

Linguaggi e Compilatori

Progetto, parte B

Enrico Santi, mat. 148687

January 2024

Esercizio 1

Parser SLR

Iniziamo enumerando le produzioni ed aumentando la grammatica con un nuovo simbolo iniziale (S') in modo che quest'ultimo presenti una sola produzione:

0. $S' \rightarrow S$
1. $S \rightarrow L = R S$
2. $S \rightarrow \epsilon$
3. $L \rightarrow * L$
4. $L \rightarrow L1$
5. $L1 \rightarrow L1 [R]$
6. $L1 \rightarrow id$
7. $L1 \rightarrow (L)$
8. $R \rightarrow R + R$
9. $R \rightarrow R * R$
10. $R \rightarrow L$
11. $R \rightarrow (R)$

Per costruire il parser SLR dobbiamo poi ricavare la collezione canonica degli stati:¹

¹Nella collezione canonica non si considera lo stato vuoto. Inoltre CL indica la funzione closure.

$I_0 = CL(S \rightarrow S) = \{S \rightarrow S, S \rightarrow L = RS, S \rightarrow \circ, L \rightarrow \circ \cdot L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L), L \rightarrow \circ \cdot L1(R)\}$
$I_1 = Goro(I_0, S) = CL(S \rightarrow S \circ) = \{S \rightarrow S \circ\}$
$I_2 = Goro(I_0, L) = CL(S \rightarrow L \circ = RS) = \{S \rightarrow L \circ = RS\}$
$I_3 = Goro(I_0, *) = CL(L \rightarrow \circ \cdot L) = \{L \rightarrow \circ \cdot L, L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L), L \rightarrow \circ \cdot L1[R]\}$
$I_4 = Goro(I_0, L1) = CL(L \rightarrow \circ \cdot L1 \circ) = \{L \rightarrow \circ \cdot L1 \circ, L1 \rightarrow \circ \cdot L1 \circ [LR]\}$
$I_5 = Goro(I_0, id) = CL(L \rightarrow \circ \cdot id \circ) = \{L \rightarrow \circ \cdot id \circ\}$
$I_6 = Goro(I_0, ()) = CL(L \rightarrow \circ \cdot (-L)) = \{L \rightarrow \circ \cdot (-L), L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L), L \rightarrow \circ \cdot L1 [LR]\}$
$I_7 = Goro(I_0, =) = CL(S \rightarrow L \circ = RS) = \{S \rightarrow L \circ = RS, R \rightarrow \circ \cdot L + R, R \rightarrow \circ \cdot R \star R, R \rightarrow \circ \cdot L, R \rightarrow \circ \cdot (R), L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L), L \rightarrow \circ \cdot L1 [LR]\}$
$I_8 = Goro(I_0, L) = CL(L \rightarrow \circ \cdot L \circ) = \{L \rightarrow \circ \cdot L \circ\}$
$Goro(I_0, *) = CL(L \rightarrow \circ \cdot *L) = I_3$
$Goro(I_0, L1) = I_4$
$Goro(I_0, id) = I_5$
$Goro(I_0, ()) = CL(L \rightarrow \circ \cdot (-L)), \{L \rightarrow \circ \cdot (-L), L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L)\} = I_6$
$I_9 = Goro(I_0, L) = CL(L \rightarrow \circ \cdot (L \circ)) = \{L \rightarrow \circ \cdot (L \circ)\}$
$Goro(I_0, *) = CL(L \rightarrow \circ \cdot *L) = I_3$
$Goro(I_0, L1) = CL(L \rightarrow \circ \cdot L1 \circ, L1 \rightarrow \circ \cdot L1 \circ [LR]) = I_4$
$Goro(I_0, id) = I_5$
$Goto(I_0, ()) = I_6$
$I_{10} = Goro(I_0, E) = CL(L1 \rightarrow \circ \cdot L1 [LR]) = \{L1 \rightarrow \circ \cdot L1 [LR], R \rightarrow \circ \cdot R + R, R \rightarrow \circ \cdot R \star R, R \rightarrow \circ \cdot L, R \rightarrow \circ \cdot (R), L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L1 \rightarrow \circ \cdot L1 [LR], L1 \rightarrow \circ \cdot id, L1 \rightarrow \circ \cdot (L)\}$
$I_{11} = Goro(I_0, R) = CL(S \rightarrow L = RS, R \rightarrow \circ \cdot R + R, R \rightarrow \circ \cdot R \star R)$
$\{S \rightarrow L = RS, R \rightarrow \circ \cdot R + R, S \rightarrow \circ \cdot L = RS, S \rightarrow \circ \cdot L, L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L \rightarrow \circ \cdot id, L \rightarrow \circ \cdot (L), L \rightarrow \circ \cdot L1 [LR], L1 \rightarrow \circ \cdot id, L1 \rightarrow \circ \cdot (L)\}$
$I_{12} = Goro(I_0, L) = \{R \rightarrow \circ \cdot L \circ\}$
$I_{13} = Goro(I_0, ()) = CL(R \rightarrow \circ \cdot (R)), \{R \rightarrow \circ \cdot (R), R \rightarrow \circ \cdot R + R, R \rightarrow \circ \cdot R \star R, R \rightarrow \circ \cdot L, R \rightarrow \circ \cdot (R), L \rightarrow \circ \cdot *L, L \rightarrow \circ \cdot L1, L1 \rightarrow \circ \cdot L1 [LR], L1 \rightarrow \circ \cdot id, L1 \rightarrow \circ \cdot (L)\}$
$Goro(I_0, L) = I_3$
$Goro(I_0, L1) = I_4$
$Goro(I_0, id) = I_5$
$I_{14} = Goro(I_0,)) = CL(L1 \rightarrow \circ \cdot (L \circ)) = \{L1 \rightarrow \circ \cdot (L \circ)\}$
$I_{15} = Goro(I_0, R) = CL(L1 \rightarrow \circ \cdot L1 [R \circ], R \rightarrow \circ \cdot R + R, R \rightarrow \circ \cdot R \star R)$
$\{L1 \rightarrow \circ \cdot L1 [R \circ], R \rightarrow \circ \cdot R + R, R \rightarrow \circ \cdot R \star R\}$
$Goro(I_{10}, L) = CL(R \rightarrow \circ \cdot L \circ) = I_{12}$
$Goro(I_{10}, ()) = CL(R \rightarrow \circ \cdot (R), L1 \rightarrow \circ \cdot (L)) = I_{13}$
$Goro(I_{10}, *) = I_3$

