INFO-F403 Introduction to Language Theory and Compilation

Chapeaux Thomas
Dagnely Pierre

February 27, 2013

Lexical units	regular expressions
INT	([0-9])*
FLOAT	([0-9])*.DOT.([0-9])*
BOOL	(0+1+true+false+”)
STRING	’.([A-Za-z]+[0-9])*.’
FAC	!
MUL	*
DIV	/
MINUS	-
ADD	+
LT	<
GT	>
LE	<=
GE	>=
EQUIV	==
DIF	!=
AND	&&
OR	||
NOT	not
LT-S	lt
GT-S	gt
LE-S	le
GE-S	ge
EQ-S	eq
NE-S	ne

Lexical units	regular expressions
EQUAL	=
DOT	.
SEMICOLON	;
COMA	,
OPEN-PAR	(
CLOSE-PAR	)
OPEN-BRAC	{
CLOSE-BRAC	}
OPEN-COND	IF
CLOSE-COND	ELSE
ADD-COND	ELSE IF
NEG-COND	UNLESS
RET	return
FUNCT-DEF	SUB
ID	STRING
FUNCT-CALL	&.STRING
PERL-DEF	defined
PERL-INT	int
PERL-LENG	length
PERL-SCAL	scalar
PERL-SUBS	substr
PERL-PRIN	print
COMM	#.STRING
VARIABLE	$.STRING

coma peut définir l’opérateur coma ou juste un coma entre deux param, mais même lexical unit, c’est le parser qui se charge du reste

DFA

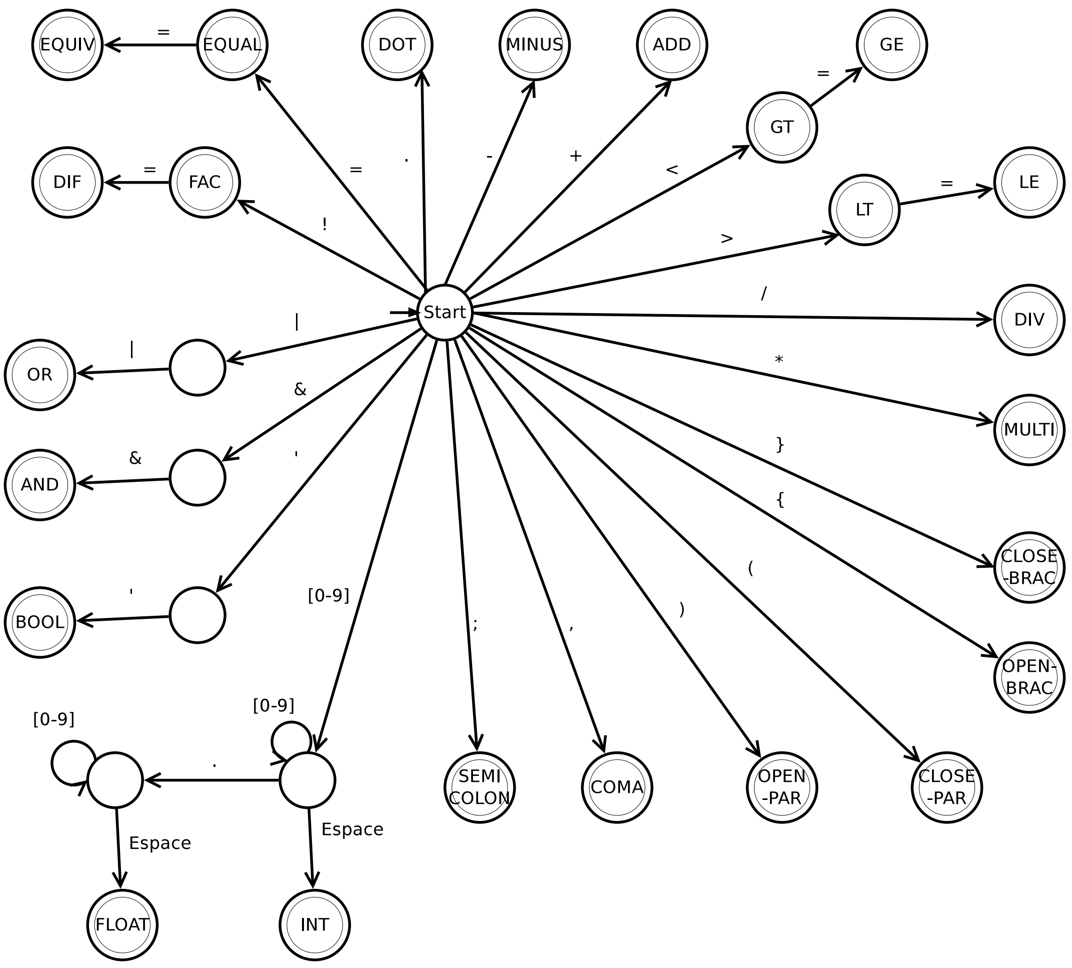

Figure 1: automate ”non alphabétique”

La plupart des noeuds pointent vers le token ID, trop lourd a représenter, donc met une petite fleche bleu à la place.

