CS 480: Compiler Design
Problem Set 3

Due: 07/01/2017

Instructions:
I leave plenty of space on each page for your computation. If you need more sheet, please attach your work right behind the corresponding problem. If your answer is incorrect but you show the computation process, then partial credits will be given. It is prefered that you type up your solution (you can use the DocHub extension on Google Drive to edit pdf files and insert images without using LaTeX) and upload your solution onto blackboard for grading.

First Name:

Last Name:

Score: **/100**

Problem 1 SLR(1): 4*5 =20pts

Here is a CFG for strings of balanced parenthesis:

$$S \to P$$
$$P \to (P)P$$
$$P \to \epsilon$$

Hence, this grammar can generate the following strings $(), (()())(), (((()))()(()()))$ and the terminal symbols are $(,), \$$ where \$ is the end of input marker.
(a) Construct the LR(0) configurating sets for this grammar. Show your result. There are six total configuration sets. For the production $P \to \epsilon$, there is only one LR(0) item, which is $P \to \cdot$.

(b) Compute the Follow sets for each nonterminal

(c) Construct SLR(1) table, based on your result in (a) an (b).

(d) Identify one entry in the parsing table that would be a shift/reduce conflict in an LR(0) parser (if there is none, explain why)

(e) Identify one entry in the parsing table that would be a reduce/reduce conflict in an LR(0) parser (if there is none, explain why)

Problem 2 SLR(1): 2+2+8+8 =20pts

Here is a CFG :

$$A \to aAa$$
$$A \to bAb$$
$$P \to \epsilon$$

The terminal symbols are $a, , \$$ where \$ is the end of input marker.
(a) Describe the language that this grammar defines.

(b) Is this CFG ambiguous? Why?

(c) Construct SLR parse table for this CFG

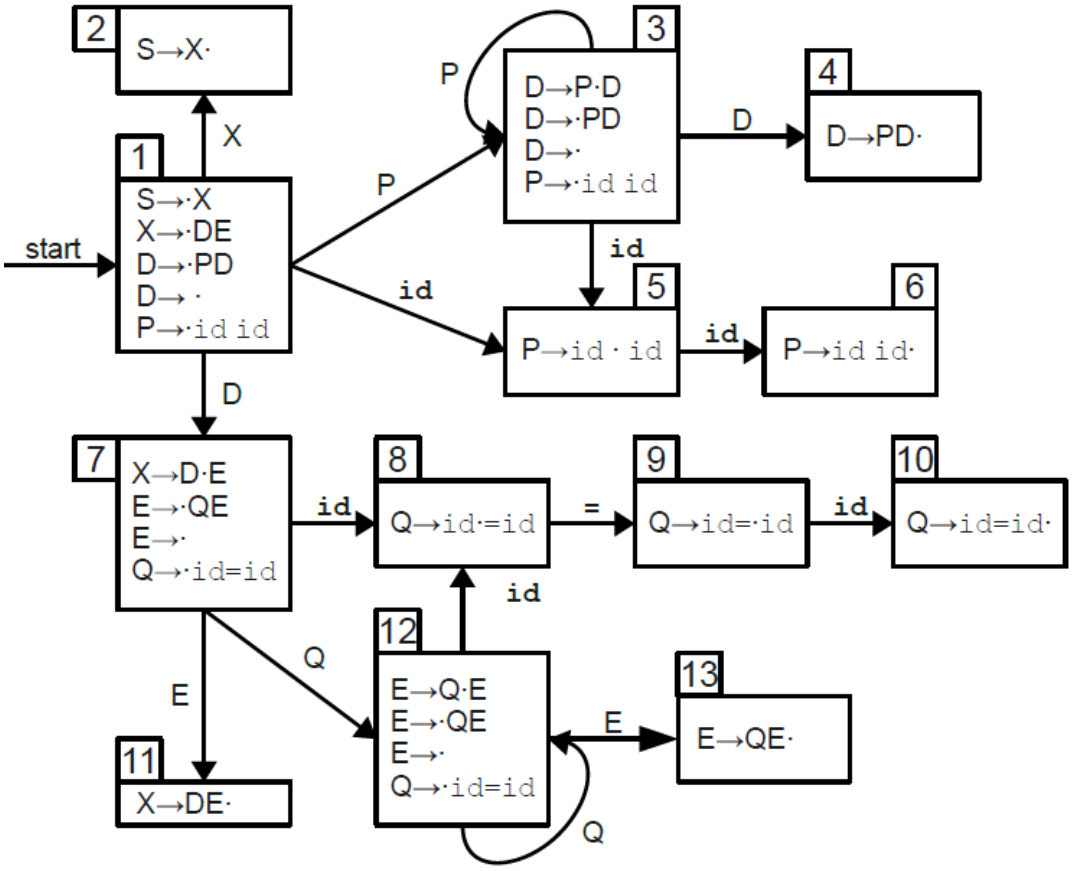
(d) Can the conflicts in this table be eliminated?

