引论

czh

2019 年 8 月 27 日

目录

1 语言处理器

编译器本身就是一个程序，它可以阅读其他语言编写的程序，并把该程序翻译成一个等价的、用另一种语言编写的程序。解释器则不通过编译的方式生成目标程序，他利用程序提供的输入执行源程序中指定的操作。

编译器生成的程序在通常情况下比一个解释器快的多，但解释器的错误诊断通常比编译器更好，因为他逐行执行语句。

对一个语言处理系统而言，创建一个可执行程序还需要很多东西。把源程序聚合在一起的任务由预处理器 (preprocessor) 完成。经过预处理处理

后，源程序交由编译器。编译器处理后，源程序变为目标汇编程序。目标编译程序交由汇编器 (assembler) 处理，并生成可重定位的机器代码。到此基本结束，但大型程序可能有链接器和加载器，链接器一般解决外部内存地址，加载器把所有可执行的目标文件放在一起执行。

2 编译器的结构

编译器由两部分组成：分析部分和综合部分。也就是我们常说的编译器前端和后端。

前端把程序分解为多个组成要素，并在这些要素上加上语法结构，然后用这个结构创建源程序的一个中间表示，并检查语法错误。他还会收集有关源程序的信息，并把信息放在一个称为符号表的数据结构中。

后端部分根据中间表示和符号表中的信息来构造用户期待的目标程序。

2.1 词法分析

词法分析器读入组成源程序的字符流，并把他们组织成有意义的词素 (lexeme) 的序列. 输出一般为 <token-name,attribute-name>

2.2 语法分析

语法分析器使用由词法分析器生成的各个词法单元的第一个分量来创建树形的中间表示. 可在原文档中见到.

2.3 语义生成

语义生成器 (semantic analyzer) 使用语法树和符号表中的信息来检查源程序是否语言定义的语义一致.

2.4 中间代码生成

一个编译器可能构造出一个或多个中间表示

2.5 代码优化

机器无关的代码优化步骤试图改进中间代码, 以便生成更好的目标代码.

2.6 代码生成

代码生成器以源程序的中间表示形式作为输入, 并把他映射到目标语言.

2.7 符号表管理

符号表数据结构为每个变量名字创建了记录条目, 我们很快就能见到.

2.8 将各个步骤组合成趟

多个步骤可以合成一趟 (pass), 每趟读入一个输入文件并产生一个输出文件.

2.9 编译器构造工具

一些常用的编译器构造工具包括:

(1) 语法分析器的生成器

(2) 词法分析的生成器

(3) 语法制导的翻译引擎

(4) 代码生成器的生成器

(5) 数据流分析引擎

(6) 编译器构造工具集