Model Answer for Class Test 1

Dipayan Mukherjee : 11CS30045

1. *Consider the following grammar with 2 missing productions :*

$S \rightarrow aS \mid Prod\ 1$

$A \rightarrow Prod2 \mid \epsilon$

$X \rightarrow cS \mid \epsilon$

$Y \rightarrow dS \mid \epsilon$

$Z \rightarrow eS$

Terminal =$\{a, b, c, d, e\}$*, Non Terminal = {S,A,X,Y,Z}. S is the start symbol.*

Fortunately we have the First and Follow sets for this grammar:

	First	Follow
S	$\{a, b, c, d, e\}$	$\{\$\} \cup Follow\ (X) \cup Follow(Y) \cup Follow(Z)$
A	$\{c, d, e, \epsilon\}$	$\{b\}$
X	$\{c, \epsilon\}$	$\{First(Y) - \epsilon\} \cup First\ (Z)$
Y	$\{d, \epsilon\}$	$First\ (Z)$
Z	$\{e\}$	$Follow\ (A)$

Reconstruct the grammar by filling in the missing two productions (Prod1) and (Prod 2).

Answer: -

- *Since Follow (Z) = Follow (A), hence in the Prod2 of A, Z is the last non terminal in the production, because only then Follow (Z) can be Follow (A).*
- *Follow (Y) = First (Z) implies that whenever non terminal Y appears Z follows it. That is in any production they will be of type* $P \rightarrow \cdots YZ \mid \ldots$
- *Similar to previous argument from Follow (X) we can deduce that whenever X appears YZ follows it as only then Follow (X) =* $\{First\ (Y) - \epsilon\ \}$ $\cup$ *First (Z).*
- *Since there are no production starting with* $\{b\}$ *hence any production where A occurs it will be* $P \rightarrow \cdots Ab \mid \ldots$
- *Now since First (A) has all First of X, Y, Z only so we can write the production of* $A \rightarrow XYZ \mid \epsilon$ *since we showed that X must be followed by YZ and Y by Z.*
- *Since by the given production of S only* $\{a\}$ *can be its First but it also includes the First of A and Follow (A). Hence the production of S is*

$$S \rightarrow aS \mid Ab$$

Hence Prod 1: - Ab *and Prod 2:* XYZ

2. *Consider the following regular definition :*

$$sign \rightarrow [+\ -]$$
$$digit \rightarrow [0-9]$$
$$dot \rightarrow \backslash.$$
$$\exp \rightarrow [eE] \qquad \text{// exponential}$$

a) *Give the regular definition for signed/ unsigned floating point numbers. In exponential power, the number should be an integer.*

b) *Draw DFA for accepting floating point numbers using the above regular definition.*

Answer:

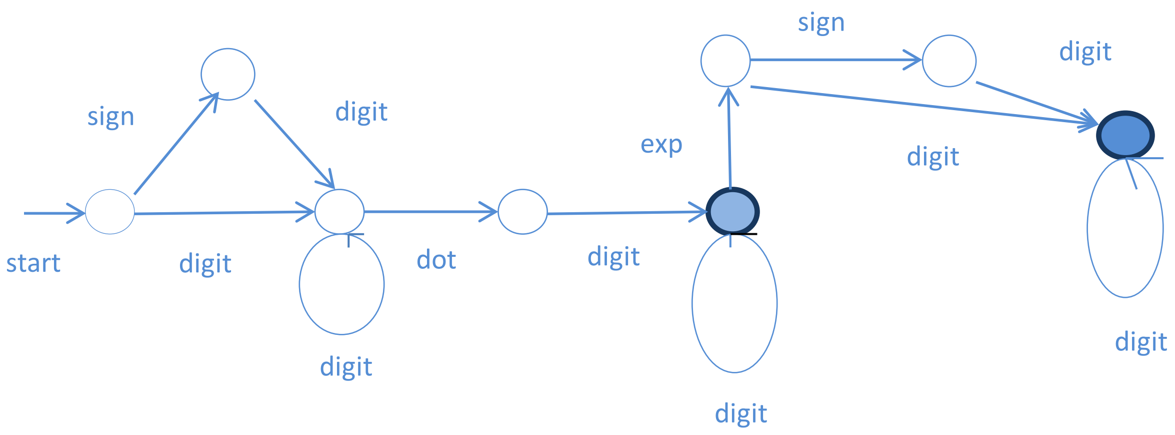
a) Let us define a few more regular expressions using those above to shorten our definition:

$$sgn \rightarrow sign \mid \epsilon$$
$$digits \rightarrow digit^{+}$$

Now using these along with the given expression we give the regular definition as follows:

$$float \rightarrow sgn\ digits\ dot\ digits\ (\{\exp\ \ sgn\ digits\} \mid \epsilon)$$

b)



This is the DFA for accepting the floating point number

3. *a) Let's assume that in a programming language specification P++, the last symbol of any program written in P++ is 'ENDP' (like in C, where last symbol is a closing parenthesis} of main ()). In a panic mode error recovery implementation for a predictive parser for P++, Prof. X includes only 'ENDP' symbol in the synchronization set. What kind of problem/ inconvenience do you expect from his compiler? (Answer in 2-3 sentences)*

   Answer:

   - *This will stop the program from being parsed whenever any error is encountered and will skip till the end when ENDP is encountered so no recovery is actually done, as it simply stops parsing.*
   - *It will report only one error during a parsing so if there are multiple errors we will have to run the parsing multiple times, causing inconvenience.*

   b) True or False. Justify with logic (in 2-3 sentences) (no marks without logic)

   i) "If all the addresses in the final object program contains absolute addresses, then an absolute loader is sufficient to load the program in main memory"

   Answer:
   Since in absolute loader the programmer has to specify the address to the assembler where the program is to be loaded, then the loader loads the program at that point if the memory location is free. Since as per the statement we already have the absolute addresses of the code, hence an absolute loader is sufficient to load the entire program into the main memory.

   - ***True***

   ii) "Dynamic loader can work without dynamic linker (as long as a static linker is available)"

   Answer:
   Yes there can be Dynamic loader without Dynamic Linking. As with static linking the executable has an address/ offset table generated during the compile time but the actual code/ data aren't loaded to the memory at the start of the process. Hence there can be dynamic loader without dynamic linking.

   - ***True***