青少年计算机竞赛指导丛书

青少年国际和全国信息学（计算机）奥林匹克竞赛指导

——图论的算法与程序设计

吴文虎 王建德 编著

清华大学出版社

青少年计算机竞赛指导丛书

青少年国际和全国信息学（计算机）奥林匹克竞赛指导

——图论的算法与程序设计

吴文虎　王建德　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书介绍了有关图论的基本概念和典型的图论算法，结合历届赛题分析如何把一个实际问题抽象化为数学中的图论问题，并给出了程序解法。本书的特点是既有基本概念的讲解及对所解问题的分析，又有编程的思路与参考程序。是参加国际和全国奥林匹克竞赛学生的必读书，也可作为大学生的参考书。

图书在版编目(CIP)数据

青少年国际和全国信息学(计算机)奥林匹克竞赛指导：图论的算法与程序设计/吴文虎，王建德编著．—北京：清华大学出版社，1996.8
(青少年计算机竞赛指导丛导)
ISBN 7-302-02237-2

Ⅰ．青…　Ⅱ．① 吴…　② 王…　Ⅲ．图论算法-程序设计-中学-计算机课-教学参考资料　Ⅳ．G634.673

中国版本图书馆 CIP 数据核字(96)第 13326 号

出版者：清华大学出版社（北京清华大学校内，邮编 100084）
印刷者：人民文学印刷厂
发行者：新华书店总店北京科技发行所
开　本：787×1092 1/16　印张：11　字数：258 千字
版　次：1997 年 3 月第 1 版　1997 年 3 月第 1 次印刷
书　号：ISBN 7-302-02237-2/TP・1087
印　数：0001—5000
定　价：12.00 元

前　言

信息科学与技术正在对人类社会的发展产生难以估量的深远影响，并将成为新世纪的一个标志。作为人类总体智慧的结晶，电脑已经成为一种新的现代文化。“计算机的普及要从娃娃做起”已经成为“科教兴国”的一项重要内容。

一个国家、一个民族要想不落伍，要想跻身于世界先进民族之林，关键在于拥有高素质的人才；综合国力的竞争，说到底也是人才的竞争。电子计算机是现代科学与技术的基础和核心，它的飞速发展，把社会生产力水平提到前所未有的高度，人类进入了信息时代。电脑对人类社会的发展所起的巨大作用，特别是对人类智能的发展所起的促进作用，已为人们普遍认识到。计算，跟 语言一样是人类社会每时每刻都不可缺少的。现在，人类已经拥有了帮助自己进行复杂计算与思维的工具，电子计算机起到了人脑延伸的作用。以往历史上的技术革命，只能起到创造和改进工具，用机器代替人的体力的作用；而计算机则是把人从重复性的或有固定程式的脑力劳动中解放出来，使自己的智能获得空前的发展。作为“人类通用智力工具”计算机在开发人类智能方面所起的无与伦比的作用不容忽视。这也就是计算机与基础教育相结合，能够成为当今世界的大趋势的一个原因。从信息社会要求人才具备的科学素养看，数学、物理学、化学、生命科学和信息科学是五大支柱，这正是联合国教科文组织倡导举行五项国际学科奥林匹克竞赛的内容。

国际信息学奥林匹克(International Olympiad in Informatics，简称 IOI)始于 1989 年，到 1996 年已成功地举办了 8 届。这是一种智力与应用计算机能力的大赛，从益智的角度看，是用电脑帮助开发人脑，重在提高思维能力，培养创新意识。在中国队的训练中强调德智体美全面发展；心态上自立、自尊、自信、自强，要怀着中华民族的自豪感和自信心去参赛；这种心态是学习、训练和取胜的重要条件。8 届比赛，中国队每届都取得了名列前茅的好成绩，31 人次参赛，夺得 31 块奖牌，其中金牌 17 块、银牌 6 块、铜牌 8 块。特别是 IOI'95(荷兰)突破了前 6 届比赛女孩与金奖无缘的纪录，两名中国女选手荣登金牌领奖台。在 IOI'96(匈牙利)上中国队又实现了全金的突破，四名选手，每人夺得 1 块金牌。

