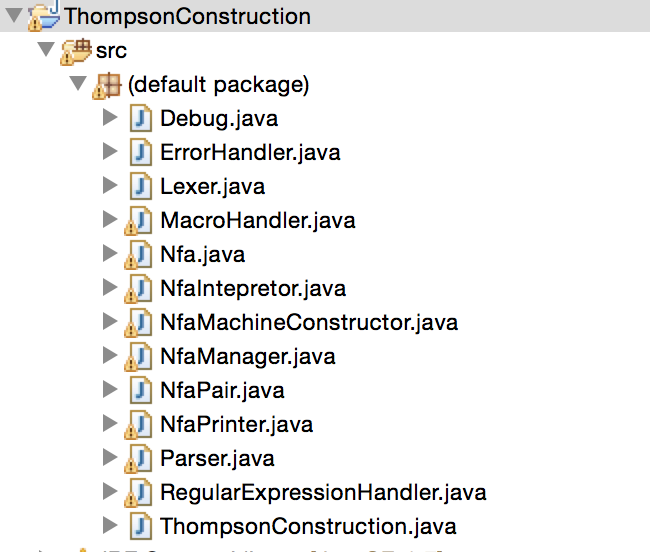
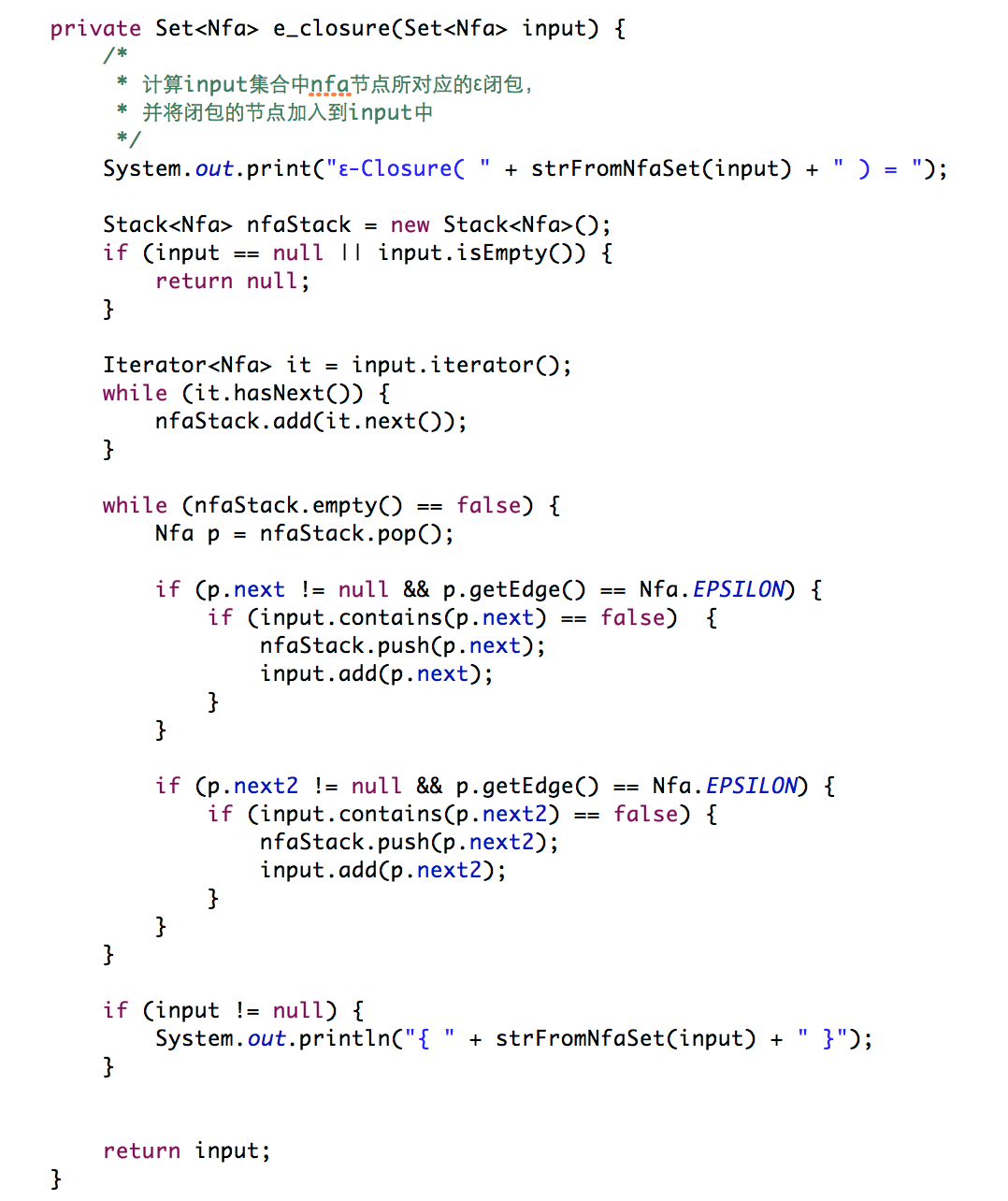
NFA状态机识别输入字符串的代码解析

