形式语言与自动机　第 3 章作业

1951112　林日中

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. 表示上述语言的正则表达式分别为：

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)　$(0+10)^*(11+\varepsilon)(0+01)^*$.

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. 表示上述语言的正则表达式分别为：

a)　$0^*(1^*000^*)^*1^*0^*$.

b)　$(01+10)^*$.

c)　$\Big(00000+11+(01+10)(11)^*10000+\big(001+(01+10)(11)^*(0+101)\big)(11)^*1000+\big(0001+(01+10)(11)^*1001+\big(001+(01+10)(11)^*(0+101)\big)(11)^*(0+101)\big)(11)^*100+\big(00001+(01+10)(11)^*10001+\big(001+(01+10)(11)^*(0+101)\big)(11)^*1001+\big(0001+(01+10)(11)^*1001+\big(001+(01+10)(11)^*(0+101)\big)(11)^*(0+101)\big)(11)^*(0+101)\big)\big(11+00(11)^*10001+00(11)^*(0+101)(11)^*1001+\big(00(11)*1001+00(11)^*(0+101)(11)^*(0+101)\big)(11)^*(0+101)\big)^*\big(10+01+00(11)^*10000+00(11)^*(0+101)(11)^*1000+\big(00(11)^*1001+00(11)^*(0+101)(11)^*(0+101)\big)(11)^*100\big)\Big)^*$.

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. 上述正则表达式语言的自然语言描述如下所示.

a) 不包含连续 1 的所有 0 和 1 的串的集合.

b) 包含 000 作为子串的所有 0 和 1 的串的集合.

c) 除了末尾可能有一串 1 以外不含连续 1 的所有 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.

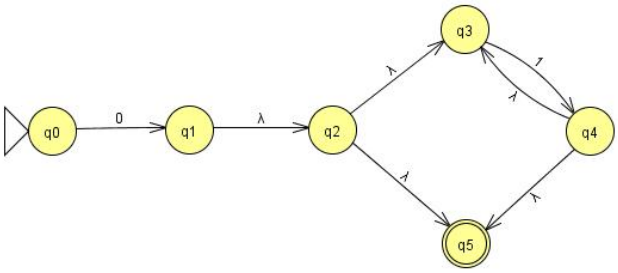
a) 01^*.

b) $(0+1)01$.

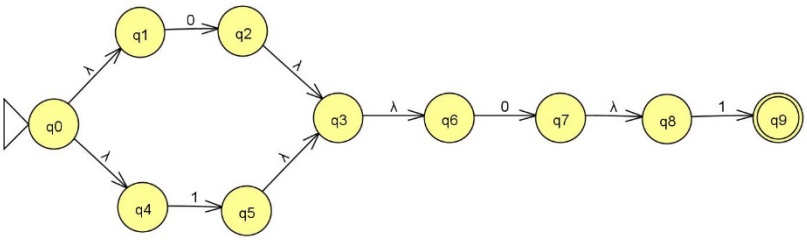
c) $00(0+1)^*$.

解答. 上述正则表达式转化成的带 ε 转移的 NFA 如下所示.

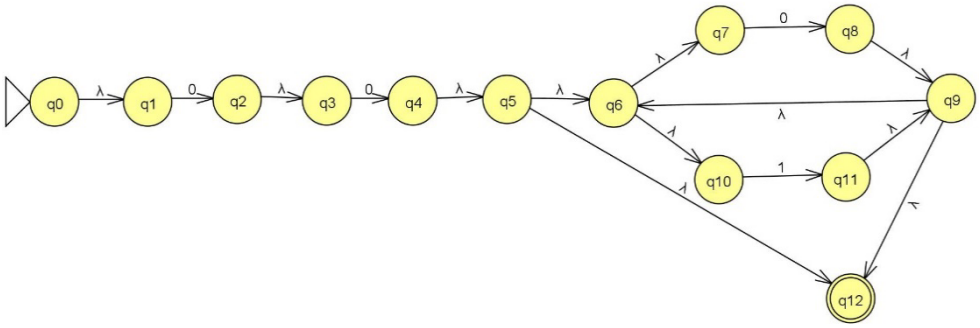
a)



b)

