ['VPFUT1']
[]            []       ['NP']  ['S', 'VPFUT1']
[]            []       []      ['NP1', 'PREQ', 'VPFUT1', 'ADJ']
------------  -------  ------  --------------------------------
```

- Ben okul gittim.

```
Tokens : ['ben', 'okul', 'gittim']
POS Tags : [['PRO1'], ['NP'], ['VPPAST1']]
Sentence is grammatically correct.
(S(PRO1 ben ) (VPPAST1(NP okul ) (VPPAST1 gittim ) )) Score : 164.29

######### CKY CHART #########
--------  ------  -----------------
ben       okul    gittim
['PRO1']  []      ['S']
[]        ['NP']  ['S', 'VPPAST1']
[]        []      ['VPPAST1']
--------  ------  -----------------
##### BEST SENTENCE STRUCTURE #####
(S(PRO1 ben ) (VPPAST1(NP okul ) (VPPAST1 gittim ) ))
```

- Ben kitap okundu.

```
Tokens : ['ben', 'kitap', 'okundu']
POS Tags : [['PRO1'], ['NP'], ['VPPAST3']]
Sentence is not grammatically correct...

######### CKY CHART #########
--------  ------  -----------------
ben       kitap   okundu
['PRO1']  []      []
[]        ['NP']  ['S', 'VPPAST3']
[]        []      ['VPPAST3']
--------  ------  -----------------
```

- Ben okulda gittim.

```
Tokens : ['ben', 'okulda', 'gittim']
POS Tags : [['PRO1'], ['LOC'], ['VPPAST1']]
Sentence is not grammatically correct...

######### CKY CHART #########
--------  --------------  -----------
ben       okulda          gittim
['PRO1']  ['LOCCLAUSE1']  []
[]        ['LOC']         []
[]        []              ['VPPAST1']
--------  --------------  -----------
```

4 Conclusion

The CKY parser for the Turkish language is created with dynamic CNF grammar rules powered by Morphological Procedure. For a limited test cases, lexicon was built manually but the parser is capable of handling unknown words since its uses ZEYREK morphological analyzer. A best sentence structure selection method has been created with probabilities of non-terminals. For the grammatically correct test cases, the parser works pretty well. For the grammatically wrong cases, the parser fails sometimes a more wide agreement procedure should be defined.

References

John Cocke and Jacob T Schwartz. 1970. Programming languages and their compilers. preliminary notes. courant inst, of math. *Sciences, New York University*.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2nd Edition)*. Prentice-Hall, Inc., USA.

Appendix

A Handcrafted Grammar Rules

