第 2 次作业 - 正则表达式

2154312 郑博远

习题 3.1.1 写出表示下列语言的正则表达式：

a) 字母表 $\{a, b, c\}$ 上包含至少一个 a 和至少一个 b 的串的集合。

b) 倒数第 10 个符号是 1 的 0 和 1 的串的集合。

c) 至多只有一对连续 1 的 0 和 1 的串的集合。

解答：

a) $c^*a(a+c)^*b(a+b+c)^* + c^*b(b+c)^*a(a+b+c)^*$

b) $(0+1)^*1(0+1)^9$

c) $(0+10)^*(11+1+\varepsilon)(0+01)^*$

习题 3.1.3 写出表示下列语言的正则表达式：

a) 不包含 101 作为子串的所有 0 和 1 的串的集合。

b) 具有相同个数的 0 和 1，使得在任何前缀中，0 的个数不比 1 的个数多 2，1 的个数也不比 0 的个数多 2，所有这种 0 和 1 的串的集合。

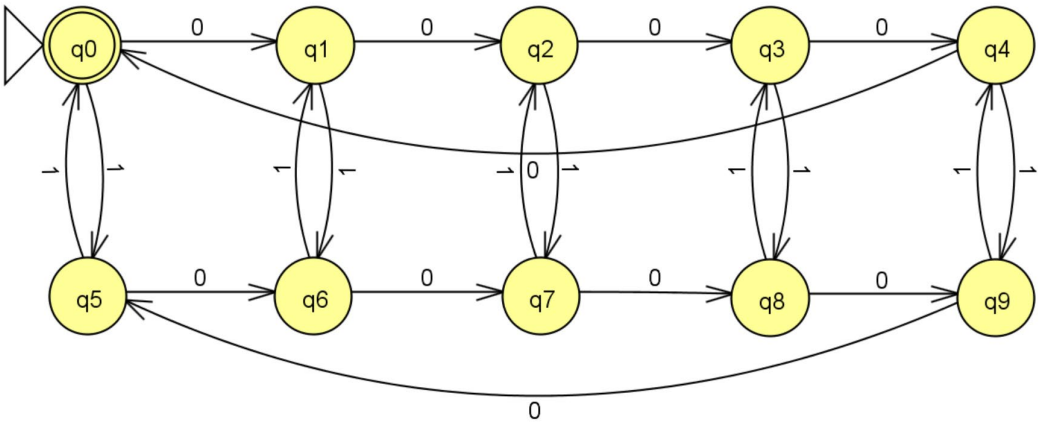
c) 0 的个数被 5 整除且 1 的个数是偶数的所有 0 和 1 的串的集合。

解答：

a) $0^*(1+000^*)^*0^*$

b) $(01+10)^*$

c) 设计对应该语言的 NFA 如下图所示：



转换为正则表达式如下：

(00000+11+(01+10)(11)*10000+(001+(01+10)(11)*(0+101))(11)*1000+(0001+(01+10)(11)*1001+(001+(01+10)(11)*(0+101))(11)*(0+101))(11)*100+(00001+(01+10)(11)*10001+(001+(01+10)(11)*(0+101))(11)*1001+(0001+(01+10)(11)*1001+(001+(01+10)(11)*(0+101))(11)*(0+101))(11)*(0+101))(11+00(11)*10001+00(11)*(0+101)(11)*1001+(00(11)*1001+00(11)*(0+101)(11)*(0+101))(11)*(0+101))*(10+01+00(11)*10000+00(11)*(0+101)(11)*1000+(00(11)*1001+00(11)*(0+101)(11)*(0+101))(11)*100))*

习题 3.1.4 给出下列正则表达式语言的自然语言描述：

a) $(1 + \varepsilon)(00^*1)^*0^*$。

b) $(0^*1^*)^*000(0+1)^*$。

c) $(0+10)^*1^*$。

解答：

a) 不包含连续的 1 的 0 和 1 的串的集合。

b) 包含 000 作为子串的 0 和 1 的串的集合。

c) 除了字符串末尾外不包含连续的 1 的 0 和 1 的串的集合。

习题 3.2.4 把下列正则表达式转化成带 ε 转移的 NFA.