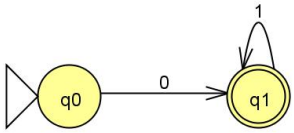


c)

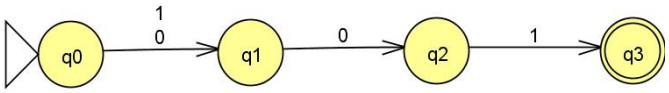


以上 ε-NFA 分别简化后，有

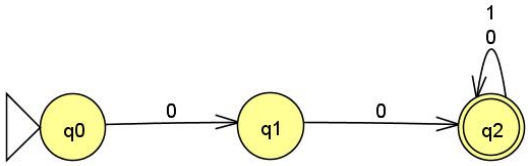
a)



b)



c)



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 的转移，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 的每一个上述修改所接受的语言的描述如下：

a) $L_a = L^+$

b) $L_b = \{v \in \Sigma^* | uv \in L, u \in \Sigma^*\}$

c) $L_c = \{u \in \Sigma^* | uv \in L, v \in \Sigma^*\}$

d) $L_d = \{v \in \Sigma^* | uvw \in L, u, w \in \Sigma^*\}$

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.

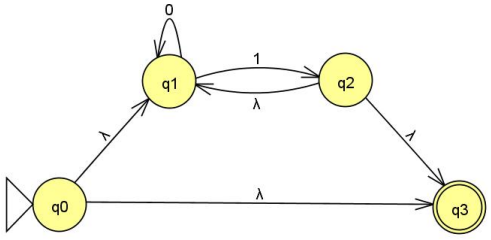
2. 对于连接运算符，把第一个自动机的接受状态与第二个自动机的初始状态合并.

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.

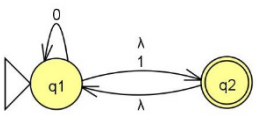
每一个这种化简本身仍然产生正确的构造；也就是说，对于任何正则表达式，得出的 ε-NFA 接受这个表达式的语言. 变化 (1)、(2) 和 (3) 的哪些子集可以一起用在构造中，对于每一个正则表达式，仍然产生正确的自动机?

解答. 变化 (3) 有误. 变化 (1) 和 (2) 可以一起构造中，对于每一个正则表达式，仍然产生正确的自动机. 证明如下：

对于变化 (3)，当**闭包运算符包含的部分存在从初始状态到初始状态的路径**时，若将变化 (3) 应用于原自动机以产生新的构造，会导致这部分串错误地被接受. 考察正则表达式 $(0^*1)^*$，根据定理 3.7，对应的自动机如下所示.



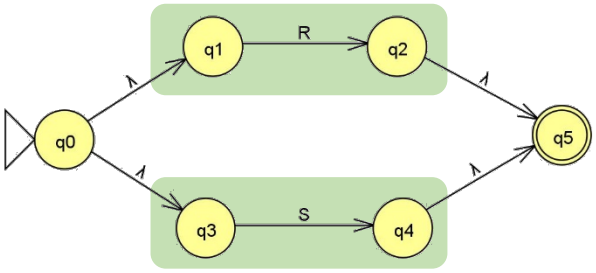
将变化 (3) 应用到该自动机，生成的新构造如下所示.



显然，串 00 在原自动机中被拒绝，而在新构造中被接受. 因此，变化 (3) 有误.

下面证明变化 (1) 的正确性.

