

Lenguajes y Sistemas Informáticos para la resolución de problemas complejos



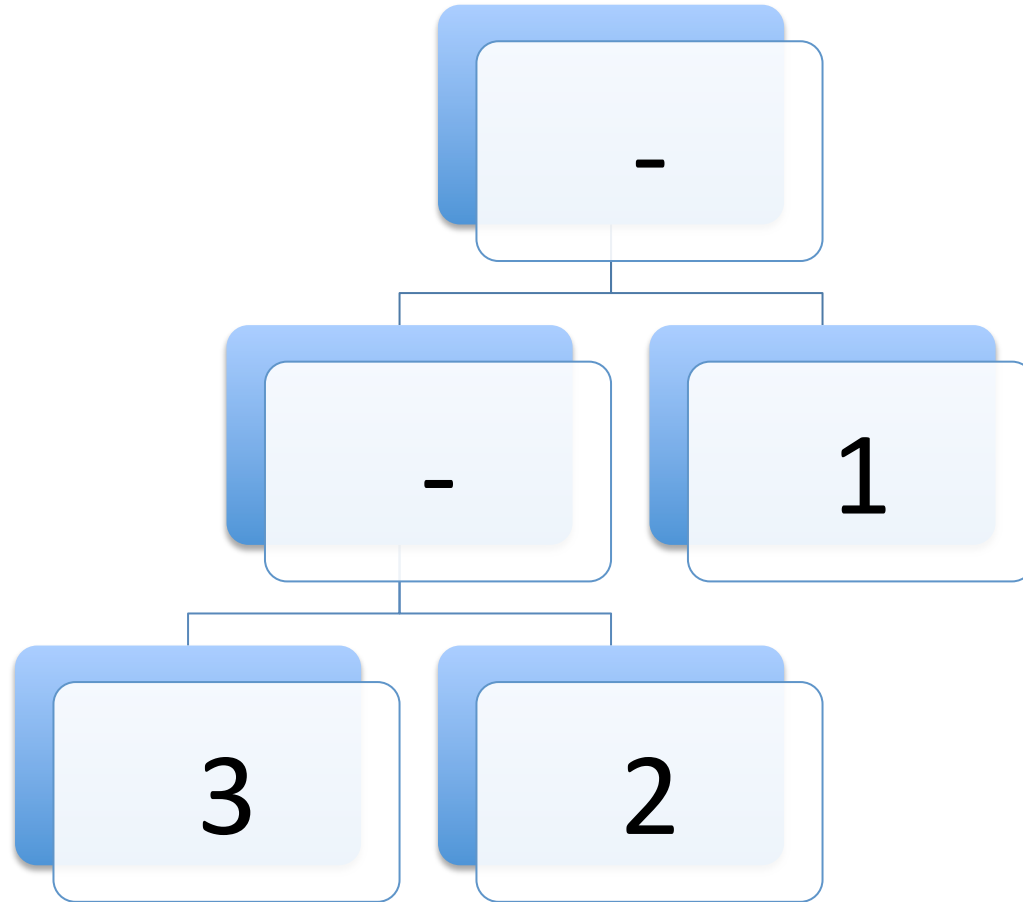
Procesadores de Lenguajes

Casiano Rodríguez León

3 - 2 - 1

Árbol Sintáctico Abstracto

$(3-2)-1$



Semántica 3 - 2 - 1

$$0 = 1 - 1$$

-

$$1 = 3 - 2$$

-

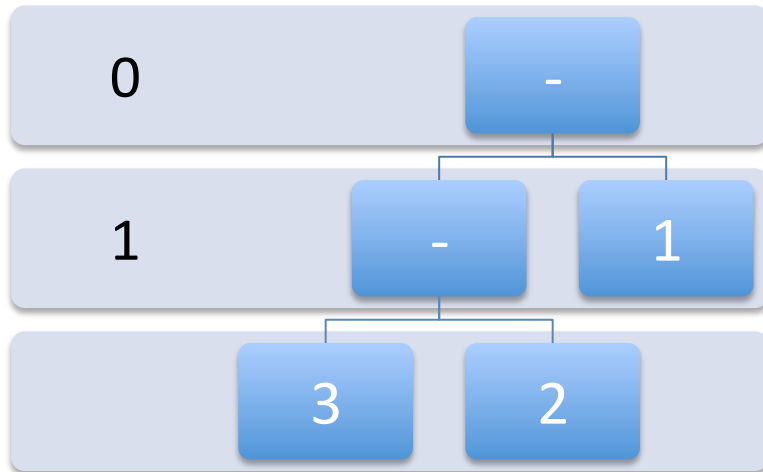
