

The Exercises of The Chapter Three

- 2.12 a. Use Thompson's construction to convert the regular expression $(a|b)^*a(a|b)\epsilon$ into an NFA.
 b. Convert the NFA of part (a) into a DFA using the subset construction.

同作业一 2.1.2

2.12 a. Use Thompson's construction to convert the regular expression $(a|b)^*a(a|b)\epsilon$ into an NFA.

b. Convert the NFA of part (a) into a DFA using the subset construction.

$(a|b)^*a(a|b)\epsilon$

a.

b.

	I	I_a	I_b
$\rightarrow \{0, 1, 2\}$	0	$\{1, 3, 4\}$	1
$\{1, 3, 4\}$	1	$\{1, 4\}$	3
$\{1\}$	2	$\{1\}$	2
$\{1, 4\}$	3	$\{1\}$	2

3.3 Given the grammar

$exp \rightarrow exp \text{ addop } term \mid term$

$addop \rightarrow + \mid -$

$term \rightarrow term \text{ mulop } factor \mid factor$

$mulop \rightarrow *$

$factor \rightarrow (exp) \mid number$

3 add 4 mu 5 add 6

Write down leftmost derivations, parse trees, and abstract syntax trees for the following expression:

- a. $3+4*5-6$ b. $3*(4-5+6)$ c. $3-(4+5*6)$

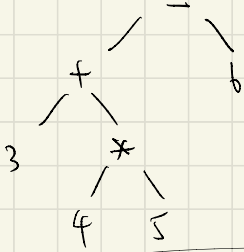
a.

parse tree

leftmost derivation

$exp \Rightarrow exp \text{ addop } term \Rightarrow exp \text{ addop } term$
 $\Rightarrow term \text{ addop } term \text{ addop } term$
 $\Rightarrow factor \text{ addop } term \text{ addop } term$
 $\Rightarrow number \text{ addop } term \text{ addop } term$
 $\Rightarrow 3 \text{ addop } term \text{ addop } term$
 $\Rightarrow 3 + term \text{ addop } term$
 $\Rightarrow 3 + term \text{ mulop } factor \text{ addop } term$
 $\Rightarrow 3 + factor \text{ mulop } factor \text{ addop } term$
 $\Rightarrow 3 + number \text{ mulop } factor \text{ addop } term$
 $\Rightarrow 3 + 4 \text{ mulop } factor \text{ addop } term$
 $\Rightarrow 3 + 4 * factor \text{ addop } term$
 $\Rightarrow 3 + 4 * number \text{ addop } term$

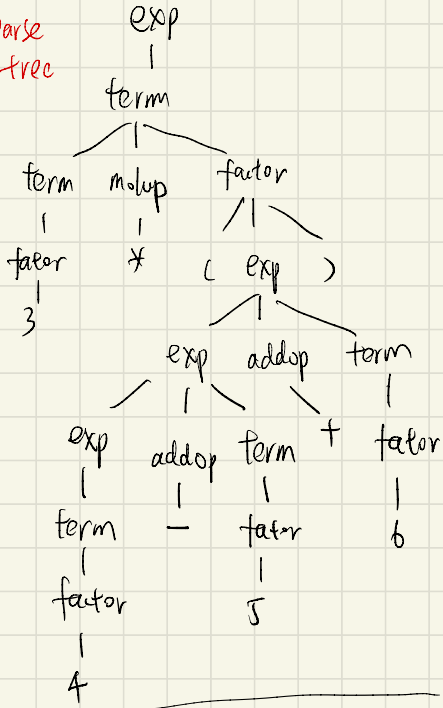
syntax tree



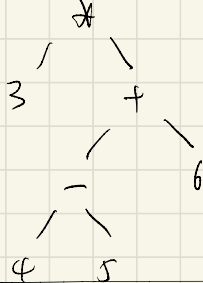
- ⇒ 3 + 4 * 5 add op term
- ⇒ 3 + 4 * 5 - factor
- ⇒ 3 + 4 * 5 - number
- ⇒ 3 + 4 * 5 - b

b. 3 * (4 - 5 + 6)