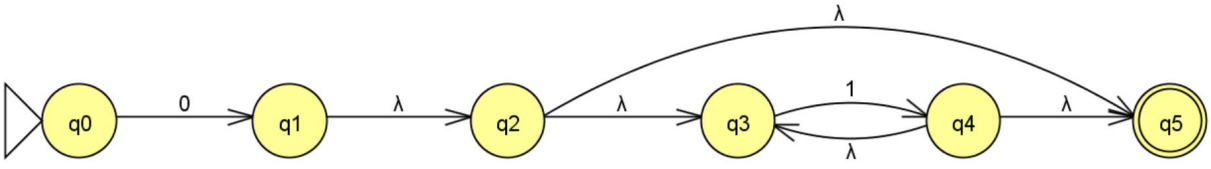
a) 01^*。

b) $(0+1)01$。

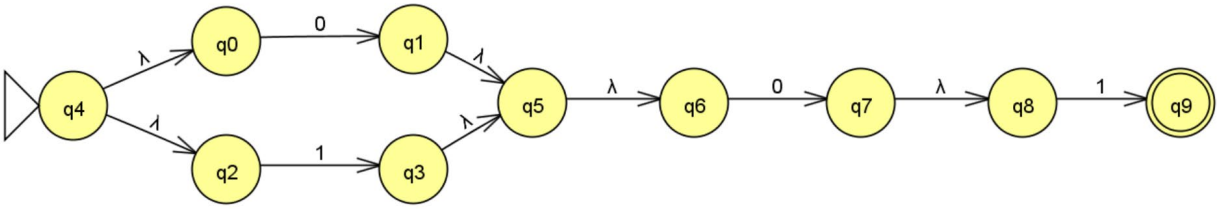
c) $00(0+1)^*$。

解答：

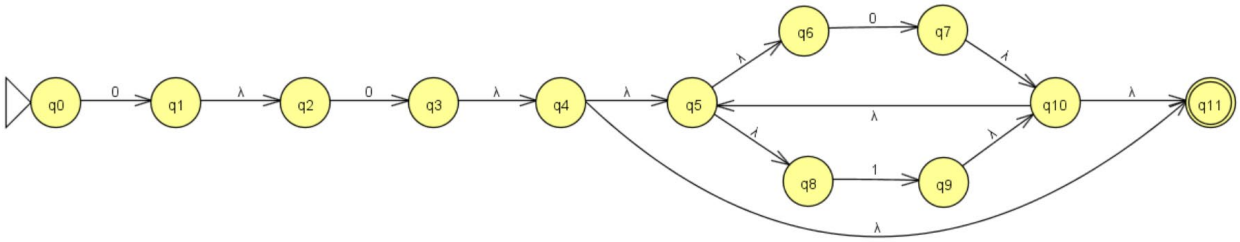
a) 正则表达式能转化为如下带 ε 转移的 NFA：



b) 正则表达式能转化为如下带 ε 转移的 NFA：



c) 正则表达式能转化为如下带 ε 转移的 NFA：



习题 3.2.6 设 $A=(Q,\ \Sigma,\ \delta,q_0,\{q_f\})$ 是一个 ε-NFA，使得既没有进入 q_0 转移，也没有离开q_f的转移。就 $L=L(A)$ 而言，描述 A 的每一个下列修改所接受的语言：

a) 通过增加从 q_f 到 q_0 的 ε 转移，从 A 构造的自动机。

b) 通过增加从 q_0 到每个从 q_0 可达（沿着标记包含 Σ 中符号和 ε 的路径）的状态的 ε 转移，从 A 构造的自动机。

c) 通过增加从每个能沿着某条路径到达 q_f的状态到 q_f的 ε 转移，从 A 构造的自动机。

d) 通过同时做 (b) 和 (c) 的修改从 A 构造的自动机。

解答：

a) $L_a=L^+$

b) $L_b=\{\,y\mid xy\in L;\ x,y\in\Sigma^*\,\}$

c) $L_c=\{\,x\mid xy\in L;\ x,y\in\Sigma^*\,\}$

d) $L_d=\{\,y\mid xyz\in L;\ x,y,z\in\Sigma^*\,\}$

习题 3.2.7 把正则表达式转化为 ε - NFA 的定理 3.7 的构造，有些地方可以化简。这里有三处：

1. 对于并运算符，不是构造新的初始状态和接受状态，而是把两个初始状态合并成一

个具备两个初始状态的所有转移的状态。同样，合并两个接受状态，让所有的转移相应地进入合并状态。

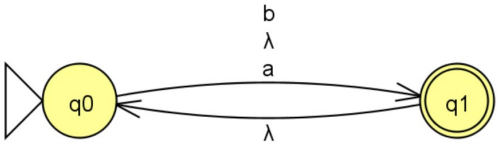
2. 对于连接运算符，把第一个自动机与第二个自动机的接受状态合并。

3. 对于闭包运算符，只是增加从接受状态到初始状态的以及反方向的 ε 转移。

每一个这种化简本身仍然产生正确的构造；也就是说，对于任何正则表达式，得出的 ε - NFA 接受这个表达式的语言。变化(1)、(2)和(3)的哪些子集可以一起用在构造中，对于每一个正则表达式，仍然产生正确的自动机?

