算法分析与设计
微课视频版

李恒武　编著

清华大学出版社
北 京

内 容 简 介

本书是中国大学 MOOC、智慧树和学银在线一流课程配套教材，也是工科联盟和一流专业课程配套教材。

本书以问题求解为主线，全面介绍问题求解的方法与优化技巧，分为算法与问题、算法分析、算法设计、问题复杂性与求解、图算法 6 部分。算法与问题着重介绍问题求解过程和问题变换；算法分析主要介绍算法复杂度、复杂度分析与比较方法、时空均衡；算法设计主要介绍枚举算法、贪心算法、递推算法、分治算法、动态规划算法、回溯算法、分支限界、网络流算法策略与优化方法；问题复杂性与求解主要介绍问题复杂性分类、NP 完全问题证明与求解策略、随机算法、近似算法等；图算法介绍和总结图的可图性、连通性、可行遍性和平面图问题。

本书提供了大量热点问题、应用实例和常用算法，每章均附有 POJ 配套编程实践题、思考题和习题。全书配套微课视频、PPT、知识梳理、章节测验、实践作业、在线题库和文档资源。

本书适合作为高等院校计算机科学与技术、软件工程、人工智能、信息安全、信息与计算、金融信息化、金融大数据、数字媒体与技术类专业高年级本科生、研究生的教材，也可作为 ACM 竞赛培训和成人教育自学教材，同时可供程序设计开发人员、广大科技工作者和研究人员参考。

图书在版编目(CIP)数据

算法分析与设计：微课视频版/李恒武编著. —北京：清华大学出版社，2022.1(2024.7 重印)
ISBN 978-7-302-58509-1

Ⅰ. ①算… Ⅱ. ①李… 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(2021)第 121977 号

责任编辑：刘向威 常晓敏
封面设计：文 静
责任校对：焦丽丽
责任印制：沈 露

出版发行：清华大学出版社
网 址：https://www.tup.com.cn，https://www.wqxuetang.com
地 址：北京清华大学学研大厦 A 座 **邮 编**：100084
社 总 机：010-83470000 **邮 购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：https://www.tup.com.cn，010-83470236
印 装 者：三河市龙大印装有限公司
经 销：全国新华书店
开 本：185mm×260mm **印 张**：21 **字 数**：510 千字
版 次：2022 年 1 月第 1 版 **印 次**：2024 年 7 月第 4 次印刷
印 数：4001～5000
定 价：69.00 元

产品编号：090284-01

前言

PREFACE

智能时代的今天，互联网是道开胃菜，人工智能是主菜。但不管是 AlphaGo 打遍天下无敌手，还是红客与黑客网络大战，程序设计都是必要的元技能。“如果你控制了代码，那就控制了世界。”这是未来学家 Marc Goodman 的预言，现在正在慢慢成为现实。

“Pascal 之父”Nicklaus Wirth 提出的公式“算法＋数据结构＝程序”展示了程序的本质。算法不但是程序，也是计算机科学的核心和灵魂。David Berlinski 更是认为算法成就了现代世界。

“科学殿堂里陈列着两颗熠熠生辉的宝石：一颗是微积分；另一颗就是算法。微积分成就了现代科学，而算法成就了现代世界。”

面对各个应用领域的大量复杂问题，最重要的是建立数学模型并设计高效的求解算法。在当今复杂、海量信息的大数据处理中，好算法往往是一锤定音的利器。

本书从解决问题和应用实例入手，按照提出问题、分析问题、解决问题、总结问题的步骤，培养学生分析问题和解决问题的能力。以实践和能力为导向，线上开放与线下实践相结合，自动评测和互动交流相结合，专题案例与思考讨论相结合，聚集和重组课程内容，适应探究性和碎片化学习，适应自主性和多样化学习，适应多元化和个性化需求，培养学生主动学习、研究和创新意识。

本书共 14 章，主要内容包括算法与问题、算法分析、枚举算法、贪心算法、递推算法、分治算法、动态规划算法、回溯算法、分支限界、网络流算法、随机算法、计算复杂性、近似算法和图算法。

本书适合作为高等院校计算机相关专业高年级本科生和研究生的教材，也可作为 ACM 竞赛培训和成人教育自学教材，还可作为电子工程技术人员的参考用书。

本书由李恒武老师编写。特别感谢耿蕾蕾老师和张琦乾同学的审核和建议，张琦乾同学给出全书算法示例和 POJ 编程习题的标准模板。在本书的编写过程中，得到了许多老师和学生的支持与帮助，在此表示诚挚的感谢！

本书是中国大学 MOOC、智慧树和学银在线一流课程配套教材，也是工科联盟和一流专业课程配套教材，提供完整的视频、电子教案、知识梳理、章节测验、实践作业、思考讨论、在线题库和文档资源，便于教学和学生实践。

最后，衷心祝愿读者能够从此书中获益，从而实现自己的编程梦想。由于本书的内容较多、牵涉的技术较广，书中疏漏之处在所难免，欢迎读者在使用过程中提出宝贵意见。

李恒武

2021 年 4 月

目录

CONTENTS

第1章

算法与问题


视频讲解

1.1 稳定匹配问题

小波的爸爸 GS 开了一个婚姻介绍所，会员有 n 个单身男孩和 n 个单身女孩。爸爸想让小波设计一个程序，快速配对 n 对新人，并且使男孩和女孩都比较满意，并比较稳定。小波与大李老师进行探讨。

