Quiz 2 – CFG and Parsing Solution

- Date: Thu, July 9, 2020.
- Duration: 30 mins

Questions

- [8pts] Q1 - First sets
- [4pts] Q2 - ambiguous grammar
- [8pts] Q3 - disambiguating grammar

[8pts] Q1. First Set

Find the first sets for all the non terminals for the CFG provided below

$$S \rightarrow\ T\ T\ Y$$

$$T \rightarrow\ a \mid X\ T\ b \mid \epsilon$$

$$X \rightarrow\ d \mid f \mid\ Y\ S$$

$$Y \rightarrow\ a \mid b \mid c$$

1. [2pts] First set of Y

$$\{a, b, c\}$$

2. [2pts] First set of X

$$\{d, f\} \cup First(Y) = \{d, f, a, b, c\}$$

3. [2pts] First set of T

$$\{a\} \cup First(X) \cup \{\epsilon\} = \{d, f, a, b, c, \epsilon\}$$

4. [2pts] First set of S

$$First(T) \cup First(Y) = \{d, f, a, b, c, \epsilon\}$$

Note: There is the y because `T` can give an epsilon in both cases yielding a `Y` as the first set

[4pts] Q2. Ambiguous Grammar Give an example

Let's say you want to make a grammar to represent addition and exponents of natural numbers. It is represented by the following CFG:

$$S \rightarrow S + S | S\hat{\ } S | n$$

where n is any natural number

Provide a proof that the above grammar is ambiguous.

Soln:

```
S -> S + S -> 1 + S -> 1 + S + S ----.
                                      \
                                       |---> 1 + 2 + S -> 1 + 2 + S -> 1 + 2 + 3
                                      /
S -> S + S -> S + S + S -> 1 + S + S-.
```

[8pts] Q3. Disambiguate grammar

1. Disambiguate the grammar above CFG(reproduced here) and enforce addition's (i.e. +) lower precedence over exponentiation (i.e. ^)

$$S \rightarrow S + S \mid S\hat{\ } S \mid n$$

where n is any natural number

Soln:

$$S \rightarrow T + S \mid T$$

$$T \rightarrow N\hat{\ } T \mid N$$

$$N \rightarrow n$$

where n is a natual number Note: You want the higher prcedence to be deeper in the tree. This will ensure that it gets bound before the lower precedence ones.

2. Using the disambiguated CFG to find the left most derivation the following expression:

$$1 + 2\hat{\ } 3$$

Soln:

```
S -> T + S -> N + S -> 1 + S -> 1 + T
  -> 1 + N ^ T -> 1 + 2 ^ T -> 1 + 2 ^ N -> 1 + 2 ^ 3
```