大家好，欢迎大家来到Coding迪斯尼，上一节，我们探讨了NFA识别输入字符串的算法原理。这一节，我们将详细分析算法的实现原理，并通过代码逐行调试的方式，向大家演示代码的运行流程。

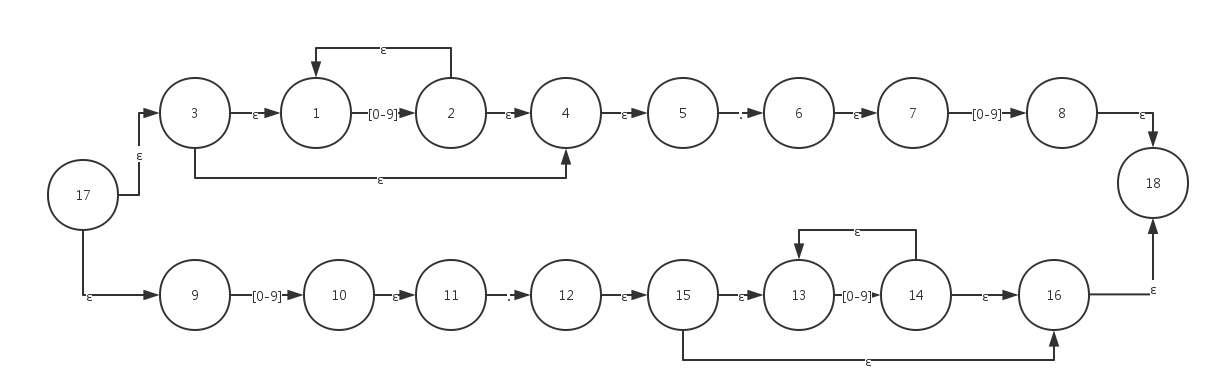


上图是我们当前工程的代码目录，其中的NfaIntepretor.java 的内容就是对NFA状态机识别算法的实现。打开它，我们看看代码内容。

在代码中，最重要的也就两个函数，一个是e\_closure:



大家或许已经想到，这个函数的功能就是计算ε闭包，它的输入参数是一个Nfa对象集合，这个集合就对应于我们上节所说的ε闭包运算的输入集合。它的输出是输入集合的节点通过ε边所能抵达的所有状态节点的集合。我们以上节的NFA状态机图为例，看看该函数是如何计算ε闭包集合的。



该函数第一次被调用时，是状态机启动，进入初始状态的时候，此时，我们要计算初始节点的ε闭包，因此该函数的输入集合只包含节点17. 接着下面这三行将节点17放入到一个栈上：

Iterator<Nfa> it = input.iterator();

**while** (it.hasNext()) {

nfaStack.add(it.next());

}

在接下来的循环中，将会把栈上的节点逐个取出，计算它通过ε边可直接抵达的节点，并把这些节点也放入栈上。我们继续往下看：

**while** (nfaStack.empty() == **false**) {

Nfa p = nfaStack.pop();

**if** (p.next != **null** && p.getEdge() == Nfa.*EPSILON*) {

**if** (input.contains(p.next) == **false**) {

nfaStack.push(p.next);

input.add(p.next);

}

}

**if** (p.next2 != **null** && p.getEdge() == Nfa.*EPSILON*) {

**if** (input.contains(p.next2) == **false**) {

nfaStack.push(p.next2);

input.add(p.next2);

