UNIVERSITÉ LIBRE DE BRUXELLES
Faculté des Sciences
Département d'Informatique

Natural Language Processing
Assignment 2

Chatri Alaaedine

Année académique 2016 - 2017

Contents

1 Introduction

In this assignment, we have to use the Dependency Grammar framework to describe and formalize the syntactic structure of sentences. Dependency grammars are quite important in contemporary speech and language processing systems.The syntactic structure of a sentence is described solely in terms of the words (or lemmas) in a sentence and an associated set of directed binary grammatical relations that hold among the words. [1] The purpose of this assignment is the implementation of the shift-reduce algorithm and the construction of an oracle to guide the algorithm in assigning the correct dependency relations.
There are 4 parts in this project:

1. Part 1: Annotation of toy sentences with dependency relations
2. Part 2: Implementation of a transition-based dependency parser
3. Part 3: Evaluation
4. Part 4: Report

2 Part1

For this part, we have 8 sentences and we have to annotate them with universal part-of-speech tags and universal dependency relations. For all the sentences we also have to Provide the annotations in the CoNNL-U format and add them as a separate text file (named: annotation.txt). Example with the sentence "I gave an apple to the teacher" in the CONNL-U-format:

ID	FORM	LEMMA	UPOSTAG	XPOSTAG	FEATS	HEAD	DEPREL	DEPS	MISC
1	I	I	PRON	_	_	2	nsubj	_	_
2	gave	give	VERB	_	_	0	root	_	_
3	an	an	DET	_	_	4	det	_	_
4	apple	apple	NOUN	_	_	2	obj	_	_
5	to	to	ADP	_	_	7	case	_	_
6	the	the	ET	_	_	7	det	_	_
7	teacher	teacher	NOUN	_	_	2	obl	_	_

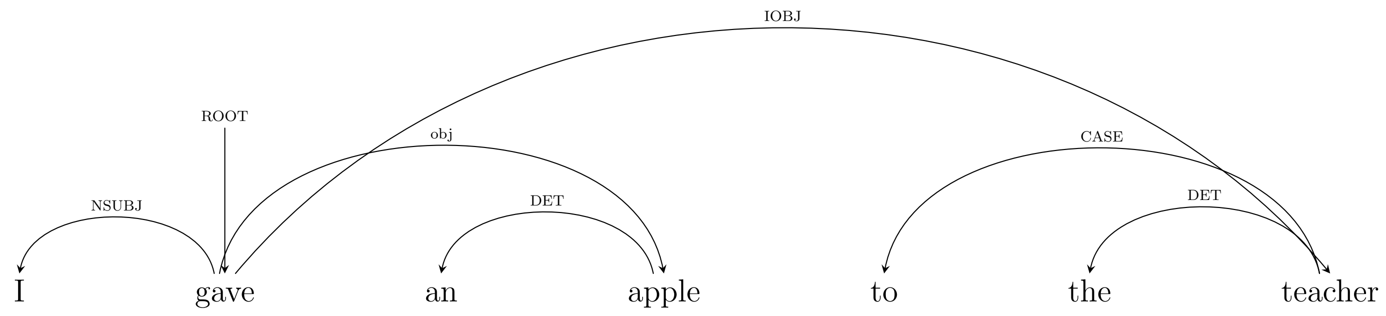
Table 1: CONNL-U-Format for the first sentence

For this assignment, we only need to specify values for the following fields of the CoNNL-U format:

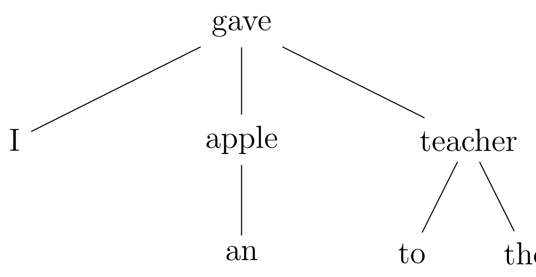
- 1.ID: Word index, integer starting at 1 for each new sentence.
- 2.FORM: Word form or punctuation symbol.
- 3.LEMMA: Lemma or stem of word form.
- 4.UPOSTAG: Universal part-of-speech tag.
- 7.HEAD: Head of the current word, which is either a value of ID or zero (0).
- 8.DEPREL: Universal dependency relation to the HEAD (root iff HEAD = 0) or a defined language-specific subtype of one.

Other fields are unspecfied and indicated with an underscore in the CoNNL-U format.

