Exercises About LL(K) Grammars

Compilers course

Masters in Informatics and Computing Engineering (MIEIC), 3rd Year

João M. P. Cardoso

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Dep. de Engenharia Informática
Faculdade de Engenharia (FEUP)
Universidade do Porto,
Porto, Portugal
Email:jmpc@acm.org

Exercise 1

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid NUM$$

- **Group 1. *First*, *Follow* Sets and *Parsers* LL(k) (5 pts)**
- Consider the CFG1 in the left. **NUM** is a terminal symbol representing numbers.
- **1.a)** [1pt] Give the *First* and *Follow* sets for each grammar variable;
- **1.b)** [2pts] Is the grammar LL(1)? Show the table for the *parser* LL(1);
- **1.c)** [2pts] Modify the grammar in order it can be implemented as a top-down recursive parser with K=1 (*lookahead*).

Source: First midterm exam (MT1): April 6, 2017

Exercise 1

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid NUM$

- **1.a)** [1pt] Give the *First* and *Follow* sets for each grammar variable;
- First(E)={(,NUM}
- First(T)={(,NUM}
- First(F)={(,NUM}

- Follow(E)={+,)}
- Follow(T)={*,+,)}
- Follow(F)={*,+,)}

Source: First midterm exam (MT1): April 6, 2016

Note that here you need to determine the First of each production (not the first of each variable) and the follow of the variables that have empty productions.

Exercise 1

- **1.b)** [2pts] Is the grammar LL(1)? Show the table for the *parser* LL(1);
  - The grammar is not LL(1), there are cases with more than one production for a given input symbol as can be seen below

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid NUM$$

	+	*	(	)	NUM
E			E → E + T E → T		E → E + T E → T
T			T → T * F T → F		T → T * F T → F
F			F → (E)		F → NUM

Source: First midterm exam (MT1): April 6, 2016

What we need?

- Eliminate ambiguity
- Eliminate left recursion
- Do left factorization (may need to consider modifications of productions across different variables)

Exercise 1

- **1.c)** [2pts] Modify the grammar in order it can be implemented as a top-down recursive parser with K=1 (*lookahead*).

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid NUM$$

The grammar is non-ambiguous

Source: First midterm exam (MT1): April 6, 2016

What we need?

- Eliminate ambiguity
- Eliminate left recursion
- Do left factorization (may need to consider modifications of productions across different variables)

Exercise 1

- **1.c)** [2pts] Modify the grammar in order it can be implemented as a top-down recursive parser with K=1 (*lookahead*).

$E \rightarrow E + T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid NUM$

The grammar is non-ambiguous

1st: eliminate left recursion

$E \rightarrow T + E \mid T$

$T \rightarrow F * T \mid F$

$F \rightarrow (E) \mid NUM$

2nd: eliminate for each variable the existence of productions with First sets with one or more equal symbols

$E \rightarrow T\ E1$

$E1 \rightarrow + E \mid \varepsilon$

$T \rightarrow F\ T1$

$T1 \rightarrow * T \mid \varepsilon$

$F \rightarrow (E) \mid NUM$

$E \rightarrow T\ [+ E]$

$T \rightarrow F\ [* T\ ]$

$F \rightarrow (E) \mid NUM$

Source: First midterm exam (MT1): April 6, 2016

Exercise 2

- **1d)** Show the grammar is LL(1) by using the table for the *parser* LL(1)

E → T E1

E1 → + E | ε

T → F T1

*T1 → * T | ε*

F → (E) | NUM

Exercise 2

- **1d)** Show the grammar is LL(1) by using the table for the *parser* LL(1)
  - The table presented below has at most one production per cell and thus the grammar is LL(1)

E → T E1

E1 → + E | ε

T → F T1

*T1 → * T | ε*

F → (E) | NUM

	+	*****	**(**	**)**	**NUM**
E			E → T E1		E → T E1
E1	E1 → + E				
T			T → F T1		T → F T1
T1	T1 → ε	T1 → * T			
F			F → (E)		F → NUM

