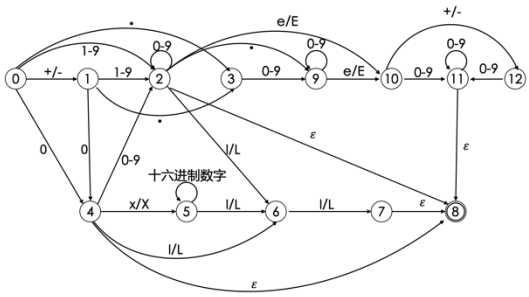
FSM 在字符串中的应用

姓名：白云鹏 学号：1004161221

一、 FSM 在编译原理中的应用

在编译原理中，FSM，更具体而言应是其中的 DFA，在词法分析中占据了很重要的地位。一个字符串能够被 DFA 识别，那么其在编译中即为正确的代码。而在编译原理中，我们认为一个语言应该是上下文无关的左线性文法，而一个左线性文法，其必然能够使用 DFA 来进行描述。而在接下来的叙述中，主要使用 DFA 完成了一个简单的程序，它能对所输入的字符串进行判定，判断其是否是一个合法的 C/C++式的常数表达式。

首先，先对 DFA 的状态转换图进行绘制以说明原理：



图表 1 识别 C/C++常数状态转换图

如图所示，比较关键的地方有以下两点：

1) 十六进制的判断

在起始状态读入一个字符 0 之后，转入状态 4，如果继续读人一个 x/X,则表明此字符串可能为十六进制数字。

2) 科学计数法的表示

在 2 状态（表示其前均为整数而无小数点）以及 9 状态（表示其前有小数点）时读人 e/E，转入 10 状态，表明此字符串可能为科学计数法表示的数字。

有了状态转换图之后，实现也更加简单，结果如图：

```
0x700
Stat:0 开始判断
Stat:4 这里需要判断是十进制还是十六进制
Stat:5 在这里读入全部的十六进制数字
Stat:5 在这里读入全部的十六进制数字
Stat:5 在这里读入全部的十六进制数字
Stat:5 在这里读入全部的十六进制数字
Stat:8 字符串可接受
Yes, it is.
-1.33
Stat:0 开始判断
Stat:1 字符前有+/-号
Stat:2 在这里需要读入全部剩下的整数部分
Stat:9 这里需要读入剩下全部的小数部分
Stat:9 这里需要读入剩下全部的小数部分
Stat:9 这里需要读入剩下全部的小数部分
Stat:8 字符串可接受
Yes, it is.
```

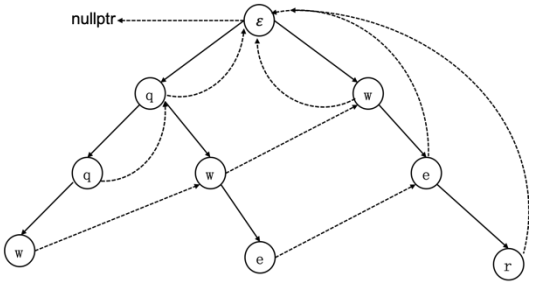
```
-1e3
Stat:0 开始判断
Stat:1 字符前有+/-号
Stat:2 在这里需要读入全部剩下的整数部分
Stat:10 这可能是一个科学计数法表示的字符串
Stat:11 这里则需要读入全部的e指数
Stat:8 字符串可接受
Yes, it is.
0x9i
Stat:0 开始判断
Stat:4 这里需要判断是十进制还是十六进制
Stat:5 在这里读入全部的十六进制数字
Stat:5 在这里读入全部的十六进制数字
No, it isn't.
```

图表 2 判断字符串是否为 C/C++的常数表达运行结果

二、 Aho-Corasick Automation(AC 自动机)

字符串中，总是有一个问题是避不开的，那就是模式串的匹配，其经典的匹配算法莫过于暴力法以及 KMP 算法。而今天所说的 AC 自动机，它是在字典树(Trie 树)上，加以 KMP 算法的失配指针的思想所产生的，它将字典树上的每一个节点作为自动机中的一个状态，其 Fail 指针作为其失配时的状态转移映射，如果当前字符在这个状态下失配，那么就进行状态转移至其 Fail 指针所指向的状态，重新进行匹配，如果当前状态是起始状态也就是字典树中的 root 节点，那么表示这个字符已经彻底失配，继续下一个字符的匹配。AC 自动机能够完成对多个模式串的匹配，这是 KMP 算法所不能做到的，其时间复杂度为：$\boldsymbol{\Theta(N+M)}$其中$\boldsymbol{N}$为模式串总长度，$\boldsymbol{M}$为大文本的长度，而 KMP 算法则达到了$\boldsymbol{\Theta(N+LM)}$，L 为模式串的个数。因此其在多模匹配情况下时间复杂度更低。

类似的，我们做出 AC 自动机的失配指针图，我们假定其中字符集只有小写字母，如果有其他的字符则可以通过哈希方法而得到，其中实线表示其匹配到的转移，而虚线则表示没有匹配到的转移，也就是 Fail 指针。



图表 3 AC 自动机的失配指针图

如图，我们进行 qqw、qwe、wer 模式串的插入并建立失配指针，root 的全部节点其失配指针指向 root,而其余节点则需要不断查询其父节点的失配指针然后再进行失配指针建立。其匹配的结果如图：

```
输入模式串的个数:3
输入模式串:
qqw
qwe
wer
输入大文本:
qwer
共有2个子字符串出现在原字符串中，它们分别为:
qwe
wer
```

图表 4 AC 自动机进行多模匹配结果