网络流专题

清华大学
neither_nor

首先是目录

- 最大流最小割
- 费用流
- 上下界网络流

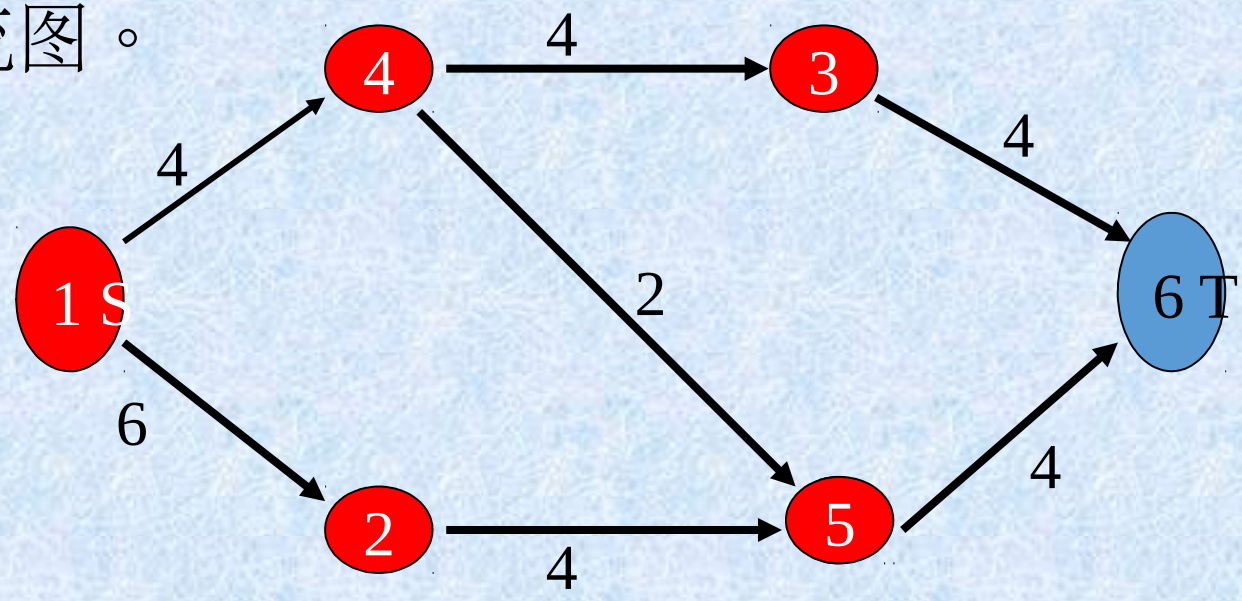
一、网络流的定义

有向图 G = (V, E) 中:

- 有唯一的一个源点 S （入度为 0： 出发点）
- 有唯一的一个汇点 T （出度为 0： 结束点）
- 图中每条弧（u, v）都有一非负容量 c (u, v)

满足上述条件的图 G 称为网络流图。

记为: G = (V, E ， C)