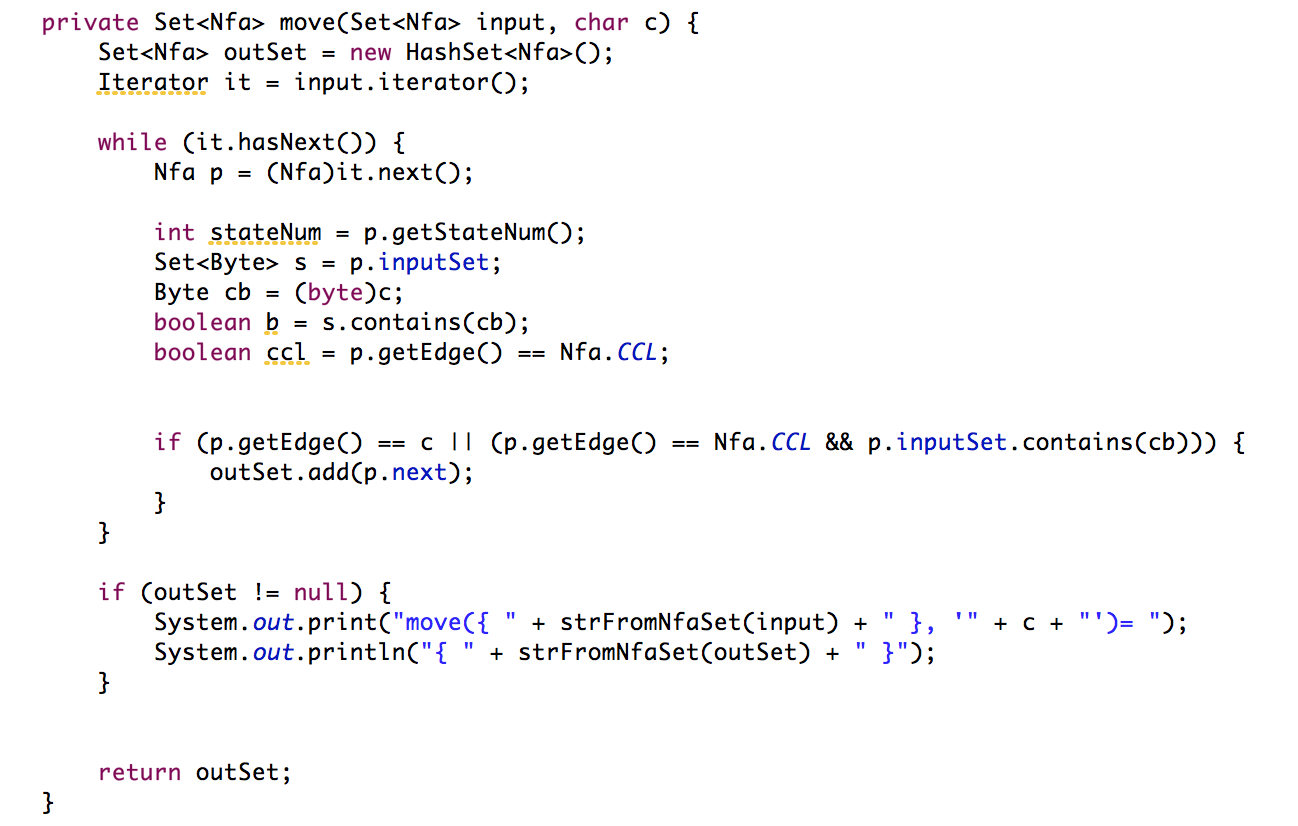
}

}

}

nfaStack.pop() 把节点17取出来，然后看看节点17有没有ε边，显然节点17有两条边满足要求，于是进入if(p.next != null && p.getEdge() == Nfa.EPSILON) 判断分支内，接着代码看看p.next指向的节点是否已经在栈上，节点17的next指向的节点是3，此时3并没有在栈上，所以节点3被压入堆栈，以备以后计算它的ε边可达节点，同理17节点的next2指向节点9，于是节点9也被加入堆栈，同时由于节点3和9是通过ε边抵达的，所以他们属于节点17的ε闭包集合，因此把他们加入input, 在while的第二次循环，节点3被取出来，基于同样的到来，它有两条ε边，分别指向节点1，4，于是1，4被加入堆栈，同时节点1，4也是节点17通过ε边可以抵达的，因此也加入input集合。由此反复运算，所有由节点17经过ε边可抵达的节点都会被加入到input中。最后input集合由于在函数调用前已经包含了ε闭包的初始结合节点，经过函数执行后，input还包含了初始集合节点通过ε边可抵达的所有节点，因此最后在input集合中所包含的节点就是节点17的ε闭包集合。

第二个重要的函数是move, 也就是计算转移集合：



假定我们要计算的转移如下：

move({ 1,3,4,17,9,5 }, '1')

在move函数中，它先将集合{ 1,3,4,17,9,5 }中的节点逐个取出，然后判断节点转移边的类型是否对应单字符或是字符集类，如果是单字符，那么看看节点边的转移字符是否跟move输入的跳转字符相同，如果是字符集类，那么判断字符集类是否包含move函数传入的跳转字符，以上面的move运算为例子，跳转字符是’1’, 在函数运行中，首先将节点1取出，由于节点1的边类型对应的是字符集类[0-9]因此节点1满足判断:

if (p.getEdge() == c || (p.getEdge() == Nfa.CCL && p.inputSet.contains(cb))

于是节点1的next指向的节点2被加入结合outSet，在集合{ 1,3,4,17,9,5 }

中，满足判断条件的还有节点9，于是9的next指向的节点10也被加入到outSet中，这样outSet就包含了结合{ 1,3,4,17,9,5 }应对输入字符1后跳转的集合。

这两个函数是算法的关键，理解了他们，算法流程也就基本清晰了，接下来我将以调试的方式给大家展示算法的实现流程。