# 浙江理工大学 2020—2021 学年第 一 学期

## 《编译原理（双语）》期末试卷（ A ）卷

（试题共 5 页）

本人郑重承诺：本人已阅读并且透彻地理解《浙江理工大学考场规则》，愿意在考试中自觉遵守这些规定，保证按规定的程序和要求参加考试，如有违反，自愿按《浙江理工大学学生违纪处分规定》有关条款接受处理。

承诺人签名：_____ 学号：_____ 班级：_____

1. （10 points）Write English description for the languages generated by following regular expression:

1) 0+(0 |1)1+

2) 0*(100*)*1*

2. （12 points）

a. Please check out which strings can be generated by the regular expression (ab|b)*cc?

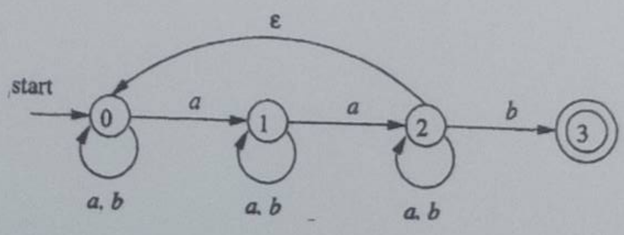
abbc， abab， bcc， babcc， aaabc

b. Please check out which strings can be generated by the regular expression (b | a)b+(ba)* ?

aba, abb, ababa, aab, bbb

c. please determine which strings can be accepted by the NFA.

aab, bab, bbab, aaabb, abababab



3. （12 points）Consider the following regular expression from the alphabet {a,b}:

b*a | bb

a. Use Thompson's construction to make an NFA from the regular expression (show it as a state diagram).

b. Use subset construction to create a DFA equivalent to the NFA you gave for part A.

4. (6 points) Given the grammar:

E→T|E+T|E-T

T→F|T*F|T/F

F→（E）|i

Please list all non-terminals and terminals in this grammar, and give the start symbol of the grammar.

5. (10 points) Given the grammar

*exp* → *exp* + *term* | *exp* - *term* | *term*

*term* → *term* * *factor* | *term* / *factor* | *factor*

*factor* → ( *exp* ) | ***number***

Write down ***leftmost derivations*** for: 3*(6-5) and ***rightmost derivations*** for 16*6/4

6. (25 point) Consider the following grammar:

$S \rightarrow Sb \quad S \rightarrow Ab \quad S \rightarrow b \quad A \rightarrow Aa \quad A \rightarrow a$

a. remove the left recursion. (5 point)

b. Construct First and Follow sets for the nonterminals of the resulting grammar. (6 point)

c. Construct the LL(1) parsing table for the resulting grammar. (6 point)

d. show the action of LL(1) parser that used the parsing table to recognize the following string: aaabb. (8 point)

sulting grammar. (6 point)

5 point)

e to recognize the following string:

7.(10 points)write an attribute grammar for the integer value of a number given by following grammar:

number→digit number | digit

digit → **0|1|2|3|4|5|6|7|8|9**

8. (15 point)Consider the following grammar with numbered productions

1) E --> E **x** T

2) E --> E **x**

3) E --> **y** T

4) T --> **y** T

5) T --> **z**

Construct the SLR parsing tables for the grammar. In particular, show the following:

a. The augmented grammar

b. The DFA to recognize viable prefixes, including the set of items for each state.

c. The action and goto tables

浙江理工大学 2020—2021 学年第一学期

《编译原理（双语）》期末试卷（ A ）卷标准答案和评分标准

**1.Sol:** (10 points)

1) 001, 011, 0001, 0011; any string of length 3 or greater that is one or more 0's are followed by one or more 1's.

2) 0, 1, 01, 0101; any string that has no substring 110

2. **Sol:**(12 points)

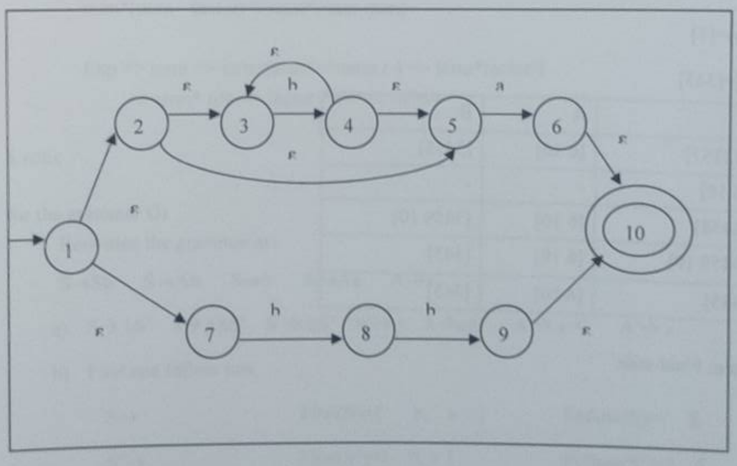
a) ~~abbe~~ ~~abab~~ bcc babcc ~~aaabe~~

b) ~~aba~~, abb, ~~ababa~~, ~~aab~~, bbb

c) aab ~~bab~~ ~~bbab~~ aaabb ababababab

3. **Sol:** (12 points)