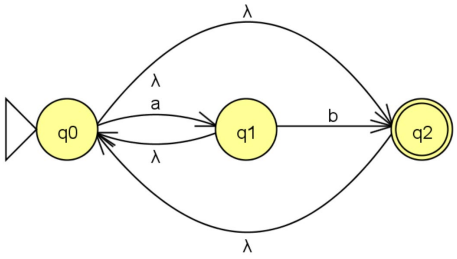
解答：

1. (1)和(3)组合，考虑表达式 $a^* + b$，按照题干的规则化简为如下的 NFA：



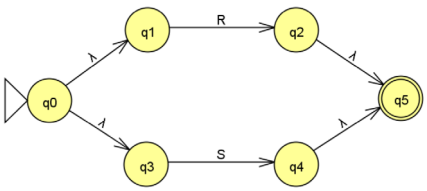
该 NFA 也可以接受不符合表达式的字符串 bb，因此不正确，(1)与(3)不能组合。

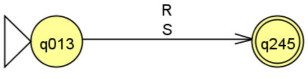
2. (2)和(3)组合，考虑表达式 $(a^*b)^*$，按照题干的规则化简为如下的 NFA：



该 NFA 也可以接受不符合正则表达式的字符串 aa，因此不正确，(2)与(3)不能组合。

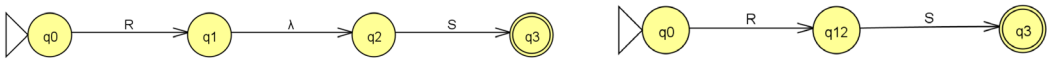
3. (1)和(2)组合





对于化简(1)，对任意 q 满足 $\delta(q,\ a) = q_0\ (a \in \Sigma)$，有 $\delta(q,\ a) = \delta(\delta(q,\ a),\ \varepsilon) = \{q_0, q_1, q_3\}$。若不存在 q_i 使得 $\delta(q_i,\ a) = q_1$ 或 $\delta(q_i,\ a) = q_3$，则状态 q_0、q_1、q_3 能够合并为 q_{013}；若存在这样的 q_i，则对于前者由于不存在 $\delta(q_i,\ a) = q_3$（反之亦然）而不能将状态合并为 q_{013}，从而不能使用(1)进行化简。q_2、q_4、q_5的状态合并同理。由于定理 3.7 中要求 ε −NFA

满足没有箭弧进入初始状态，也没有箭弧离开接受状态；只经过(1)和(2)化简的 ε −NFA 也不会存在箭弧进入初始状态或离开接受状态的情况，因此(1)和(2)的组合不会使化简(1)出错。



对于化简 2，同理可得，由于只经过(1)和(2)化简的 ε −NFA 不会存在箭弧进入第二个自动机的初始状态或离开第一个自动机的接受状态的情况，因此 q_1 能够与 q_2 进行状态合并，即(1)和(2)的组合不会使化简(2)出错。因此，(1)和(2)的组合可以构造正确的自动机。

综上所述，变化(1)、(2)、(3)中，(1)和(2)的组合可以一起用于构造使得自动机正确，其他的组合都会让自动机构造出错。

习题 3.2.8 给出一个算法：输入一个 DFA A，对于给定的 n（与 A 的状态个数无关），计算出 A 所接受的长度为 n 的串的个数。这个算法应当对于 n 和 A 的状态数来说都是多项式的。提示：使用定理 3.4 的构造所提示的技术。

解答：

设对于某个整数 s，A 的状态是 $\{1, 2, \ldots, s\}$。用 $N_{ijl}^{(k)}$ 表示字符串的个数，其中字符串 ω 满足：ω 是 A 中从状态 i 到状态 j 的路径的标记，其长度为 l，而且这条路径没有编号大于 k 的中间顶点。

基础：基础是 $k = 0$ 的情况，有如下表达式：

$$N_{ijl}^{0} = \begin{cases} 0 & i = j,\ l \geq 1 \\ 1 & i = j,\ l = 0 \\ i \text{ 到 } j \text{ 的路径数} & i \neq j \end{cases}$$

归纳：假设存在从 i 到 j 的路径不经过比 k 高的状态。有两种可能的情形需要考虑。

1. 这条路径根本不经过 k 状态。在这种情形下，路径的标记属于$N_{ijl}^{(k-1)}$的字符串个数。

2. 这条路径经过状态 k 至少一次。把路径分成几段，第一段不经过 k 而从状态 i 到状态 k，最后一段不经过 k 而从 k 到 j，所有中间路段都不经过 k 而从 k 到自身。这部分字符串的个数可以表示为：$\sum N_{ikl_1}^{(k-1)} \cdot N_{kkl_2}^{(k-1)} \cdot \cdots \cdot N_{kkl_{n-1}}^{(k-1)} \cdot N_{kjl_n}^{(k-1)}$，其中 l_i 满足$\sum_i^n l_i = l\ (n \geq 2)$。

