

编译原理

班晓娟

编译课程的内容：

- 介绍编译程序的工作原理与构造方法；
- 详细介绍如何将一个用高级语言编写的源程序翻译成机器指令程序。

第一章 编译概述

本章内容

1.1 程序设计语言

1.2 什么叫编译程序

1.3 编译程序的组成

1.4 编译程序的生成

1.1 程序设计语言

- Machine Code
- Assembly Language
- High-level Language

如Algol, Fortran, Pascal, C语言等。

1.2 什么叫编译程序

翻译程序：就是将源程序转换成目标程序的程序。

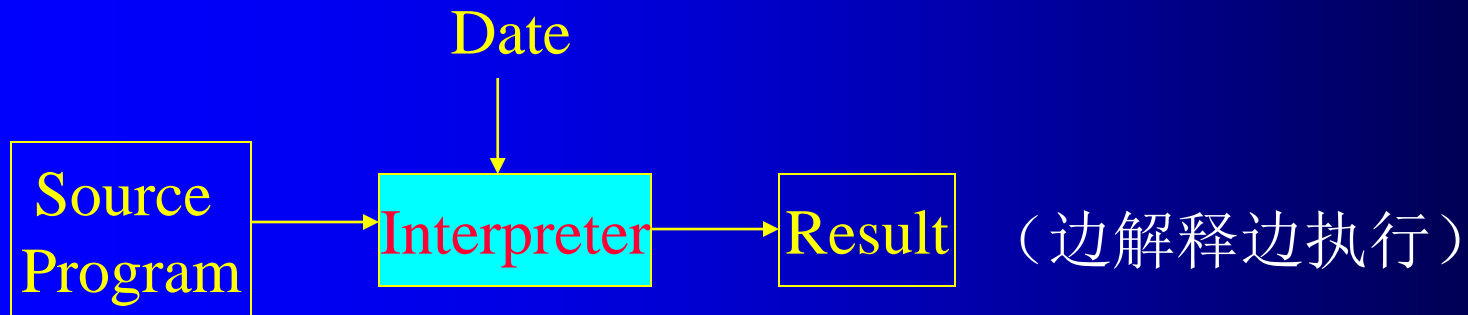


High-level Language

Low-level Language

Compiler

翻译程序和解释程序的区别：



编译程序和汇编程序的区别：



1.3 编译程序的组成

将 “I wish you success”译成中文的过程：

1、词法分析： 单词： I, wish, you, success

词性： 代, 动, 代, 名

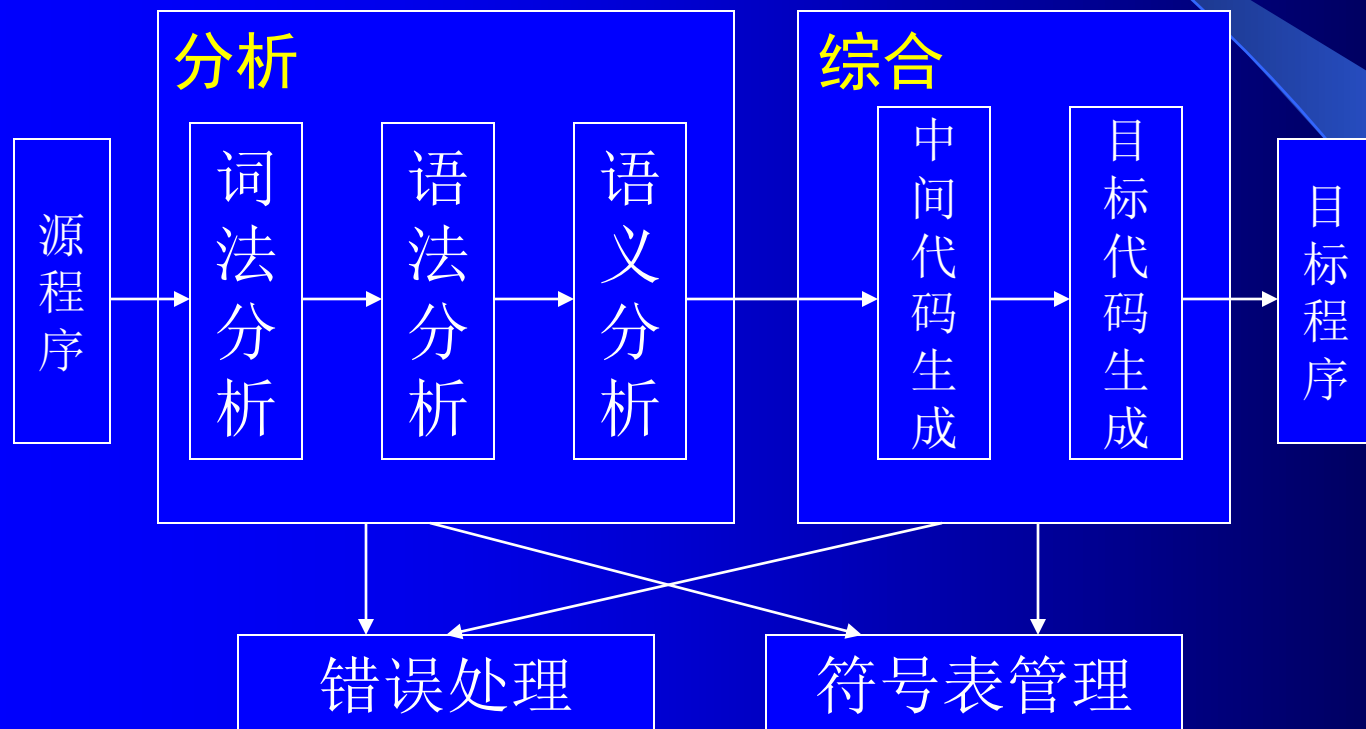
2、语法分析： 语法单位： 主, 谓, 间宾, 直宾

3、语义分析： 语义： 我, 祝愿, 你, 成功

4、中间形式生成： 我祝愿你成功

5、目标生成： 祝你成功

一个典型的编译程序应具有以下的组成：



编译程序的两个主要任务：

一是分析，二是综合。

- 1、词法分析Scanner
- 2、语法分析 Parser
- 3、语义分析Semantic Analyzer
- 4、Intermediate Code Generator
- 5、Target Code Generator
- 6、代码优化
- 7、符号表管理
- 8、错误检测及处理 Error Handler
- 9、前端和后端
- 10、遍

① Scanner

任务：识别单词符号，是最初级的语法分析。

依循的规则：语言的构词规则

例：Do 150 I=1,100 → 基本字：Do
标号：150
标示符：I

.....

