Petunia 项目开发记录

陆巍

2023 年 10 月 5 日

前言

目录

第一章　词法分析

1.1　状态转换图

第二章　开发日记

2.1　2023 年 10 月

2.1.1　10 月 5 日

这个项目最初只是打算使用简单的判断方法来解决，但想到以后要开发其他编译器，所以先用这个微小项目来练练手。项目将按照编译器开发方法来实现，当然，本项目过于微小，大概只会用到词法分析与语法分析。

Windows 系统和 Linux 系统在文本处理上是有些差异的，其中的换行就不相同。Windows 系统中的换行实际上包含了两个字符，即 $\backslash r$（回车，0xD）与 $\backslash n$（换行，0xA），而 Linux 系统中只有 $\backslash n$（换行，0xA）。我现在主要使用的是 Linux 系统，但为了兼顾 Windows 系统，可能需要在读入配置文件后，先把其中的 $\backslash r\backslash n$ 替换成 $\backslash n$，然后才做词法分析。

目前暂时只解析裸键名，引号键名以后再考虑。

2.1.2　10 月 6 日

在绘制状态转换图时，我们看到在识别某些内容时，可以按照不同的权衡有不同的处理方式。例如在判断整数时，可以在出现非数字符号就截止，也可以规定必须要出现空格、换行符或 # 才截止，两种方式一个宽松，一个严格，各有各的好处与不足。前一种方式对 TOML 的书写格式比较宽松，但也因为过于宽松可能导致混乱，并增加后期处理的负担。后一种方式要求严格，书写时会有更多约束，但可以减少后期处理的工作量。这里说的后期处理主要是指语法分析阶段。

10 月 7 日

随着状态转移图绘制的深入，会让人感到越来越繁琐，或许应该创建一个专门的工具来绘制，并且在绘制完成后自动转换成相应表格直接供词法分析器调用。这个工具的原理并不复杂，麻烦的是图形操作方面的支持问题，这将涉及到图形库方面，这是一个老话题了，先放一放。

2.1.3　10 月 12 日

绘制状态转换图时，我曾经想到对于不合法的符号要如何在图中去处理，这是一种流程图的思维习惯。实际上，在状态转换图中并不需要显式指明如何处理不合法的符号，而是已经暗含了处理方式。合法的符号串可以从状态转换图的开始（start）处走到某一个终点，不合法的符号是没有路径的，在程序处理上会自动跳到错误处理模块，通常会向用户报告某行某列出现词法错误。通常情况下，每发现一个错误就退出程序并报告此错误，也可以把每一行视为一个单元，全部扫描后统一报告。全部扫描的方式还有一些细节问题需要考虑，并非简单的逐行处理就可以。

在把状态转换图映射为状态转换表时，需要把使用到的符号、状态都列出来，这项工作的繁琐程度会随着语言的复杂程度的增加而增加。对于本项目，即使只是简化版的，其状态转换图已经有些繁琐。