Jesse Opitz
Lab 9

Crafting a Compiler
5.5
Original Grammar:
1. DeclList → DeclList ; Decl
2. | Decl
3. Decl → IdList : Type
4. IdList → IdList , id
5. | id
6. Type → ScalarType
7. | array (ScalarTypeList) of Type
8. ScalarType → id
9. | Bound .. Bound
10. Bound → Sign intconstant
11. | id
12. Sign → +
13. | -
14 | λ
15. ScalarTypelist → ScalarTypelist , ScalarType
16. | ScalarType

Answer:
Left recursion is the issue: there are no common prefixes
Left recursion is an issue in the bolded steps

1. DeclList → Decl DeclList'
2. DeclList' → ; DeclList
3. | λ
4. Decl → IdList : Type
5. IdList → id IdList'
6. IdList' → , IdList
| λ
7. Type → ScalarType
8. | array (ScalarTypeList) of Type
9. ScalarType → id
10. | Bound .. Bound
11. Bound → Sign intconstant
12. | id
13. Sign → +
14. | -
15 | λ
16. ScalarTypelist → ScalarType ScalarTypelist'
17. ScalarTypelist' → , ScalarTypelist
18. | λ

ScalarTypelist , ScalarType
16. | ScalarType

6.51? ←Optional?
The bottom-up parsing techniques given in this chapter are more powerful than top-down techniques given in Chapter 5.
Using the alphabet { a, b }, devise a language that is not LL(k) for any k but is LR(k) for some k. What property of LR(k) parsing allows such a grammar to be constructed?
A → a + B
  | a – b
B → a + b

This is not LL(k) because it is left recursive and has common prefixes.
It can be LR(k) because it is not right recursive, which means it has a unique prefix when reading right to left.

Dragon
4.5.3
Give bottom-up parses for the following input strings and grammars:
a)
Ex 4.5.1:
S → 1. 0 S 1 | 2. 0 1
The input 000111 according to the grammar of Ex 4.5.1
S => 1. 0 S 1 => 1. 0 0 S 1 1 => 2. 0 0 0 1 1 1

b)
Ex 4.5.2:
S → 1. S S + | 2. S S * | 3. a

The input aaa*a++ according to the grammar of Ex 4.5.2
Shift Reduce:

a => S => a S => S S => a S S => S S S => S S S * => S S => S S a => S S S => S S S + => S S => S S + => S

4.6.5
Show that the following grammar:
S → A a A b | B b B a
A → ϵ
B → ϵ

Is LL(1) but not SLR(1) [‘Don’t worry about SLR’ – Alan Labouseur]

Parse Table:

	a	b
S	1	2
A		
B		