② Parser

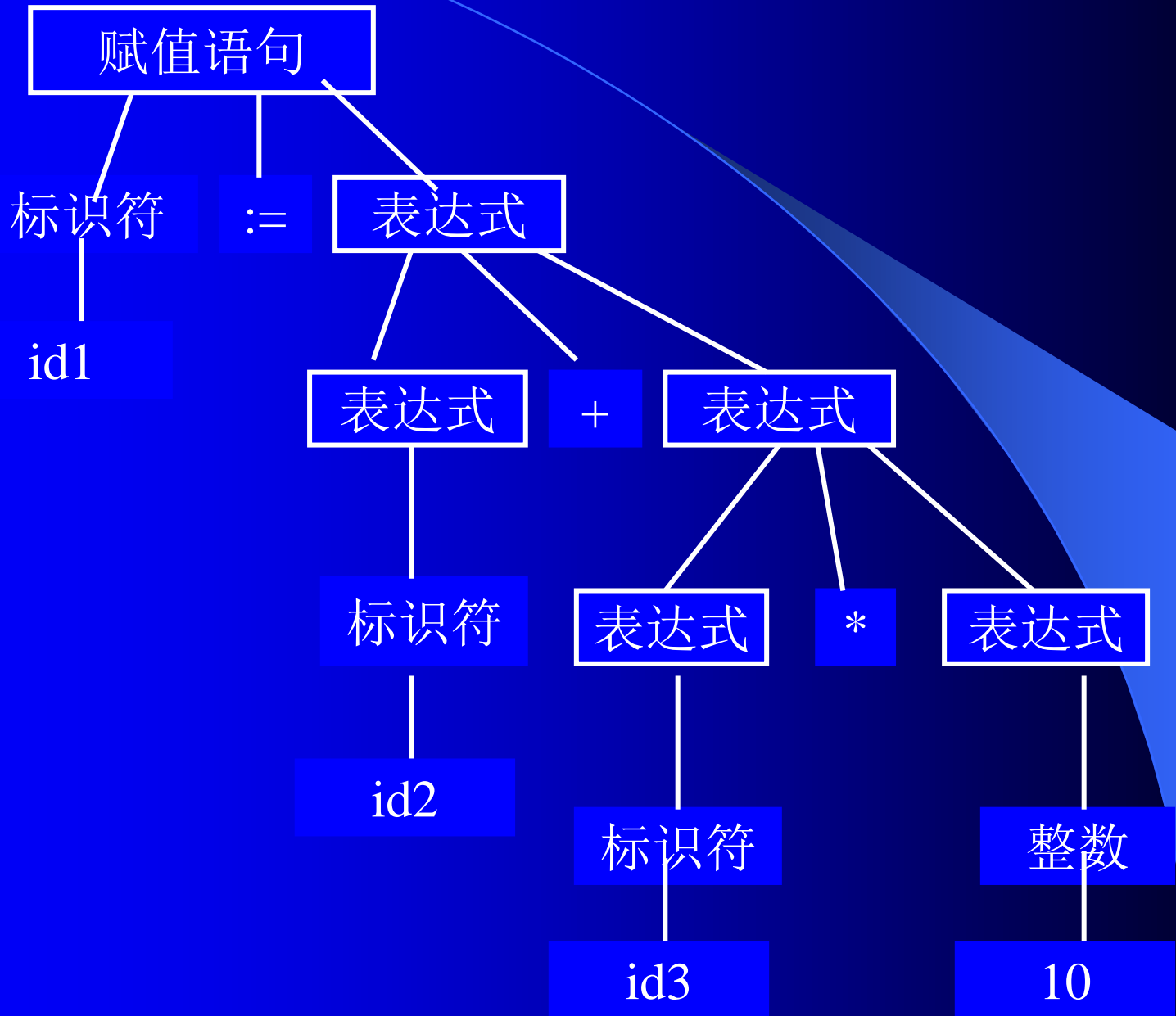
任务：对词法分析的输出即单词符号串进行分析，识别出一个个语法成分，并进行语法检查。