$$\begin{aligned}\tau_0 &= \circ \cdot L, \circ \cdot (L), \circ \\ \text{NR} &: S, S \circ, L, L1, R\end{aligned}$$

Figure 1: Calcolo della collezione canonica (pt.1). Nota, $goto(I_7, L1) = I_4$ è errato, è da sostituirsi con $goto(I_7, *) = I_3$.

$$\begin{aligned}
& \text{Goro } (\mathcal{I}_{10}, \vee) = \text{cl} ((L \rightarrow L \cdot, L \cdot \rightarrow L \cdot [R])) = \mathcal{I}_4 \\
& \text{Goro } (\mathcal{I}_{10}, \text{id}) = \mathcal{I}_5 \\
& \mathcal{I}_{16} = \text{Goro } (\mathcal{I}_4, s) = \text{cl} (S \rightarrow L = RS \cdot) = \{ S \rightarrow L = RS \cdot \} \\
& \mathcal{I}_{17} = \text{Goro } (\mathcal{I}_4, +) = \text{cl} (R \rightarrow R + \cdot R) = \{ R \rightarrow R + \cdot R, R \rightarrow R + R + R, R \rightarrow R * R, R \rightarrow \cdot L, R \rightarrow \cdot (R), \\
& \quad L \rightarrow \cdot L, L \rightarrow L \cdot, L \cdot \rightarrow L \cdot [R], L \cdot \rightarrow \text{id}, L \cdot \rightarrow \cdot (L) \} \\
& \mathcal{I}_{18} = \text{Goro } (\mathcal{I}_4, *) = \text{cl} (R \rightarrow R * \cdot R, L \rightarrow \cdot L) = \{ R \rightarrow R * \cdot R, L \rightarrow \cdot L, R \rightarrow R + R, R \rightarrow R * R \\
& \quad R \rightarrow \cdot L, L \rightarrow \cdot L, L \rightarrow \cdot L \cdot, L \cdot \rightarrow \cdot L \cdot [R], L \cdot \rightarrow \text{id}, L \cdot \rightarrow \cdot (L), R \rightarrow \cdot (R) \} \\
& \text{Goro } (\mathcal{I}_4, L) = \mathcal{I}_2 \\
& \text{Goro } (\mathcal{I}_4, \vee) = \mathcal{I}_4 \\
& \text{Goro } (\mathcal{I}_4, \text{id}) = \mathcal{I}_5 \\
& \text{Goro } (\mathcal{I}_4, ()) = \mathcal{I}_6 \\
& \mathcal{I}_{19} = \text{Goro } (\mathcal{I}_{13}, R) = \text{cl} (R \rightarrow (R \cdot), R \rightarrow R \cdot + R, R \rightarrow R \cdot * R) \\
& \quad = \{ R \rightarrow (R \cdot), R \rightarrow R \cdot + R, R \rightarrow R \cdot * R \} \\
& \mathcal{I}_{20} = \text{Goro } (\mathcal{I}_{13}, L) = \text{cl} (R \rightarrow L \cdot, L \cdot \rightarrow (L \cdot)) = \{ R \rightarrow L \cdot, L \cdot \rightarrow (L \cdot) \} \\
& \text{Goro } (\mathcal{I}_{13}, ()) = \mathcal{I}_{13} \\
& \text{Goro } (\mathcal{I}_{13}, *) = \mathcal{I}_3 \\
& \text{Goro } (\mathcal{I}_{13}, \text{id}) = \mathcal{I}_4 \\
& \text{Goro } (\mathcal{I}_{13}, \text{id}) = \mathcal{I}_5 \\
& \mathcal{I}_{21} = \text{Goro } (\mathcal{I}_{13}, \vee) = \{ L \cdot \rightarrow \cdot L \cdot [R] \cdot \} \\
& \text{Goro } (\mathcal{I}_{13}, +) = \text{cl} (R \rightarrow R + \cdot R) = \mathcal{I}_{17} \\
& \mathcal{I}_{22} = \text{Goro } (\mathcal{I}_{13}, *) = \text{cl} (R \rightarrow R * \cdot R) = \{ R \rightarrow R * \cdot R, R \rightarrow R + R, R \rightarrow R * R, R \rightarrow \cdot L, R \rightarrow \cdot (R), \\
& \quad L \rightarrow \cdot L, L \rightarrow L \cdot, L \cdot \rightarrow \cdot L \cdot [R], L \cdot \rightarrow \text{id}, L \cdot \rightarrow \cdot (L) \} \\
& \mathcal{I}_{23} = \text{Goro } (\mathcal{I}_{17}, R) = \text{cl} (R \rightarrow R + \cdot R, R \rightarrow R \cdot + R, R \rightarrow R \cdot * R) = \{ R \rightarrow R + R \cdot, R \rightarrow R \cdot + R, \\
& \quad R \rightarrow R \cdot * R \} \\
& \text{Goro } (\mathcal{I}_{17}, L) = \mathcal{I}_{12} \\
& \text{Goro } (\mathcal{I}_{17}, ()) = \mathcal{I}_{13} \\
& \text{Goro } (\mathcal{I}_{17}, *) = \mathcal{I}_3 \\
& \text{Goro } (\mathcal{I}_{17}, \vee) = \mathcal{I}_4 \\
& \text{Goro } (\mathcal{I}_{17}, \text{id}) = \mathcal{I}_5 \\
& \mathcal{I}_{24} = \text{Goro } (\mathcal{I}_{18}, R) = \text{cl} (R \rightarrow R * R \cdot, R \rightarrow R \cdot + R, R \rightarrow R \cdot * R) = \{ R \rightarrow R * R \cdot, R \rightarrow R \cdot + R, R \rightarrow R \cdot * R \} \\
& \mathcal{I}_{25} = \text{Goro } (\mathcal{I}_{18}, L) = \text{cl} (L \rightarrow \cdot L \cdot, R \rightarrow L \cdot) = \{ L \rightarrow \cdot L \cdot, R \rightarrow L \cdot \} \\
& \text{Goro } (\mathcal{I}_{18}, ()) = \mathcal{I}_{18} \\
& \text{Goro } (\mathcal{I}_{18}, *) = \mathcal{I}_3 \\
& \text{Goro } (\mathcal{I}_{18}, \vee) = \mathcal{I}_4 \\
& \text{Goro } (\mathcal{I}_{18}, \text{id}) = \mathcal{I}_5
\end{aligned}$$