```
S -> PRO1 VPPAST1
S -> PRO2 VPPAST2
S -> PRO3 VPPAST3
S -> PROPL1 VPPAST1PL
S -> PROPL2 VPPAST2PL
S -> PROPL3 VPPAST3PL
S -> NP VPPAST1
S -> NP VPPAST2
S -> NP VPPAST3
S -> NPPL VPPAST1
S -> NPPL VPPAST2
S -> NPPL VPPAST3
S -> PASTNOUN VPPAST1
S -> PASTNOUN VPPAST2
S -> PASTNOUN VPPAST3
S -> NP PPPASTCLAUSE1
S -> NP PPPASTCLAUSE2
S -> NP PPPASTCLAUSE3
S -> NPPL PPPASTCLAUSE1
S -> NPPL PPPASTCLAUSE2
S -> NPPL PPPASTCLAUSE3
S -> NP3PL VPPAST3
##
S -> PRO1 VPPRE1
S -> PRO2 VPPRE2
S -> PRO3 VPPRE3
S -> PROPL1 VPPRE1PL
S -> PROPL2 VPPRE2PL
S -> PROPL3 VPPRE3PL
S -> NP VPPRE1
S -> NP VPPRE2
S -> NP VPPRE3
S -> NPPL VPPRE1
S -> NPPL VPPRE2
S -> NPPL VPPRE3
S -> PRENOUN VPPRE1
S -> PRENOUN VPPRE2
S -> PRENOUN VPPRE3
S -> NP PPPRECLAUSE1
S -> NP PPPRECLAUSE2
S -> NP PPPRECLAUSE3
S -> NPPL PPPRECLAUSE1
S -> NPPL PPPRECLAUSE2
S -> NPPL PPPRECLAUSE3
S -> NP3PL VPPRE3
##
S -> PRO1 VPFUT1
S -> PRO2 VPFUT2
S -> PRO3 VPFUT3
S -> PROPL1 VPFUT1PL
S -> PROPL2 VPFUT2PL
S -> PROPL3 VPFUT3PL
S -> NP VPFUT1
S -> NP VPFUT2
S -> NP VPFUT3
S -> NPPL VPFUT1
S -> NPPL VPFUT2
S -> NPPL VPFUT3
S -> FUTNOUN VPFUT1
S -> FUTNOUN VPFUT2
S -> FUTNOUN VPFUT3
S -> NP PPFUTCLAUSE1
S -> NP PPFUTCLAUSE2
S -> NP PPFUTCLAUSE3
S -> NPPL PPFUTCLAUSE1
S -> NPPL PPFUTCLAUSE2
S -> NPPL PPFUTCLAUSE3
S -> NP3PL VPFUT3
##
S -> GENITIVE1 NP1
S -> GENITIVE2 NP2
S -> GENITIVE3 NP3
S -> GENITIVECLAUSE VPPAST3
S -> GENITIVECLAUSE VPFUT3
S -> GENITIVECLAUSE VPPRE3
##
S -> LOCCLAUSE VPPAST1
S -> LOCCLAUSE VPPAST2
S -> LOCCLAUSE VPPAST3
S -> LOCCLAUSE VPPAST1PL
S -> LOCCLAUSE VPPAST2PL
S -> LOCCLAUSE VPPAST3PL
##
S -> NPACC VPPAST1
S -> NPACC VPPAST1PL
S -> NPACC VPPAST2
S -> NPACC VPPAST2PL
S -> NPACC VPPAST3
S -> NPACC VPPAST3PL
S -> NPACC VPPRE1
S -> NPACC VPPRE1PL
S -> NPACC VPPRE2
S -> NPACC VPPRE2PL
S -> NPACC VPPRE3
S -> NPACC VPPRE3PL
S -> NPACC VPFUT1
S -> NPACC VPFUT1PL
S -> NPACC VPFUT2
S -> NPACC VPFUT2PL
S -> NPACC VPFUT3
S -> NPACC VPFUT3PL
##
S -> NP VPIMP
S -> ADV VPIMP
##
S -> NP Q
S -> DATCLAUSE Q
###########
VPPAST1 -> DAT VPPAST1
VPPAST2 -> DAT VPPAST2
VPPAST3 -> DAT VPPAST3
VPPAST1PL -> DAT VPPAST1PL
VPPAST2PL -> DAT VPPAST2PL
VPPAST3PL -> DAT VPPAST3PL
VPPAST1 -> ACC VPPAST1
VPPAST2 -> ACC VPPAST2
VPPAST3 -> ACC VPPAST3
VPPAST1PL -> ACC VPPAST1PL
VPPAST2PL -> ACC VPPAST2PL
VPPAST3PL -> ACC VPPAST3PL
#
VPPRE1 -> DAT VPPRE1
VPPRE2 -> DAT VPPRE2
VPPRE3 -> DAT VPPRE3
VPFUT1 -> DAT VPFUT1
VPFUT2 -> DAT VPFUT2
VPFUT3 -> DAT VPFUT3
###########
VPPAST1 -> NP VPPAST1
VPPAST2 -> NP VPPAST2
VPPAST3 -> NP VPPAST3
VPPAST1PL -> NP VPPAST1PL
VPPAST2PL -> NP VPPAST2PL
VPPAST3PL -> NP VPPAST3PL
VPPRE1 -> NP VPPRE1
VPPRE2 -> NP VPPRE2
VPPRE3 -> NP VPPRE3
VPFUT1 -> NP VPFUT1
VPFUT2 -> NP VPFUT2
VPFUT3 -> NP VPFUT3
##
VPPAST1 -> ADV VPPAST1
VPPAST2 -> ADV VPPAST2
VPPAST3 -> ADV VPPAST3
VPPRE1 -> ADV VPPRE
VPPRE1 -> ADV VPPRE1