依循的规则：语法规则

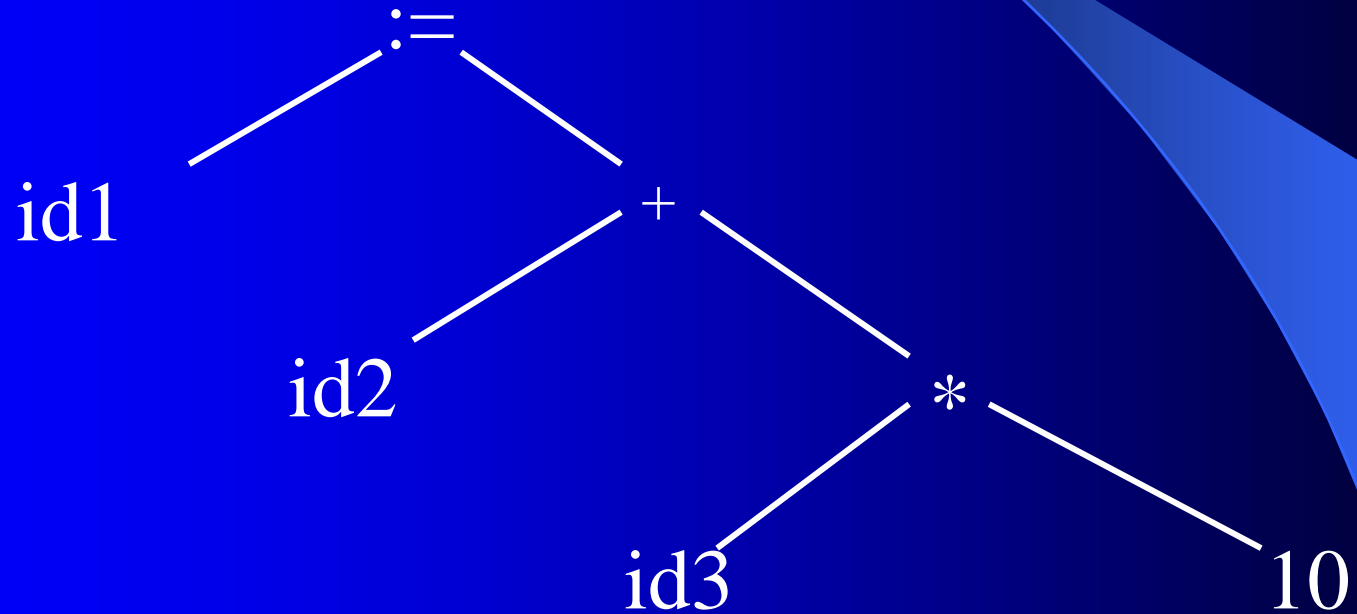
语法单位：算术表达式、短语、句子、子程序.....