Figure 2: Calcolo della collezione canonica (pt.2).

$\Sigma_{26} = \text{Goro}(\Sigma_{15}, 1) = \frac{1}{2} R \rightarrow (R) \cdot 1$
 $\text{Goro}(x_5, \cdot) = I_{12}$
 $\text{Goro}(x_5, *) = I_{22}$
 $\text{Goro}(x_6, 1) = I_{14}$
 $\text{Goro}(I_{12}, R) = I_{24}$
 $\text{Goro}(I_{12}, L) = I_{12}$
 $\text{Goro}(\Sigma_{12}, 1) = I_{13}$
 $\text{Goro}(\Sigma_{12}, *) = I_3$
 $\text{Goro}(I_{22}, L) = I_4$
 $\text{Goro}(I_{22}, \text{id}) = I_6$
 $\text{Goro}(I_{23}, +) = I_{17}$
 $\text{Goro}(I_{23}, *) = I_{26}$
 $\text{Goro}(I_{26}, +) = I_{17}$
 $\text{Goro}(I_{24}, *) = I_{22}$

Figure 3: Calcolo della collezione canonica (pt.1).

Per il calcolo della tabella ACTION sono necessari i valori delle funzioni follow per i simboli non terminali²:

- $FOLLOW(S')$: { $\$$ }
- $FOLLOW(S)$: { $\$$ }
- $FOLLOW(L)$: { $\$, =, *, id, (,]$, +}
- $FOLLOW(L1)$: { $\$, =, [$, $), *, id, (,]$, +}
- $FOLLOW(R)$: { $\$, (,]$, $*, id, +,)$ }

Date tali funzioni possiamo ora esplicitare la tabella di parsing che guiderà il processo del parser³:

	()	*	+	=	[]	id	\$	S'	S	L	L1	R
I_0	S6	e1	S3	e1	e1	e1	e1	S5	R2		1	2	4	
I_1	e3	e3	e3	e3	e3	e3	e3	e3	Acc.					
I_2	e4	e4	e4	e4	S7	e4	e4	e4	e5					
I_3	S6	e18	S3	e18	e18	e18	e18	S5	e6			8	4	
I_4	R4	R4	R4	R4	R4	S10	R4	R4	R4					
I_5	R6	R6	R6	R6	R6	R6	R6	R6	R6					
I_6	S6	e7	S3	e8	e8	e8	e8	S5	e6			9	4	
I_7	S13	e9	S3	e10	e10	e10	e10	S5	e6			12	4	11
I_8	R3	R3	R3	R3	R3	e21	R3	R3	R3					
I_9	e11	S14	e11	e11	e11	e11	e12	e11	e6					
I_{10}	S13	e10	S3	e10	e10	e10	e10	S5	e6			12	4	15
I_{11}	S6	e14	S18	S17	e13	e14	e14	S5	R2		16	2	4	
I_{12}	R10	R10	R10	R10	e16	e14	R10	R10	R10					
I_{13}	S13	e7	S3	e10	e10	e10	e10	S5	e6			20	4	19
I_{14}	R7	R7	R7	R7	R7	R7	R7	R7	R7					
I_{15}	e19	e14	S22	S17	e14	e14	S21	e20	e6					
I_{16}	ise	ise	ise	ise	ise	ise	ise	ise	R1					
I_{17}	S13	e10	S3	e10	e10	e10	e10	S5	e6			12	4	23
I_{18}	S13	e10	S3	e10	e10	e10	e10	S5	e6			25	4	24
I_{19}	e19	S26	S22	S17	e17	e12	e17	e20	e6					
I_{20}	R10	R10/S14	R10	R10	e13	e14	R10	R10	R10					
I_{21}	R5	R5	R5	R5	R5	R5	R5	R5	R5					
I_{22}	S13	e10	S3	e10	e10	e10	e10	S5	e6			12	4	24
I_{23}	R8	R8	R8/S22	R8/S17	e13	e14	R8	R8	R8					
I_{24}	R9	R9	R9/S22	R9/S17	e13	e14	R9	R9	R9					
I_{25}	R3/R10	R3/R10	R3/R10	R3/R10	R3	e14	R3/R10	R3/10	R3/R10					
I_{26}	R11	R11	R11	R11	e13	e14	R11	R11	R11					

Table 1: La tabella di parsing (aggregazione delle tablelle ACTION e GOTO)

²Le funzioni FIRST sono omesse per brevità.

