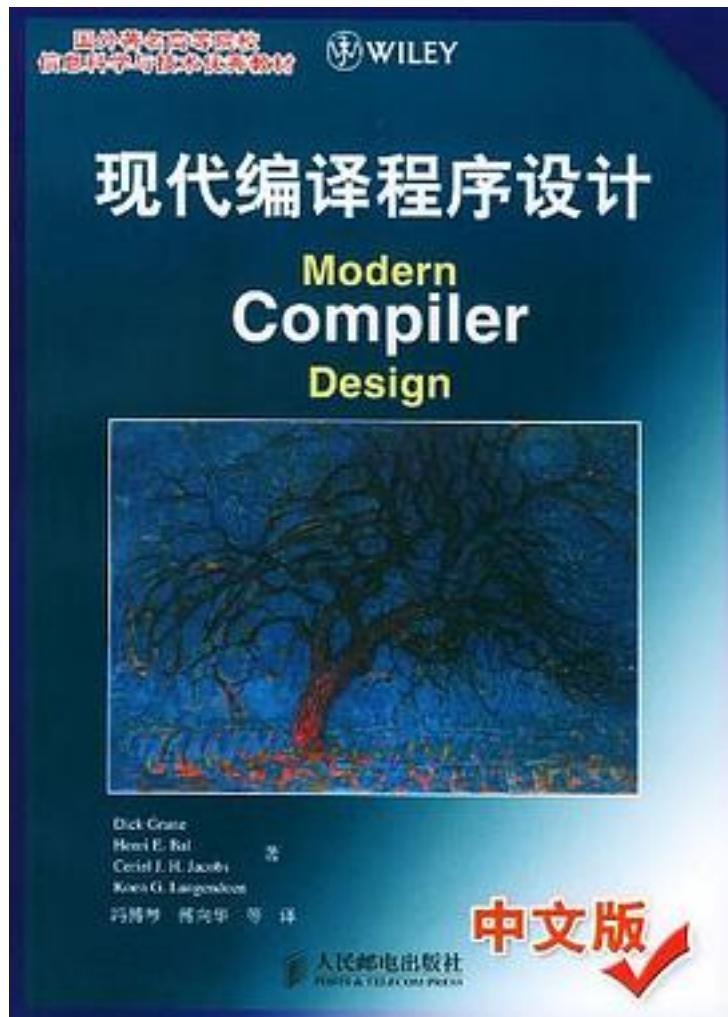


# 现代编译程序设计 Modern Compiler Design 中文版



[现代编译程序设计 Modern Compiler Design 中文版 下载链接1](#)

著者:格伦

出版者:人民邮电

出版时间:2003-9

装帧:平装

isbn:9787115111494

《现代编译程序设计》全面地介绍了现代编译技术，结构上分为通用编译技术和高级编译技术两大部分。第一部分介绍通用的编译程序实现技术，包括词法和语法分析、上下文处理、代码生成以及存储器管理的一般方法。第二部分介绍特定范型语言的高级编译技术，包括命令式语言、面向对象语言、逻辑式语言、函数式语言及并行/分布式语言的上下文处理和代码生成等内容。《现代编译程序设计》注重编译程序的具体实现和优化技术，实例丰富，具有很强的可读性和实用性。

《现代编译程序设计》可作为高校计算机专业本科和研究生编译程序设计课程的教科书，也可供从事计算机软件开发的人员参考。

作者介绍：

## 目录: 第1章 导论 1

1.1 为什么学习编译程序构造 4
1.1.1 编译程序构造是非常成功的 4
1.1.2 编译程序构造的广泛应用 6
1.1.3 编译程序包含普遍适用的算法 6
1.2 一个简单的传统的模块化编译程序/解释程序 6
1.2.1 抽象语法树 7
1.2.2 范例编译程序的结构 8
1.2.3 范例编译程序的语言 9
1.2.4 范例编译程序的词法分析 10
1.2.5 范例编译程序的语法分析 11
1.2.6 范例编译程序的上下文处理 14
1.2.7 范例编译程序的代码生成 14
1.2.8 范例编译程序的解释程序 15
1.3 一个更接近于实际的编译程序的结构 16
1.3.1 结构 17
1.3.2 运行时系统 18
1.3.3 捷径 18
1.4 编译程序体系结构 18
1.4.1 编译程序的宽度 19
1.4.2 谁主控 20
1.5 一个优秀编译程序的特性 22
1.6 可移植性和可重定目标性 23
1.7 优化的位置和效用 23
1.8 编译程序构造简史 24
1.8.1 1945~1960年：代码生成 24
1.8.2 1960~1975年：分析 24
1.8.3 1975年至今：代码生成和代码优化；范型 24
1.9 文法 25
1.9.1 文法形式 25
1.9.2 产生式过程 25
1.9.3 文法的扩展形式 27
1.9.4 文法特性 27
1.9.5 文法形式化方法 28
1.10 闭包算法 29
1.10.1 闭包算法的迭代实现 31
1.11 本书使用的概要代码 33
1.12 小结 33
第2章 从程序文本到抽象语法树 38
2.1 从程序文本到记号——词法结构 41

