网络流

清华大学 茹逸中

目录

1. 网络流的基本概念

定义

- 有向图 G=(V,E) ，源点 S ，汇点 T 。边权 C 表示边的容量。
- 流量总是从 S 点流出，经过一些边后到达 T 。
- 反向弧：“流量反悔”机制。当一条边有流量 f 通过时，建立一条反向的边，该反向边的容量为 f ，表示这条边通过的流量是可以流回去的。
- 残量网络：在网络中通过了一些流量后，每条边还剩余的容量组成的网络。残量网络还包括反向弧
- 所有网络流基本算法都是基于残量网络的。

1. 网络流的基本概念

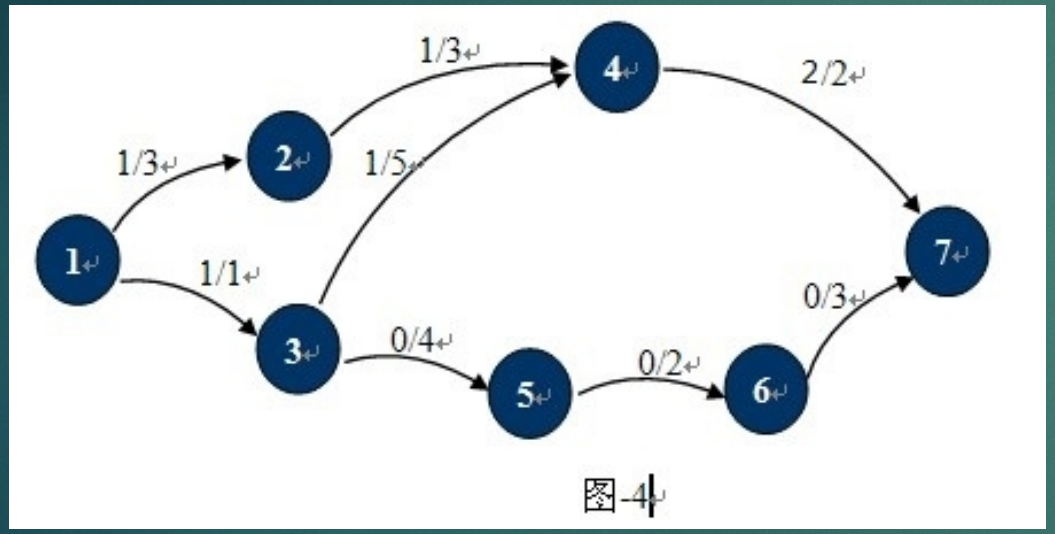


图-4

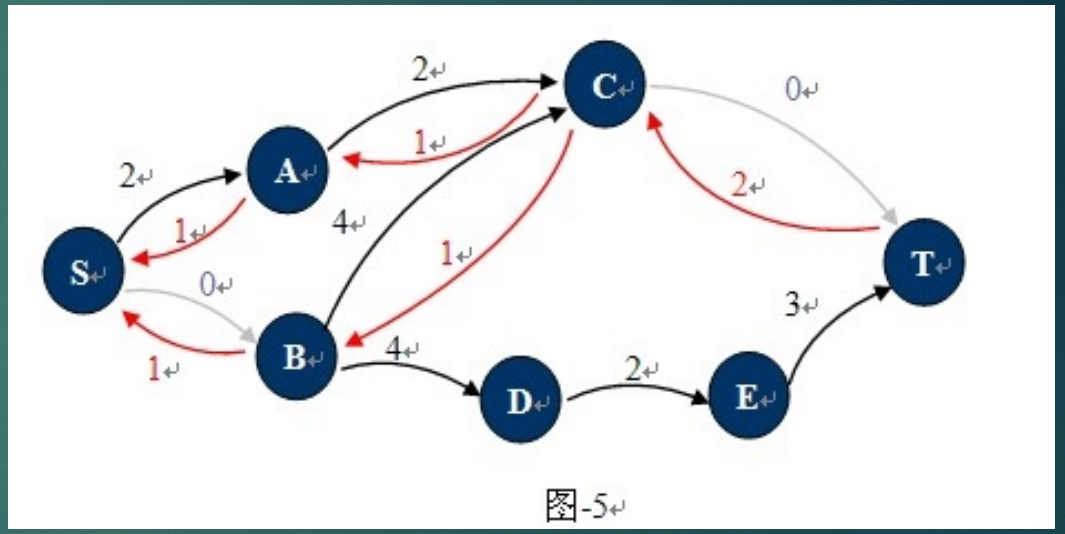


图-5

2.1 dinic 算法

- 求最大流有很多算法，例如 dinic、 E-K、预流推进等。这些算法各有优劣，但在一般情况下，只要实现得较好，这些算法都能通过竞赛中网络流题目的全部数据。因此我们只要掌握其中的一种算法即可。在这里我们介绍 dinic 算法。
- 我们先考虑朴素的最大流算法，即每次寻找一条流的通路。但是这种算法存在一个致命的缺陷，流没有方向性，有可能会陷入循环当中。
- Dinic 算法为了解决这个缺陷，采用了分层图的思想。即将图分为若干层，流只能从较高层次的点流入较低层次的点。分层的方法很简单，是基于贪心的思想。
- 分层结束后，则寻找在当前分层结果下的最大流。需要注意的是网络流本身没有分层的条件，因此在分层约束下的最大流不是整个网络的最大流。在求得当前分层条件下的最大流之后还要继续求新的分层。

2.1 dinic 算法

- 分层 (bfs) ：
  1. 将 T 的标号置为 0 ，加入队尾
  2. 取出队首元素 u ，设 u 的标号为 d[u] 。对于所有的 v ，在残量网络中 <v,u> 大于 0 ，即残量网络中的流量能从 v 流到 u ，则将 v 的标号置为 d[u]+1 ，并将 v 加入队尾
