# CC50A - Compiladores
# Pauta Control 2

Universidad de Chile

primavera 2008

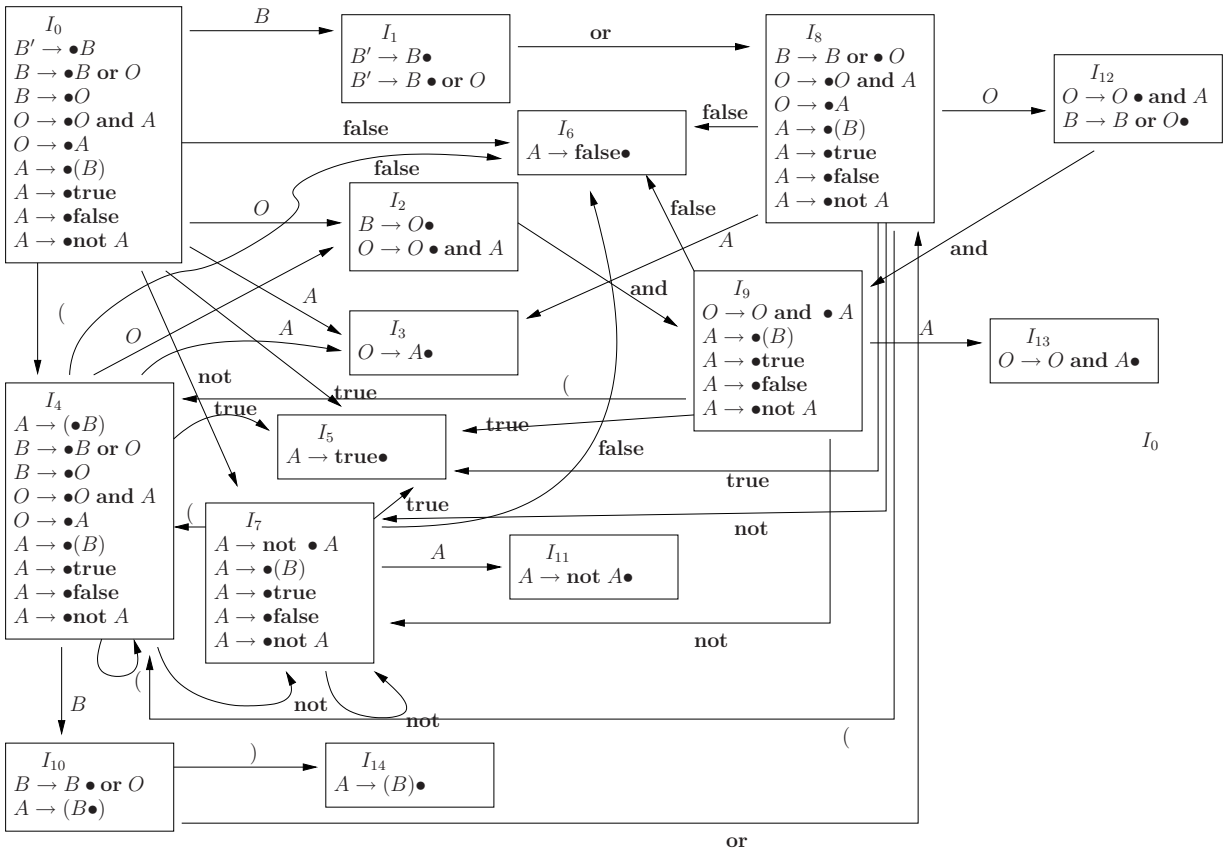
## Pregunta 1

Considere la gramática:

$$B' \to B$$
$$B \to B \textbf{ or } O \mid O$$
$$O \to O \textbf{ and } A \mid A$$
$$A \to \textbf{not } A \mid (\ B\ ) \mid \textbf{true} \mid \textbf{false}$$

Construya la tabla de análisis sintáctico SLR de esta gramática.

