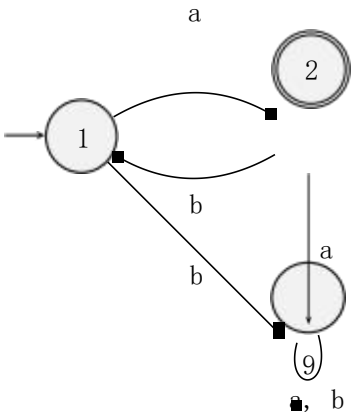
常规语言和自动机：第2部分

1语言等效性

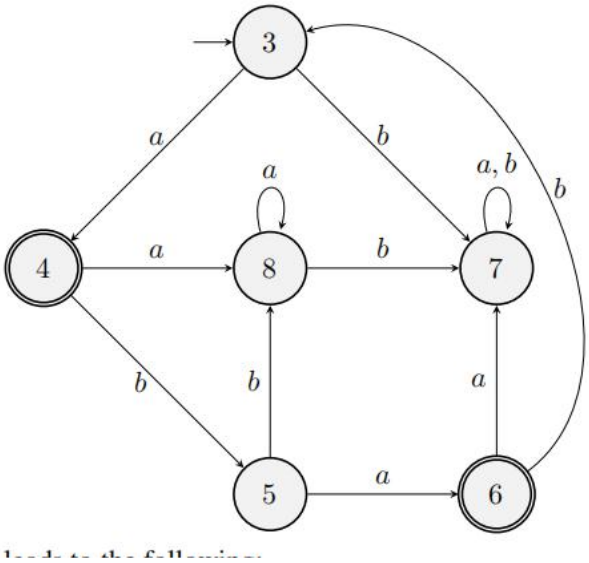
参见画布上的视频“语言等价性”。

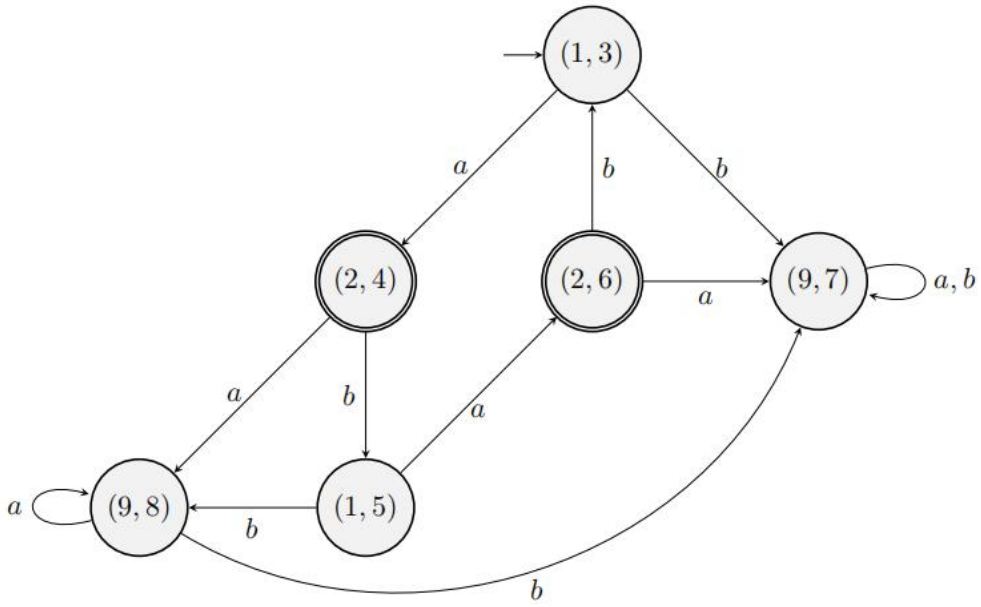
如何测试两个规则程序是否等同于语言并不明显。但是，让我们看看如何测试两个dfa是否与语言相等。

