正则表达式-自动机转换工具

说明文档

小组成员:

张许可　2009212639

耿卉卉　2009212626

2009/11/26

正则表达式-自动机的转换工具说明文档

1 开发环境

- Microsoft Windows XP Professional SP3
- Pentium(R) Dual-Core CPU E5300 @ 2.60GHz，2G 内存，406G 硬盘
- Eclipse3.5 Integrate Develop Environment
- Java 语言

2 项目要求

- 用 java 语言编写，保证可扩展性
- 正则表达式转换到自动机
- 自动机转换到正则表达式

3 功能描述

该转换工具实现了正则表达式和自动机的相互转换。由于其他模块已经实现了非确定有限自动机(NFA)到确定有限自动机(DFA)的转换，在本项目中只实现正则表达式到 NFA 的转换和 DFA 到正则表达式的转换。下面是相应的转换功能的一些简要描述。

3.1 正则表达式转换为非确定自动机（NFA）

使用 Thompson 构造法，将基于最小正则表达式经过并运算、连接运算、括号运算和闭包运算产生的所有正则表达式的转换，同时实现了基于字母表的通配符(.)的正则表达式到 NFA 的转换。示例：

输入正则表达式 *r=(a|b)*abb* 可以生成以下的 NFA

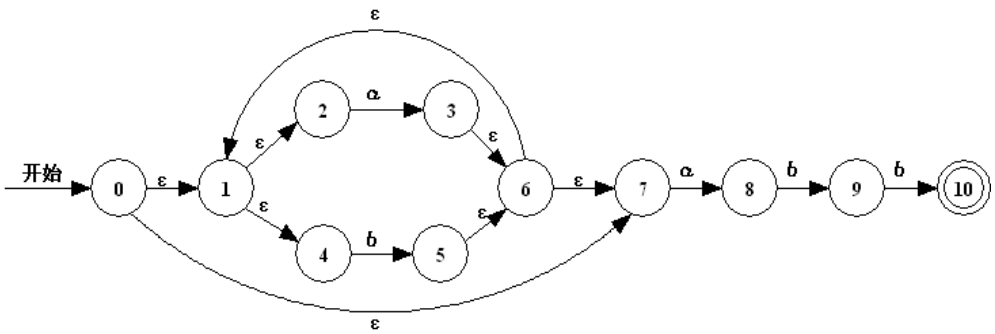

图 3-1 正则表达式 *r=(a|b)*abb* 生成的 NFA

3.2 确定有限自动机（DFA）转换为正则表达式

使用状态消去法，将给定的确定有限自动机(DFA)转换为未化简的正则表达式，因为正则表达式的化简不在项目要求之内，所以对正则表达式的化简未进行实现，可作为以后的扩展模块。

