Flipped Wednesday

What is the difference between an LR(1) and LL(1) parser? Which is more powerful

LL(1) predicts

predict

verify

LR(1) delayed commitment

shift

reduce

strictly more powerful

$LR(*) \leq DCFG$

Left –factor this grammar

A ::= yAAxxA
| yAAxAy
| yAy
| x

$\Rightarrow$

$A ::= yAA'$
$\quad | \; x$

$A' ::= AxxA$
$\quad | \; AxAy$
$\quad | \; y$

$\Rightarrow$

$A ::= yAA'$
$\quad | \; x$

$A' ::= AxA''$
$\quad | \; y$

$A'' ::= xA$
$\quad | \; Ay$

Eliminate left-recursion in the following grammar:

α

L ::= L L E
L ::= G
G ::= G b
~~| a~~ $G ::= a$
E ::= k

$E\ k$

$|\ z$

$$L ::= G L'$$
$$\Rightarrow L' ::= L E L'$$
$$|\ \epsilon$$
$$G ::= G b$$
$$|\ a$$
$$E ::= k$$

L
$|$
G
$|$
a

$$L ::= G L'$$
$$L' ::= L \cancel{E} L' \quad k$$
$$|\ \epsilon$$
$$G ::= a G'$$
$$G' ::= b G'$$
$$|\ \epsilon$$
$$\cancel{E ::= k}$$

What are the FIRST and FOLLOW sets for the following grammar:

A ::= A y B
A ::= z
A ::= ε
B ::= A

C_1:

FIRST(A) = {z, ε, y}
FIRST(B) = {z, ε, y}

FOLLOW(A): eof y

FOLLOW(B): eof y

FOLLOW

FOLLOW(X):

C_1: If X is the start symbol, add eof

Forall Y ::= α X β :

C_2: add FIRST(β)

C_3: If β has length 0: add FOLLOW(Y)

C_4: If β is nullable: add FOLLOW(Y)

Is it possible to generate FIRST sets for a grammar with syntactic ambiguity? Explain your reasoning.

$$A ::= x$$
$$\quad\;\; | \; x$$