Class test 1

Write your name and secondary ID (of the form abc123) on the top left corner of all your answer sheets.

Question 1

Consider the following grammar:

$$\begin{array}{rcl} S & \to & a\,S\,b\,S \\ S & \to & c\,S\,d\,S \\ S & \to & \varepsilon \end{array}$$

Show how the LL machine can accept the input

$$a\,b\,c\,d$$

by giving an accepting machine run. You may use FIRST and FOLLOW, but you do not need to compute them formally. [40%]

Solution

This question was intentionally easy, as the grammar is isomorphic to Dyck(2).

$$\begin{array}{rl} & \langle S\;,\;a\,b\,c\,d\rangle \\ \xrightarrow{\text{predict}} & \langle a\,S\,b\,S\;,\;a\,b\,c\,d\rangle \\ \xrightarrow{\text{match}} & \langle S\,b\,S\;,\;b\,c\,d\rangle \\ \xrightarrow{\text{predict}} & \langle b\,S\;,\;b\,c\,d\rangle \\ \xrightarrow{\text{match}} & \langle S\;,\;c\,d\rangle \\ \xrightarrow{\text{predict}} & \langle c\,S\,d\,S\;,\;c\,d\rangle \\ \xrightarrow{\text{match}} & \langle S\,d\,S\;,\;d\rangle \\ \xrightarrow{\text{predict}} & \langle d\,S\;,\;d\rangle \\ \xrightarrow{\text{match}} & \langle S\;,\;\varepsilon\rangle \\ \xrightarrow{\text{predict}} & \langle \varepsilon\;,\;\varepsilon\rangle \end{array}$$

Question 2

Suppose we have a rule in our grammar like this:

$$E \to E\,\alpha$$

Explain why this grammar rule causes a problem for the LL(1) machine. Hint: suppose we also have $E \to b$, and think about b. [40%]

Solution

There is a FIRST/FIRST conflict (due to left recursion). Suppose there is another rule

$$E \to b$$

Now consider a machine state

$$\langle E\,\sigma\ ,\, b\,w \rangle$$

By the definition of FIRST, we have $b \in$ FIRST(b), therefore also $b \in$ FIRST(E) and $b \in$ FIRST$(E\alpha)$. Thus the LL(1) machine has a choice of two different predict steps. This makes the machine nondeterministic. The aim of the LL(1) construction is to get a deterministic machine, so that goal has not been achieved in this case.

It could also be the case that there is a FIRST/FOLLOW conflict, but that requires E to be nullable, which is not stated in the question.

Example of a wrong explanation: E calls itself so it loops forever; therefore recursion is bad! It is also not correct to claim that the recursion causes a problem for computing FIRST.

Question 3

Consider the following grammar:

$$\begin{array}{rcl} S & \to & A \\ A & \to & b \\ D & \to & b \end{array}$$

Discuss whether this grammar presents a problem for LR(0) parsing, or not. Your answer should refer to the LR and/or LR(0) machines.

[20%]

Solution

There is potential nondeterminism here, but items are able to resolve it. Given the input b, the nondeterministic LR machine may shift b and then reduce to either A or D. This is a reduce/reduce conflict. If the machine reduces to D, it gets stuck, as D is not reachable from S.

However, this situation does not cause problems for LR(0) parsing, as items can guide reduce steps. When we have an item

$$[A \to \bullet b]$$

and shift b, we get the item

$$[A \to b\bullet]$$

This item tells the LR machine to reduce to A and not to D.

It was not required to compute the LR(0) automaton formally, but for full marks something reasonable had to be said about items.