1

3

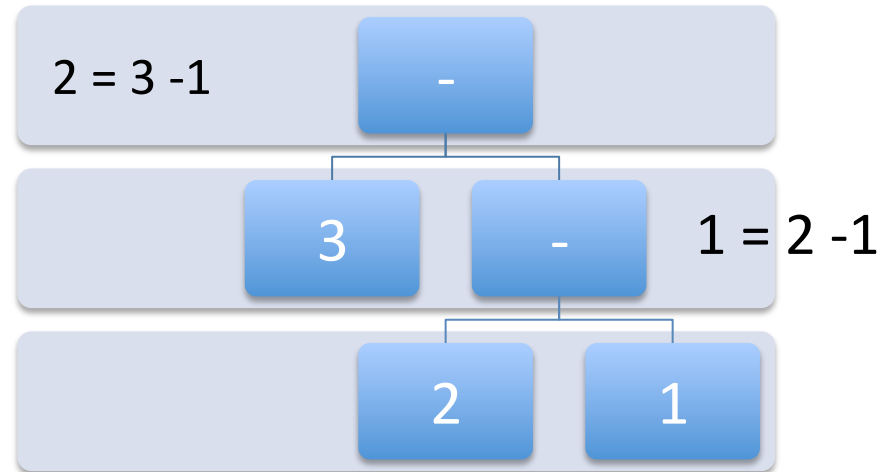
2

Semántica y Ambigüedad

$(3-2)-1$



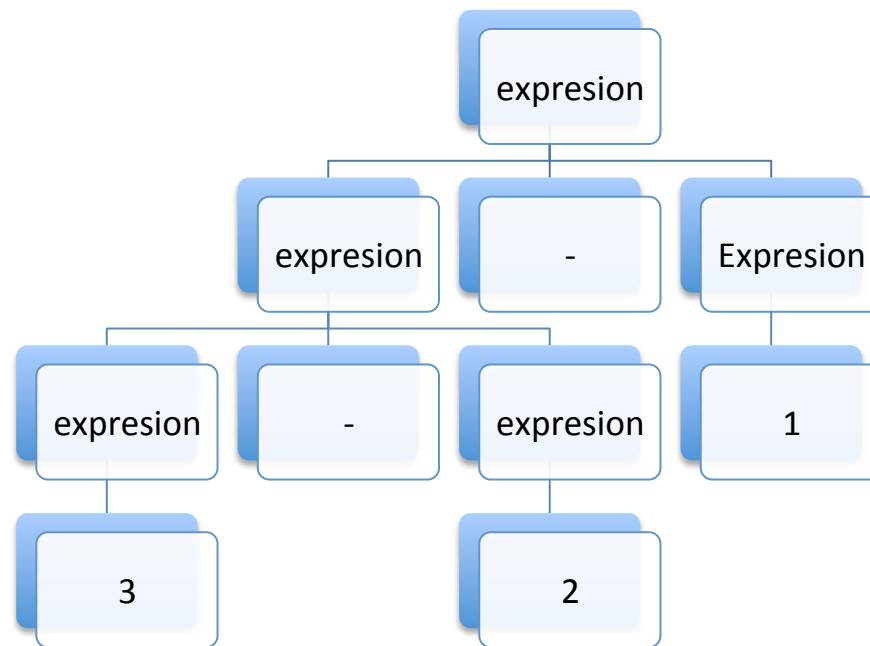
$3-(2-1)$



Gramática Independiente del Contexto

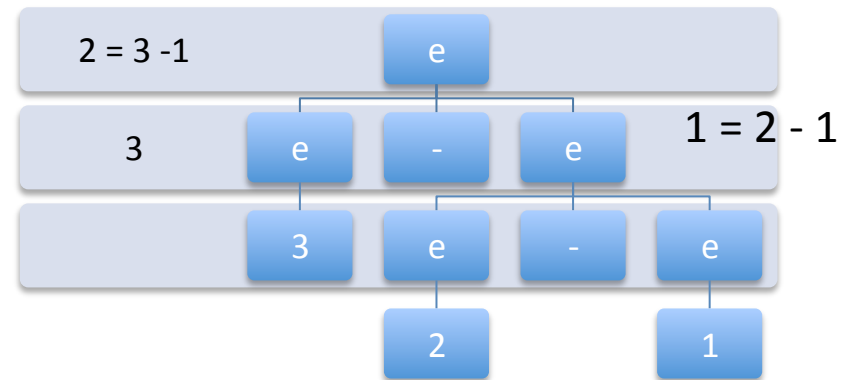
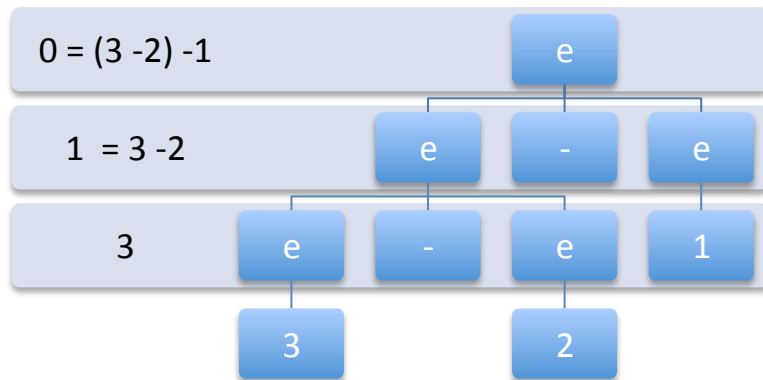
- $\text{expresion} \rightarrow \text{expresion} \text{ '-' expresion}$
