Atitit 词法分析器手工编写还是自动生成？？

[1.1. 词法分析器 1](#_Toc31234)

[1.2. 词法分析器生成器的核心是正则表达式引擎 1](#_Toc16495)

[1.3. 汤普森构造法 1](#_Toc15377)

[1.4.  词法分析器（Lexer）词法分析器又称为Scanner，Lexical analyser和Tokenizer 2](#_Toc24539)

[1.5. 不管是Lexer还是Parser都是一种识别器 2](#_Toc26138)

**[2. Dfa缺点 为什么很多语言的实现里面的 Lexer 都没有使用 DFA？](#_Toc7548)**[修改 2](#_Toc7548)

[2.1. 产品级别的编译器需要对源代码中的各种错误给出友好的提示信息。而DFA在这方面无法满足需要 2](#_Toc31785)

[2.2. DFA有两种实现方式：1. Nested case statement （手写）2. table-driven （自动生成） 3](#_Toc26649)

[2.3. 生成的代码（特别是基于表驱动方式的代码）通常都远不如手写代码可读。ANTLR可能算是个例外吧。 3](#_Toc2321)

[3. 自动实现的语言 4](#_Toc16830)

## 词法分析器

事实上很多编译器的词法分析器也是手写的，这种方式的优点是直观容易理解，缺点是开发效率低且易出错。编译技术经过几十年的发展已经形成了一套成熟的理论，应用这些理论可以让我们实现词法分析器的自动构造。

## 词法分析器生成器的核心是正则表达式引擎

，下面将会介绍一些基础理论，以及如何实现一个基本的正则表达式引擎。

## 汤普森构造法

汤普森构造法是**[C语言](http://lib.csdn.net/base/c" \o "C语言知识库" \t "http://blog.csdn.net/xinghongduo/article/details/_blank)**&Unix之父之一的肯·汤普森(Ken Thompson)提出的构造识别正则表达式ε-NFA的方法，其原理非常简单，先构造识别子表达式的ε-NFA，再通过几个简单的规则将ε-NFA合并，最终得到识别完整正则表达式的ε-NFA。汤普森构造法的优点是构造速度快，且构造的ε-NFA状态数较少。

## ****词法分析器（Lexer）****词法分析器又称为Scanner，Lexical analyser和Tokenizer

## 不管是Lexer还是Parser都是一种识别器

，Lexer是字符序列识别器而Parser是Token序列识别器。他们在本质上是相似的东西，而仅仅是在分工上有所不同而已

# **Dfa缺点 为什么很多语言的实现里面的 Lexer 都没有使用 DFA？**[修改](https://www.zhihu.com/question/javascript:;)

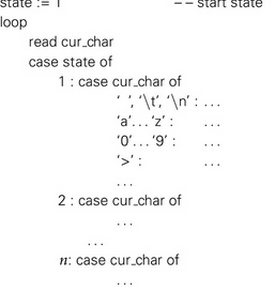
最近看完龙书的 Front end 部分，也参观了很多语言的实现，比如 Python（CPython）、Lua、Ruby（CRuby），发现他们在词法分析部分基本上都没有使用 DFA 来写，请问，龙书花了那么多篇幅讲的有限自动机为什么没有在这些语言的实现里面看到？

Lua也是同理：词法语法都简单，而且要尽量不依赖外部工具，所以选择手写是很自然的。  
  
毕竟，自己手写的代码，逻辑全部都浮现在代码中，容易让人读懂，也容易调试

## 产品级别的编译器需要对源代码中的各种错误给出友好的提示信息。而DFA在这方面无法满足需要

## DFA有两种实现方式：1. Nested case statement （手写）2. table-driven （自动生成）

然后给了你代码框架（这里以nested case statement 为例）：



然后告诉你最外层case覆盖DFA的所有状态，内层每个case覆盖这个状态的所有转换。特别清晰简单有木有！！！编程时候直接填空就行了！

其实手写的lexer/scanner也是一种实现DFA的方式，只是状态迁移可能隐含在代码中而没有一个显式的“状态”变量

## 生成的代码（特别是基于表驱动方式的代码）通常都远不如手写代码可读。ANTLR可能算是个例外吧。

Ruby的词法和语法极其复杂，它的词法分析其实是上下文相关的，必须要让parser能反过来向lexer传递状态才可以继续分析下去。这超出了flex的常见使用场景，用它写反而不方便，于是就手写了。同样的情况在IronRuby也可以看到，parser是用GPPG生成的，但lexer是手写的。  
  
作者：RednaxelaFX  
链接：https://www.zhihu.com/question/28121746/answer/39479336  
来源：知乎  
著作权归作者所有，转载请联系作者获得授权。

# 自动实现的语言

Groovy就是个好例子。Groovy语法：[groovy-core/groovy.g](https://link.zhihu.com/?target=https://github.com/groovy/groovy-core/blob/master/src/main/org/codehaus/groovy/antlr/groovy.g" \t "https://www.zhihu.com/question/_blank)，parser和lexer都是用ANTLR生成出来的。不过Groovy的实现好不好也见仁见智…

DFA这样的东西是用来实现自动化工具的，但是“人肉”实现Lexer就完全不必了，不过状态机是重要而且需要的，与此同理，语法分析器也有First和Follow集合等理论（也是用于ANTLR等自动化工具），其实实际编写也很多是“人肉观察法”加直接递归下降就完了，很少去分析其First和Follow集合  
  
作者：蓝色  
链接：https://www.zhihu.com/question/28121746/answer/39478633  
来源：知乎  
著作权归作者所有，转载请联系作者获得授权。