³In rosso sono specificate gli handle a procedure di error recovery.

Vediamo ora come sono definite le diverse procedure di error recovery e cosa gli errori indichino:

- *e1*: Invalid start of an assignment. Un’istruzione di assegnamento non può iniziare con [,],=,). Consumiamo i simboli in input fino ad un carattere valido per tale stato (i.e. (, *, *id*, \$).
- *e2*: Missing LHS. Si assume che l’utente si sia dimenticato prima dell’uguale di inserire il left hand side dell’espressione. In tal caso, per evitare di scartare l’eventuale RHS, viene consumato il simbolo = e inserito sulla pila lo stato I_7 per iniziare il riconoscimento del RHS.
- *e3*: Assignment recognised, unexpected extra characters. L’input, dopo aver riconosciuto l’assegnamento dovrebbe essere terminato, dunque si consumano tutti i rimanenti token in ingresso.
- *e4*: After LHS an = was expected. Vengono consumati i token in ingresso fino al primo =, nel tentativo di riconoscere poi una right expression (in certi casi specifici può portare ad una catena di errori) o fino alla fine dell’input.
- *e5*: Assignment expected, only LHS found. L’input è terminato, il parsing termina. Questo caso è stato distinto da *e4* per fornire un’output più informativo.
- *e6*: Unexpected end of input. Errore generale nel caso l’input termini improvvisamente, l’unica cosa possibile è segnalare l’errore e terminare il parsing.
- *e7*: Left expression can’t be (). Si assume l’utente abbia nell’apertura di una parentesi accidentalmente inserito anche la sua chiusura (ad esempio causa auto-completamento). Viene consumato tale token.
- *e8*: Malformed left expression. Consumiamo i successivi simboli fino al primo *id* o (.
- *e9*: Left/right expresion can’t be (). Si assume l’utente abbia nell’apertura di una parentesi accidentalmente inserito anche la sua chiusura (ad esempio causa auto-completamento). Viene consumato tale token.
- *e10*: Malformed left/right expression. Consumiamo i successivi simboli fino ad un token valido per tale stato.
- *e11*: Valid left expression, wrong symbols following. Consumiamo i successivi simboli fino ad un token valido per tale stato.
- *e12*: Wrong closed parenthesis. Si assume l’utente abbia confuso) con], si consuma il singolo token e viene inserito nella pila lo stato I_{14} , come se fosse stata riconosciuta la parentesi corretta.

- *e13*: Missing LHS. Un’assegnamento è stato riconosciuto e si presume che un’altro debba seguire, il token in ingresso è però $=$, dunque si assume che l’utente abbia dimenticato il left hand side dell’assegnamento. Viene consumato il simbolo $=$ e inserito sulla coda lo stato I_7 per iniziare il riconoscimento del RHS.
- *e14*: Bad characters after right expression/before left expression. Consumiamo i successivi simboli fino ad un token valido per tale stato.
- *e15*: Malformed right expression. In questo caso si assume che “ R ” non sia seguito da un’altro assegnamento, ma da un’altra serie di token che completano il RHS attualmente in fase di riconoscimento. Per tale motivo consumiamo i successivi simboli fino ad un token valido per tale stato.
- *e16*: Unexpected $=$ after left expression. Abbiamo correttamente riconosciuto una left expression, non quella relativa ad un assegnamento però, dunque non possiamo aspettarci un $=$. Si assume l’utente abbia provato ad inserire un assegnamento (con tanto di right hand side) da usare come left expression, dunque viene consumato $=$ e viene fatto il push dello stato I_7 per il riconoscimento di R .
- *e17*: Malformed right expression, valid operand needed. Gestire le diverse casistiche richiederebbe alcune assunzioni che in molti casi potrebbero non essere valide, dunque consumiamo solamente i successivi simboli fino ad un token valido per tale stato.
- *e18*: Invalid start for a left expression. Consideriamo l’eventuale left expression come valida, i token in input qui considerati risultano essere un caso abbastanza irrealistico, nulla di riconducibile a qualche errore di tipo logico del programmatore, viene dunque fatto il pop di I_3 (e dell’eventuale I_{15} nel caso questo preceda I_{13}) e consumato il token.
- *e19*: Missing operand. Si assume che l’utente dopo il simbolo $($ abbia scritto un’altra right expression, dunque $($ dovrebbe marcarne l’inizio, tuttavia non è presente ne $+$ ne $*$ prima della parentesi, dunque, inseriamo I_{13} sulla pila per riconoscerla.
- *e20*: Missing operand before *id*. Caso analogo al precedente ma, in questo caso consumiamo solo il token *id*, assumendo che fosse l’ultima parte della right expression.
- *e21*: Bad characters after left expression. Caso abbastanza irrealistico, consumiamo solamente i successivi simboli (che dovrebbe ragionevolmente essere solo uno) fino ad un token valido per tale stato.
- *ise*: Internal state error. Tale errore non dovrebbe mai verificarsi, lo stato I_{16} infatti denota la fine del riconoscimento di un assegnamento e funge solo da stato di passaggio per applicare la prima riduzione. In questo caso, data l’inconsistenza della situazione il parsing verrebbe interrotto.

Per quanto riguarda i conflitti nella tabella 1 le azioni da eseguire sono quelle sottolineate. Il primo conflitto si verifica nello stato I_{20} , in questo caso come indicato in casi di conflitti shift/reduce non dipendenti da precedenze o regole di associatività viene eseguito lo shift (ciò avviene anche per il conflitto $R8/S17$ in I_{23} ⁴). In modo analaogo i conflitti reduce/reduce in I_{25} sono risolti scegliendo la prima tra le riduzioni. Per quanto riguarda il conflitto shift/reduce $R8/S22$ in I_{23} l'azione reduce è stata scelta in quanto l'operatore $*$ deve avere una maggiore priorità rispetto al $+$. Per il conflitto $R9/S22$ in I_{24} è svolta l'azione di shift in quanto il $*$ deve associare a destra (infatti l'espressione $R * R$ riconosciuta è seguita da un'altra moltiplicazione). Infine per $R9/S17$ l'operazione di reduce è stata scelta in quanto l'espressione $R + R$ riconosciuta sarà seguita da una somma, che però ha priorità minore, dunque viene prima calcolata la (eventuale) moltiplicazione riducendo la regola.