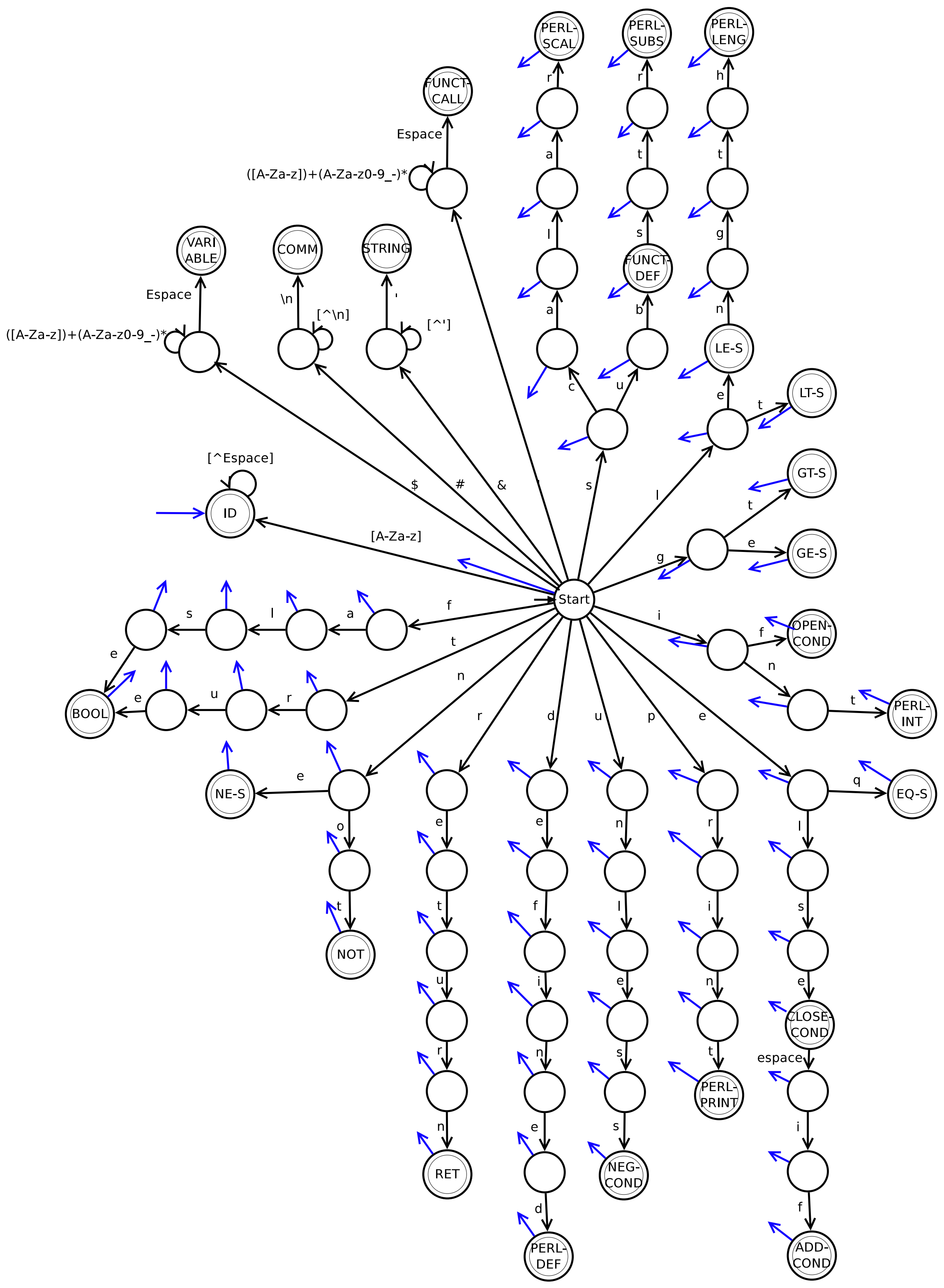

Figure 2: automate "alphabétique"

Grammar

VALUE	→ INT
	→ FLOAT
	→ BOOL
	→ STRING
OPERATOR	→ FAC
	→ MUL
	→ DIV
	→ MINUS
	→ CONC
	→ ADD
OPERATOR-COMP	→ LT
	→ GT
	→ LE
	→ GE
	→ EQUIV
	→ DIF
	→ AND-LOGIC
	→ OR
	→ NOT
	→ LT-S
	→ GT-S
	→ LE-S
	→ GE-S
	→ COMA-LOGIC
	→ EQ-S
	→ NE-S
EXPRESSION	→ VARIABLE
	→ EXPRESSION OPERATOR EXPRESSION
	→ EXPRESSION-COMP
EXPRESSION-COMP	→ EXPRESSION OPERATOR-COMP EXPRESSION
ASSIGNATION	→ VARIABLE EQUAL VALUE
	→ VARIABLE EQUAL EXPRESSION
CONDITION	→ OPEN-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS CLOSE-
	→ NEG-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS CLOSE-B
	→ EXPRESSION OPEN-COND EXPRESSION-COND
	→ EXPRESSION NEG-COND EXPRESSION-COND
CONDITION-END	→ ADD-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS CLOSE-B
	→ ADD-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS CLOSE-B
	→ CLOSE-COND OPEN-BRAC INSTRUCTIONS CLOSE-BRAC
	→ EPSILON
INSTRUCTIONS	→ CONDITION SEMICOLON INSTRUCTIONS
	→ EXPRESSION SEMICOLON INSTRUCTIONS
	→ FUNCT-CALL SEMICOLON INSTRUCTIONS
	→ ASSIGNATION SEMICOLON INSTRUCTIONS