对于变化 (1)，记并运算符作用的两个正则表达式为 R 和 S，对应的自动机分别为 A 和 B，则 $R+S$ 的标准构造方式可以表示为：



记该自动机 $C=(Q_1,\Sigma,\delta_1,q_0,\{q_5\})$. 将变化 (1) 应用到 C，记新构造的自动机 $D=(Q_2,\Sigma,\delta_2,q_{013},\{q_{245}\})$. 根据题设，有

$$Q_2=Q_1-\{q_0,q_1,q_3,q_2,q_4,q_6\}+\{q_{013},q_{245}\}$$

$$\delta_2(q,a)=\begin{cases}\{\delta_1(q_1,a),\delta_1(q_3,a)\} & q=q_{013}\\ \{q_{246}\} & \delta_1(q,a)\in\{q_2,q_4\}\\ \{\delta_1(q,a)\} & \text{else}\end{cases}$$

任取串 ω，考虑 $\delta_2(q,a)$ 的定义，有

$$\omega\in L(C)$$

$$\Leftrightarrow \widehat{\delta_1}(q_0,\omega)=\widehat{\delta_1}(q_1,\omega)\cup\widehat{\delta_1}(q_3,\omega)=\widehat{\delta_1}(q_2,\varepsilon)\cup\widehat{\delta_1}(q_4,\varepsilon)=\{q_2,q_4,q_5\}$$

$$\Leftrightarrow \widehat{\delta_2}(q_{013},\omega)=\widehat{\delta_1}(q_1,\omega)\cup\widehat{\delta_1}(q_3,\omega)=\widehat{\delta_1}(q_2,\varepsilon)\cup\widehat{\delta_1}(q_4,\varepsilon)=\{q_{246}\}$$

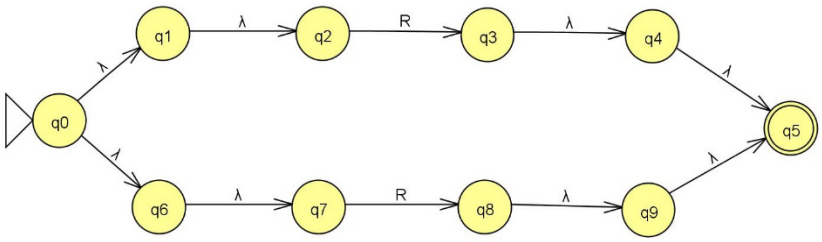
$$\Leftrightarrow \omega\in L(D)$$

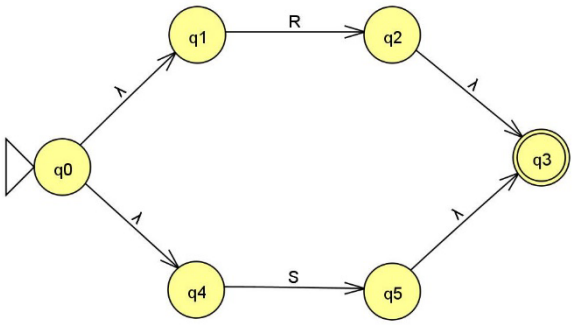
即自动机 C 和 D 等价. 因此，变化 (1) 正确.

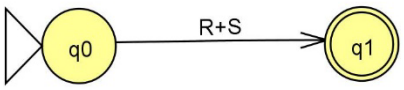
同样地，对于变化 (2)，我们可以利用原自动机的转移函数定义应用变化 (2) 后的新构造的自动机的转移函数，同理可证变化 (2) 正确.

下面证明变化 (1) 和 (2) 可以一起构造中，对于每一个正则表达式，仍然产生正确的自动机.

一方面，在并运算与连接运算复合作用时，变化组合 (1)(2) 正确. 考察并运算对连接运算的右复合（即先进行连接运算，反之同理）. 作用步骤如下所示.

