设计文档 21371064 权尚浩然

参考编译器介绍

在自己设计编译器之前，我主要阅读了往届学长们的课设作品。我选择的是C++实现编译器，在我阅读的几个github上开源的C++版本中，一般会写一个主文件专门负责读入与输出，进行与文件的交互，然后词法分析、语法分析、中间代码生成、代码优化分别单列目录，每一个目录下对于不同成分（比如语法分析中不同的非终结符、优化部分中不同的优化）再单设文件，仅描述这一个成分，最后用include,extern把各个文件串联起来。

编译器总体设计

由于我本人习惯能不多加文件就不多加，又加之这是独立开发作业，我选择在一个cpp文件中实现所有功能，从前到后分别为词法分析、语法分析（附带生成中间代码）、中间代码优化及目标代码生成，在主函数里完成文件读取，再顺序调用各子结构，最后完成结果的输出。

词法分析设计

词法分析其实比较简单，只需要识别特定单词即可，手工实现过程中可以简单模拟、而无需掌握或使用高级理论（例如Lex）。基本上是先将待编译代码读入到字符数组中，一个个读取，对于空格、换行、注释等进行判断。将需要识别的单词直接塞进一个map里，通过字符串比较来识别字符串。具体实施过程中没有遇到太多困难，大体按照之前设计完成。

语法分析设计

语法分析是本门课程中遇到的第一个难点（准确说倒也不难，但属于第一次接触需要时间领悟），由于理论课程上也刚好介绍到了文法改写与递归下降，我基本打算按照老师讲解的来。当时一开始觉得很困难而无从下手，但开始写出第一个递归子程序后，发现基本就是对着文法产生式写递归子程序就好了，所以我每次把下一个非终结符的产生式复制过来，对着翻译，实现出递归下降子程序，然后后面重复性工作就很简单了。此外对于后面的表达式文法是以左递归给出的，需要先进行文法改写。语法分析作业需要在每一个语法成分分析完成前输出这一成分，我记得当时编写好之后有奇怪的BUG，比如少输出成分或者直接死循环，由于有辅助测试样例，对着找错、改错就可以了。

错误处理设计

错误处理给了各种需要考虑的错误的错误方式，基本上就是对于每一种错误进行数据点分治解决。但是难点在于在语法分析的基础上，它要求我们能处理一些语义错误，比如使用的变量未定义。因此这部分作业实际上也在考察语义分析。语义分析是直接在递归下降子程序的相关位置进行编写的，而错误处理也基本上就是要找到我们该在哪里进行这个错误的判断。

代码生成设计

代码生成是第二个难点。代码生成先要做语义分析，有了进行错误处理的设计之后会容易一些，此外很重要的是设计中间代码，对于中间代码的设计我选择了自己进行摸索而不是直接照用。我采用了四元式，由于当时的需求是生成代码，我开始设计的中间代码很多都是很接近底层，基本很接近mips汇编了的那种。具体做的时候就是把很多判断、处理都放在了递归子程序里，对于每一种情况都进行讨论、设计、解决。最后完成的效果就是中间代码情况数比较多，对于每一种情况很具有针对性但总体缺乏普遍性。这很好地帮我完成了代码生成，但在后续优化过程中，发现中间代码类别太多、有的太过细节，讨论起来很麻烦。因此后来以优化为目标后，我重构了一次中间代码，将很多之前的中间代码拆分成多个更加简单而通用的中间代码。最后我发现很多设计与郭佬的四元式不谋而合。

代码优化设计

自己看我的中间代码时，我发现有很多明显冗余的代码，比如连续赋值等等。在看了郭佬的申优文档后，我决定仿照他的文档写SSA优化。后来去答疑掌握了SSA的结构精髓，用那周的空闲时间写成了SSA，并打算完成SSA+GVN+图着色寄存器分配。然而神奇的事情是从那两周之后，由于到了下半学期，各个课程压力增大，我竟再难找出大块时间写编译器了。最终没有完成寄存器分配，竞速部分等同于摆烂。这块确实是我时间统筹出了问题，其实不写SSA，先写寄存器分配，结果应该会好很多。