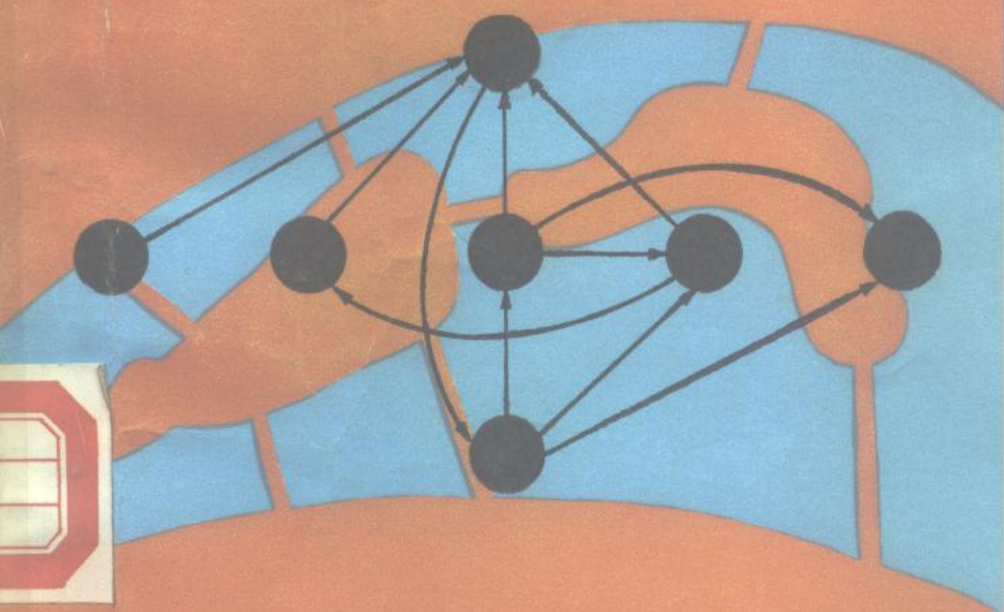
网络和图的最优化算法

[美] E·米涅卡 著

李家滢 赵关旗 译



中国铁道出版社

网络和图的最优化算法

〔美〕E·米涅卡 著

李家滢
赵关旗 译

中国铁道出版社

内 容 简 介

本书综合归纳了可以用网络或图描述的各种问题的最优化算法，是一本很有特色的关于算法的教材。

全书共有九章。第一章的内容为基本概念、定义及数学方面的预备知识，其余八章则叙述各种问题的最优化算法。书中有许多例题以帮助读者了解其基本内容，各章均附有习题供复习思考用。内容结合实际，绝大部分例题和习题均从实际生活中提出来的，读来饶有兴味。

本书可供大专院校师生和企业管理干部学习参考。

本书第一～四、八、九章李家滢译，第五～七章赵关旗译

全书由刘义生校阅

网络和图的最优化算法

〔美〕E·米涅卡著　李家滢　赵关旗　译

Optimization Algorithms for Network and Graphs

Edward Minieka

Marcel Dekker, Inc., New York and Basel 1978

中国铁道出版社出版

责任编辑　林瑞耕　封面设计　翟　达

新华书店北京发行所发行

各地新华书店经售

中国铁道出版社印刷厂印

开本：787×1092 1/32　印张：10.75　字数：246 千

1984年4月第1版　1984年4月第1次印刷

印数：0001—15,000 册　定价：1.25 元

序

本书并非图论教材而是关于算法的教材。本书叙述了可以用网络或图描述的各种问题的最优化算法。

我在写学位论文时曾研究过这些算法，觉得这些算法多种多样、很巧妙而且相互之间有联系，同时也觉得有必要把这些散见于各种杂志和专题论文的算法收集和汇总起来。在这本书中，我试图对这方面的知识作一综合性的、有条理的和清晰的叙述。

这本书可供各学科高年级大学生和初入学的研究生阅读。当然，具备运筹学和数学的基础知识将有助于理解其内容，但是并非必需具备这些基础。

本书的目的在于用直观的方法介绍各种算法的运算过程、相互依存关系及其应用。至于各种算法在高等数学中的地位，以及计算机编码的详细内容，则不是重点。

第一章包括基础知识和定义。在其余各章中，除了一种算法要应用另一种算法作为其子程序的情况而涉及另一章以外，我试图使每一章都尽可能地成为独立的部分。读者在阅读某一章时，如果不详细研究子程序算法，可无需参考其他章节。这本教材完全可以供一学期的课程使用，但是，在略去一部分内容之后，也可以供四分之一学期的课程使用。