从大局看，竞赛不是目的，是推动普及的手段，我们的目的只有一个：“科教兴国”。竞赛活动带有因材施教、因材施测的特点。普及是有层次的，与学科竞赛有关的普及活动，对青少年而言属于比较高的层次，当然就有相当的难度。我们编写的这套青少年计算机竞赛指导丛书，涉及程序设计语言、常用算法、组合数学、图论、人工智能搜索等的基本知识和基本方法。这些理论知识往往都是通过竞赛当中的一些实例来讲解的。目前已拟就了三本书：其一是《组合数学的算法与程序设计》，其二是《图论算法与程序设计》；其三是《国际国内青少年信息学(计算机)竞赛试题解析(1994～1995)》，前两本书试图从数学的基本概念和基本理论出发，给出算法设计和编写程序的方法，第三本书收集了最近国际国内比赛的典型试题及其解法，重点放在解题思路上。这里许多题目比较新颖，很难去套固定算法或固定模式，这中间有些招数是选手们想出来的。从中可以看出信息学奥林匹克要求创

新，鼓励创新。当然，书中给出的解法，对青少年读者而言，我们希望仅仅起到抛砖引玉的作用，并且热切盼望引出更多的玉来。作为老师，我和王建德都这样想，“精心育桃李，热望青胜兰”就是我们编写这套丛书的初衷。

中国计算机学会普及委员会主任
国际信息学奥林匹克中国队总教练
清华大学计算机科学与技术系教授
吴文虎
1996 年 11 月 20 日

目　录

第一章 基本概念

1.1 引　　言

图论是一个应用十分广泛而又极其有趣的数学分支。物理、化学、生物、科学管理、计算机等各个领域都可找到图论的足迹。本书不想对图论作广泛而深入的探讨，主要介绍一下图论的一些基本知识、图论中常用的初等方法和典型的图论程序，试图在抽象理论和具体编程之间为读者架设一座桥梁。

为了让大家知道这本书主要讲些什么，我们先举几个例子。

【例 1】 图 1-1 画的是一个"图"，当然究竟什么叫做"图"，以后还要仔细讲。这里只要先记住，我们研究的图，指的是由顶点和线组成的图形。我们先把图 1-1 看成是一个公路网，$V_1,V_2,\cdots,V_{10}$是一些城镇，每条线旁边的数字代表这一段公路的长度。现在问，要从 V_1 把货物运到 V_{10}，走哪条路最近？

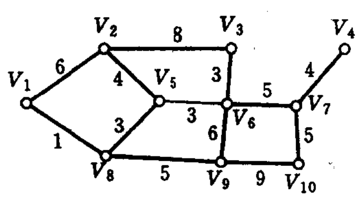


图　1-1

这个问题通常叫做最短路径问题。读者不难看出，这是一个有很大现实意义的问题，它不仅出现在各种运输问题中，而且在电路设计等问题中也有用。因为这种问题研究的是从 V_1 到 V_{10}的所有路径中，哪一条路最短？因此它是一个极大极小问题，即极值问题，而它又和一个"图"密切联系着，因此这种问题就叫做图论中的极值问题。

【例 2】 还是把图 1-1 看成公路网，$V_1,V_2,\cdots,V_{10}$看成公路网的一个站点，若这个公路网目前被敌方占领。请分析一下，能否仅破坏其公路网的一个站点或者至少破坏敌人哪几个站点，就可摧毁敌方整个运输线。

这类问题称为图的连通性问题。原来的公路网，任意两个站点之间互相可达，这样的图称作连通图。一旦删去一些(或一个)点以及和它们关联的边时，这张图就不再成为一张完整的连通图了。因此也有人把这种问题叫做"割点"问题。军事指挥中这类问题就很多。

上面二个问题都是明显地和一个图联系着的。让我们再来讲一个例子，从表面上看，它与图并没有什么关系，但经过仔细分析，可以把它归结为图论中的一个匹配问题。

【例 3】 飞行大队有若干个来自各地的驾驶员，专门驾驶一种型号的飞机，这种飞机每架有两个驾驶员。由于种种原因，例如相互配合的问题，有些驾驶员不能在同一架飞机上飞行，问如何搭配驾驶员，才能使出航的飞机最多。