VPPRE2 -> ADV VPPRE2
VPPRE3 -> ADV VPPRE3
VPFUT1 -> ADV VPFUT1
VPFUT2 -> ADV VPFUT2
VPFUT3 -> ADV VPFUT3
##
VPPRE1 -> ADV ADJCLAUSE
##
VPIMP -> NP VPIMP
VPIMP -> ADV VPIMP
##
GENITIVECLAUSE -> GENITIVE1 NP1
GENITIVECLAUSE -> GENITIVE2 NP2
GENITIVECLAUSE -> GENITIVE3 NP3
```

GENITIVE1 -> DET GENITIVE1
GENITIVE2 -> DET GENITIVE2
GENITIVE3 -> DET GENITIVE3
##
PPPASTCLAUSE1 -> POSTP VPPAST1
PPPASTCLAUSE2 -> POSTP VPPAST2
PPPASTCLAUSE3 -> POSTP VPPAST3
PPPRECLAUSE1 -> POSTP VPPRE1
PPPRECLAUSE2 -> POSTP VPPRE2
PPPRECLAUSE3 -> POSTP VPPRE3
PPFUTCLAUSE1 -> POSTP VPFUT1
PPFUTCLAUSE2 -> POSTP VPFUT2
PPFUTCLAUSE3 -> POSTP VPFUT3
##
VPPAST1 -> PASTNOUN VPPAST1
VPPAST1 -> PASTNOUN VPPAST2
VPPAST1 -> PASTNOUN VPPAST3
VPPRE1 -> PRENOUN VPPRE1
VPPRE2 -> PRENOUN VPPRE2
VPPRE3 -> PRENOUN VPPRE3
VPFUT1 -> FUTNOUN VPFUT1
VPFUT1 -> FUTNOUN VPFUT2
VPFUT1 -> FUTNOUN VPFUT3
##
NP -> NUM NP
NP1 -> NUM NP1
NP2 -> NUM NP2
NP3 -> NUM NP3
NP -> NP NP
NP -> NP PP
NP -> SG NP
NP -> NP NPABL
NPPL -> ADJ NPPL
NP -> ADJ NP
NP -> NP POSTP
ADV -> ADV ADV
ADV -> ADJ ADV
ADJ -> ADJ ADJ
ADJCLAUSE -> ADJ VPPAST1
ADJCLAUSE -> ADJ VPPAST2
ADJCLAUSE -> ADJ VPPAST3
ADJCLAUSE -> ADJ VPPRE
ADJCLAUSE -> ADJ VPPRE1
ADJCLAUSE -> ADJ VPPRE2
ADJCLAUSE -> ADJ VPPRE3
ADJCLAUSE -> ADJ VPFUT1
ADJCLAUSE -> ADJ VPFUT2
ADJCLAUSE -> ADJ VPFUT3
##
DATCLAUSE -> ADJ DAT
DATCLAUSE -> DET DATCLAUSE
DATCLAUSE -> ADJ DATCLAUSE
##
NPACC -> ADJ NPACC
NPACC -> DET NPACC
##
Q -> PREQ Q
VPPAST1 -> QP VPPAST1
VPPAST1PL -> QP VPPAST1PL
VPPAST2 -> QP VPPAST2
VPPAST2PL -> QP VPPAST2PL
VPPAST3 -> QP VPPAST3
VPPAST3PL -> QP VPPAST3PL
QP -> Q NP
Q -> DATCLAUSE Q
##
LOCCLAUSE -> GENITIVE3 LOC
LOCCLAUSE1 -> PRO1 LOC
LOCCLAUSE2 -> PRO2 LOC
LOCCLAUSE3 -> PRO3 LOC
##
DAT -> GENITIVE1 DAT
DAT -> GENITIVE2 DAT
DAT -> GENITIVE3 DAT
DAT -> GENITIVE1PL DAT
DAT -> GENITIVE2PL DAT
DAT -> GENITIVE3PL DAT
###
###LEXICON FOR TEST CORPUS
SG -> tek
FUTNOUN -> yarın
PRENOUN -> şimdi
POSTP -> ile
Q -> mı
Q -> mi
#
PASTNOUN -> dün
SG -> bir
NP -> hediye
VPPAST1 -> aldım
ADJ -> tarihi
NPACC -> romanları
ADV -> keyifle
VPPRE1 -> okuyorum
PRO1 -> ben
NP -> akşam
NP -> yemeği
POSTP -> için
DAT -> anneme
NP -> yardım
VPPAST1 -> ettim
NP3PL -> destanlar
NP -> milli
NP -> kültürümüzü
POSTP -> ve
NP -> tarihimizi
VPPRE3 -> anlatır
NP -> yaz
VPIMP -> yaz
NPABL -> meyvelerinden
NP -> karpuz
ADV -> bence
ADJ -> en
ADJ -> güzel
VPPRE -> meyvedir
DET -> bu
ADJ -> akşamki
DAT -> toplantıya
PREQ -> katılacak
Q -> mısınız
GENITIVE3 -> ağacın
LOC -> altında
DET -> her
ADJ -> her
NP -> gece
ACC -> mehtabı
VPPAST1PL -> izlerdik
PROPL2 -> siz
DAT -> buraya
NP -> son
ADJ -> son
Q -> ne
NP -> zaman
VPPAST2PL -> geldiniz
NP -> okul
GENITIVE1PL -> bizim
DAT -> köye
ADV -> epeyce
VPPAST3 -> uzaktaydı
ADJ -> yüksek
ADV -> sesle
NP -> müzik
VPIMP -> dinleme

B Moprhological Analyse Procedure

