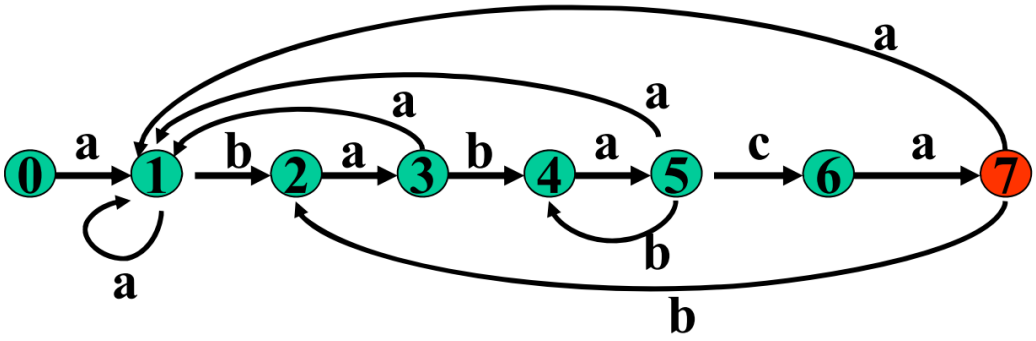
自动机模型的应用

自动机是有限状态机(FSM)的数学模型。 FSM 是给定符号输入，依据转移函数“跳转”过一系列状态的一种机器。在常见的 FSM 的“Mealy”变体中，这个转移函数告诉自动机给定当前状态和当前字符的时候下一个状态是什么。在实际生活中，通常会把一个问题抽象成为几个状态之间的转换，这样就可以应用自动机理论的方法来解决实际的问题。

有穷自动机，或有穷状态的机器，是描述特定类型算法的数学方法。特别地，有穷自动机可用作描述在输入串中识别模式的过程。有穷自动机首先包含一个有限状态的集合，还包含了从一个状态到另外一个状态的转换。有穷自动机看上去就像是一个有向图，其中状态是图的节点，而状态转换则是图的边。此外这些状态中还必须有一个初始状态和至少一个接受状态。有穷自动机在理论研究和实际生活中的应用是非常普遍的。

应用一：字符串匹配算法

例如解决如下问题，在一个由字母组成的字符串中，查找“ababaca”这样的子串。可以定义如下 8 个状态，并且构建出这样一个状态转换图。例如以 C 语言的实现为例，可以定义 8 个结构体并组成如图所示的链表。定义一个指针，初始状态下指针在结构体 0 的位置，然后依次扫描待检测的字符串，根据扫描到的字符决定指针的下一步走向，每扫描一个字符之后判断指针位置是否为结构体 7 的位置，如果是则判断为有这一子串，如果扫描到字符串的最后仍然没有在 7 的状态，则字符串中没有这样的子串。由于空间有限，该图中并未标识出回到初始状态的情况，例如指针在结构体 4 的位置的时候扫描到的字符为 c，这时指针回到结构体 0 的位置。应用有穷自动机算法来实现字符串的查询，时间复杂度为 O(n)，n 为待查找字符串的长度。但是在实际的运行中，由于这种方法不需要任何的回溯，且不需要像 KMP 算法那样待检测字符串中间的字符会被扫描多次，因此实际运行速度比经典的 KMP 算法要快。



应用二：编译器

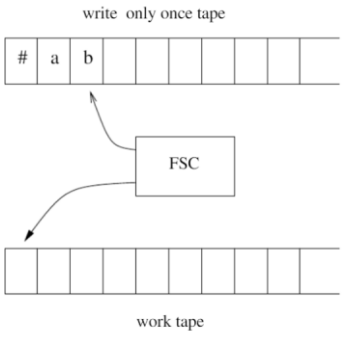
在编译器对代码进行编译的时候，首先要进行词法分析。词法分析是编译器中最底层的分析。构造词法分析器的前提是给出语言中单词结构的定义。将输入程序转化为词法分析器就使用了有穷自动机的表示方法。在词法分析中，需要识别标识符、关键字、运算符、界符、常数、注释等内容，因此需要通过构造有穷自动机来实现，即针对每种需要识别的信息，均需要构造各自对应的自动机。同时在词法分析中也通过把代码输入到自动机之后判断最终的状态来判断词法时候符合规范，并且可以提示出报错信息。例如，在识别常数的时候，还可以构造

出更多种复杂的自动机用来分别识别整形常量、浮点型常量、科学记数法表示的浮点数、二进制整数、八进制整数、十六进制整数等。自动机可以有效地构造出词法分析序列，并将其继续应用到之后的语法分析之中。由于自动机的方法易于实现，并且时间复杂度为线性，使得在编译代码的时候使用有穷自动机来进行词法分析成了最好的方法，而且设计编译器的时候通常使用有穷自动机来进行词法分析。

图灵机指一个抽象的机器，它有一条无限长的纸带，纸带分成了一个一个的小方格，每个方格有不同的颜色。有一个机器头在纸带上移来移去。机器头有一组内部状态，还有一些固定的程序。在每个时刻，机器头都要从当前纸带上读入一个方格信息，然后结合自己的内部状态查找程序表，根据程序输出信息到纸带方格上，并转换自己的内部状态，然后进行移动。每一个会决策、会思考的人都可以被抽象地看成一台图灵机。该模型主要有四要素：输入集合、输出集合、内部状态和固定的程序。如果把人进行抽象，那么输入集合就是所处环境中所看到、听到、闻到、感觉到的一切；输出集合就是人的每一言每一行，还有表情动作；内部状态集合则可以把神经细胞的状态组合看成一个内部状态，所有可能的状态集合将是天文数字。图灵机有着更加广泛的应用。

应用三：枚举器

枚举器是图灵机的一个变种。图灵机可以将打印机用作输出字符串的输出设备。枚举器枚举的语言是最终打印出来的字符串的集合。可以以任何顺序生成字符串，可能重复进行。另外，如果枚举器不停止，则它可能会打印出无限的字符串列表。相比于图灵机，枚举器不需要输入，可以直接进行输出，这种图灵机的变种在生活中有着广泛的应用，比如设计一个霓虹灯系统就可以使用枚举器的思想，只要连上电源，枚举器开机，就可以开始在不需要任何的输入的情况下进行灯光的输出。枚举器的示例如下图所示：



应用四：Universal Turing Machine

Universal Turing Machine 是一种可以接受另一台图灵机的描述作为输入并模拟该图灵机的操作的通用图灵机。要构建通用图灵机，需要设计一种编码方案以服务所有图灵机。每台图灵机从它的字母表得到字元串计算一确定的固定偏可计算函数。从外观上它的行为就像一台使用固定程式的电脑。尽管如此，可以把任何图灵机的动作表格编码到一条字元串。因此，我们可以建构出一台图灵机，它期待的纸带上记载有一条用以描述动作表格的字元串紧跟着一条用以描述输入的字元串，从而计算那台被编码的图灵机所计算的。这种图灵机为现代的计算机结构奠定了基础。这种多用途单机器模型可以“运行”任何任意指令序列。这模型被一些人认为是“存储程序电脑”的原点。存储程序电脑一词由约翰·冯·诺伊曼

使用在他的《电子计算装置》("Electronic Computing Instrument")。之后这个模型演变成为了我们通常所说的冯·诺伊曼结构。