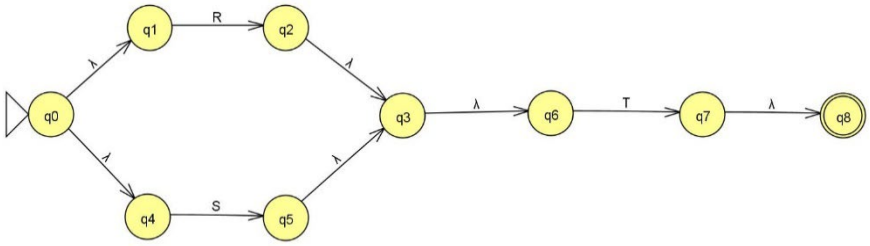


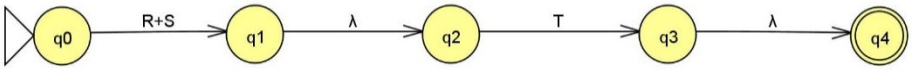


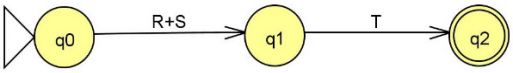


由于等价关系具有传递性，变化 (1) 和 (2) 作用构造的自动机与原自动机等价.

另一方面，在并运算与连接运算相继作用时，变化组合 (1)(2) 正确. 考察以下自动机，并有相应的作用步骤如下所示.







由于等价关系具有传递性，变化 (1) 和 (2) 作用构造的自动机与原自动机等价.

综上所述，变化 (1) 和 (2) 可以一起构造中，对于每一个正则表达式，仍然产生正确的自动机. 证毕.

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. 记 A 的状态有 s 个，其中有 q 个接受状态，$1 \le q \le s-1$. 为 A 各状态赋连续正整数编号，即将初始状态赋 1，接受状态分别赋 $s-q+1, s-q+2, \dots, s$. 记 $R_{ijm}^{(k)}$ 为从状态 i 到状态 j 的长度为 m 且不经过编号高于 k 的状态的路径的数量.

对于一个 DFA A，给定 i、j 和 n，我们可以通过对 k 的归纳来给出 $R_{ijm}^{(k)}$ 的算法.

基础：$k=0$ 时，R_{ij1}^{0} 是从状态 i 到状态 j 的转移路径的数量. 有

$$R_{iim}^{(0)} = \begin{cases} 1, & m = 0, \\ 0, & m \ge 1. \end{cases}$$

归纳：对于特定的 m 和任意的 i, j，假设 $R_{ij0}^{(k-1)}, R_{ij1}^{(k-1)}, \dots, R_{ijm}^{(k-1)}$ 均已求得，那么分别考虑从 i 到 j 不经过状态 k 与 经过状态 k 的路径数量，有

$$R_{ijm}^{(k)} = R_{ijm}^{(k-1)} + \prod_{s=1}^{r} R_{ikp_s}^{(k-1)}$$

其中 $r, p_s \in Z^+, r \ge 2, \sum_{s=1}^{r} p_s = m$. 显然，**递推算法是多项式的**.

根据题设，DFA A 所接受的长度为 n 的串的个数可以表示为

$$\sum_{j=s-q+1}^{s} R_{1jn}^{(s)}$$

其中，s 是状态的数量，1 是起始状态，j 是任意接受的状态，n 是路径的长度.

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. **本题的可能情况非常多**. 我们考虑以“+”开头、带国际电话区号的，可能含有空格符号或“-”符号作为分隔符的电话号码.

在 UNIX 中，能满足题设条件的正则表达式如下所示：

```
^\+(?:[0-9]( |-)?){6,14}[0-9]$
```

其中，各部分的意义分别为：

- `^` ——定位行首.
- `\+` ——匹配索引为 43_{10}（$2B_{16}$ 或 53_{8}）的字符“+”（区分大小写）.
- 非捕获组 `(?:[0-9]( |-)?){6,14}` 匹配前一个字符.
    - `(?: )` ——非捕获组.
    - `{6,14}` ——匹配 6 到 14 次之间的前一个标记，次数越多越好，根据