Source: First midterm exam (MT1): April 6, 2016

Exercise 2

E → T E1

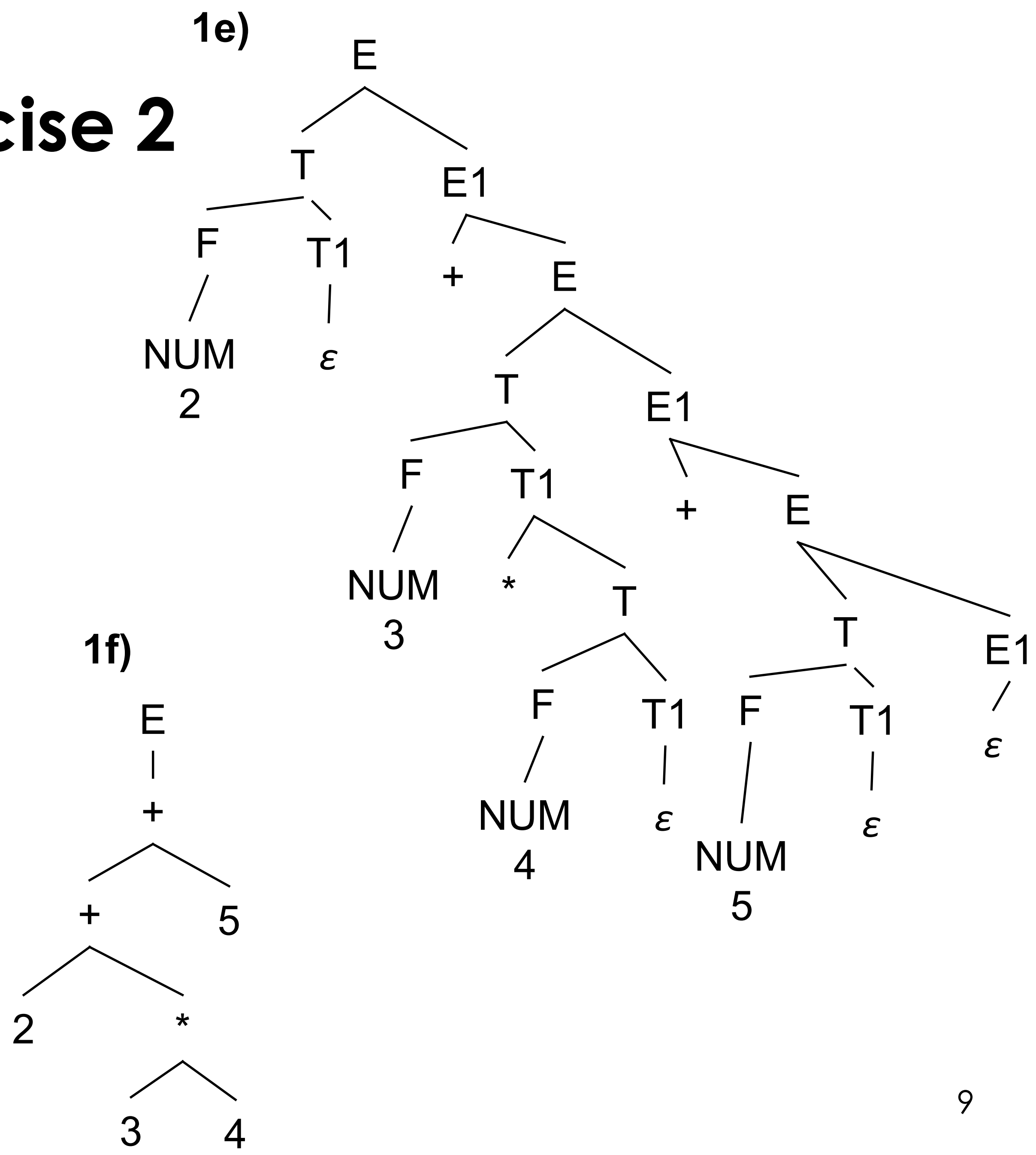
E1 → + E | ε

T → F T1

*T1 → * T | ε*

F → (E) | NUM

- **1e)** Show the concrete syntax tree (CST) for 2+3*4+5
- **1f)** Show a possible abstract syntax tree (AST) for 2+3*4+5


1e)
E
T
E1
F
T1
+
E
NUM
2
ε
T
E1
F
T1
+
E
NUM
3
*
T
T
E1
F
T1
F
T1
ε
NUM
4
ε
NUM
5
ε
1f)
E
+
+
5
2
*
3
4