parse tree

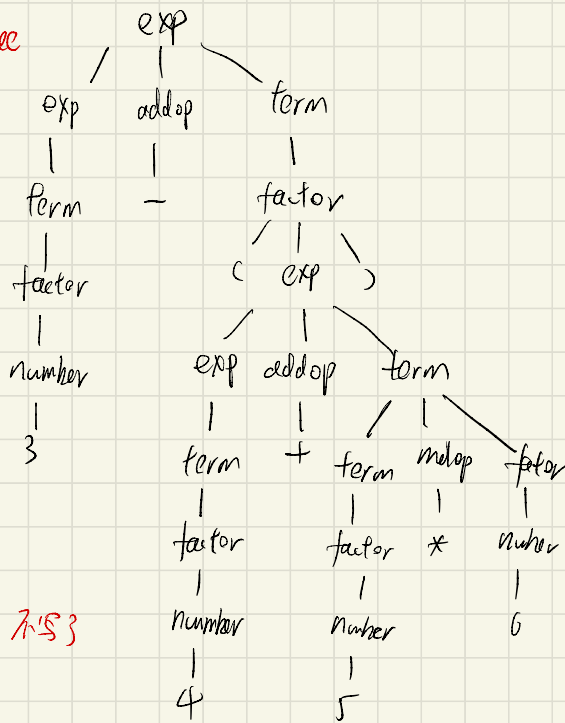


syntax tree



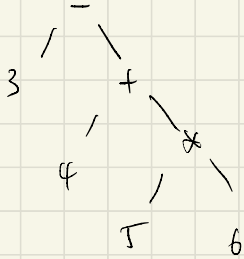
最左推导太长? 不写了
参见分析树

parse tree



c. 3 - (4 + 5 * 6)

syntax tree



最左推导太长? 不写了
参见分析树

3.10 a. Translate the grammar of exercise 3.6 into EBNF.

3.6 Consider the following grammar representing simplified LISP-like expressions:

$$\begin{aligned} \text{lexp} &\rightarrow \text{atom} \mid \text{list} \\ \text{atom} &\rightarrow \text{number} \mid \text{identifier} \\ \text{list} &\rightarrow (\text{lexp-seq}) \\ \text{lexp-seq} &\rightarrow \text{lexp-seq lexp} \mid \text{lexp} \end{aligned}$$
$$\begin{aligned} \text{lexp} &\rightarrow \text{atom} \mid \text{list} \\ \text{atom} &\rightarrow \text{number} \mid \text{identifier} \\ \text{list} &\rightarrow (\text{lexp-seq}) \\ \text{lexp-seq} &\rightarrow \text{lexp} \{ \text{lexp-seq} \} \end{aligned}$$

The Exercises of The Chapter Four

4.8 Consider the following grammar:

$$\begin{aligned} \text{lexp} &\rightarrow \text{atom} \mid \text{list} \\ \text{atom} &\rightarrow \text{number} \mid \text{identifier} \\ \text{list} &\rightarrow (\text{lexp-seq}) \\ \text{lexp-seq} &\rightarrow \text{lexp-seq lexp} \mid \text{lexp} \end{aligned}$$

- remove the left recursive.
- Construct First and Follow sets for the nonterminals of the resulting grammar.
- Show that the resulting grammar is LL(1).
- Construct the LL(1) parsing table for the resulting grammar.
- Show the actions of the corresponding LL(1) parser, given the input string (a (b (2)) (c)).

a.
$$\begin{aligned} \text{lexp} &\rightarrow \text{atom} \mid \text{list} \\ \text{atom} &\rightarrow \text{number} \mid \text{identifier} \\ \text{list} &\rightarrow (\text{lexp-seq}) \\ \text{lexp-seq} &\rightarrow \text{lexp} \text{ lexp-seq}' \\ \text{lexp-seq}' &\rightarrow \text{lexp} \text{ lexp-seq}' \mid \epsilon \end{aligned}$$

