



中国科学技术大学
University of Science and Technology of China

词法分析 II

《编译原理和技术(H)》

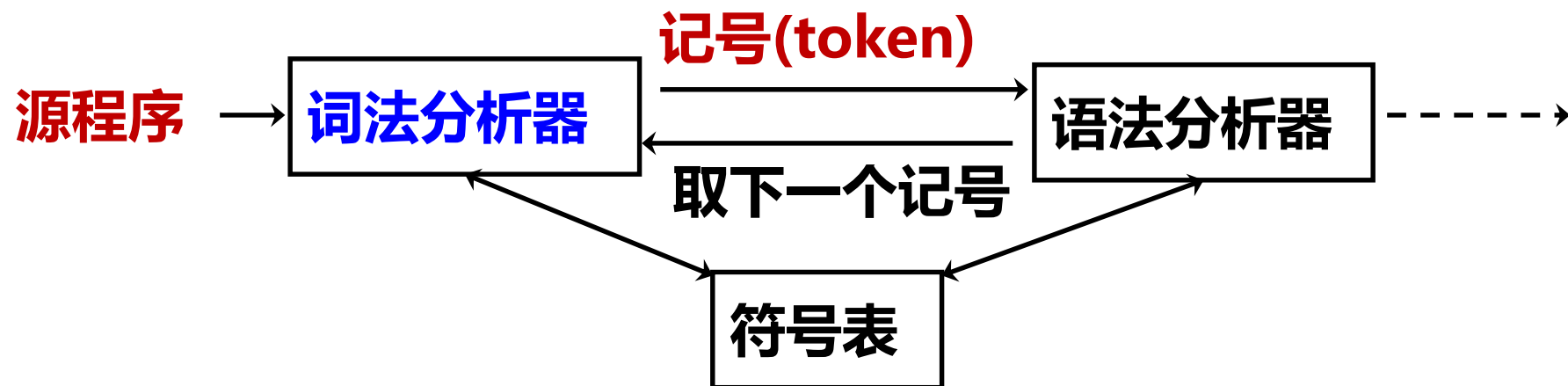
张昱

0551-63603804, yuzhang@ustc.edu.cn

中国科学技术大学
计算机科学与技术学院



本章内容



□ 词法分析及要解决的问题

- 向前看(Lookahead)、歧义(Ambiguities)

□ 词法分析器的自动生成

- 词法的描述: **正规式**; 词法记号的识别: **转换图**
- **有限自动机: NFA、DFA**



中国科学技术大学
University of Science and Technology of China

2.4 有限自动机

- 描述分析器：NFA、DFA
(对转换图的形式定义和分类)



有限自动机

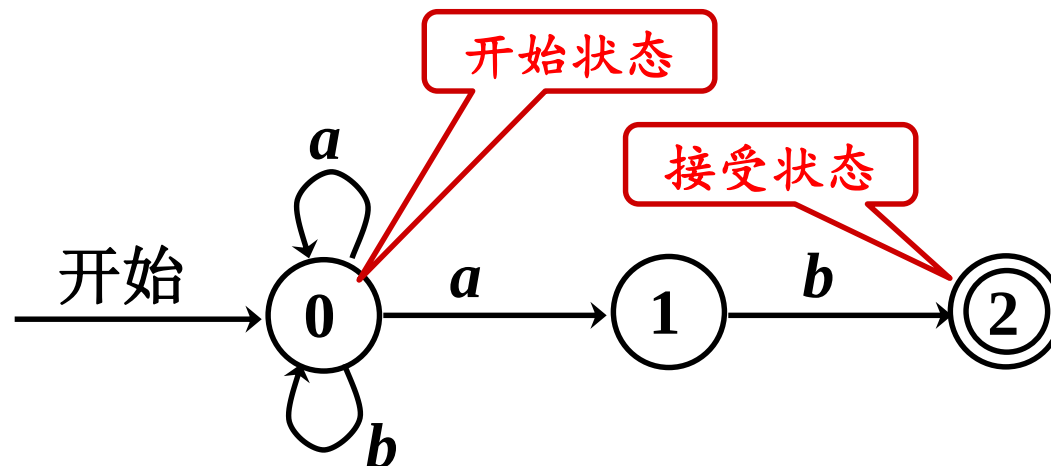
□ 不确定的有限自动机 (NFA)

一个数学模型，它包括：

(nondeterministic finite automaton)

- 1、有限的状态集合 S
- 2、输入符号集合 Σ
- 3、转换函数 $move : S \times (\Sigma \cup \{\epsilon\}) \rightarrow P(S)$
- 4、状态 s_0 是唯一的开始状态
- 5、 $F \subseteq S$ 是接受状态集合

识别语言
 $(a|b)^*ab$
的NFA



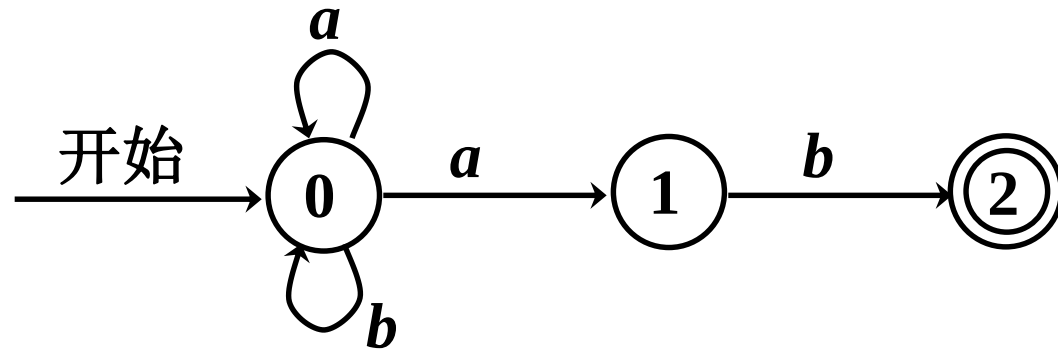


转换表

□ NFA的转换表

	输入符号	
	<i>a</i>	<i>b</i>
0	{0, 1}	{0}
1	∅	{2}
2	∅	∅

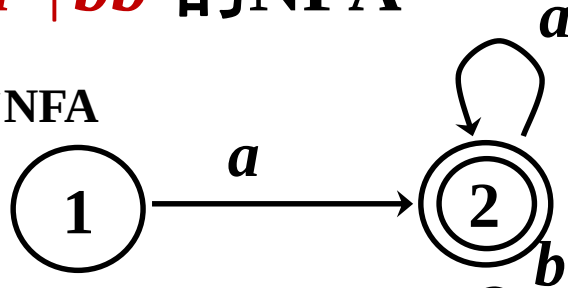
识别语言
 $(a|b)^*ab$
的NFA



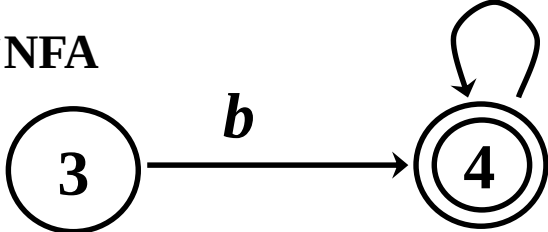
例题3

识别 $aa^* | bb^*$ 的NFA

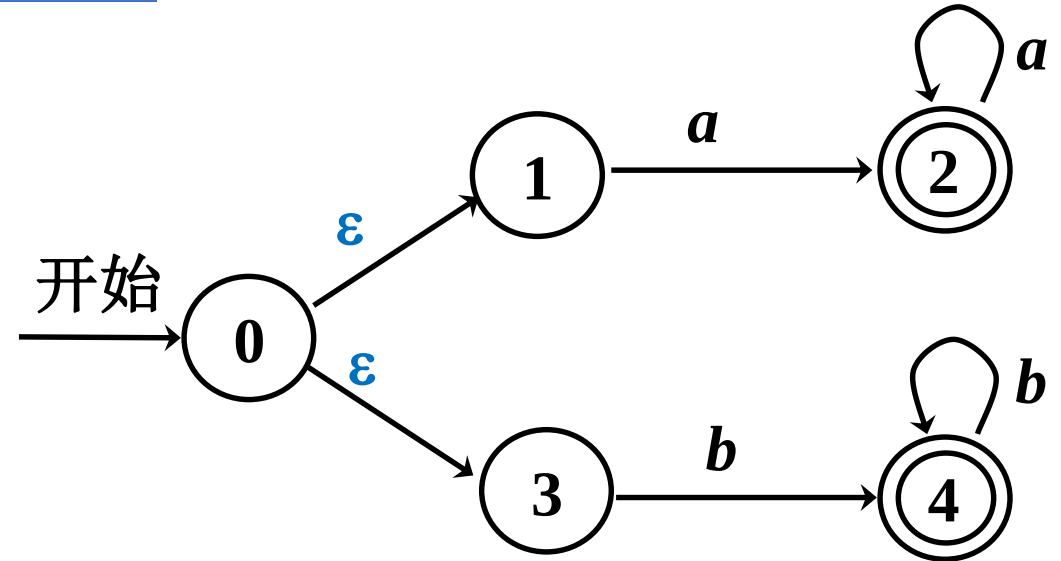
识别 aa^* 的NFA



识别 bb^* 的NFA

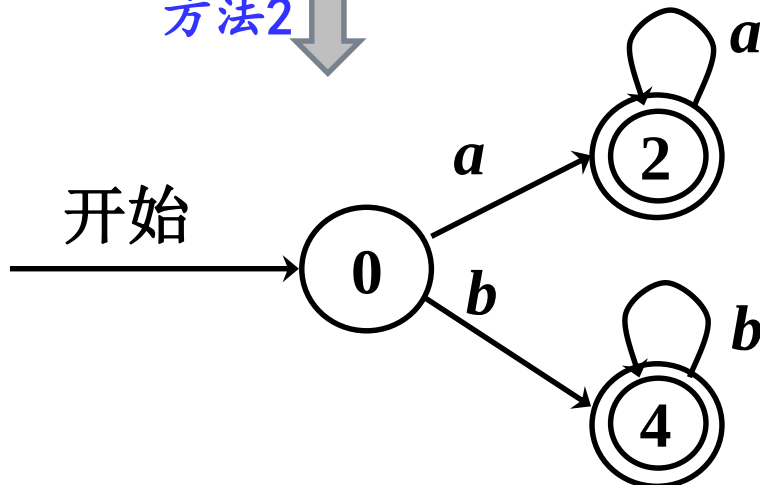


方法1



输入串 $aaaa$ 由开始状态0、到1、2、2、2的路径来接受
或者由开始状态0、到3、回退到0，再到1、2、2、2的路径来接受

方法2



合并两个子NFA的
开始状态1和3，设为0



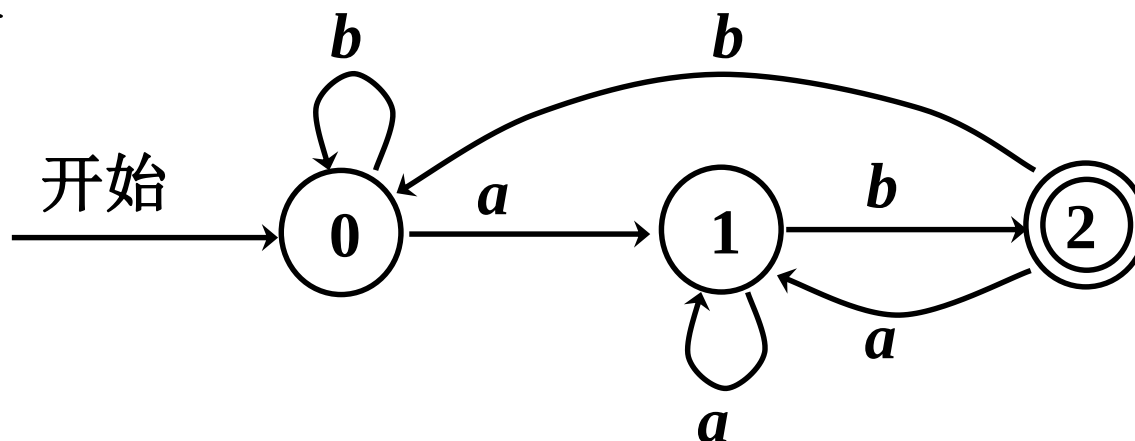
确定的有限自动机

□ 确定的有限自动机 (DFA)

一个数学模型，它包括：

- 1、有限的状态集合 S
- 2、输入符号集合 Σ
- 3、转换函数 $move : S \times \Sigma \rightarrow S$ ，且可以是部分函数
- 4、状态 s_0 是唯一的开始状态
- 5、 $F \subseteq S$ 是接受状态集合

识别语言
 $(a|b)^*ab$
的DFA





NFA vs. DFA

□ 主要差异在转换函数

■ NFA: $move : S \times (\Sigma \cup \{\epsilon\}) \rightarrow P(S)$

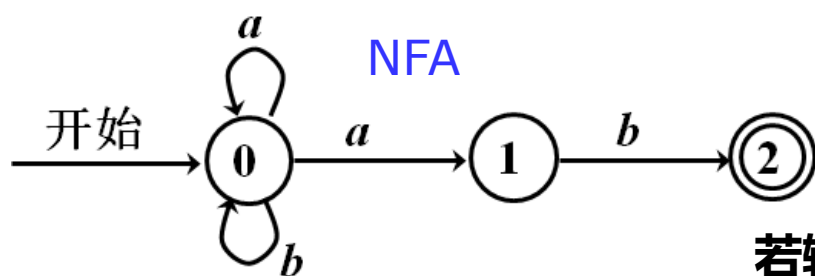
效率低

□ 要识别一个token, 需要对多种可能的路径**试探+失败回退**

■ DFA: $move : S \times \Sigma \rightarrow S$, 可以是**部分函数**

快速高效

□ 对于面临的 Σ 中的符号, 状态转换是明确的



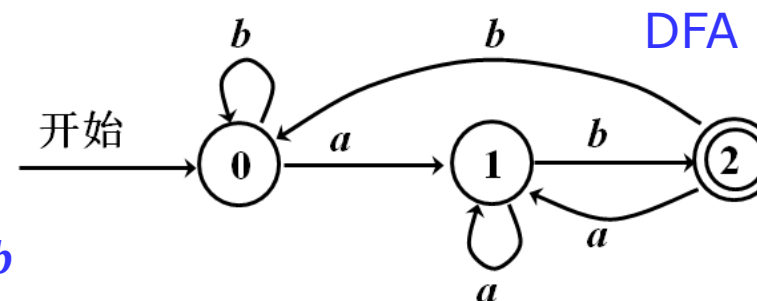
识别语言
 $(a|b)^*ab$

若输入串是 ab

0-(a)→0-(b)→0-结束符
0←回退b

0←回退a

0-(a)→1-(b)→2-结束(接受状态)



0-(a)→1-(b)→2-结束(接受状态)

但是, 由正规式不易构造DFA



例题4

构造一个DFA，它能识别 $\{0,1\}$ 上能被5整除的二进制数。

解答

	已读过	尚未读	已读部分的值
某时刻	101	0111000	5
读进0	1010	111000	$5 \times 2 = 10$
读进1	10101	11000	$10 \times 2 + 1 = 21$

