LR(0) Parsing Tables Example

CS 4447/CS9545 -- Stephen Watt
University of Western Ontario

Example Generating LR(0) Tables

Grammar

Terminals = { **$** , **;** , **id** , **:=** , **+** }

Nonterminals = {S’, S, A, E }

Start Symbol = S’

Productions = {

```
1.    S' → S $
2.    S  → S ; A
3.    S  → A
4.    A  → E
5.    A  → id := E
6.    E  → E + id
7.    E  → id
}
```

The Start State

- Find the closure of items with start symbol S' as LHS.

I0 := closure({ S' → . S $ })

```
    = {
        S' → . S $,
        S  → . S ; A,
        S  → . A,
        A  → . E,
        A  → . id := E,
        E  → . E + id,
        E  → . id
}
```

The First Transitions

- Find the next states by moving the dot past the first symbol in each rule:

```
t1 := { S' → S . $, S → S . ; A }
t2 := { S  → A . }
t3 := { A  → E . , E  → E . + id }
t4 := { A  → id . := E, E  → id . }
```

- Notice that we make a separate set for moving the dot past each symbol. E.g. t1 moves the dot past S.

  In each case now the dot either precedes a terminal or is at the end.
  There cannot be any rules such that $A \rightarrow \alpha\ .\ B\ \beta$, where $B \rightarrow \gamma$ is a production in the grammar.

- Therefore

```
I1 := closure(t1) = t1
I2 := closure(t2) = t2
I3 := closure(t3) = t3
I4 := closure(t4) = t4
```

The First Transitions (contd)

- We use the notation `A →[S] B` to say that we move from state A to state B on symbol S, i.e. when `Goto(A,S) = B`.

- We have

```
I0 →[S] I1
I0 →[A] I2
I0 →[E] I3
I0 →[id] I4
```

Next Transitions

- We now need to determine the sets given by moving the dot past the symbols in the RHS of the productions in each of the new sets `I1-I4`.

- `I1 = {S'→S.$, S→S.;A},` so the only symbol the dot can move past is ";"
- Likewise the only symbol dot can move past in I3 is "+" and in I4 is ":=".

```
Goto(I1,;) = closure({S→ S ;.A})= I5 =
    {S → S;. A, A→.E, A→.id := E, E→.E+id, E→.id }

Goto(I3,+) = closure({E→E+.id}) = I6 = {E → E + . id}

Goto(I4,:=)= closure({A→id:=.E})= I7 = {A→id:=.E, E→.E+ d, E →.id }
```

- We therefore have the following transitions:

```
I1 →[;]  I5
I3 →[+]  I6
I4 →[:=] I7
```

More Transitions

- We must compute GoTo sets for I5, I6 and I7.

  GoTo(I5,A) = closure({ S → S ; A . }) = I8 = { S → S ; A . }

  GoTo(I5,E) = closure({ A → E ., E → E . + id }) = I3

  GoTo(I5,id) = closure({ A → id . := E, E → id . }) = I4

  GoTo(I6,id) = closure({ E → E + id . }) = I9 = { E → E + id . }

  GoTo(I7,E) = closure({ A → id := E ., E → E .+ id }) = I10 = { A → id := E ., E → E . + id }

  GoTo(I7,id) = closure({ E → id . }) = I11 = { E → id . }

- These are the transitions:

  I5 →[A] I8

  I5 →[E] I3

  I5 →[id] I4

  I6 →[id] I9

  I7 →[E] I10

  I7 →[id] I11

- There is only one transition left to compute, GoTo(I10, +)

  Goto(I10,+) = closure({ E → E + . id }) = I6

  I10 →[+] I6

Parsing Automaton

- The parsing automaton has the following states:

```
I0 := { S' → . S $, S  → . S ; A,
        S  → . A, A  → . E, A  → . id := E, E  → . E + id, E  → . id }
I1 := { S' → S . $, S → S . ; A }

I2 := { S → A . }
I5 := { S → S ; .  A, A → . E, A → . id := E, E → . E + id, E → . id }
I8 := { S → S ;  A . }

I3 := { A → E . , E  → E . + id }
I4 := { A → id . := E, E → id . }
I7 := { A → id := . E, E → . E + id, E → . id }
I10:= { A → id := E ., E → E . + id }

I6 := { E → E + . id}

 I9:= { E → E + id . }
I11:= { E → id . }
```