（下略）

E·米涅卡

目　录

此为试读，需要完整PDF请访问：www.ertongbook.com

第一章　图和网络导论

第一节　导　　论

图论是数学中有广泛实际应用的一个分支。心理学、化学、电工学、运输规划、管理学、销售学以及教育学等各个不同领域内的许多问题都可以描述为图论的问题。因此，图论的意义不仅在于其本身，而且图论还是其他学科的公共基础，由此可以取得其他领域的许多成果，使其为各门学科共同享用，并能使这些成果得到扩充和展开。

和其他学科不同的是，图论的诞生日期是可以确定的。关于图的第一篇论文是瑞士数学家列昂纳德·欧拉(*Euler*，1707—1783）写出的，并于1736年由圣·彼得堡科学院发表。欧拉对于图的研究是由所谓哥尼斯堡桥问题引起的。东普鲁士的哥尼斯堡城（现在称为加里宁格勒）是建在两条河流的汇合处以及河中的两个小岛上（见图1—1）。总共有七座小桥将两个小岛以及小岛与城市的其他部分连接起来。哥尼斯堡人从其住所出发，能否恰好只经过每座小桥一次，仍返回原处？这个问题的答案是否定的。以后（第六章）在研究这个问题的一般形式即所谓“邮递员问题”时，我们就会明了。

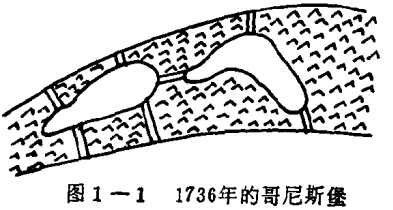


图1—1　1736年的哥尼斯堡

在十九世纪末和二十世纪初，图论因分子理论和电的理论的推动而得到进一步的发展。在二十世纪五十年代，图论

这个领域形成了两个本质上不同的发展方向，即：图论的代数方面和图论的最优化方面。后者由于计算机的问世和线性规划技巧的发现而得到很大的发展。这本书几乎近于专门论述图论的最优化方面的问题。

图是什么？图包括两个部分，即：点和连结各点的矢线。点可以画成平面上的点，也可以随意画成没有任何特定的物理位置的点。矢线可以画成连结一对点的直线或曲线。例如，在图1—2中，图的点是a，b，c，d，图的矢线是α，β，γ，δ，ε，ϕ。注意从点a至点b有两条矢线α及β，即这两条矢线的尾部都在a处而头部都在b处。

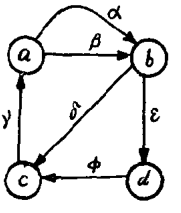


图 1—2

这同一个图也可以不用图形来表示，而只须简单地列出图的点a，b，c，d和列出图的矢线$\alpha=(a,b)$，$\beta=(a,b)$，$\gamma=(c,a)$，$\delta=(b,c)$，$\varepsilon=(b,d)$，$\phi=(d,c)$，这些就是点的有序偶。有序偶中的第一个点表示矢线尾部的点，第二个点表示矢线头部的点。按照标准术语，图的点称为**顶点**（*vertex*），矢线称为**弧**（*arc*）。由此，现在我们可以给出图的正式定义：

图（*graph*）是顶点集合X和顶点的有序偶集合A。A的元素称为弧。图用（X，A）表示。

在本书内，我们假定集合X和集合A都只包含有限个元素。

通常，顶点用小写的罗马字母表示，而弧用小写的希腊字母表示或是写为顶点的有序偶。例如，在图1—2中，弧γ可以用（c，a）表示，此处，顶点c在弧γ的尾部，顶点a在弧γ的头部。如有一条以上的弧从一个顶点走向另一个顶点，则其中的每一条弧可用有下标的顶点有序偶表示。例