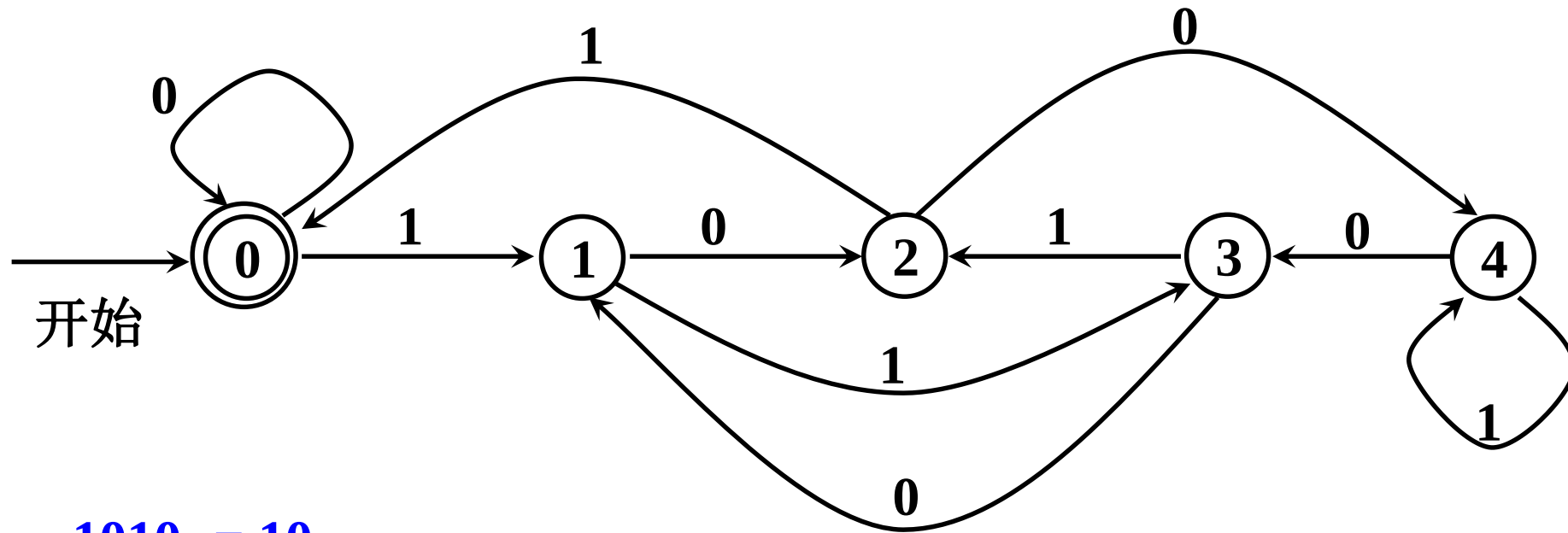
引入5个状态即可，分别代表已读部分的值除以5的余数



例题4

构造一个DFA，它能识别 $\{0,1\}$ 上能被5整除的二进制数。

解答



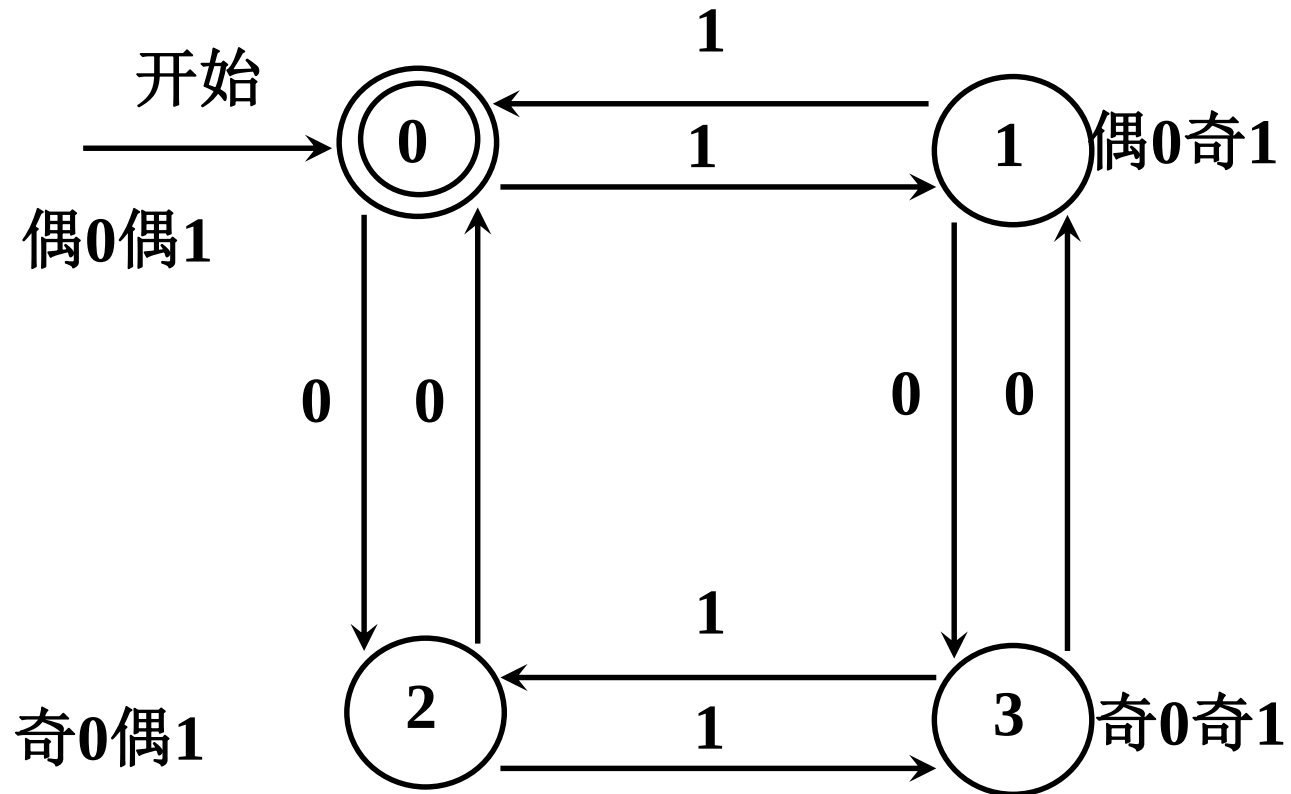
$$1010_2 = 10_{10}$$

$$111_2 = 7_{10}$$



例题5

构造一个DFA，它能接受0和1的个数都是偶数的字符串。





2.5 从正规式到有限自动机

□ 分析器的自动构造

- 正规式 $\rightarrow$ NFA $\rightarrow$ DFA $\rightarrow$ 化简的DFA
采用语法制导的算法来构造NFA



词法分析器的自动生成技术

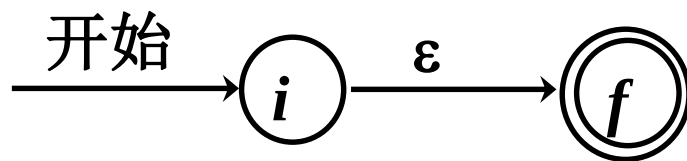
- 正规式：描述语言的词法
- 有限自动机：刻画词法分析的实现

- 词法分析器自动生成的主要过程
 - 正规式→NFA (语法制导的构造算法)
 - NFA→DFA (子集构造法)
 - DFA化简
 - 根据DFA构造词法分析器源码

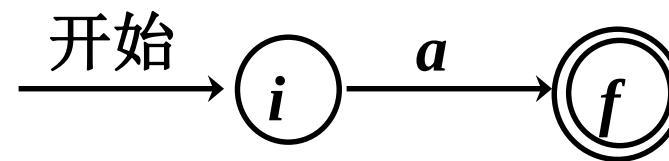


语法制导的NFA构造算法

- 语法制导(Syntax-directed): 按正规式的语法结构来指导构造
- 首先, 构造识别 ε 和字母表中一个符号的NFA
 - 重要特点: 仅有一个接受状态, 接受状态没有出边



识别正规式 ε 的NFA



识别正规式 a 的NFA

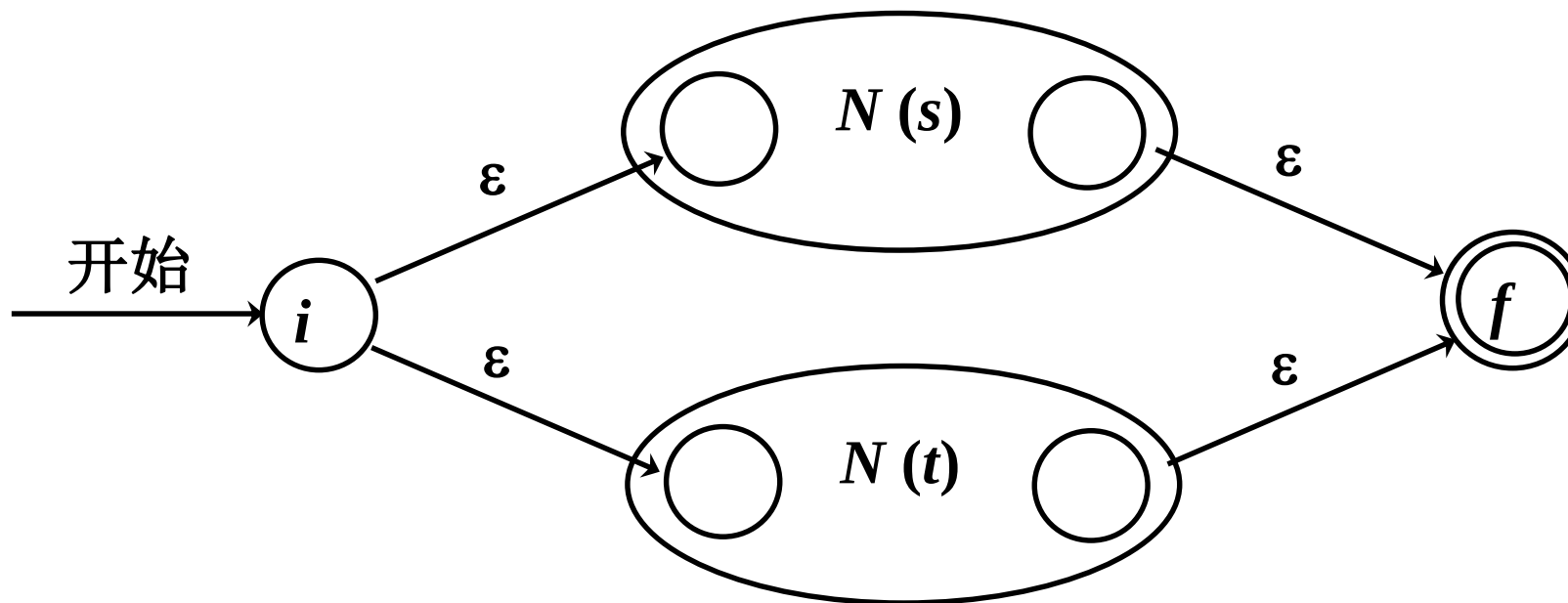
- 对于带括号的正规式(s), 使用 s 对应的NFA $N(s)$ 本身作为(s)的NFA



语法制导的NFA构造算法

□ 构造识别主算符为**选择**的正规式的NFA

■ **重要特点**：仅一个接受状态，接受状态没有出边



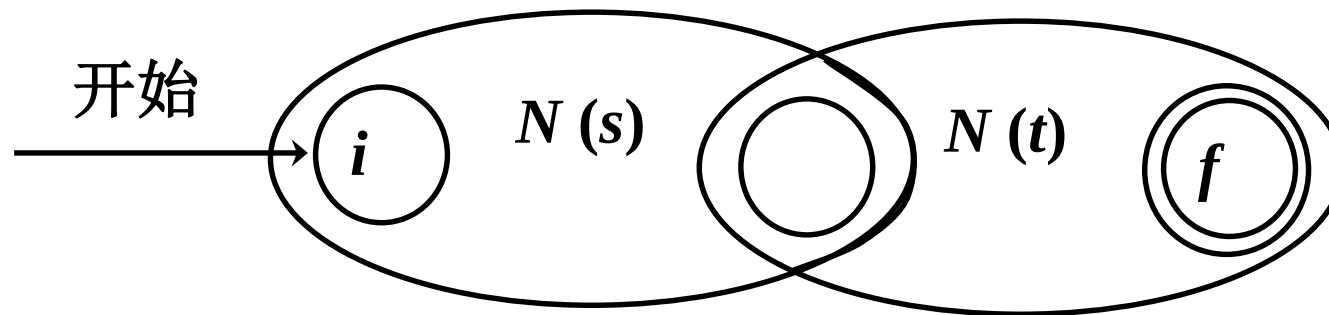
识别正规式 $(s) | (t)$ 的NFA



语法制导的NFA构造算法

□ 构造识别主算符为**连接**的正规式的NFA

■ **重要特点**：仅一个接受状态，接受状态没有出边

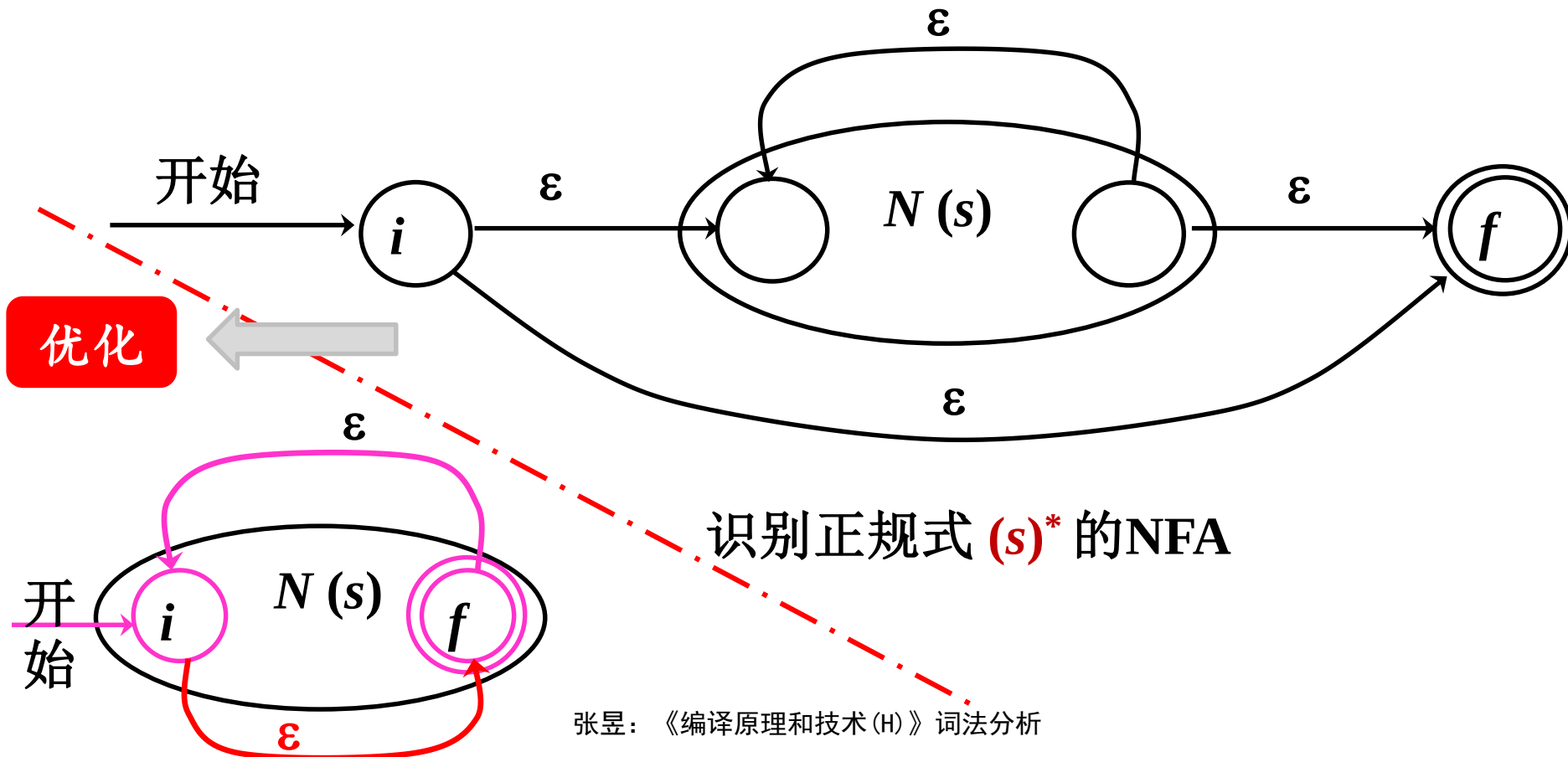


识别正规式 **(s)(t)** 的NFA

语法制导的NFA构造算法

□ 构造识别主算符为**闭包**的正规式的NFA

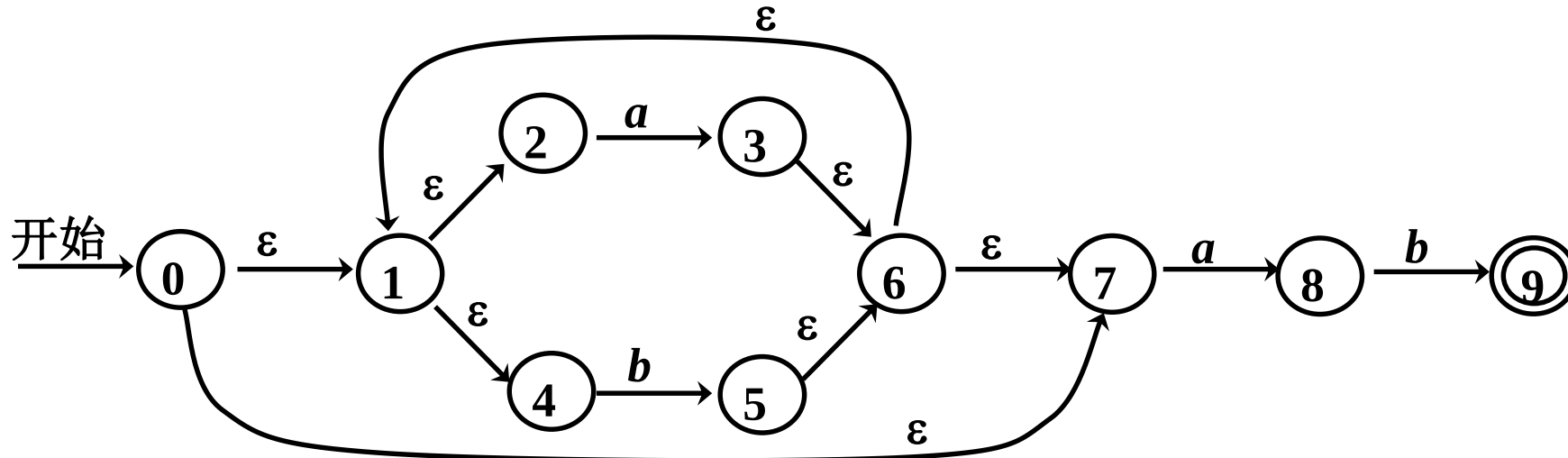
■ **重要特点**：仅一个接受状态，接受状态没有出边



语法制导的NFA构造算法

□ 本方法产生的NFA有下列性质

- $N(r)$ 的状态数最多是 r 中符号和算符总数的两倍
- $N(r)$ 只有一个接受状态，接受状态没有向外的转换
- $N(r)$ 的每个非接受状态有
 - 一个用 Σ 的符号标记指向其它结点的转换，或者
 - 最多两个指向其它结点的 ϵ 转换