- $\text{expresion} \rightarrow \text{NUMERO}$

(3-2)-1



Gramática Ambigua

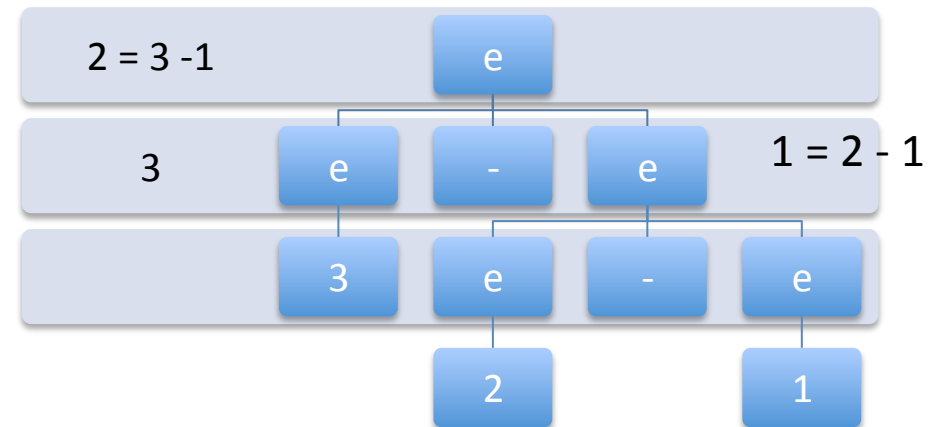
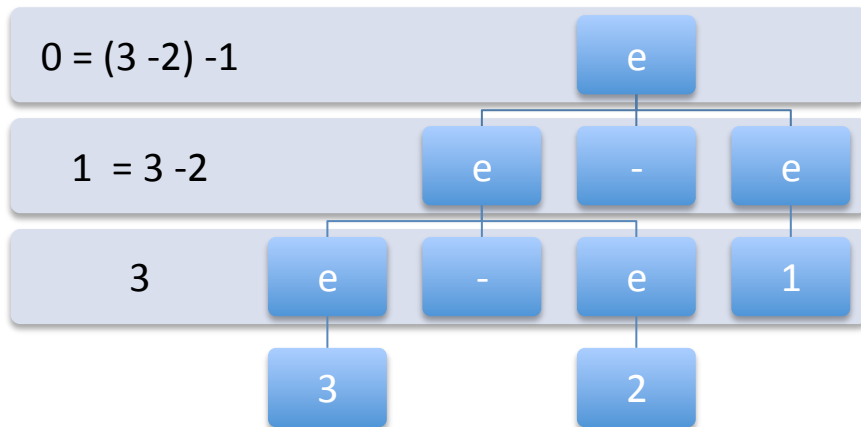
- $\text{expresion} \rightarrow \text{expresion} \text{ '-' } \text{expresion}$
- $\text{expresion} \rightarrow \text{NUMERO}$