如，在图 1 — 2 中，弧α可用$(a,b)_1$表示，弧β可用$(a,b)_2$表示。在不致于混淆时，则可省略下标。

例1. 令X表示依利诺斯所有机场的集合。令A表示满足下列条件的所有各对机场（x，y）的集合，即一架直达货运飞机可由机场x飞行至机场y。显然，（X，A）是一个图。

例2. 令X表示乘坐某一架越过大西洋的飞机上所有旅客的集合。令A表示满足下列条件的各对旅客（x，y）的集合，即，旅客x的年龄大于旅客y而且两人操同一种语言。显然，（X，A）是具有顶点集合X与弧集合A的一个图。问该图能否既有弧（x，y）又有弧（y，x）？

图论中的某些问题，例如我们以后将要遇到的简单匹配问题，仅需知道每条弧的端点。在这种情况下，每条弧的头部和尾部都无须规定，换句话说，即每条弧的**方向**（*direction*）无须规定。未规定弧的方向的图称为**无向图**（*undirected graph*）。无向弧称为**边**（*edge*）。例如，从图 1 — 2 中去掉箭头，剩下的图就是无向图，而弧就称为边。

本书通篇使用符号（X，E）表示顶点集合为X、边集合为E的无向图；使用符号（X，A）表示顶点集合为X、弧集合为A的图。

图中的每条弧附有一个或几个数字，这样的图就是网络（*network*）。例如，如将空中航线的英里数附注在图1—2的每条弧旁，则该图就是一个网络。

以下的每一章都只研究一种类型的图或网络在实际应用中的最优化问题。为求出这些问题的最优数值解而给出的过程称为**算法**（*algorithm*）。本书因此而命名为“网络和图的最优化算法”。由于某些最优化算法须以其他算法为基础，故本书叙述问题的先后次序是限定的，各章的编排顺序

均依此而定。

第二节　某些概念和定义

为减少各章之间的相互牵连，此处先介绍本书所有各章均需要了解的某些基本概念和定义。下面几章，在较深入地讨论应用问题时，将进一步叙述这些定义。

弧的头部和尾部均为同一个顶点时，这种弧称为**自环**（*loop*）。图 1—3 中的弧 β 就是自环。

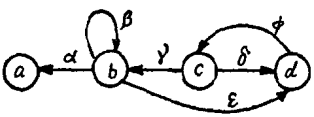


图　1—3

如某一顶点为某一条弧的一个端点（无论是头部或尾部），则该顶点与该条弧称为相互**关联**（*incident*）。在图 1—3 中，弧 α 和顶点 b 为相互关联。

如两条弧都和同一个顶点关联，则该两条弧称为相互关联。在图 1—3 中，弧 α 和弧 γ 为相互关联，因为它们都和顶点 b 关联。

如有一条弧将两个顶点连结起来，则这两个顶点称为**邻接**（*adjacent*）。在图 1—3 中，顶点 b 和顶点 c 是相互邻接的。因为弧 γ 连结着这两个顶点。

考虑任一顶点序列 $x_1, x_2, \cdots x_n, x_{n+1}$。弧 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 的任一序列称为**链**（*chain*），若弧 α_i 的两个端点是 x_i 和 x_{i+1}，$i=1,2,\cdots,n$。因此，$\alpha_i=(x_i, x_{i+1})$ 或 $\alpha_i=(x_{i+1}, x_i)$。顶点 x_1 称为链的**始点**（*initial vertex*）。顶点 x_{n+1} 称为链的**终点**（*terminal vertex*）。链被说成是从其始点延伸至其终点。链的**长度**（*length*）等于这条链中弧的数目。在图 1—3 中，弧的序列 α，β，γ，δ，ϕ 构成从顶点 a 至顶点 c 的一条链，其长度为 5。

若$a_i=(x_i,x_{i+1})$，$i=1,2,\cdots,n$，则这样的链就是**路**（*path*）。路的长度、始点和终点都可以与链一样定义。例如，在图1—3中，弧β，ε，ϕ构成从顶点b至顶点c的一条路，其长度为3。

如一条链的始点和终点为同一点，这样的链就是**圈**（*cycle*）。如一条路的始点和终点为同一点，这样的路就是**回路**（*circuit*）。圈或回路的长度定义为这条相应的链的长度。例如，在图1—3中，弧γ，ε，δ构成一个长度为3的圈，弧γ，ε，ϕ构成一个长度为3的回路。