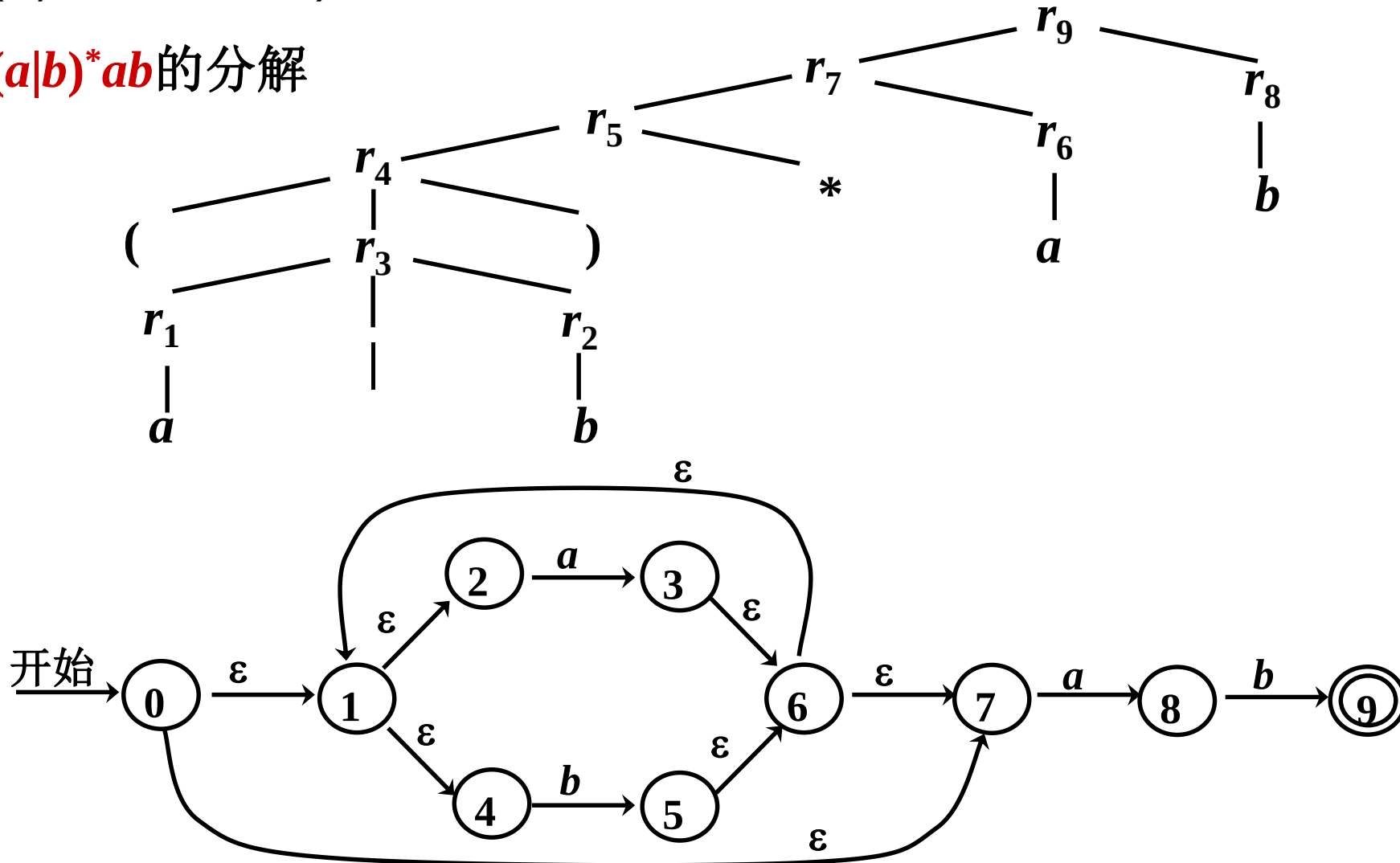




NFA构造过程举例

语法制导(Syntax-directed): 按正规式的语法结构来指导构造

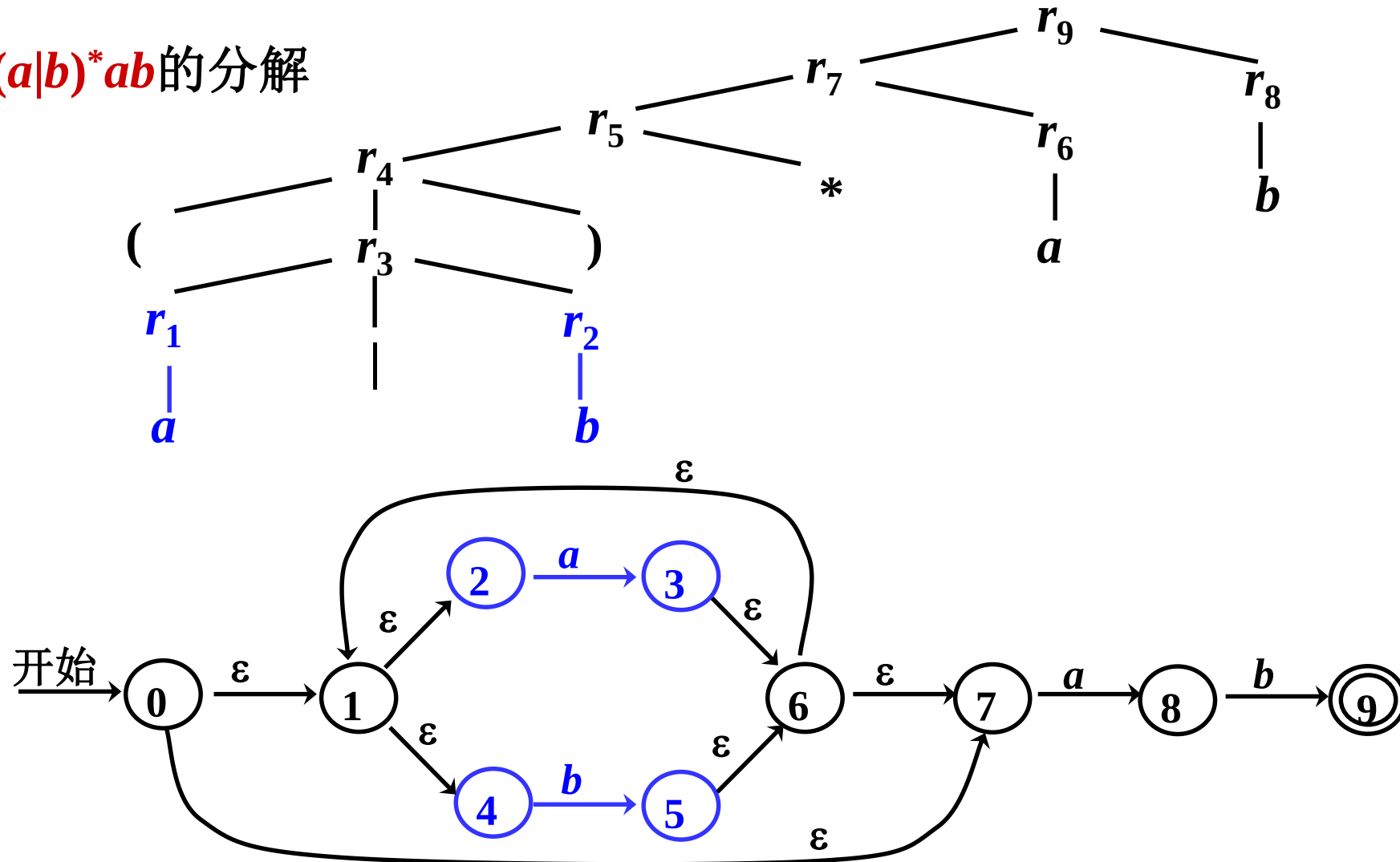
$(a|b)^*ab$ 的分解





NFA构造过程举例

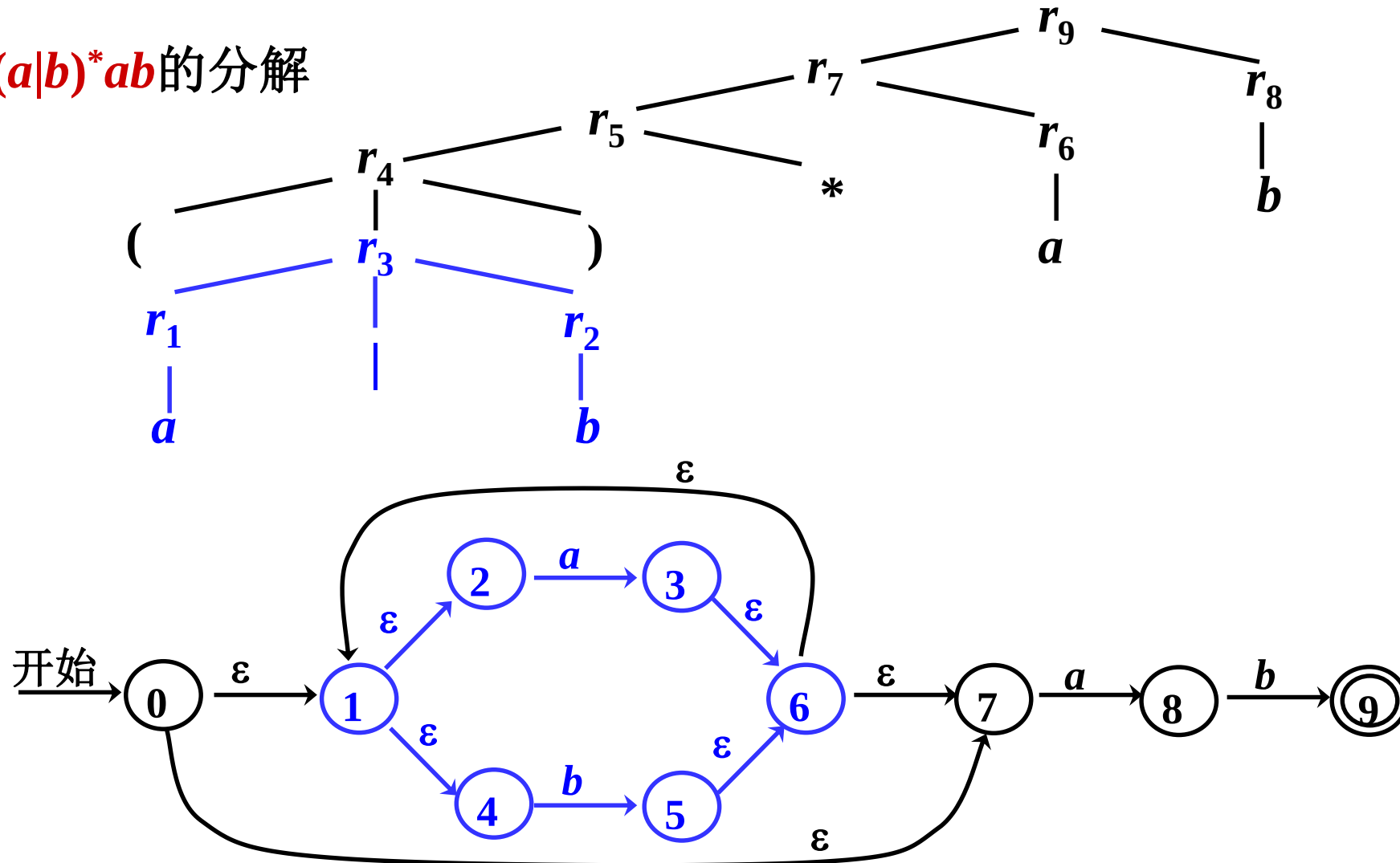
$(a|b)^*ab$ 的分解





NFA构造过程举例

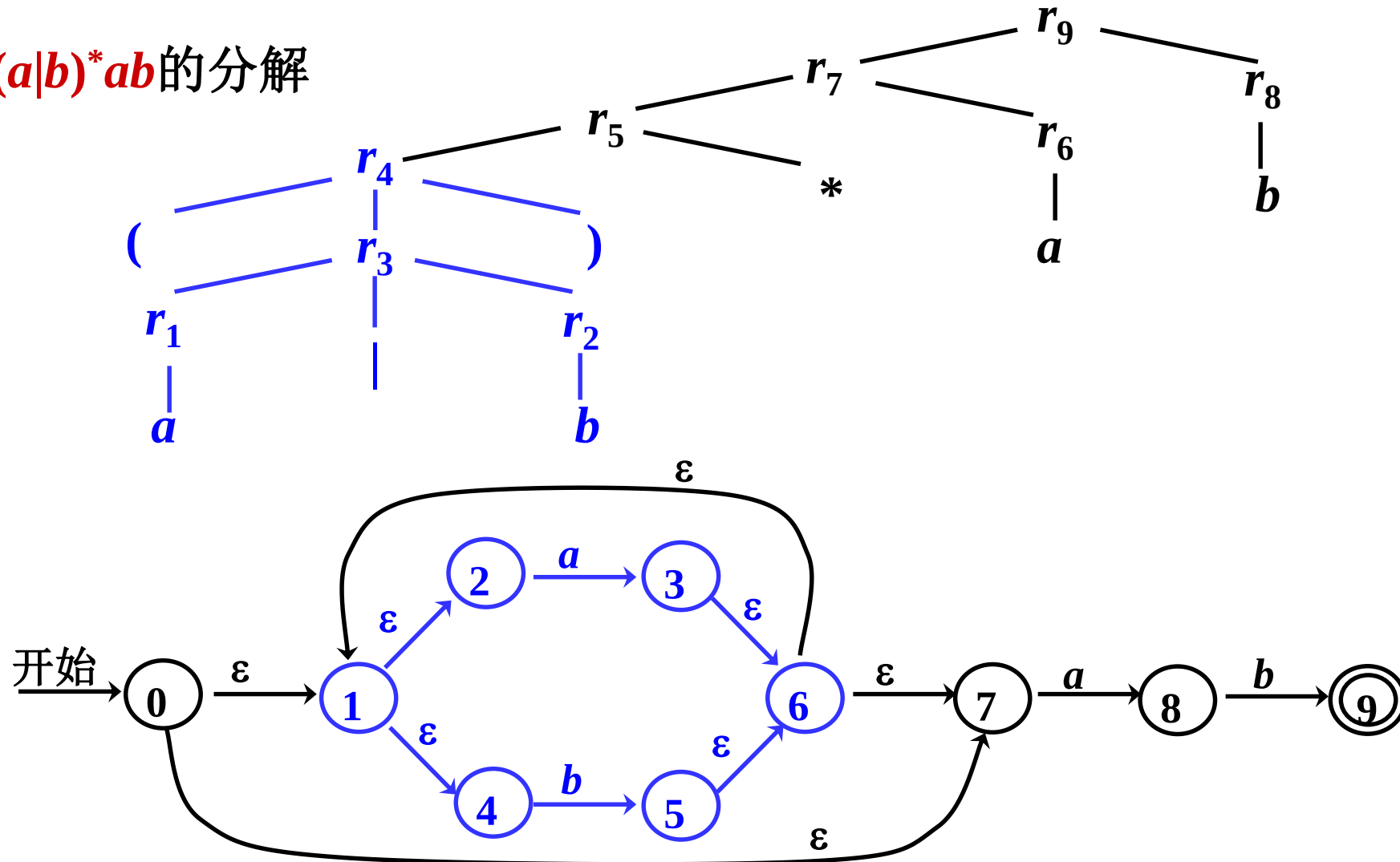
$(a|b)^*ab$ 的分解





NFA构造过程举例

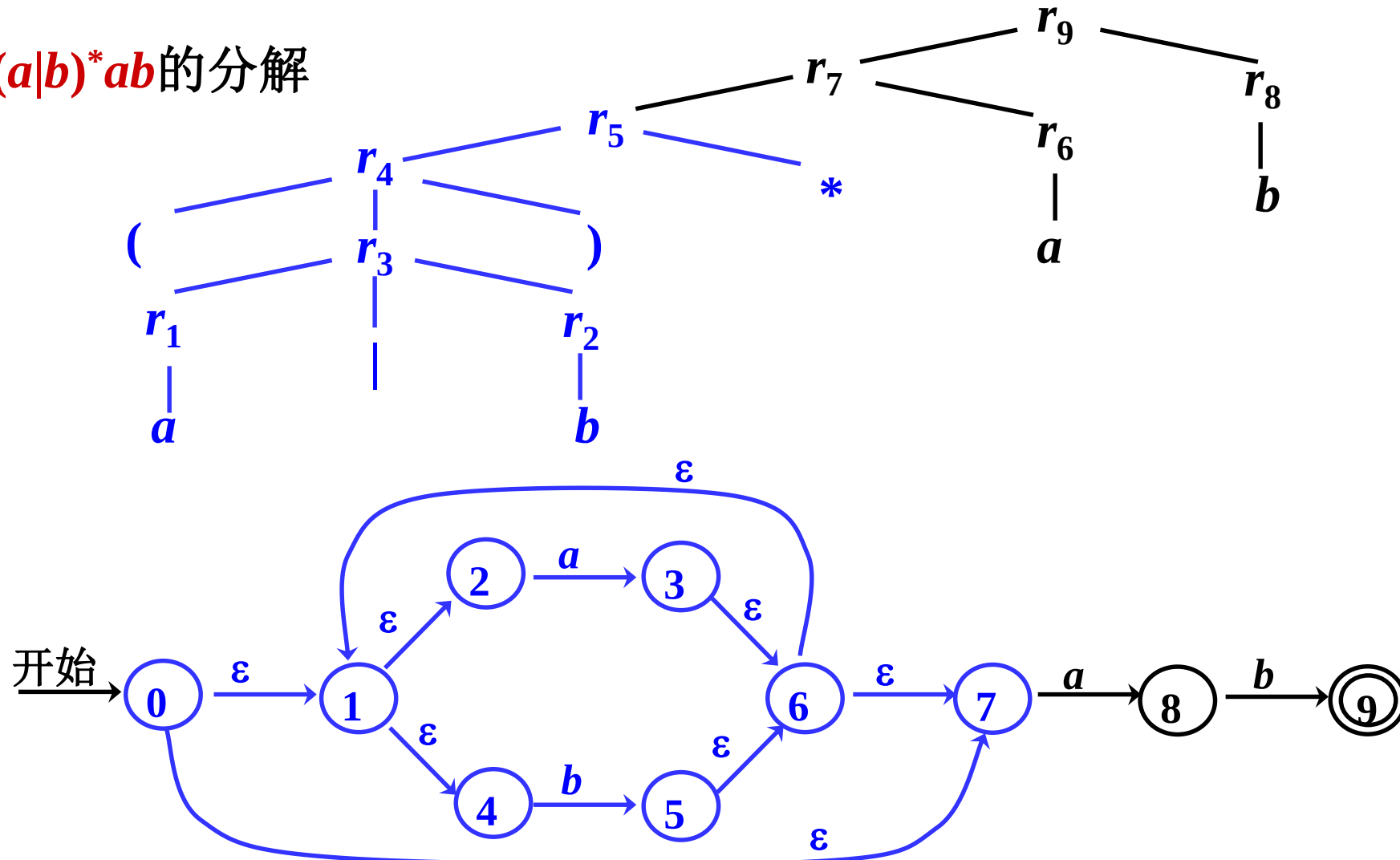
$(a|b)^*ab$ 的分解





NFA构造过程举例

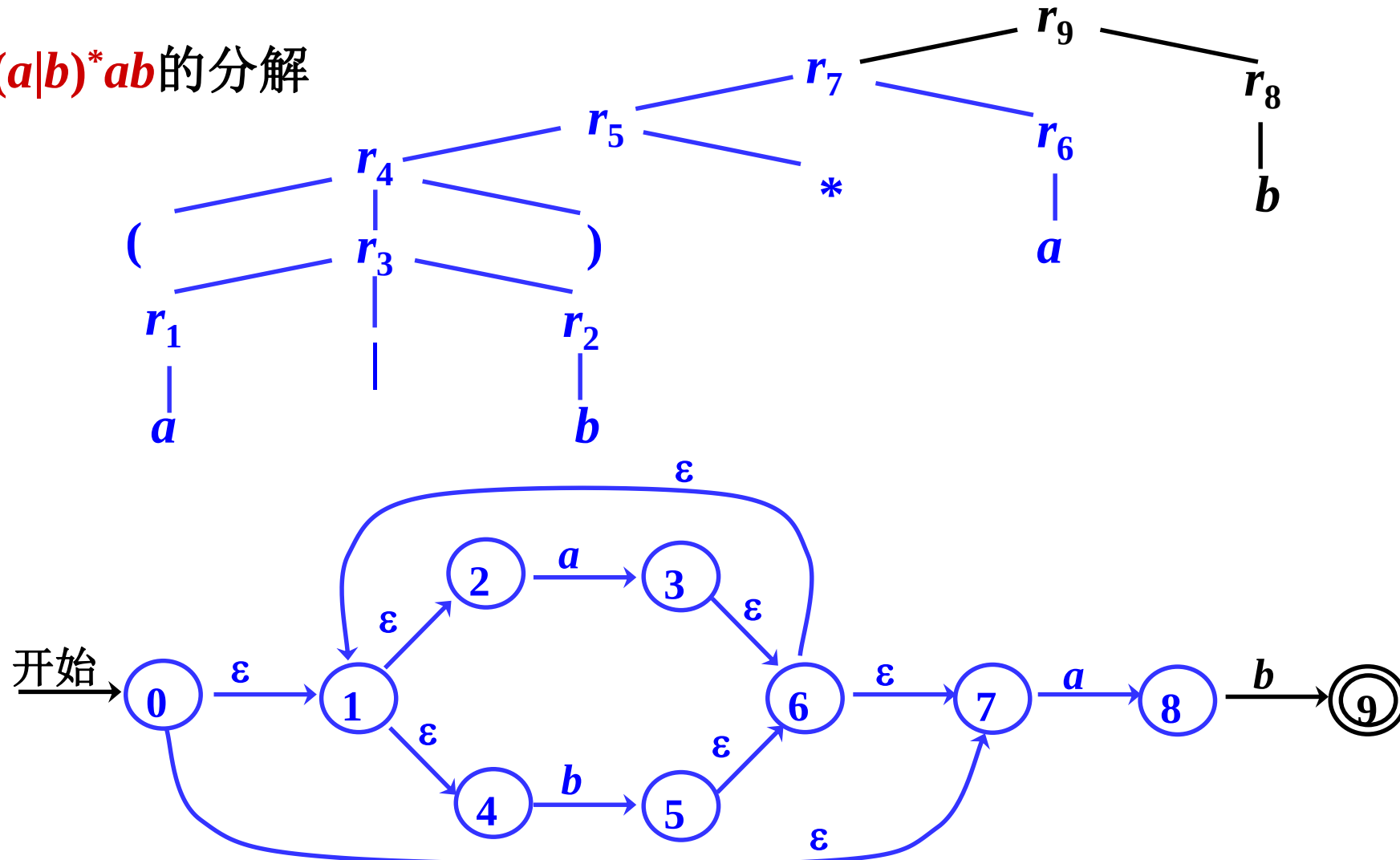
$(a|b)^*ab$ 的分解





NFA构造过程举例

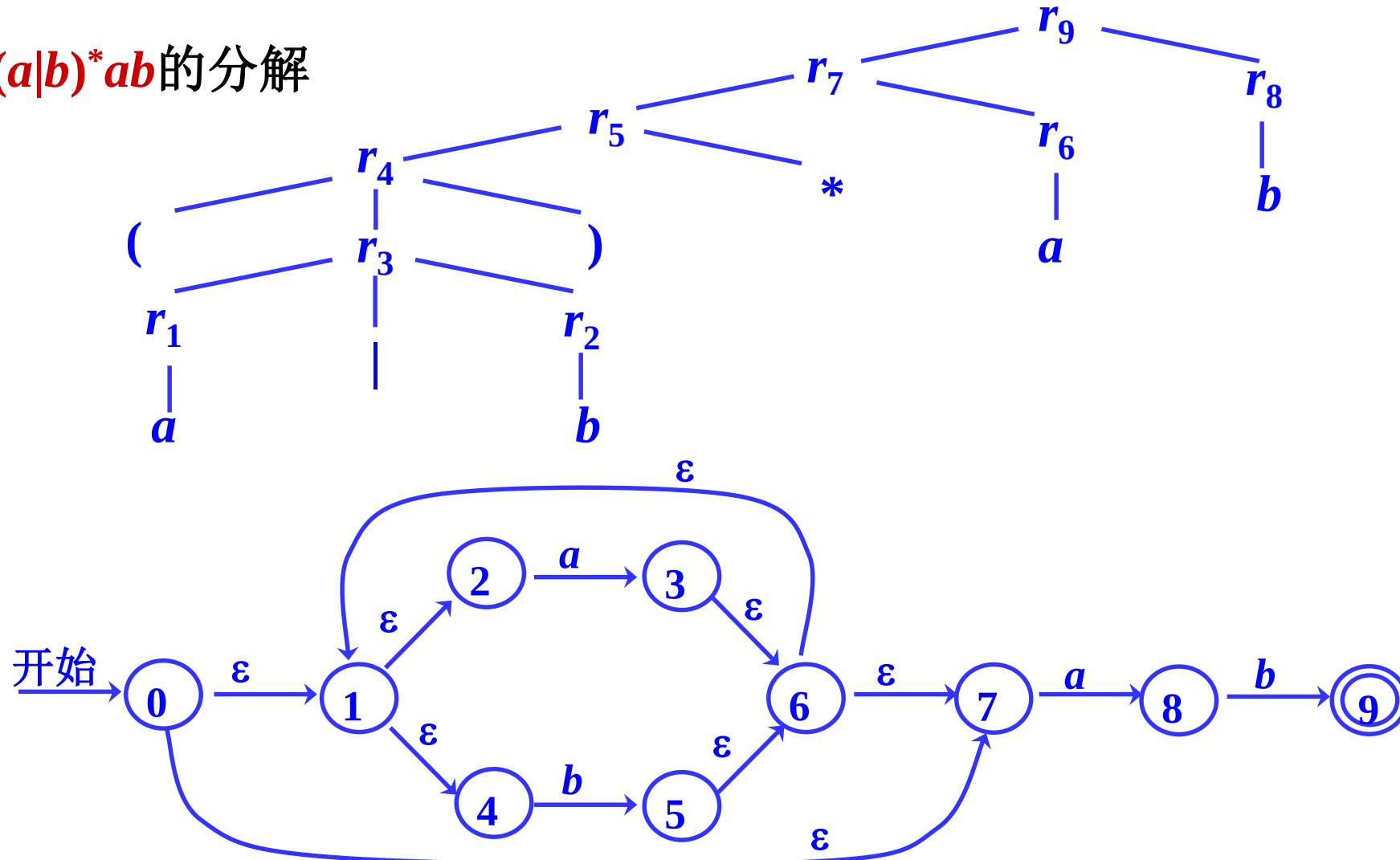
$(a|b)^*ab$ 的分解





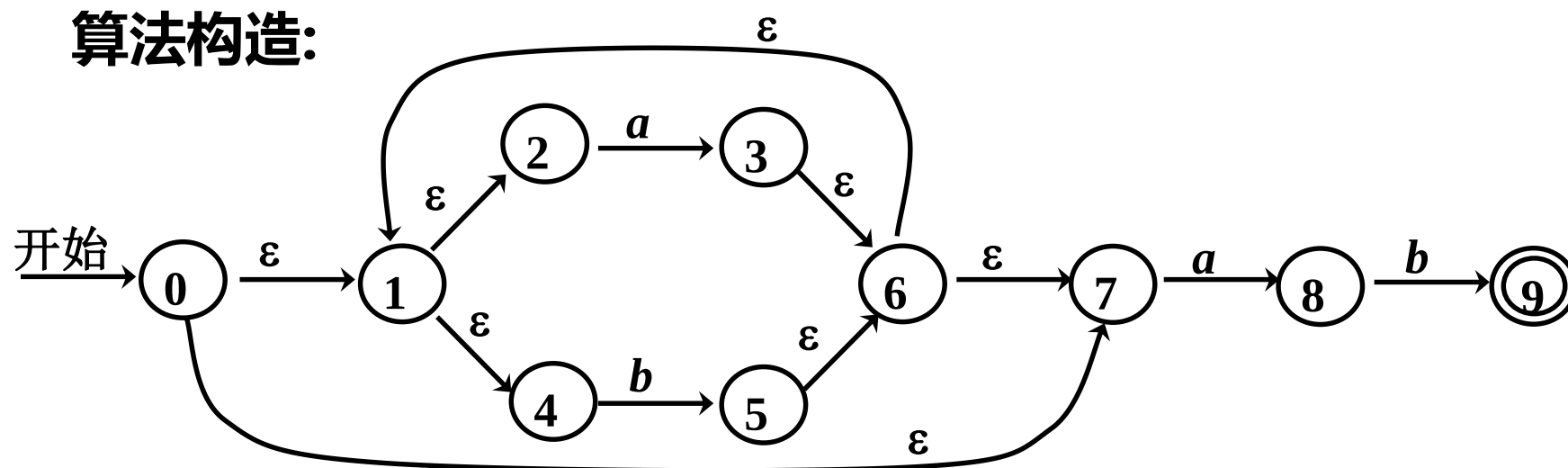
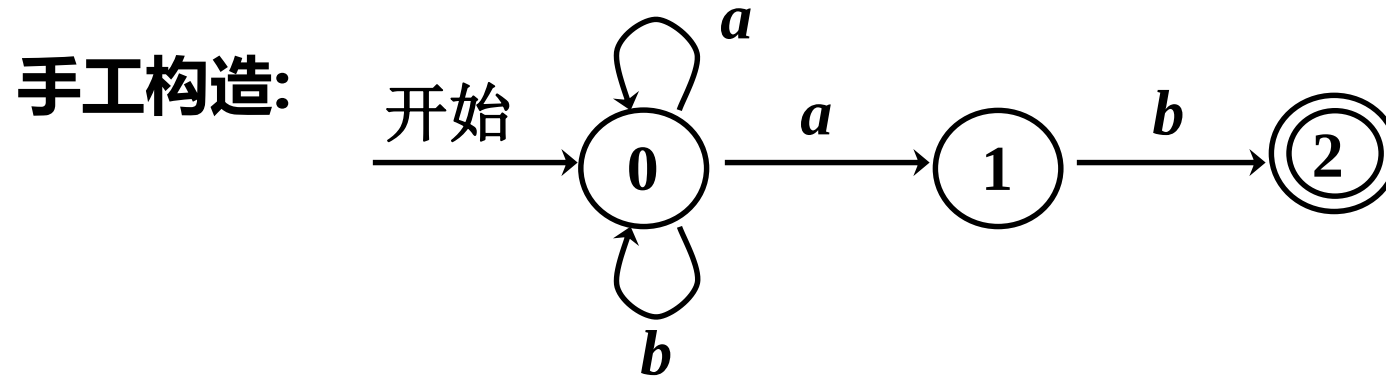
