**浙江大学2023–2024学年春夏学期**

**《编译原理》课程期中考试试卷**

课程号： 21120471 ，开课学院： 计算机学院

考试形式：闭、√开卷（请在选定项上打√），允许带\_ 1页A4纸\_入场

考试日期：2023年 4月 23 日,考试时间： 120 分钟

**诚信考试，沉着应考，杜绝违纪。**

**考生姓名： 学号： 所属院系： \_**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **题序** | **一** | **二** | **三** | **四** | **五** | **总 分** |
| **得分** |  |  |  |  |  |  |

1. **Mark each statement true or false (2 points each, 20 cents )**
   1. We can write a recursive-descendant parser for a left-recursive CFG. ( F )
   2. Given a legal string of tokens for a CFG, there must be a unique parsing tree to derivate the string. ( T )
   3. The language L={anbn|n>=1} can’t be described by a regular expression but a grammar. ( T )
   4. There is only one derivation for the string of an unambiguous grammar. ( T )
   5. If a grammar is LR(1), but not LALR(1). There may be shift-reduce conflicts in its parsing table of LALR(1). ( F )
   6. There might be shift-reduce, shift-shift and reduce-reduce conflicts during the LR parsing. ( F )
   7. For each regular expression there exists at least one equivalent DFA. ( T )
   8. Left recursion is commonly used to make operations left associative. ( T )
   9. The syntax tree will completely reflect the derivation steps for a string. ( F )
   10. Both DFA and NFA can recognize regular set. ( T )

1. **Single Choice (2 points each, 20 cents)**
   1. In the production A →BαC，we have \_\_\_\_\_\_\_. ( C )

[A] Follow (C) ⊂ Follow (A), First( B) ⊂First(A) [B] Follow (C) ⊂ Follow (A)，First( A) ⊂ First(B)

[C] Follow (A) ⊂ Follow (C)，First( B) ⊂ First(A) [D] Follow (A) ⊂ Follow (C)，First( A) ⊂ First(B)

感觉这里有些问题，First(B) 包含ε的话 First(A) 不一定包含ε，如果用虎书的 nullable 的形式的话就没问题了

* 1. Which of the following string can be defined by the regular expression (a|c)\*ac\*b(a|b|c)\* ? ( C )

[A] bbccaa [B] bbaacc [C] aaccbb [D] ccbbaa

* 1. Give the LR(1) item , we have . ( A )

(A) {a,b}⊂Follow(A) (B) {a,b}⊂Follow(B)

(C) {a,b}⊂First(A) (D) {a,b}⊂First(B)

* 1. If one CFG grammar is LL(1) and contains the rules: A→α1|α2; B→β1|β2, then the following condition \_\_\_\_\_\_ must be satisfied. ( B )

[A] First (A) ∩First (B) is empty [B] First(α1) ∩First(α2) is empty

[C] First (A) ∩Follow (A) is empty [D] First (α1) ∩First(β1) is empty

* 1. Here is a grammar: ( C )

X→a X→ε Y→b Y→X

Z→c Z→XYZ W→d W→XY

Which symbol is not nullable?

[A]. X [B].Y [C].Z [D].W

* 1. The parsing method of YACC is \_\_\_\_\_\_\_. ( A )

[A]LALR(1) [B]LR(1) [C]SLR(1) [D]LL(1)

* 1. Which action is not in a LR parsing table? ( B )

[A]Shift [B]Push [C]Accept [D]Reduce

* 1. In the following parsing methods, \_\_\_\_\_\_\_ is the top-down parsing? ( A )

[A] LL(k) [B] LR(k) [C] SLR(k) [D] LALR(k)

* 1. The output of the scanner is \_\_\_\_\_\_\_. ( A )

[A]token [B]syntax tree [C]target code [D]intermediate code

* 1. Which one is not related to the LL(1) parsing method. ( D )

