SRI RAMKRISHNA SARADA VIDYAMAHAPITHA

FIRST and FOLLOW in Compiler Design

Why we use FIRST?

Suppose the compiler would have come to know in advance. In that case, the “first character of the string produced when a production rule is applied” and comparing it to the current character or token in the input string it sees, and it can wisely decide which production rule to apply.

Let’s take the grammar

```
S -> cAd
A -> bc|a
And the input string is “cad”.
```

In this example, our input string is “cad”. We are applying S->cAd for character ‘c’. Next character in the input string is ‘a’. It would have disregarded the production rule A->bc (for the reason that ‘b’ is the first character of the string produced by this production rule, not ‘a’), and directly use the production rule A->a (because ‘a’ is the first character of the string produced by this production rule, and is same as the current character of the input string which is also ‘a’).

Hence it is validated that if the compiler/parser knows about first character of the string that can be obtained by applying a production rule, then it can wisely apply the correct production rule to get the correct syntax tree for the given input string.

Why we use FOLLOW?

The parser faces one additional problem. Let us consider below grammar to understand this problem.

```
A -> aBb
B -> c | ε
And suppose the input string is “ab” to parse.
```

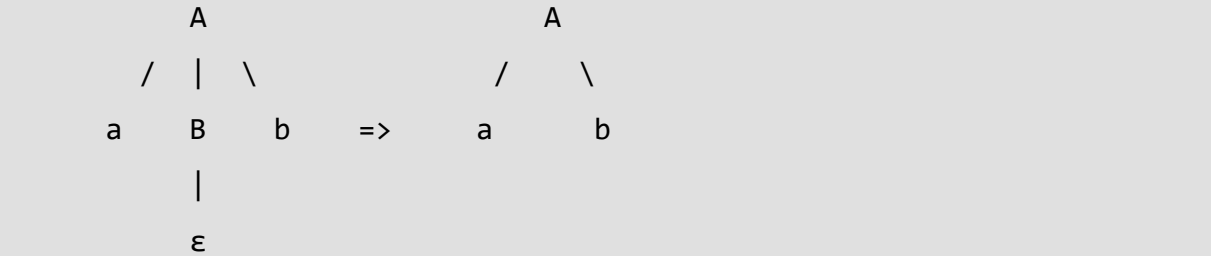
As the first character in the input is a, the parser applies the rule A->aBb.

```
          A
        / |  \
      a   B   b
```

Now the parser checks for the second character of the input string which is b, and the Non-Terminal to derive is B, but the parser can’t get any string derivable from B that contains b as first character.

But the Grammar does contain a production rule B -> ε, if that is applied then B will vanish, and the parser gets the input “ab” , as shown below. But the parser can apply it only when it knows that the character that follows B in the production rule is same as the current character in the input.

In RHS of A -> aBb, b follows Non-Terminal B, i.e. FOLLOW(B) = {b}, and the current input character read is also b. Hence the parser applies this rule. And it is able to get the string “ab” from the given grammar.



So FOLLOW can make a Non-terminal to vanish out if needed to generate the string from the parse tree.

The conclusions is, we need to find FIRST and FOLLOW sets for a given grammar, so that the parser can properly apply the needed rule at the correct position.

PRACTICE PROBLEMS BASED ON CALCULATING FIRST AND FOLLOW

Problem-01:

Calculate the first and follow functions for the given grammar-

$$S \rightarrow aBDh$$

$$B \rightarrow cC$$

$$C \rightarrow bC / \in$$

$$D \rightarrow EF$$

$$E \rightarrow g / \in$$

$$F \rightarrow f / \in$$

Solution-

The first and follow functions are as follows-

First Functions-

- First(S) = { a }
- First(B) = { c }
- First(C) = { b , ∈ }
- First(D) = { First(E) – ∈ } ∪ First(F) = { g , f , ∈ }
- First(E) = { g , ∈ }
- First(F) = { f , ∈ }

Follow Functions-

- Follow(S) = { $ }
- Follow(B) = { First(D) – ∈ } ∪ First(h) = { g , f , h }
- Follow(C) = Follow(B) = { g , f , h }

- Follow(D) = First(h) = { h }
- Follow(E) = { First(F) – ∈ } ∪ Follow(D) = { f , h }
- Follow(F) = Follow(D) = { h }

Assignment

Problem-01

Calculate the first and follow functions for the given grammar-

$$S \rightarrow A$$
$$A \rightarrow aB / Ad$$
$$B \rightarrow b$$
$$C \rightarrow g$$

Problem-02

Calculate the first and follow functions for the given grammar-

$$S \rightarrow AaAb / BbBa$$
$$A \rightarrow \in$$
$$B \rightarrow \in$$

Problem-03

Calculate the first and follow functions for the given grammar-

$$E \rightarrow E + T / T$$
$$T \rightarrow T \times F / F$$
$$F \rightarrow (E) / id$$