示例：

表 3-1 DFA 状态转移表

	a	b
→A	B	C
B	B	D
C	B	C
D	B	E
*E	B	C

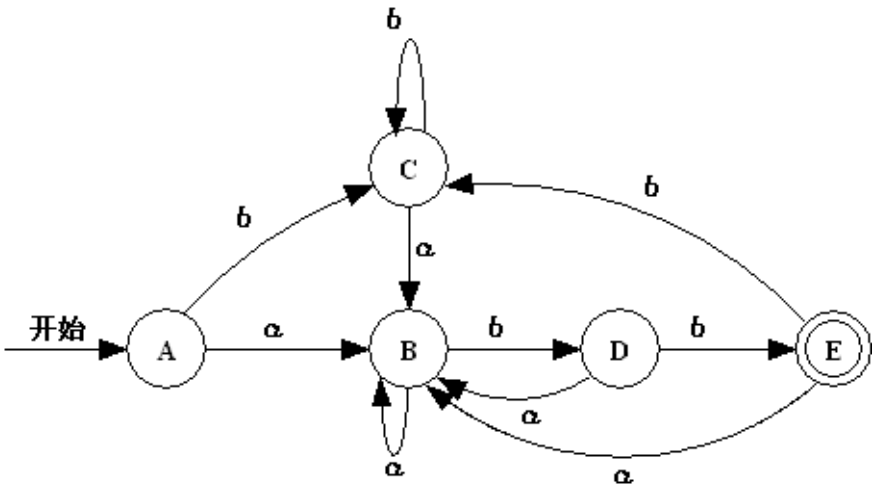

图 3-2 表 3-1 表示的 DFA

该 DFA 生成未化简的正则表达式为：

`(((aa*b+bb*(aa*b))(aa*b)*b)((aa*b+bb*(aa*b))(aa*b)*b)*)`。

虽然该正则表达式比(a|b)*abb 要复杂得多，但是我们通过使用状态消去法，用手工化简该确定有限自动机，我们会得到相同的结果，这证明该结果是正确的。由于正则表达式的化简非常复杂，所以在本项目中没有给出实现。

4 源代码的说明

简要说明具体实现中，源代码采用的算法、数据结构及源代码的结构。

4.1 正则表达式转换为非确定有限自动(NFA)的源代码说明

在具体实现中，源代码采用了 Thompson 构造法。算法过程的描述及采用的数据结构见 AM_FinalProject.pptx。由于在源代码中给出了详细的注释，同时在项目中生成了帮助文档(doc 目录下)，这里不再对源代码进行说明。源代码的结构如下：