	→ CONDITION SEMICOLON
	→ EXPRESSION SEMICOLON
	→ FUNCT-CALL SEMICOLON
	→ ASSIGNATION SEMICOLON
	→ EPSILON
PARAM	→ DOLLAR VARIABLE
	→ DOLLAR VARIABLE PARAM-END
	→ EPSILON
PARAM-END	→ COMA DOLLAR VARIABLE
	→ COMA DOLLAR VARIABLE PARAM-END
	→ EPSILON
USER-FUNCT-CALL	→ AND FUNCT-NAME OPEN-PAR CLOSE-PAR
	→ AND FUNCT-NAME OPEN-PAR PARAM CLOSE-PAR
	→ AND FUNCT-NAME PARAM
	→ AND FUNCT-NAME
LIST	→ STRING
	→ STRING LIST
	→ EPSILON
PERL-FUNCT-CALL	→ PERL-DEF EXPRESSION
	→ PERL-INT EXPRESSION
	→ PERL-LENG EXPRESSION
	→ PERL-SCAL EXPRESSION
	→ PERL-SUBS EXPRESSION COMA INT COMA INT
	→ PERL-SUBS EXPRESSION COMA INT
	→ PERL-PRIN LIST
FUNCTION-CALL	→ USER-FUNCT-CALL
	→ PERL-FUNCT-CALL
FUNCTION	→ FUNCT-ID FUNCT-NAME OPEN-BRAC INSTRUCTIONS RETURN CLOSE
	→ FUNCT-ID FUNCT-NAME OPEN-PAR CLOSE PAR OPEN-BRAC INSTRUC
	→ FUNCT-ID FUNCT-NAME OPEN-PAR PARAM CLOSE-PAR OPEN-BRAC I
RETURN	→ RET EXPRESSION SEMICOLON
	→ RET EXPRESSION-COND SEMICOLON
	→ RET VARIABLE SEMICOLON
	→ EPSILON
FUNCTION-LIST	→ FUNCTION
	→ FUNCTION FUNCTION-LIST
	→ EPSILON
PROGRAM	→ PROGRAM FUNCTION-LIST
	→ PROGRAM INSTRUCTIONS
	→ FUNCTION-LIST
	→ INSTRUCTIONS
	→ EPSILON

EXPRESSION (?)	VARIABLE OPERATOR VARIABLE EXPRESSION OPERATOR VARIABLE
EXPRESSION-COND (?)	VARIABLE OPERATOR-COMP VARIABLE EXPRESSION OPERATOR-COMP VARIABLE
ASSIGNATION	VARIABLE EQUAL VALUE
CONDITION (?)	((OPEN-COND+NEG-COND)EXPRESSION-COND OPEN-BRAC INSTRUCTIONS* CLOSE-BRAC (ADD-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS* CLOSE-BRAC)* (CLOSE-COND EXPRESSION-COND OPEN-BRAC INSTRUCTIONS* CLOSE-BRAC)) + EXPRESSION (OPEN-COND + NEG-COND) EXPRESSION-COND
INSTRUCTIONS	((CONDITION SEMICOLON)* + (EXPRESSION SEMICOLON)* + (FUNCTION-CALL SEMICOLON)* + (ASSIGNATION SEMICOLON)*)*
PARAM	DOLLAR VARIABLE (COMA DOLLAR VARIABLE)*
USER-FUNCT-CALL	AND FUNCTION-NAME (OPEN-PAR CLOSE-PAR + OPEN-PAR PARAM CLOSE-PAR + PARAM) SEMICOLON
PERL-FUNCT-CALL	defined EXPRESSION + int EXPRESSION + length EXPRESSION scalar EXPRESSION + substr EXPRESSION COMA INT COMA INT scalar EXPRESSION + substr EXPRESSION COMA INT + print (?liste de string)
FUNCTION-CALL	USER-FUNCT-CALL + PERL-FUNCT-CALL
FUNCTION	FUNCTION-ID FUNCTION-NAME (OPEN-PAR CLOSE PAR + OPEN-PAR PARAM CLOSE-PAR) OPEN-BRAC INSTRUCTIONS (RETURN EXPRESSION + RETURN EXPRESSION-COND + RETURN VARIABLE) SEMICOLON CLOSE-BRAC
FUNCTION-LIST	FUNCTION*
PROGRAM	(FUNCTION-LIST + INSTRUCTIONS)*

(slide 13)