



高等学校计算机科学与技术教材

算法分析与设计 (C++描述)

□ 石志国 刘冀伟 姚亦飞 编著



- 原理与技术的完美结合
- 教学与科研的最新成果
- 语言精炼，实例丰富
- 可操作性强，实用性突出

清华大学出版社

● 北京交通大学出版社

高等学校计算机科学与技术教材

算法分析与设计 (C++描述)

石志国 刘冀伟 姚亦飞 编著

清华大学出版社
北京交通大学出版社

· 北京 ·

内 容 简 介

本书以程序设计作为基础、数据结构作为工具、五大核心算法作为目标,系统地介绍了算法设计中典型问题的求解过程。

全书分成程序设计基础、数据结构和五大核心算法3个部分共10章。第1部分为算法分析与程序设计基础,介绍了算法分析的时间和空间复杂度,以及C++算法相关的程序设计基础;第2部分为算法设计数据结构基础,介绍了线性和非线性数据结构基础,同时对常用的排序和搜索算法作了详细介绍;第3部分为典型算法分析与问题求解,介绍了经典算法设计中的“五虎上将”:分治法、贪心法、动态规划、回溯法和分支限界法。

本书可以作为高校及各类培训机构相关课程的教材或参考书,提供全书源代码、软件和授课幻灯片等资料,可以从图书支持网站<http://www.gettop.net>下载,也可以从出版社网站<http://press.bjtu.edu.cn>的下载栏目中下载。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

算法分析与设计(C++描述)/石志国,刘冀伟,姚亦飞编著. —北京:清华大学出版社;北京交通大学出版社,2010.9

(高等学校计算机科学与技术教材)

ISBN 978-7-5121-0223-1

I. ①算… II. ①石… ②刘… ③姚… III. ①电子计算机-算法分析-高等学校-教材 ②电子计算机-算法设计-高等学校-教材 IV. ①TP301.6

中国版本图书馆CIP数据核字(2010)第151521号

责任编辑:谭文芳 特邀编辑:方元元

出版发行:清华大学出版社 邮编:100084 电话:010-62776969 <http://www.tup.com.cn>

北京交通大学出版社 邮编:100044 电话:010-51686414 <http://press.bjtu.edu.cn>

印刷者:北京瑞达方舟印务有限公司

经 销:全国新华书店

开 本:185×260 印张:15.5 字数:394千字

版 次:2010年9月第1版 2010年9月第1次印刷

书 号:ISBN 978-7-5121-0223-1/TP·607

印 数:1~4000册 定价:26.00元

本书如有质量问题,请向北京交通大学出版社质监组反映。对您的意见和批评,我们表示欢迎和感谢。

投诉电话:010-51686043, 51686008; 传真:010-62225406; E-mail: press@bjtu.edu.cn。

前 言

算法分析与设计是一门重要课程。算法设计在各类科学计算中具有核心地位和作用，没有好的算法，计算机完成一件工作可能需要很长时间，而有好算法，完成一件工作可能仅需要几秒。同时，算法被公认为是计算科学的基石，翻开重要的学术刊物，算法都占有一席之地，没有算法，程序将不复存在。

1. 体系介绍

算法是一门方法论方面的课程，方法论是人们认识世界、改造世界的一般方法，是人们用什么样的方式、方法来观察事物和处理问题的一般依据。在哲学上，世界观主要解决世界“是什么”的问题，方法论主要解决“怎么办”的问题。在科学计算方面，算法也是解决“怎么办”的问题。作为一门方法论的课程，主要讲解一些普适的、抽象的解决问题的方法。从认知角度来讲，对于抽象内容需要一些具体实际的例子来帮助消化吸收，否则知识将变得非常空泛。

目前，算法分析与设计方面的图书从来源上可以分成两类：国内图书和国外引进。从内容上也可以分成两类：单纯讲解算法和同时讲解数据结构与算法，总体内容上几乎没有太大的区别。从算法描述上，主要以采用 C/C++ 和 Java 语言描述居多，以程序实现算法框架为主，完整的算法实现的实例相对较少，这使得算法课程成为一门理论性较强的课程。

编者对国内众多专业课程进行了调研，考虑到读者前期一般具有 C 或者 C++ 语言基础，Java 语言大多是高年级以后以选修课程的形式开设，而算法课程开设的时间一般早于 Java 语言，因此选择 C/C++ 语言描述。