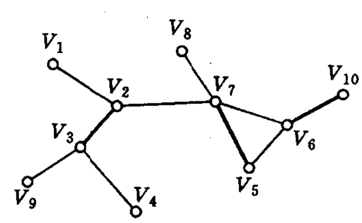


图　1-2

为简单起见，假设有 10 个驾驶员，图 1-2 中的 V_1，

$V_2,\cdots,V_{10}$就代表这10个驾驶员。如果两个人可以同机飞行，就在代表他们两个之间连一条线；两个人不能同机飞行，就不连。例如V_1和V_2可以同机飞行，而V_1和V_3就不行。画了这个图后，就可以研究搭配飞行员的问题了。图1-2中画的3条粗线就代表了一种搭配方案。由于一个飞行员不能同时派往两架飞机，因此任何两条粗线不能有公共的端点，今后我们把一个图中没有公共端点的一组线叫做一个"匹配"。这样定义以后，上面问题就成为：如何找一个包含最多线的匹配？这个问题叫做图的最大匹配问题。请大家试试看，能不能从图1-2中找出一个包含4条线的匹配，再试试能不能找到包含5条线的匹配。

像上述例子那样，将实际生活中的事物分析转化为图论问题的实例还很多，后面还要讲，这里就不多介绍了。

现在可以把这本书的目的说一说了。

这本书主要是围绕一些有趣的数学难题、计算机竞赛试题展开对图论的讨论，讲它们是怎么从实际生活中提炼出来的，怎样通过编程求解这些问题。哪些问题已有精确的解答，哪些仅是近似算法。当然本书不可能穷尽图论知识，只能选一些比较基本的来讲讲。重点是放在编程解题和实际应用上。限于篇幅，对一些编程过程中未涉及的抽象定理只能做简单的介绍而不再推导。市面上有关图论知识的教科书和普及读物甚多，有兴趣的读者可查阅这方面的资料。

1.2 图的定义

图论研究的对象是图，什么是图呢？

图1-3就是一个图，它有若干个不同的点$V_1,V_2,\cdots,V_{11}$，我们称之为顶点。这些顶点中有一些是用直线段或曲线段连接的，我们把这些直线段和曲线段称作边。例如V_1与V_2之间有两条边。若连接两个顶点的边有多条，则这些边称之为平行边。V_2与V_3之间有一条边，V_2与V_4之间没有边……等等。图1-3中，V_1与V_1本身也有边相连，这样的边叫做环。当然，也可能出现某顶点与图中除它外的每一顶点均不相连的情况，这种顶点称为孤立点，例如V_{11}。

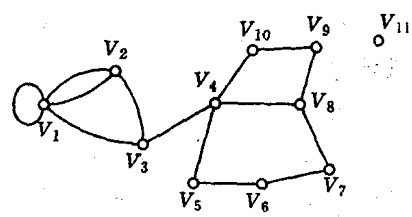


图 1-3

由此得出图的定义：

由若干个不同顶点与连接其中某些顶点的边所组成的图形就称为图。

要注意的是，在图的定义中，顶点的位置以及边的曲直长短都是无关紧要的，而且也没有假定这些顶点和边都要在一个平面内（譬如说，正多面体的顶点和棱也构成一个图）。我们只关心顶点的多少及这些边是连结哪些顶点的。确切地说，如果两个图G与G'的顶点之间可以建立起一对一的对应，并且当且仅当G的顶点V_i与V_j之间有K条边相连的，G'的相应的顶点U'_i与U'_j之间也有K条边相连，我们就说G与G'有相同的结构，简称为同构的。同构的两个图，我们认为是没有区别的。

图1-4与图1-3乍看起来很不一样，其实这两个图却是相同的，这只要将图1-4中的

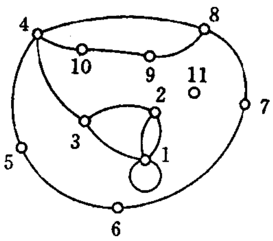


图 1-4

顶点 V_i' 与图 1-3 中的 V_i 相对应($1 \leqslant i \leqslant 11$)就明白了。

通常用一个大写字母 G 来表示图,用 V 来表示所有顶点的集合,E 表示所有边的集合,并且记成 $G=(V,E)$。如果顶点个数 $|V|$ 与边的条数 $|E|$ 都是有限的,图 G 就称为有限图。如果 $|V|=1$,$|E|=0$,图 G 称为平凡图。这种仅含一个孤立点的图是有限图的一种特例。如果 $|V|$ 或 $|E|$ 是无限的,图 G 称为无限图。

本书讨论的图形式都是有限图。

如果对图 $G=(V,E)$ 与 $G'=(V',E')$,G' 的顶点集是 G 的顶点集的一个子集($V' \subseteq V$),G' 的边集是 G 的边集的一个子集($E' \subseteq E$),我们说 G' 是 G 的子图。例如,一个正方形就可以看作是图 1-5(a)的一个子图,一个五边形也可以看作是图 1-5(b)的一个子图。