NFA构造过程举例

$(a|b)^*ab$ 的分解



NFA的手工构造和算法构造

□ $(a|b)^*ab$ 的两个NFA的比较





词法分析器的自动生成技术

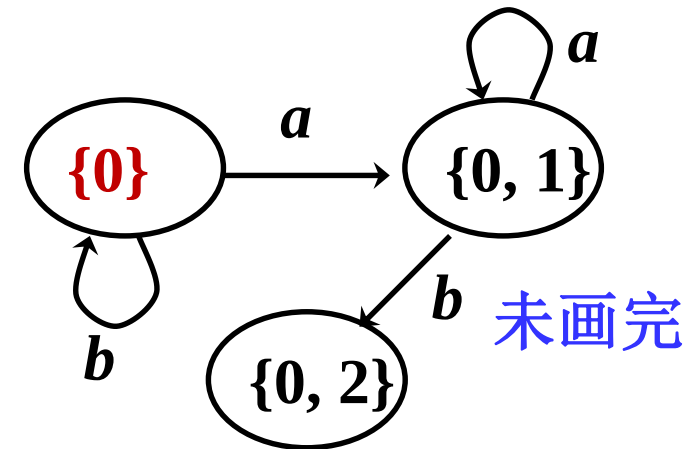
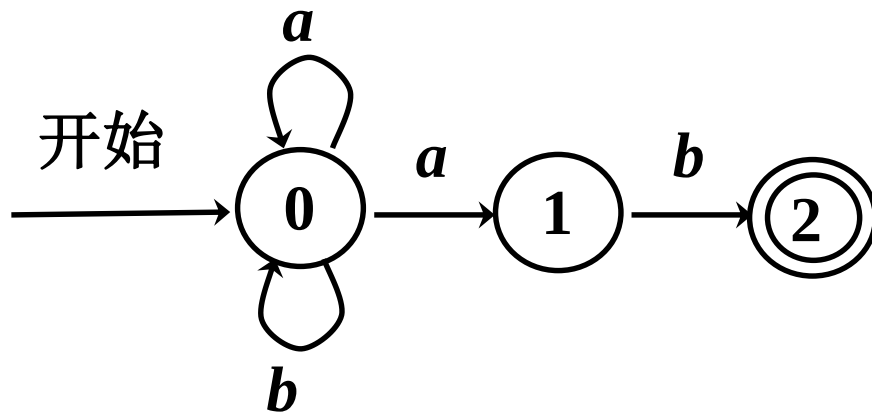
- 正规式：描述语言的词法
- 有限自动机：刻画词法分析的实现

- 词法分析器自动生成的主要过程
 - 正规式→NFA（语法制导的构造算法）
 - NFA→DFA (子集构造法)
 - DFA化简
 - 根据DFA构造词法分析器源码

NFA到DFA的变换

□ 子集构造法(subset construction)

1. **DFA的一个状态**是**NFA的一个状态集合**
2. DFA的开始状态是包含NFA的开始状态的状态集合
3. 读了输入 a_i 后, NFA能到达的所有状态: $s_1, s_2, \dots, s_k$, 则 DFA到达一个状态, 对应于NFA的 $\{s_1, s_2, \dots, s_k\}$

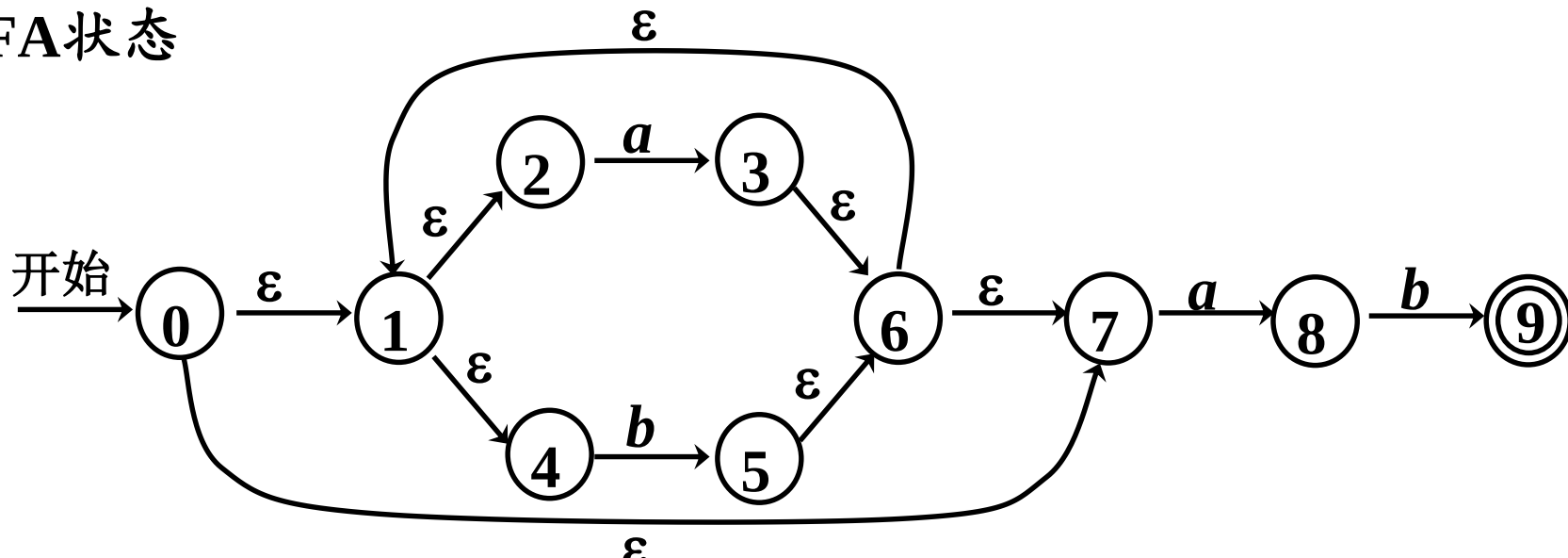




NFA到DFA的变换

□ 子集构造法(subset construction)