Esquema de Traducción (yacc)

$e \rightarrow e \text{ '-' } e \quad \{ \$\$ = \$1 - \$3; \}$
 $e \rightarrow \text{NUM} \quad \{ \$\$ = \text{Number}(\$1); \}$

3 - 2 - 1



Parsing: Construcción del Árbol

$e \rightarrow e \text{ '-' } e \quad \{ \$\$ = \$1 - \$3; \}$

$e \rightarrow \text{NUM} \quad \{ \$\$ = \text{Number}(\$1); \}$

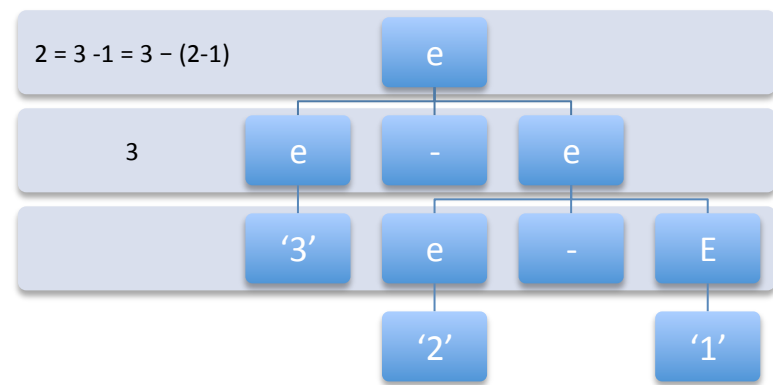
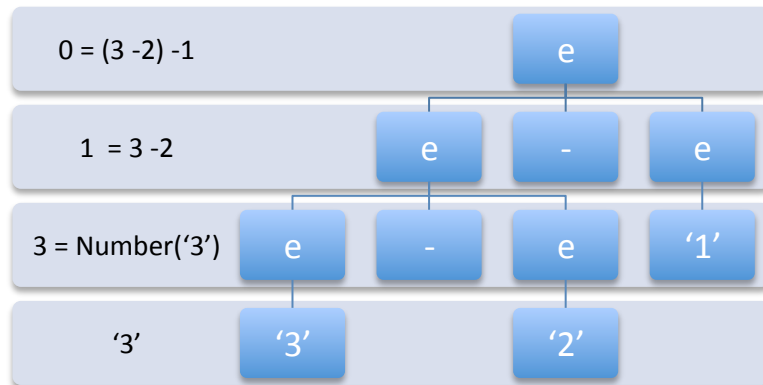
Análisis Sintáctico Ascendente:

$.3 - 2 - 1 \leq e. - 2 - 1 \leq = e -. 2 - 1 \leq = e - 2. - 1 \leq e - e. - 1$

¿Qué hacer?