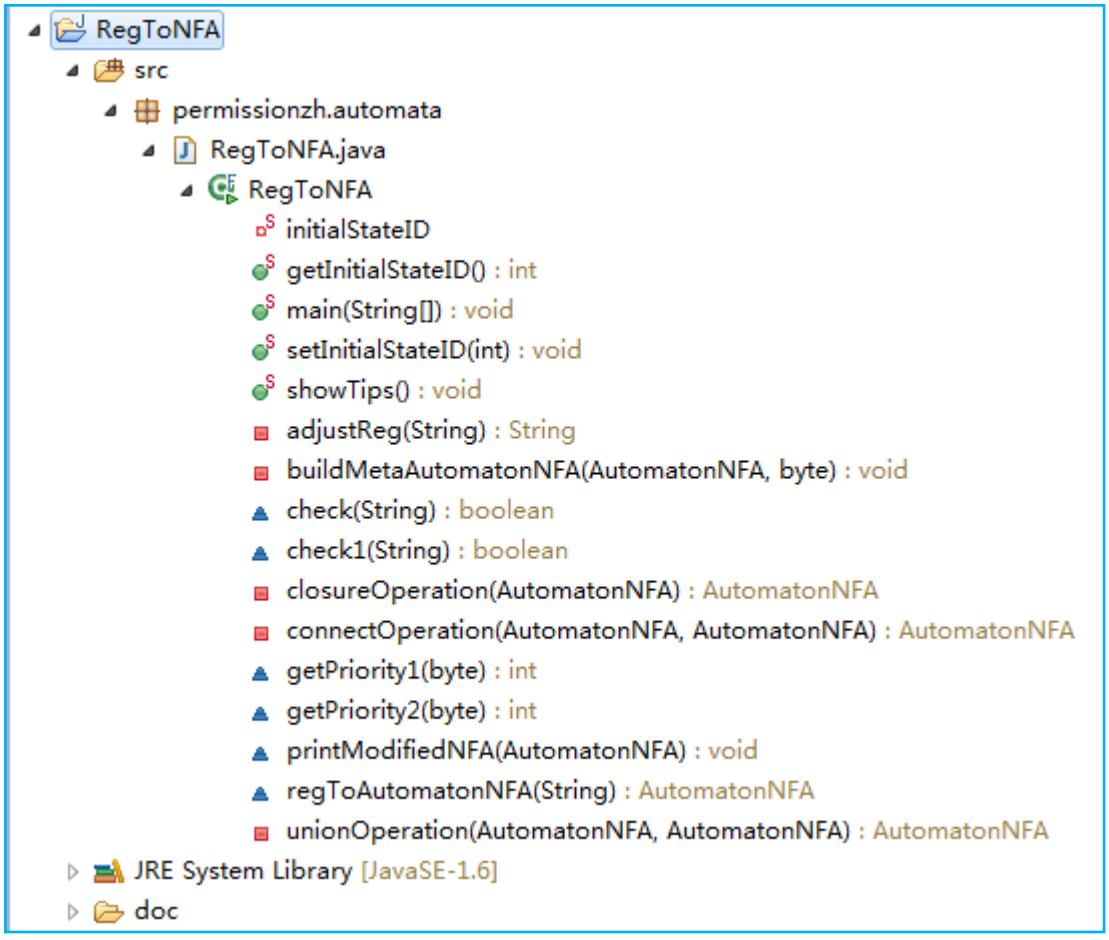

图 4-1 RegToNFA 源代码结构图

4.2 确定有限自动机(DFA)转化为正则表达式的源代码说明

在具体实现中，源代码采用了优化的状态消去法，即在进行转换之前，对确定有限自动机(DFA)进行调整，添加一个初始状态，使其通过过空转移向 DFA 原来的初始状态，添加一个结束状态，使所有的结束状态通过空转移指向它。这样保证要转换的 DFA 通过状态消去

最后只省下两个状态，即加入的开始状态和结束状态，这样便会相对容易的进行正则表达式的转换。算法过程的描述及采用的数据结构见 AM_FinalProject.pptx。由于在源代码中给出了详细的注释，同时在项目中生成了帮助文档(doc 目录下)，这里不再对源代码进行说明。源代码的结构如下：

