

**仿Xlex生成器**

**课 程： 编译原理**

**姓 名： 侯佳耀**

**学 号： 20182131055**

**班 级： 计科4班**

**设计时间：2021年11月2日**

|  |
| --- |
| **一．实验内容及目的**  1.实验内容：  设计一个应用软件，以实现将正则表达式-->NFA--->DFA-->DFA最小化-->词法分析程序。  2.必做实验要求：  （1）正则表达式应该支持单个字符，运算符号有： 连接 选择 闭包 括号  （2）要提供一个源程序编辑界面，让用户输入正则表达式（可保存、打开源程序）  （3）需要提供窗口以便用户可以查看转换得到的NFA（用状态转换表呈现即可）  （4）需要提供窗口以便用户可以查看转换得到的DFA（用状态转换表呈现即可）  （5）需要提供窗口以便用户可以查看转换得到的最小化DFA（用状态转换表呈现即可）  （6）需要提供窗口以便用户可以查看转换得到的词法分析程序（该分析程序需要用C语言描述）  （7）应该书写完善的软件文档  3.选做实验要求：  （1）扩充正则表达式的运算符号，如？、[ ]、 + 等 |
| **二．实验文档：**  1.设计思路  首先需要手动从对话框中输入或从文件读入正则表达式，然后通过用户主界面刷新后台表达式并选择相应功能。  a.正则表达式到NFA：  获取的正则表达式先判断是否含有禁止符号(即：~，意为空)，然后将正则表达式中的连接运算使用“.”表示，然后执行中缀转后缀操作，最后使用使用汤姆森构造法将正则表达式构造为NFA。 注：Thompson构造法：是[计算机科学](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6" \t "_blank)中是指一个能将[正则表达式](https://baike.baidu.com/item/%E6%AD%A3%E5%88%99%E8%A1%A8%E8%BE%BE%E5%BC%8F" \t "_blank)转化为一个与之等价的[非确定有限状态自动机](https://baike.baidu.com/item/%E9%9D%9E%E7%A1%AE%E5%AE%9A%E6%9C%89%E9%99%90%E7%8A%B6%E6%80%81%E8%87%AA%E5%8A%A8%E6%9C%BA" \t "_blank)（NFA）的[算法](https://baike.baidu.com/item/%E7%AE%97%E6%B3%95" \t "_blank)。算法得到的NFA可以在编程中用于匹配一个正则表达式，这也是正则表达式引擎实现的基本思路之一。其构造规则为通过[递归](https://baike.baidu.com/item/%E9%80%92%E5%BD%92" \t "_blank)地将一个正则表达式划分成构成它的子表达式，在得到每个子表达式对应的NFA之后，根据子表达式之间的运算关系和一系列规则构造表达式自身对应的NFA。对于正则表达式为ε或者只由一个符号构成的情况，则无需继续递归。非确定有限状态自动机：在[计算理论](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E7%90%86%E8%AE%BA" \t "_blank)中，非确定有限状态自动机或非确定有限自动机（NFA）是对每个状态和输入符号对可以有多个可能的下一个状态的[有限状态自动机](https://baike.baidu.com/item/%E6%9C%89%E9%99%90%E7%8A%B6%E6%80%81%E8%87%AA%E5%8A%A8%E6%9C%BA" \t "_blank)。 b.从NFA到DFA：  将获取到的NFA使用子集构造法转为DFA，然后使用二维数组记录下闭包运算的各个子集以便输入状态转移表，最后记录下各个节点的状态即可。  **注：**确定有穷自动机：（DFA）D是一个五元组：D=（K，Σ，M，S，F）其中  K：有穷非空的状态集合；  Σ：有穷非空的输入符号字母表；  M：转换函数，是在K×Σ→K上的映像，即，如 M(ki,a)=kj,(ki∈K,kj∈K)就意味着，当前状态为ki，输入符为a时，将转换为下一个状态kj，我们把kj称作ki的一个后继状态；且S∈K是唯一的一个初态；F⊂ K是非空的终态集合。  c.从DFA到最小DFA：  使用b步骤记录下来的各个子集的二维状态数组和状态表，对每一个状态进行分割，对分割所得子集不断遍历比较，判断是否可进行合并，以此最小化DFA。  d.词法分析程序：使用扫描生成chList的方法生成词法分析程序   1. 实验过程 2. 数据结构：     数据结构如图，由基本nfa状态类，NFA类和转换弧类组成。  b.GUI界面:  使用了Python的tkinte模块实现GUI 界面，大致界面如下图：  其中按钮功能如下：   1. 打开文件：即调出windows的文件选择框并使用askopenfile返回路径，在后台写入文件中的正则表达式并将str的值更新至Entry文本框（使用自建函数setTextInput实现），同时更新正则表达式的值。 2. 表达式转NFA：将正则表达式通过汤普森构造法转换为nfaStatusChart并显示在treeview中（nfa状态表为NxN的numpy二维数组） 3. NFA转DFA：通过子集构造法，自建函数（epClosure,aftertrans\_epClosure）分别求某状态的ep闭包和move(A,a)闭包，去重后填入DFA表（数据类型为DataFrame），并打印输入到treeview中。 4. DFA最小化：原本通过状态集划分遍历的方法实现，但是发现算法实现上有错误，所以进行了后续处理。具体在实验总结有体现。 5. 更新Entry值：认为在输入框中更改了字符串的值后，需要通过此按钮实现正则表达式值的更新。原因是发现Tkinter的按钮监控功能过于繁琐，故自己构建函数实现更新。 6. 关于：弹出作者和软件版本相关信息   c.实验测试：  输入框默认文本为“abc”，测试文件例子为“ab|c”。首先使用默认值测试，后打开文件测试，最后更改文本框值后测试  默认情况：    界面不完善，后台数据结构此时为：          打开测试文件：           1. 主程序：   采用pyinstaller 打包完整文件，无需额外准备电脑环境 |
| **四、实验总结**  通过本次实验，更加深入的学习了词法分析的相关知识，加深了对正则表达式和有穷自动机的理解。虽然通过理论课学习有了解正则表达式以及NFA、DFA之间的转换规则，但在真正编程实现过程中是非常不简单的，需要用得到大量数据结构的知识，也需要用到大量的算法。  因为时间不足，部分功能有bug，部分功能有瑕疵，在DFA最小化过程中，发现大部分NFA生成的DFA即为最小化的DFA（不清楚原因，可能是我算法不一样，生成闭包过程中有去重），故后来删除一部分代码。我也意识到在使用函数进行调用过程中，使用了过多的global变量，使得最后的函数联调十分困难，写法十分别扭，多个函数对同一个全局变量进行编辑也不安全，且重构困难。而且NFA表和DFA表的数据类型并不一致，且DFA表在生成过程中也全部使用的是NFA的数据结构，违背面向对象编程的原则。  我感觉实验总体难度不低，特别是数据结构的设计方面，起初设计的数据结构到后面写函数才发现十分的鸡肋，于是重构数据结构模型，导致整个实验思路十分混乱。代码逻辑乱飞，注释也不足。  想要实现自动化要付出的代价是十分高的，但是收益也是巨大的。编译原理正是连接人类日常编程和电脑底层的桥梁。 |
| 五、参考文献：  （1）CSDN blog  （2）编译原理及实践，机械工业出版社，Kenneth C.Louden著，冯博琴 冯岚等译  （3）数据结构（用C++面向对象语言描述） 清华大学第二版 |