1. $\leq e. - 1 \leq e -. 1 \leq e - 1. \leq e - e. \leq e.$

2. $\leq e - e -. 1 \leq e - e - 1. \leq e - e - e. \leq e - e. \leq e.$



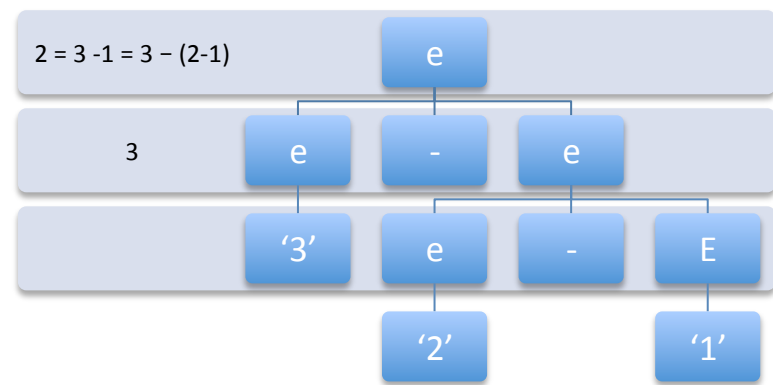
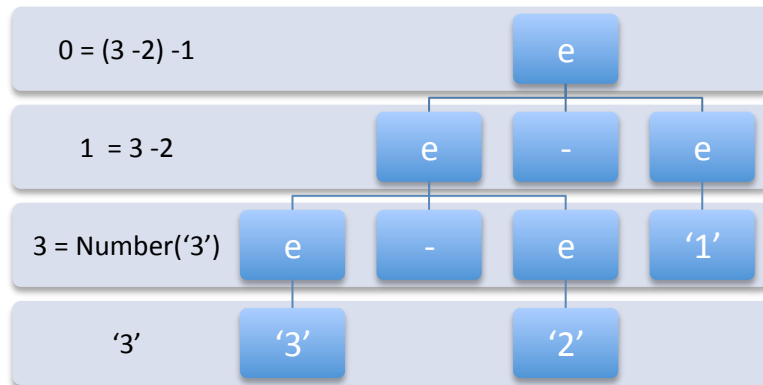
Conflicto Shift/Reduce

.3 - 2 - 1 <= e. - 2 - 1 <= = e -. 2 - 1 <= = e - 2. - 1 <= e - e. - 1

¿Qué hacer?

1. <= e. - 1 <= e - . 1 <= e - 1. <= e - e. <= e.
2. <= e - e - . 1 <= e - e - 1. <= e - e - e. <= e - e. <= e.

El conflicto puede verse como una lucha entre la regla $e \rightarrow e \text{ '-'}$ e y el terminal/token '-'



Un programa Yacc

%left ' _ ' ← En la lucha entre la regla $e \rightarrow e \text{ ' - '}$ e y el terminal/token ' - ' debe “ganar” la regla

%%

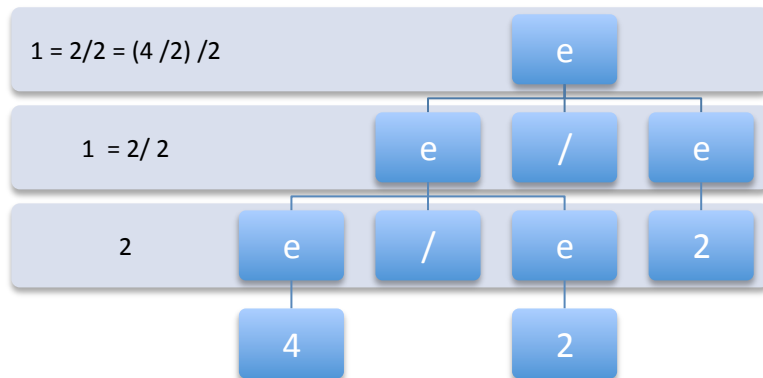
```
s : e      { return $1; }  
;
```

```
e : e ' _ ' e { $$ = $1 - $3; }  
  | NUM      { $$ = Number($1); }  
;
```