如果一个图没有环,并且每两个顶点之间最多只有一条边,这样的图称之为简单图。在简单图中,连接 V_i 与 V_j 的边可以记成 (V_i,V_j)。

如果 G 是一个简单图,并且每两个顶点之间都有一条边,我们就称 G 为完全图。通常将具有 n 个顶点的完全图记为 K_n。例如图 1-5(b)就是一个完全图 K_5。

如果 G 是一个简单图,它的顶点集合 V 是由两个没有公共元素的子集 $X=\{X_1,X_2,\cdots,X_n\}$ 与 $Y=\{Y_1,Y_2,\cdots,Y_m\}$ 组成的,并且 X_i 与 X_j($1 \leqslant i,j \leqslant n$),$Y_s$ 与 Y_t($1 \leqslant s,t \leqslant m$)之间没有边连接,则 G 叫做二分图。

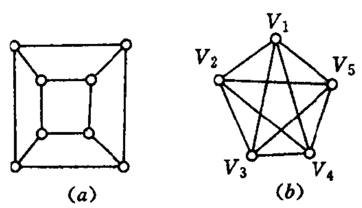


图 1-5

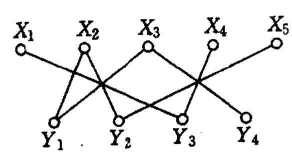


图 1-6

如果在二分图 G 中,$|X|=N$,$|Y|=M$,每一个 $X_i \in X$ 与每一个 $Y_s \in Y$ 有一条边相连,则 G 叫做完全二分图,记为 $K_{n,m}$。显然,图 1-6 为完全二分图 $K_{5,4}$。

如果 G 是一个 N 个顶点的简单图,从完全图 K_n(如图 1-5(b))中把属于 G 的边全部去掉后,得到的图称为 G 的补图,通常记为 $\overline{G}$。例如,图 1-7(a)的补图 $\overline{G}$ 为图 1-7(b)。

显然 $G=\overline{\overline{G}}$。即一个图的补图的补图就是原来的那个图。

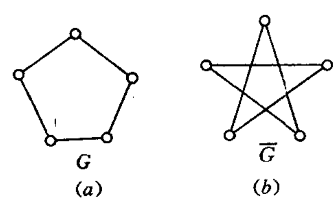


图 1-7

下面我们还要介绍一下相邻与次数这两个术语。

如果图 G 的两个顶点 V_i 与 V_j 之间有边相连,我们就说 V_i 与 V_j 是相邻的,否则就说

V_i 与 V_j 是不相邻的。如果顶点 V 是边 e 的一个端点，就说顶点 V 与边 e 是相邻的，e 是从 V 引出的边。从一个顶点 V 引出的边的条数，称为 V 的度数，记作 $d(V)$。例如图 1-5(b) 中，$d(V_1)=d(V_2)=d(V_3)=d(V_4)=d(V_5)=5-1=4$。图 1-6 中的 $d(Y_3)=2$，等等。

以后，我们把每个顶点的次数值为常数 K 的图叫做 K 度正则图。我们也经常使用下面两个符号：

$\& = \min\limits_{1\leqslant i\leqslant |V|}\{d(V_i)\}$，即所有顶点的次数的最小值为 &；

$\Delta = \max\limits_{1\leqslant i\leqslant |V|}\{d(V_i)\}$，即所有顶点的次数的最大值为 Δ。

我们可以从顶点次数问题的讨论中，引出一些有趣的结论：

1. $\sum\limits_{V_i\in V} d(V_i) = 2\times |E|$；

2. 对于任意的图 G，奇次顶点的个数一定是偶数。

这两个结论可以帮助我们分析一些问题。

【例 1】 空间是否有这样的多面体存在，它们有奇数个面，而每个面又有奇数条边？

分析：构作一个图，以面为顶点，当且仅当两个面有公共棱时，则在 G 的相应两顶点间连一条边，得到图 G。依题意，图的顶点个数是奇数，而且每个顶点的度数 $d(V)$ 是奇数，从而 $\sum\limits_{V_i\in V} d(V_i)$ 也是奇数，与结论 1 相违，故这种多面体不存在。

【例 2】 晚会上大家握手联欢，问是否会出现握过奇次手的人是奇数的情况？

分析：构作一个图，以人为顶点，两人握手时，则相应的两个顶点之间连一条边，于是每人握手的次数即相应顶点的次数。由结论 2，奇次顶点的个数总是偶数，所以握过奇次手的人数是奇数的情况不可能出现。