1.1.1 问题分析

小波：大李老师，这个问题怎么解决？

大李老师：这个问题比较复杂，也比较模糊，需要你有火眼金睛，透过现象看本质。用计算机处理问题，需要分析输入和输出的关系，建立数学模型，然后设计算法来解决问题。计算机处理问题常常是对现实的模拟，小波，你爸爸平时是怎么做的呢？

小波：每个人心中都有一杆秤。爸爸让男孩给女孩打分并进行排序，女孩给男孩打分也进行排序，然后安排他们约会。

大李老师：对，这就是输入，输入就是 2 张喜欢列表，如表 1-1 和表 1-2 所示。X 喜欢 A 甚于 B 甚于 C，A 喜欢 Y 甚于 X 甚于 Z……

表 1-1 男孩的喜欢列表

喜欢次序	1^{st}	2^{nd}	3^{rd}
X	A	B	C
Y	B	A	C
Z	A	B	C

表 1-2 女孩的喜欢列表

喜欢次序	1^{st}	2^{nd}	3^{rd}
A	Y	X	Z
B	X	Y	Z
C	X	Y	Z

小波：我知道了，输出是 n 对新人。现代社会一夫一妻制，每个男孩正好和一个女孩是一对。例如，X 和 A、Y 和 B、Z 和 C。因此，这是一个完美匹配。

大李老师：对。关键是怎么保证比较满意和比较稳定呢？

X-C、Y-B、Z-A 是否稳定？Y 和 Z 找到最喜欢的女孩，满意了，但是 X 肯定不满意。如果 X 和 B 呢？这样组合的话，X 喜欢 B 甚于现在的女朋友 C，B 喜欢 X 甚于现在的男朋友 Y，双方都对现在的组合不满意，也不稳定，因此称 X 和 B 为不稳定配对。

X-A、Y-B、Z-C 是否稳定呢？X 和 Y 找到自己最喜欢的女孩，满意了，但是 Z 肯定不满意。但是，Z-B 和 Z-A 时，B 和 A 都喜欢现在的男朋友甚于 Z，因此情况不会发生改变。所以，称 X-A、Y-B、Z-C 为稳定匹配。

因此，程序最终的输出应该是一个稳定匹配：没有不稳定配对的完美匹配。这就是问题分析。

1.1.2 稳定匹配算法

大李老师：现在怎样找出这个稳定匹配呢？小波，你爸爸是怎样处理的？

小波：让男孩根据喜欢列表约会女孩，从中撮合。

大李老师：现在用表 1-3 和表 1-4 作为示例模拟约会过程。

表 1-3 男孩的喜欢列表

喜欢次序	1^{st}	2^{nd}	3^{rd}	4^{th}
W	D	B	A	C
X	B	C	D	A
Y	A	D	C	B
Z	B	D	A	C

表 1-4 女孩的喜欢列表

喜欢次序	1^{st}	2^{nd}	3^{rd}	4^{th}
A	Z	X	Y	W
B	X	W	Y	Z
C	W	X	Y	Z
D	Z	Y	X	W

初始时，所有人单身。W 首先约会最喜欢的女孩 D，D 单身，同意相处看看，他们临时成为一对。同理，X 约会最喜欢的女孩 B，B 单身，他们临时成为一对。Y 约会最喜欢的女孩 A，A 单身，他们临时成为一对。随后 Z 约会最喜欢的女孩 B，B 喜欢男朋友 X 甚于 Z，因此拒绝了 Z。Z 再约会 D，D 是 Z 还没约会过的女孩中最喜欢的。然而，D 喜欢 Z 甚于现在

的男朋友 W，因此 Z 和 D 临时成为一对，W 成为单身。W 再约会 B，B 更喜欢现在的男朋友 X，因此 W 被拒绝。W 再约会 A，A 更喜欢现在的男朋友 Y，W 再次被拒绝。W 再约会 C，C 单身，因此 Z 和 C 成为一对。这样就形成了 4 对：C-W、B-X、A-Y 和 D-Z。小波，这是一个稳定匹配吗？

小波：女孩 B、C 和 D 找到最喜欢的男朋友，肯定满意。只有 A 可能不满意。A 和 Z 行吗？Z 更喜欢现在的女朋友 D，不同意。A 和 X 行吗？X 更喜欢现在的女朋友 B，也不同意。因此，这是一个稳定匹配。

大李老师：这个算法是 GS(Gale-Shapley)稳定匹配算法，由数理经济学家 David Gale 和 Lloyd Shapley 于 1962 年设计。算法描述如下。

GS 算法：

```
1.  初始化,每个人单身
2.  while (某个男孩 m 单身,并且他有没约会过的女孩) do
3.      选择男孩 m 没约会过的女孩中最喜欢的 w
4.      if (w 单身) then m 和 w 成为一对
5.      else if (w 喜欢 m 甚于现在的男朋友 m') then
6.              m 和 w 成为一对,并且 m'成为单身
7.          else w 拒绝 m
8.  return 稳定匹配
```