**Solución**



(1) $B' \rightarrow B$
(2) $B \rightarrow B$ **or** $O$
(3) $B \rightarrow O$
(4) $O \rightarrow O$ **and** $A$
(5) $B \rightarrow A$
(6) $A \rightarrow$ **not** $A$(7) $A \rightarrow$ ( $B$ )(8) $A \rightarrow$ **true**(9) $A \rightarrow$ **false**

$FOLLOW(B) = \{), \$\}$
$FOLLOW(O) = \{\textbf{or}, ), \$\}$
$FOLLOW(A) = \{\textbf{or}, \textbf{and}, ), \$\}$

| estado | **true** | **false** | **or** | **and** | ( | ) | **not** | $ | $B$ | $O$ | $A$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | s5 | s6 | | | s4 | | s7 | | 1 | 2 | 3 |
| 1 | | | s8 | | | | | acc | | | |
| 2 | | | | s9 | | r3 | | r3 | | | |
| 3 | | | r5 | | | r5 | | r5 | | | |
| 4 | s5 | s6 | | | s4 | | s7 | | 10 | 2 | 3 |
| 5 | | | | | | | | | | | |
| 6 | | | r8 | r8 | | r8 | | r8 | | | |
| 7 | s5 | s6 | | | s4 | | s7 | | | | 11 |
| 8 | s5 | s6 | | | s4 | | s7 | | | 12 | 3 |
| 9 | s5 | s6 | | | s4 | | s7 | | | | 13 |
| 10 | | | s8 | | | s14 | | | | | |
| 11 | | | r6 | r6 | | r6 | | r6 | | | |
| 12 | | | | s9 | | r2 | | r2 | | | |
| 13 | | | r4 | r4 | | r4 | | r4 | | | |
| 14 | | | r7 | r7 | | r7 | | r7 | | | |

# Pregunta 2

Considerando las siguientes clases de Cool:

```
class Root {
   count:Int;
   visited:Bool;
   me:SELF_TYPE;
   child:Root;
};

class First inherits Root {
   mycount:Int;
   something:Object;
   myfirst:First;
};

class Second inherits First {
   myself:SELF_TYPE;
   mycount:Int;
   node:Root;
};
```

y la siguiente modificación a las reglas de tipos de Cool:

$$\frac{\begin{array}{l} T_0' = \begin{cases} \texttt{SELF\_TYPE}_C & \text{if } T_0 = \texttt{SELF\_TYPE} \\ T_0 & \text{otherwise} \end{cases} \\ O, M, C \vdash e_1 : T_1 \\ T_1 \leq T_0' \\ O[T_0'/x], M, C \vdash e_2 : T_2 \end{array}}{O, M, C \vdash \text{let } x : T_0 \leftarrow e_1 \text{ in } e_2 : T_0} \quad \text{[Let-Init]}$$

Determine el tipo de datos que se asocia a las siguientes expresiones si ejecutan dentro de las clases indicadas (si existe un error explique porqué):

1. Clase Root: `count + count`.

   `Int + Int -> Int`

2. Clase Root: `me + count`

   `SELF_TYPE{Root} + Int -> Error`

3. Clase First: `isvoid something`

   `isvoid Object -> Boolean`

4. Clase First: `something <- myself`

   `(Object <- ERROR) -> Error` (error de scope)

5. Clase Second: `let me:SELF_TYPE <- node in mycount:=1`

   `(SELF_TYPE{Second} <- Root in Int) -> SELF_TYPE{Second}`

6. Clase Second: `node <- myself`

   `(Root <- SELF_TYPE{Second}) -> SELF_TYPE{Second}`

# Pregunta 2

Escriba código MIPS para el siguiente fragmento de Cool. El valor inicial de la variable `i` se encuentra en el inicio del stack.

```
while(i < 50) {
   i = f(i);
   i = i * 2 + 9;
}
g(i);
```

**Solución**.

```
        lw $r0 4($sp)
        addi $sp $sp 4
        li $r1 50
comp:   bge $r0 $r1 finish
        sw $fp 0($sp)
        addi $sp $sp -4
        sw $r0 0($sp)
        addi $sp $sp -4
        jal f
        lw $r0 4($sp)
```

```
        addi $sp $sp 4
        add $r0 $r0 $r0
        addi $r0 $r0 9
        b comp
finish: sw $fp 0($sp)
        addi $sp $sp -4
        sw $r0 0($sp)
        add $sp $sp -4
        jal g
```