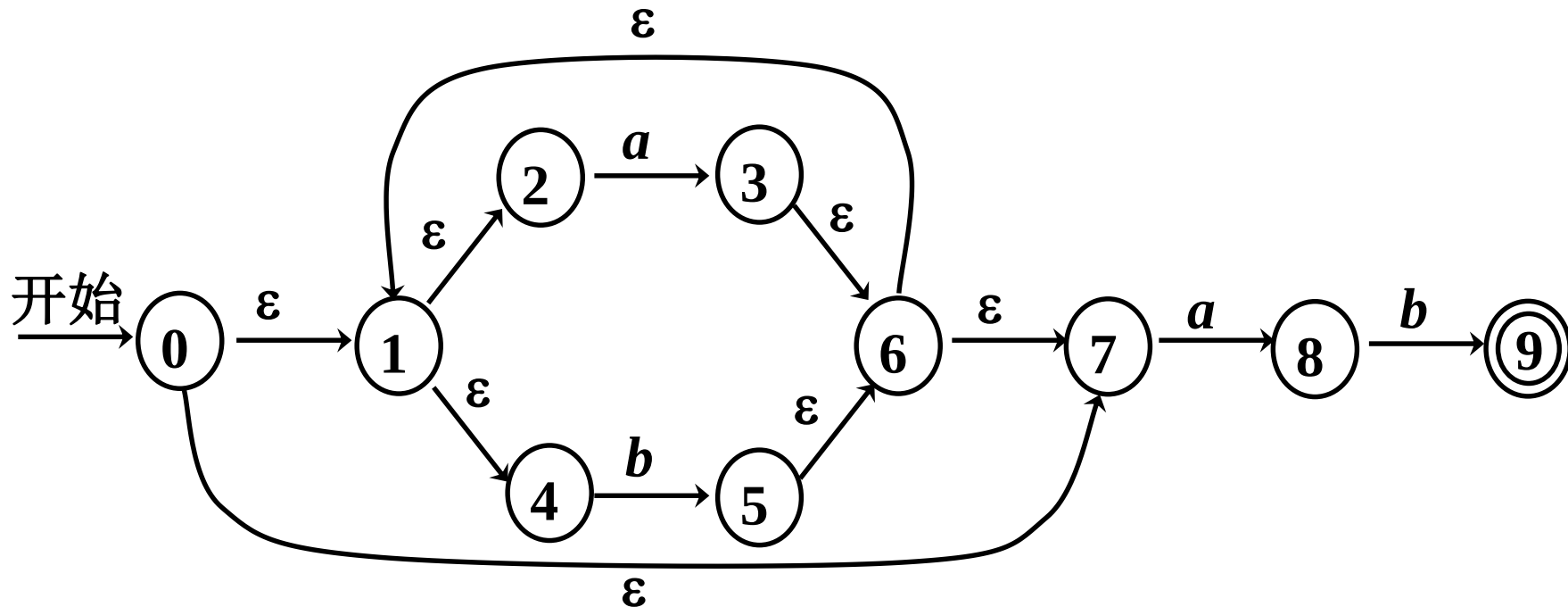
1. **ϵ -闭包 (ϵ -closure)** 状态 s 的 ϵ -闭包是 s 经 ϵ 转换所能到达的状态集合
2. **NFA的初始状态的 ϵ -闭包**对应于DFA的初始状态
3. 针对每个DFA 状态 ——NFA状态子集 A , 求输入每个 a_i 后能到达的NFA状态的 ϵ -闭包并集, 该集合对应于DFA中的一个已有状态, 或者是一个要新加的DFA状态