1.1.3　正确性证明

大李老师：上面的例子求得一个稳定匹配，是否对于任意两个喜欢列表都能得到一个稳定匹配？这需要证明以下两点。

(1) GS 算法在有限时间结束，得到结果。

(2) GS 算法的结果是一个稳定匹配：一个完美匹配，并且没有不稳定配对。

小波，你能证明吗？

小波：每个男孩至多约会 n 次，因此至多 n^2 次循环算法会终止。实际上，最后一个单身女孩接受约会，算法就会终止。因此，GS 算法会在有限时间结束。

大李老师：证明稳定匹配的结果之前，先看下面两个规律。

(1) 男孩按照喜欢列表从高到低约会女孩。

(2) 一旦一个女孩有男朋友，她将一直有男朋友。女孩对于新约会的男孩要么拒绝，要么接受，结果还是有男朋友。

首先证明结果是一个完美匹配：所有的男孩和女孩最终都一一配对。使用反证法，假设 Z 最后没有女朋友，肯定有个女孩没有男朋友，假设是 A。算法终止时，Z 单身。Z 肯定向所有女孩发过约会邀请，否则算法不会终止。Z 向 A 发出约会邀请，根据规律(2)，A 不管接受或拒绝，A 都应该有男朋友。这与假设矛盾。因此，结果肯定是一个完美匹配。

再证明结果中没有不稳定配对。使用反证法，假设 A-Z 是不稳定配对，有如下两种情况。

(1) Z 从来没向 A 发出约会邀请。

根据规律(1)，Z 按照喜欢列表从高到低约会。

⇒　Z 喜欢目前的女朋友甚于 A。

⇒ A 和 Z 是稳定的。

(2) Z 向 A 发出过约会邀请。

⇒ A 拒绝 Z(约会时或以后)。

⇒ A 喜欢目前的男朋友甚于 Z。

⇒ A 和 Z 是稳定的。

只有上述 2 种情况,并且都与假设相矛盾。因此,最后结果是一个稳定匹配,一个没有不稳定配对的完美匹配。小波,给定喜欢列表会有几个稳定匹配?通过 GS 算法找到的是哪个?

小波:利用表 1-5 和表 1-6,可有 2 个稳定匹配。X-A 和 Y-B,这个匹配中男孩都找到最喜欢的。X-B 和 Y-A,这个匹配中女孩找到最喜欢的。这 2 个都是稳定匹配,GS 算法找到的是第 1 个匹配。

表 1-5 男孩的喜欢列表

喜欢次序	1^{st}	2^{nd}
X	A	B
Y	B	A

表 1-6 女孩的喜欢列表

喜欢次序	1^{st}	2^{nd}
A	Y	X
B	X	Y

大李老师:对,GS 算法找到的是对男孩来说最好的匹配,对女孩来说最差的匹配。因此,积极主动会有好的结果。

1.1.4 算法实现

大李老师:小波,有了 GS 算法,你能结合数据结构实现吗?

小波:没问题。

Step1:初始化,每个人单身。

n 个单身男孩和 n 个单身女孩,使用 1 到 n 表示,2 张喜欢列表用二维数组 boy 和 girl 表示。单身和配对使用数组 wife 和 husband 表示。其中,wife[1]=0 表示 1 号男孩单身;wife[1]=3 表示 1 号男孩的女朋友是 3 号女孩。

Step2:while(某个男孩 m 单身,并且他有没约会过的女孩)。

如果从前面的数组 wife 中搜索取值为 0 的男孩,最坏情况下需要 n 次。可以单独建一个数组 free 存储单身男孩,但数组不利于维护动态变化的元素,因此使用栈或队列实现,便于随时插入或随时删除。初始化时栈 free 存放所有男孩,每次循环在栈顶插入或删除。

Step3:选择男孩 m 没有约会过的女孩中最喜欢的 w。

可以使用计数数组 count 存放约会次数,boy[m][count[m]+1]就是男孩 m 下一个要约会的女孩 w。

Step4:if (w 单身) then m 和 w 成为一对。

即 wife[m]=w,husband[w]=m。

Step5：else if（w 喜欢 m 甚于现在的男朋友 m'）。

可以查 girl[w]表(表 1-7)比较喜欢程度。

Step6：then m 和 w 成为一对，并且 m'成为单身。

即 wife[m]=w,husband[w]=m,wife[m']=0,m'入栈。

Step7：else w 拒绝 m。

即 wife[m]=0,m 入栈。

Step8：return 稳定匹配。

即返回数组 wife 或数组 husband。

大李老师：查表 girl[w]可以比较女孩 w 对 2 个男孩的喜欢程度，如表 1-7 所示，最坏情况下需要比较 n 次。程序循环次数为 n^2，这样比较次数总共需要 n^3 数量级。算法能不能改进呢？

把数组 girl[w]变换为 inverse_girl[w]，由 i^{th} 喜欢的某个男孩，变为对 i 号男孩的喜欢程度，分别如表 1-7 和表 1-8 所示。例如，girl[w]表中第 2 喜欢的是 3 号男孩，第 7 喜欢的是 6 号男孩。在 inverse_girl[w]表中，对 3 号男孩的喜欢程度为 2^{nd}，对 6 号男孩的喜欢程度为 7^{th}，这样直接查找 2 次，进行比较即可。

表 1-7　girl[w]

女孩 w 的喜欢程度	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}	8^{th}
男孩	8	3	7	1	4	5	6	2

表 1-8　inverse_girl[w]

男孩	1	2	3	4	5	6	7	8
女孩 w 的喜欢程度	4^{th}	8^{th}	2^{nd}	5^{th}	6^{th}	7^{th}	3^{rd}	1^{st}

