

CS5012 - Practical 2: Grammar Engineering

Student ID: 180029539

Introduction:

The purpose of the practical was to engineer a grammar to accept a small subset of English language. So, I created grammar rules which accepts the small subset of sentences that are correctly formed. Also, it rejects the sentences that are syntactically wrong.

Sample Sentences:

Accepted Sentences:

- 1) Stan carries Ollie
- 2) Stan sneezes
- 3) Stan wears a grey suit
- 4) Ollie sneezed in the house
- 5) Stan and Ollie carry a piano
- 6) Ollie gave Stan a piano with wheels
- 7) Ollie carried a large heavy piano up the stairs
- 8) Stan always walks up the stairs
- 9) Ollie thinks Stan wears a suit in the house
- 10) when Ollie sneezes the piano rolls down the stairs
- 11) when does Stan walk up the stairs

Rejected Sentences:

- 1) carry Stan Ollie
- 2) carry carry
- 3) Stan carry Ollie
- 4) Stan carries Ollie Ollie
- 5) Stan sneeze
- 6) when do Ollie sneezes
- 7) Ollie thinks the piano

Grammar Creation:

First, I created the grammar rules to accept each sentence individually. So, for the first sentence 'Stan carries Ollie' is a basic sentence with a subject verb and object. So, the basic rules 'S->NP VP' and 'VP -> V ARG' was sufficient to accept the grammar. Then, for the second sentence, its structure is similar to first sentence but without object. So, adding 'VP -> V' was enough. Then, the third sentence has same structure but the argument was a noun phrase with an adjective phrase embedded. So, I added the respective rules to accept the sentence. In, the fourth sentence, the presence of a determiner required me to modify the rules to include the usage of determiner in the sentence. Next for fifth sentence, the usage of conjunction made it unique. So, I had to add NP->NP Conj NP to support the change. In the sixth sentence, we use a di-transitive verb 'gives'. We will see the implementation in the sub-categorisation, where we

restrict the sentences with 2 objects after the verb. But, now we just create a rule that accepts verb followed by two objects (VP → V ARG ARG). For the seventh sentence, it has an adjective phrase embedded in the sentence. So, replaced the rule from third sentence which takes only one adjective with rules to accept consecutive adjectives. Next, the eighth sentence uses an adverb, so simple introduction of the rule 'VP → Adv VP' parses the sentence. Next, ninth sentence uses 'thinks' which needs to be followed by a sentence clause. In the subcategorization we enforce this rule. In the tenth sentence, we have a conjunction followed by two sentences. So, I created a new rule 'S → Conj S S' to support these type of sentences. For the last sentence, I had to create new rule to accept question formats. The new rule is 'S → Adv Aux NP VP'. Finally, I added the lexical part of the grammar to accept all the lexicons. After this all the positive sentences are parsing but the negative sentences are also getting accepted. We will apply number agreement and subcategorization to the rules to avoid these from happening.

Number Agreement:

First, I went through all the verbs and nouns in lexicons and added the [num=sg] or [num=pl] based on their format. Words like 'gave' I did not classify as they can be used for both the formats. Then, I came to the noun-phrase rules and appended all the noun rules with [num=?x] on both sides as the nouns number format doesn't change if there is only a single noun phrase on the right hand side. But, for the conjunction rule (NP → NP Conj NP), we have two NP's on RHS. So, the parameter becomes [num=pl]. Then, I applied it to VP rules by appending [num=?x] on both sides. As we know number agreement is an agreement between subject and verb. I converted all the Starting rules such that NP and VP have the same number format. In the question rule, i converted it as 'S → Adv Aux[num=?x] NP[num=?x] VP[num=pl]' as here Auxiliary verb should be in agreement with the NP.

Subcategorisation:

For subcategorisation, I went through each verb and applied subcategorisation and I checked if positive sentences with that verb are accepted and negative are rejected. First, I took the verb 'carries'. It can have two forms. It is either followed by an object or an object followed by a prepositional phrase. So, I changed the rule into two rules.

V[num = sg, SUBCAT=[HEAD=np, TAIL=[HEAD=pp, TAIL=nil]]] → 'carries'

V[num = sg, SUBCAT=[HEAD=np, TAIL=nil]] → 'carries'.

The first you can see accepts a following NP and the NP that is followed by PP.

Then, I changed the 'VP → VP ARG' into 'VP[num=?x, SUBCAT=?rest] → VP[num=?x, SUBCAT=[HEAD=?arg, TAIL=?rest]] ARG[CAT=?arg]'.

Next, I verified the sentences with 'carries'. 'Stan carries Ollie' is accepted. Then, I added my own sentence 'Stan carries Ollie down the stairs' to test the other form. It is also accepted. Next, I tested the negative forms like 'Stan carries Ollie Ollie' which is rejected.

Then, I applied similar rules to 'carry', 'carried', 'wears'. Next, for 'rolls', there was an extra scenario where its followed by a PP. So, I added 'V[num = sg,SUBCAT=[HEAD=pp, TAIL=nil]] -> 'rolls'.

Then, I modified rules for verb 'gave'. This is a di-transitive verb. So, I modified the rules for 'gave' to below format. As it requires two objects to be followed by VP. The advantage of using this way is Verb and both the following NP will remain in the same level.

V[SUBCAT=[HEAD=npnp, TAIL=[HEAD=pp, Tail=nil]]] -> 'gave'

V[SUBCAT=[HEAD=npnp, TAIL=nil]] -> 'gave'

VP[num=?x, SUBCAT=?args] -> VP[num=?x, SUBCAT=[HEAD=npnp, TAIL=?args]]

ARG[CAT=np] ARG[CAT=np]

Then, I verified the sentences with 'gave'. The positive sentences like 'Ollie gave Stan a piano with wheels' were accepted. The negative sentences like 'Stan gave piano' is rejected.

Finally for intransitive verbs like 'sneeze'. I added rules in the below format.

V[num = pl, SUBCAT = [HEAD=pp, TAIL=nil]] -> 'sneeze'

V[num = pl, SUBCAT = nil] -> 'sneeze'

VP[num=?x, SUBCAT=?args] -> V[num=?x, SUBCAT=?args]

This allows only prepositions being followed and rejects all other sentences. Positive sentences like 'Stan sneezes' are accepted and negative sentences like 'Ollie sneezes piano' are rejected.

Next, for verbs like 'thinks', I have used below format as they are to be followed by sentences.

V[num = sg,SUBCAT=[HEAD=s, TAIL=nil]] -> 'thinks'

Then, I tested the rules. The sentences like 'Ollie thinks Stan wears a suit in the house' are accepted. Then, sentences like 'Ollie thinks the piano' are rejected.

Extensions:

I added support for below extra sentences.

Positive sentences:

Stan likes walking up the stairs

Stan sneezes in the house

Stan sneezes in the house and when Ollie sneezes the piano rolls down the stairs

Stan carries Ollie down the stairs

Negative sentences:

Ollie thinks the piano

Ollie sneezes piano

Stan gave piano

