编译原理/Compiler
--C语言描述

周雅倩
复旦大学计算机科学技术学院
zhouyaqian@fudan.edu.cn
2018/9/2

编译原理

主要内容

- 上课注意事项
- 什么是编译？
- 编译器历史
- 编译过程简介

2018/9/2

编译原理

课程内容

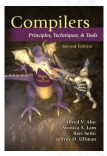
- 讲述将程序设计语言转换成可执行代码时使用的技术、数据结构和算法。

2018/9/2

编译原理

Text Books

- "Tiger Book"
  - 现代编译原理--C语言描述(英文影印版)
  - Andrew W. Appel, Maia Ginsburg
  - 出版社:人民邮电出版社 出版日期:2005年9月
- "Dragon book"
  - Compilers: Principles, Techniques, and Tools (2nd Edition)
  - Alfred V. Aho;Monica S. Lam;Ravi Sethi;Jeffrey D. Ullman
  - Publisher: Addison-Wesley Pub Co



编译原理

Other References

- http://dragonbook.stanford.edu/
- Stanford CS143: Compilers
- Coursera
  - https://www.coursera.org/
  - Compilers

编译原理

成绩

- 程序项目 30% （C++, Java）
- 6-8次课后作业 5%
  - 可以用电子版写好，交打印稿
  - 工具：Graphviz
    - http://www.graphviz.org/
    - The DOT Language
- 期末考试 65%

编译原理

教学进度安排-1

- 第1周(9月12日)：简介
- 第2周(9月19日)：词法分析1
- 第3周(9月26日)：词法分析2
- 第4周(10月3日)：国庆节放假
- 第5周(10月10日)：语法分析1
- 第6周(10月17日)：语法分析2
- 第7周(10月24日)：语法分析3
- 第8周(10月31日)：抽象语法树
- 第9周(11月7日)：语法分析器自动生成，语义分析

编译原理

教学进度安排-2

- 第10周(11月14日)：活动记录
- 第11周(11月21日)：中间代码生成
- 第12周(11月28日)：基本块和生成分析
- 第13周(12月5日)：指令选择
- 第14周(12月12日)：寄存器分配
- 第15周(12月19日)：项目分享
- 第16周(12月26日)：项目分享

编译原理

语言是什么？

- 据不完全统计，世界上的编程语言超过 2500种。
- 不同语言的区别是什么？

编译原理

程序语言

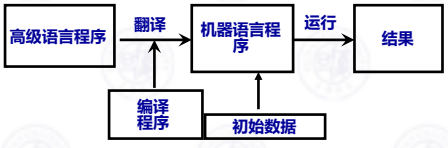
- 程序设计语言：用来编写计算机程序的语言。

程序设计语言
- 高级语言
  - 过程式语言 Fortran, Pascal, C
  - 函数式语言 Lisp
  - 逻辑式语言 Prolog
  - 对象式语言 C++, Java, C#
- 低级语言：面向机器的语言
  - 汇编语言
  - 机器语言

编译原理

什么是编译程序？

- 编译程序(compiler)
- 把某一种高级语言程序等价地转换成另一种低级语言程序(如汇编语言或机器语言程序)的程序
  - 诊断编译程序
  - 优化编译程序
  - 交叉编译程序
  - 可变目标编译程序



编译原理

编译器发展历史

IBM 704大型主机

- 1951年：John Backus (IBM) 针对汇编语言的缺点着手研究开发FORTRAN语言(Formula Translation)。
- 同时，Noam Chomsky开始了他对自然语言结构的研究。他的发现最终使得编译器的结构异常简单，甚至还带有了一些自动化。
  - 形式语言
- 1957年，IBM开发出第一套FORTRAN语言，是第一个被实现出具备完整功能的编译器。
- 1962年，Tim Hart 和Mike Levin (MIT) 第一个自编译语言LISP（LISt Processor，即列表处理语言）。
- 1976年，Fortran 77，具有结构化特性的编程语言。Fortran77在短时间内取得了巨大的成功，广泛地应用于科学和工程计算，几乎统治了数值计算领域。
- 1970s，Pascal和C语言开始流行
- 1995年，Sun公司发布Java，实现了“一次编写，到处运行”的跨平台特性。
- 。。。