1、可行流

◆每条弧（u, v）上 给定一个实数 f(u,v), 满足：有 0 <= f（u, v）<= c(u, v), 则 f(u,v) 称为弧（u, v）上的流量。

◆如果有一组流量满足条件：

源点 s ： 流出量 = 整个网络的流量

汇点 t ： 流入量 = 整个网络的流量

中间点：总流入量 = 总流出量

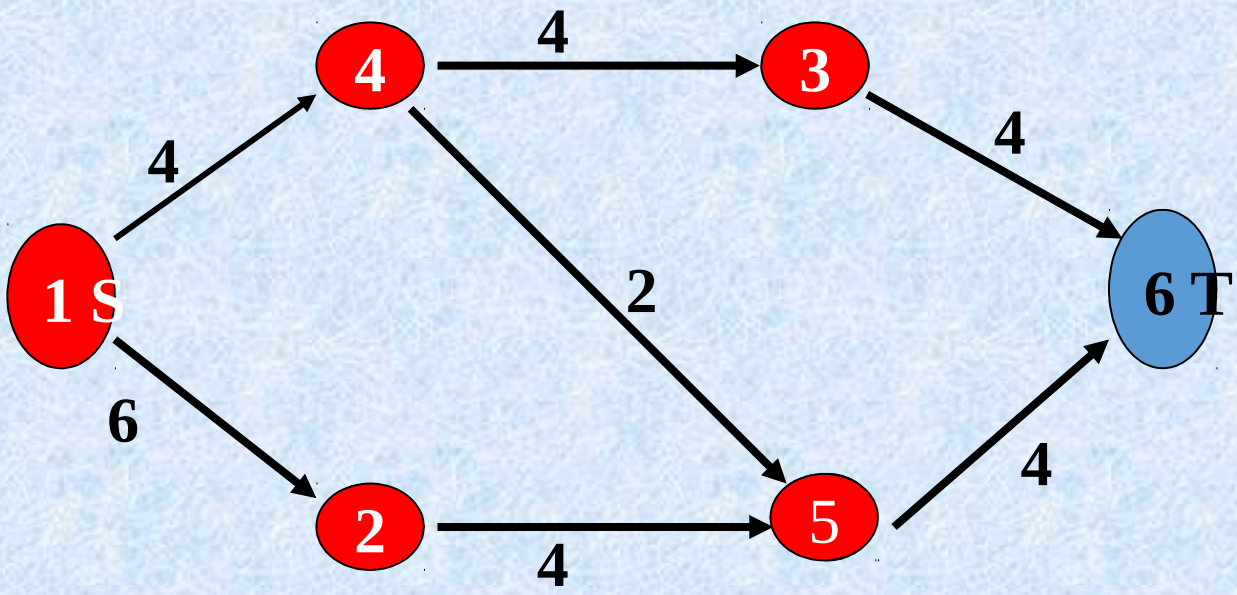
那么整个网络中的流量成为一个可行流。

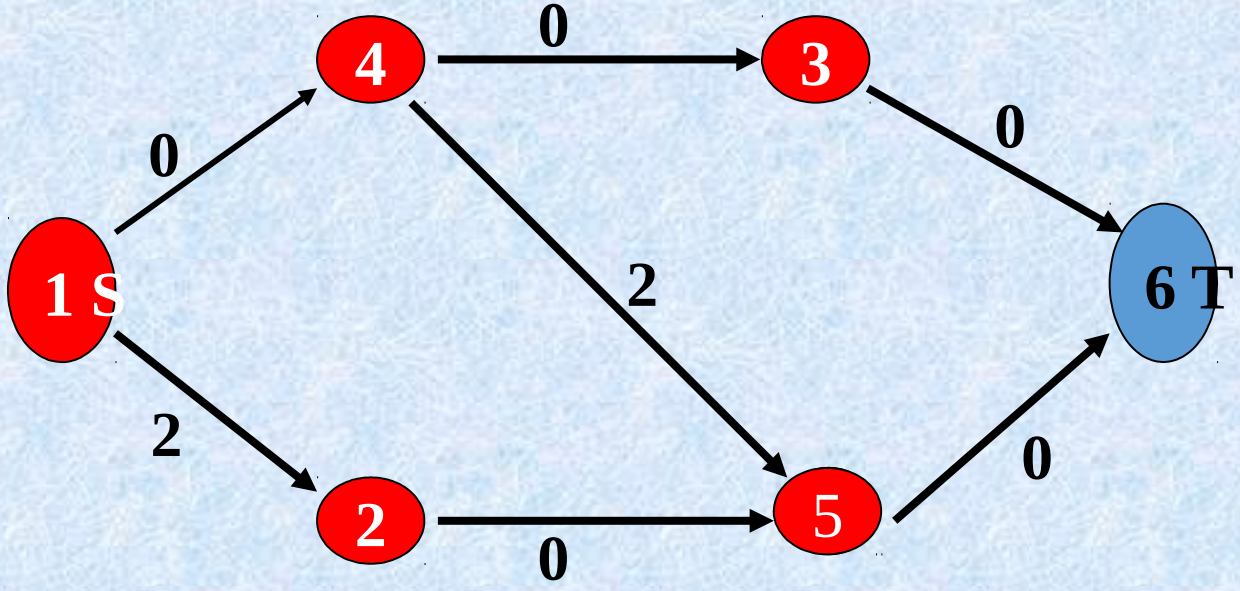
区分：容量和流量

2、最大流

- 在所有的可行流中， 流量最大的一个流的流量

最大流可能不只一个。





如何求最大流?