Problem 3 LR Parsing: 6*5 = 30pts

We will explore how several different LR parsing algorithms handle or fail to handle a particular grammar:

$$S \to X$$
$$X \to DE$$
$$D \to PD$$
$$D \to \epsilon$$
$$E \to QE$$
$$E \to \epsilon$$
$$P \to id \quad id$$
$$Q \to id = id$$

Below is an LR(0) automaton for this language. Stages are numbered for your convenience.



The terminals in the grammar are $id, =, \$$ where \$ is the end of input symbol.

(a) LR(0): Why is this language not LR(0)) Why is this grammar not SRL(1)?

(b) SLR(1): Is this grammar SLR(1)? Why?

(c-1) LALR(1)-by-SLR(1) algorithm: Create the augmented LALR(1)

(c-2) Compute the Follow sets for each nonterminal in the LALR(1) augmented grammar

(d) Is this grammar LALR(1) ? Why? Please use your result in (c-1),(c-2) to prove (disprove)

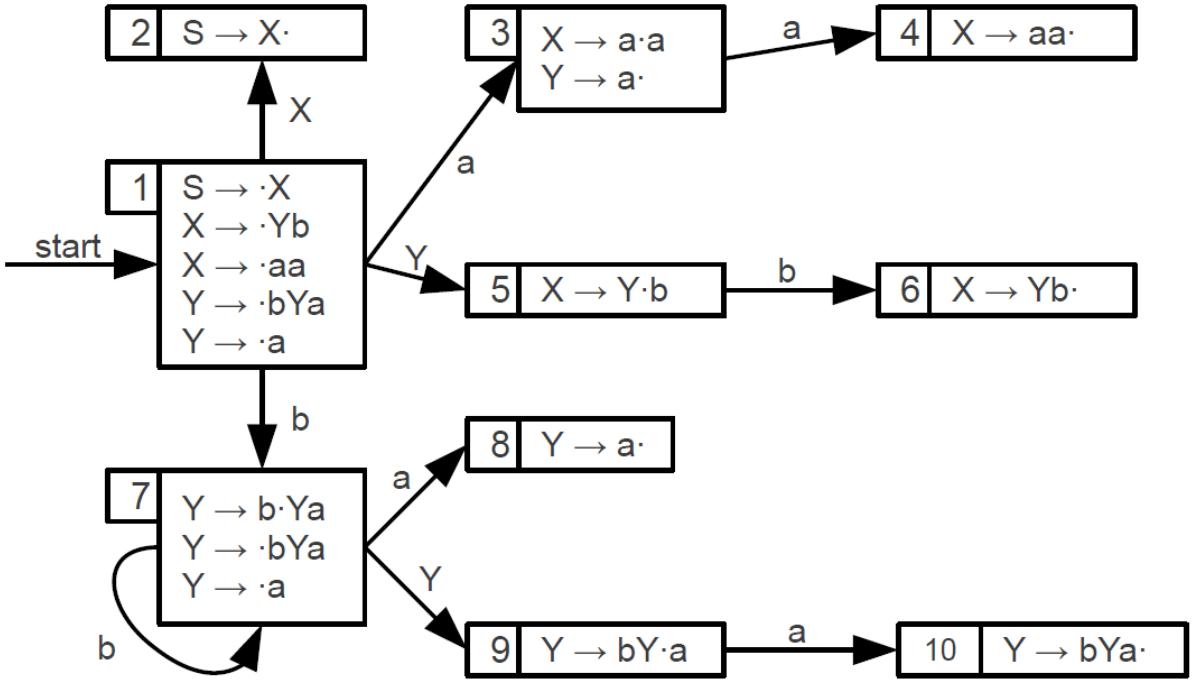
(e) Is this gramma LR(1)? Why?

Problem 4 SLR(1) → LALR Algorithm: 6*5 = 30pts

Here is a grammar that is not SLR(1):

$$S \to X$$
$$X \to YB|aa$$
$$Y \to a|bYA$$

Here is the associated LR(0) automaton for this grammer:



(a) Why is this grammar not SRL(1)?

(b) Using the LR(0) automaton, construct the augmented grammar that youwill use in the algorithm. Show result

(c) Compute the FOLLOW sets for every noterminals in the grammar. Show the result

(d) Using the Follow sets and LR(0) automaton, consctruct the LALR(1) lookaheads for each reduce item in the automaton. Show results. (there are 6 reduced items in the automaton)

(e) Is this grammar LALR(1) ? Why ?

(f) Is this gramma LR(1)? Why?