例：“ $x+5y$ ”：算术表达式

语句id1:=id2+id3*10的语法树



id1:=id2+id3*10



③ Semantic Analyzer

任务：

对语法分析所识别的各种语法成分的意义(即语义)进行确定并加以处理。

依循的规则：语义规则

例，“+”：加 “*”：乘

③ Semantic Analyzer

语义审查(静态语义)

- 上下文相关性
- 类型匹配
- 类型转换

例: Program p();
 Var rate:real;
 procedure initial;
 ...
 position := initial + rate * 60
/* error */ /* error */ /* warning */;
 ...

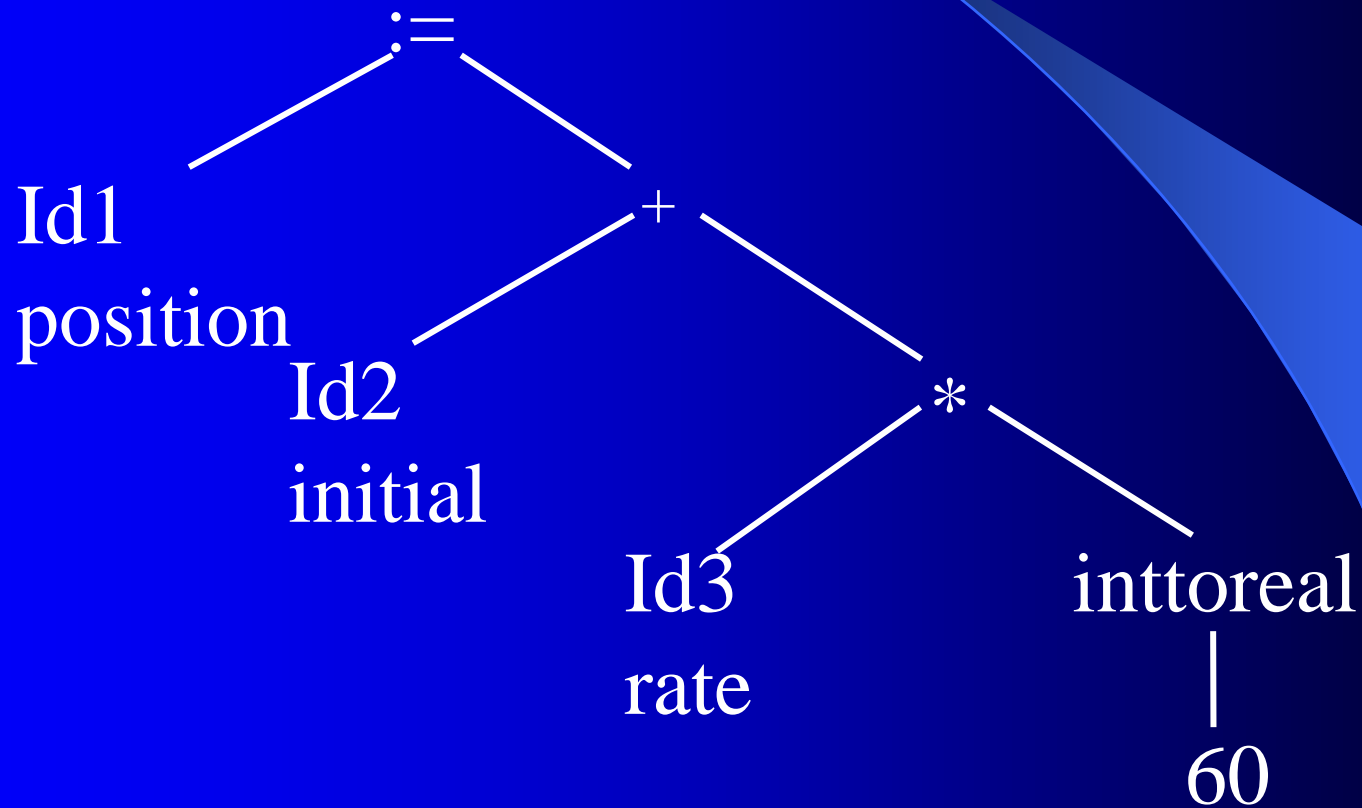
又如:

- `int arr [2],abc;`
- `abc = arr * 10;`

...