1.3 道路与回路

1736 年数学家欧拉(Euler 1707—1783)发表了一篇论文，解决了著名的七桥问题，这一节，我们就来谈谈七桥问题及一些有关内容。

一条河从城市穿过，河中有两个岛 A 与 D，河上有七座桥，连接这两个岛及河的两岸 B，C(图 1-8(a))。

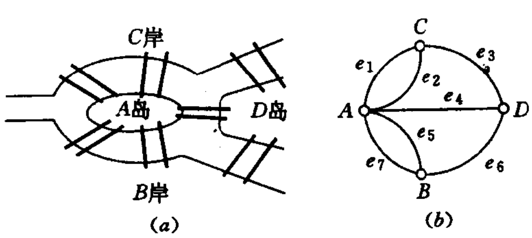


图 1-8

问：

(1) 一个旅行者能否经过每座桥恰好一次，既无重复也无遗漏？

(2) 能否经过每座桥恰好一次，并且最后能够回到原来出发点？

我们把图 1-8(a)改画成如图 1-8(b)所示的图，A,B,C,D 变成四个点，七座桥变成七条边，七桥问题就变成通常所说的一笔画问题：能否一笔画出这个图(每条边都无遗漏，也无重复地画到)？或能否一笔画出这个图、并且最后可以回到原来的出发点？

为了叙述方便及今后的需要，我们再引入几个概念：

在图 G 中，一个由不同的边组成的序列 $e_1,e_2,\cdots,e_g$，如果 e_i 是连接 V_i-1 与 $V_i(i=1,2,\cdots,g)$的，我们就称这个序列为从 V_0 到 V_g 的一条道路，数 g 称为路长，V_0 与 V_g 称为这条道路的两个端点，$V_i(1\leqslant i\leqslant g-1)$叫做道路的内点。如果 G 是简单图，这条道路也可以记作$(V_0,V_1,\cdots,V_g)$。

例如，图 1-9 中，e_1,e_2,e_3,e_4,e_5,e_6 组成一条道路。

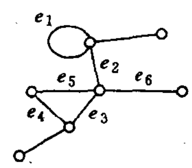


图 1-9

注意在道路的定义中，并不要求 V_0 至 V_g 互不相同。如果 V_0 至 V_g 互不相同，这样的道路称为轨道，记成 $P(V_0,V_g)$。$V_0=V_g$ 的路叫做回路。$V_0=V_g$ 的轨道叫做圈。长为 K 的圈叫做 K 阶圈。不难看出，如果有一条从 V 到 V' 的道路上去掉若干个回路，便可得到一条从 V 到 V' 的轨道。

U,V 两顶点的距离是指 U,V 间最短轨道的长度，记作为 $D(U,V)$。若 U 与 V 之间存在道路，则称 U 与 V 相连通。图 G 中任意两个顶点皆连通时，称 G 为连通图。

利用道路(回路)的概念，图能否一笔画成(并且回到原出发点)的问题，就等价于这个图是不是一条道路(回路)？

如果图 G 是一条从 V_0 到 V_g 的道路，那么该条道路上的每一个内点 $V_i(1\leqslant i\leqslant g-1)$ 都是度数为偶数的顶点。因为对 V_i 来说，有一条进入 V_i 的边，就有一条从 V_i 引出的边，而且进出的边不能重复已走过的边，所以与 V_i 相邻的边总是成双的。故图 G 至多有两个奇顶点，即 V_0 与 V_g。如果 G 是一条回路，那么根据上面推理，V_0 与 V_g 的度数也是偶数。由此，我们可以引出下面一个结论：

有限图 G 是一条道路(即可以一笔画成)的充分必要条件是 G 是连通的，且奇顶点的个数等于 0 或 2，并且当且仅当奇顶点的个数为 0 时，连通图 G 是一条回路(孤立点可以看作是回路)。

显然，由于图 1-8(b)中有 4 个奇顶点，因而不能一笔画成，即一个旅行者要既无重复也无遗漏地走过图 1-8(b)中的七座桥是不可能的。

如果你再深入地探讨一下一笔画的内涵，还可以引伸出下述一些结论，我们可以借助这些结论来分析实际问题：

1. 若连通图 G 有 $2K$ 个奇顶点，那么图 G 可以用 K 笔画成，并且至少用 K 笔才能画成。

例如，图 1-8(b)有 4 个奇顶点，该图可以用 2 笔画成：其中一种方案是：

第一笔画：$B \xrightarrow{e_6} D \xrightarrow{e_4} A \xrightarrow{e_1} C \xrightarrow{e_3} D$