2.1.1 读程序文本	41
2.1.2 词法分析与语法分析	42
2.1.3 正则表达式和正则描述	43
2.1.4 词法分析	44
2.1.5 手动产生词法分析程序	45
2.1.6 自动产生词法分析程序	50
2.1.7 转换表压缩	63
2.1.8 词法分析程序的错误处理	68
2.1.9 一个传统的词法分析程序产生器——lex	69
2.1.10 记号的词法识别	70
2.1.11 符号表	72
2.1.12 宏处理和文件包含	76
2.1.13 小结	80
2.2 从记号到语法树——语法分析	81
2.2.1 语法分析的两种方法	82
2.2.2 错误检测和错误恢复	84
2.2.3 手工生成一个自顶向下的语法分析程序	86
2.2.4 自动生成一个自顶向下的语法分析程序	88
2.2.5 自动创建一个自底向上的语法分析程序	111
2.3 小结	132
第3章 注释抽象语法树--上下文	142
3.1 属性文法	143
3.1.1 依赖图	146
3.1.2 属性计算	147
3.1.3 循环处理	153
3.1.4 属性分配	158
3.1.5 多次访问属性文法	158
3.1.6 属性文法类型的总结	167
3.1.7 L-属性文法	167
3.1.8 S-属性文法	170
3.1.9 L-属性文法与S-属性文法的等价性	171
3.1.10 扩展的文法符号和属性文法	172
3.1.11 小结	173
3.2 手工方法	173
3.2.1 线性化AST	174
3.2.2 符号解释	178
3.2.3 数据流方程	184
3.2.4 过程间的数据流分析	188
3.2.5 上传信息流——活跃分析	189
3.2.6 符号解释和数据流方程的比较	194
3.3 小结	194
第4章 处理中间代码	202
4.1 解释	203
4.1.1 递归解释	203
4.1.2 迭代解释	207
4.2 代码生成	210
4.2.1 避免完全的代码生成	213
4.2.2 开始点	214
4.2.3 直接代码生成	214
4.2.4 简单代码生成	218
4.2.5 基本块的代码生成	230
4.2.6 BURS代码生成和动态程序设计	241
4.2.7 通过图着色的寄存器分配	255
4.2.8 超级编译	259

4.2.9 代码生成技术的评价 261  
4.2.10 代码优化器的调试 261  
4.2.11 预处理中间代码 262  
4.2.12 后处理目标代码 265  
4.2.13 机器代码生成 267  
4.3 汇编程序、连接程序和装入程序 268  
4.3.1 汇编程序设计问题 270  
4.3.2 连接程序设计问题 272  
4.4 小结 273

## 第5章 存储管理 283

5.1 显式回收的数据空间分配 284  
5.1.1 基本存储空间分配 285  
5.1.2 链表 288  
5.1.3 可扩展数组 290  
5.2 隐式回收的数据空间分配 291  
5.2.1 基本垃圾收集算法 291  
5.2.2 背景预备 292  
5.2.3 引用计数 297  
5.2.4 标记和扫描 300  
5.2.5 两空间复制 303  
5.2.6 紧缩 306  
5.2.7 世代垃圾收集 307  
5.3 小结 307

## 第6章 命令式和面向对象程序 313

6.1 上下文处理 314  
6.1.1 识别 315  
6.1.2 类型检查 321  
6.1.3 小结 328  
6.2 源语言数据表示和处理 328  
6.2.1 基本类型 329  
6.2.2 枚举类型 329  
6.2.3 指针类型 329  
6.2.4 记录类型 332  
6.2.5 共用体类型 333  
6.2.6 数组类型 334  
6.2.7 集合类型 336  
6.2.8 例程类型 336  
6.2.9 对象类型 337  
6.2.10 接口类型 344  
6.3 例程及其活动 345  
6.3.1 活动记录 345  
6.3.2 例程 347  
6.3.3 例程上的操作 348  
6.3.4 非嵌套例程 350  
6.3.5 嵌套例程 352  
6.3.6 Lambda提升 357  
6.3.7 迭代器和协作例程 358  
6.4 控制流语句的代码生成 359  
6.4.1 局部控制流 359  
6.4.2 例程调用 366  
6.4.3 运行时错误处理 372  
6.5 模块的代码生成 374  
6.5.1 名字生成 375  
6.5.2 模块初始化 375