在一条链、路、圈或回路中，如果不存在和两条以上（不包括两条）的弧相关联的顶点时，这样的链、路、圈或回路称为**简单链**（*simple chain*）、**简单路**(*simple path*)、**简单圈**（*simple cycle*）或**简单回路**（*simple circuit*）（亦即这样的链、路、圈或回路不真含圈）。例如，在图1—3中，链α，γ是简单链，而链α，β，γ则不是；圈γ，ε，δ是简单圈，但圈γ，β，ε，δ则不是。

在一个图内，如每一对顶点均有一条链将它们连结起来，这样的图就称为连通图（*connected graph*）。例如，图1—2和图1—3中的图都是连通图，而图1—4中的图则不是连通图，因为其中不存在连结顶点d和e的链。一个图可以看成是由连通图的一个集组成的，其中每一个连通图称为原图的一个分图（*component*）。图1—4中的图有两个分图（试考虑一下这两个分图是什么？）。

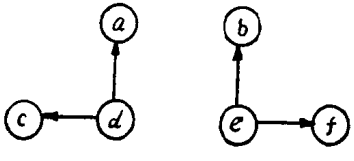


图 1—4

令X'为图$G=(X,\ A)$中顶点集合X的任一子集。有一个图，其顶点集合为X'，弧集合由A中两个端点都在X'

此为试读，需要完整PDF请访问：www.ertongbook.com

内的所有各条弧组成，这样的图称为由X'生成的子图（*subgraph generated by* X'）。

令A'为图$G=(X,A)$中弧集合A的任一子集。有一个图，其弧集合为A'，顶点集合由A'中的弧相关联的所有各个顶点组成，这样的图称为**由 A′ 生成的子图**（*subgraph generated by* A'）。例如，图 1—3 的顶点$\{a,b,c\}$生成的子图如图 1—5 所示。弧$\{\gamma,\delta,\phi\}$生成的子图如图 1—6 所示。

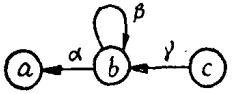


图 1—5 $\{a, b, c\}$生成的子图

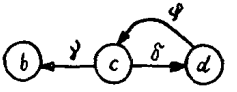


图 1—6 $\{\gamma, \delta, \phi\}$生成的子图

弧的集合如满足以下两个条件则称为**树**（*tree*）：

1. 这些弧生成一个连通图；
2. 这些弧不包含圈。

在图 1—3 中，下列每一个弧集合都构成一棵树：

$\{\alpha,\gamma,\varepsilon\}$，$\{\alpha,\gamma,\phi\}$，$\{\alpha,\varepsilon,\delta\}$，$\{\phi,\gamma\}$，$\{\alpha,\gamma\}$，$\{\varepsilon\}$，$\{\gamma\}$。

弧$\{\phi,\gamma,\varepsilon\}$不能构成一棵树，因为这些弧包含圈。

不包含圈的任一个弧集合称为森林（*forest*）。因此，森林包含一棵或几棵树。图 1—4 中的各条弧构成一个由两棵树组成的森林。

由包含图中每一个顶点的弧所构成的任一棵树称为图的**生成树**（*spanning tree*）。在图 1—3 中，弧$\{\alpha,\varepsilon,\phi\}$构成一棵生成树，因为这些弧包括所有各个顶点$a$，$b$，$c$，$d$。显然，有一个以上分图的图不存在生成树，而每一个连通图有一棵生成树。

有一条弧的树包含两个顶点，有两条弧的树包含三个顶

点；有三条弧的树包含四个顶点。一般地，有 $n-1$ 条弧的树一定包含 n 个顶点。因此，有 n 个顶点的（连通）图，它的每一棵生成树由 $n-1$ 条弧组成。

从图中移去弧的一个集合将增加图中分图的数目时，被移去的弧集合称为**截**（*cut*）。一个截如果不包含作为其真子集（*proper subset*）的另一个截，这样的截就称为**简单截**（*simple cut*）。在图 1—3 中，弧 $\{\delta,\varepsilon,\gamma,\phi\}$ 构成一个截，因为从图中移去这些弧将产生有三个分图的图。弧 $\{\delta,\varepsilon,\phi\}$ 也构成一个截，如从图中移去这些弧，将产生有两个分图的图。截 $\{\delta,\varepsilon,\gamma,\phi\}$ 包含另一个截 $\{\delta,\varepsilon,\phi\}$，所以不是简单截。$\{\delta,\varepsilon,\phi\}$ 是一个简单截。