- Ford-Fulkerson 最大流算法（普通 DFS 算法）
- Dinic
- SAP
- 预流推进
- ……

Dinic

- 标号

```
int dfs(int x,int v)
{
    if(x==e||v==0) return v;
    int i,j,ret=0;
    for(i=pre[x];i;i=nxt[i])
    if(w[i])
    {
        j=to[i];
        if(d[j]!=d[x]+1) continue;
        int f=dfs(j,min(w[i],v));
        v-=f;
        ret+=f;
        w[i]-=f;
        w[i^1]+=f;
        if(!v) break;
    }
    if(!ret) d[x]=-1;
    return ret;
}
```

```
int dinic()
{
    int ans=0;
    while(bfs())
    ans+=dfs(s,inf);
    return ans;
}
```

- 当前弧优化
- 一些细节上的优化

最小割

割

编辑

割：设Ci为网络N中一些弧的集合，若从N中删去Ci中的所有弧能使得从源点Vs到汇点Vt的路集为空集时，称Ci为Vs和Vt间的一个割。通俗理解，一个图或网络的割，表示一个切面或切线，将图或网络分为分别包含源点和漏点的两个子集，该切线或切面与网络相交的楞或边的集合，称为图像的割。

最小割

编辑

最小割：图中所有的割中，边权值和最小的割为最小割。

最大流最小割定理

- 对于任意网络流图，从 S 到 T 的最大流 = 最小割

最直观的裸题

- 最大流： usaco-4.2.1Drainage Ditches 草地排水

- 最小割： **BZOJ1001: [BeiJing2006] 狼抓兔子（直接最小割不能过）**

[Usaco2007 Open]Dining 吃饭

- 有 n 头牛， F 种食物， G 种饮料
- 每头牛分别喜欢一些食物和一些饮料
- 每种饮料和食物都只有一个
- 问最多能让几头牛吃到自己喜欢的食物并喝到自己喜欢的饮料

- N,G,F<=100

样例

- 牛 1: 食品从 {1,2}, 饮料从 {1,2} 中选
  牛 2: 食品从 {2,3}, 饮料从 {1,2} 中选
  牛 3: 食品从 {1,3}, 饮料从 {1,2} 中选
  牛 4: 食品从 {1,3}, 饮料从 {3} 中选

- 3
  Cow 1: 不吃
  Cow 2: 食品 #2, 饮料 #2
  Cow 3: 食品 #1, 饮料 #1
  Cow 4: 食品 #3, 饮料 #3

- 如果没有饮料的存在，直接二分图匹配即可
- 二分图匹配可以用最大流来做

- 所以我们可以类比一下，想一下这道题的建图

- 将源点连向所有食物，流量为 1，代表最多能使用一次
- 同理将饮料连向所有汇点，流量为 1
- 然后在中间放上 n 头牛，用两个点来代表一头牛，中间连一条流量为 1 的边
- 每头牛向自己喜欢的饮料连一条流量为 1 的边
- 从自己喜欢的食物向他再连一条流量为 1 的边

经典的最小割模型——最大权闭合子图

有一些点，每个点有点权，点权可正可负。

对于图中的任意一条有向边 i 和 j，代表如果选择了点 i 就必须选择点 j

你需要选择一些点使得得到权值最大。

建模方式

- 先把答案加上所有的正权值

- 然后从源点向所有正权值的点连一条流量为权值的边
- 所有负权值的点向汇点连一条流量为权值相反数的边
- 原图中所有的限制关系 `(i,j)` 均建一条从 `i` 到 `j` 的流量为 `inf` 的边

- 最终答案减去这个图的最小割即可

[NOI2006] 最大获利

在前期市场调查和站址勘测之后，公司得到了一共 N 个可以作为通讯信号中转站的地址

而由于这些地址的地理位置差异，在不同的地方建造通讯中转站需要投入的成本也是不一样的

所幸在前期调查之后这些都是已知数据：建立第 i 个通讯中转站需要的成本为 Pi （ 1⩽i⩽N ）

另外公司调查得出了所有期望中的用户群，一共 M 个

关于第 i 个用户群的信息概括为 Ai, Bi 和 Ci ： 这些用户会使用中转站 Ai 和中转站 Bi 进行通讯，公司可以获益 Ci 。（ 1⩽i⩽M, 1⩽Ai, Bi⩽N ）