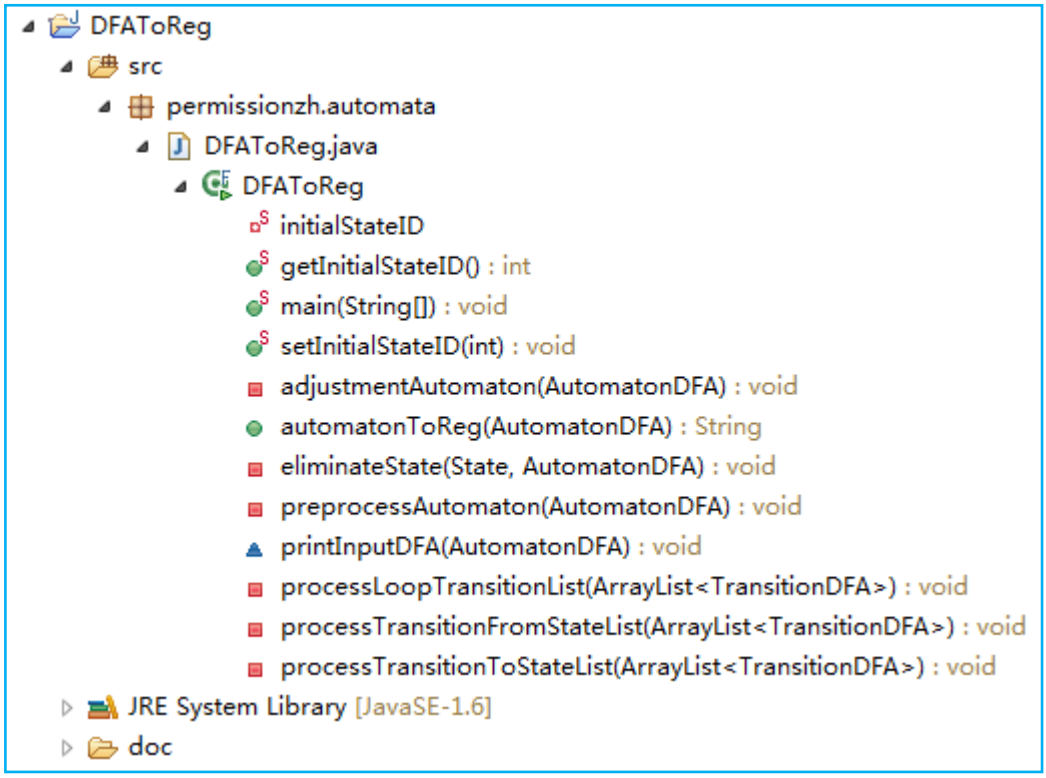

图 4-2 DFAToReg 源代码结构图

5 运行结果显示

5.1 正则表达式转换为非确定有限自动(NFA)

```
*********************************************
请输入正则表达式：
a(a+b)*
正则表达式成功转换为NFA...

NFA的信息如下：
automaton name:RegNFA
automaton Type:NFA
inputSymbolSet:a,b
state number:10
transition number:12

NFA的状态转移表为(每个数字表示一个状态)：
          ε         a         b
============================
3         {7}       Ø         Ø
2         Ø         {3}       Ø
1         {8}       Ø         Ø
→0        Ø         {1}       Ø
7         {6,9}     Ø         Ø
6         {2,4}     Ø         Ø
5         {7}       Ø         Ø
4         Ø         Ø         {5}
*9        Ø         Ø         Ø
8         {6,9}     Ø         Ø
```

图 5-1 不含通配符(.)的正则表达式到 NFA 的转换测试结果显示

```
*********************************************
请输入正则表达式：
a.b
正则表达式成功转换为NFA...

NFA的信息如下：
automaton name:RegNFA
automaton Type:NFA
inputSymbolSet:a,b
state number:10
transition number:10

NFA的状态转移表为(每个数字表示一个状态)：
          ε         a         b
============================
3         {7}       Ø         Ø
2         Ø         {3}       Ø
1         {6}       Ø         Ø
→0        Ø         {1}       Ø
7         {8}       Ø         Ø
6         {2,4}     Ø         Ø
5         {7}       Ø         Ø
4         Ø         Ø         {5}
*9        Ø         Ø         Ø
8         Ø         Ø         {9}
```

图 5-2 含通配符(.)的正则表达式到 NFA 的转换的测试结果显示

注：具体实现中，通配符只对字母表中的字母进行通配，也就是通配集为字母表。

5.2 确定有限自动机(DFA)转化为正则表达式

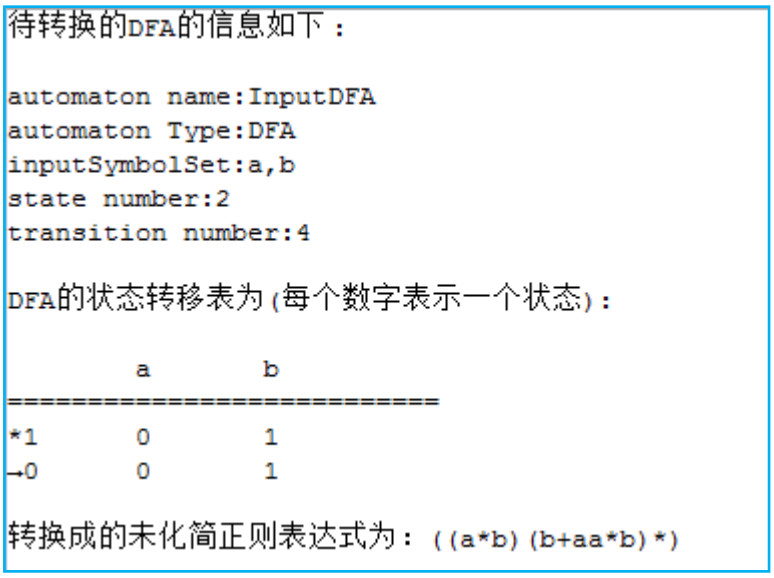

图 5-2 DFA 转换为正则表达式测试结果显示

注：为了便于测试，待转换的 DFA 是写死在程序中的。同时，在程序中提供了三个不同复杂程度的 DFA，其中两个在 Main 函数中被注释，可以通过添加和去除注释来测试相应的 DFA。

6 完成情况说明

完成了项目要求的基本功能：

- 正确实现了基本正则表达式到非确定有限自动机(NFA)的转换。
- 正确实现了确定有限自动机(DFA)到正则表达式的转换。

完成的扩展功能：

- 实现了对含有通配符(.)的正则表达式到非确定有限自动机(NFA)的转换。此处的通配是对字母表中字符的通配。