2.1 A dependency diagram



2.2 A dependency tree



I gave an apple to the teacher

2.2.1 Questions

1. Are there ambiguous sentences with multiple possible parses?If yes, what form does the ambiguity take?

   Yes they are. For example the sentence ”Mary missed her train to work”, we can have 2 different parses:

   - First one

ID	FORM	LEMMA	UPOSTAG	HEAD	DEPREL
1	Mary	Mary	PROPN	2	nsubj
2	missed	miss	VERB	0	root
3	her	her	DET	4	det
4	train	train	NOUN	2	obj
5	to	to	ADP	6	case
6	work	work	VERB	4	nmod

Table 2: First parse

   - Second one

ID	FORM	LEMMA	UPOSTAG	HEAD	DEPREL
1	Mary	Mary	PROPN	2	DEP
2	missed	miss	VERB	6	DEP
3	her	her	DET	4	DEP
4	train	train	NOUN	2	DEP
5	to	to	ADP	6	DEP
6	work	work	VERB	0	DEP

Table 3: Second parse

   As we can see on this two tables, we can have 2 configurations for the head so we have an ambiguous sentence. One possible justification is the choice of grammar to define the root of the sentence.

2. Which sentence containsnon-projective dependency relations? Which relationsexactlyand why are they non-projective?

   An arc from a head to a dependent is said to be projective if there is a path from the head to every word that lies between the head and the dependent in the sentence. A dependency tree is then said to be projective if all the arcs that make it up are projective. A dependency tree is projective if it can be drawn with no crossing edges. [1]
   In the sentence "I saw the doctor this morning who is treating me", the arc between doctor and treating is crossing by the arc between saw and morning. So this sentence is non-projective.

2.3 Discuss Part 1 - Annotation issues

For the annotation part, the most frequent problem was to define the root. Then to fill the DEPREL column because according to our choice of grammar the word had a different relation in the sentence.

3 Part 2

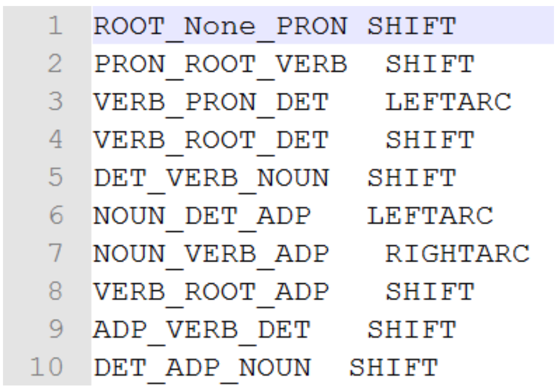
We have to write a transition-based dependency parser in Python(3.x) using the following guidelines:

- Use the shift-reduce algorithm
- Only consider unlabeled dependencies and hence, only 3 possible actions (leftArc, rightArc, shift)for the algorithm
- Our parser should take as INPUT:
  1. An input.txt with sentences in the CoNNL-U format and only the first 4 columns from the annotations file
  2. A text file that contains for each sentence a trace of the configuration states that the shift-reduce parser went through while parsing the sentence.
  3. Manually construct a rule-based oracle

3.1 Discuss Part 2 - Design choices and oracle construction

First of all, we have to take sentences 1 to 4 as starting point to choose and define feature templates which our parser will use to generate features from the configurations while executing the shift-reduce algorithm.
My feattemp.txt, I use the UPOSTAG and the file is defined like this :

- s1_s2_b1 action where:
  - s1 = first item in the stack
  - s2 = second item in the stack
  - b1 = first item in the buffer
  - action = SHIFT, LEFT or RIGHT



```
ROOT_None_PRON SHIFT
PRON_ROOT_VERB  SHIFT
VERB_PRON_DET   LEFTARC
VERB_ROOT_DET   SHIFT
DET_VERB_NOUN  SHIFT
NOUN_DET_ADP   LEFTARC
NOUN_VERB_ADP   RIGHTARC
VERB_ROOT_ADP   SHIFT
ADP_VERB_DET   SHIFT
DET_ADP_NOUN  SHIFT
```

Figure 1: Sample of feattemp.txt

3.1.1 Implementation

- Retrieve the features of the file and put them in a table
- Parse the input file
- For each sentence:
  - Fill the buffer with the line of the input file (example: buffer [0] == [1, I, I, PRON])
  - Fill the temporary buffer in the same way as the buffer
- Call the method which represents the algorithm seen in class

  ```
  def dependency_parse(self,myStack,myBuffer,features,tmpBuffer,conftableParam):
  ```