公司可以有选择的建立一些中转站（投入成本），为一些用户提供服务并获得收益（获益之和）。

那么如何选择最终建立的中转站才能让公司的净获利最大呢？（净获利 = 获益之和 - 投入成本之和）

N⩽5000， M⩽50000， 0⩽Ci⩽100， 0⩽Pi⩽100

样例

- 输入
- 5 5
  1 2 3 4 5
  1 2 3
  2 3 4
  1 3 3
  1 4 2
  4 5 3
- 输出
- 4

将每个用户群的信息新建一个点，权值为 `ci`，并且若想选择这个点必须选择 `ai` 和 `bi`

这就成了一个经典最大权闭合子图的模型了

直接最小割解决就可以了

另一道稍稍麻烦的题： **[NOI2009] 植物大战僵尸**

经典的最小割模型——二分图最小点权覆盖

给出一个二分图，每个点有一个非负点权

要求选出一些点构成一个覆盖，问点权最小是多少

建模方式

- 设二分图的两个点集为 A， B

- 从源点向 A 集合中的点连一条流量为对应点权的边
- B 集合中的点向汇点连一条流量为对应点权的边
- 原图中所有的边 (Ai,Bj) 均建一条从 Ai 到 Bj 的流量为 inf 的边

- 最终答案就是最小割

游戏

小明和小刚正在玩如下的游戏。首先小明画一个有 N 个顶点， M 条边的有向图。然后小刚试着摧毁它。在一次操作中他可以找到图中的一个点，并且删除它所有的入边或所有的出边。

小明给每个点定义了两个值： Wi+ 和 Wi- 。如果小刚删除了第 i 个点所有的入边他要给小明付 Wi+ 元，如果他删除了所有的出边就需要给小明付 Wi 元。

找到小刚删除图中所有边需要的最小花费。

经典的最小割模型——二分图最大点权独立集

给出一个二分图，每个点有一个非负点权

要求选出一些点构成一个独立集，问点权最大是多少

建模方式

- 先设答案为所有的权值之和
- 设二分图的两个点集为 A， B

- 从源点向 A 集合中的点连一条流量为对应点权的边
- B 集合中的点向汇点连一条流量为对应点权的边
- 原图中所有的边 (Ai,Bj) 均建一条从 Ai 到 Bj 的流量为 inf 的边

- 最终答案减去这个图的最小割即可

方格取数问题

在一个有 m*n 个方格的棋盘中

每个方格中有一个正整数

现要从方格中取数，使任意 2 个数所在方格没有公共边

求取出的数的总和最大是多少。

将棋盘黑白染色形成一个二分图

就成了一个经典的模型了

直接最小割解决就可以了

- 最大流最小割
- 费用流
- 上下界网络流

什么是费用流

如果一个网络的边不仅有容量，还有单位流量费用的话，那我们自然想在求得最大流的同时，使总费用更低。因此，最小费用最大流实际上是一个更通用的模型，因而其应用也更广。

除了最小费用最大流之外，费用流实际还包含：
最大费用最大流
最小费用流
最大费用流
(*) 正费用最大流
(*) 负费用最大流

如何求费用流

连续最短路算法

消圈算法

zkw 费用流

....

连续最短路算法

- 用贪心的思想
- 从源点开始每次寻找总费用最小的增广路
- 直到不能增广为止

- ① 连续最短路算法可以保证当前求出的方案永远是当前流量下的最优费用
- ② 因为反向弧的存在，所以这个算法最后一定会求出最大流

直观体会： [SDOI2009] 晨跑

- 给一个 n 个点 m 条边的带权有向图
- 要求找出尽量多的从 1 到 N 的路径
- 使得任意两条路径除了起点和终点不经过相同的点
- 在路径条数最多的情况下，所经过的边的总费用最小

- $N \leqslant 200$ ， $M \leqslant 20000$

- 如果没有费用限制，显然是一个最大流问题
- 把每个点**拆点**然后连一条流量为 1 的边即可
- 加上费用跑最小费用最大流即可

经典的费用流模型——

二分图最大点权完美匹配
二分图最大点权匹配
二分图最小点权完美匹配
二分图最小点权匹配

给出一个二分图，每个点有一个点权 (可以负数)