b.
$$\begin{aligned} \text{First}(\text{lexp}) &= \{ \text{number}, \text{identifier}, (\} \\ \text{First}(\text{atom}) &= \{ \text{number}, \text{identifier} \} \\ \text{First}(\text{list}) &= \{ (\} \\ \text{First}(\text{lexp-seq}) &= \{ \text{number}, \text{identifier}, (\} \\ \text{First}(\text{lexp-seq}') &= \{ \epsilon, \text{number}, \text{identifier}, (\} \end{aligned}$$

$$\begin{aligned} \text{Follow}(\text{lexp}) &= \{ \$, \text{number}, \text{identifier}, (,) \} \\ \text{Follow}(\text{atom}) &= \{ \$, \text{number}, \text{identifier}, (,) \} \\ \text{Follow}(\text{list}) &= \{ \$, \text{number}, \text{identifier}, (,) \} \\ \text{Follow}(\text{lexp-seq}) &= \{ \} \\ \text{Follow}(\text{lexp-seq}') &= \{ \} \end{aligned}$$

$$c. \text{First}(\text{'exp lexp-seq'}) \cap \text{Follow}(\text{'lexp-seq'}) \\ = \emptyset$$

d.

$$\begin{aligned} \text{select}(\text{'exp} \rightarrow \text{atom}) &= \{\text{number, identifier}\} \\ \text{select}(\text{'exp} \rightarrow \text{list}) &= \{ \{ \} \\ \text{select}(\text{atom} \rightarrow \text{number}) &= \{\text{number}\} \\ \text{select}(\text{atom} \rightarrow \text{identifier}) &= \{\text{'identifier'}\} \\ \text{select}(\text{'list} \rightarrow (\text{'exp-seq'})) &= \{ \{ \} \\ \text{select}(\text{'exp-seq'} \rightarrow \text{lexp lexp-seq'}) &= \{\text{number, identifier, C}\} \\ \text{select}(\text{'lexp-seq'} \rightarrow \text{'lexp lexp-seq'}) &= \{\text{number, identifier, C}\} \\ \text{select}(\text{'lexp-seq'} \rightarrow \epsilon) &= \{ \} \end{aligned}$$

	number	identifier	[)	\$
lexp	lexp \rightarrow atom	lexp \rightarrow atom	lexp \rightarrow list		
atom	atom \rightarrow number	atom \rightarrow 'identifier'			
list			list \rightarrow ('exp-seq')		
lexp-seq	lexp-seq \rightarrow lexp lexp-seq'				
lexp-seq'	lexp-seq' \rightarrow lexp lexp-seq'			lexp-seq' \rightarrow ϵ	

d. $(a(b(2))(\epsilon))$

parsing stack	input	action
\$ lexp	$(a(b(2))(\epsilon))\$$	lexp \rightarrow list
\$ list	$(a(b(2))(\epsilon))\$$	list \rightarrow ('exp-seq')
\$)lexp-seq C	$(a(b(2))(\epsilon))\$$	match
\$)lexp-seq	$a(b(2))(\epsilon)\$$	lexp-seq \rightarrow lexp lexp-seq'
\$)lexp-seq' lexp	$a(b(2))(\epsilon)\$$	lexp \rightarrow atom
\$)lexp-seq' atom	$a(b(2))(\epsilon)\$$	atom \rightarrow identifier
\$)lexp-seq' identifier	$a(b(2))(\epsilon)\$$	match
\$)lexp-seq'	$(b(2))(\epsilon)\$$	lexp-seq' \rightarrow lexp lexp-seq'
\$)lexp-seq' lexp	$(b(2))(\epsilon)\$$	lexp \rightarrow list
\$)lexp-seq' list	$(b(2))(\epsilon)\$$	list \rightarrow ('exp-seq')
\$)lexp-seq')lexp-seq C	$(b(2))(\epsilon)\$$	match
\$)lexp-seq')lexp-seq	$b(2))(\epsilon)\$$	lexp-seq \rightarrow lexp lexp-seq'

