2.路径、树和循环

2.1 介绍

本章主要有三个目标:

1.介绍网络流和图论的基本定义

2.介绍几种不同的数据结构

3.用不同的方法转换网络流问题，通过转换可以用对任何模型开发的算法解决其他模型问题

2.2 符号与定义

有向图和网络（Directed Graphs and Networks）：有向图 G=（N，A）由一组 N 个节点和一组弧组成，弧的元素是不同节点的有序对。在这本书中，一般不区分图和网络。用 n 表示节点数，m 表示 G 中的弧数。

无向图和网络（Undirected Graphs and Networks）：定义无向图的方式与定义有向图的方式相同，只是弧是不同节点的无序对。

尾部和头部（Tails and Heads）：有向弧（i，j）有两个端点 i 和 j。节点 i 为弧（i，j）的尾部，节点 j 为其头部。

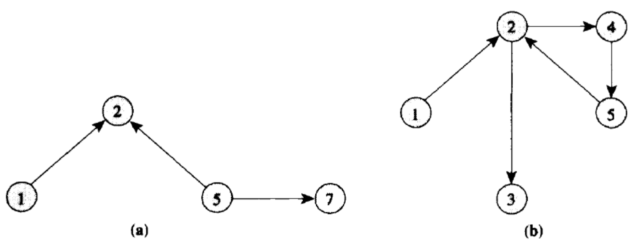
度（Degrees）：节点的 indegree 是该节点的传入弧数，outdegree 是其传出弧数。

邻接列表（Adjacency List:）：节点 i 的弧邻接列表 A（i）是从该节点发出的弧的集合

多弧和环（Multiarcs and Loops）：多弧是具有相同尾部和头部节点的两个或多个弧。环是一条尾部节点与其头部节点相同的弧。

子图（Subgraph）：如果 N'⊆N 和 A'⊆A，图 G'=（N'，A'）是 G=（N，A）的子图。

链（Walk）：一组有序的弧和节点被称为链，如图

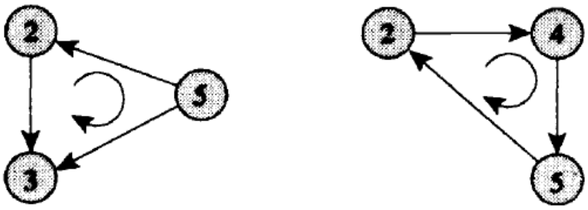


有向链：有向链是链的一个“有向”版本。上图（a）所示的链没有方向性；（b）中所示的有链方向。

路径：路径是无重复节点的链。（a）所示的路径中，弧（1，2）和（5，7）是正向弧，弧（5，2）是反向弧

有向路径：有向路径没有反向弧

循环和有向循环:



(a) (b)

无环图：如果一个图不包含有向环，它就是一个无环图。

连通性：如果图中至少包含一条从节点 i 到节点 j 的路径，则两个节点 i 和 j 是连通的。

强连通性：如果连通图每个节点都至少有一条到其他节点的有向路径，则连通图是强连通的。

切割：切割是将节点集 N 分成两部分，被切开的弧称为割

树：树是不包含循环的连通图。

树有一些重要的性质：

（a）n 个节点上的树正好包含 n-1 个弧。

（b）一棵树至少有两个叶节点（即，度为 1 的节点）。

（c）树的每两个节点通过一条唯一的路径连接。

林：不包含循环的图是林。林是树的集合。

子树：树的连通子图是子树。

有根树：有根树是有一个特别指定的节点的树，称为根；我们把一棵有根的树看作是从根上垂下来的。（特别指定的节点可以任意指定吗）

生成树：如果 T 是 G 的生成子图，T 是 G 的生成树。

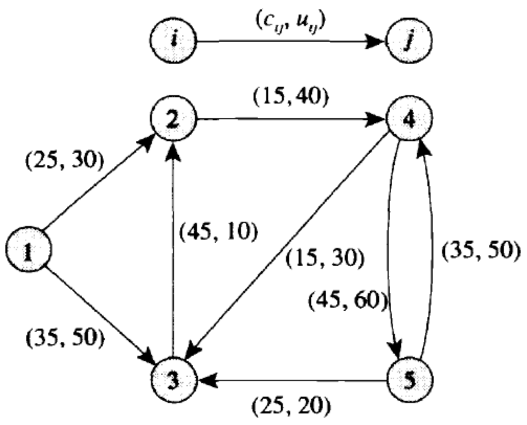
基本循环：设 T 是图 G 的生成树。将任何非树弧添加到生成树 T 中，只创建一个循环。这样的循环称为 G 关于树 T 的基本循环。由于网络包含 m-n+1 个非树弧，所以它有 m-n+1 个基本环。删除一个基本循环中的任何弧，我们再次得到一个生成树。