令 G 为有 m 个顶点和 n 条弧的任一无自环的图。令 $\underline{G}$ 为有 m 行（每个顶点对应于一行）和 n 列（每条弧对应于一列）的一个矩阵。令 $\underline{G}_{ij}$ 表示第 i 行和第 j 列的元素并且定义如下：

$$\underline{G}_{ij}=\begin{cases}+1 & \text{如第 } i \text{ 行的顶点是第 } j \text{ 列的弧的头部；}\\ -1 & \text{如第 } i \text{ 行的顶点是第 } j \text{ 列的弧的尾部；}\\ 0 & \text{不属于以上两种情况。}\end{cases}$$

因此，矩阵 $\underline{G}$ 的每一列除包含 1 个 $+1$ 和一个 -1 外均为零。矩阵 $\underline{G}$ 称为图 G 的**矩阵**（*matrix of graph G*）。图 1—2 的矩阵为：

	α	β	γ	δ	ε	ϕ
a	−1	−1	+1	0	0	0
b	+1	+1	0	−1	−1	0
c	0	0	−1	+1	0	+1
d	0	0	0	0	+1	−1

显然，当且仅当矩阵的每一列除包含一个 $+1$ 和一个 -1 外

均为零时，这个矩阵就对应于一个图。

矩阵表示法无须列出顶点和弧，也无须画出图形，从而提供了一种便利的方式来描述一个图。本书叙述的最优化算法的计算机程序均使用矩阵表示法。但是，为便于介绍以后部分的内容，本书中所有各个图一律采用更为直观的图形表示法。

第三节　线 性 规 划

以后各章所讨论的许多问题都可以构造成为线性规划问题。在本节中，简单复习一下以后各章需要用到的线性规划知识。本节所述内容并不是想使读者对线性规划有一个深刻的和直观的理解，而只是向读者提供学习以后各章所必需具备的知识。关于线性规划的详细论述，读者可以参考线性规划方面的教材。

令$x_1,x_2,\cdots,x_n$为可取任一非负实值的**决定变量**(*decision variable*)。令$c_1,c_2,\cdots,c_n$，$b_1,b_2,\cdots,b_m$，a_{ij}为实数，$i=1,2,\cdots,m$；$j=1,2,\cdots,n$。

线性规划问题就是可以化为下列形式的任一问题：

$$\text{求}\max\sum_{j=1}^{j=n}c_jx_j, \tag{1}$$

约束条件：

$$\sum_{j=1}^{j=n}a_{1j}x_j\leqslant b_1$$

$$\sum_{j=1}^{j=n}a_{2j}x_j\leqslant b_2 \tag{2}$$

$$\cdots\cdots\cdots\cdots$$

$$\sum_{j=1}^{j=n} a_{mj}x_j \leqslant b_m$$

$$x_1 \geqslant 0,\ x_2 \geqslant 0,\cdots,x_n \geqslant 0 \qquad (3)$$

（2）中任一关系式中的符号“$\leqslant$”可以被符号“$\geqslant$”或“$=$”代替。（3）中任一关系式中的符号“$\geqslant$”也可以被符号“$\leqslant$”代替。这些关系式还可以完全省略，此时，决定变量将没有任何约束条件。表达式（1）的“max”可以被“min”代替。下列问题就有线性规划的形式：

求$\max(2x_1+7x_2-3x)$，

约束条件：

$$2x_1-2x_2+1x_3 \leqslant 6$$

$$4x_1-6x_2-3x_3 \geqslant 7.325$$

$$-8.25x_1+1x_2-0.3x_3 = 8$$

$x_1 \geqslant 0$，$x_2 \leqslant 0$，x_3（无限制条件）。

求极大值或极小值的表达式（1）称为**目标函数**（*objective function*）。目标函数是决定变量$x_1,x_2,\cdots,x_n$的线性组合。决定变量必须满足的m个表达式（2）称为线性规划问题的**约束条件**（*constraints*）。注意每一个约束条件的左边是决定变量的线性组合。表达式（3）称为线性规划的**非负性条件**（*nonnegativity conditions*）。