编译器发展历史

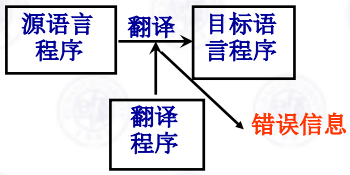
- 复旦大学 1956年研制成功我国第一台电子模拟计算机“复旦601型电子积分机”，
- 1971年开发了我国第一个ALGOL-60编译器。

编译简介

- 本课程介绍程序设计语言编译程序构造的基本原理和基本实现技术.
  - 词法分析
  - 语法分析
  - 中间代码产生
  - 优化
  - 目标代码产生
  - 编译器自动构造工具

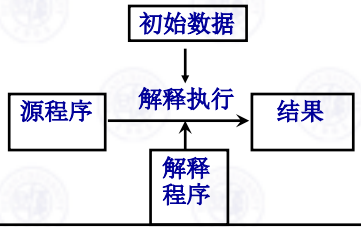
什么是编译程序？

- 把某一种语言程序(称为源语言程序)等价地转换成另一种语言程序(称为目标语言程序)的程序。



什么是解释程序？

- 解释程序(Interpreter) 把源语言写的源程序作为输入，但不产生目标程序，而是边解释边执行源程序本身



编译程序vs解释程序

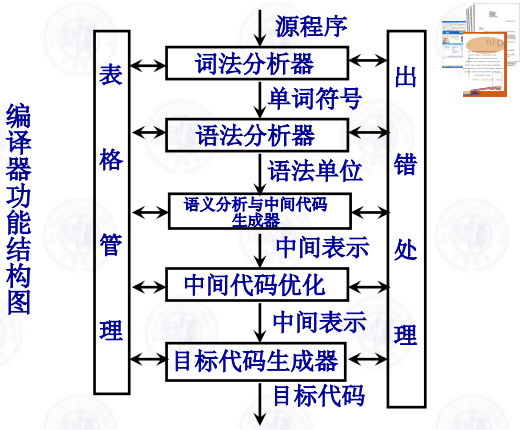
- 解释器是源程序的执行系统
- 编译器是源程序的转换系统
- 编译程序要比解释程序高效

二. 编译过程

- 把英文翻译为中文
  - 识别出句子中的一个个单词；
  - 分析句子的语法结构；
  - 根据句子的含义进行初步翻译；
  - 对译文进行修饰；
  - 写出最后的译文。

编译过程

- 编译程序的工作一般分为五个阶段:
  - 词法分析
  - 语法分析
  - 中间代码产生
  - 优化
  - 目标代码产生



1. 词法分析(Lexical Analysis)

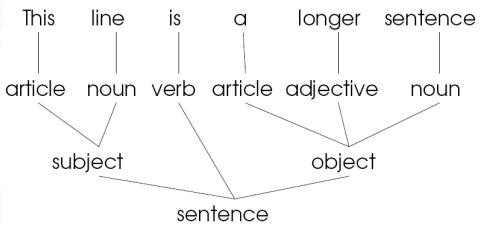
- 任务:
  - 输入源程序，对构成源程序的字符串进行扫描和分解识别出一个个单词符号。
    - 依循的原则：构词规则
  - 删除无用符合（空格符，回车符）
  - 删除注释
  - 检查词法错误
- 描述工具：正则表达式和有限自动机
- FOR I := 1 TO 100 DO
- 关键词 标识符 等符 整常数 关键词 整常数 关键词

例子：Z = X + 0.618 * Y;

2. 语法分析(Syntax Analysis)

- 任务:
  - 在词法分析的基础上，根据语言的语法规则把单词符号串分解成各类语法单位。
  - 检查语法错误
  - 依循的原则：语法规则
  - 描述工具：上下文无关文法

生成语法树



例子：Z := X + 0.618 * Y
算术表达式，赋值语句（层次结构分析）

语义分析(Semantic Analysis)