  3. 重复 2 ，直到队列为空
  4. 若 S 有标号，则表明在残量网络中从 S 出发至少有一条通路到 T ，可以继续执行寻找流通路算法
  5. 若 S 没有标号，则表明在残量网络中从 S 出发已经找不到到 T 的通路了，算法结束。

2.1 dinic 算法

- 寻找流通路 (dfs) ：

函数 dfs(u, limit) ， u 表示当前节点编号， limit 表示从 S 到 u 的通路上最多通过的流量。返回值表示从 u 到 T 最多可以通过的流量。局部变量 flow 用来记录 u 到 T 最多可以通过的流量，最后返回。

1. 若 u = T ，则返回 limit
2. 对于残量网络中的每条边 <u,v> ，若 d[u] > d[v] ，则
   - 令 flowV = dfs(v, min(limit – flow, <u,v>))
   - Flow+=flowV
   - 维护残量网络 <u,v>-=flowV, <v,u>+=flowV
3. 返回 flow

2.1 dinic 算法

- 寻找流通路 (dfs) ：

函数 dfs(u, limit) ， u 表示当前节点编号， limit 表示从 S 到 u 的通路上最多通过的流量。返回值表示从 u 到 T 最多可以通过的流量。局部变量 flow 用来记录 u 到 T 最多可以通过的流量，最后返回。

1. 若 u = T ，则返回 limit
2. 对于残量网络中的每条边 <u,v> ，若 d[u] > d[v] ，则
   - 令 flowV = dfs(v, min(limit – flow, <u,v>))
   - Flow+=flowV
   - 维护残量网络 <u,v>-=flowV, <v,u>+=flowV
3. 返回 flow

2.2 当前弧优化

基本思想：在一次 dfs 中，考虑从 u 点出发的某条边 <u,v> ，若在 dfs 时传入的参数 limit 大于其返回值，可以断言这条路已经被流满了，因此下次搜索到这个点时不必考虑这条边。