要求选出一个点权最 x 的 xx 匹配

建模方式

- 设二分图的两个点集为 A， B

- 从源点向 A 集合中的点连一条流量 1，费用为对应点权的边
- B 集合中的点向汇点连一条流量为 1，费用为对应点权的边
- 原图中所有的边 (Ai,Bj) 均建一条从 Ai 到 Bj 的流量为 1(其实只要不是 0 就行)，费用为 0 的边

- 跑一边最大费用最大流

经典的费用流模型——连续 M 个元素最多选 K 个

给你 N 个元素，每个元素有一个权值 ai(可正可负)，要求从中你选出一些，满足对于任意相邻的 M 个元素当中至多只有 K 个元素被选出，使得总权值最大

建模方式

- “N 个元素中任意相邻的 M 个元素当中至多只有 K 个元素被选出”
- 这个条件其实相当于“每次从 N 个元素选出一些元素，其中任意两个元素距离至少为 M，一共选 K 次”

- 于是我们可以把这 N 个元素顺次相连，流量为 inf（比 K 大就行），费用为 0，走一次 i->i+1 的边就代表 i 这个元素这次不选
- 然后对于 i<=N-M，我们从 i 向 i+M（其余向 T 连）连一条流量为 1，费用为 ai，走了这条边就相当于选择了 i 这个元素，并跳过中间的
- S 向 1 连流量为 K，费用为 0 的，代表选 K 次
- 跑一边最大费用最大流

[Sdoi2016] 数字配对

- 有 n 种数字，第 i 种数字是 ai、有 bi 个，权值是 ci。
- 若两个数字 ai、aj 满足，ai 是 aj 的倍数，且 ai/aj 是一个质数，
- 那么这两个数字可以配对，并获得 ci×cj 的价值。
- 一个数字只能参与一次配对，可以不参与配对。
- 在获得的价值总和不小于 0 的前提下，求最多进行多少次配对。
- n⩽200，ai⩽10^9，bi⩽10^5，|ci|⩽10^5

样例

- 3
  2 4 8
  2 200 7
  -1 -2 1

- 4

- 把所有的数字按质因子次数之和的奇偶性拆成二分图
- 按照费用流模式建图
- 由于费用流可以保证当前的费用永远是当前流量下最优的
- 所以跑到费用总和小于 0 时停止即可

方格取数问题

在一个有 m*n 个方格的棋盘中
每个方格中有一个正整数
现要从在方格中从左上角到右下角取数，只能向右或向下走
每走到一个格子就可以把这个位置上的数取走（下次经过就没有了）

让你走 1 次，求取出的数的总和最大是多少
让你走 2 次，求取出的数的总和最大是多少
让你走 k 次，求取出的数的总和最大是多少

- 把每个格子拆点，连一条费用为权值，流量为 1 的边，代表取走
- 再连一条费用为 0 ，流量为 inf 的边，代表路过

- 从左上到右下按顺序建出一个费用为 0 ，流量为 inf 的网格图

- 超级源点向左上角的格子连一条流量为 k 的边

- 跑最大费用最大流

[SCOI2007] 修车

- 同一时刻有 N 位车主带着他们的爱车来到了汽车维修中心。
- 维修中心共有 M 位技术人员，不同的技术人员对不同的车进行维修所用的时间是不同的。
- 现在需要安排这 M 位技术人员所维修的车及顺序，使得顾客平均等待的时间最小。
- 说明：顾客的等待时间是指从他把车送至维修中心到维修完毕所用的时间。

- 2<=M<=9,1<=N<=60, 1<= 修每辆车的时间 <=1000

样例

- 2 2
  3 2
  1 4

- 1.50

- 把每个工作人员拆成 n 个点
- 第 i 个点代表这个人修的车中倒数第 i 辆
- 然后将每辆车向这些点连边，流量为 1 费用为 F(i,j)*i
- 源点连向车，工作人员连向汇点