编写算法对语言要求相对较高，涉及面的也广，因此安排一章对 C++ 算法设计进行介绍。并在此基础上介绍了线性、非线性数据结构和排序搜索算法设计，最后详细介绍了目前算法中被称为“五虎上将”的分治法、贪心法、动态规划、回溯法和分支限界法，同时对于每一种算法都列举了几个完整的实例。

2. 内容划分

作为一本基础性的图书，全书以简短的章节解决典型的算法问题，涵盖了绝大多数算法设计中的常用技术，回避了一些生涩而且不常使用的算法。在表达每一种算法时，先对问题进行分析，给出解题的一般流程，然后提供了相应解决实际问题的完整程序，并用图来直观表示其结果。同时也注重对算法的复杂性分析。

全书从程序设计基础、数据结构基础和五大算法三个方面分为三个部分共 10 章，每章都附有大量的习题，有利于读者对书中内容的理解和应用。

3. 致谢与后期支持

本书在编写过程中，得到众多老师的指导和帮助。感谢中科院软件所贺也平研究员，吴永军工程师；清华大学计算机系林闯教授、尹浩副教授；北京科技大学胡广大教授、王志良教授、徐正光教授、李晓理教授和解仑教授；中央广播电视大学崔林教授、徐孝凯教授、田萧老师和王春风老师；中国软件行业协会邱钦伦高级工程师；IBM 公司的金洁华经理；用友

软件公司的刘达总监。感谢他们为本书提供了大量且详尽的编程资料，并为本书解决了很多编程方面的问题。尤其要感谢北京交通大学出版社的谭文芳老师，她的支持是本书能够顺利出版的关键。感谢众多老师和同学的支持，他们的每一个问题，都是本书要强调并解决的知识点，他们的认可是我最大的动力。

同时相关内容的编写得到国家自然科学基金（60903067）和校级教学研究项目基金资助，在此表示感谢！

本书可以作为高校及各类培训机构相关课程的教材或参考书，也可供程序设计人员参考供用。本书完整的教学幻灯片、书中的所有软件和源代码及相关学习资源，将在 <http://www.gettop.net> 或者 <http://press.bjtu.edu.cn> 下载栏目中发布，欢迎访问和下载。由于作者水平和时间有限，难免出现错误，对于本书的任何建议与意见请发邮件至作者邮箱：shizhiguo@tom.com。

编 者

2010年6月

目 录

第 1 部分 算法分析与程序设计基础

第 1 章 算法的基本概念	3
1.1 算法的基本概念	3
1.1.1 算法的特征	3
1.1.2 算法的 4 个标准	4
1.1.3 算法的描述形式	5
1.2 算法复杂性分析框架	6
1.2.1 增长次数	6
1.2.2 渐进符号	6
1.2.3 时间复杂度	7
1.2.4 空间复杂度	9
本章小结	10
课后习题	10
第 2 章 C++ 算法程序设计基础	12
2.1 C++ 语言概述	12
2.1.1 C++ 语言的优势	12
2.1.2 C++ 语言的内容	13
2.1.3 编程工具	13
2.2 C++ 程序结构初步	17
2.2.1 预处理指示符初步	17
2.2.2 注释	17
2.2.3 基本输入/输出	17
2.3 使用 C++ 语言编写简单代码	17
2.3.1 面向过程的 C 语言	18
2.3.2 面向过程的 C++ 语言	18
2.3.3 面向对象的 C++ 语言	19
2.4 C++ 面向对象基础	23
2.4.1 数据成员	24
2.4.2 成员函数	24
2.4.3 类对象成员的访问	26
2.4.4 类的访问限制	27

2.4.5 动态内存分配	28
2.4.6 C++ 程序内存分配	30
2.5 构造函数和析构函数	30
2.5.1 构造函数的概念	30
2.5.2 析构函数的概念	31
2.5.3 带参数的构造函数	32
2.5.4 重载构造函数	33
2.6 类中的 this 指针	34
2.7 类中的 const 修饰符	35
2.7.1 常对象	35
2.7.2 常成员函数	36
2.7.3 常数据成员	37
2.8 模板的基本概念	37
2.8.1 使用模板的必要性	37
2.8.2 模板的分类	38
2.9 函数模板	38
2.9.1 函数模板的定义	38
2.9.2 使用函数模板	38
2.9.3 函数模板的重载	40
2.10 类模板	41
2.10.1 类模板的定义	41
2.10.2 使用类模板	41
2.11 继承的基本概念	43
2.11.1 继承的必要性	43
2.11.2 继承的实现方式	45
2.11.3 继承中的静态数据成员	47
2.12 基类和派生类的关系	48
2.12.1 基类指针	48
2.12.2 继承下的构造函数和析构函数	49
2.12.3 重写基类成员	50
2.12.4 调用基类成员函数	51
2.13 详解 protected 关键字	53
2.14 保护继承与私有继承	54
2.14.1 公有继承	54
2.14.2 私有继承	55
2.14.3 保护继承	56
本章小结	56
课后习题	56

