

Práctica PL

Elena Kaloyanova Popova y Álvaro Borja Velasco García

2018

Índice general

1. Fase 1: Analizador léxico	2
1.1. Clases Léxicas	2
1.2. Especificación Formal	3
1.3. Diseño	5
2. Fase 2: Analizador sintáctico	6
2.1. Gramática incontextual	6
2.1.1. Operadores	6
2.1.2. Gramática incontextual	6
2.1.3. Gramática transformada LL(1)	7
2.1.4. Primeros y siguientes	9
2.1.5. Directores	10
3. Fase 3: Analizador sintáctico ascendente	12

Capítulo 1

Fase 1: Analizador léxico

1.1. Clases Léxicas

Todo programa consta de dos secciones: una para las declaraciones y otra para las instrucciones, separadas por un token «&&». La sección de declaraciones está formada por una serie de declaraciones compuestas por el nombre de tipo y el de variable y separadas por un punto y coma. La sección de instrucciones, por su parte, consta de una serie de asignaciones (variable=expresión), separadas también por un punto y coma. Las clases léxicas que hemos considerado para representar los tokens del lenguaje son las siguientes:

- **SEC:** Representa el seccionador de las dos partes del programa («&&»).
- **NUM:** Palabra reservada «num».
- **BOOL:** Palabra reservada «bool».
- **VAR:** Representa el nombre de la variable. Comienza necesariamente por una letra, seguida por una secuencia de cero o más letras, dígitos o el símbolo «_».
- **ASIG:** Representa el signo igual de las asignaciones.
- **NXT:** Representa el signo «;» que marca el comienzo de la siguiente instrucción.
- **TRUE:** Palabra reservada «true».
- **FALSE:** Palabra reservada «false».

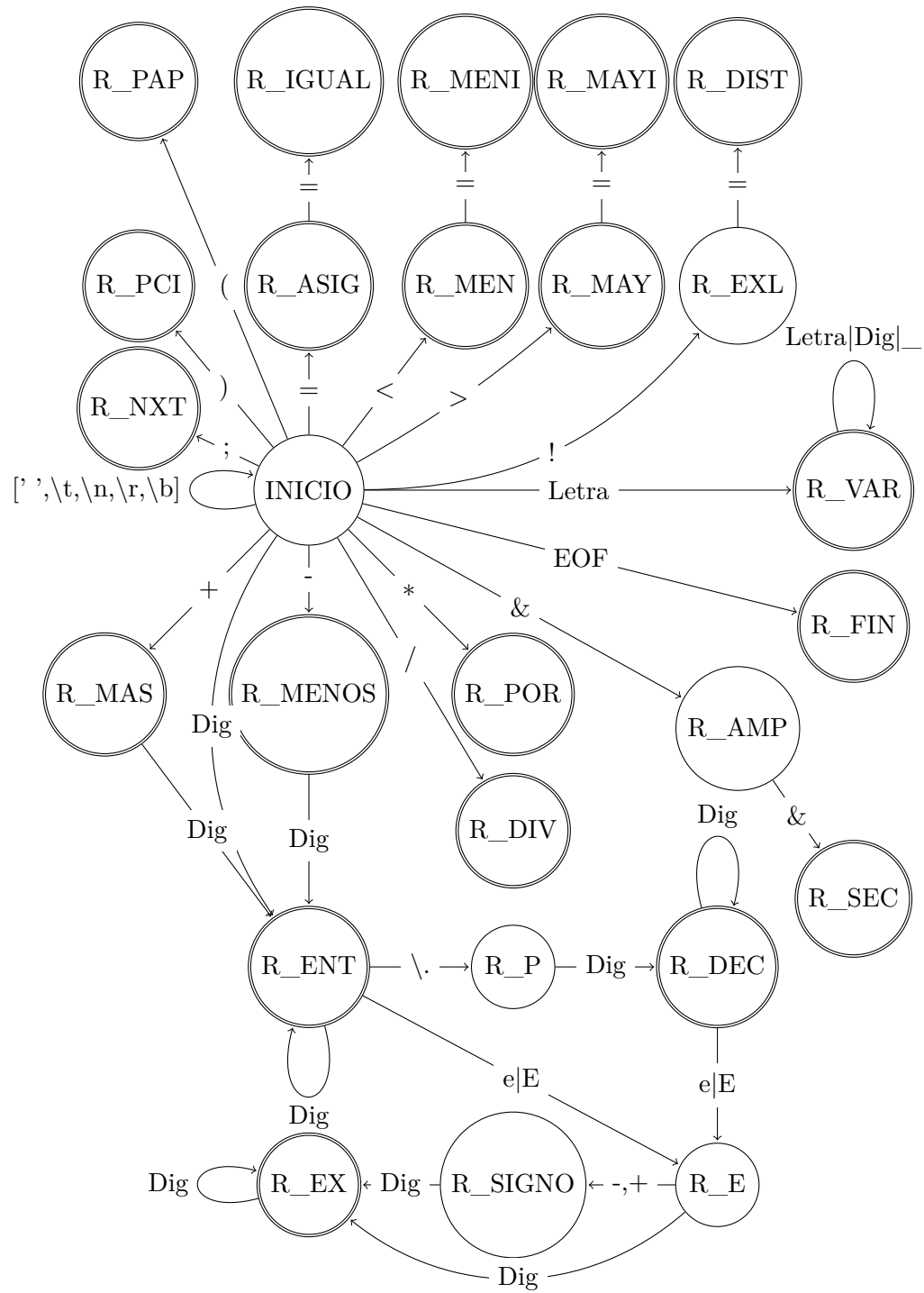
- **NUMR:** Representa un número real. Puede empezar opcionalmente con un signo seguido de una secuencia de uno o más dígitos cualesquiera, pudiendo poner ceros no significativos a la izquierda. Puede opcionalmente estar seguido por una parte decimal y/o una parte exponencial.
- **MAS:** Operador suma ($\backslash +$).
- **MENOS:** Operador resta ($\backslash -$).
- **POR:** Operador multiplicación ($\backslash *$).
- **DIV:** Operador división ($\backslash /$).
- **AND:** Palabra reservada «and».
- **OR:** Palabra reservada «or».
- **NOT:** Palabra reservada «not».
- **MAY:** Operador mayor ($>$).
- **MEN:** Operador menor ($<$).
- **MAYI:** Operador mayor o igual ($>=$).
- **MENI:** Operador menor o igual ($<=$).
- **IGUAL:** Operador igual a ($=$).
- **DIST:** Operador distinto a (\neq).
- **PAP:** Signo de apertura de paréntesis.
- **PCI:** Signo de cierre de paréntesis.
- **EOF:** Representa el final de fichero.