因此结合前两种情况有表达式：

$$N_{ijl}^{(k)} = N_{ijl}^{(k-1)} + \sum N_{ikl_1}^{(k-1)} \cdot N_{kkl_2}^{(k-1)} \cdot \cdots \cdot N_{kkl_{n-1}}^{(k-1)} \cdot N_{kjl_n}^{(k-1)}$$

由题意，对 DFA A 所接受的长度为 n 的串的个数可以表示为：

$$\sum_j N_{1jn}^{(s)}$$

其中，s 表示 DFA 的总状态数，1 为起始状态，j 为任意的接收状态。

习题 3.3.1 给出一个正则表达式，来描述所能想到的所有不同形式的电话号码。考虑国际号码以及不同国家有不同位数的区号和本地电话号码。

解答：

1) 国际号码的电话格式如下：**国际冠码 + 国际电话区号 + 电话号码**

- 国际冠码：不同国家有不同的国际冠码（如中国大陆是“00”），拨打国际电话的时候要根据拨出地区确定国际冠码，但可以统一用“+”表示；
- 国际电话区号：每个国家分配的一个代码，如中国大陆为“86”；

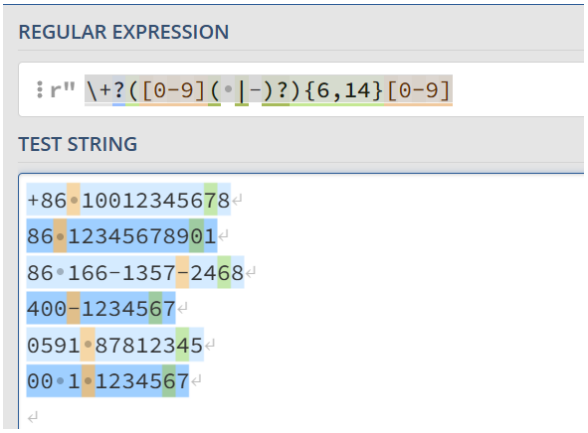
2) 以中国大陆为例的国内电话号码格式如下：**长途冠码 + 省市区号 + 电话号码**

- 长途冠码：在国内拨打长途需加拨长途冠码“0”；
- 省市区号：不同省、直辖市、大型城市分配的代号，比如北京市为“10”；

总的来说，电话号码的形式十分多样。从号码的长度来看，最短的是 7 位的加拿大等国家，最长的是 11 位的中国等国家；考虑到冠码与区号，号码的总长度大概在 7 位到 16 位之间。一般来说，电话号码可以是连续的数字，也可能被“-”分割成几个为单位的多个字段。由于号码的形式十分多样，因此我希望表达式能够尽可能的识别更多的电话号码，这就不可能避免的识别到一些并非电话号码的数字序列。综上，给出的 UNIX 风格的表达式如下：

```
\+?([0-9]( |-)?){6,15}[0-9]
```

该正则表达式能够识别长度在 7 位到 16 位，任意位数个数字以空格或横杠分割的电话号码，并可以以国际冠码“+”开头。测试用例如下：



习题 3.3.2 给出一个正则表达式，来表示在招聘广告中可能出现的薪水。考虑可能按小时、周、月或年发放的薪水。这些薪水可能有也可能没有$（如美元）符号或其他单位（如后面跟着的“K”）。可能有一个或多个邻近的单词标志着薪水。提示：查看报纸上的分类广告或在线职位列表，来获得些关于什么样的模式可能有用的想法。

解答：

考虑到招聘广告的情况比较复杂，仅涉及英文语境下的美元($)、人民币(¥)、英镑(£)、欧元(€)四种货币(其他货币类似)的情况。招聘广告场景中，常用的数字单位有千(K)、百万(M)、十亿(B)，工作时间单位有小时、日、周、月、年。此外，在许多场景下薪资并不是固定的价格，而是价格区间。综上考虑，设计 UNIX 风格的表达式如下：

```
(\$|¥|£|€)? ?[0-9]+(\.[0-9]+)?(k|K|m|M|b|B)?( ?(to|-) ?(\$|¥|£|€)? ?[0-9]+(\.[0-9]+)?(k|K|m|M|b|B)?)? ?(per hour|hourly|\/hour|\/h|\/hr\.?|per day|daily|\/day|\/d|per week|weekly|\/week|\/w|\/wk\.?|per month|monthly|\/month|\/mth\.?|\/mo\.?|per year|yearly|\/year|\/yr\.?)
```

该正则表达式能够识别如下的多种形式的薪水表达：