第2部分 算法设计数据结构基础

第3章 线性数据结构基础	61
3.1 抽象数据类型	61
3.2 线性表基础	61
3.2.1 线性表定义及特点	62
3.2.2 顺序表	62
3.2.3 链表	72
3.2.4 数组与链表性能比较	79
3.3 栈与队列基础	80
3.3.1 栈	80
3.3.2 队列	83
本章小结	91
课后习题	91
第4章 非线性数据结构基础	93
4.1 树与二叉树	93
4.1.1 树的基本概念	93
4.1.2 二叉树	94
4.2 树与二叉树的存储结构	95
4.2.1 二叉树的存储结构	95
4.2.2 树的存储结构	96
4.2.3 二叉树的遍历	97
4.3 图	104
4.3.1 图的基本概念	104
4.3.2 图的存储结构	105
本章小结	113
课后习题	113
第5章 排序与搜索算法基础	114
5.1 排序算法的基本概念	114
5.1.1 排序的分类	114
5.1.2 排序算法的评价标准	114
5.2 简单排序算法	114
5.2.1 插入排序	115
5.2.2 选择排序	117
5.2.3 冒泡排序	119
5.3 快速排序	121
5.4 堆排序	124

5.5 归并排序	127
5.6 希尔排序	130
5.7 线性表查找	133
5.7.1 顺序查找	134
5.7.2 二分查找	134
5.8 树与图的搜索	135
5.8.1 二叉排序树搜索	135
5.8.2 B-树	136
5.8.3 广度优先搜索	138
5.8.4 图的深度优先搜索	140
本章小结	141
课后习题	141

第3部分 典型算法分析与问题求解

第6章 递归与分治法算法设计	147
6.1 递归法	147
6.1.1 递归算法的特性	148
6.1.2 递归的执行过程	148
6.2 递归法应用举例	148
6.2.1 汉诺塔问题求解	148
6.2.2 斐波那契数列问题求解	150
6.2.3 八皇后问题	151
6.3 分治法	154
6.3.1 问题提出	154
6.3.2 分治法概述	156
6.4 分治法应用举例	157
本章小结	158
课后习题	159
第7章 贪心算法设计	161
7.1 贪心法	161
7.1.1 问题提出	161
7.1.2 贪心法的基本思路	162
7.2 贪心法应用举例	163
7.2.1 背包问题	163
7.2.2 哈夫曼编码	167
7.2.3 单源最短路径	173
7.2.4 最小生成树	177
本章小结	184

课后习题	185
第8章 动态规划算法设计	187
8.1 动态规划法	187
8.1.1 动态规划法的基本概念	187
8.1.2 多阶段决策	188
8.1.3 动态规划法适用条件	189
8.1.4 动态规划法解决问题的步骤	189
8.2 动态规划法应用举例	190
8.2.1 多源最短路径	190
8.2.2 最大公共子序列问题	194
8.2.3 导弹拦截问题	197
本章小结	200
课后习题	200
第9章 回溯法算法设计	202
9.1 回溯法	202
9.1.1 回溯法的基本概念	202
9.1.2 回溯法的基本思想	203
9.1.3 回溯法求解问题的步骤	204
9.2 回溯法应用举例	205
9.2.1 小老鼠走迷宫问题	205
9.2.2 子集合问题	209
9.2.3 全排列问题	210
9.2.4 八皇后问题	211
9.2.5 0-1 背包问题	213
本章小结	216
课后习题	216
第10章 分支限界算法设计	218
10.1 分支限界	218
10.1.1 分支限界法的基本思想	218
10.1.2 求解问题的适用条件和步骤	219
10.1.3 分支限界的优缺点	219
10.2 分支限界应用举例	219
10.2.1 0-1 背包问题	219
10.2.2 旅行售货员问题	226
本章小结	230
课后习题	230
附录 A 部分习题参考答案	232
参考文献	235