这是第一个建议。从每个自动机的初始状态开始。如果一个接受，另一个拒绝，那么自动机就不是语言等价的(因为一个接受e，另一个不接受)。如果它们都接受或拒绝，那么看看我们通过输入a转换到什么状态，通过输入b转换到什么状态。我们永远下去。例如，比较自动机



和





在这个例子中，我们看到每一对状态都由两个接受状态或两个拒绝状态组成。所以这两个自动机在语言上是等价的。[1]

在两个语言不相等的自动机的情况下，那么有一个词接受，而另一个词不接受。这个过程最终会找到它并告诉我们。

不管怎样，这个过程总是停止，因为只有有限的状态对。

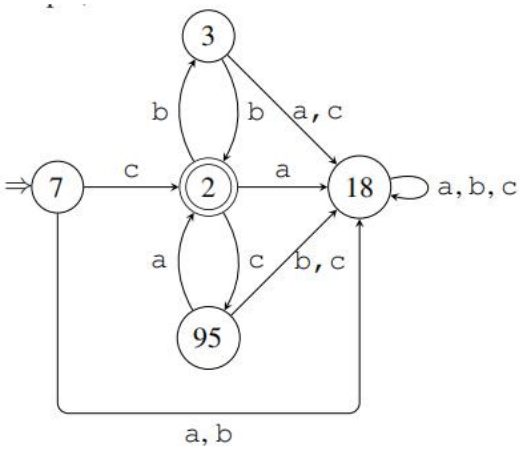
2最小自动机

参见画布上的视频“最小自动机”。

到目前为止，我们还没有担心过自动机的大小。但事实上，某些自动机是微小的，这非正式地意味着它们不能变得更小。准确地说，当一个DFA具有以下两个属性时，就说它是最小的。

1. *每个状态的北都是可访问的。*这意味着有一个从初始状态到北的路径。
2. *任意两个不同的状态北，y都是不相等的。*这意味着北接受了这样一个词。从北到接受状态)，但y拒绝，反之亦然。

为了



我们证明了它是最小的，如下所示。

7可以通过”，2通过c，3通过cb，95通过cc，18通过ca。

2与其他州不等同，因为2接受“它”，而其他州拒绝它。

(9; 7) a; b

[1]请注意，在视频中的过渡 丢失。

3与其他国家不等同，因为3接受b，而其他国家拒绝它。

95与其他州不等同，因为95接受a，而其他州拒绝它。

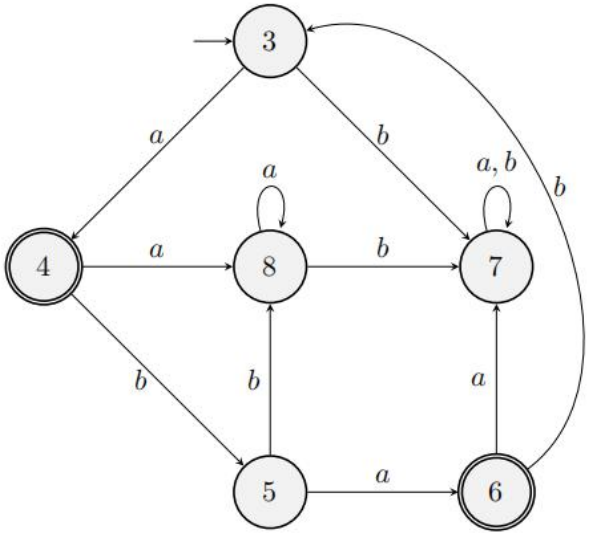
7与其他国家不等同，因为7接受c，而其他国家拒绝它。

3最小化自动机

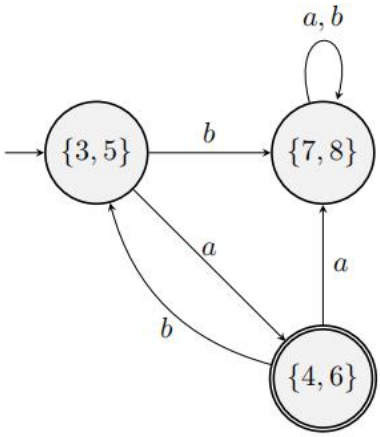
参见画布上的视频“最小化DFA”。

现在让我们来看看如何最小化一个DFA，即。让它尽可能的小。有两个步骤。首先，删除任何不可到达的状态。其次，找到等价的状态，这样我们就可以统一它们。首先，统一所有接受国家和所有拒绝国家。然后尝试绘制a转换和b转换——您可能会发现您需要分割这些块。继续走下去，直到你不需要再分开什么了。