给定一个有n个决定变量和m个约束条件的线性规划问题，形式如（1），（2），（3），我们可以由此而产生另一个称为其**对偶的**（*dual*）线性规划问题。对偶线性规划问题有n个约束条件，与原始问题的各个决定变量一一对应；对偶线性规划问题有m个决定变量$y_1,y_2,\cdots,y_m$，与各个原始约束条件一一对应。线性规划问题（1），（2），（3）的对偶线性规划问题为：

求 $\min \sum_{i=1}^{i=m} b_i y_i$ (1′)

约束条件：

$$\begin{aligned}
&\sum_{i=1}^{i=m} a_{i1} y_i \geqslant c_1 \\
&\sum_{i=1}^{i=m} a_{i2} y_i \geqslant c_2 \\
&\cdots\cdots\cdots\cdots \\
&\sum_{i=1}^{i=m} a_{in} y_i \geqslant c_n
\end{aligned} \qquad (2')$$

$$y_1 \geqslant 0,\ y_2 \geqslant 0,\ \cdots,\ y_m \geqslant 0 \qquad (3')$$

如（2）中的第 i 个原始约束条件是有≥号的不等式而不是有≤号的不等式，则对偶问题（3′）中第 i 个非负性约束条件是 $y_i \leqslant 0$。如（2）中第 i 个原始约束条件是等式而不是不等式，则对偶问题（3′）中第 i 个非负性约束条件可以省略，y_i 的符号没有限制。

如第 j 个原始非负性约束条件是 $x_j \leqslant 0$，则对应的第 j 个对偶约束条件（2′）是有≤号的不等式，亦即，$\Sigma a_{ij} y_i \leqslant c_j$。如 x_j 的符号没有限制，则第 j 个对偶约束条件是等式约束条件，亦即，$\Sigma a_{ij} y_i = c_j$。

最后，如原始线性规划问题是极大化问题，则其对偶问题是极小化问题。如原始线性规划问题是极小化问题，则其对偶问题是极大化问题。

原始线性规划问题通常称为**初始问题**（*primal*）。由于对偶问题也是一个线性规划问题，因此，一个线性规划问题可以认为是一对初始一对偶线性规划问题。读者还可以很

容易地证明，对偶线性规划问题的对偶就是原来的初始问题。

上述线性规划问题的对偶是：

求$\min(-6y_1+7.325y_2+8y_3)$，

约束条件：

$2y_1+4y_2-8.25y_3\geqslant 2$

$-2y_1-6y_2+1y_3\leqslant 7$

$1y_1-3y_2-0.3y_3=-3$

$y_1\geqslant 0$，$y_2\leqslant 0$，y_3（无限制条件）。

我们怎样确定满足（2）和（3）中所有关系式的一个解$x_1,x_2,\cdots,x_n$是否能使目标函数（1）最优化呢？再者，我们怎样确定满足（2′）和（3′）中所有关系式的一个对偶解$y_1,y_2,\cdots,y_m$是否能使目标函数（1′）最优化呢？换句话说，我们怎样来确定线性规划的一个可行解是否为最优解呢？

由线性规划理论得到一个结果，即所谓**互补松弛性条件**（*complementary slackness conditions*），为这些问题提供了答案。

令x_1，x_2，…，x_n为初始决定变量的一组可行值。令y_1，y_2，…，y_m为对偶决定变量的一组可行值。于是由方程（2）和（2′）得出，初始目标函数等于：

$$\sum_{i=1}^{i=n}c_ix_i\leqslant\sum_{i=1}^{i=n}x_i\sum_{j=1}^{j=m}a_{ji}y_j$$

$$=\sum_{j=1}^{j=m}y_j\sum_{i=1}^{i=n}a_{ji}x_i\leqslant\sum_{j=1}^{j=m}y_jb_j \qquad (4)$$

$\sum_{j=1}^{j=m}y_jb_j$就是对偶目标函数。因此，初始目标函数总是

此为试读，需要完整PDF请访问：www.ertongbook.com