第 1 部分

算法分析与程序设计基础

本部分包括两章：

☑ 第 1 章 算法的基本概念

☑ 第 2 章 C++ 算法程序设计基础

所谓教育，是忘却了在校学的全部内容之后剩下的本领。

——阿尔伯特·爱因斯坦（Albert Einstein）

对一切来说，只有热爱才是最好的老师，它远远胜过责任感。

——阿尔伯特·爱因斯坦（Albert Einstein）

不是所有能计算的都有价值，不是所有有价值的都能被计算。

——阿尔伯特·爱因斯坦（Albert Einstein）

第 1 章 算法的基本概念

本章要点

- 算法的基本概念与描述方法
- 算法复杂度分析框架

1.1 算法的基本概念

算法 (Algorithm) 的概念在计算机科学领域中几乎无处不在, 在各种计算机系统的实现中, 算法的设计往往处于核心的位置。不过, 计算机算法的研究受到人们如此重视是在 20 世纪 70 年代以后。确切地说是美国斯坦福大学著名计算机科学家 D.E.Knuth 的《The act of computer programming》和 A.V.Aho 的《The design and analysis of computer algorithms》等著作对算法的研究起到了奠基的作用, D.E.Knuth 因这一著作, 获得了计算机领域的最高奖——图灵奖。

1.1.1 算法的特征

计算机的问世是 20 世纪人类最伟大的发明之一, 它把人类社会带进了信息技术时代, 而算法是计算机科学的重要基础, 就像算盘一样, 人们需要为计算机编制各种各样的“口诀”即算法, 才能使其工作。

虽然每天都在和算法打交道, 但是能严格地指出什么是算法却不是一件容易的事。著名的 Webster 词典在“algorithm”词条下指出: “算法即在有限步骤内解一个数学问题的过程, 步骤中常常包括某一操作的重复”。更广义地说, 一个算法就是解一个问题或实现某一目标的逐步过程。这个定义并未与计算机相关, 事实上, 我国的数学著作《九章算术》就是采用问题集的形式编的, 该书共有 246 个问题的求解算法, 远在计算机出现之前就已提出。

D.E.Knuth 给出了另一个说明: 一个算法, 就是一个有穷规则的集合, 规定了一个解决某一特定类型问题的运算序列, 此外还应具有如下 5 个重要特性。

1. 输入性

一个算法要具有 0 个或多个外部量作为算法的输入, 这些外部量通常体现为算法中的一组变量, 有些输入量需要在算法执行过程中输入。从表面上看, 有些算法好像没有输入量, 实际上是输入量已被嵌入算法之中。

2. 输出性

一个算法必须具有一个或多个输出, 以反映算法对输入数据加工后的结果, 没有输出的算法是毫无意义的。

3. 确定性

算法的每一个步骤必须具有确定的定义, 即每一步要执行的动作是确定的, 是无二义性

的。在任何条件下, 算法只有唯一的一条执行路径, 即对于相同的输入得出的输出结果也是相同的。

4. 有穷性

对于任何合法的输入值, 算法必须在执行有限个步骤之后结束, 并且每一步都可以在有限的时间内完成。

5. 可行性

算法中描述的操作都可以通过已经实现的基本运算的有限次执行来实现, 即算法的具体实现应该能够被计算机执行。

1.1.2 算法的 4 个标准

就像生物学家把自然界的所有生物作为自己的研究对象一样, 计算学科则把问题作为自己的研究对象, 用计算机来解决问题。在计算机专家的头脑中, 世界是一个个要解决的问题集合。假如如下分段函数, 要求对用户输入的一个自变量 x 的值, 给出相应的函数值。

$$f(x) = \begin{cases} x+8, & x > 0 \\ 8, & x = 0 \\ x-8, & x < 0 \end{cases}$$

实现分段函数求值功能的过程可以作以下描述。

① 输入自变量 x 的值。

② 依据自变量 x 的值进行判断:

如果 $x > 0$, 执行 $x+8 \Rightarrow f(x)$ 操作;

如果 $x < 0$, 执行 $x-8 \Rightarrow f(x)$ 操作;

如果 $x = 0$, 执行 $8 \Rightarrow f(x)$ 操作。

其中, $A \Rightarrow B$ 表示将表达式 A 的值赋给变量 B 。

③ 输出步骤②计算的结果。

显然, 上述描述满足算法的 5 个重要特征, 因此, 该描述就是一个算法。在现实社会中, 不同的人对于同一问题会有不同的看法或解决方法。同样, 在计算机领域, 对于同一问题可能存在多种算法也是很自然的事情。例如, 对于一批数据的排序问题, 就存在多种排序方法。判断一个算法的好坏主要依据以下 4 个标准。