第二笔画：$C \overset{e_2}{\text{————}} A \overset{e_7}{\text{————}} B \overset{e_5}{\text{————}} A$

2. 如果图 G 有两个奇顶点、K 个互相没有公共顶点的连通子图，那么图 G 可以分解成 $K-1$ 条回路和一条道路。

例如图 1-10 中有两个奇顶点 V_2 和 V_5、两个分支(V_1, V_2, V_4, V_5)和 V_3。

图 G 可以分解成：

1 条回路：(e_1, e_3, e_2)；

1 条道路：(e_8, e_5, e_6, e_7, e_4)。

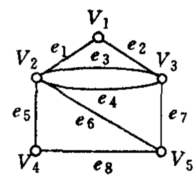


图 1-10

3. G 为二分图的充分必要条件是 G 中无奇顶点。

例如有一只老鼠在 3×3×3 的乳酪块上咬出一条洞，这个洞通过 1×1×1 的 27 个小立方体的中心，它从大立方体的一角咬起，只要还有它没尝过的小点心块，就继续向前咬。问这只老鼠能否在 3×3×3 立方体中心停止？设这只老鼠是从一个 1×1×1 的小立方体中心沿侧面正交的方向向另一未咬过的小点心块的中心咬去的。

分析：以 1×1×1 的小立方体为顶点构作一个图，把 3×3×3 立方体中心那块小立方体与开始被咬的小立方体之间连一条边，再把有公共侧面的小立方体连上边；以 8 个角上及 6 个侧面中心处的小立方体为 X 集合，其余的小立方体为 Y 集合，于是构成一个二分图。由结论 3，此图不会有 27 阶圈，所以老鼠不会停留在 3×3×3 立方体的中心。

1.4 树

【例 1】 《红楼梦》中荣国府的世系图如下(见图 1-11(a))：

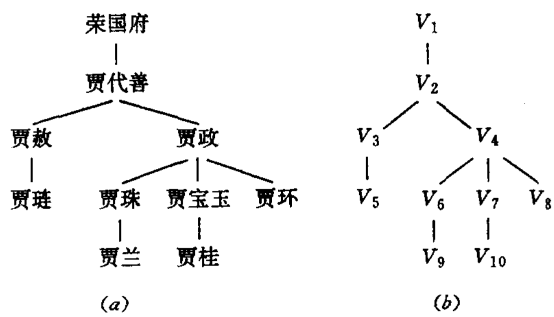


图 1-11

如果将每个人用一个顶点来表示，并且在父子之间连一条边，便得到图 1-11(b)。

这种图称为树，因为它的形状很像一棵倒悬的树。

现在我们给出树的定义：

没有圈的连通图称作树，通常用 T 表示。T 中 $d(V)=1$ 的顶点叫做叶；每个连通分支皆为树的图叫做森林，孤立的顶点叫做平凡树。

图 1-12 是一个森林，每个连通分支皆为树。

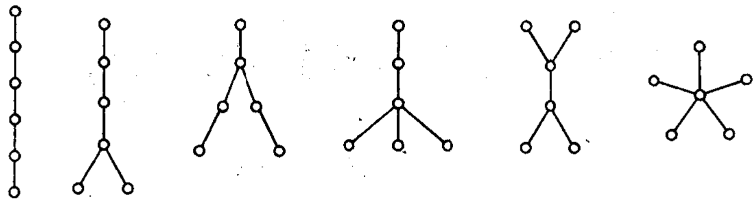


图 1-12

下面，我们通过树的顶点数与边数之间的关系，揭示树的一个图论特征。

如果树 T 的顶点数为 N，那么它的边数 $M=N-1$；倒过来，一个具有 N 个顶点、$M=N-1$ 条边的连通图 G，一定是一棵树。

树 T 具有以下性质：

1. 在 T 中去掉一边后所得的图 G 是不连通的；
2. T 添加一条边后所得的图 G 一定有圈；
3. T 的每一对顶点 V 与 V' 之间有且仅有一条轨道相连。

设 G 是一个连通图，如果 G 中有圈，我们在这个圈中去掉一条边，得到的 G' 还是连通的，如果 G' 仍然有圈，再在圈中去掉一条边得连通图 G''，……，这样继续下去，最后得到一个树 T。T 与 G 的顶点是相同的，并且从 T 陆续添加一些边就得到 G。具有这样性质的树称为连通图 G 的生成树。从 G 中删除 T 的边得到的子图称为 G 的余树。