为了说明第二步，让我们取这个自动机



首先，我们有一个接受状态和一个拒绝状态的状态。从46的过渡把我们带到3578，b过渡也是如此。但是当我们试图从{3578}画一个过渡时，我们不能，因为从35的过渡进入{46}但是从78的过渡进入{3578}。所以我们必须把它分成两部分，即。3,5和7,8。现在我们有三个状态，我们可以没有任何问题地完成这个图：



作为一个额外的检查，你应该显示这是最小的。

4非常规语言

参见画布上的视频“非规则语言”。

我们知道有些语言不是规则的，因为只有可数的规则语言和不可数的规则语言。但是有什么有用的非规则语言吗？答案是肯定的。这里有一个例子。

假设一个单词是由开括号a和闭括号b括起来的，我们想知道它是否用括号很好。这是一个常见的问题；例如，当您在Eclipse上编写Java程序时，即jave

检查方括号是否正确匹配。这可以使用一个堆栈来完成。当你读a时，你把卵石推到堆上，当你读b时，你把卵石从堆上弹出来。如果你到达单词的末尾，并且堆栈是空的，那么你就知道这个单词有很好的括号。但是如果它不是空的，或者当堆栈为空时读b，这个单词就没有很好的括号。

让我们来思考一下你的电脑。它有一个有限的内存，因此只有有限数量的许多状态。它可以尝试运行上述程序，分配部分内存来表示堆栈。但是如果单词以大量的a开头，堆栈就会溢出。

你的计算机能运行一个适合所有单词的程序吗？程序应该用一个单词，一个字母一个字母地读出来，然后宣布这个单词是否有大括号。

一种解决方案是使用外部堆栈。这样，即使您的计算机只有有限的内存，堆栈可以占用的内存量也没有限制。但如果你无法访问外部内存呢？你能在你的电脑上安装这样的程序吗？答案是No.我们现在将证明这一点。

为了更严格地讲，我们将证明在括号上很好的单词的集合L是不规则的。假设L是规则的，然

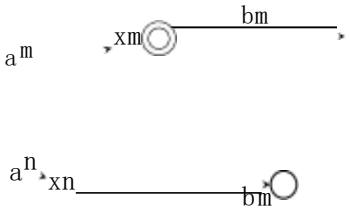
后有一个DFA(X、p、6、Acc)来识别它。让我们这样说

- x0是初始状态p吗
- x1是我们从p开始读取a时所达到的状态吗
- x2当我们从p开始读aa时，我们到达的状态
- x3当我们从p开始读aaa时，我们到达的状态吗

Subar等。

总而言之，xn是我们从p开始读取a时所达到的状态吗n.现在我们要证明这些状态都是不同的，这意味着有无限多的状态，这是一个矛盾。

假设m和n是m<n的自然数。我们想证明这一点xm xn≠.如果我们从xm和阅读bm我们达到了一个可以接受的状态，因为ambm21，但如果我们从xn和阅读bm我们达到了一个不接受的状态，因为anbm\2 L.如此xm和xn不能是一样的。



这是证明一种语言不是规则的一种通用方法。对于另一种语言，您需要调整的定义xn，并调整你的显示方式xm xn≠为m<n。

让字母表是{a，b}。证明a出现次数超过b的一组单词不是规则的。

让我们重复一下要点：一个具有有限多状态、无法访问外部内存的计算机，都不能解决任何非规则语言的匹配问题。例如，它不能确定一个单词（逐字母阅读）是否有很好的括号。