Inverse 变换算法：

```
1.  for j = 1 to n do
2.      for i = 1 to n do
3.          inverse_girl[girl[j]][girl[j][i]] = i
```

使用 Inverse 变换算法，girl[w]表变换为 inverse_girl[w]表，变换的计算次数不超过 n^2 的数量级。这样，GS 算法计算的数量级为 n^2，否则为 n^3。因此，算法需要结合良好的数据结构以减少计算量。

1.1.5　算法总结

稳定匹配问题：给定 n 个男孩和 n 个女孩的喜欢列表，是否存在一个稳定匹配？如果存在，则求解该稳定匹配。

Gale-Shapley 算法：对于稳定匹配问题的任意实例，n^2 次数量级计算保证找到一个稳定匹配。这个匹配对男孩来说是最优的。

问题求解过程：问题给出后首先要分析问题，稳定匹配问题的输入是 2 张喜欢列表，输

出是稳定匹配。然后设计 GS 算法，证明算法的正确性，证明算法在有限时间终止并且结果正确。随后，结合数据结构分析算法的性能和特点，结合编程语言实现算法。

本节思考题

作弊问题：如果某个人得到 2 张喜欢列表，是否可以改变自己的喜欢次序，从而在最后稳定匹配结果中匹配更喜欢的异性？

视频讲解

1.2 算法概述

1.2.1 算法的概念

算法是一步一步正确解决问题的方法和策略。对计算机算法来说，任何问题使用计算机求解，最终要设计程序，然后转为机器指令，运行指令序列得到结果。因此，计算机算法是由若干条指令组成的有穷序列，规定了解决某个特定类型问题的一系列运算步骤，具有分步、有序、有穷、有目的、可操作的特点。

例 1：使用数字 2，得到数值 8。

可以使用加法、乘法、幂乘以及移位操作等实现。一般情况下，基本运算的时间排序为幂乘＞除法＞乘法＞加减＞移位，因此使用移位操作更快。

例 2：计算 a^{10}。

可以使用连乘、乘法、乘方实现，有如下 3 种算法。

(1) $a^{10}=a\times a\times a\times a\times a\times a\times a\times a\times a\times a$。

(2) $y=a\times a$，$z=y\times y$，$w=z\times z$，$a^{10}=y\times w$。

(3) $a^{10}=[(a\times a)^2\times a]^2$。

显然，方法(2)使用 4 次乘法更快，但增加了 4 个变量，通过以空间换时间来减少计算。实际上，关注的重点是如何求 a^n 这类特定问题的通用算法。a^{10} 仅是 $n=10$ 的一个特例。当 n 很大时，方法(2)需要增加的变量太多，并不适合作通用算法。

Q：为什么要研究算法呢？

A：计算同一问题，不同算法的计算时间不一样。对于百万次计算机，当 $n=50$ 时，n^2 算法不到 1 秒可以完成，2^n 算法则需要 30 多年，如表 1-9 所示。即使计算机的速度提高 1000 倍，2^n 算法计算同样的工作量，n 至多等于 60。因此如果算法需要的计算量太大，单纯提高计算机的速度难以解决，必须研究同一问题的不同算法，找出好的有效的算法。

表 1-9 百万次计算机需要的时间

n	50	60
n^2	0.0025 秒	0.0036 秒
2^n	35.7 年	36 558 年

1.2.2 算法的性质

算法具有如下 4 个性质。

(1) 输入：算法有 0 个或多个输入。

(2) 输出：算法产生 1 个或多个输出，与输入有某种特定关系。

(3) 确定性：算法的每条指令(步骤)必须要有确切的含义，必须是清楚的、无歧义的。

(4) 有穷性：算法中的指令有限，每条指令的执行次数有限，执行时间也有限，因此算法总是在有限时间终止。也就是说算法步骤有限，每步都可以通过基本运算得以实现，基本运算的次数和时间有限，因此算法总是在有限时间终止。

算法有 0 个或多个输入，有 1 个或多个输出。例如，计算机智能吗？这个问题没有输入，至少要输出 1 个是或否的结果，因此至少有 1 个输出。

同一算法可以使用不同的形式描述。算法可以使用流程图、程序、自然语言、伪代码等描述，但要保证确定性。使用自然语言进行描述，容易产生歧义、不严密，复杂算法难以表达。使用流程图进行描述，不方便，不易表示数据结构。使用伪代码进行描述，方便、精确，但不简洁。因此，一般使用自然语言和伪代码的混合结构进行描述。

1.2.3　算法与程序

Q：算法和程序有什么区别？

A：程序是算法用某种程序设计语言的具体实现。算法与程序的主要区别是有穷性，程序可以不满足算法的性质(4)。程序使用 C、Java 等计算机可以理解的语言实现，不能使用自然语言实现。而算法可以使用自然语言描述，但要保证确定性。

Q：操作系统是不是一个算法？

A：操作系统是一个无限循环执行的程序，不是一个算法。操作系统中的各项任务由各子程序通过特定的算法来实现。子程序得到输出结果后便终止，满足算法的 4 个性质，因此子程序是算法。