  - Define the first 2 elements of the stack (ROOT and None)
  - As long as the buffer is not empty or the stack has more than one element:
  - Take s1, s2 and b1 and make a concatenation (as explained above)
  - If the concatenation is in the table of features then
  - The action is retrieved from the table
  - The apply method is called with the action, the stack, and the buffer

    ```
    def apply(self, action, myStack, bufferParam, bufferTempParam):
    ```

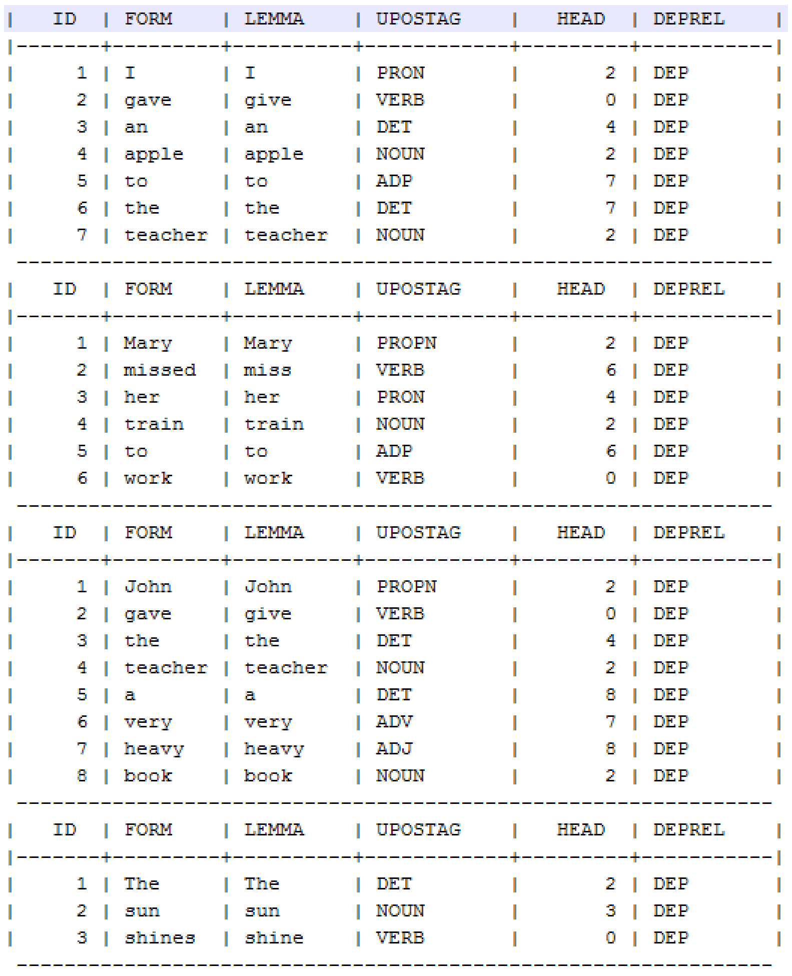
  - Apply returns a relation according to the action and updates the stack and the buffer
  - If the action is Shift
    - SHIFT: the stack receives the first element of the buffer, which is then removed from the buffer.
    - LEFT: s2 is removed from the stack. The head of s2 receives the ID of s1. The relation is written 's2<s1'
    - RIGHT: s1 is removed from the stack. The head of s1 receives the ID of s2 unless s2 is ROOT then the head is equal to 0. The relation is written 's2>s1'.
  - The steps are incremented
- Generate the 2 output files: output.txt and conftable.txt

4 Part 3

4.1 Discuss Part 3 - Evaluation

For didactic purposes and in the toy-world setting of this assignment, we have to evaluate our parser against the same four sentences used in the development stage (part 1). Our manually annotated dependency relations from Part 1 will function as the “gold standard” for the evaluation.