```
def word_parse(self, token):
    parses = self.analyzer.analyze(token)[0]
    possible = []
    for parse in parses:
        pos = parse.pos
        suff = parse.morphemes[-1]
        time = self.get_time(parse.morphemes)
        if pos == "Verb":
            if time != None:
                nonterminal = "VP"
                if "Past" in time: nonterminal += "PAST"
                elif "Present" in time: nonterminal += "PRE"
                elif "Fut" in time: nonterminal += "FUT"

                if suff == "A1pl": nonterminal += "1PL"
                elif suff == "A1sg": nonterminal += "1"
                elif suff == "A2pl": nonterminal += "2PL"
                elif suff == "A2sg": nonterminal += "2"
                elif suff == "A3pl": nonterminal += "3PL"
                elif suff == "A3sg": nonterminal += "3"

                possible.append(nonterminal)
            elif "Imp" in parse.morphemes:
                possible.append("VPIMP")

            else: continue
        elif suff == "Gen":
            nonterminal = "GENITIVE"
            if "1pl" in parse.morphemes[-2]: nonterminal += "1PL"
            elif "1sg" in parse.morphemes[-2]: nonterminal += "1"
            elif "2pl" in parse.morphemes[-2]: nonterminal += "2PL"
            elif "2sg" in parse.morphemes[-2]: nonterminal += "2"
            elif "3pl" in parse.morphemes[-2]: nonterminal += "3PL"
            elif "3sg" in parse.morphemes[-2]: nonterminal += "3"
            possible.append(nonterminal)

        elif pos == "Noun":
            nonterminal = "NP"

            if "P1pl" in suff: nonterminal += "1PL"
            elif "P1sg" in suff: nonterminal += "1"
            elif "P2pl" in suff: nonterminal += "2PL"
            elif "P2sg" in suff: nonterminal += "2"
            elif "P3pl" in suff: nonterminal += "3PL"
            elif "P3sg" in suff: nonterminal += "3"
            elif "A3pl" in suff: nonterminal += "3PL"
            elif suff == "Dat": nonterminal = "DAT"
            elif suff == "Loc": nonterminal = "LOC"
            elif suff == "Abl": nonterminal = "ABL"
            elif suff == "Acc": nonterminal += "ACC"

            possible.append(nonterminal)
        elif pos == "Pron":

            nonterminal = "PRO"
            if "1pl" in suff: nonterminal += "1PL"
            elif "1sg" in suff: nonterminal += "1"
            elif "2pl" in suff: nonterminal += "2PL"
            elif "2sg" in suff: nonterminal += "2"
            elif "3pl" in suff: nonterminal += "3PL"
            elif "3sg" in suff: nonterminal += "3"

            possible.append(nonterminal)
            return [nonterminal]

        elif token.endswith("le"):
            possible.append("ADV")

        elif "ADV" in pos.upper():
            possible.append("ADV")

        elif "ADJ" in pos.upper():
            if "Verb" in parse.morphemes: possible.append("PREQ")

            possible.append("ADJ")

        elif "Ques" == pos:
            possible.append("Q")

        else: possible.append(pos.upper())

    return possible
```