- 英语里的语义分析：
  Jack said Jerry left his assignment at home.
  ◈What does “his” refer to? Jack or Jerry?

```
{
  int Jack = 3;
  {
    int Jack = 4;
    System.out.println(Jack);
  }
}
```

程序语言用严格的规则来消除二义性！

编译原理

3. 中间代码生成 (Intermediate Code Generation)

- 任务:
  - 对各类不同语法范畴按语言的语义进行初步翻译。
- 依循的原则:
  - 语义规则
- 中间代码:
  - 三元式、四元式、抽象语法树等

- Z=X + 0.618 * Y 翻译成四元式为:
- * 0.618 Y T1
- \+ X T1 T2
- = T2 - Z

编译原理

4. 优化(Optimization)

- 任务
  - 对于前阶段产生的中间代码进行加工变换，以期在最后阶段产生更高效的目标代码。
- 依循的原则
  - 程序的等价变换规则
- 优化标准
  - 空间指标
  - 时间指标
- 优化方式
  - 局部优化
  - 全局优化

编译原理

优化

Ex:

```
for(k = 1;k<100;k++){
    x = i+1;
    m = i+10*k;
    n = j+10*k;
}
```

编译原理

中间代码（一）

序号	OPR	OPN1	OPN2	RESULT	注释
(1)	:=	1		K	K=1
(2)	j<	100	K	(10)	if (100<K) goto (10)
(3)	+	I	1	X	X=I+1
(4)	*	10	K	T1	T1=10*K
(5)	+	I	T1	M	M=I+T1
(6)	*	10	K	T2	T2=10*K
(7)	+	J	T2	N	N=J+T2
(8)	+	K	1	K	K=K+1
(9)	j			(2)	goto (2)
(10)					

编译原理

转换后的等价代码（二）

序号	OPR	OPN1	OPN2	RESULT	注释
(1)	+	I	1	X	X:=I+1
(2)	:=	I		M	M:=I
(3)	:=	J		N	N:=J
(4)	:=	1		K	K:=1
(5)	j<	100	K	(10)	if (100<K) goto (10)
(6)	+	M	10	M	M:=M+10
(7)	+	N	10	N	N:=N+10
(8)	+	K	1	K	K:=K+1
(9)	j			(5)	goto (5)
(10)					

编译原理

5. 目标代码产生

- 任务
  - 把中间代码变换成特定机器上的目标代码。
  - 通常为汇编
- 依赖于硬件系统结构和机器指令的含义
- 目标代码三种形式:
  - 绝对指令代码: 无需连接可直接运行
  - 可重新定位指令代码: 需连接装配后运行
  - 汇编指令代码: 需要进行汇编

5. 目标代码产生

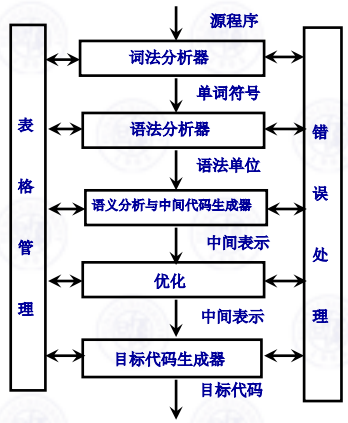
- 例: b=a+2

```
MOV a, R1
ADD #2, R1
MOV R1, b

0001  01  00  00000000 *
0011  01  10  00000010
0100  01  00  00000100 *
L=00001111
0001  01  00  00001111
0011  01  10  00000010
0100  01  00  00010011
```

中间语言(Intermediate Languages)

- 许多编译器在连续的中间形式进行转换
  - 除了源语言和目标语言，中间的形式都是中间语言
- 中间语言一般从低到高排序
  - 最高是原语言
  - 最低是机器语言



1. 错误处理(Error Handler)

- 出错处理程序：发现源程序中的错误，把有关错误信息（位置、类型）报告给用户
  - 词法错误
    - 5x，256.
  - 语法错误
    - X=(A*(B+C)+1
  - 语义错误
    - 静态语义错误：上下文有关的错误（重复声明）
    - 动态语义错误：下标超界，除以0