1.2. Especificación Formal

Las definiciones regulares correspondientes a las clases léxicas definidas son:

- (*) **SEC:** $[\&][\&]$
- (*) **VAR:** $\text{LETRA}([\text{LETRA}|\text{DIG}|\backslash_]*)$
 LETRA: $([a-z,A-Z])$

- DIG:** ([0-9])
 (★) **NUM:** ([n][u][m])
 (★) **BOOL:** ([b][o][o][l])
 (★) **TRUE:** ([t][r][u][e])
 (★) **FALSE:** ([f][a][l][s][e])
 (★) **NUMR:** SIGNO?(DIG+(DEC)?(EX)?)
 DEC: (\.)DIG+
 EX: [e|E](SIGNO?DIG+)
 SIGNO: [\+, \-]
 DIG: [0-9]
 (★) **AND:** ([a][n][d])
 (★) **OR:** ([o][r])
 (★) **NOT:** ([n][o][t])
 (★) **MAS:** (\+)
 (★) **MENOS:** (\-)
 (★) **DIV:** (/)
 (★) **POR:** (*)
 (★) **MAY:** (>)
 (★) **MEN:** (<)
 (★) **MAYI:** ([>][=])
 (★) **MENI:** ([<][=])
 (★) **IGUAL:** ([=][=])
 (★) **DIST:** (![=])
 (★) **ASIG:** (=)
 (★) **NXT:** (;)
 (★) **PAP:** (\()
 (★) **PCIERRE:** (\))
 (★) **SEP:** [« », \t, \n, \r, \b]



Capítulo 2

Fase 2: Analizador sintáctico

En esta fase desarrollaremos el analizador sintáctico descendente predictivo para el lenguaje descrito en la primera fase.

2.1. Gramática incontextual

2.1.1. Operadores

Empezaremos definiendo la gramática incontextual que define el lenguaje. Los operadores que utiliza nuestro lenguaje aparecen en la tabla 2.1.