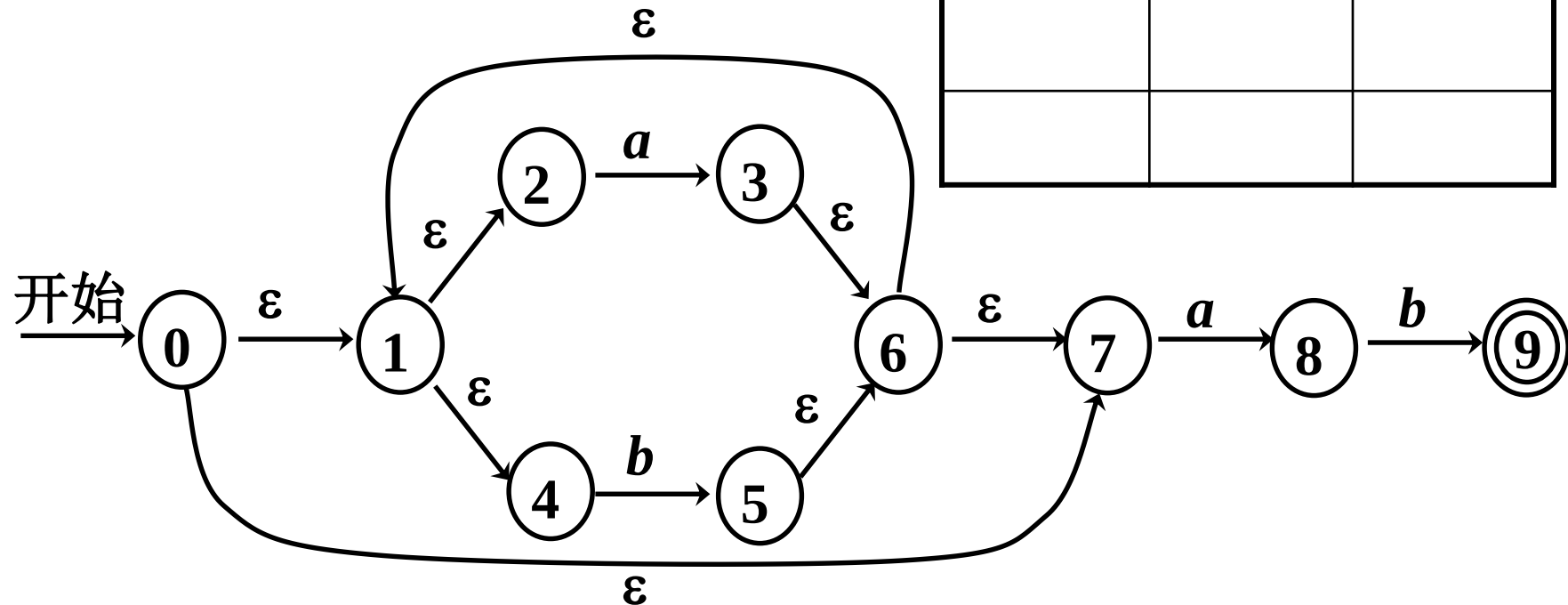
例题6

正规式 $(a|b)^*ab$ 对应的NFA如下，把它变换为DFA





例题6

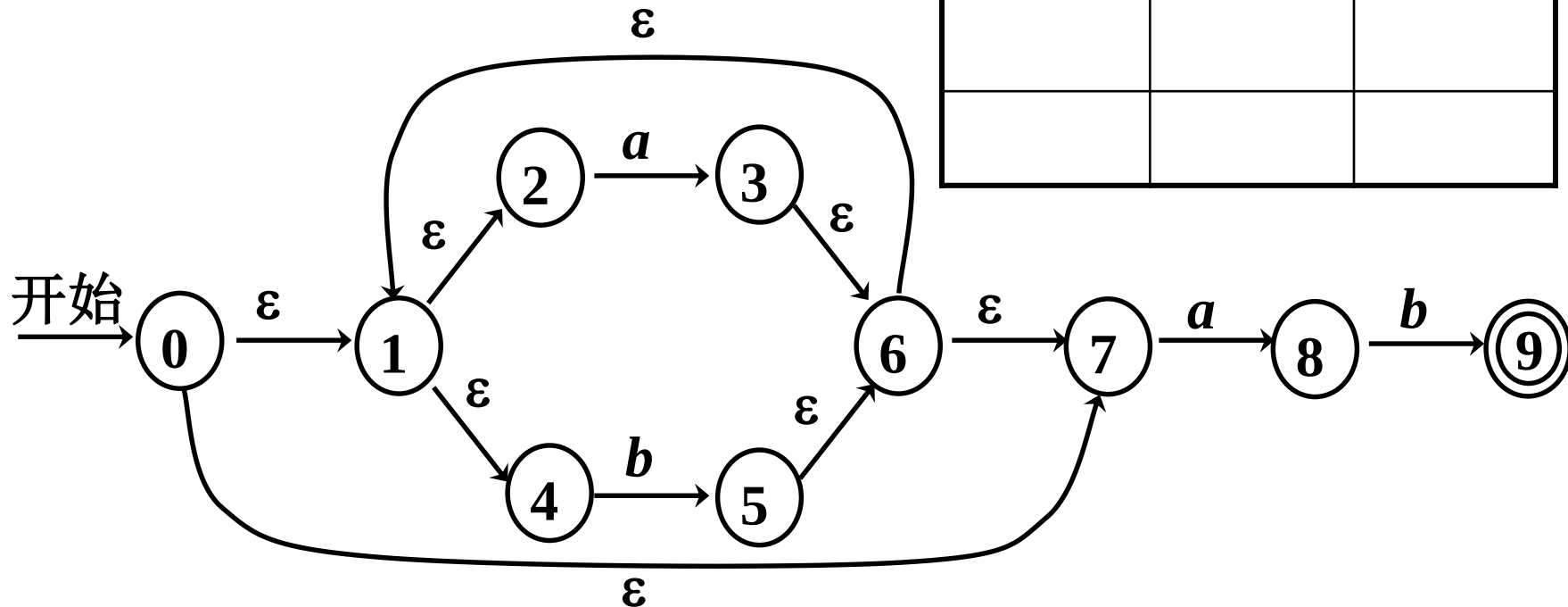




例题6

$A = \{0, 1, 2, 4, 7\}$

状态	输入符号	
	a	b
A		



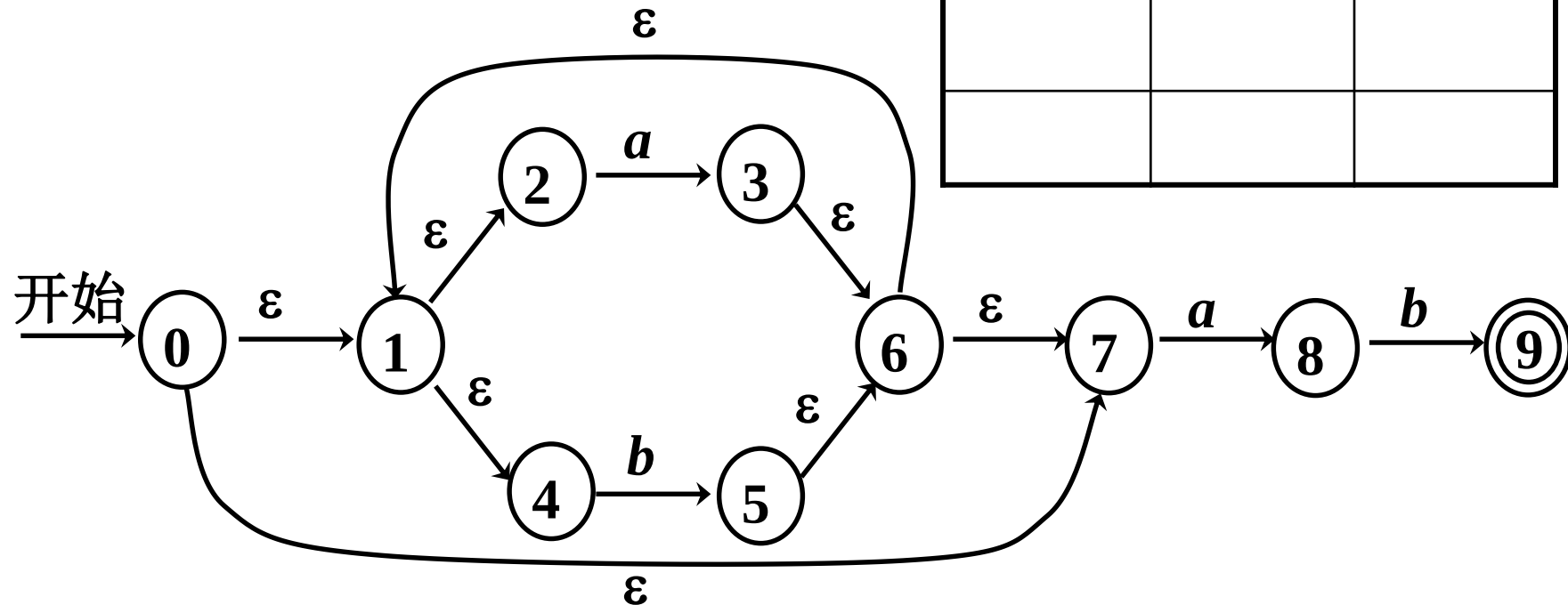


例题6

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

状态	输入符号	
	a	b
A	B	



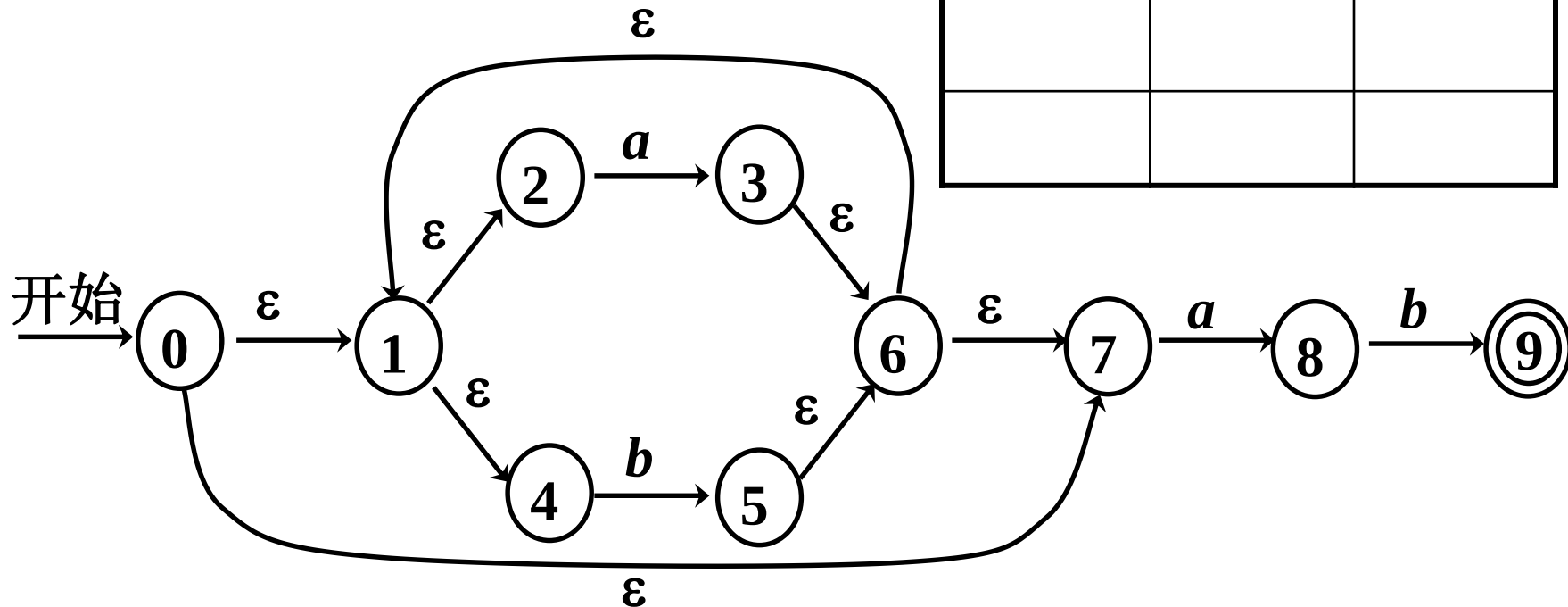


例题6

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

状态	输入符号	
	a	b
A	B	
B		





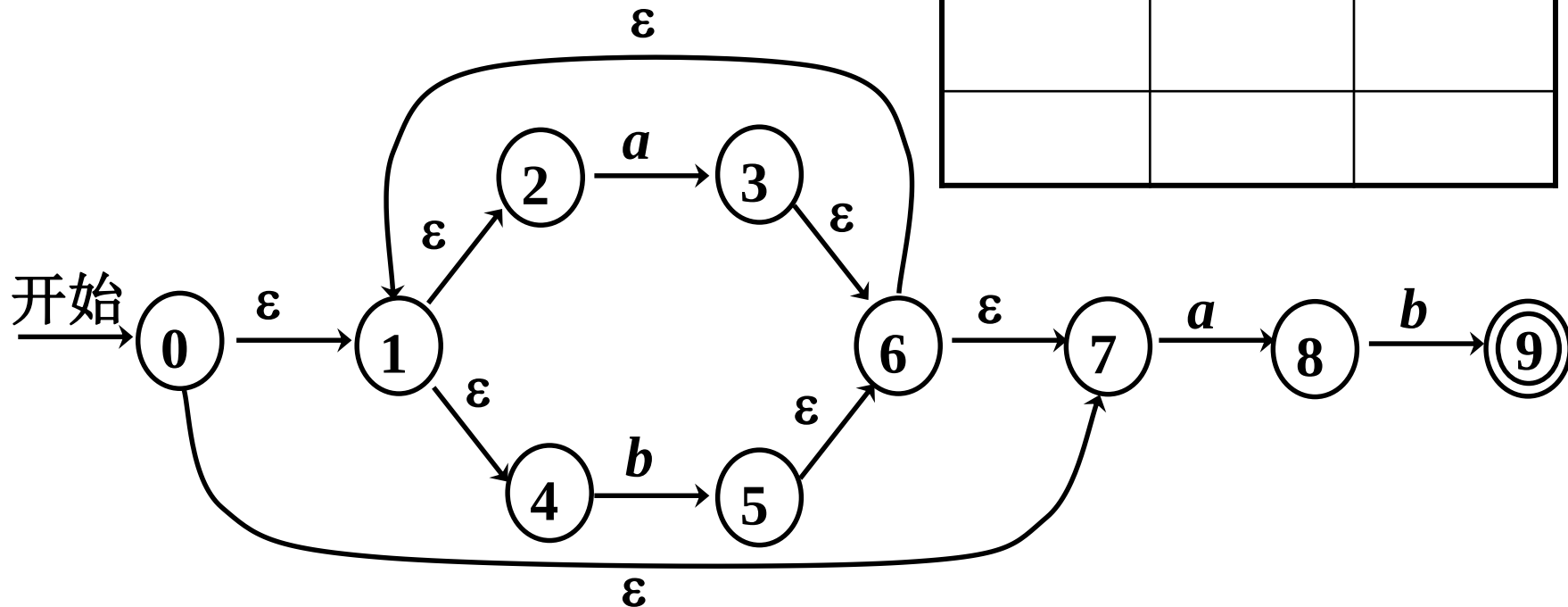
例题6

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

状态	输入符号	
	a	b
A	B	C
B		





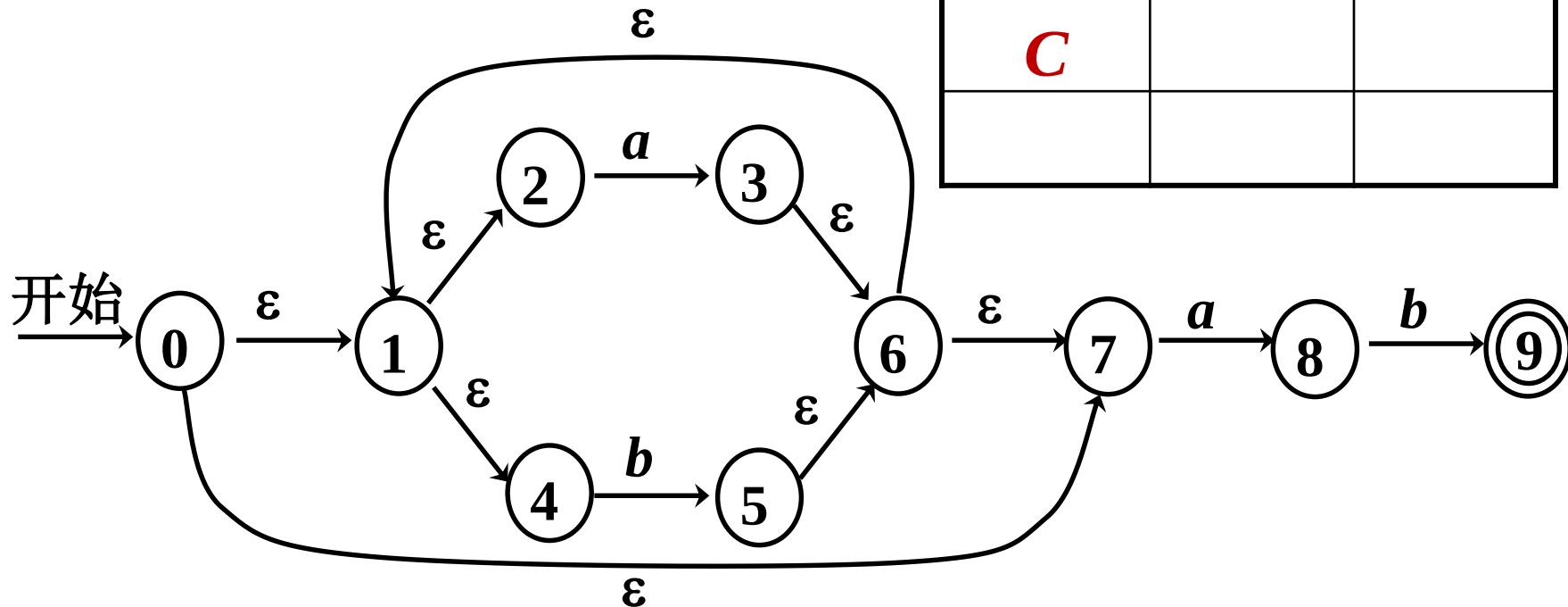
例题6

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

状态	输入符号	
	a	b
A	B	C
B		
C		





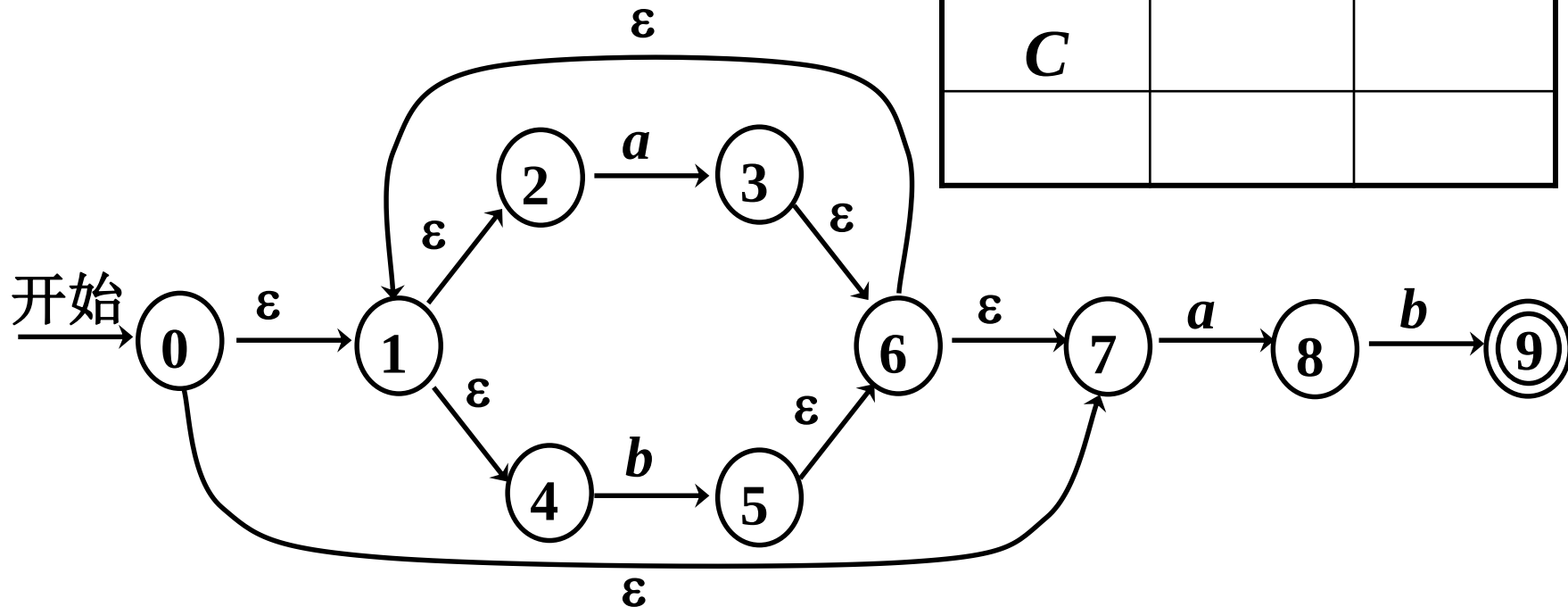
例题6

$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

状态	输入符号	
	a	b
A	B	C
B	B	
C		





例题6

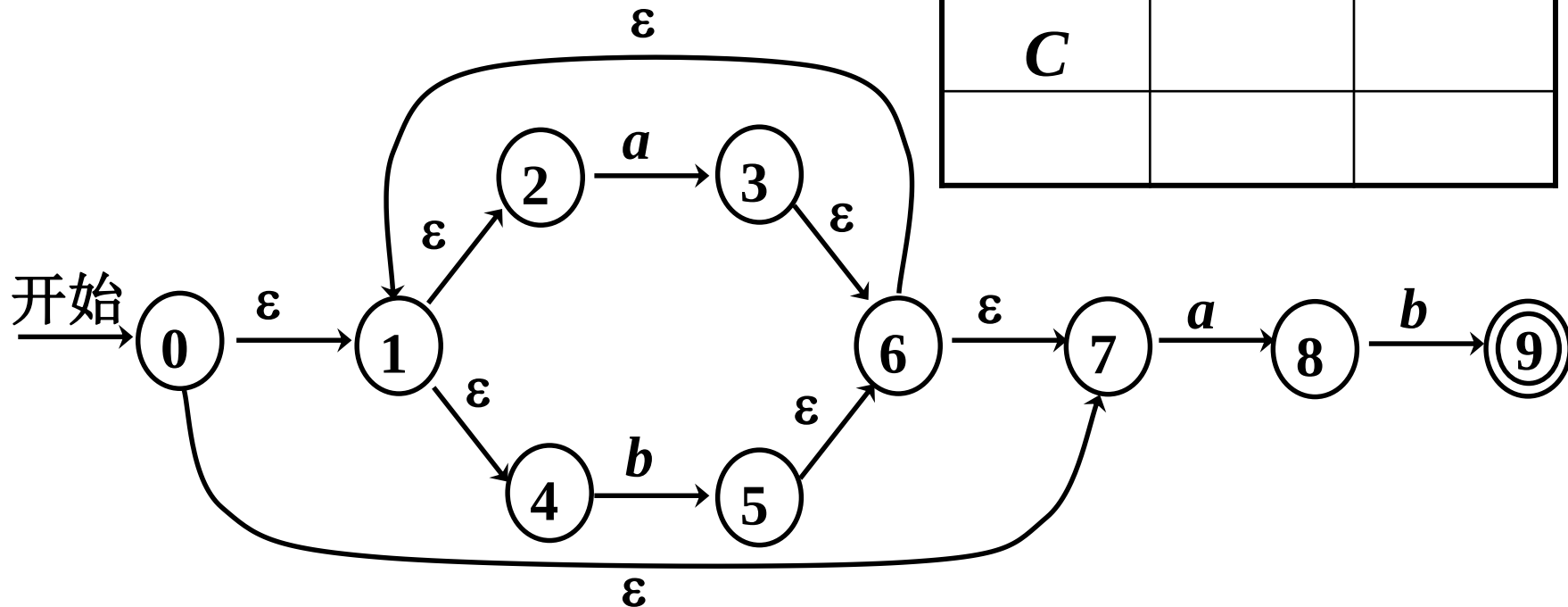
$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

$D = \{1, 2, 4, 5, 6, 7, 9\}$

状态	输入符号	
	a	b
A	B	C
B	B	D
C		





例题6

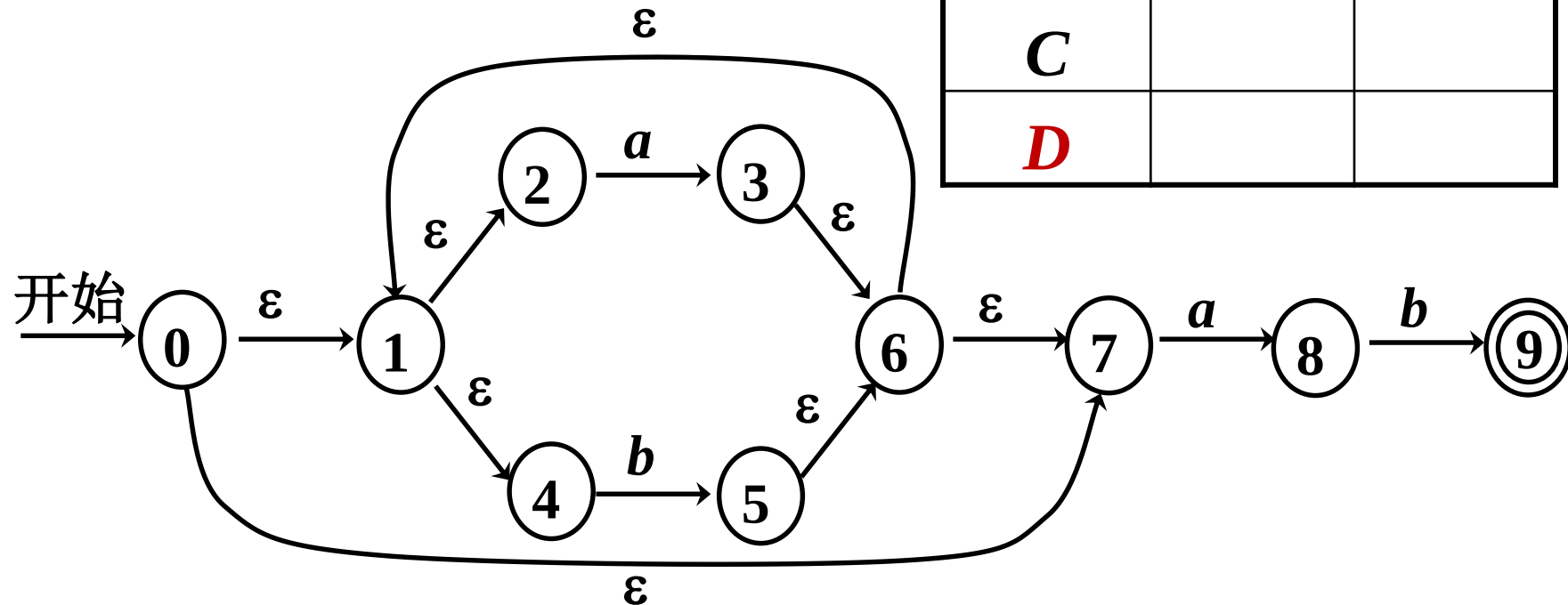
$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

$D = \{1, 2, 4, 5, 6, 7, 9\}$

状态	输入符号	
	a	b
A	B	C
B	B	D
C		
D		





例题6

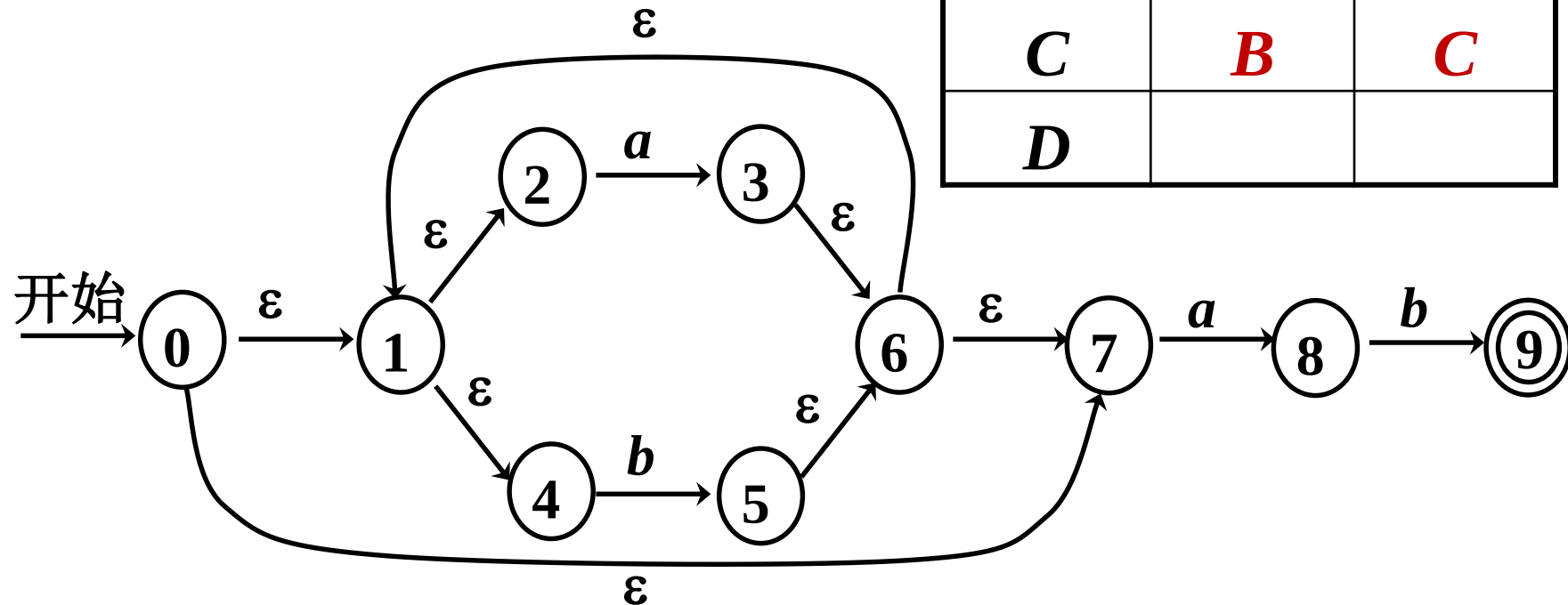
$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