Parser LALR

Per la realizzazione del parser è stata utilizzata la tecnica che non prevede il passaggio per gli stati dell'automa LR(1), in particolare nelle Figure 4 e 5 sono calcolate le closure. Le Figure 6 e 7 mostrano invece la propagazione dei simboli di lookahead ed infine il calcolo degli stessi è riportato (Figure 8 e 9). Terminata la costruzione si è notato che la tabella GOTO del LALR risulta uguale a quella del parser SLR. Lo stesso vale per la tabella ACTION, nella quale le uniche righe diverse da quelle del parser SLR sono quelle relative agli stati I_{20} e I_{25} . In particolare per lo stato I_{20} sono ora vuote le celle che prima comportavano una riduzione tramite la regola 10 quando i seguenti lookahead erano rilevati: $(,]$, id , $\$$. Nelle celle vuote è possibile inserire le seguenti procedure di error recovery: $e12$ nel caso il lookahead sia $($, $e6$ nel caso $\$$ ed $e12$ nei rimanenti casi. Per quanto riguarda lo stato I_{23} non sono presenti più conflitti e le azioni (secondo i lookahead) sono le seguenti:

- $(: R10.$
- $) : \text{None, la cella diventa vuota.}$
- $* : R10.$
- $+ : R10.$
- $= : \text{Rimane invariata.}$
- $[: \text{None, la cella diventa vuota.}$
- $] : \text{None, la cella diventa vuota.}$
- $id : R10.$
- $\$: R10.$

Nelle celle vuote è possibile ora associare la procedura di error recovery $e3$.

⁴In questo modo il $+$ associa a sinsitra.

I kernel degli svari LR(0)

$$\begin{aligned}
 I_0 &\rightarrow S^* \Rightarrow S \\
 I_1 &\rightarrow S^* \Rightarrow S^* \\
 I_2 &\rightarrow S \Rightarrow L \Rightarrow RS \\
 I_3 &\rightarrow L \Rightarrow *L \\
 I_4 &\rightarrow L \Rightarrow L1^*, L \Rightarrow L1[R] \\
 I_5 &\rightarrow L1 \Rightarrow id \\
 I_6 &\rightarrow L1 \Rightarrow (\cdot L) \\
 I_7 &\rightarrow S \Rightarrow L = R S \\
 I_8 &\rightarrow L \Rightarrow *L^* \\
 I_9 &\rightarrow L1 \Rightarrow (L) \\
 I_{10} &\rightarrow L1 \Rightarrow L1[R] \\
 I_{11} &\rightarrow S \Rightarrow L = R \circ S, R \Rightarrow R \circ + R, R \Rightarrow R \circ * R \\
 I_{12} &\rightarrow R \Rightarrow L \\
 I_{13} &\rightarrow R \circ (R), L \Rightarrow (\circ L) \\
 I_{14} &\rightarrow L1 \Rightarrow (L) \\
 I_{15} &\rightarrow L1 \Rightarrow L1[R], R \Rightarrow R \circ + R, R \Rightarrow R \circ * R \\
 I_{16} &\rightarrow S \Rightarrow L = R S^* \\
 I_{17} &\rightarrow R \Rightarrow R + R \\
 I_{18} &\rightarrow R \Rightarrow R^* \circ R, L \Rightarrow *L
 \end{aligned}$$

CALCOLIAMO LE CLOSURE

$$I_0 \Rightarrow CL(\{(S^* \Rightarrow S, *)\}) = \{(S^* \Rightarrow S, *), (S \Rightarrow L = RS, *), (S \Rightarrow \circ, *), (L \Rightarrow *L, *), (L \Rightarrow L1, *)\}$$

$$I_1 \Rightarrow CL(\{(S^* \Rightarrow S^*, *)\}) = \{(S^* \Rightarrow S^*, *)\}$$

$$I_2 \Rightarrow CL(\{(S \Rightarrow L = RS, *)\})$$

$$I_3 \Rightarrow CL(\{L \Rightarrow *L, *\}) = \{(L \Rightarrow *L, *), (L \Rightarrow *L, *), (L \Rightarrow L1, *), (L1 \Rightarrow L1[R], *), (L1 \Rightarrow id, *), (L1 \Rightarrow (L), *)\}$$

$I_4 \Rightarrow$ Oresse, BANAU

$$I_5 \Rightarrow CL(\{\circ\}) = \{\circ\}$$

$$I_6 \Rightarrow CL(\{L1 \Rightarrow (\cdot L), *\}) = \{L1 \Rightarrow (\cdot L), *, (L \Rightarrow *L, *), (L \Rightarrow L1, *), (L1 \Rightarrow L1[R], *)\}$$

$$I_7 \Rightarrow CL(\{S \Rightarrow L = RS, *\}) = \{S \Rightarrow L = RS, *, (R \Rightarrow R + R, */*id/*), (R \Rightarrow R \circ R, */*id/*), (R \Rightarrow L, */*id/*), (R \Rightarrow (R), */*id/*), (L \Rightarrow *L, */*id/*)\}$$

Figure 4: Calcolo dei kernel e delle closure.