例如图 1-13 中的粗边便构成该图的一个生成树，而细边便是余树边。可见余树可能不连通。

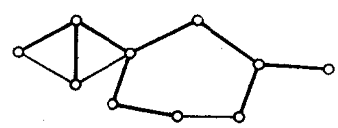


图 1-13

一个具有 n 个顶点的完全图 K_n 可以产生 N^{n-2} 个不同的树。

本章只是图论的一个开头，刚起步就冒出这么多的概念、术语和符号，以后各章节将会讲述这些概念的应用。图论的概念为数甚多。为了加强理解，读者可以参阅图论的其他书籍，但大可不必死记硬背。要多画示意图，从正反两个方面把本质的和易于误解的地方搞清楚，而且要学会用。

第二章　求最短路径的算法及应用

2.1　求最短路

一、什么是最短路问题

这一章先讲一个比较简单但又是很重要的问题——最短路问题。这个问题有着大量的生产实际的背景。事实上大至海陆空各种运输，小至一个人每天上班，都会遇到这一问题，甚至有些问题从表面上看与最短路问题没有什么关系，却也可以归结为最短路问题。下面就举一个这样的例子。

【例 1】 渡河问题。一个人带了一只狼、一只羊和一棵白菜想要过河。河上有一只独木船，每次除了人以外，只能带一样东西。另外如果人不在旁时狼就要吃羊，羊就要吃白菜。问应该怎样安排渡河，才能做到既把所有东西都带过河，在河上来回的次数又最少？

我们设变量 M 代表人，W 代表狼，S 代表羊，V 代表白菜，Φ 代表空，什么都没有。开始时设人和其它三样东西在河的左岸，这种情况用 $MWSV$ 表示。

我们用一个集合表示目前左岸的情况。很显然，可能出现的情况有 16 种：

[$MWSV$]，　[MWS]，　[MWV]，　[MSV]，
[WSV]，　[MW]，　[MS]，　[MV]，
[WS]，　[WV]，　[SV]，　[M]，
[W]，　[S]，　[V]，　[Φ]。

剔除下述 6 种可能发生狼吃羊、羊吃白菜的情况：

[WSV]，　[MW]，　[MV]，　[WS]，　[SV]，　[M]。

现在我们就来构造一个图 G，它的顶点就是剩下的 10 种情况。G 中的边是按下述原则来连的：如果经过一次渡河，情况甲可以变成情况乙，那么就在情况甲与情况乙之间连一条边（见图 2-1）。作了图 G 以后，渡河的问题就归结为下述问题了：在 G 中找一条连接顶点 $MWSV$ 与 Φ，并且包含边数最少的路。如果我们设 G 中各边的长度都是 1，那么也可以把渡河问题归结为："找一条连接 $MWSV$ 与 Φ 的最短路"。把问题转化为图论后，就可用一种系统的方法解决，而不是通常人们所用的凑的方法和凭经验的方法。

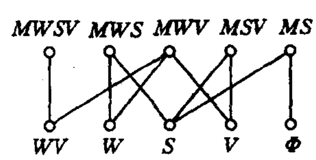


图　2-1

下面，我们可以给最短路问题下一个抽象的定义：

1. 求有向图（图中从一个顶点连到相邻顶点的边有方向性）的最短路问题

设 $G=(V,A)$ 是一个有向图，它的每一条弧 A_i 都有一个非负的长度 $L(A_i)$，在 G 中指定一个顶点 V_s，要求把从 V_s 到 G 的每一个顶点 V_j 的最短有向路找出来（或者指出不存在从 V_s 到 V_j 的有向路，即 V_s 不可达 V_j）。

2. 求无向图(图中连接两个顶点的边无方向性)的最短路问题

设 $G=[V,E]$ 是一个无向图,它的每一条边 e_i 都有一个非负长度 $L(e_i)$。在 G 中指定一个顶点 V_s,要求把从 V_s 到 G 的每一个顶点 V_j 的最短无向路找出来(或者指出不存在从 V_s 到 V_j 的无向路,即 V_s 不可达 V_j)。