二分图：如果我们能把图的节点集划分成两个子集 N1 和 N2，对于 A 中的每个弧（i，j）i，j 分属 N1 和 N2，则图 G=（N，A）是一个二分图，

当且仅当 G 中的每个循环包含偶数个弧时图 G 是二部图。

2.3 网络的表示

节点弧关联矩阵：此表示法将网络存储为 n*m 矩阵，一行代表一个节点，一列代表一条弧。对应于弧（i，j）的列只有两个非零元素：它在对应于 i 的行为 1，在对应于 j 的行为 -1。



$$
\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} (1,2) & (1,3) & (2,4) & (3,2) & (4,3) & (4,5) & (5,3) & (5,4) \end{array} \\
\left[\begin{array}{cccccccc}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 1 & -1 & 0 & -1 & 0 \\
0 & 0 & -1 & 0 & 1 & 1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 1 & 1
\end{array}\right]
\end{array}
$$

Figure 2.14 Node–arc incidence matrix of the network example.

节点邻接矩阵：节点邻接矩阵表示法或简单的邻接矩阵表示法将网络存储为 nxn 矩阵 $\mathscr{H}$。矩阵一行和一列对应于每个节点，如果（i，j）∈a，h_{ij}等于 1，否则等于 0。

$$
\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5
\end{array}
\begin{array}{c}
\begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\
\left[\begin{array}{ccccc}
0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & 0
\end{array}\right]
\end{array}
$$

Figure 2.15 Node–node adjacency matrix of the network example.

邻接表：节点 i 的节点邻接列表将是具有|A（i）|个单元的链表，并且每个单元将对应于一个弧（i，j）∈A。对应于弧（i，j）的单元格将具有与我们希望存储的信息量相同的字段。一个数据字段将存储 j。我们可以使用另外两个数据字段来存储成本 Cij 和容量 Uij。每个单元格将包含一个称为链接的附加字段，该字段存储指向邻接列表中下一个单元格的指针。如果一个单元格恰好是邻接列表中的最后一个单元格，我们将其链接设置为 0。

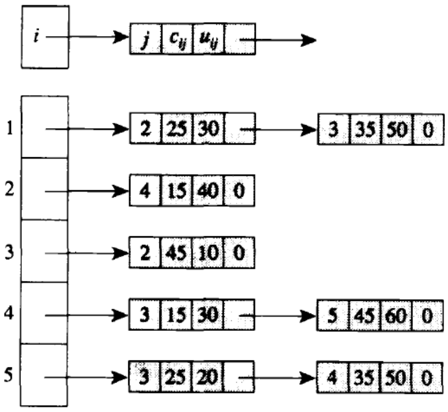


Figure 2.16 Adjacency list representation of the network example.

正向和反向星形表示法：网络的前向星形表示不是将列表作为链表来维护，而是将它们存储在单个数组中。我们首先将一个唯一的序列号与每个弧相关联，从而定义弧列表的顺序。按照特定的顺序对弧进行编号：首先是从节点 1 发出的弧，然后是从节点 2 发出的弧，依此类推。以任意方式对来自同一节点的弧进行编号。然后，在弧列表中按顺序存储关于每个弧的信息。将弧的尾部、头部、成本和容量存储在四个数组中：尾部、头部、成本和容量。为每个节点 i 维护一个指针，用 rpoint（i）表示，它表示从节点 i 发出的弧列表中编号最小的弧[如果节点 i 没有输出弧，将点（i）设置为点（i+1）。]因此，前向星表示将节点 i 的输出弧存储在弧列表中 rpoint（i）到第指针 rpoint（i+1）-1 的位置。如果 rpoint（i）>rpoint（i+1）-1，则节点 i 没有引出弧。rpoint（1）=1 和 rpoint（n+1）=m+1。

	point		tail	head	cost	capacity
1	1	1	1	2	25	30
2	3	2	1	3	35	50
3	4	3	2	4	15	40
4	5	4	3	2	45	10
5	7	5	4	3	15	30
6	9	6	4	5	45	60
		7	5	3	25	20
		8	5	4	35	50

(a)

存储平行弧：平行弧会存在一些符号上的困难，因为（i，j）不会唯一地指定弧。对于具有平行弧的网络，需要更复杂的符号来指定弧、弧成本和容量。然而，这种困难仅仅是符号化的，在计算上不存在任何问题

