**编译原理实践第4次课**

**（从NFA到DFA）**

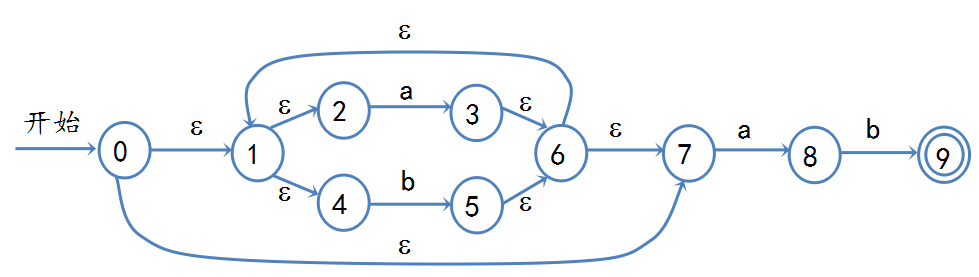
1. 基于子集构造法，从NFA构建DFA

1）输入为基于NFA的状态转换表，如下图所示（和上次课得到的NFA状态转换表相同）

|  |  |  |
| --- | --- | --- |
| **起始状态** | **符号** | **结束状态** |
| 0 | a | 1 |
| 1 | b | 2 |

2）输出为基于DFA的状态转换表

1. 核心是要实现eclosure（S）函数，即从状态集合S出发经过ε可以达到的所有状态集合。
2. 完成(a|b)\*ab的NFA到DFA的转换，NFA如下图所示：



1. 给定一个字符串，通过遍历生成的DFA，判断是否可到到达结束状态，如果可以到达，则接受，否则不接受。

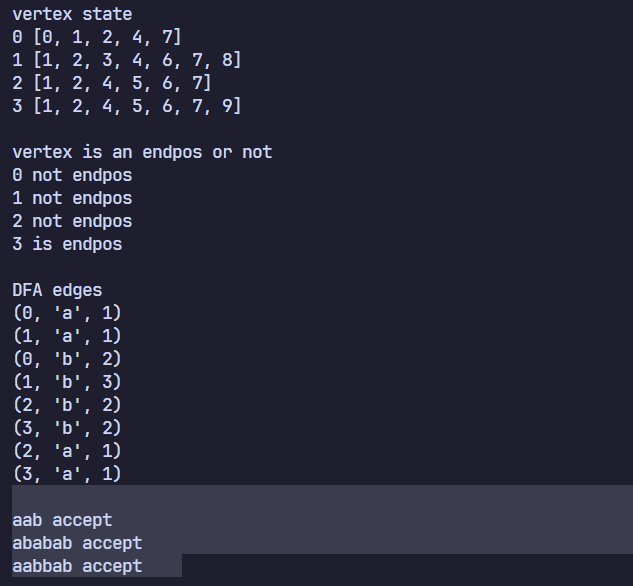
DFA为上一题生成得到，字符串序列如下：

1）aab

2）ababab

3）aabbab

运行结果：

****

**代码：**

**#! /usr/bin/env python**

**#coding=utf-8**

**S= [(0, 'eps', 1), (1, 'eps', 2), (2, 'a', 3), (3, 'eps', 6), (1, 'eps', 4), (4, 'b', 5), (5, 'eps', 6), (6, 'eps', 1), (0, 'eps', 7), (6, 'eps', 7), (7, 'a', 8), (8, 'b', 9)]**

**def e\_closure(p):**

**T = []**

**T1 = p**

**while len(T) != len(T1):**

**T = T1**

**for q in T:**

**for a, sym, c in S:**

**if a == q and sym=='eps' and c not in T1:**

**T1.append(c)**

**T1.sort()**

**return T1**

**def move(p, x):**

**T1 = []**

**for q in p:**

**for a, sym, c in S:**

**if a == q and sym == x and c not in T1:**

**T1.append(c)**

**T1 = e\_closure(T1)**

**return T1**

**re = [e\_closure([0])]**

**edge = []**

**def work(x): #x弧转换**

**p = 0**

**for i in re:**

**j = move(i, x)**

**if j not in re:**

**re.append(j)**

**p = 1**

**if (i, x, j) not in edge:**

**edge.append((i, x, j))**

**return p**

**while True:**

**p = work('a') + work('b')**

**if p == 0:**

**break**

**print('vertex state')**

**for i in re:**

**print(re.index(i), i)**

**print()**

**def get\_pos(t):**

**i = -1**

**for k in re:**

**i += 1**

**if t == k:**

**return i**

**DFA = []**

**for u, w, v in edge:**

**DFA.append((get\_pos(u), w, get\_pos(v)))**

**endpos = [9 in re[i] for i in range(len(re))] #标记终态**

**print('vertex is an endpos or not')**

**for i in range(len(endpos)):**

**print(i, 'is endpos' if endpos[i] else 'not endpos')**

**print()**

**print('DFA edges')**

**for i in DFA:**

**print(i)**

**print()**

**lis = ['aab', 'ababab', 'aabbab']**

**def check(string) -> bool:**

**queue = [(0, 0)]**

**while len(queue):**

**tmp = queue[0]**

**queue.pop(0)**

**if endpos[tmp[0]] and tmp[1] == len(string):**

**return True**

**if tmp[1] == len(string):**

**continue**

**for u, w, v in DFA:**

**if u == tmp[0] and w == string[tmp[1]]:**

**queue.append((v, tmp[1] + 1))**

**return False**

**for i in lis:**

**if check(i):**

**print(i, 'accept')**

**else:**

**print(i, 'not accept')**