$$\begin{aligned}
I_8 \rightarrow CL(\{(~)\}) &= \{ (L_1 \rightarrow *L_0, \#) \} & (L \rightarrow L_1, \#/\text{id}/\#/\#), (L_1 \rightarrow -L_1 [R], \#/\text{id}/\#/\#/\#/\#) \\
I_9 \rightarrow CL(\{(\sim)\}) &= \{ (L_1 \rightarrow (L_1, \#)) \} & (L \rightarrow \text{id}, \#/\text{id}/\#/\#/\#/\#), (L_1 \rightarrow -(L_1), \#/\text{id}/\#/\#/\#/\#) \\
I_{10} \rightarrow CL(\{(L_1 \rightarrow L_1 [-R], \#)\}) &= \{ (L_1 \rightarrow L_1 [R], \#), (R \rightarrow R + R, \#/\#/\#), (R \rightarrow R * R, \#/\#/\#) \\
&\quad (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow *L, \#/\#/\#/\#/\#/\#) \} & (L \rightarrow L_1, \#/\#/\#/\#/\#/\#), (L_1 \rightarrow -L_1, \#/\#/\#/\#/\#/\#) \\
&\quad (L_1 \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L_1 \rightarrow -(L_1), \#/\#/\#/\#/\#/\#) \\
I_{11} \rightarrow CL(\{S \rightarrow L = R = S, \#\}) &= \{ (S \rightarrow L = R = S, \#), (S \rightarrow \text{id}, \#), (S \rightarrow L = R, \#/\#), (L \rightarrow *L, \#) \} \\
I_{12} \rightarrow CL(\{\sim\}) &= \{ (R \rightarrow L, \#) \} & (L \rightarrow L_1, \#), (L_1 \rightarrow -L_1 [R], \#), (L_1 \rightarrow \text{id}, \#/\#), (L_1 \rightarrow -(L), \#/\#) \\
I_{13} \rightarrow CL(\{R \rightarrow (\circ R), \#\}) &= \{ (R \rightarrow (\circ R), \#), (R \rightarrow R + R, \#/\#/\#), (R \rightarrow R * R, \#/\#/\#), (R \rightarrow -L, \#/\#/\#) \\
&\quad (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow *L, \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#) \} & (R \rightarrow (\circ R), \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#) \\
I_{14} \rightarrow \text{Onesse} & \\
I_{15} \rightarrow \text{Onesse} & \\
I_{16} \rightarrow \text{Onessa} & \\
I_{17} = CL(\{R \rightarrow R + R, \#\}) &= \{ (R \rightarrow R + R, \#), (R \rightarrow R + R, \#/\#/\#), (R \rightarrow R * R, \#/\#/\#/\#/\#/\#), (R \rightarrow -R, \#/\#/\#/\#/\#/\#) \\
&\quad (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow *L, \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#) \} & (L \rightarrow L_1 [R], \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow -(L), \#/\#/\#/\#/\#/\#) \\
I_{18} \rightarrow CL(\{R \rightarrow R * R, \#\}) &= \{ (R \rightarrow R * R, \#), (R \rightarrow R + R, \#/\#/\#/\#/\#/\#), (R \rightarrow R * R, \#/\#/\#/\#/\#/\#) \\
&\quad (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (R \rightarrow -R, \#/\#/\#/\#/\#/\#), (L \rightarrow *L, \#/\#/\#/\#/\#/\#), \\
&\quad (L \rightarrow L_1, \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow -L_1 [R], \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#) \\
&\quad (L \rightarrow -(L), \#/\#/\#/\#/\#/\#) \} & \\
I_{19} \rightarrow CL(\{L \rightarrow *L, \#\}) &= \{ (L \rightarrow *L, \#), (L \rightarrow -L, \#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L_1 \rightarrow -L_1 [R], \#/\#/\#/\#/\#/\#) \\
&\quad (L_1 \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L_1 \rightarrow -(L), \#/\#/\#/\#/\#/\#) \} & \\
I_{20} = CL(\{R \rightarrow R * R, \#\}) &= \{ (R \rightarrow R * R, \#), (R \rightarrow R + R, \#/\#/\#/\#/\#/\#), (R \rightarrow R * R, \#/\#/\#/\#/\#/\#) \\
&\quad (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (R \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L \rightarrow *L, \#/\#/\#/\#/\#/\#), (L \rightarrow \text{id}, \#/\#/\#/\#/\#/\#) \\
&\quad (L_1 \rightarrow L_1 [R], \#/\#/\#/\#/\#/\#), (L_1 \rightarrow \text{id}, \#/\#/\#/\#/\#/\#), (L_1 \rightarrow -(L), \#/\#/\#/\#/\#/\#) \} &
\end{aligned}$$

Le numerate closure sono onesse. (dai I_{15} - I_{20})

Figure 5: Calcolo delle closure.

