

Relatório do trabalho SLR vs LALR  
Nome: Leonardo Pereira dos Santos  
Data: 17/07/2021

## 1) Introdução

Esse é um relatório do trabalho SLR vs LALR da matéria de compiladores, com o intuito de verificar se gramática

```
S -> L = R | R
L -> * R | id
R -> L
```

é válida para ambos os analisadores sintáticos (SLR e LALR).

Os dois próximos tópicos são apenas cópias das saídas do arquivo parser.out gerado pelo meu próprio código, para os dois tipos de analisadores. Já o tópico 3 eu verifico o conflito para SLA

## 2) parser.out da analisador LALR

### Grammar

```
Rule 0   S' -> S
Rule 1   S -> L igual R
Rule 2   S -> R
Rule 3   L -> multe R
Rule 4   L -> ident
Rule 5   R -> L
```

### Terminals, with rules where they appear

```
error      :
ident      : 4
igual      : 1
multe      : 3
```

### Nonterminals, with rules where they appear

```
L          : 1 5
R          : 1 2 3
S          : 0
```

Parsing method: LALR

state 0

- (0)  $S' \rightarrow \cdot S$
- (1)  $S \rightarrow \cdot L \text{ igual } R$
- (2)  $S \rightarrow \cdot R$
- (3)  $L \rightarrow \cdot \text{ multe } R$
- (4)  $L \rightarrow \cdot \text{ ident}$
- (5)  $R \rightarrow \cdot L$

multe	shift and go to state 4
ident	shift and go to state 5

S	shift and go to state 1
L	shift and go to state 2
R	shift and go to state 3

state 1

- (0)  $S' \rightarrow S \cdot$

state 2

- (1)  $S \rightarrow L \cdot \text{ igual } R$
- (5)  $R \rightarrow L \cdot$

igual	shift and go to state 6
\$end	reduce using rule 5 ( $R \rightarrow L \cdot$ )

state 3

- (2)  $S \rightarrow R \cdot$

\$end	reduce using rule 2 ( $S \rightarrow R \cdot$ )
-------	---

state 4

- (3)  $L \rightarrow \text{ multe } \cdot R$
- (5)  $R \rightarrow \cdot L$
- (3)  $L \rightarrow \cdot \text{ multe } R$
- (4)  $L \rightarrow \cdot \text{ ident}$

multe	shift and go to state 4
ident	shift and go to state 5

R	shift and go to state 7
L	shift and go to state 8

state 5

(4) L -> ident .

igual	reduce using rule 4 (L -> ident .)
\$end	reduce using rule 4 (L -> ident .)

state 6

(1) S -> L igual . R

(5) R -> . L

(3) L -> . multe R

(4) L -> . ident

multe	shift and go to state 4
ident	shift and go to state 5

L	shift and go to state 8
R	shift and go to state 9

state 7

(3) L -> multe R .

igual	reduce using rule 3 (L -> multe R .)
\$end	reduce using rule 3 (L -> multe R .)

state 8

(5) R -> L .

igual	reduce using rule 5 (R -> L .)
\$end	reduce using rule 5 (R -> L .)

state 9

(1) S -> L igual R .

\$end	reduce using rule 1 (S -> L igual R .)
-------	--

3)parser.out da analisador SLR

Para usar o analisador SLR é preciso comentar a linha 35  
que se parece com isso: parser = yacc.yacc(method="LALR")  
e descomentar a linha 36 que se parece com isso:  
parser = yacc.yacc(method="SLR")

arquivo parser.out gerado :

Grammar

Rule 0 S' -> S  
Rule 1 S -> L igual R  
Rule 2 S -> R  
Rule 3 L -> multe R  
Rule 4 L -> ident  
Rule 5 R -> L

Terminals, with rules where they appear

error :  
ident : 4  
igual : 1  
multe : 3

Nonterminals, with rules where they appear

L : 1 5  
R : 1 2 3  
S : 0

Parsing method: SLR

state 0

(0) S' -> . S  
(1) S -> . L igual R  
(2) S -> . R  
(3) L -> . multe R  
(4) L -> . ident  
(5) R -> . L

multe shift and go to state 4  
ident shift and go to state 5

S shift and go to state 1

L	shift and go to state 2
R	shift and go to state 3

state 1

(0)  $S' \rightarrow S \cdot$

state 2

(1)  $S \rightarrow L \cdot$  igual R

(5)  $R \rightarrow L \cdot$

! shift/reduce conflict for igual resolved as shift

igual	shift and go to state 6
-------	-------------------------

\$end	reduce using rule 5 ( $R \rightarrow L \cdot$ )
-------	---

! igual [ reduce using rule 5 ( $R \rightarrow L \cdot$ ) ]

state 3

(2)  $S \rightarrow R \cdot$

\$end	reduce using rule 2 ( $S \rightarrow R \cdot$ )
-------	---

state 4

(3)  $L \rightarrow multe \cdot R$

(5)  $R \rightarrow \cdot L$

(3)  $L \rightarrow \cdot multe R$

(4)  $L \rightarrow \cdot ident$

multe	shift and go to state 4
-------	-------------------------

ident	shift and go to state 5
-------	-------------------------

R	shift and go to state 7
---	-------------------------

L	shift and go to state 8
---	-------------------------

state 5

(4)  $L \rightarrow ident \cdot$

igual	reduce using rule 4 ( $L \rightarrow ident \cdot$ )
-------	---

\$end	reduce using rule 4 ( $L \rightarrow ident \cdot$ )
-------	---

state 6

(1) S -> L igual . R

(5) R -> . L

(3) L -> . multe R

(4) L -> . ident

multe            shift and go to state 4

ident            shift and go to state 5

L                    shift and go to state 8

R                    shift and go to state 9

state 7

(3) L -> multe R .

igual            reduce using rule 3 (L -> multe R .)

\$end            reduce using rule 3 (L -> multe R .)

state 8

(5) R -> L .

\$end            reduce using rule 5 (R -> L .)

igual            reduce using rule 5 (R -> L .)

state 9

(1) S -> L igual R .

\$end            reduce using rule 1 (S -> L igual R .)

WARNING:

WARNING: Conflicts:

WARNING:

WARNING: shift/reduce conflict for igual in state 2 resolved as shift

#### 4) Conflito para o SLR

Um dos conflitos que costumam acontecer no analisador SLR é o shift/reduce.

ele ocorre quando chegamos em um estado onde temos duas opções, uma redução e um shift, no entanto existe uma particularidade, o follow que calculamos para saber onde temos que colocar os reduces na tabela é o mesmo que um dos shift.

Para a gramática estudada temos esse caso no estado 2

Estado 2

$S \rightarrow L \bullet = R$

$R \rightarrow L \bullet$

obs: para facilitar a escrita estou utilizando o símbolo de = no lugar da palavra igual como está escrito no arquivo parser.out

observe que "=" pertence ao follow(R). Com isso temos uma redução encontrando o "=" e um shift também encontrando o "=" então temos o conflito shift/reduce

observado o final do arquivo parser.out do SLR percebemos no "WARNING: shift/reduce conflict for igual in state 2 resolved as shift" que ele também achou o conflito, observe também que ele decide utilizar o shift, pois esse é o padrão para esse conflito.

#### Conclusão

Como temos um conflito do tipo shift/reduce no SLA podemos concluir que a gramática

$S \rightarrow L = R \mid R$

$L \rightarrow * R \mid id$

$R \rightarrow L$

não é do tipo SLA.

Como não temos nenhum tipo de conflito na LALR podemos concluir que a gramática acima é do tipo LALR