2. 符号表和符号表管理 (Symbol Table Management)

- 常见的符号表:
  - 符号名表，常数表，标号表，入口名表，四元式表和过程引用表。
- 格式:

名字	信息

例: PASCAL程序段：

```
PROCEDURE INCWAP(M,  N:INTEGER);
LABEL  START;
VAR
  K:INTEGER;
BEGIN
START:
     K:=M+1;
     M:=N+4;
     N:=K;
END
```

表0.1 符号名表SNT

	NAME	INFORMATION
	M	形式参数，整型，值参数
	N	形式参数，整型，值参数
	K	整型，变量

表0.2 常数表CT

	值(VALUE)
(1)	1
(2)	4

表0.3 入口名表ENT

	NAME	INFORMATION
(1)	INCWAP	二目子程序，入口四元式：1

表0.4 标号表LT

	NAME	INFORMATION
(1)	START	四元式：(4)

表0.5 四元式表QT

	OPR	OPN1	OPN2	RESULT
(1)	link			
(2)	par	INCWAP	1	M
(3)	par	INCWAP	2	N
(4)	+	M	1	K
(5)	+	N	4	M
(6)	:=	K		N
(7)	return			

遍(pass)

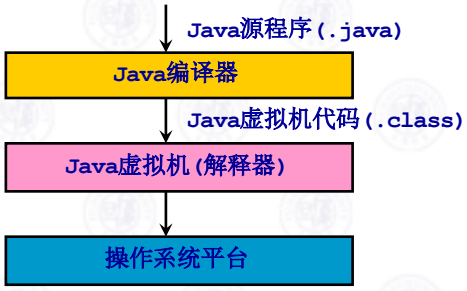
- 所谓"遍"， 就是对源程序或源程序的中间表示从头到尾扫描一次。
- 阶段与遍是不同的概念。一遍可以由若干段组成，一个阶段也可以分若干遍来完成。

编译前端与后端

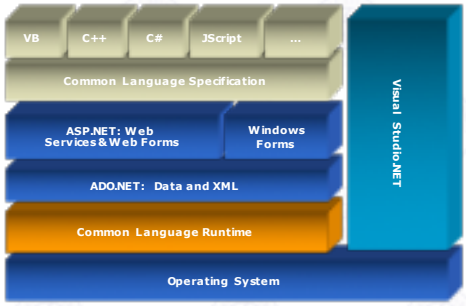
源语言 → 前端 → 中间语言 → 后端 → 目标语言

- 编译前端：与源语言有关，如词法分析，语法分析，语义分析与中间代码产生，与机器无关的优化
- 编译后端：与目标机有关，与目标机有关的优化，目标代码产生
  - ◈ 优点：减少对内存容量的要求，程序逻辑结构清晰；优化更充分，有利于移植。
  - ◈ 不足：编译程序运行的效率低

JAVA语言



.NET Framework与VS.NET



四.编译程序与程序设计环境

- 程序设计环境
  - 编辑程序
  - 编译程序
  - 连接程序
  - 调试工具
- 集成化的程序设计环境

与编译器相关的程序

- 解释程序
- 汇编程序
- 连接程序
- 装入程序
- 预处理器
- 编辑器
- 调试程序
- 描述器
- 项目管理程序

五.编译程序生成

- 设计编译程序应考虑：可移植性、可维护性、可扩展性
- 编译程序的性能：可靠性、编译速度、目标代码的运行速度、空间节省
- 设计编译程序的过程：
  - 确定编译程序的组织结构和各部分的功能、方案
  - 确定使用工具、编写各组成部分

五.编译程序生成

- 编译器中的主要数据结构：
  - 记号（token）
  - 语法树（syntax tree）
  - 符号表（symbol table）
  - 常数表（literal table）
  - 中间代码（intermediate code）
  - 临时文件（temporary file）