6.5.3 泛型的代码生成	376
6.6 小结	377
第7章 函数式程序	386
7.1 Haskell简介	387
7.1.1 越位规则	387
7.1.2 列表	388
7.1.3 列表内涵	388
7.1.4 模式匹配	389
7.1.5 多态类型	390
7.1.6 引用透明性	391
7.1.7 高阶函数	391
7.1.8 惰性计算	392
7.2 编译函数式语言	393
7.2.1 函数核	394
7.3 多态类型检查	395
7.3.1 多态函数应用	396
7.4 脱糖	397
7.4.1 列表的翻译	397
7.4.2 模式匹配的翻译	397
7.4.3 列表内涵的翻译	399
7.4.4 嵌套函数的翻译	401
7.5 图归约	402
7.5.1 归约顺序	405
7.5.2 归约引擎	406
7.6 函数核程序的代码生成	409
7.6.1 避免一些应用框架的构造	411
7.7 优化函数核	412
7.7.1 严格性分析	413
7.7.2 装箱分析	417
7.7.3 尾部调用	417
7.7.4 累加器转换	419
7.7.5 局限性	420
7.8 高级图处理	421
7.8.1 可变长度结点	421
7.8.2 指针标记	421
7.8.3 聚集结点分配	421
7.8.4 向量应用结点	422
7.9 小结	422
第8章 逻辑式程序	427
8.1 逻辑式程序设计模型	428
8.1.1 构建模块	428
8.1.2 推理机制	430
8.2 解释的通用实现模型	431
8.2.1 解释程序指令	432
8.2.2 避免冗余目标列表	434
8.2.3 避免复制目标列表尾部	434
8.3 合一	435
8.3.1 结构、列表和集合的合一	435
8.3.2 合一的实现	437
8.3.3 两个自由变量的合一	440
8.3.4 小结	441
8.4 编译的通用实现模型	441
8.4.1 列表程序	442
8.4.2 编译子句的搜索和合一	444

8.4.3 WAM中的优化子句选择	448
8.4.4 应用“cut”机制	450
8.4.5 谓词assert和retract的实现	452
8.5 合一的编译代码	455
8.5.1 WAM中的合一指令	456
8.5.2 通过手工局部计算得到合一指令	457
8.5.3 WAM中的结构合一	462
8.5.4 一种优化：读/写模式	464
8.5.5 WAM中合一结构的进一步优化	466
8.5.6 小结	467
第9章 并行和分布式程序	472
9.1 并行程序设计模型	474
9.1.1 共享变量和管程	474
9.1.2 消息传递模型	476
9.1.3 面向对象语言	477
9.1.4 Linda元组空间	477
9.1.5 数据并行语言	478
9.2 进程和线程	479
9.3 共享变量	481
9.3.1 锁	481
9.3.2 管程	481
9.4 消息传递	482
9.4.1 接收方定位	483
9.4.2 编组	483
9.4.3 消息的类型检查	484
9.4.4 消息选择	484
9.5 并行的面向对象语言	485
9.5.1 对象定位	485
9.5.2 对象迁移	486
9.5.3 对象复制	487
9.6 元组空间	488
9.6.1 避免关联寻址的开销	488
9.6.2 元组空间的分布实现	490
9.7 自动并行	492
9.7.1 自动地使用并行性	492
9.7.2 数据依赖	494
9.7.3 循环转换	495
9.7.4 分布式存储器的自动并行	496
9.8 小结	498
附录A 一个简单的面向对象编译程序/解释程序	502
附录B 练习答案	509
附录C 参考文献	519
附录D 术语表	527
· · · · · (收起)	

[现代编译程序设计 Modern Compiler Design 中文版 下载链接1](#)

标签

编译原理

编译

compiler

Compiler

计算机科学

广州时期

已入柜

@myLibrary

## 评论

感觉好一般阿，不过有比较好的语言例子

---

好书，翻译略坑。第二版也出了，目测引进遥遥无期，有精力读第二版吧。

---

[现代编译程序设计 Modern Compiler Design 中文版 下载链接1](#)

## 书评

---

[现代编译程序设计 Modern Compiler Design 中文版 下载链接1](#)