- Run our parser on sentences 1 to 4 listed in part 1 of the assignment

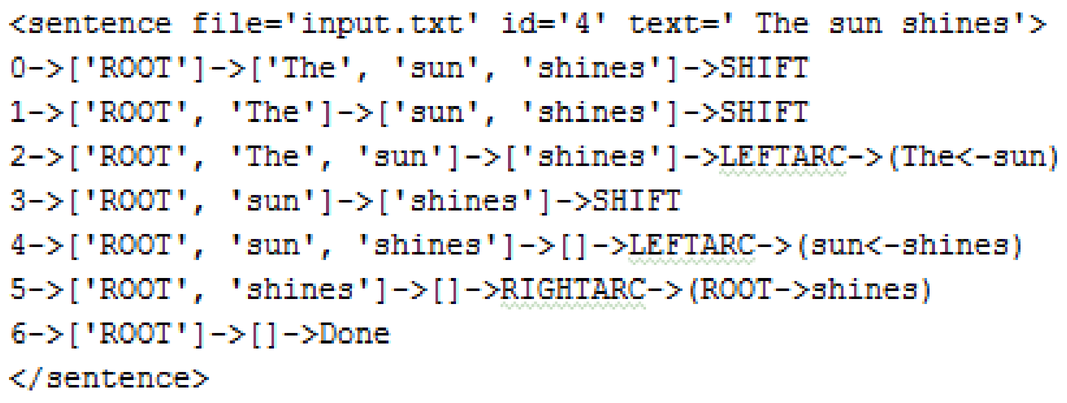


```
|   ID   | FORM     | LEMMA     | UPOSTAG      |    HEAD  | DEPREL     |
|--------+----------+-----------+-------------+----------+------------|
|      1 | I        | I         | PRON        |        2 | DEP        |
|      2 | gave     | give      | VERB        |        0 | DEP        |
|      3 | an       | an        | DET         |        4 | DEP        |
|      4 | apple    | apple     | NOUN        |        2 | DEP        |
|      5 | to       | to        | ADP         |        7 | DEP        |
|      6 | the      | the       | DET         |        7 | DEP        |
|      7 | teacher  | teacher   | NOUN        |        2 | DEP        |
 ----------------------------------------------------------------------
|   ID   | FORM     | LEMMA     | UPOSTAG      |    HEAD  | DEPREL     |
|--------+----------+-----------+-------------+----------+------------|
|      1 | Mary     | Mary      | PROPN       |        2 | DEP        |
|      2 | missed   | miss      | VERB        |        6 | DEP        |
|      3 | her      | her       | PRON        |        4 | DEP        |
|      4 | train    | train     | NOUN        |        2 | DEP        |
|      5 | to       | to        | ADP         |        6 | DEP        |
|      6 | work     | work      | VERB        |        0 | DEP        |
 ----------------------------------------------------------------------
|   ID   | FORM     | LEMMA     | UPOSTAG      |    HEAD  | DEPREL     |
|--------+----------+-----------+-------------+----------+------------|
|      1 | John     | John      | PROPN       |        2 | DEP        |
|      2 | gave     | give      | VERB        |        0 | DEP        |
|      3 | the      | the       | DET         |        4 | DEP        |
|      4 | teacher  | teacher   | NOUN        |        2 | DEP        |
|      5 | a        | a         | DET         |        8 | DEP        |
|      6 | very     | very      | ADV         |        7 | DEP        |
|      7 | heavy    | heavy     | ADJ         |        8 | DEP        |
|      8 | book     | book      | NOUN        |        2 | DEP        |
 ----------------------------------------------------------------------
|   ID   | FORM     | LEMMA     | UPOSTAG      |    HEAD  | DEPREL     |
|--------+----------+-----------+-------------+----------+------------|
|      1 | The      | The       | DET         |        2 | DEP        |
|      2 | sun      | sun       | NOUN        |        3 | DEP        |
|      3 | shines   | shine     | VERB        |        0 | DEP        |
 ----------------------------------------------------------------------
```

Figure 2: Results in output.txt

- Give the unlabeled attachment accuracy across all 4 sentences and discuss the errors (if any) and why your think the parser made them Like we saw in part 1, table 1 is from my annotation.txt made manually and 1 is from my output.txt made by the parser. We can see that there are 2 different heads. So we can say that we have 22 head on 24 so the accuracy is 91.67%. The problem is coming from a conflict in features table, for one concatenation NOUN_VERB_ADP in the first sentence we got RIGHT as action and in the second we have SHIFT.
- The precision and recall for the head-dependency relations across all 4 sentences.

5 Sample of the result in Conftable.txt



```
<sentence file='input.txt' id='4' text=' The sun shines'>
0->['ROOT']->['The', 'sun', 'shines']->SHIFT
1->['ROOT', 'The']->['sun', 'shines']->SHIFT
2->['ROOT', 'The', 'sun']->['shines']->LEFTARC->(The<-sun)
3->['ROOT', 'sun']->['shines']->SHIFT
4->['ROOT', 'sun', 'shines']->[]->LEFTARC->(sun<-shines)
5->['ROOT', 'shines']->[]->RIGHTARC->(ROOT->shines)
6->['ROOT']->[]->Done
</sentence>
```

Figure 3: Sample of contable.txt

References

[1] James H. Martin Dan Jurafsky. *Speech and Language Processing (3rd ed. draft).* Springer Science and Business Media, 2017.