

Homework Series 1

Antoine Veenstra s1378791

May 20, 2015

1

Some parsers:

1.1 ANTLR4

Written in Java

Target Language C#, Java, Python

Algorithms LL

Source <https://theantlrguy.atlassian.net/wiki/display/ANTLR4/>

1.2 JFLAP

Written in Java

Target Language Java

Algorithms LL1 and LALR1

Source <http://en.wikipedia.org/wiki/Compiler-compiler>

1.3 Parsec

Written in Haskell

Target Language Haskell

Algorithms LL and Backtracking

Source Lennart

1.4 PLY

Written in Python

Target Language Python

Algorithms LALR1

Source Experience with this parser generator

1.5 Racc

Written in Ruby

Target Language Ruby

Algorithms LALR1

Source <https://github.com/tenderlove/racc>

1.6 Bison

Written in C

Target Language C, C++, Java

Algorithms LALR1, LR1, IELR1, GLR

Source <http://www.gnu.org/software/bison/>

1.7 Yapps

Written in Python

Target Language Python

Algorithms LL1

Source Experience with this parser generator

1.8 Citrus

Written in Ruby

Target Language Ruby

Algorithms PEG

Source <https://github.com/mjackson/citrus>

1.9 Jison

Written in Javascript

Target Language Javascript

Algorithms LALR1, LR0, SLR1, LR1, LL1

Source <https://zaach.github.io/jison/docs/>

1.10 Happy

Written in Haskell

Target Language Haskell

Algorithms Generalized LR

Source <https://www.haskell.org/happy/#what>

2

2.1

literal	int	long
binary	$/0b[0-1]^+ /$	$/0b[0-1]^+(l-L) /$
octal	$/0(0 [1-7][0-7])^+ /$	$/0(0 [1-7][0-7])^+(l-L) /$
hexadecimal	$/0x[0-9a-f]^+ /$	$/0x[0-9a-f]^+(l/L) /$
decimal	$/(0 [1-9][0-9]^*) /$	$/(0 [1-9][0-9]^* (l/L)) /$