- `Program p();`
- `Var rate:real;`
- `Var initial :real;`
- `Var position :real ;`
- ...
- `position := initial + rate * 60`


语义分析



④ Intermediate Code Generator

任务：从源程序的树形或其它形式，
产生源程序的中间代码。

中间代码：四元式、三元式.....

例：a:=b+c 

op	arg1	arg2	result
+	b	c	T1
:=	T1		a

⑤ Target Code Generator

任务：将中间代码转换成汇编程序或者机器语言。

(* , id3 60.0 t1)

(+ , id2 t1 id1)



```
movf id3,R2
mulf #60.0,R2
movf id2,R1
addf R2,R1
movf R1,id1
```

⑥代码优化

id1:= id2 + id3 * 60

(1) (inttoreal 60 - t1)

(2) (* id3 t1 t2)

(3) (+ id2 t2 t3)

(4) (:= t3 - id1)

变换 \Rightarrow

(1) (* id3 60.0 t1)

(2) (+ id2 t1 id1)

代码优化

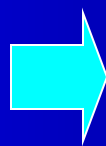
$t1 = b * c$

$t2 = t1 + 0$

$t3 = b * c$

$t4 = t2 + t3$

$a = t4$



$t1 = b * c$

$t2 = t1 + t1$

$a = t2$

7、符号表管理

- 符号表管理是一个贯穿编译全过程的工作。
- 编译程序在分析源程序时，需要记录标识符的各种属性信息；
- 在语义分析和代码生成阶段，还要对建立的符号表进行检索，提取相应的属性信息。--类型、作用域、分配存储信息

Const1 常量 值: 35

Var1 变量 类型: 实 层次: 2

8、Error Handler

- 错误可发生在编译的各个阶段，错误处理也是贯穿编译全过程。
- 词法分析阶段可查出的错误，如标识符的组成不符合词法规则；
- 语句结构错误是在语法分析中可查出的错误；
- 语义分析阶段可查出的错误，即结构正确，但所涉及的操作无意义或错误。
- 在编译时查出的，叫Comple-time error，在运行时表现才表现出来的错误叫Run-time error。

编译程序结构(components)

- 词法分析程序
- 语法分析程序
- 语义分析程序
- 中间代码生成程序
- 代码优化程序
- 目标代码生成程序
- 符号表管理程序
- 出错处理程序

9、前端和后端

前端包括编译逻辑结构中的分析部分，即词法分析、语法分析、语义分析和中间代码生成，除此还包括符号表建造及相应分析中的错误处理以及与机器无关的优化部分。

后端包括与目标机有关的部分，即综合部分，它包括目标代码生成及生成期间对符号表的相应检索操作和错误处理操作，以及与机器相关的代码优化部分。

将编译系统划分为前后端，有利于移植编译系统和利用后端为同一目标机配置不同语言的编译系统。

10、遍(pass)

对源程序(或其中间形式)从头至尾扫描一次并进行有关加工处理，生成新的中间形式或最终目标程序，称为一遍。

分遍原则：

- ①目标质量高低(高则多遍)
- ②机器内存大小(小则多遍)
- ③源语言简繁(繁则多遍)
- ④设计人员多少(多则多遍)

优缺点：

2.4 编译程序的生成

**实现工具: Low-level Language;
High-level Language;
Automatic Builder。**

**生成方法: Self-compiler;
transplant**

参考书

- 吕映芝等，编译原理，清华大学出版社
- 陈意云等，编译原理，高等教育出版社
- 杜淑敏等，编译程序设计原理，北京大学出版社
- Alfred V.Aho等，编译原理，机械工业出版社
- Terrence W.Pratt,Marvin V.Zelkowitz
Programming Languages Design and
Implementation, Prentice-Hall 1996