[A] Left factoring [B] First set and follow set

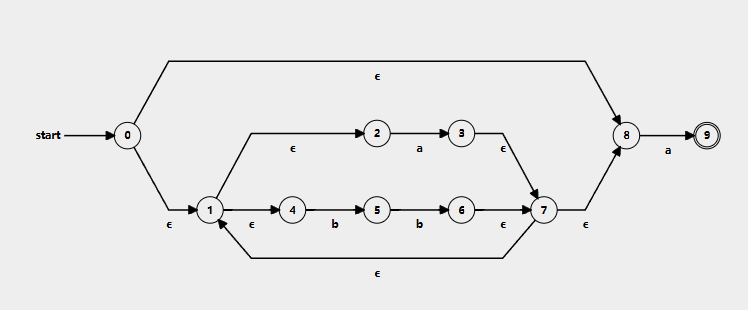
[C]Remove left recursion [D] Remove right recursion

1. (1) Convert the regular expression (a|bb)\*a into an NFA; ( 10 cents)

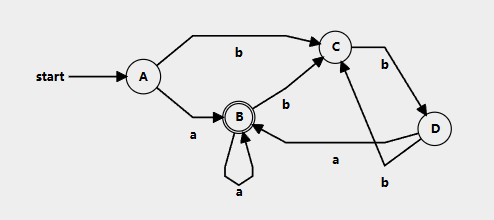
(2) Then, convert the above NFA into a DFA using the subset construction. ( 10 cents)

**Solution:**

(1)



(2)



这里可以加一个额外的状态使状态 c 有一条接受字符 a 的转移（dead state）

1. Consider the following grammar. (25 cents)

R→aFe

F→Y **;** F

F→Y

Y→i **:** t

Y→ε

(R is the start symbol. Semicolon ’ **;**’ and colon ‘:’ are terminals.)

1. Construct the FIRST and FOLLOW sets of nonterminals in the grammar. (10 cents)
2. Construct the LR(1) DFA for the grammar. (15 cents)

**Solution**:

1. The first sets and follow sets are as follows.

First(R) = { ‘a’ } Follow(R) = { $ }

First(F) = { ‘i’ } Follow(F) = { ‘e’ }

First(Y) = { ‘i’ } Follow(Y) = { ‘;’ ‘e’ }

Nullable = { F, Y }

这里用了虎书的表示方法

1. The LR(1) DFA of this grammar is as follows:

R→▪aFe, $

R→a▪Fe, $

F→▪Y;F, e

F→▪Y, e

Y→▪i:t, e/;

Y→ε▪, e/;

R→aF▪e, $

a

F→Y▪;F, e

F→Y▪, e

F

Y

Y→i▪:t, e/;

t

e

F→Y; ▪F, e

F→▪Y;F, e

F→▪Y, e

Y→▪i:t, e/;

Y→ε▪, e/;

R→aFe▪, $

**;**

Y

i

Y→i: ▪t, e/;

Y→i: t▪, e/;

:

i

1. Consider the following grammar. (15 cents)

0: S’→S

1: S→aAb

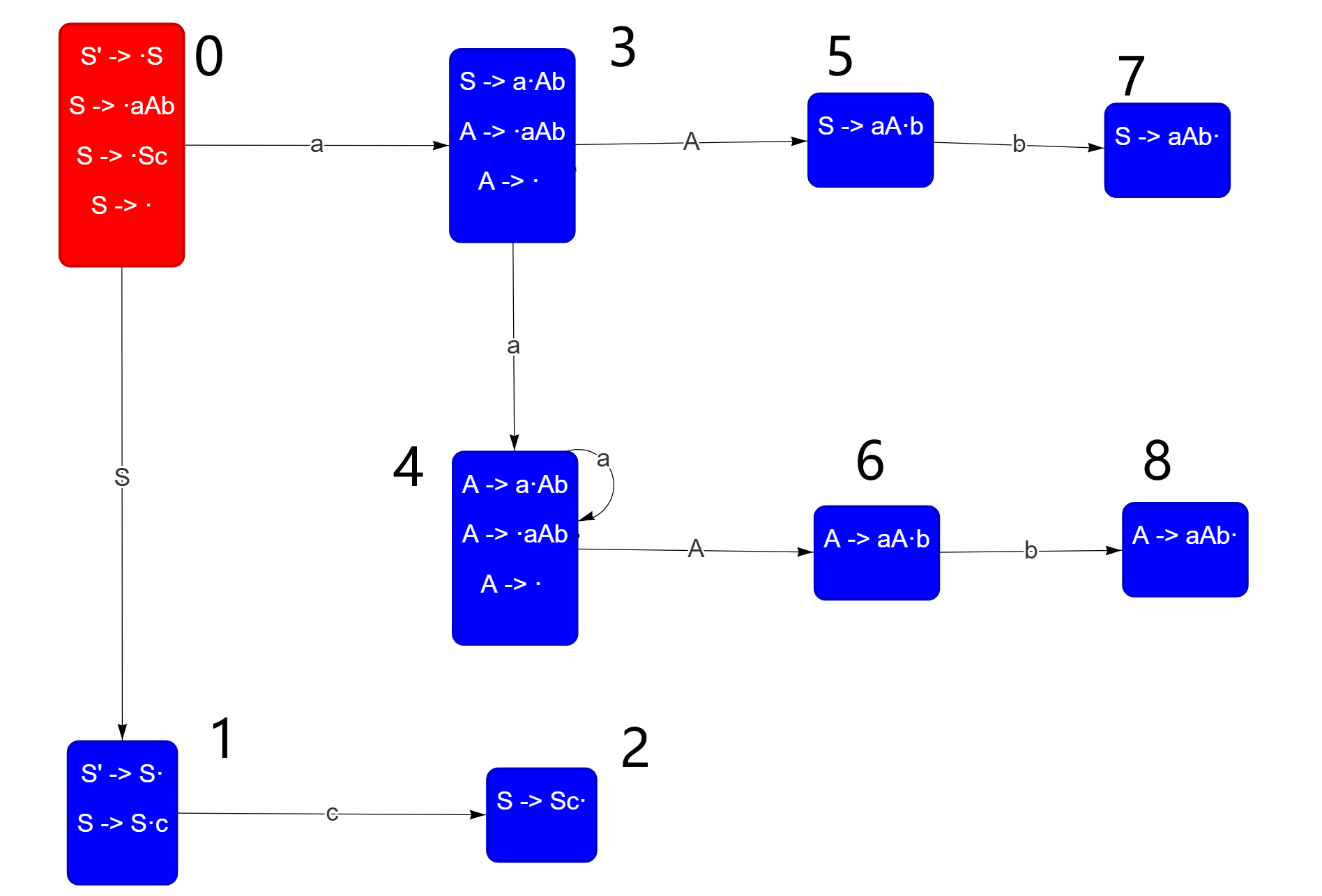
2: S→Sc

3: S→ε

4: A→aAb

5: A→ε

1. Construct the DFA of LR(0) for this grammar. (8cents)



(2) Construct the SLR(1) parsing table. (7 cents)

**Solution**:

Follow(S) = { $, c }

Follow(A) = { b }

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| state | ACTION | | | | GOTO | |
| a | b | c | $ | S | A |
| 0 | s3 |  | r3 | r3 | g1 |  |
| 1 |  |  | s2 | accept |  |  |
| 2 |  |  | r2 | r2 |  |  |
| 3 | s4 | r5 |  |  |  | g5 |
| 4 | s4 | r5 |  |  |  | g6 |
| 5 |  | s7 |  |  |  |  |
| 6 |  | s8 |  |  |  |  |
| 7 |  |  | r1 | r1 |  |  |
| 8 |  | r4 |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

写完后记得检查一遍，DFA 的每个 nonterminal 的转移边对应一个 goto，每个 terminal 对应一个 shift，每个完成的 term （点在最右侧，到空串视为已完成）对应一个 reduce

有同学多写了 goto，不是每个 reduce 必定对应一个 goto 的