$D = \{1, 2, 4, 5, 6, 7, 9\}$

状态	输入符号	
	a	b
A	B	C
B	B	D
C	B	C
D		





例题6

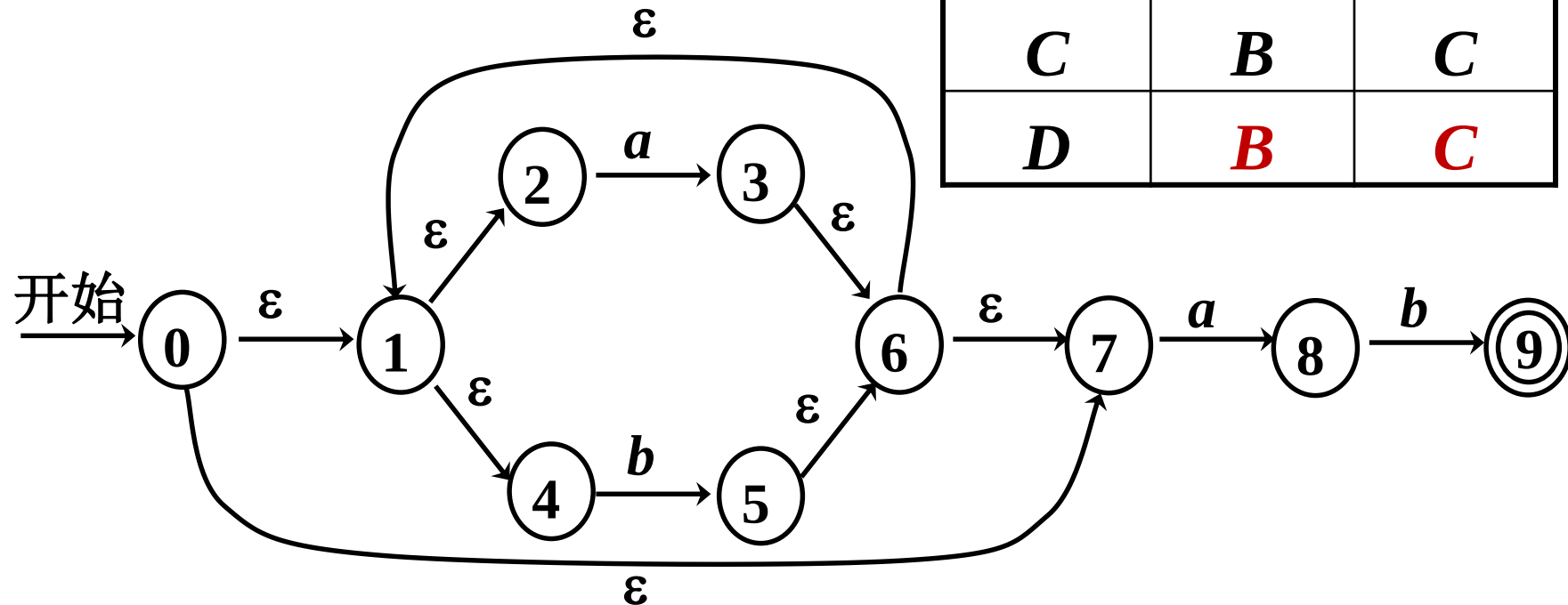
$A = \{0, 1, 2, 4, 7\}$

$B = \{1, 2, 3, 4, 6, 7, 8\}$

$C = \{1, 2, 4, 5, 6, 7\}$

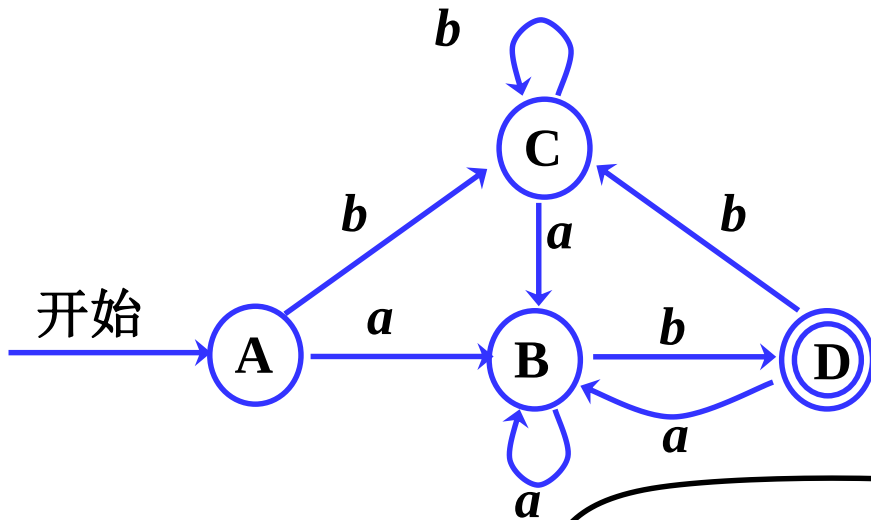
$D = \{1, 2, 4, 5, 6, 7, 9\}$

状态	输入符号	
	a	b
A	B	C
B	B	D
C	B	C
D	B	C

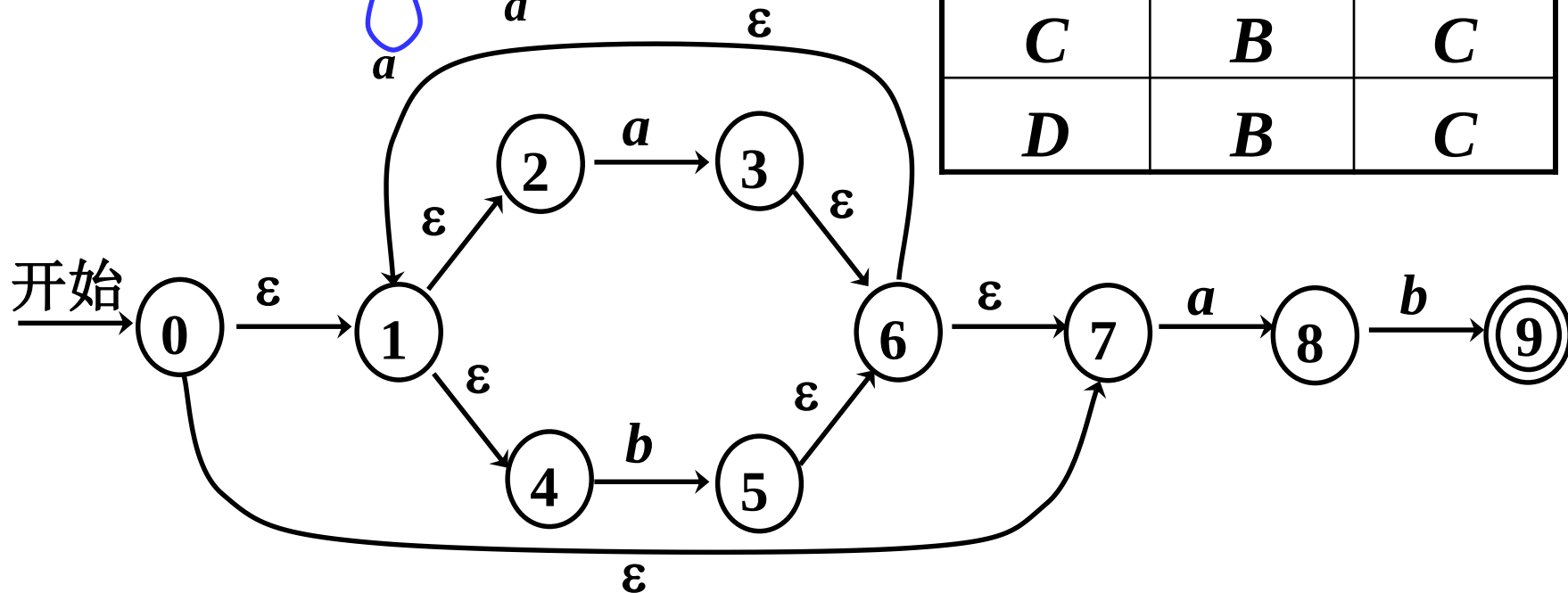




例题6

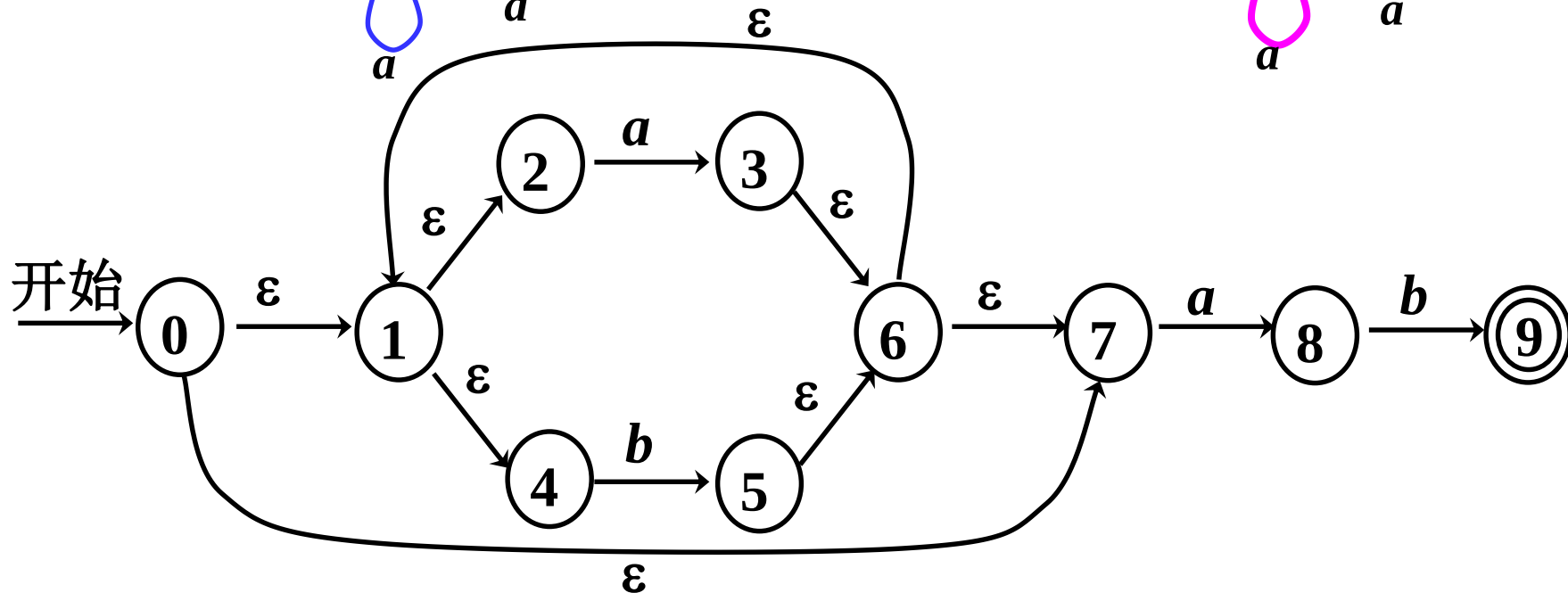
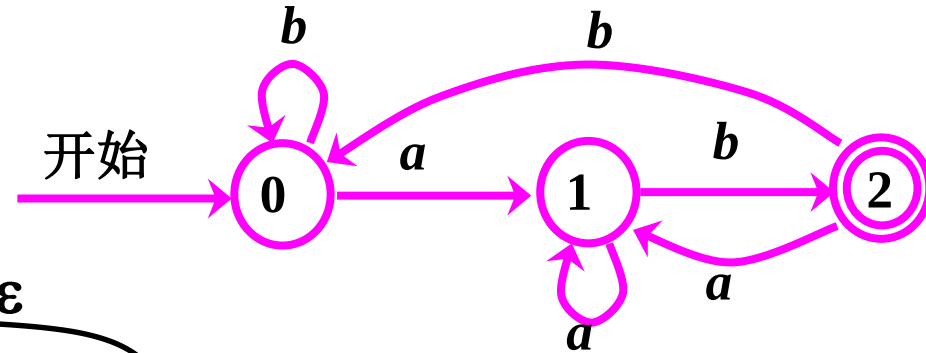
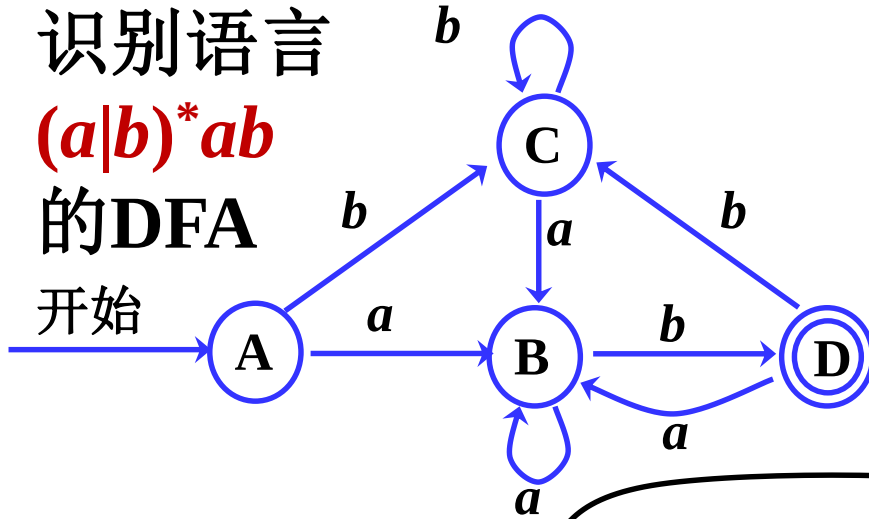


状态	输入符号	
	<i>a</i>	<i>b</i>
<i>A</i>	<i>B</i>	<i>C</i>
<i>B</i>	<i>B</i>	<i>D</i>
<i>C</i>	<i>B</i>	<i>C</i>
<i>D</i>	<i>B</i>	<i>C</i>



例题6

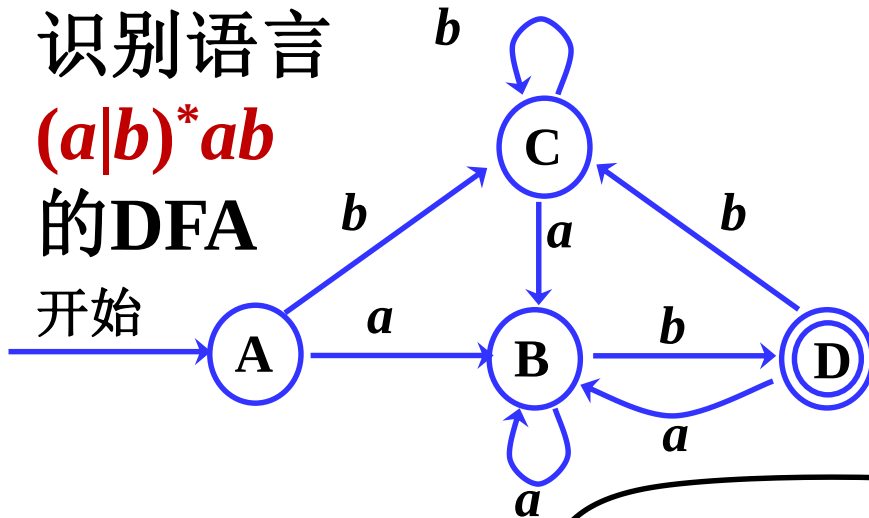
识别语言
 $(a|b)^*ab$
的DFA



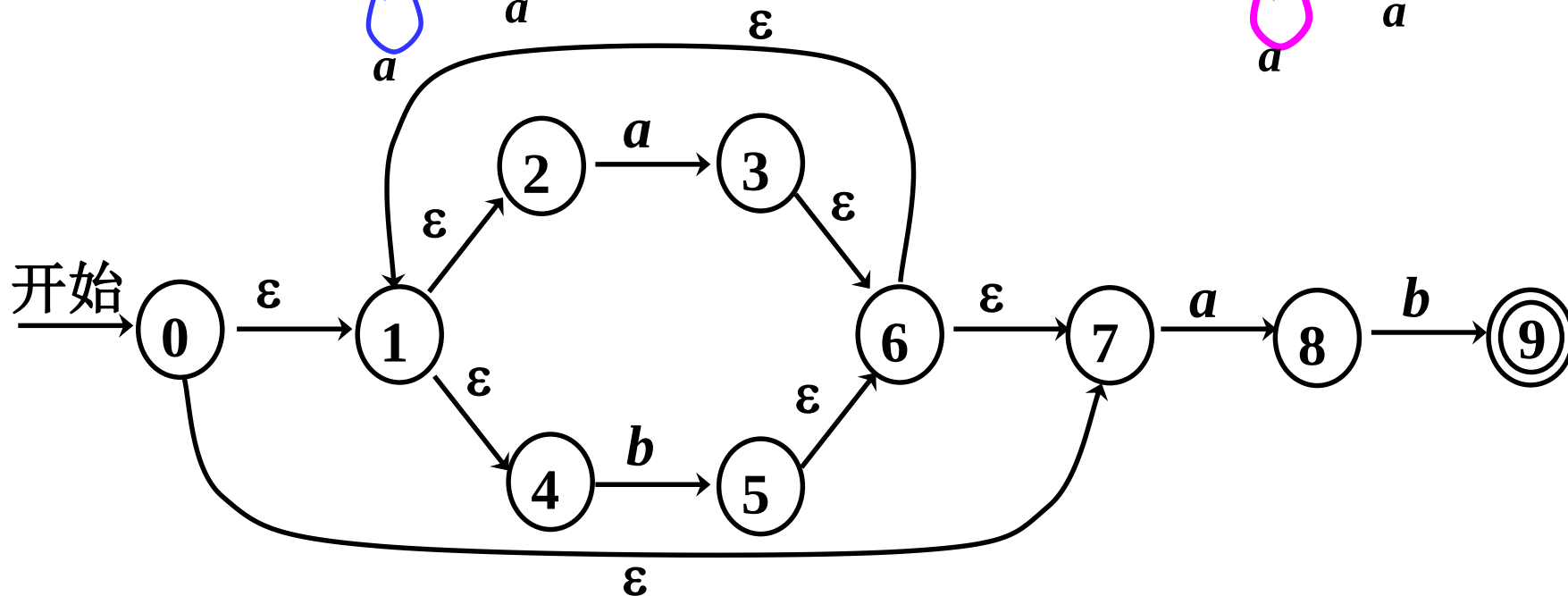
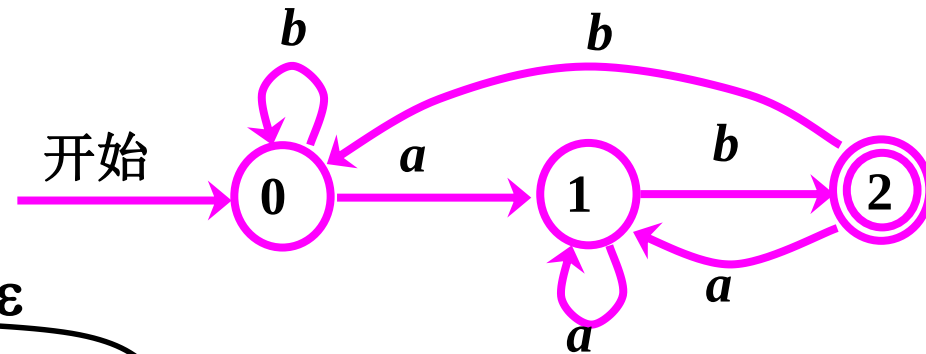


例题6

识别语言
 $(a|b)^*ab$
的DFA



子集构造法不一定
得到最简DFA





词法分析器的自动生成技术

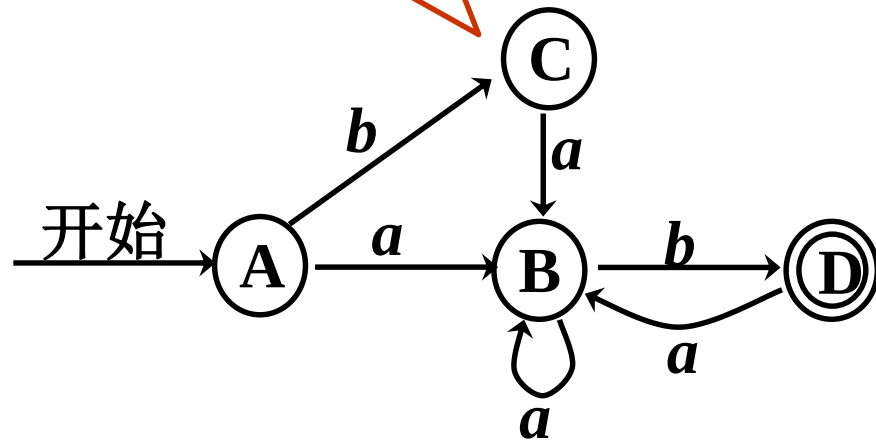
- 正规式：描述语言的词法
- 有限自动机：刻画词法分析的实现

- 词法分析器自动生成的主要过程
 - 正规式→NFA（语法制导的构造算法）
 - NFA→DFA (子集构造法)
 - DFA化简
 - 根据DFA构造词法分析器源码