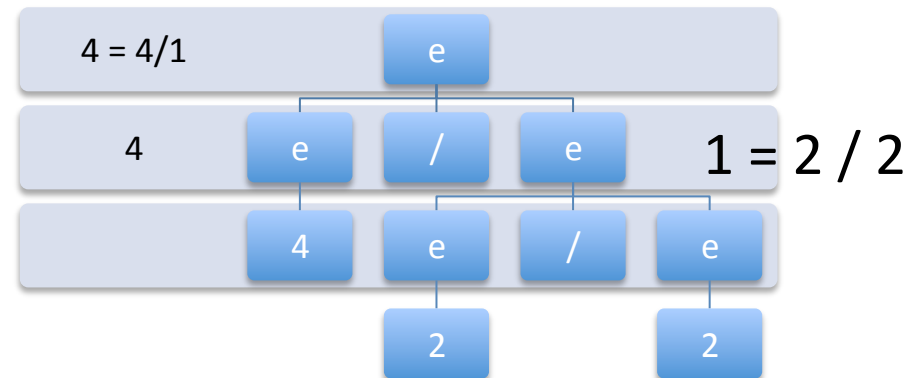
Ambigüedad: Asociatividad

$4/2/2$

$(4/2)/2$



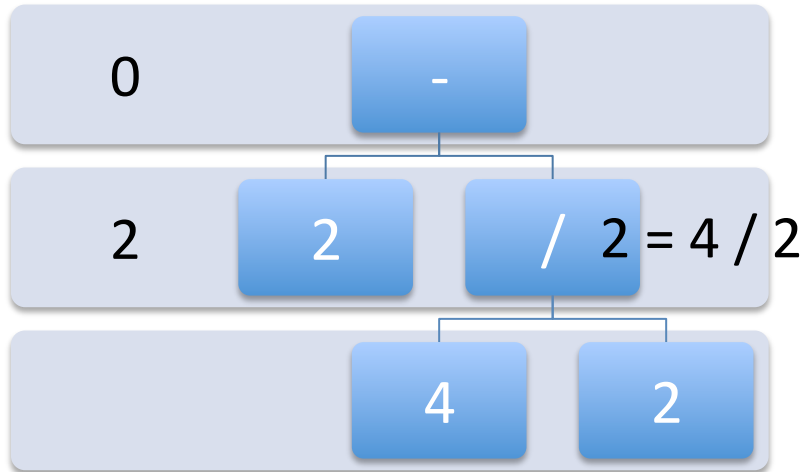
$4/(2/2)$



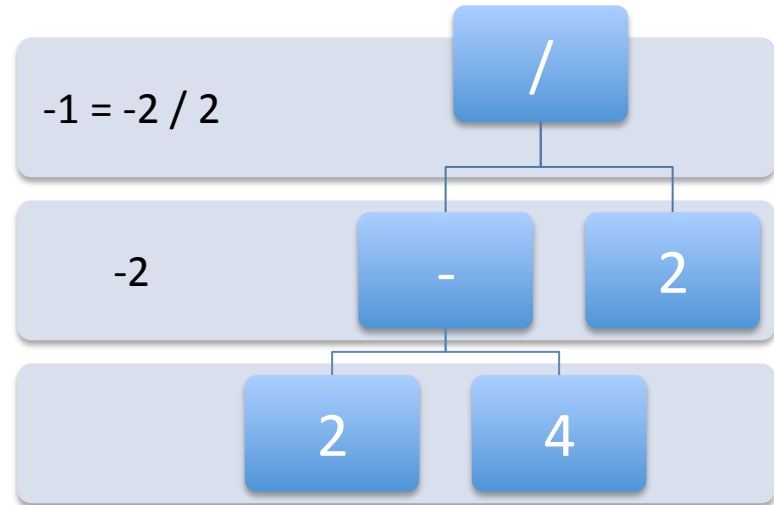
Ambigüedad: Prioridad

```
e : e ' _ ' e { $$ = $1 - $3; }  
  | e ' / ' e { $$ = $1 / $3; }  
  | NUM { $$ = Number($1); }  
  ;
```

2-(4/2)



(2-4)/2



Ambigüedad: Prioridad

2-4/2

2.-4/2 <= e.-4/2<=e-.4/2<=e-4./2<=e-e./2

¿Qué hacer?