建议使用邻接表实现残量网络，这样只要修改 last 数组就可以删除最后的几条没有用的边了。

2.3 例题 Ditches(HDU1532)

直接给出网络求最大流

裸题，可做算法练习用。

2.3 例题 Strong Kings(POJ2699)

- N 个人进行循环赛，每场比赛赢者得一分。
- 如果一个人打败了所有得分比他高的人，那么他就是 StrongKing
- 给定每个人的得分，求最多的 StrongKing 的数量

- 由于只要求数量最多，因此我们只需要枚举数量，然后判断可不可行即可
- 又考虑到得分高的人更容易成为 StrongKing（需要打败的人更少，打败了的人更多）
- 所以枚举后我们价格得分前 k 高的人是 StrongKing ，然后判断可行性。

2.3 例题 Strong Kings(POJ2699)

- 如何判断可行性？
- 网络流！
- N 个人和 N(N-1)/2 场比赛分别作为网络中的点，并额外引入源点和汇点。
- 源点向所有人连边，边权为得分
- 比赛向汇点连边，边权为 1
- 对于结果确定的比赛，只将胜者与其连边，边权为 1
- 对于结果不确定的比赛，将双方都与其连边，边权为 1
- 如果最大流 =N(N-1)/2 则可行，否则不可行

2.3 例题 PIGS(POJ1149)

- 有 n 个猪圈，每个猪圈中有若干头猪。
- 有 m 个顾客按顺序来买猪，每个顾客需要若干头猪。每个顾客都有某几个猪圈的钥匙，猪圈打开后可以任意分配这些猪。
- 问最多能卖掉几头猪。

- 这题的难点主要在于顾客可以分配猪圈里的猪
- 注意到顾客是按顺序来的
- 如果后来的顾客 y 与先来的顾客 x 有同一个猪圈的钥匙，则 x 可以将他能打开的别的猪圈里的猪给 y 。

2.3 例题 PIGS(POJ1149)

- 构图如下：
- 每个顾客为一个点，额外引入源点和汇点。
- 若某个顾客是拥有某个猪圈钥匙的第一个顾客，则从源点向其连一条容量为该猪圈初始猪的数量的边。
- 若两个顾客同时拥有某个猪圈的钥匙，假设 x 先来， y 后来，则从 x 向 y 连一条容量无穷大的边。
- 上一条也可优化成对于每个猪圈只连接相邻两个人

3.1 最小割

- 什么是最小割？
- 给定网络 G ，选择一些边，删去后使得 S 与 T 不连通，则称这些边组成的集合为一个割。边权和最小的割称为最小割。
- 对于一个割，将网络 G 划分为两部分。一部分与源点相连通，称为源集；另一部分与汇点相连通，称为汇集
- 最大流 = 最小割的证明：
- 对于一个流 f ，从源点 S 出发，在其残量网络中可以走到的所有点中必然没有 T ，可以组成一个割。
- 显然不存在比最大流还小的割

3.2 例题 狼和羊的故事 (BZOJ1412)

- 在 n*m 的矩阵格子中有一些格子是狼的领地，还有一些格子是羊的领地，在两个相邻的格子之间可以建设篱笆，要求用最少长度的篱笆把羊的领地和狼的领地分隔开
- 最小割转化为最大流问题
- 源点表示羊，汇点表示狼
- 从源点向所有羊的领地连一条边，边权为无穷大，表示羊的领地必然属于源集
- 向所有狼的领地向汇点连一条边，边权为无穷大，表示狼的领地必然属于割集
- 相邻两个格子之间连一条边，边权为 1 ，表示相邻两个格子若属于不同集合，则需要建立长度为 1 的篱笆

3.2 例题 最大获利 (BZOJ1497)