FROM	TO
$I_0: S \rightarrow S$	$I_1: S \rightarrow S$ $I_2: S \rightarrow L = RS$
$I_2: S \rightarrow L = RS$	$I_7: S \rightarrow L = RS$
$I_3: L \rightarrow *_0 L$	$I_8: L \rightarrow *_0 L$ $I_9: L \rightarrow *_0 L$ $I_{10}: L \rightarrow L1.$ $I_{11}: L1 \rightarrow L1[LR]$ $I_{12}: L1 \rightarrow id.$ $I_{13}: L1 \rightarrow L1(L)$
$I_4: L1 \rightarrow L1[LR]$	$I_{14}: L1 \rightarrow L1[LR]$
$I_5: L1 \rightarrow (*L)$	$I_{15}: L1(L)$
$I_7: S \rightarrow L = RS$	$I_6: S \rightarrow L = RS$ $I_8: R \rightarrow R = R$ $I_9: R \rightarrow R = *R$ $I_{10}: R \rightarrow L =$ $I_{11}: R \rightarrow (*R)$ $I_{12}: L \rightarrow *_0 L$ $I_{13}: L \rightarrow L1.$ $I_{14}: L1 \rightarrow L1[LR]$ $I_{15}: L1 \rightarrow id.$ $I_{16}: L1 \rightarrow L1(L1)$
$I_8: L1 \rightarrow (L)$	$I_{17}: L1 \rightarrow L(L)$
$I_{10}: L1 \rightarrow L1[LR]$	$I_{18}: L1 \rightarrow L1[LR]$
$I_{11}: S \rightarrow L = RS$	$I_{19}: S \rightarrow L = RS$ $I_{20}: S \rightarrow L = RS$
$R \rightarrow R + R$	$I_{21}: R \rightarrow R$
$R \rightarrow R \cdot *R$	$I_{22}: R \cdot R$
$I_{13}: R \rightarrow (*R)$	$I_{23}: R \rightarrow (R)$
$L \rightarrow (*L)$	$I_{24}: L \rightarrow (L)$
$I_{15}: L1 \rightarrow L1[LR]$ $R \rightarrow R \cdot R$ $R \rightarrow R \cdot *R$	$I_{25}: L1 \rightarrow L1[LR]$ $I_{26}: R \rightarrow R \cdot R$ $I_{27}: R \rightarrow R \cdot *R$
$I_{17}: R \rightarrow R + R$	$I_{28}: R \rightarrow R + R$ $I_{29}: R \rightarrow R \cdot R$ $I_{30}: R \rightarrow R \cdot *R$ $I_{31}: R \rightarrow (R)$
	$I_{32}: R \rightarrow L =$ $I_{33}: L \rightarrow *_0 L$ $I_{34}: L \rightarrow L1.$ $I_{35}: L1 \rightarrow id.$ $I_{36}: L1 \rightarrow L1(L)$
	$I_{37}: L1 \rightarrow L1[LR]$

Figure 6: Propagazione dei lookahead (pt.1).

$I_{18}: R \Rightarrow R^* \cdot R$	$I_{24}: R \Rightarrow R^* R \circ$ $I_{24}: R \Rightarrow R \circ + R$ $I_{24}: R \Rightarrow R \circ \# R$ $I_{24}: R \Rightarrow L \circ$ $I_{25}: R \Rightarrow (R)$	$I_{23}: L \Rightarrow * \cdot L$ $I_{23}: L \Rightarrow L \cdot \circ$ $I_{23}: L \Rightarrow L \cdot \# R$ $I_{23}: L \Rightarrow L \cdot \text{id}$ $I_{23}: L \Rightarrow L \cdot (\cdot L)$
$L \Rightarrow * \cdot L$	$I_{23}: L \Rightarrow * \cdot L$ $I_{23}: L \Rightarrow * \cdot L$ $I_{23}: L \Rightarrow * \cdot L$	$I_{23}: L \Rightarrow L \cdot \# R$ $I_{23}: L \Rightarrow \text{id}$ $I_{23}: L \Rightarrow (\cdot L)$
$I_{19}: R \Rightarrow (R) \circ$ $R \Rightarrow R \circ + R$ $R \Rightarrow R \cdot * R$	$I_{26}: R \Rightarrow (R) \circ$ $I_{17}: R \Rightarrow R$ $I_{22}: R \Rightarrow R \cdot * R$	
$I_{20}: \cancel{R \Rightarrow R}$ $L \cdot \Rightarrow (L \cdot)$	$\cancel{I_{20}}$ $I_{24}: L \cdot \Rightarrow (L \cdot)$	
$I_{22}: R \Rightarrow R^* \cdot R$	$I_{24}: R \Rightarrow R^* R \circ$ $I_{24}: R \Rightarrow R \circ + R$ $I_{24}: R \Rightarrow R \circ \# R$ $I_{24}: R \Rightarrow L \circ$ $I_{25}: R \Rightarrow (R)$	$I_{23}: L \Rightarrow * \cdot L$ $I_{23}: L \Rightarrow L \cdot \circ$ $I_{23}: L \Rightarrow L \cdot \# R$ $I_{23}: L \Rightarrow L \cdot \text{id}$ $I_{23}: L \Rightarrow L \cdot (\cdot L)$
$I_{23}: R \Rightarrow R \circ + R$ $R \Rightarrow R \cdot * R$	$I_{17}: R \Rightarrow R \circ + R$ $I_{22}: R \Rightarrow R \cdot * R$	
$I_{24}: (\text{come } IL \text{ caso sopra})$		$(\text{come } IL \text{ caso sopra})$

Figure 7: Propagazione dei lookahead (pt.2).

