CS 164 **Fall 2015**

WA4

Nikhil Unni (cs164-es), Section : Monday 3pm

1. (a) Is the following grammar LL(1)? Is it LR(1)? Explain both of your answers.

$$F \rightarrow mv$$

$$F \rightarrow ma$$

It is not LL(1), because the grammar is not left-factored (there will be a first/first conflict on "m"). It is LR(1) because there can be a valid parsing table which just shifts on m, and reduces on either v or a to F on a $ in the input.

(b) Is the following grammar ambiguous? Is it LR(1)? Explain both of your answers.

$$A \rightarrow Bad$$

$$A \rightarrow Cat$$

$$B \rightarrow r$$

$$B \rightarrow r$$

It is not ambiguous, just because there is only 1 parse tree each for both "rat" and "rad", the only two stings in the language. It is **not** LR(1) because there is a reduce/reduce conflict in the parsing table between $B \rightarrow r$ and $C \rightarrow r$. This can be remedied by changing $A \rightarrow Cat$ to $A \rightarrow Bat$, and eliminate the nonterminal C.

2. Consider the following grammar with start symbol S:

$$S \rightarrow Dx$$

$$D \rightarrow pwD|n$$

(a) Complete the DFA skeleton.

I drew it on the back.

(b) Is the grammar LR(1)? LR(2)? LL(1)?

The grammar is LR(1) because we have no conflicts in our DFA. Because the grammar is LR(1) it **has** to be LR(2) as well, (and, generally, LR(k), $k > 0$). The grammar is also LL(1) because it has no left recursion and is completely left factored.

(c) Use your DFA to parse pwpwpwnx.

Stack	Input	Action
	$\triangleright$pwpwpwnx$	shift 7
pwpwpwn	pwpwpwn$\triangleright$x$	reduce $D \rightarrow n$
pwpwpwD	pwpwpwn$\triangleright$x$	reduce $D \rightarrow pwD$
pwpwD	pwpwpwn$\triangleright$x$	reduce $D \rightarrow pwD$
pwD	pwpwpwn$\triangleright$x$	reduce $D \rightarrow pwD$
D	pwpwpwn$\triangleright$x$	shift
Dx	pwpwpwnx$\triangleright$$	reduce $S \rightarrow Dx$
S	pwpwpwnx$\triangleright$$	accept