1.2.4　算法与问题

问题是一个要求给出解答的一般性提问。问题的两个要素是输入和输出，且要有严格的描述。输入描述问题的所有参量，又称为实例；输出描述问题所求解的格式和应满足的性质，又称为询问。例如，稳定匹配问题的输入是 2 个喜欢列表，输出是 1 个稳定匹配，稳定是需要满足的性质，匹配是求解的格式。

用计算机求解问题，要有符合规范的输入，在有限时间获得要求的输出。求解问题的方法或策略是算法，算法的目的是求解问题。同一问题可能有几种不同的算法，解题思路和解题速度也会显著不同。例如，排序问题有插入、冒泡、选择、堆、快速排序等算法。

输入：n 个元素的序列$<a_1,a_2,\cdots,a_n>$。

输出：对输入序列排序为$<a'_1,a'_2,\cdots,a'_n>$，使当 $i<j$ 时 $a'_i \leqslant a'_j$。

实际计算机的输入是一个序列，如$<5,4,2,10,7>$，是输入参数的一组赋值，称为实例。计算机每次求解只是针对问题的一个实例求解，问题的描述针对该问题的所有实例，是通用的描述。如果一个算法能应用于问题的任意实例，并保证得到正确解答，才能称这个算法解答了该问题。

1.2.5　问题求解

问题的求解是提出问题、分析问题、解决问题的过程，如图 1-1 所示。

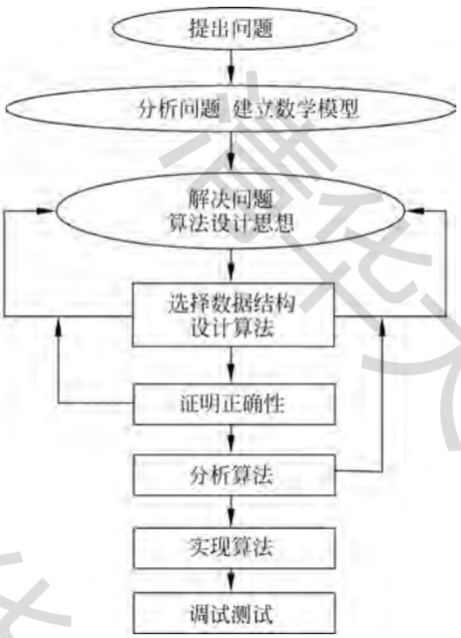

图 1-1 问题求解过程

提出问题，科学上需要对问题的领域融会贯通，才能高屋建瓴地提出新问题。

提出问题以后要对问题进行分析。现实中的问题往往错综复杂，模糊不清，需要透过现象看本质，弄清问题的输入和输出、输入和输出之间的隐含关系。通过分析给出其描述，为了使描述更清晰明白，需要使用最准确的数学语言来描述，并在此基础上建立该问题的数学模型。对于同一问题可以使用不同的数学工具建立不同的数学模型，因此还需要对模型进行分析、比较和优化，选出最有效的模型。

有了数学模型之后，再寻找求解问题的方法，这就是算法设计。设计出求解问题的步骤，这是解决问题的核心。一开始这种方法可能比较粗略，仅是一种算法思想。需要结合数据结构，进一步细化，才能设计出有效的算法。可以使用结构化的自顶向下、逐步求精的方法，也可以使用面向对象的方法。同一模型使用不同的数据结构会有不同的算法，其有效性差别很大，因此需要对数据结构进行优化。

设计算法后需要证明算法的正确性，分析算法的性能。算法正确性需要证明两点：算法在有穷步终止，并且对一切合法的输入都能得出正确的结果。算法的正确性一般使用数学归纳法或反证法证明。证明算法不正确更简单，只需给出一个反例，说明算法不能正确处理即可。

同一问题，使用不同模型、不同数据结构，会存在多种算法，需要分析、比较其性能和特点，进行方案选择。需要分析算法所耗费的时间和空间资源，分析算法的最好、最坏和平均情况，分析算法的健壮性、适应性等指标。算法设计与分析是一个不断反复的过程，这几个步骤可能会有多次反复。

确定方案后，使用高级语言实现算法。算法的实现方式对运算速度和所需存储容量也有很大影响。最后对程序进行调试、测试和结果分析。

本节思考题

玻璃瓶样品测试：给定 n 级台阶，找最高安全台阶（样品下落而不摔碎的最高台阶）。如果给定 1 个瓶子、2 个瓶子或 k 个瓶子，将如何测试？

1.3　问题变换


视频讲解

1.3.1　大学入学申请

许多大学录取实行申请制，学生提出申请，大学发录取通知。如果 W 大学发了录取通知给 A，A 又收到了 X 大学的录取通知，A 更喜欢 X 大学，A 会拒绝 W 大学。W 大学再发录取通知给 B，B 又收到了 Y 大学的录取通知，B 更喜欢 Y 大学，拒绝了 W 大学。W 大学又向 C 发出录取通知……这样的过程会反复不断地进行，如何设计一个程序，自动完成匹配，使学校和学生都满意？

