**编译原理实践第1次课**

**正则表达式到NFA和DFA的转换   
(2周实践课内容)**

1. **编程环境 (不限)**

Python (2.7, 3.6)

C++

Java

1. **实现正则表达式至最简DFA的转换，包括:**

**1) MYT算法的实现 (正则表达式转NFA)**

**要求(以Python为例):**

python \*\*\*.py input.txt output.txt

其中:

1) \*\*\*.py: 您的MYT算法主程序

2) input.txt: 需要待转换的正则表达式. 为方便起见, 该文件只包含一行, 即只有一个正则表达式. 如:

|  |
| --- |
| a(b|c)\*d |

3) output.txt: 存放生成的NFA，其中以三元组形式存放，每行一个表示NFA的一条边，格式为 1 a 2, 表示状态1经过符号a到达状态2. 最后start state: 1表示状态1为开始状态; accepting states: 2 3表示状态2和3为接受状态.

|  |
| --- |
| 1 a 2 |
| …. |
| start state: 1  accepting states: 2 3 |

**2) 子集构造算法的实现 (NFA转DFA)**

**要求(以Python为例):**

python \*\*\*.py input.txt output.txt

其中:

1) \*\*\*.py: 您的子集构造算法主程序

2) input.txt: 需要待转换的NFA (格式见上面output.txt).

3) output.txt: 存放生成的DFA (格式与input.txt格式一致)

**3) DFA最小化**

**要求(以Python为例):**

python \*\*\*.py input.txt output.txt

其中:

1) \*\*\*.py: 您的DFA最小化算法主程序

2) input.txt: 需要待最小化的DFA (格式见上面output.txt).

3) output.txt: 存放最小化的DFA (格式与input.txt一致)

**注意: 为方便起见，用字符 e 表示空串ε, 并且假设输入的正则表达式不涉及符号e 。**

1. **提交源码，以及运行说明readme.pdf。readme.pdf中说明如何编译及运行，以及一个例子 (a|b)\*abb的例子运行结果。**