1. 正确性

正确性是设计一个算法的首要条件, 如果一个算法不正确, 其他方面就无从谈起。一个正确的算法是指在合理的数据输入下, 能在有限的时间内得出正确的结果。

2. 可读性

算法主要是为了人的阅读与交流, 其次才是让计算机执行, 因此算法应该易于人的理解; 另外, 晦涩难读的算法易于隐藏较多错误而使实现该算法的程序的调试工作变得更加困难。

3. 健壮性

算法应当具备检查错误和对错误进行适当处理的能力。一般而言, 处理错误的方法不应是中断程序的执行, 而应是返回一个表示错误或错误性质的值, 以便在更高的抽象层次上进

行处理。

4. 效率

效率是指算法执行时所需计算机资源的多少,包括运行时间和存储空间两方面的要求。运行时间和存储空间都与问题的规模有关。存储空间指的是算法执行过程中所需的最大存储空间。

在设计一个算法时,要从上述4个方面综合考虑。同时还要考虑到算法的使用频率及所使用机器的软硬件环境等因素,这样才能设计出一个好的算法。

1.1.3 算法的描述形式

算法的描述形式多种多样,不同的算法描述形式对算法的质量有一定的影响。描述同一个算法可以采用自然语言、流程图、盒图、伪代码、程序设计语言等,常用的描述算法方法有如下4种。

1. 自然语言描述法

最简单的描述算法的方法是使用自然语言,用自然语言来描述算法的优点是简单且便于人们对算法的理解和阅读,缺点是不够严谨,易产生歧义。当算法比较复杂且包含很多转移分支时,用自然语言描述就不是那么直观清晰了。

2. 算法框图法

使用程序流程图、盒图等算法描述工具来描述算法。其特点是简洁、明了,便于理解和交流。

3. 伪码语言描述法

用上述两种方法描述的算法并不能够直接在计算机上执行。为了解决理解与执行之间的矛盾,人们常常使用一种称为伪码语言的描述方法来对算法进行描述。伪码语言介于高级程序设计语言和自然语言之间,它忽略高级程序设计语言中一些严格的语法规则与描述细节,因此它比程序设计语言更容易描述和被人理解,而比自然语言或算法框图更接近程序设计语言。

4. 高级程序设计语言描述法

使用特定的可以直接在计算机上执行的程序描述算法。优点是不用转换直接可以编译执行,缺点是需要对特定的程序设计语言比较理解。

大部分的算法最终是需要通过能够向计算机发送一系列命令的程序来实现的。所谓“程序”是指对所要解决问题的各个对象和处理规则的描述,或者说是数据结构和算法的描述,因此有人说“数据结构+算法=程序”。

程序与算法不同。程序可以不满足算法的第4个特性。例如,操作系统,它是在无限循环中执行的程序,因而不是算法。然而可以把操作系统的各种任务看作一些单独的问题,每一个问题由操作系统中的一个子程序通过特定的算法实现,该子程序得到输出结果后便终止。

算法设计方法主要有分治策略、动态规划、贪心算法、回溯法、分支限界、概率算法等,这些将在后面的章节中陆续介绍,并采用C++语言来描述算法。C++语言的优点是类型丰富、语句精练,具有面向对象和面向过程的双重优点,用C++来描述算法可使整个算法结构紧凑、可读性强。

1.2 算法复杂性分析框架

算法复杂性是算法运行所需要的计算机资源的量，需要时间资源的量称为时间复杂性，需要空间资源的量称为空间复杂性。算法复杂性的度量主要是针对运行该算法所需要的计算机资源的多少。当算法所需要的资源越多，该算法的复杂性越高；反之，当算法所需要的资源越少，算法的复杂性越低。

算法的时间效率和空间效率都用输入规模的函数进行度量。对于所有的算法，规模更大的输入都需要运行更长的时间。经常使用一个输入规模参数为 n 的函数来研究算法的效率。选择输入规模的合适量度，要受到所讨论算法的操作细节影响。

1.2.1 增长次数

设 5 种复杂度函数分别为 n , n^2 , n^5 , 2^n , 3^n ，采用速度为每秒钟处理 1 000 000 次基本操作的计算机，其处理问题长度分别为 $n=10$, $n=30$, $n=60$ 的时间代价需求如表 1-1 所示。

表 1-1 时间代价需求

$T(n)$	$n=10$	$n=30$	$n=60$
n	0.01 毫秒	0.03 毫秒	0.06 毫秒
n^2	0.1 毫秒	0.9 毫秒	3.6 毫秒
n^5	0.1 秒	24.3 秒	13.0 分
2^n	1.0 毫秒	17.9 分	366.0 世纪
3^n	0.06 秒	6.5 年	1.3×10^{13} 世纪