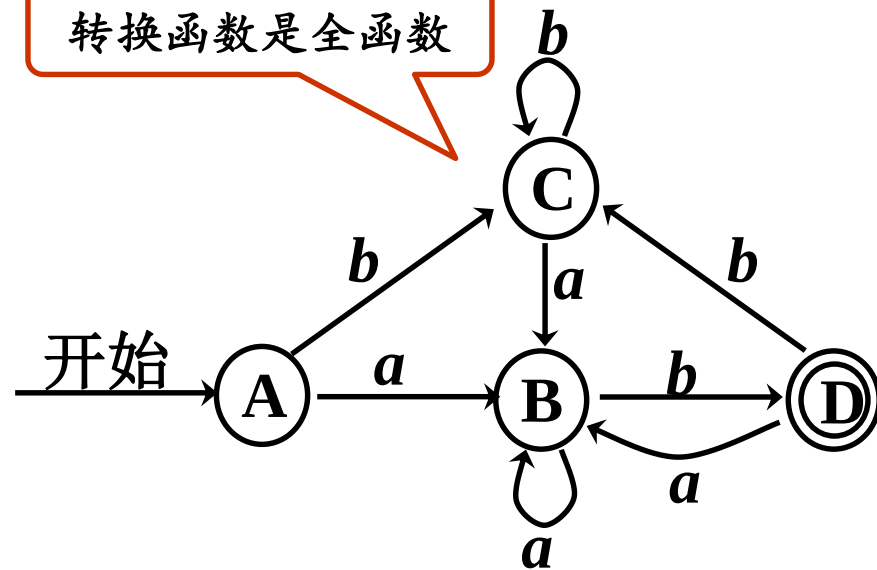
DFA的化简

□ 该方法用于化简转换函数是**全函数**的DFA

转换函数是部分函数



转换函数是全函数



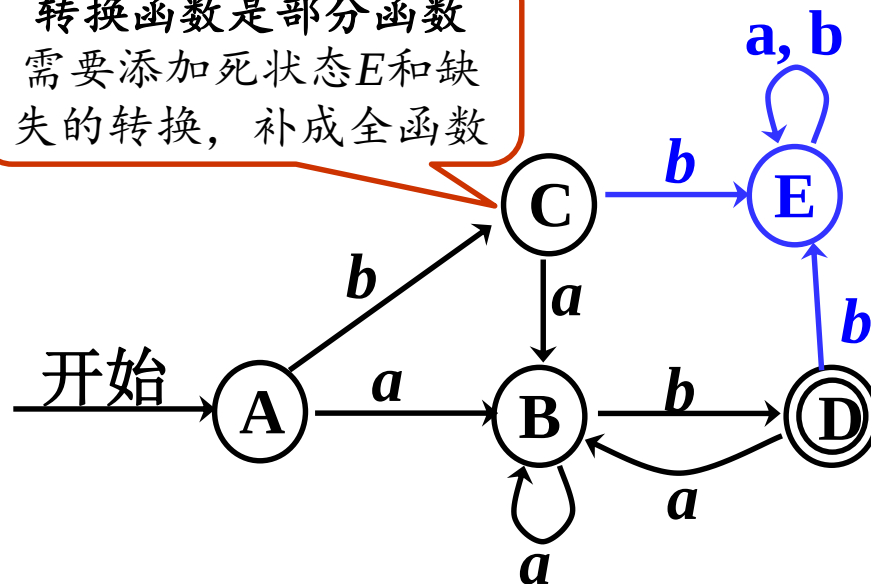
DFA的化简

□ 该方法用于化简转换函数是**全函数**的DFA

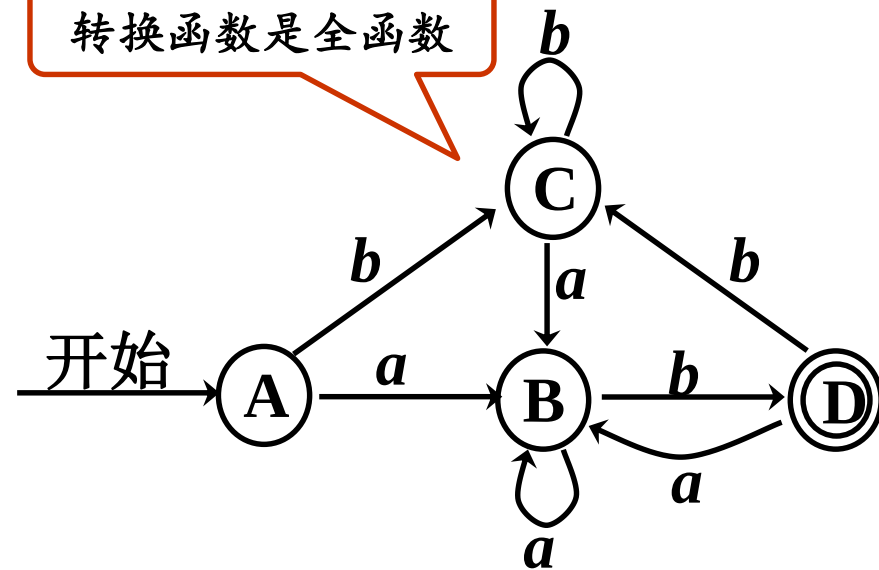
□ 死状态 (dead state)

- 当DFA的转换函数由部分函数改成全函数表示时，要在左图引入**死状态E**，将缺失的状态转换都指向该死状态

转换函数是部分函数
需要添加死状态E和缺失的转换，补成全函数



转换函数是全函数





DFA的化简

□ 可区别的状态(distinguishable states) s 和 t

将状态分成不相交的子集(初始按是否为接受状态来划分),

某子集中的状态 s 、 t **可区别**是指, 存在一个输入符号 w , 使得它们分别到达的状态落入当前划分中的不同子集。

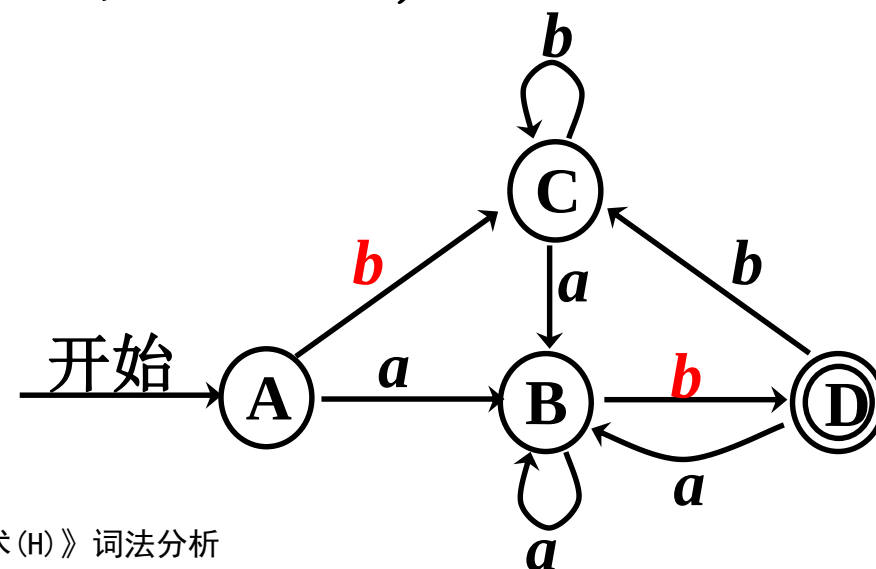
■ A 和 B 是可区别的状态:

从A出发, 读入 b 后到达非接受状态C; 从B出发, 读过 b 后到达接受状态D

■ A 和 C 是不可区别的状态:

无任何输入符号可用来区别它们

可区别的状态
要分开对待





DFA的化简

□ 方法

1. 按状态**是否接受**来划分状态集合

$\{A, B, C\}, \{D\}$

2. 考察每个子集(至少含2个状态)的状态转换

$move(\{A, B, C\}, a) = \{B\}$

$move(\{A, B, C\}, b) = \{C, D\}$ //面临**b**,转换到的状态落入不同子集

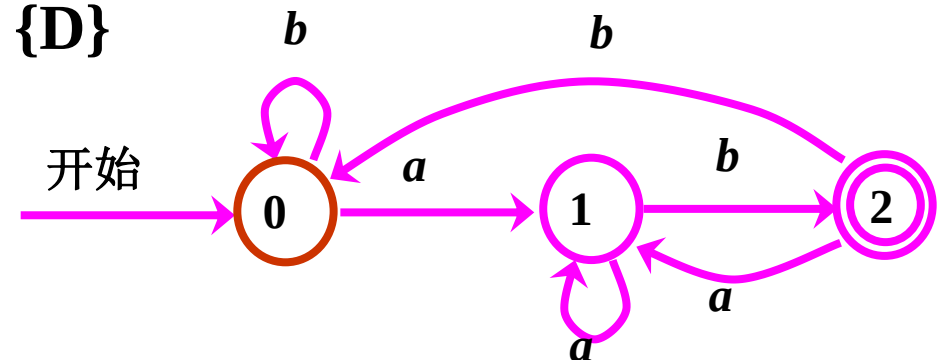
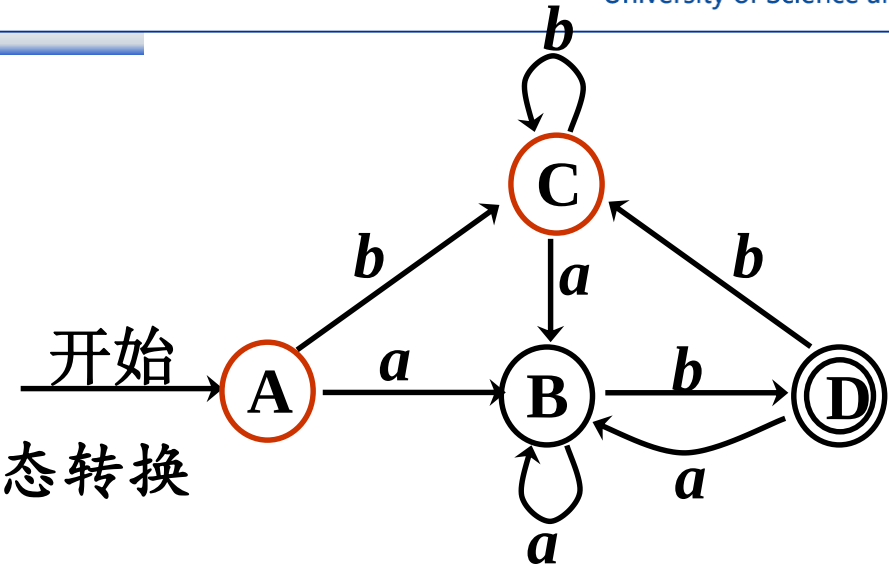
说明 **$\{A, C\}$** , **$\{B\}$** 是可区别的

3. 继续分解, 当前划分为 **$\{A, C\}$** , **$\{B\}$** , **$\{D\}$**

4. 考察 **$\{A, C\}$** , 不可区分, 合并状态

$move(\{A, C\}, a) = \{B\}$

$move(\{A, C\}, b) = \{C\}$





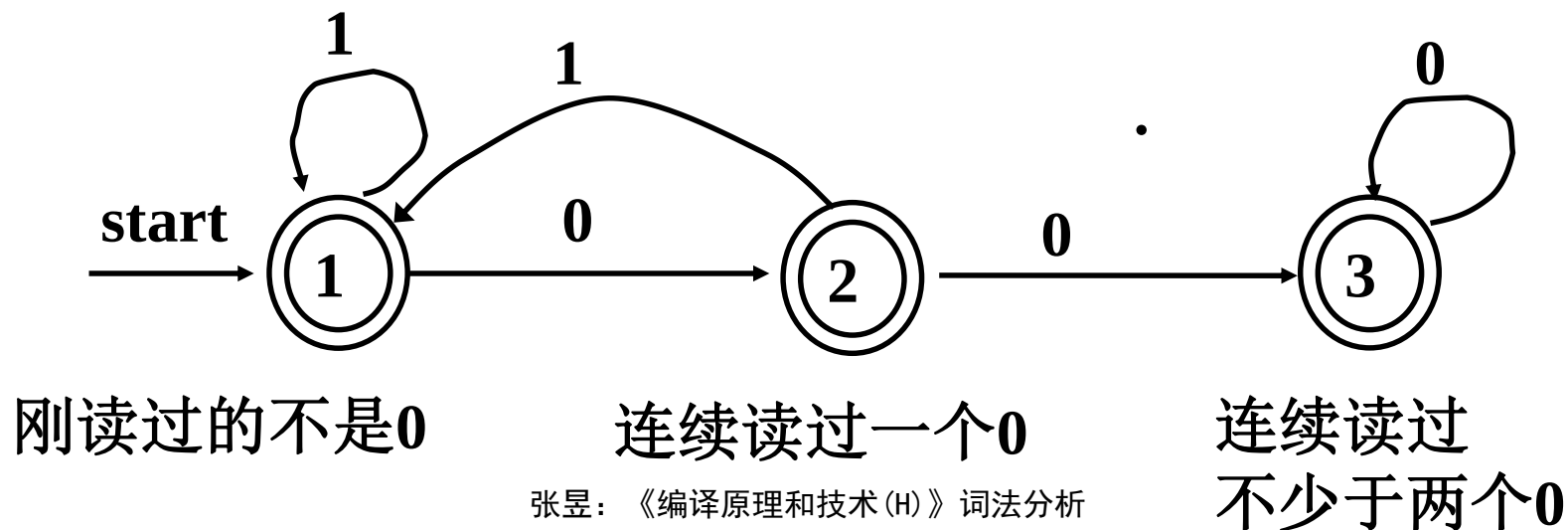
例题7

叙述下面的正规式描述的语言，并画出接受该语言的最简DFA的状态转换图

$(1|01)^* 0^*$

解答

描述的语言是，所有不含子串001的、由0和1组成的串



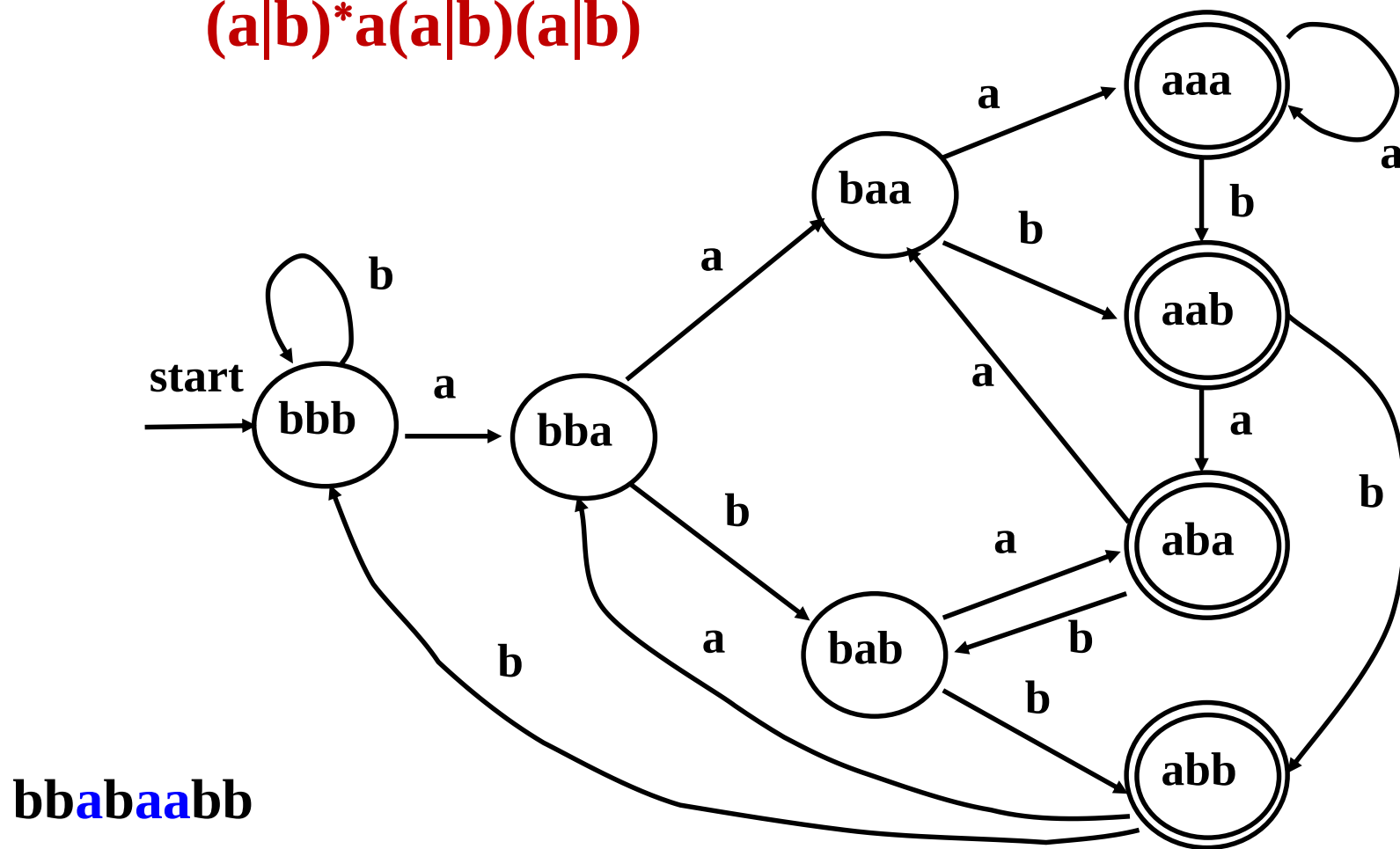


例题8

用状态转换图表示接受如下正规式的DFA

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

解答





本章要点

- 词法分析器的作用和接口，用高级语言编写词法分析器等
- 掌握下面的相关概念，它们之间转换的技巧、方法或算法
 - 非形式描述的语言 $\leftrightarrow$ 正规式
 - 正规式 $\rightarrow$ NFA
 - 非形式描述的语言 $\leftrightarrow$ NFA
 - NFA $\rightarrow$ DFA
 - DFA $\rightarrow$ 最简DFA
 - 非形式描述的语言 $\leftrightarrow$ DFA（或最简DFA）