二、求最短有向路的标号法

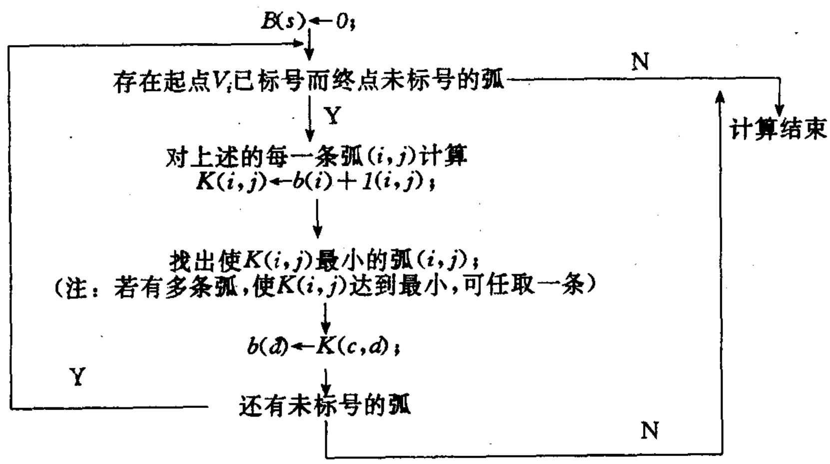
所谓标号,是指与图的每一个顶点对应的一个数字。设:

$b(j)$——顶点 V_j 的标号,代表的是 V_s 到 V_j 的最短的长度。V_j 已标号则意味着 V_s 到 V_j 的最短路以及这条路径的长度已经求出。显然初始时 $b(s)=0$。

$l(i,j)$—弧 (V_i,V_j) 的非负长度。

$K(i,j)$ —当前有向路加入弧 (V_i,V_j) 后,V_s 到 V_j 的有向路长度。

标号法的算法流程如下:



由标号法的算法流程可以看出,标号法采用顺推的方法,每边检测一次,没有重复的回溯搜索,因此是一种最佳算法。

注意几个问题:

1. 若是有向图的最短路问题,则用下述两种方法求解:

(1) 将图的每一条边修改为两条方向相反的弧,构成有向图 G'。然后直接用标号法求最短路;

(2) 将标号法略加修改。

① 各个步骤中的弧改为边;② 判断条件"存在起点 V_i 已标号而终点 V_j 未标号的弧"改为"存在一端已标号而另一端未标号的边"就可以直接在无向图上求最短路了。

2. 如果只要求 V_s 到某顶点 V_j 最短路,那么也可以在 V_j 得到标号后就结束计算。

3. 若计算结束,还有一些未标号的顶点,则肯定 V_s 到这些顶点的有向路或无向路不存在。

三、标号法程序

```
program shortest_way;
```

```
uses crt;
const  maxn = 100;
type   list = array[1..maxn] of integer;
var    chart : array[1..maxn] of list;          { 有向图的邻接矩阵 }
       mark : array[1..maxn] of boolean;
       { 标志表
       { mark[i]= {true   V1 至 Vi 的最短路径已求出}
                  {false  V1 至 Vi 的最短路径未求出 }
       b : list;
       { b[i] ——V1 到 Vi 的最短路径长度 }
       n : integer;                                { 顶点数 }
procedure init;
  var i,j,a,k : integer;
    begin;
      clrscr;
      repeat write('n='); { 输入顶点数 }
            readln(n);
    until (n>0) and (n<maxn);
    for i :=1 to n do { 邻接矩阵初始化 }
      for j :=1 to n do
        chart[i,j] :=0;
    write('number of lines :  '); { 输入边数 }
    readln(a);
    for k :=1 to a do { 输入各边的权 }
      begin
      read(i);
      read(j);
      readln(chart[i,j]);
      end;
    for i :=2 to n do  { 标志表初始化 }
      mark[i] :=false;
    mark[1] :=true;  { 从 V1 顶点出发搜索 }
    b[1] :=0;
  end;
procedure main;
  var  best,                      { 最短路径代价 }
       best_j,                    { 当前的最短路径的端点序号_d}
       i,j : integer;
    begin
      repeat best :=0;
             for i :=1 to n do
        { 从所有起点已标号、终点未标号的弧集中,选一条弧(i,best_j), }
        { 使 V1 至 best_j 的路径长度最短 }
               if mark[i] then
        { V1 至 Vi 顶点的最短路已求出,即起点已标号 }
                 for j :=1 to n do
                   if (not mark[j]) and (chart[i,j]>0) then
                     if (best=0) or (b[i]+chart[i,j]<best)
        { 存在一条终点未标号的弧(Vi,Vj)且 V1 至 Vj 的路目前最短 }
```