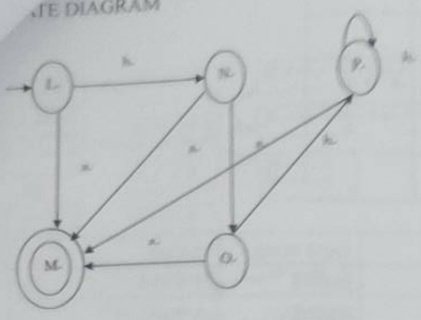
**Thompson's Construction**




**part B. Use subset construction to create a DFA equivalent to the NFA you gave for part A. Show your work. Show it as a state table, *using the sets from the NFA as the names for the new states*, as we did in examples in lecture.**

Start state: [1]

TE DIAGRAM



4.Solu:（6 points）

The set of the terminals VT={+, -, *, /, (, ), i}. The set of the nonterminals VN={E, T, F}.

With E being the start symbol

5、10 points

The leftmost derivations for the expression 3*(6-5) and 16*6/4:

Exp => term => term * factor => factor * factor => num * factor =>num * (exp) =>
num*(exp - term) => num*(term - term) => num*(factor - term) => num*(num - term)=>
num*(num - factor)=>num*(num-num)

Exp => term => term/factor =>term / 4 => term*factor/4
=>term* 6/4 => factor *6/4 => 16*6/4

6.solu:

for the grammar G:

Rewritten the grammar as:

S→Sb S→Ab S→b A→Aa A→a

a) S→ bS' S→AbS' S'→ bS' S'→ ε A→aA' A'→ a A' A'→ ε

b) First and follow sets

| | | |
|---|---|---|
| S→ | First(S)={ b, a } | Follow(S)={ $ } |
| S'→ | First(S')={ b, ε } | Follow(S')={ $ } |
| A→ | First(A)={ a } | Follow(A)={ b } |
| A'→ | First(A')={ a , ε } | Follow(A')={ b } |

c) LL(1) Parsing table:

ε closure[1]=[12357]

mov(12357,a)=[6]

ε closure[6]=[6 10] –Final state

mov(12357,b)=[48]

ε closure[48]=[3458]

mov(3458,a)=[6]

ε closure[6]=[6 10] –Final state

mov(3458,b)=[49]

ε closure[49]=[3459 10]

mov(3 4 5 9 10,a)=[6]

ε closure[6]=[6 10] –Final state

mov(3 4 5 9 10,b)=[4]

ε closure[4]=[345]

mov(3 4 5,a)=[6]

ε closure[6]=[6 10] –Final state

mov(3 4 5,b)=[4]

ε closure[4]=[345]

| | | A | B |
|---|---|---|---|
| L | [12357] | [6 10] | [3458] |
| M* | [6 10] | - | - |
| N | [3458] | [6 10] | [3459 10] |
| O | [3459 10] | [6 10] | [345] |
| P | [345] | [6 10] | [345] |

*Indicates Final state

| | a | b | $ |
|---|---|---|---|
| S | S→AbS' | S→bS' | |
| S' | | S'→bS' | S'→ ε |
| A | A→aA' | | |
| A' | A'→ a A' | A'→ ε | |

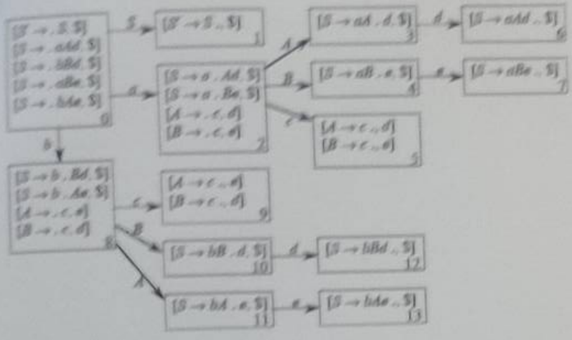
d)

| Parsing stack | Input string | Action |
|---|---|---|
| $ S | aaabb$ | S→AbS' |
| $ S'bA | aaabb$ | A→aA' |
| $ S'bA'a | aaabb$ | match |
| $ S'bA' | aabb$ | A→aA' |
| $ S'bA'a | aabb$ | match |
| $ S'bA' | abb$ | A→aA' |
| $ S'bA'a | abb$ | match |
| $ S'bA' | bb$ | A'→ ε |
| $ S'b | bb$ | match |
| $ S' | b$ | S'→bS' |
| $ S'b | b$ | match |
| $ S' | $ | accept |

7. sol:

| Grammar Rule | Semantic Rules |
|---|---|
| Number1→number2 digit | number1.val = number2.val*10+digit.val. |
| Number→digit | number.val= digit.val. |
| digit→0 | digit.val = 0. |
| digit→1 | digit.val = 1. |
| digit→2 | digit.val = 2. |
| digit→3 | digit.val = 3. |
| digit→4 | digit.val = 4. |
| digit→5 | digit.val = 5. |
| digit→6 | digit.val = 6. |
| digit→7 | digit.val = 7. |
| digit→8 | digit.val = 8. |
| digit→9 | digit.val = 9. |

8.solu:



But here is an LALR(1) DFA for the grammar. Because state 8 contains a reduce-reduce conflict, the grammar is not LALR(1).

But here is an LALR(1) DFA for the grammar. Because state 8 contains a reduce-reduce conflict, the grammar is not LALR(1).