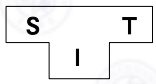
五.编译程序生成

- 编译工具的选择
  - 汇编
  - 已有编译程序的高级语言
  - 自动化工具

以汇编语言和机器语言为工具

- 优点：可以针对具体的机器，充分发挥计算机的系统功能。生成的程序效率高。
- 缺点：程序难读、难写、易出错、难维护、生产的效率低。

高级语言书写



S 源程序　T 目标程序　I 实现语言

- 优点：程序易读、易理解、容易维护、生产的效率高。
- 缺点：难以充分发挥计算机的系统功能，生成的程序效率低。

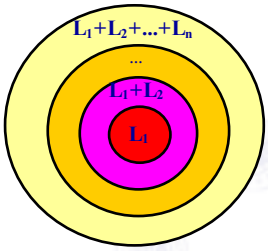
移植法

- 假设在A机上已有L语言的编译程序，想在B机上开发一个L语言的编译程序
  - 直接转换法：将L语言编译程序的A机代码直接转换成B机代码
  - 交叉编译：在A机上产生B机的目标代码

- 问题1：A机上有一个C语言的编译器，是否可利用此编译器实现B机上的C语言编译器？
- 解决：
  - 用C语言写一个B机上的编译程序（P0：C－>B)
  - 用A机上的C编译器（P1）编译P0，得到可在A机上运行的“B机上的编译器”（P2：C－>B)
  - 在A机上用P2编译P0，得到可在B机上运行的C编译器（P3）

自展技术

- Li语言能写出Li+1语言的编译程序



编译程序自动产生

- 编译程序书写系统

FLEX 词法分析程序产生器
YACC(Bison) 语法分析程序产生器
更多的自动生成工具：Javacc,ANTLR,JFlex等

编译发展

- 程序语言
- 机器结构
- 语言理论
- 算法
- 软件工程

（程序）语言

- 输入和输出
- 目标
  - 怎样给计算机发出指令
  - 怎样使计算机有效地执行指令

（程序）语言

- 自然语言
  - 中文，英文
  - 我们的最终目标，但是远没有实现！
- 程序语言
  - 汇编,Fortran, Basic, C, C++, Java, C#

程序语言

- 程序语言应当有如下特性：
  - 无二义性
  - 准确性
  - 简洁性
  - 表达性
  - ？

机器结构

- 16位、32位、64位
- 超线程
- 并行编译
  - SIMD
  - MIMD
  - 分布式计算

其他

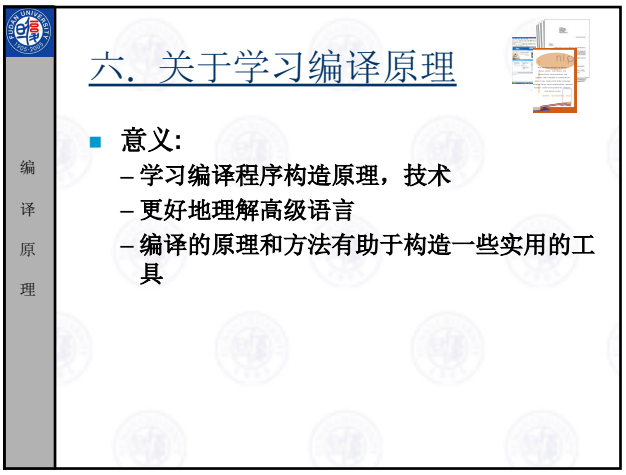
- 程序语言
- 机器结构
- 语言理论
- 算法
- 软件工程

编译器现状(Compiler Today)

- 总体框架保持不变
- 但是各部分的重要程度与Fortran相比已经发生了很多变化
  - 早期：词法、语法分析是更复杂、高代价的
  - 目前：着重点已经转到其他阶段的优化，而词法、语法分析相对廉价的

关于学习编译原理

- 构造编译程序的前提:
  - 掌握源语言
  - 掌握目标语言
  - 掌握编译方法


编
译
原
理
六. 关于学习编译原理
意义:
– 学习编译程序构造原理，技术
– 更好地理解高级语言
– 编译的原理和方法有助于构造一些实用的工具