\$) lexp-seq') lexp	b(2))(ε))\$
\$) lexp-seq') lexp-seq' atom	b(2))(ε))\$
\$) lexp-seq') lexp-seq' identifier	b(2))(ε))\$
\$) lexp-seq') lexp-seq'	(2))(ε))\$
\$) lexp-seq') lexp-seq' lexp	(2))(ε))\$
\$) lexp-seq') lexp-seq' list	(2))(ε))\$
\$) lexp-seq') (lexp-seq') lexp-seq ((2))(ε))\$
\$ - - - - - lexp-seq)) (ε))\$	
\$ - - - - - lexp-seq' lexp)) (ε))\$	
\$ - - - - - atom)) (ε))\$	
\$ - - - - - number)) (ε))\$	
\$ - - - - - lexp-seq')) (ε))\$	
\$ - - - - -)) (ε))\$	
\$ - - - - - lexp-seq') (ε))\$	
\$) lexp-seq')) (ε))\$
\$) lexp-seq'	(ε))\$
\$) lexp-seq' lexp	(ε))\$
\$ - - - list	(ε))\$
\$ - - -) lexp-seq ((ε))\$
\$ - - -) lexp-seq	(ε))\$
\$ - - - lexp-seq' lexp	(ε))\$
\$ - - - lexp-seq' atom	(ε))\$
\$ - - - identifier	(ε))\$
\$ - - - lexp-seq')\$
\$) (lexp-seq'))\$
\$) lexp-seq')\$
\$))\$
\$	\$

```

lexp → atom
atom → identifier
match
lexp-seq' → lexp lexp-seq'
lexp → list
list → ( lexp-seq )
match
lexp-seq → lexp lexp-seq'
lexp → atom
atom → number
match
lexp-seq' → ε
match
lexp-seq' → ε
match
lexp-seq' → lexp lexp-seq'
lexp → list
list → ( lexp-seq )
match
lexp-seq → lexp lexp-seq'
lexp → atom
atom → identifier
match
lexp-seq' → ε
match
lexp-seq' → ε
match
ACCEPT!

```

4.9 Consider the following grammar:

lexp → atom | list

atom → number | identifier

list → (lexp-seq)

lexp-seq → lexp, lexp-seq | lexp

a. Left factor this grammar.

b. Construct First and Follow sets for the nonterminals of the resulting grammar.

c. Show that the resulting grammar is LL(1).

d. Construct the LL(1) parsing table for the resulting grammar.

e. Show the actions of the corresponding LL(1) parser, given the input string (a,(b,(2)),(c)).

a. $\text{lexp} \rightarrow \text{atom} \mid \text{list}$
 $\text{atom} \rightarrow \text{number} \mid \text{identifier}$
 $\text{list} \rightarrow (\text{lexp-seq})$
 $\text{lexp-seq} \rightarrow \text{lexp lexp-seq}'$
 $\text{lexp-seq}' \rightarrow , \text{lexp-seq} \mid \epsilon$

b. $\text{First}(\text{lexp}) = \{ \text{number}, \text{identifier}, (\}$
 $\text{First}(\text{atom}) = \{ \text{number}, \text{identifier} \}$
 $\text{First}(\text{list}) = \{ (\}$
 $\text{First}(\text{lexp-seq}) = \{ \text{number}, \text{identifier}, (\}$
 $\text{First}(\text{lexp-seq}') = \{ , , \epsilon \}$

$\text{Follow}(\text{lexp}) = \{ , ,) , \$ \}$

$\text{Follow}(\text{atom}) = \{ , ,) , \$ \}$

$\text{Follow}(\text{list}) = \{ , ,) , \$ \}$

$\text{Follow}(\text{lexp-seq}) = \{) \}$

$\text{Follow}(\text{lexp-seq}') = \{) \}$

c. $\text{First}(,) \cap \text{Follow}(\text{lexp-seq}') = \emptyset$

d.