[Sdoi2013] 费用流

- 对于一张给定的运输网络， Alice 先确定一个最大流，如果有多种解， Alice 可以任选一种；之后 Bob 在每条边上分配单位花费（单位花费必须是非负实数），要求所有边的单位花费之和等于 P 。总费用等于每一条边的实际流量乘以该边的单位花费。
- 需要注意到， Bob 在分配单位花费之前，已经知道 Alice 所给出的最大流方案。现在 Alice 希望总费用尽量小，而 Bob 希望总费用尽量大。
- 我们想知道，如果两个人都执行最优策略，最大流的值和总费用分别为多少。
- N < = 100 ， M < = 1000 ，流量 <=50000,P<=10

样例

- 3 2 1
  1 2 10
  2 3 15

- 10
  10.0000

- 当 Alice 给出方案之后， Bob 一定会选择流量最大的一条边分配上 P 的费用
- 所以 Alice 一定要让流量最大的边流量最小
- 显然需要二分来解决
- 每次让每条边的流量上限与二分值取 min
- 判定一下这样能否流出最大流即可

- 注意小数网络流的精度问题

BZOJ3438: 小 M 的作物

- 小 M 在 MC 里开辟了两块巨大的耕地 A 和 B （你可以认为容量是无穷），现在，小 P 有 n 中作物的种子，每种作物的种子有 1 个（就是可以种一棵作物）（用 1...n 编号），现在，第 i 种作物种植在 A 中种植可以获得 ai 的收益，在 B 中种植可以获得 bi 的收益，而且，现在还有这么一种神奇的现象，就是某些作物共同种在一块耕地中可以获得额外的收益，小 M 找到了规则中共有 m 种作物组合，第 i 个组合中的作物共同种在 A 中可以获得 c1i 的额外收益，共同总在 B 中可以获得 c2i 的额外收益，所以，小 M 很快的算出了种植的最大收益，但是他想要考考你，你能回答他这个问题么？

- 1<=k< n<= 1000,0 < m < = 1000 保证所有数据及结果不超过 2*10^9 。

样例

- 3
  4 2 1
  2 3 2
  1
  2 3 2 1 2

- 11
- A 耕地种 1 ， 2 ， B 耕地种 3 ，收益 4+2+3+2=11 。

- 源点向所有作物连边，流量为 ai
- 所有作物向汇点连边，流量为 bi
- 对于每组作物，新建两个点
- 源点向点 1 连一条流量为 c1i 的边，点 2 向汇点连一条流量为 c2i 的边
- 点 1 点 2 分别与组内作物连流量为 inf 的边
- 对于一个割，每个作物与源点联通代表种在 A 地，否则种在 B 地，需要割去另一部分的收益
- 如果组内的点不都与源点联通，则 c1i 必被割掉
- c2i 亦是如此

- 所以答案为总收益 - 最小割

上下界网络流

- 每条边除了有流量上界外，还有一个流量下界
- 由于这些条件的产生，随之而来的一系列问题还有求解“可行流”，“最小流”

- 求解上下界网络流一般有两种方法：
- ① 强行费用流法
- ② 传统超级源汇法

强行费用流法

- 对于每条边 (from,to,low,high) ，我们改建成两条边
- (from,to,high-low,0)
- (from,to,low,-inf)
- 这样最小费用流会优先走下面这种边，我们可以通过最后跑出的总费用来判断是不是把所有这样的边都流满了，并灵活的进行改进来求出最小流或最大流

- 缺点：负环

传统超级源汇法

- 新建超级源汇 SS ， TT
- 对于每条边 (from,to,low,high) ，我们改建成三条边
- (from,to,high-low)
- (S,to,low)
- (from,E,low)
- 对于原来的 S,T ，建一条 (T,S,inf) 的边
- 这样我们跑一遍从 SS 到 TT 的最大流，如果所有 SS 的出边全部满流，则当前残量网络去掉 SS ， TT 之后就是一个可行解的残量网络
- 在这个网络上进行 S,T 之间的退 (进) 流，即可得到最小流和最大流

更多例题

- **BZOJ4554**
- **BZOJ3158**
- **BZOJ4501**
- **BZOJ4873**
- **BZOJ3511**
- **BZOJ3275**
- **BZOJ1189**
- **BZOJ2400**
- **BZOJ4443**
- **BZOJ2502**