问题形式化描述：n 个申请者向 m 所大学递交申请，每所大学有招生计划，求一个稳定匹配。

大学入学申请问题的输入和输出与稳定匹配问题非常类似，输入是 2 个喜欢列表，输出是 1 个稳定匹配。这 2 个问题的不同如表 1-10 所示。

表 1-10　稳定匹配问题与大学入学申请问题的不同点

不　同　点	稳定匹配问题	大学入学申请问题
参与者	n 男 n 女	n 校 m 生
匹配	一对一	一对多
参与者不喜欢	未涉及	不申请或不录取

因此，大学入学申请问题当 $n=m$ 且一校只录取一个学生时，变换为稳定匹配问题。稳定匹配问题是大学入学申请问题的核心问题。

Q：能不能利用稳定匹配算法解决大学入学申请问题？

A：分析不同的 3 种情况。

1）校生数不同

假设 $n<m$，运行稳定匹配算法，学校发出录取通知，学生接受或拒绝，最多执行 nm 次循环算法会终止，存在 $m-n$ 个没收到录取通知的学生，因为学生一旦收到录取通知，会一直有录取通知。最后，收到录取通知的 n 个学生是稳定匹配，因为学校喜欢已录取的学生胜过没有收到录取通知的学生。

2）一校多生

当 $n<m$ 且一校多生时，每所学校录取的学生不超过招生计划。运行稳定匹配算法，学校根据招生计划发出录取通知，学生接受或拒绝，当所有学校的招生计划完成时终止算法。最后，收到录取通知的学生是稳定匹配，因为学校喜欢已录取的学生胜过没有收到录取通知的学生。

3）不申请(录取)

当有人坚决不申请某些学校或学校不录取某些学生时，直接在“girl”和“boy”表中标记，算法执行过程中遇到标记则不进行匹配，稳定匹配算法仍然可以运行。

综合上述3种情况，可以解决大学入学申请问题。这样稳定匹配问题变换为大学入学申请问题。稳定匹配问题是大学入学申请问题 $n=m$ 且一校一生的特殊情况，也是大学入学问题简化的核心问题。所以，解决问题的一个思路就是首先要把一个实际问题表述为尽可能简单的但又能抓住问题本质的核心问题和模型，然后设计求解这一核心问题的算法，进而分析算法的正确性与有效性。在核心问题的算法基础上进一步扩展用于解决实际问题。

问题求解是一个不断螺旋式上升的过程，不断解决旧问题，且不断产生新问题。掌握的问题越多，则掌握同一问题的不同算法越多，越有利于求解新问题。这也是学习算法的一个目的，熟悉旧问题和算法，用于解决新问题。

1.3.2 问题变换

问题变换是求解新问题的常用方法，问题变换可以将复杂的问题转化为简单的问题，将难解的问题转化为易解的问题，将隐式的问题转化为显式的问题，将未知的问题转化为已知的问题。

1. 复杂的问题转化为简单的问题

可以通过实例化简，将问题变为同一问题的更简单、更便利的实例。例如，使用预排序将无序数组变为有序数组，多次查找时使用折半查找更简单。求解线性方程组，将系数矩阵变为更简单的上三角矩阵，便于求解。

元素唯一性问题：判断 n 个元素的数组中每个元素是否都是唯一的。

如果使用蛮力法，a_1 与 $a_2,a_3,\cdots,a_n$ 分别比较，需要比较 $n-1$ 次；a_2 与 $a_3,a_4,\cdots,a_n$ 分别比较，需要比较 $n-2$ 次，……这样总的比较次数为 $\sum_{i=1}^{n-1}(n-i)=n(n-1)/2$。如果首先进行预排序，然后遍历查看相邻元素是否相同，则只需比较 $n-1$ 次。

2. 难解的问题转化为易解的问题

如大学入学申请问题，将问题限制为 n 生 n 校且一生一校，就转变为稳定匹配问题。然后，在稳定匹配算法的基础上进行扩展，两者不同的3种情况都得到了解决，也就解决了大学入学申请问题。

可以通过表达变换，把问题变为同一实例的不同表示。例如，使用堆排序和合并排序的时间复杂度相同，但空间复杂度由 $O(n)$ 降为 $O(1)$；二叉查找树的时间复杂度，平均情况为 $O(\log n)$，但最坏情况是退化为 $O(n)$ 时间的单边树，可以变换为AVL树和红黑树限制左右子树的高度差，也可以变换为2-3树、2-3-4树和B树允许一个节点包含多个元素，这样既保留二叉查找树的特性，又避免退化到最差的情况。

求解多项式 $P_n(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$。

如果从左向右进行计算，需要计算 $n+(n-1)+\cdots+1+n=n(n+3)/2$ 次。可以运用霍纳法则进行变换，从内向外计算，只需要计算 $2n$ 次。

霍纳法则：$P_n(x)=((\cdots(((a_nx+a_{n-1})x+a_{n-2})x+a_{n-3})\cdots)x+a_1)x+a_0$。

霍纳法则多项式时间算法 $P(A,x)$：

```
1.  p = a[n]
2.  for i = n-1 to 0 do
3.      p = p*x+a[i]
4.  return p
```

3. 隐式的问题转化为显式的问题

许多复杂问题可以约简到树或图，可使用图算法求解，如渡河问题。

渡河问题：有 3 个传教士和 3 个野人来到河边准备渡河，河岸有一条船，每次最多可乘坐 2 个人。在任何时刻，河的两岸以及船上传教士人数不能少于野人的人数，则传教士应如何规划摆渡方案？

使用三元组表示左岸的传教士、野人和船的数量。每个三元组表示一个状态，开始状态(3,3,1)表示左岸有 3 个传教士、3 个野人、1 条船，终止状态(0,0,0)表示左岸没有传教士、没有野人、没有船。去掉不合理的状态，只有 16 个状态。一个状态可以变换为另一个状态，如从开始状态到一个传教士和一个野人上船渡河，则(3,3,1)变换为(2,2,0)。