	number	identifier	()	,	\$
lexp	$\text{lexp} \rightarrow \text{atom}$	$\text{lexp} \rightarrow \text{atom}$	$\text{lexp} \rightarrow \text{list}$			
atom	$\text{atom} \rightarrow \text{number}$	$\text{atom} \rightarrow \text{identifier}$				
list			$\text{list} \rightarrow (\text{lexp-seq})$			
lexp-seq	$\text{lexp-seq} \rightarrow \text{lexp lexp-seq}'$					
lexp-seq'				$\text{lexp-seq}' \rightarrow \epsilon$	$\text{lexp-seq}' \rightarrow , \text{lexp-seq}$	

e. $(a, (b, (2)), (c))$

parsing stack

\$ lexp
 \$ list
 \$) lexp-seq c
 \$) lexp-seq
 \$) lexp-seq' lexp
 \$ ——— atom
 \$ ——— identifier
 \$) lexp-seq'
 \$) L-S' L-S ,
 \$) L-S' L-S
 \$ ——— L-S' lexp
 \$ ——— list

input

$(a, (b, (2)), (\epsilon))$ \$
 $(a, (b, (2)), (\epsilon))$ \$
 $(a, (b, (2)), (\epsilon))$ \$
 $a, (b, (2)), (\epsilon))$ \$
 $a, (b, (2)), (\epsilon))$ \$
 $a, (b, (2)), (\epsilon))$ \$
 $, (b, (2)), (\epsilon))$ \$
 $, (b, (2)), (\epsilon))$ \$
 $(b, (2)), (\epsilon))$ \$
 $(b, (2)), (\epsilon))$ \$
 $(b, (2)), (\epsilon))$ \$

action.

$\text{lexp} \rightarrow \text{list}$
 $\text{list} \rightarrow (\text{lexp-seq}$
 match
 $L-S \rightarrow \text{lexp L-S}'$
 $\text{lexp} \rightarrow \text{atom}$
 $\text{atom} \rightarrow \text{id}$
 match
 $L-S' \rightarrow , L-S$
 match
 $L-S \rightarrow \text{lexp L-S}'$
 $\text{lexp} \rightarrow \text{list}$
 $\text{list} \rightarrow (L-S)$

\$	—————) L-S ((b, (2)), (ε))	\$
\$) L-S' L-S') L-S		b, (2)), (ε))	\$
\$	—————	L-S' lexp	b, (2)), (ε))	\$
\$	—————	atom	b, (2)), (ε))	\$
\$	—————	id	b, (2)), (ε))	\$
\$) L-S' L-S') L-S'		, (2)), (ε))	\$
\$	—————	L-S ,	, (2)), (ε))	\$
\$	—————	L-S	(2)), (ε))	\$
\$	—————	L-S' lexp	(2)), (ε))	\$
\$	—————	List	(2)), (ε))	\$
\$	—————) L-S ((2)), (ε))	\$
\$) L-S' L-S') L-S') L-S		2)), (ε))	\$
\$	—————	L-S' lexp	2)), (ε))	\$
\$	—————	atom	2)), (ε))	\$
\$	—————	num	2)), (ε))	\$
\$	—————	L-S')), (ε))	\$
\$) L-S' L-S') L-S'))), (ε))	\$
\$) L-S' L-S') L-S'), (ε))	\$
\$) L-S' L-S' ,), (ε))	\$
\$) L-S' L-S'		, (ε))	\$
\$) L-S' L-S ,		, (ε))	\$
\$) L-S' L-S		(ε))	\$
\$) L-S' L-S' lexp		(ε))	\$
\$	—————	List	(ε))	\$
\$	—————) L-S ((ε))	\$
\$	—————) L-S	ε))	\$
\$	—————) L-S' lexp	ε))	\$
\$	—————	atom	ε))	\$
\$	—————	id	ε))	\$
\$) L-S' L-S') L-S'		>))	\$
\$) L-S' L-S')		>))	\$
\$) L-S' L-S')	\$
\$) L-S')	\$
\$))	\$
\$			\$	

match

L-S → (lexp L-S'

lexp → atom

atom → id

match

L-S' → , L-S

match

L-S → (lexp L-S'

lexp → List

List → (L-S)

match

L-S → lexp L-S'

lexp → atom

atom → number

match

L-S' → ε

match

L-S' → ε

match

L-S' → , L-S

match

L-S → (lexp L-S'

lexp → List

List → (L-S)

match

L-S → lexp L-S'

lexp → atom

atom → id

match

L-S' → ε

match

L-S' → ε

L-S' → ε

match

ACCEPT !!!