Operador	Prioridad	Tipo	Asociatividad
+,-	0	Binario infijo	Asocia Izquierda
and a or	1	Binarios infijos	Asocia Derecha No asocia
Relacionales	2	Binario infijo	No asocia
*,/	3	Binario infijo	Asocia Izquierda
not	4	Unarios prefijos	Asocia No asocia

Cuadro 2.1: Operadores

2.1.2. Gramática incontextual

La gramática incontextual obtenida apartir de la definición y los operadores es la siguiente:

$S \rightarrow \text{Programa EOF}$

$\text{Programa} \rightarrow \text{LDs SEC LIs}$

LDs \rightarrow LDs NXT Declaracion
 LDs \rightarrow Declaracion
 Declaracion \rightarrow NUM VAR
 Declaracion \rightarrow BOOL VAR
 LIs \rightarrow LIs NXT Instruccion
 LIs \rightarrow Instruccion
 Instruccion \rightarrow VAR ASIG EXP0

EXP0 \rightarrow EXP0 OP0 EXP1
 EXP0 \rightarrow EXP1
 EXP1 \rightarrow EXP2 AND EXP1
 EXP1 \rightarrow EXP2 OR EXP2
 EXP1 \rightarrow EXP2
 EXP2 \rightarrow EXP3 OP2 EXP3
 EXP2 \rightarrow EXP3
 EXP3 \rightarrow EXP3 OP3 EXP4
 EXP3 \rightarrow EXP4
 EXP4 \rightarrow MENOS EXP4
 EXP4 \rightarrow NOT EXP5
 EXP5 \rightarrow NUMR
 EXP5 \rightarrow VAR
 EXP5 \rightarrow TRUE
 EXP5 \rightarrow FALSE
 EXP5 \rightarrow PAP EXP0 PCIERRE
 OP0 \rightarrow MAS
 OP0 \rightarrow MENOS
 OP2 \rightarrow MAY
 OP2 \rightarrow MEN
 OP2 \rightarrow MAYI
 OP2 \rightarrow MENI
 OP2 \rightarrow IGUAL
 OP2 \rightarrow DIST
 OP3 \rightarrow POR
 OP3 \rightarrow DIV

2.1.3. Gramática transformada LL(1)

Necesitamos transformar la gramática a una LL(1). Una vez transformada, la gramática queda de la siguiente manera:

S \rightarrow Programa EOF

Programa \rightarrow LDs SEC LIs
 LDs \rightarrow Declaracion RLDS
 RLDS \rightarrow NXT Declaracion RLDS
 RLDS $\rightarrow \varepsilon$
 Declaracion \rightarrow NUM VAR
 Declaracion \rightarrow BOOL VAR
 LIs \rightarrow Instruccion RLIS
 RLIS \rightarrow NXT Instruccion RLIS
 RLIS $\rightarrow \varepsilon$
 Instruccion \rightarrow VAR ASIG EXP0

EXP0 \rightarrow EXP1 R0
 R0 \rightarrow OP0 EXP1 R0
 R0 $\rightarrow \varepsilon$
 EXP1 \rightarrow EXP2 R1
 R1 \rightarrow AND EXP1
 R1 \rightarrow OR EXP2
 R1 $\rightarrow \varepsilon$
 EXP2 \rightarrow EXP3 R2
 R2 \rightarrow OP2 EXP3 R2
 R2 $\rightarrow \varepsilon$
 EXP3 \rightarrow EXP4 R3
 R3 \rightarrow OP3 EXP4 R3
 R3 $\rightarrow \varepsilon$
 EXP4 \rightarrow MENOS EXP4
 EXP4 \rightarrow NOT EXP5
 EXP4 \rightarrow EXP5
 EXP5 \rightarrow NUMR
 EXP5 \rightarrow VAR
 EXP5 \rightarrow TRUE
 EXP5 \rightarrow FALSE
 EXP5 \rightarrow PAP EXP0 PCIERRE
 OP0 \rightarrow MAS
 OP0 \rightarrow MENOS
 OP2 \rightarrow MAY
 OP2 \rightarrow MEN
 OP2 \rightarrow MAYI
 OP2 \rightarrow MENI
 OP2 \rightarrow IGUAL
 OP2 \rightarrow DIST

OP3 \rightarrow POR

OP3 \rightarrow DIV

2.1.4. Primeros y siguientes

Los **primeros** de nuestra gramática son:

PRIM(S) = {NUM, BOOL}

PRIM(PROGRAMA) = {NUM, BOOL}

PRIM(LDS) = {NUM, BOOL}

PRIM(RLDS) = {NXT}

PRIM(DECLARACION) = {NUM, BOOL}

PRIM(LIS) = {VAR}

PRIM(RLIS) = {NXT}

PRIM(INSTRUCCION) = {VAR}

PRIM(R0) = {MAS, MENOS}

PRIM(R1) = {AND, OR}

PRIM(EXP1) = {MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(EXP2) = {MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(R2) = {MAY, MEN, MAYI, MENI, IGUAL, DIST}

PRIM(EXP3) = {MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(R3) = {POR, DIV, MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(EXP4) = {MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(EXP5) = {NUMR, VAR, TRUE, FALSE, PAP}

PRIM(EXP0) = {MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP}

PRIM(OP0) = {MAS, MENOS}

PRIM(OP2) = {MAY, MEN, MAYI, MENI, IGUAL, DIST}

PRIM(OP3) = {POR, DIV}

Los **siguientes** son:

SIG(S) = $\{\varepsilon\}$

SIG(PROGRAMA) = {EOF}

SIG(LDS) = {SEC}

SIG(RLDS) = {SEC}

SIG(DECLARACION) = {NXT, SEC}

SIG(LIS) = {EOF}

SIG(RLIS) = {EOF}

SIG(INSTRUCCION) = {NXT, EOF}

SIG(R0) = {PCI, NXT, EOF}

SIG(R1) = {PCI, MAS, MENOS, NXT, EOF}

SIG(EXP1) = {PCI, MAS, MENOS, NXT, EOF}

$SIG(EXP2) = \{PCI, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(R2) = \{PCI, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(EXP3) = \{PCI, MAY, MEN, MAYI, MENI, IGUAL, DIST, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(R3) = \{PCI, MAY, MEN, MAYI, MENI, IGUAL, DIST, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(EXP4) = \{PCI, POR, DIV, NOT, NUMR, VAR, TRUE, FALSE, PAP, MAY, MEN, MAYI, MENI, IGUAL, DIST, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(EXP5) = \{PCI, POR, DIV, NOT, NUMR, VAR, TRUE, FALSE, PAP, MAY, MEN, MAYI, MENI, IGUAL, DIST, AND, OR, MAS, MENOS, NXT, EOF\}$
 $SIG(EXP0) = \{PCI, NXT, EOF\}$
 $SIG(OP0) = \{MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP\}$
 $SIG(OP2) = \{MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP\}$
 $SIG(OP3) = \{MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP\}$

2.1.5. Directores

Obtenidos los primeros y los siguientes podemos proceder a calcular los **directores**:

$DIR(S \rightarrow Programa \underline{EOF}) = \{NUM, BOOL\}$
 $DIR(Programa \rightarrow LDs \underline{SEC} \underline{LIs} \underline{EOF}) = \{NUM, BOOL\}$
 $DIR(LDs \rightarrow Declaracion \underline{RLDS}) = \{NUM, BOOL\}$
 $DIR(RLDS \rightarrow \underline{NXT} Declaracion \underline{RLDS}) = \{NXT\}$
 $DIR(RLDS \rightarrow \epsilon) = \{SEC\}$
 $DIR(Declaracion \rightarrow \underline{NUM} \underline{VAR}) = \{NUM\}$
 $DIR(Declaracion \rightarrow \underline{BOOL} \underline{VAR}) = \{BOOL\}$
 $DIR(LIs \rightarrow Instruccion \underline{RLIS}) = \{VAR\}$
 $DIR(RLIS \rightarrow \underline{NXT} Instruccion \underline{RLDS}) = \{NXT\}$
 $DIR(RLIS \rightarrow \epsilon) = \{EOF\}$
 $DIR(Instruccion \rightarrow \underline{VAR} \underline{ASIG} EXP0) = \{VAR\}$
 $DIR(EXP0 \rightarrow EXP1 R0) = \{MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP\}$
 $DIR(R0 \rightarrow OP0 EXP1 R0) = \{MAS, MENOS\}$
 $DIR(R0 \rightarrow \epsilon) = \{PCI, NXT, EOF\}$
 $DIR(EXP1 \rightarrow EXP2 R1) = \{MENOS, NOT, NUMR, VAR, TRUE, FALSE, PAP\}$
 $DIR(R1 \rightarrow \underline{AND} EXP1 R1) = \{AND\}$
 $DIR(R1 \rightarrow \underline{OR} EXP2) = \{OR\}$