1. <= e./2 <= e/. 2 <= e/e.<= e.
2. <= e-e/.2 <= e-e/2.<= e-e/e.<= e-e. <= e.

El conflicto es entre la regla $e \rightarrow e \text{ '-'}$ e y el terminal '/'

Ambigüedad: Prioridad



Mas prioridad

%left '-'

%left '/'

%%

e : e '-' e { \$\$ = \$1 - \$3; }

| e '/' e { \$\$ = \$1 / \$3; }

| NUM { \$\$ = Number(\$1); }

;

En la lucha entre reducir por la regla $e \rightarrow e - e$ y desplazar el terminal '/' debe "ganar" el token

Dynamic Resolution of Shift-Reduce Conflicts

Write a language that accepts lists of two kind of commands: *arithmetic expressions* like $4-2-1$ or one of two *commands*: *left* or *right*.

- When a *right* command is issued, the semantic of the '-' operator is changed to be right associative.
- When a *left* command is issued the semantic for '-' returns to left associative interpretation.

Dynamic Resolution of Shift-Reduce Conflicts

```
eyapp-examples — casiano@sanclemente-2:~/.../lsi-4-rpc-1819/casiano/eyapp-examples — -bash — 106x21
...vi .gitignore  ...les — -bash  ...on — -bash  ...ad — -bash  ...pp — -bash  ...as — -bash  ...to — -bash  ...ng — -bash  ...ng — -bash  ash  +

[~/.../lsi-4-rpc-1819/casiano/eyapp-examples(master)]$ cat input_for_dynamicgrammar.txt
2-1-1 # left: 0
RIGHT
2-1-1 # right: 2
LEFT
3-1-1 # left: 1
RIGHT
3-1-1 # right: 3
[~/.../lsi-4-rpc-1819/casiano/eyapp-examples(master)]$ eyapp -C dynamicgrammar.eyp
[~/.../lsi-4-rpc-1819/casiano/eyapp-examples(master)]$ ./dynamicgrammar.pm -f input_for_dynamicgrammar.txt

0
2
1
3
[~/.../lsi-4-rpc-1819/casiano/eyapp-examples(master)]$ █
```

Dynamic Resolution of Shift-Reduce Conflicts

```
eyapp-examples — casiano@sanclemente-2:~/.../lsi-4-rpc-1819/casiano/eyapp-examples — -bash — 130x32
...vi .gitignore ...les — -bash ...on — -bash ...ad — -bash ...pp — -bash ...as — -bash ...to — -bash ...ng — -bash ...ng — -bash ...les — -bash ...20 — -bash bash +

%whites /\s*(?:#.|*)?\s*/
%token NUM = /\d+/

%conflict leftORright {
    if ($reduce) { $self->YYSetReduce('-', ':M') } else { $self->YYSetShift('-') }
}

%expect 1

%%
p: c * {} ;

c:
    $expr { print "$expr\n" }
    | RIGHT { $reduce = 0 }
    | LEFT { $reduce = 1 }

;

expr:
    '(' $expr ')' { $expr }
    | %name :M
      expr.left          %PREC leftORright
        '-' expr.right   %PREC leftORright
        { $left - $right }

    | NUM

;

%%
[~/.../lsi-4-rpc-1819/casiano/eyapp-examples(master)]$
```

Recursos

- [Repositorio GitHub con los recursos de la charla: https://github.com/ULL-LSI/campus-america-2019](https://github.com/ULL-LSI/campus-america-2019)
- [Apuntes de Procesadores de Lenguajes. Curso 2018/2019: https://ull-esit-pl-1819.github.io/introduccion/](https://ull-esit-pl-1819.github.io/introduccion/)
- [Rodriguez-Leon, Casiano & Garcia-Forte, L. \(2011\). Solving Difficult LR Parsing Conflicts by Postponing Them. Comput. Sci. Inf. Syst.. 8. 517-531. 10.2298/CSIS101116008R.](#)
- [Parse Eyapp](#) en CPAN
- [Parsing Strings and Trees with Parse::Eyapp](#) (An Introduction to Compiler Construction). 2010