自动机大作业设计文档

作者：袁一方（2009212616）

王宇辉（2009212613）

目录

1、项目说明

1.1 开发环境

操作系统：Microsoft Windows XP SP2
机器配置：内存：2G
　　CPU：Intel T6500
开发环境：Eclipse 3.5

1.2 项目要求

要求：　用 Java 编写，保证可扩展性
　　判断一个自动机是不是 DFA？是不是 NFA？
　　实现自动机的交并补操作。
　　实现自动机最小化的算法。

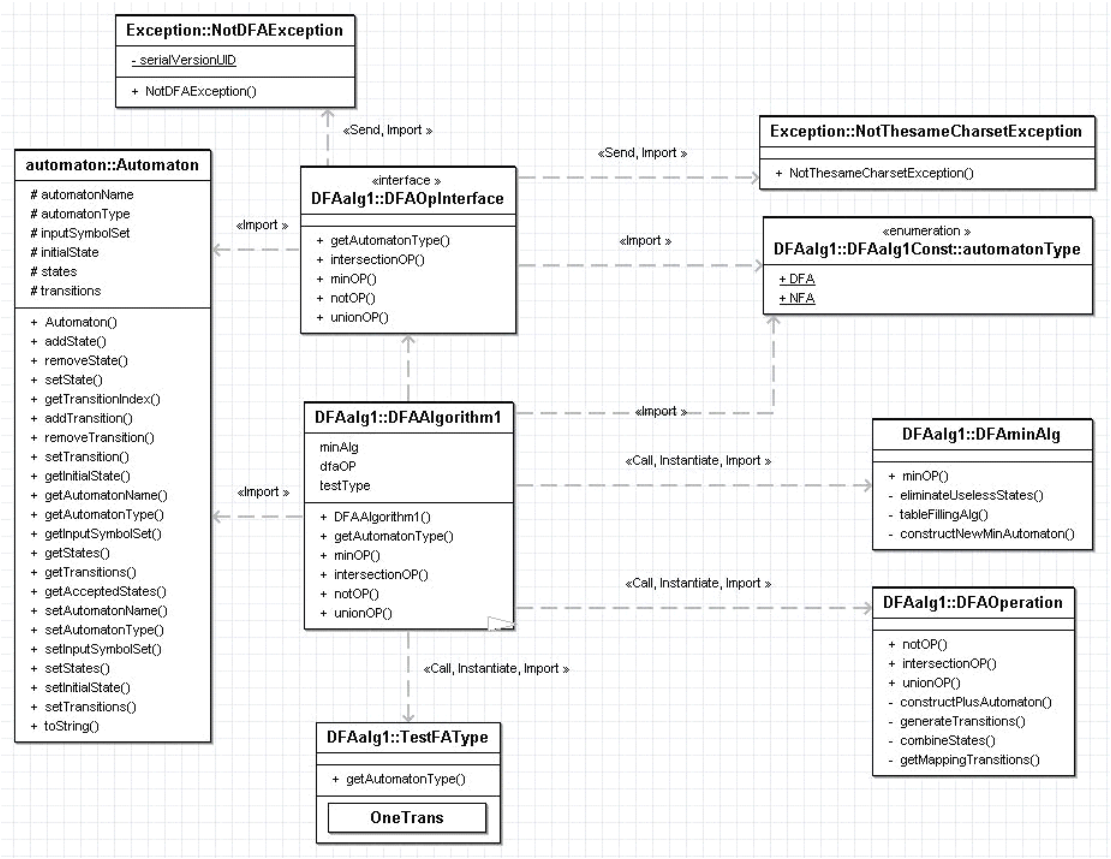
2 类设计

采用接口的类设计方案，可以根据需求更改实现，保证较少的影响其他系统的使用，保证可扩展性。

类描述:

类名称	类描述
NotThesameCharsetException	自动机交和并操作时转移条件不相同
NotDFAException	进行运算的不是个确定性自动机
DFAOpInterface	自动机操作接口包括判断是否是 DFA、DFA 的交、并、补、最小化运算
DFAAlgorithm1	实现 `DFAOpInterface` 接口
DFAminAlg	实现 DFA 最小化算法
DFAOperation	实现 DFA 交、并、补操作
TestFAType	测试自动机的类型

类图设计如下所示：



3 详细设计

3.1 判断一个自动机是不是 DFA？是不是 NFA

在这里假设自动机不是 DFA 就是 NFA。

首先判断转移的边是否等于状态的个数乘以字符集的数目。若是接着判断，将开始状态和转移和结束状态和在一起成为唯一的字符串，将其加入到集合中，当发现不能在集合中加入该转移说明已经存在重复的转移。同时通过判断转移条件中是否有“empty”转移条件，若存在 empty 转移条件则说明有 epsilon 转移，返回状态 NFA。

3.2 实现自动机的交并补操作

这里只是针对 DFA 进行的操作。

1、补操作：对于补操作：$A=(Q,\sum,\delta,s,F)$ 是一个 DFA，那么由补操作产生的新

DFA 定义为：$\overline{A}=(Q,\Sigma,\delta,s,Q-F)$。只要将 A 中的接受的状态设为不接受状态，同时把不接受的状态设为接受的状态就得到 $\overline{A}$。

2、并和交运算：对于交运算和并运算，有两个 DFA，$A_1=(Q_1,\Sigma,\delta_1,s_1,F_1)$ 和 $A_2=(Q_2,\Sigma,\delta_2,s_2,F_2)$，那么有这两个 DFA 创造出来的新的自动机定义为：$B=(Q_1\times Q_2,\Sigma,\delta_B,(s_1,s_2),M)$。其中 $M\subseteq Q_1\times Q_2$，(s_1,s_2) 为 B 的开始状态，δ_B 为 B 的转移函数，且做作如下定义：

$\forall q_1\in Q_1,q_2\in Q_2,\sigma\in\Sigma:\delta_B((q_1,q_2),\sigma)=(\delta_1(q_1,\sigma),\delta_2(q_2,\sigma))$

1、当 $M=F_1\times F_2$ 时，有上述方法得到的 B 就是 DFA A_1,A_2 的交运算。

2、当 $M=Q_1\times F_2\cup F_1\times Q_2$ 时，由上述方法得到 B 就是 DFA A_1,A_2 的并运算

3.2 实现自动机的最小化操作

采用的方法是填表算法，基于如下递归的标记可区别的状态偶对的过程

1、首先消除无用状态。

2、根据填表算法将表格填出来。

3、合并等价状态

4、计算出所有状态的转移。

其中填表算法如下所示。

-基础

若 p 为终态，q 为非终态，则 p 和 q 标记为可区别的。

-归纳

设 p 和 q 已标记为可区别的。若状态 r 和 s 通过某个输入符号 a 可分别转移到 p 和 q，即 $\delta(r,a)=p,\delta(s,a)=q$，则 r 和 s 也标记为可区别的。

直到没有状态再改变。然后合并等价集合，获得最小化的自动机。