$\text{DIR}(\text{R1} \rightarrow \varepsilon) = \{\text{PCI}, \text{MAS}, \text{MENOS}, \text{NXT}, \text{EOF}\}$
 $\text{DIR}(\text{EXP2} \rightarrow \text{EXP3 R2}) = \{\text{MENOS}, \text{NOT}, \text{NUMR}, \text{VAR}, \text{TRUE}, \text{FALSE}, \text{PAP}\}$
 $\text{DIR}(\text{R2} \rightarrow \text{OP2 EXP3 R2}) = \{\text{MAY}, \text{MEN}, \text{MAYI}, \text{MENI}, \text{IGUAL}, \text{DIST}\}$
 $\text{DIR}(\text{R2} \rightarrow \varepsilon) = \{\text{PCI}, \text{AND}, \text{OR}, \text{MAS}, \text{MENOS}, \text{NXT}, \text{EOF}\}$
 $\text{DIR}(\text{EXP3} \rightarrow \text{EXP4 R3}) = \{\text{MENOS}, \text{NOT}, \text{NUMR}, \text{VAR}, \text{TRUE}, \text{FALSE}, \text{PAP}\}$
 $\text{DIR}(\text{R3} \rightarrow \text{OP3 EXP4 R3}) = \{\text{MUL}, \text{DIV}\}$
 $\text{DIR}(\text{R3} \rightarrow \varepsilon) = \{\text{PCI}, \text{MAY}, \text{MEN}, \text{MAYI}, \text{MENI}, \text{IGUAL}, \text{DIST}, \text{AND}, \text{OR}, \text{MAS}, \text{MENOS}, \text{NXT}, \text{EOF}\}$
 $\text{DIR}(\text{EXP4} \rightarrow \underline{\text{MENOS}} \text{ EXP4}) = \{\text{MENOS}\}$
 $\text{DIR}(\text{EXP4} \rightarrow \underline{\text{NOT}} \text{ EXP5}) = \{\text{NOT}\}$
 $\text{DIR}(\text{EXP4} \rightarrow \text{EXP5}) = \{\text{NUMR}, \text{VAR}, \text{TRUE}, \text{FALSE}, \text{PAP}\}$
 $\text{DIR}(\text{EXP5} \rightarrow \underline{\text{NUMR}}) = \{\text{NUMR}\}$
 $\text{DIR}(\text{EXP5} \rightarrow \underline{\text{VAR}}) = \{\text{VAR}\}$
 $\text{DIR}(\text{EXP5} \rightarrow \underline{\text{TRUE}}) = \{\text{TRUE}\}$
 $\text{DIR}(\text{EXP5} \rightarrow \underline{\text{FALSE}}) = \{\text{FALSE}\}$
 $\text{DIR}(\text{EXP5} \rightarrow \underline{\text{PAP}} \text{ EXP0 } \underline{\text{PCIERRE}}) = \{\text{PAP}\}$
 $\text{DIR}(\text{OP0} \rightarrow \underline{\text{MAS}}) = \{\text{MAS}\}$
 $\text{DIR}(\text{OP0} \rightarrow \underline{\text{MENOS}}) = \{\text{MENOS}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{MAY}}) = \{\text{MAY}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{MEN}}) = \{\text{MEN}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{MAYI}}) = \{\text{MAYI}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{MENI}}) = \{\text{MENI}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{IGUAL}}) = \{\text{IGUAL}\}$
 $\text{DIR}(\text{OP2} \rightarrow \underline{\text{DIST}}) = \{\text{DIST}\}$
 $\text{DIR}(\text{OP3} \rightarrow \underline{\text{POR}}) = \{\text{POR}\}$
 $\text{DIR}(\text{OP3} \rightarrow \underline{\text{DIV}}) = \{\text{DIV}\}$

Capítulo 3

Fase 3: Analizador sintáctico ascendente

En esta fase desarrollaremos una versión diferente del analizador sintáctico, esta vez ascendente LR. Se implementará con JLex y Cup haciendo uso de la gramática incontextual desarrollada en la fase 2.