STATE e Iren	INIT	P1	P2	P3	P4	PF FISSO
I ₀ S' \rightarrow^o S	\$	↓	↓		↔	
I ₁ S' \rightarrow^o S ₀	\$	↓	↓			
I ₂ S \rightarrow^o L $\circ = R S$		\$/\ast / id / + / \ast	\$/\ast / id / \ast			
I ₃ L \rightarrow^o * - L	+ / *	+ / *	* / + / id / \$			
I ₄ L \rightarrow^o L ₁ \circ L \leftarrow^o L ₁ $\circ [R]$		* / +	\$/\ast / + / id / + / \ast			
I ₅ L ₁ \rightarrow^o id ₀	E / * / id / () / +	E / * / id / () / +	E / * / id / () / +			
I ₆ L ₁ \rightarrow^o (- L)	E	+ / * / E	→			
I ₇ S \rightarrow^o L $\circ = R S$		\$/\ast / id / + / \ast	\$/\ast / id / +			
I ₈ L \rightarrow^o * L ₀	+ / *	+ / *				
I ₉ L ₁ \rightarrow^o (L ₀)						
I ₁₀ L ₁ \rightarrow^o L ₁ $\circ [R]$	E / + / id / () / +	-				
I ₁₁ S \rightarrow^o L = R + S R \rightarrow^o R ₀ + R R \rightarrow^o R \circ R	* / id / () / +	→	\$/\ast / id / + / *			
I ₁₂ R \rightarrow^o L ₀	+ / *	→	\$/\ast / id / +			
I ₁₃ R \rightarrow^o (o R) L ₁ \rightarrow^o (- L)	* / id / () / +	→	\$/\ast / id / + / /			
I ₁₄ L ₁ \rightarrow^o (L ₀)	* / id / () / + / E	→	\$/\ast / id / + / C			
I ₁₅ L ₁ \rightarrow^o L ₁ $\circ [R]$ R \rightarrow^o R ₀ + R R \rightarrow^o R \circ R						
I ₁₆ S \rightarrow^o L = R S -		+ / id / () / +	\$/\ast / id / + / *			
I ₁₇ R \rightarrow^o R + S		+ / id / () / +	\$/\ast / id / + / *			
I ₁₈ R \rightarrow^o R \circ R L \rightarrow^o * o L		+ / id / () / +	\$/\ast / id / + / *			
I ₁₉ R \rightarrow^o (R ₀) R \rightarrow^o R ₀ + R R \rightarrow^o R \circ R		+ / id / () / +	+ / id / () / + / \$			
I ₂₀ R \rightarrow^o L ₀ L ₁ \rightarrow^o (- L ₀)	E / () / + / *	C / () / + / *	L / () / + / * / \$ / id			
I ₂₁ L ₁ \rightarrow^o L ₁ $\circ [R]$						
I ₂₂ R \rightarrow^o R \circ R		* / +	* / + / id / + / *			
I ₂₃ R \rightarrow^o R + R ₀ R \rightarrow^o R ₀ + R R \rightarrow^o R \circ R						
I ₂₄ R \rightarrow^o R + R ₀ R \rightarrow^o R ₀ + R R \rightarrow^o R \circ R	* / +	→	\$/\ast / id / + / + / /			
I ₂₅ L \rightarrow^o * L ₀	* / +	→	\$/\ast / id / + / C			

Figure 8: Calcolo dei lookahead (pt.1).

$R \rightarrow L^0$	* / t	* / t	* / t
$I_{26} \quad R \rightarrow (R)^0$		* / id / (/ t	* / id / (/ t / #

A UNA TERZA ITERAZIONE È POSSIBILE CREARNE (SUCCESSIVAMENTE) IL PUNTO FISSO.
 DALLA SECONDA ITERAZIONE, PER VELOCIZZARE LA PROCEDURA LA COMPUTAZIONE DEL LOOKAHEAD È STATA FATTA CONDUCENDO ANCHE LE MODIFICHE ALL'ISTRUZIONE CORRENTE (P2) E NON SOLO P1.

Figure 9: Calcolo dei lookahead (pt.2).

Esercizio 2

La Figura 10 riporta l'esecuzione del parser SLR sull'esempio richiesto ($id[id(id*)] = (*id**id)[id]id = id$). Il parsing dell'automa LALR (sullo stesso esempio) risulta uguale in quanto le parti della tabella che differiscono nei due casi (descritte in precedenza) non risultano essere raggiunte (dunque l'esempio mostra entrambe le esecuzioni).

STACK	INPUT	CURR
\$ I ₀	$id[id(id*)] = (*id**id)[id]id = id \$$	S5
\$ I ₀ I ₂	$\Sigma_{id}(id^*) = (*id^{**}id)[id]id = id \$$	R6 (L1 \rightarrow id)
\$ I ₀ I ₄	"	S10
\$ I ₀ I ₄ I ₁₀	$id(id) = (*id^{**}id)[id]id = id \$$	S5
\$ I ₀ I ₄ I ₁₀ I ₅	$(id^*) = (*id^{**}id)[id]id = id \$$	R6 (L1 \rightarrow id)
\$ I ₀ I ₄ I ₁₀ I ₄	"	R4 (L \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₂	"	R10 (R \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅	"	E15 (PUSH I ₁₅)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅	"	S13
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂	$id^* \dots$	S5
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅	$* = (*id^{**}id)[id]id = id \$$	R6 (L1 \rightarrow id)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄	"	R4 (L \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆	"	R10 (R \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅	"	S3
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃	$= (*id^{**}id)[id]id = id \$$	E18 (POP e consume)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅	$* id^{**}id [id]id = id \$$	S22
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂	$id^{**}id [id]id = id \$$	S5
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅	$**id [id]id = id \$$	R6 (L1 \rightarrow id)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄	"	R4 (L \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂	"	R10 (R \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄	"	S22
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆	$* id [id]id = id \$$	S3
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅	$id [id]id = id \$$	S5
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄	$) [id]id = id \$$	R6 (L1 \rightarrow id)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂	"	R4 (L \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂	$I_{15}I_{22} [id]id = id \$$	E10 (consume up to id)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅	$I_{15} = id \$$	S5
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅	$= id \$$	R5 (L1 \rightarrow id)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅ I ₄	$= id \$$	R4 (L \rightarrow L)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅ I ₄ I ₆	$I_{15} = id \$$	E17 (PUSH I ₇ , consume)
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅ I ₄ I ₆ I ₁₅	"	S5
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅ I ₄ I ₆ I ₁₅ I ₅	$R6(L1 \rightarrow id)$	
\$ I ₀ I ₄ I ₁₀ I ₅ I ₁₅ I ₁₂ I ₅ I ₄ I ₆ I ₁₅ I ₁₃ I ₁₅ I ₂₂ I ₅ I ₄ I ₂₂ I ₄ I ₆ I ₁₅ I ₄ I ₂₂ I ₂₂ I ₅ I ₅ I ₄ I ₆ I ₁₅ I ₅ I ₄	"	E6 (error)

Figure 10: Esempio di parsing SLR