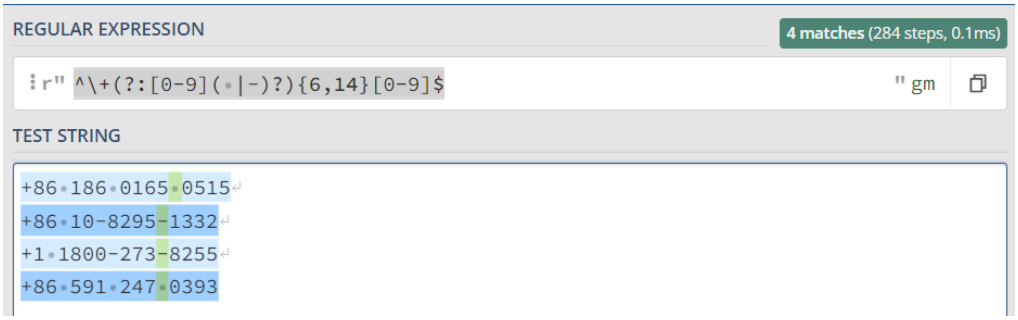
需要给予回馈（贪婪）.

- `[0-9]` ——匹配 0（索引 48）和 9（索引 57）之间的单个字符（区分大小写）.
- `( |-)?` ——捕获组.
  - `?` ——匹配 0 到 1 次之间的前一个标记，尽可能多的次数，根据需要给予回馈（贪婪）.
  - `␣` ——第一备选，匹配索引为 32_{10}（20_{16} 或 40_{8}）的字符（区分大小写）.
  - `-` ——第二备选，匹配索引为 45_{10}（$2D_{16}$ 或 55_{8}）的字符（区分大小写）.

- `[0-9]` ——匹配 0（索引 48）和 9（索引 57）之间的单个字符（区分大小写）.
- `$` ——定位行末.

正则表达式两端的 '^' 和 '$' 锚点确保它能匹配整个主题文本. 非捕获组，用 '(?:……)' 括起来，匹配一个单一的数字，后面有一个可选的空格字符. 用区间量词 '{6,14}' 重复这个分组，执行最小和最大数字数的规则，同时允许空格分隔符出现在数字的任何地方. 字符类的第二个实例 '[0-9]' 完成了数字的规则（从 6 到 14 位上升到 7 到 15 位），**并确保电话号码不以空格结束**.

调试结果如下图所示. 可以看到，该正则表达式对 4 个测试样例串匹配.



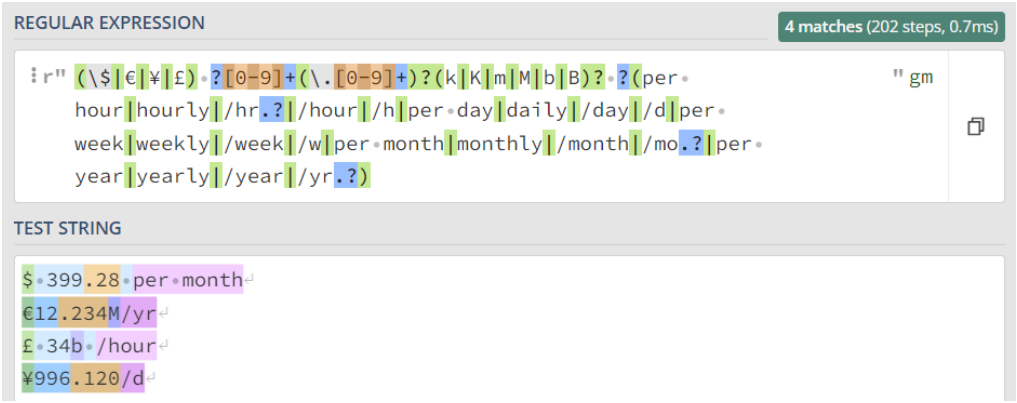
所使用的 REGEX 调试网站为 https://regex101.com/.

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. **本题的可能情况非常多**. 我们以**英文**环境下的，以**美元、欧元、人民币、英镑**共 4 中货币形式的，可能以 **k（千）、m（百万）或 b（十亿）**为单位的，按**小时、周、月或年**发放的薪水为例. 能满足题设条件的 UNIX 正则表达式如下所示：

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

调试结果如下图所示. 可以看到，该正则表达式对 4 个几乎完全不同的测试样例串匹配.



所使用的 REGEX 调试网站为 https://regex101.com/.