将每个状态变为图中的一个顶点，若一个状态可以转换为另一个状态，则在相应顶点间增加双向边，这种图称为状态空间图，如图 1-2 所示。最终，渡河问题变为在图中搜索从起点到终点的路径问题。迷宫和皇后等二维表格上的问题，五子棋和象棋等博弈问题，都可以将隐式图转化为显式图进行求解。

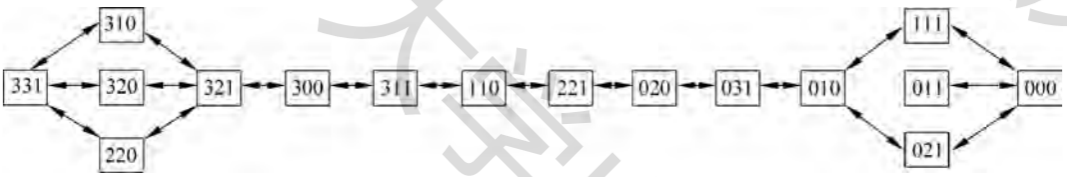

图 1-2　状态空间图

4. 未知的问题转化为已知的问题

可以使用问题化简，将一个未知问题转化为另一个问题的实例，且转化后问题的求解算法是已知的，因此得以求解原问题。可以利用等价转化为另一问题求解，也可以将特殊问题转化为一般问题求解，还可以在最大化问题和最小化问题间进行变换求解。例如，可以将最小公倍数问题变换为最大公约数问题求解。由于(m,n)的最小公倍数$=(m\times n)/(m,n)$的最大公约数，而已知欧几里得算法可以求最大公约数，因此最小公倍数问题可以被求解。

欧几里得算法：

输入：正整数 m 和 n。

输出：m 和 n 的最大公约数。

```
1. r = m mod n                    //求余数
2. if (r == 0) then return n
3. m = n, n = r, goto Step1       //互换
```

最大独立集问题：给定无向图 $G=(V,E)$，$V^*\subseteq V$，若 V^* 中任何两个顶点均不相邻接(没有边相连)，则称 V^* 为 G 的独立集。顶点数最多的独立集，就是最大独立集，如图 1-3 所示，{1,6,4,5}是最大独立集。

最小顶点覆盖问题：给定无向图 $G=(V,E)$，$V'\subseteq V$，若对于 $\forall e\in E$，$\exists v\in V'$，使得 v

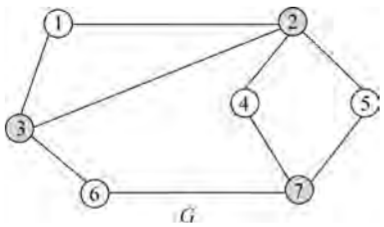

图 1-3 最大独立集和最小顶点覆盖示例

与 e 相关联，则称 v 覆盖 e，并称 V' 为 G 的顶点覆盖。顶点数最少的顶点覆盖，就是最小顶点覆盖，如图 1-3 所示，$\{2,3,7\}$ 是最小顶点覆盖。实际上 $V'=V-V^*$，如果 V^* 是最大独立集，则 V' 是最小顶点覆盖，因此两者是等价问题。

最大团问题：给定无向图 $G'=(V,E')$，$(u,v)\in E'$ 当且仅当 $(u,v)\notin E$，则 G' 是 G 的补图。无向图 G 中 $V^*\subseteq V$，若 V^* 中任何两个顶点都相邻接（有边相连），则 G 中 V^* 是一个团。顶点数最多的团，就是最大团。V^* 是 G' 的最大独立集，则 G 中 V^* 是最大团，因此二者同样是等价问题，如图 1-4 所示 G 中 $\{1,2,4\}$ 是最大团，G' 中 $\{1,2,4\}$ 是最大独立集。

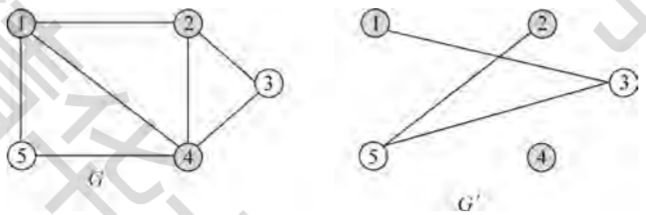

图 1-4 最大团和最大独立集示例

区间调度问题：给定一间报告厅以及 n 个报告的开始时间和结束时间，如何安排使被安排的报告数量最大？每个报告对应一个区间，开始时间和结束时间对应区间的起点和终点，区间调度问题就是最大相容区间问题。将每个区间变为图中的一个点，不相容的区间使用边相连，则最大相容区间问题转化为图中最大独立集问题。如图 1-5 所示，区间调度中最大相容区间为 $\{2,5,8\}$，转化后变为图的最大独立集。

