# 四、中间代码生成
# (12. 控制流语句的翻译 (Easy 模式))

魏恒峰

hfwei@nju.edu.cn

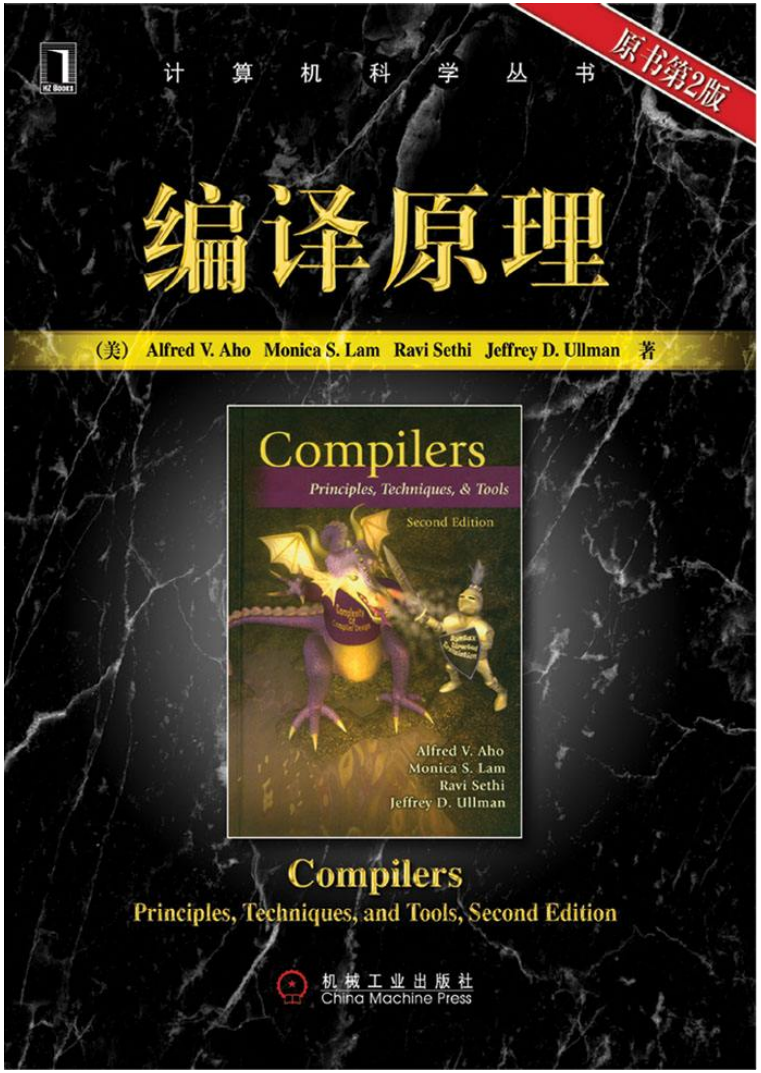
2024 年 05 月 11 日

# Control.g4

| 产生式 |
| --- |
| $P \rightarrow S$ |
| $S \rightarrow$ **assign** |
| $S \rightarrow$ **if** ( $B$ ) $S_1$ |
| $S \rightarrow$ **if** ( $B$ ) $S_1$ **else** $S_2$ |
| $S \rightarrow$ **while** ( $B$ ) $S_1$ |
| $S \rightarrow S_1\ S_2$ |

| 产生式 |
| --- |
| $B \rightarrow B_1$ \|\| $B_2$ |
| $B \rightarrow B_1$ && $B_2$ |
| $B \rightarrow$ ! $B_1$ |
| $B \rightarrow E_1$ **rel** $E_2$ |
| $B \rightarrow$ **true** |
| $B \rightarrow$ **false** |

本讲内容**非常重要**, 但**颇有难度**, 需要多多思考





虽然有**多种**讲法，教材偏偏采用了让初学者**望而生畏**的那一种。

“We will overcome all difficulties.” (我们有困难要上)


“If we don’t have enough difficulties, we’ll create them!”
(没困难, 我们创造困难还要上)

**关键问题：为什么要“创造困难”？**

**生成更短、更高效的代码**

各个非终结符的**分工、合作**明确


只需要使用**综合属性**


心中有“树”(语法分析树)

# 分工　　合作

为布尔表达式 $B$ 计算逻辑值 (假设保存在临时变量 `t1` 中)

**if**、**while** 等语句根据 $B$ 的结果改变控制流

**if** $(B)$ $S_1$


CodeGenVisitor.java

实现顺序: 标识符 $E \to \texttt{id}$, 布尔表达式 $B$、if 语句、while 语句

| 产生式 |
| --- |
| $P \to S$ |
| $S \to$ assign |
| $S \to$ if ( $B$ ) $S_1$ |
| $S \to$ if ( $B$ ) $S_1$ else $S_2$ |
| $S \to$ while ( $B$ ) $S_1$ |
| $S \to S_1\ S_2$ |

| 产生式 |
| --- |
| $B \to B_1$ \|\| $B_2$ |
| $B \to B_1$ && $B_2$ |
| $B \to$ ! $B_1$ |
| $B \to E_1$ rel $E_2$ |
| $B \to$ true |
| $B \to$ false |

# 实现方式: Listeners, Visitors, Attributed Grammar

| 产生式 |
| --- |
| $P \rightarrow S$ |
| $S \rightarrow$ **assign** |
| $S \rightarrow$ **if** ( $B$ ) $S_1$ |
| $S \rightarrow$ **if** ( $B$ ) $S_1$ **else** $S_2$ |
| $S \rightarrow$ **while** ( $B$ ) $S_1$ |
| $S \rightarrow S_1\ S_2$ |

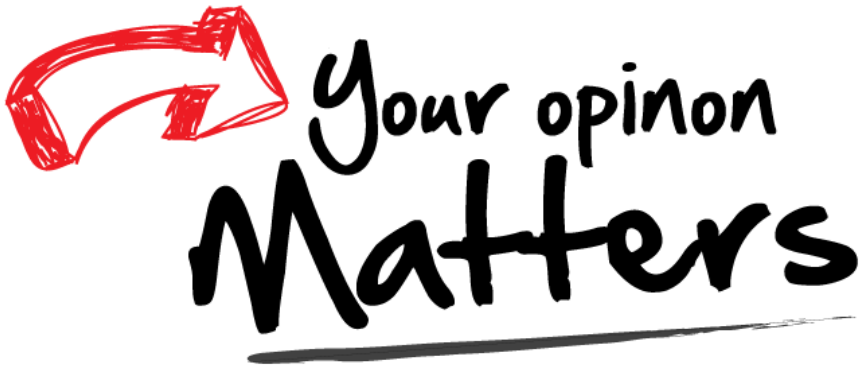
| 产生式 |
| --- |
| $B \rightarrow B_1$ **\|\|** $B_2$ |
| $B \rightarrow B_1$ **&&** $B_2$ |
| $B \rightarrow$ **!** $B_1$ |
| $B \rightarrow E_1$ **rel** $E_2$ |
| $B \rightarrow$ **true** |
| $B \rightarrow$ **false** |

及时输出生成的中间代码, 避免频繁的字符串拼接操作


如何翻译 “break” 语句?


如何实现布尔表达式的“短路求值” (Short-Circuit Evaluation)?


Thank
You!




Office 926

hfwei@nju.edu.cn