CS 241 – Week 7 Tutorial Solutions

Bottom-up Parsing

Spring 2018

1 Machine construction solutions

1.1 Non-LR(0) grammars

1. The second state of the machine will contain:

$$S \to a \bullet Sb$$
$$S \to a\bullet$$
$$S \to \bullet aSb$$
$$S \to \bullet a$$

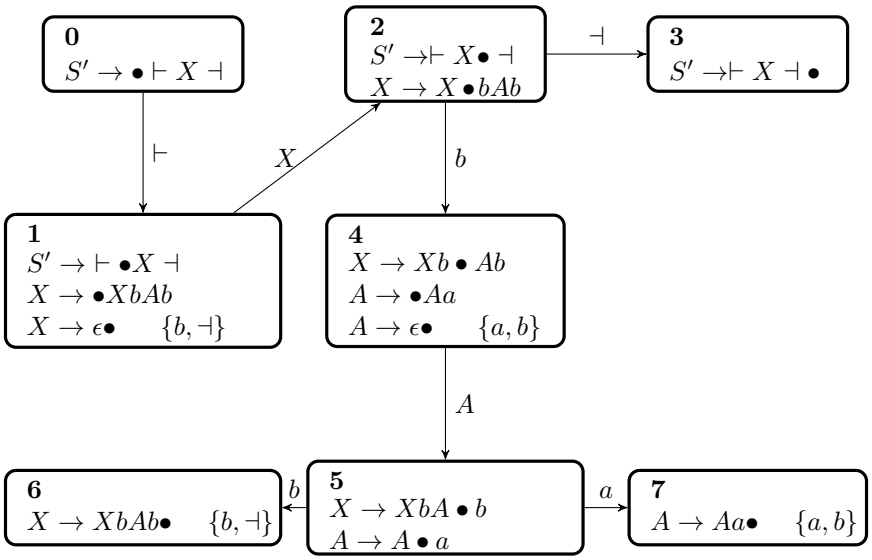
   This constitutes a shift-reduce conflict.

2. There will be a state of the machine containing:

$$X \to e\bullet$$
$$Y \to e\bullet$$

   This constitutes a reduce-reduce conflict.

1.2 SLR(1) machine construction



2 Parsing solutions

1. We can see that there would be three shift-reduce conflicts in our machine (states 1, 3, 5) if weren't for the attached follow sets. Thus this grammar is not LR(0).
2. Algorithm:
   - We begin with 0 on the state stack.
   - We use the top of the state stack and the first letter of input to determine the next action.
   - If it is a shift, move the top of the input to the symbol stack and push the new state to the state stack.
   - If it is a reduce, remove a number of items equal to the length of the right hand side of the rule we reduce by, then shift the nonterminal.
   - If the entry does not exist, reject.
   - If we shift $\dashv$, accept. If we want, we can do a final reduction for S', but this is unnecessary, hence why the machine has no actions for state 6.

Action	State Stack	Symbol Stack	Remaining Input
initialize	0		$\vdash pqab \dashv$
shift $\vdash$, 1	0 1	$\vdash$	$pqab \dashv$
shift p, 5	0 1 5	$\vdash p$	$qab \dashv$
reduce $X \to$	0 1 5 9	$\vdash pX$	$qab \dashv$
reduce $X \to pX$	0 1 3	$\vdash X$	$qab \dashv$
shift q, 7	0 1 3 7	$\vdash Xq$	$ab \dashv$
reduce $Y \to q$	0 1 3 2	$\vdash XY$	$ab \dashv$
reduce $S \to XY$	0 1 8	$\vdash S$	$ab \dashv$
shift a, 4	0 1 8 4	$\vdash Sa$	$b \dashv$
shift b, 10	0 1 8 4 10	$\vdash Sab$	$\dashv$
reduce $S \to Sab$	0 1 8	$\vdash S$	$\dashv$
shift $\dashv$, 6	0 1 8 6	$\vdash S \dashv$	

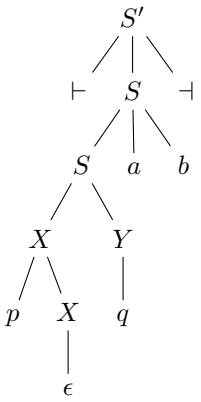
3. The reversed rightmost derivation can be obtained by reading the rules used in reductions from top to bottom, and then adding the S' rule at the end. Here is the reversed derivation in CFG-R format:

X
X p X
Y q
S X Y
S S a b
S' ⊢ S ⊣

Furthermore, we can obtain a rightmost derivation by reading the symbol stack concatenated with the remaining input from bottom to top:

$$\begin{aligned} S' &\Rightarrow \vdash S \dashv \\ &\Rightarrow \vdash Sab \dashv \\ &\Rightarrow \vdash XYab \dashv \\ &\Rightarrow \vdash Xqab \dashv \\ &\Rightarrow \vdash pXqab \dashv \\ &\Rightarrow \vdash pqab \dashv \end{aligned}$$

We can obtain a parse tree from this derivation by looking at what rule was used to expand each non-terminal symbol in the derivation.


S'
$\vdash$
S
$\dashv$
S
a
b
X
Y
p
X
q
ϵ