-

Parsing Automaton

- And the following transitions:

```
I0 →[S] I1
I0 →[A] I2
I0 →[E] I3
I0 →[id] I4

I1 →[;] I5

I3 →[+] I6

I4 →[:=] I7

I5 →[A]  I8
I5 →[E]  I3
I5 →[id] I4

I6 →[id]  I9

I7 →[E]  I10
I7 →[id] I11

I10→[10] I6
```

Table Entries

- The states imply the following table entries

```
I0: none
I1:
   S' → S . $
      Action[I1,$] = accept
I2:
   S → A .
      Action[I2,$] = reduce 3
      Action[I2,;] = reduce 3
      Action[I2,id] = reduce 3
      Action[I2,:=] = reduce 3
      Action[I2,+] = reduce 3
I3:
   A → E .
      Action[I3,$] = reduce 4
      Action[I3,;] = reduce 4
      Action[I3,id] = reduce 4
      Action[I3,:=] = reduce 4
      Action[I3,+] = reduce 4
I4:
   E → id .
      Action[I4,$] = reduce 7
      Action[I4,;] = reduce 7
      Action[I4,id] = reduce 7
      Action[I4,:=] = reduce 7
      Action[I4,+] = reduce 7

I5: none

I6: none

I7: none

I8:
  S → S ;  A .
      Action[I8,$] = reduce 2
      Action[I8,;] = reduce 2
      Action[I8,id] = reduce 2
      Action[I8,:=] = reduce 2
      Action[I8,+] = reduce 2

I9:
  E → E + id .
      Action[I9,$] = reduce 6
      Action[I9,;] = reduce 6
      Action[I9,id] = reduce 6
      Action[I9,:=] = reduce 6
      Action[I9,+] = reduce 6

I10:
  A → id := E .
      Action[I10,$] = reduce 5
      Action[I10,;] = reduce 5
      Action[I10,id] = reduce 5
      Action[I10,:=] = reduce 5
      Action[I10,+] = reduce 5

I11:
  E → id .
      Action[I11,$] = reduce 7
      Action[I11,;] = reduce 7
      Action[I11,id] = reduce 7
      Action[I11,:=] = reduce 7
      Action[I11,+] = reduce 7
```

Table Entries

- The transitions imply the following table entries:

```
I0 →[S] I1              Goto[I0,S] = I1
I0 →[A] I2              Goto[I0,A] = I2
I0 →[E] I3              Goto[I0,E] = I3
I0 →[id] I4             Action[I0,id] = shift I4

I1 →[;] I5              Action[I1,;]  = shift I5

I3 →[+] I6              Action[I3,+]  = shift I6

I4 →[:=] I7             Action[I4,:=] = shift I7
I5 →[A]  I8             Goto[I5,A] = I8
I5 →[E]  I3             Goto[I5,E] = I3
I5 →[id] I4             Action[I5,id] =  shift I4

I6 →[id]  I9            Action[I6,id] =  shift I9

I7 →[E]  I10            Goto[I7,E] = I10
I7 →[id] I11            Action[I7,id] = shift I11


I10→[+]  I6             Action[I10,+] = shift I6
```

Filling in the Table

1. S' → S $
2. S → S ; A
3. S → A
4. A → E
5. A → **id** **:=** E
6. E → E **+ id**
7. E → **id**

3 Shift-Reduce conflicts

State	Action					Goto			
	Id	**;**	**+**	**:=**	**$**	S'	S	A	E
0	S I4						I1	I2	I3
1		S I5			acc				
2	R 3	R 3	R 3	R 3	R 3				
3	R 4	R 4	**R 4** **S I6**	R 4	R 4				
4	R 7	R 7	R 7	**R 7** **S I7**	R 7				
5	S I4							I8	I3
6	S I9								
7	S I11								I10
8	R 2	R 2	R 2	R 2	R 2				
9	R 6	R 6	R 6	R 6	R 6				
10	R 5	R 5	R 5 S I6	R 5	R 5				
11	R 7	R 7	R 7	R 7	R 7				