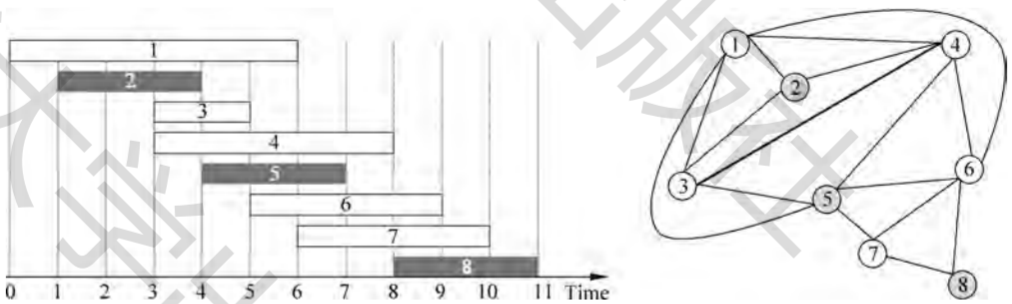

图 1-5 最大独立集和区间调度示例

本节思考题

一个猎人带着一只狼、一头羊和一筐青菜欲乘一条独木舟渡河，因独木舟太小，猎人一次至多只能带狼、羊或青菜中的一种渡河，但当它们与猎人不在河的同一边时，狼会吃掉羊、羊会吃掉青菜。试设计一个算法，帮猎人寻找一种渡河的次序，让其不受损失地将狼、羊和青菜带过河去。

本章习题

1. 编程实现最小公倍数问题（POJ 3970）。
2. 编程实现稳定匹配问题（POJ 3487）。
3. 解决问题的基本步骤是(　　)。

A. 算法设计　　B. 算法实现
C. 数学建模　　D. 算法分析
E. 正确性证明

4. 问题变换的目的有(　　)。

A. 复杂变简单　　B. 未知变已知
C. 隐式变显式　　D. 难解变易解
E. 以上都是

5. 简述算法与程序的区别。
6. 简述算法与问题的关系。
7. 下面关于程序和算法的说法正确的是(　　)。

A. 算法的每一个步骤必须要有确切的含义，必须是清楚的、无二义的
B. 程序是算法用某种程序设计语言的具体实现
C. 如果一个算法能应用于问题的任意实例，并保证得到正确解答，则称这个算法解答了该问题
D. 算法是一个过程，计算机每次求解是针对问题的一个实例求解

8. 下面关于算法的说法错误的是(　　)。

A. 同一数学模型使用不同的数据结构会有不同的算法，有效性有很大差别
B. 证明算法不正确，只需给出一个反例，说明算法不能正确处理即可
C. 算法是一个语句集合，按照顺序执行语句，处理实例，得到正确答案
D. 同一算法只有一种形式描述

9. 元素唯一性问题(重复元素问题)：n 个元素数组中的每个元素是否都是唯一的？是否存在数量级为 n 的算法？试设计该问题。

10. 稳定室友问题：$2n$ 个男孩，n 间宿舍，每个房间只能安排 2 个人，每个男孩对于其他人有一个喜欢列表。如何得到稳定的安排使关系好的在一起？

11. 给定稳定匹配问题的实例 I，存在男孩 w 和女孩 m，m 在 w 的喜欢列表中排第一，w 在 m 的喜欢列表中排第一，那么(m，w)肯定在任何稳定匹配中都是一对。该命题是否成立？请给出解释。

12. 给定无向图 $G=(V,E)$，图中任意两个顶点 u 和 v 都存在一条从 u 到 v 的路径，则称 G 为连通图。试设计算法判定 G 是否是连通图。

13. 给定有向图 $G=(V,E)$，图中任意两个顶点都相互可达，则称 G 为强连通图。试设计算法判定 G 是否是强连通图。

14. 扩展欧几里得算法：求正整数 m 和 n 的最大公约数 d，满足 $mx+ny=d$，x 和 y

为整数。

15. 欧几里得游戏：黑板上给定两个不相等的正整数，两个玩家轮流在黑板上写数字，写的数字必须是黑板上任意两个数的差，并且不是黑板上已有的数字。轮到哪个玩家写不出数字则为输方，则应该选择先行动，还是后行动？

16. 图着色问题：给定图 $G=(V,E)$，使用最少的颜色给顶点着色，使任何相邻顶点着不同色。边着色问题：给定图 $G=(V,E)$，使用最少的颜色给边着色，使任何有共同端点的边着不同色。如何将边着色问题转化为图着色问题？

17. 如何将最小化问题转化为最大化问题？

18. 给定图 $G=(V,E)$，计算任意两点间的路径数量。如何将该问题转化为矩阵问题？

19. 弹性碰撞问题：n 只蚂蚁在长度为 $p$cm 的水平杆上爬行，速度恒定为 1cm/s。当一只蚂蚁到达杆的一端时就会掉落。当两只蚂蚁相遇时，它们会转身并开始向相反方向爬行。给定 n 只蚂蚁在杆上的原始位置 $x_i(i=1,2,\cdots,n)$，但不知道蚂蚁爬行的方向。请计算所有蚂蚁从杆上掉下来的最早和最晚时间。

20. 尺取法：给定长度为 n 的整数数列 $\{a_0,a_1,\cdots,a_n\}$ 和整数 S，求连续子序列的最小长度 p，使子序列的和大于或等于 S。

21. 矩阵快速幂问题：给定矩阵 $\boldsymbol{A}_{n\times n}$，求矩阵 $\boldsymbol{A}$ 的幂之和 $\boldsymbol{S}=\boldsymbol{A}+\boldsymbol{A}^2+\cdots+\boldsymbol{A}^n$。