一个需要指数级操作次数的算法只能用来解决规模非常小的问题，否则无法在适当的时间范围内完成。

1.2.2 渐进符号

算法的渐进时间定义为一个函数，定义域为自然数集合 N 。但有时也将其扩展到实数或限制到自然数的某子集上。算法效率的主要指标是基本操作次数的增长次数。为了对这些增长次数进行比较和归类，主要的渐进符号有 3 种。

- ⊆ O : 上界。
- ⊆ Ω : 下界。
- ⊆ Θ : 上界和下界。

定义 1（符号 O ）对于足够大的 n , $t(n)$ 的上界由 $g(n)$ 的常数倍来确定，即记为 $t(n) \in O(g(n))$, $t(n) \leq cg(n)$, c 为常数。

例如：

$$\begin{aligned} n &\in O(n^2) \\ 100n+5 &\in O(n^2) \\ n(n-1)/2 &\in O(n^2) \end{aligned}$$

上界关系如图 1-1 所示。

定义 2 (符号 Ω) 对于足够大的 n , $t(n)$ 的下界由 $g(n)$ 的常数倍来确定, 即记为 $t(n) \in \Omega(g(n))$, $t(n) \geq cg(n)$, c 为常数。

例如:

$$n^3 \in \Omega(n^2)$$

$$n(n+1) \in \Omega(n^2)$$

$$4n^2+5 \in \Omega(n^2)$$

下界关系如图 1-2 所示。

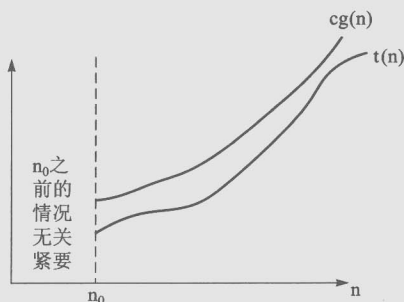


图 1-1 上界关系

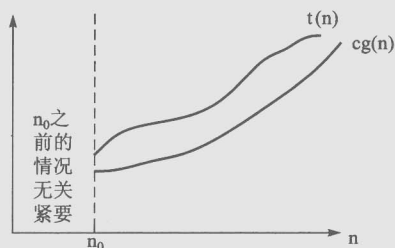


图 1-2 下界关系

定义 3 (符号 Θ) 对于足够大的 n , $t(n)$ 的上界和下界由 $g(n)$ 的常数倍来确定, 即记为 $t(n) \in \Theta(g(n))$, $c_2g(n) \leq t(n) \leq c_1g(n)$, c_1, c_2 为常数。

例如:

$$n^2+3n+2 \in \Theta(n^2)$$

$$n(n-1)/2 \in \Theta(n^2)$$

$$4n^2+5 \in \Theta(n^2)$$

上界与下界关系如图 1-3 所示。

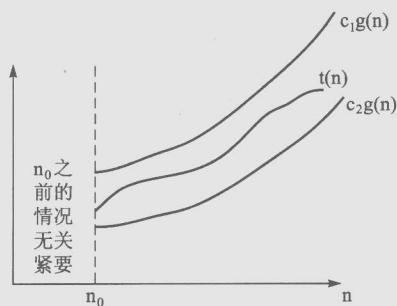


图 1-3 上界与下界的关系

1.2.3 时间复杂度

通常, 对于一个算法的复杂性分析主要是对算法效率的分析, 包括衡量其运行速度的时间效率, 以及其运行时所需要占用的空间大小。对于算法的时间效率的计算, 通常是抛开与计算机硬件、软件有关的因素, 仅考虑实现该算法的高级语言程序。

一般而言, 对程序执行的时间复杂度的分析是以分块进行的, 先分析程序中的语句, 再分析各程序段, 最后分析整个程序的执行复杂度, 通常以渐进式的 O 形式来表示算法的时间复杂度。

渐进式的 O 形式表示时间复杂度的主要运算规则有如下两种。

(1) 求和规则

$O(f(n)) + O(g(n)) = O(\max(f(n), g(n)))$, 其中, $f(n)$ 和 $g(n)$ 表示与 n 有关的一个函数。

(2) 乘法规则

$O(f(n)) * O(g(n)) = O(f(n) * g(n))$, $O(c * f(n)) = O(f(n))$, c 是一个正常数。