2.4 网络改造

无向弧到有向弧：一分为二，用两个有向弧（i，j）和（j，i）替换每个无向弧{i,j}，这两个弧都具有成本 Cij 和容量 Uij。

去除非零下界：如果弧（i，j）在弧流 Xij 上有一个非零下界 lij，则在公式中用 Xij+lij 代替 Xij。流约束则变为 lij<=xij+lij<=uij 或 0<=xij<=(uij-lij)。在质量平衡约束中进行此替换将

b（i）减少 lij 单位，并将 b（j）增加 lij 单位。可以把这种转换看作一个两步流动过程：首先在弧（i，j）上发送 lij 单位，它将 b（i）减少 lij 单位，将 b（j）增加 lij 单位，然后求解新问题的 xij‘。（我觉得这张图画错了）

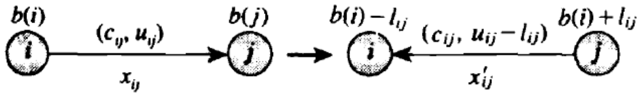


Figure 2.19 Removing nonzero lower bounds.

逆转弧：弧反转变换通常用于消除具有负成本的弧。可以理解为 i 先向 j 发送 uij 单位的流量，再求解新问题，xij=uij-xji。

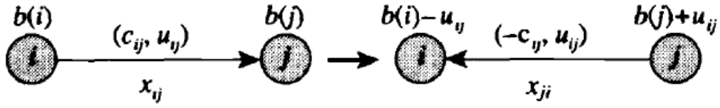


Figure 2.20 Arc reversal transformation.

消除弧容量：可以理解为引入中间节点 k，只有 i 和 j 与之相连，k 具有 uij 的需求量，且 outdegree=0，这样就能用点的需求限制取代弧的容量限制

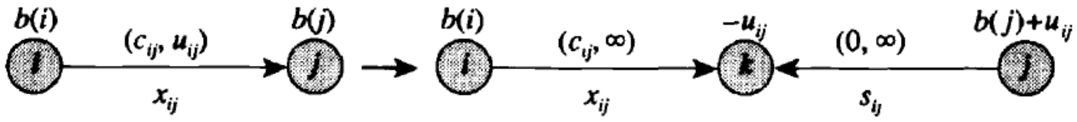
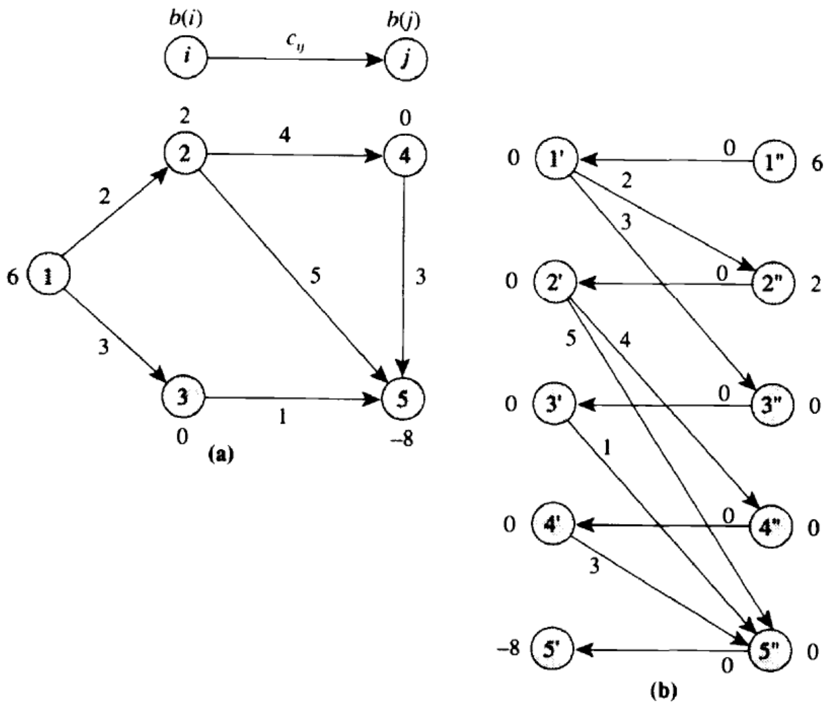


Figure 2.21 Transformation for removing an arc capacity.

节点拆分：节点拆分转换将每个节点 i 拆分为两个节点 i'和 i“，分别对应于节点的输出和输入。



这种转换的作用在于可以处理节点自身有容量或者成本的情况。

处理 reduced costs

使用剩余网络