2.2

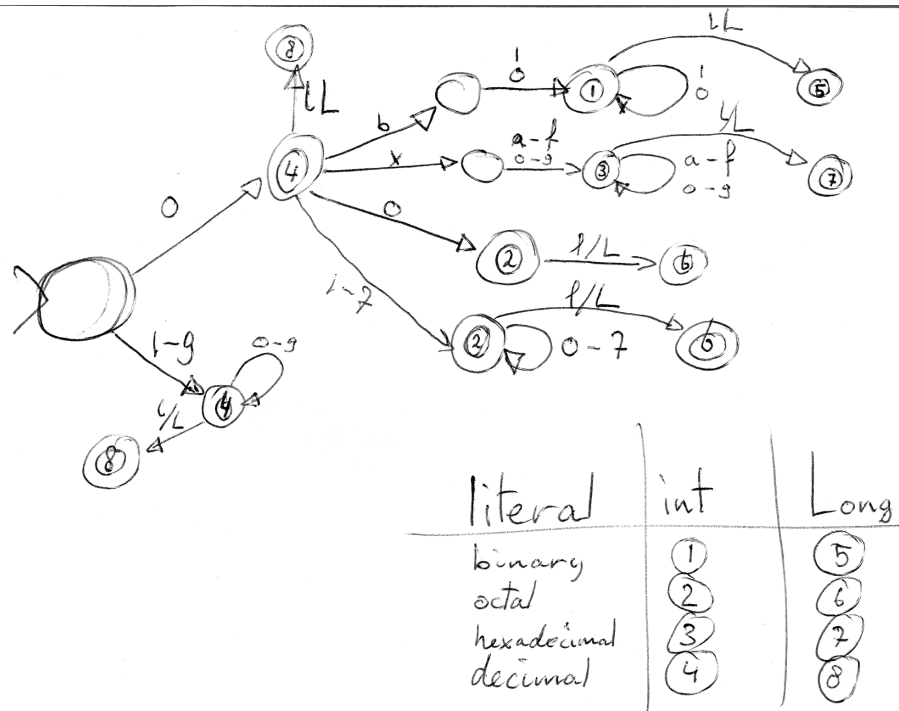


Figure 1: A DFA combining the token types

2.3

See the code in `pp.s1378791.q1_2.Numbers.g4` and `pp.s1378791.q1_2.NumbersTest`

3

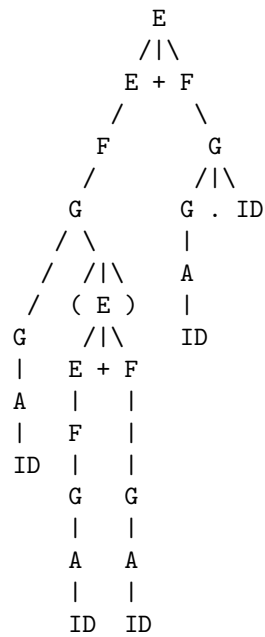
3.1

3.1.1 `a[i+1] + b.field`

E

E + F

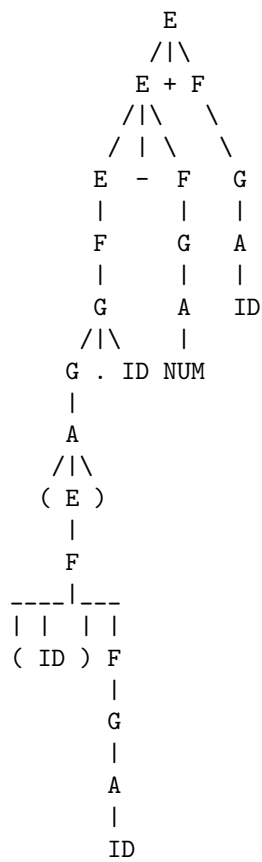
$F + F$
 $G + F$
 $G [E] + F$
 $A [E] + F$
 $ID [E + F] + F$
 $ID [F + F] + F$
 $ID [G + F] + F$
 $ID [A + F] + F$
 $ID [ID + F] + F$
 $ID [ID + G] + F$
 $ID [ID + A] + F$
 $ID [ID + NUM] + F$
 $ID [ID + NUM] + G$
 $ID [ID + NUM] + G . ID$
 $ID [ID + NUM] + A . ID$
 $ID [ID + NUM] + ID . ID$



3.1.2 ((Type) x).i - 10 + y

E
 $E + F$
 $E - F + F$
 $F - F + F$
 $G - F + F$
 $G . ID - F + F$
 $A . ID - F + F$
 $(E) . ID - F + F$
 $(F) . ID - F + F$
 $((ID) F) . ID - F + F$

((ID) G) . ID - F + F
 ((ID) A) . ID - F + F
 ((ID) ID) . ID - F + F
 ((ID) ID) . ID - G + F
 ((ID) ID) . ID - A + F
 ((ID) ID) . ID - NUM + F
 ((ID) ID) . ID - NUM + G
 ((ID) ID) . ID - NUM + A
 ((ID) ID) . ID - NUM + ID



3.2

See pp.s1378791.q1_3

3.3

E -> F E'
 E' -> "+" F E'
 | "-" F E'
 | epsilon
 F -> "(" ID ")" F
 | G

```

G  -> A G'
G' -> "[" E "]" G'
    | "." ID G'
    | epsilon
A  -> "(" E ")"
    | NUM
    | ID

```

3.4

3.4.1 FIRST

	1	2	3	4
E	\emptyset	(((, num, id
E'	+, -, epsilon	+, -, epsilon	+, -, epsilon	+, -, epsilon
F	(((, num, id	(, num, id
G	\emptyset	(, num, id	(, num, id	(, num, id
G'	[, ., epsilon	[, ., epsilon	[, ., epsilon	[, ., epsilon
A	(, num, id	(, num, id	(, num, id	(, num, id

3.4.2 FOLLOW

-	init	1	2
E	eof	eof],)	eof,],)
E'	\emptyset	eof],)	eof,],)
F	\emptyset	eof],), +, -	eof,],), +, -
G	\emptyset	eof],), +, -	eof,],), +, -
G'	\emptyset	eof],), +, -	eof,],), +, -
A	\emptyset	eof],), +, -, [, .	eof,],), +, -, [, .

3.4.3 FIRST+

Regel	first	follow	first+
$E \rightarrow FE'$	(, num, id	eof,],)	(, num, id
$E' \rightarrow "+" FE'$	+	eof,],)	+
$E' \rightarrow "-" FE'$	-	eof,],)	-
$E' \rightarrow \text{epsilon}$	epsilon	eof,],)	eof,],)
$F \rightarrow "(" ID ")" F$	(eof,],), +, -	(
$F \rightarrow G$	num, id	eof,],), +, -	(, num, id
$G \rightarrow AG'$	num, id	eof,],), +, -	(, num, id
$G' \rightarrow "[" E "]" G'$	[eof,],), +, -	[
$G' \rightarrow "." ID G'$.	eof,],), +, -	.
$G' \rightarrow \text{epsilon}$	epsilon	eof,],), +, -	eof, +, -,],)
$A \rightarrow "(" E ")"$	(eof,],), +, -, [, .	(
$A \rightarrow NUM$	num	eof,],), +, -, [, .	num
$A \rightarrow ID$	id	eof,],), +, -, [, .	id

4

5

See lab files in `pp.s1378791.q1_5`