N 个可以作为通讯信号中转站的地址，建立第 i 个通讯中转站需要的成本为 Pi。一共有 M 个用户，第 i 个用户会用 Ai,Bi 这两个中转站进行通讯，如果满足他们的要求他们会支付 Ci。求最大获利。

- 是一道线性规划的题目，但实际上可以用最小割的模型建模。
- 希望能在不建立中转站的情况下让所有用户付钱，但显然不可能。
- 把所有中转站和用户都看作是点。引入源点和汇点，在源集中的用户是付费的用户，需要满足他的中转站条件，因此在源集中的中转站是需要建立的中转站，需要付出一定的成本。

3.2 例题 最大获利 (BZOJ1497)

- 如果一个用户在汇集中，则会减少 Ci 的收入，所以从源向该点连一条容量为 Ci 的边。
- 如果一个中转站在源集中，则会增加 Pi 的成本，所以从该点到汇连一条容量为 Pi 的边。
- 对于每一个用户，他需要用 Ai 和 Bi 中转站，因此该用户和这两个中转站应属于同一集合，即从该用户向这两个中转站连一条容量为无穷大的边。

3.2 例题 happiness

高一一班的座位表是个 n*m 的矩阵，经过一个学期的相处，每个同学和前后左右相邻的同学互相成为了好朋友。

这学期要分文理科了，每个同学对于选择文科与理科有着自己的喜悦值，而一对好朋友如果能同时选文科或者理科，那么他们又将收获一些喜悦值（同时选文和同时选理不一样）。

如何分配可以使得全班的喜悦值总和最大。

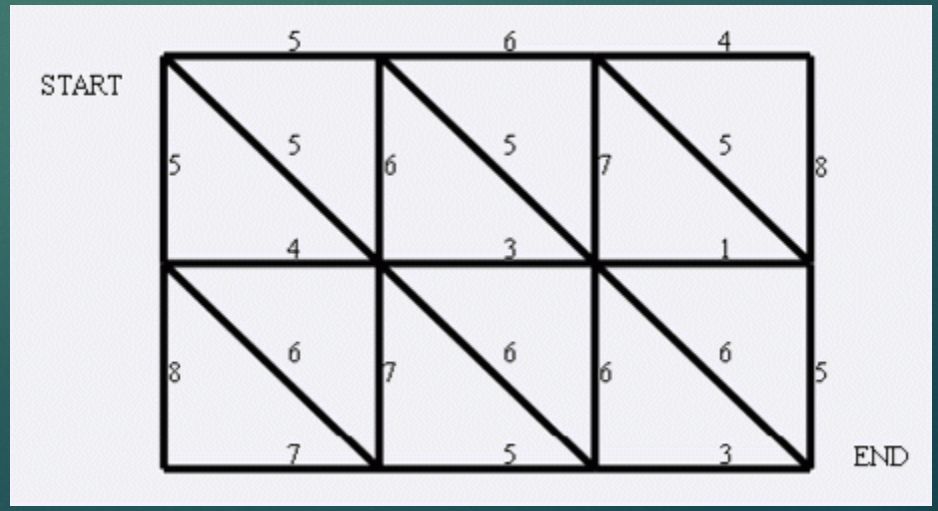
- 先考虑同时选文和同时选理喜悦值一样的情况
- 源点表示文科，汇点表示理科。先让所有人同时获得选文理科的喜悦值和与相邻的人选同样科目的喜悦值，若某个人属于源集，则去掉他选理科的喜悦值，属于汇集则去掉选文科的喜悦值
- 两个人相邻再连一条边表示若两个人属于不同集合则去掉选同样科目的喜悦值

3.2 例题 happiness

- 如何处理同时选文和同时选理不一样的情况？
- 将同时选文和同时选理的喜悦值各分配一般到两个人选文和选理的喜悦值中。
- 若两个人不同科目，则必然是一文一理，两个人之间再连一条边为同时选文和同时选理喜悦值的一半

3.3 平面图最小割 狼抓兔子 (BZOJ1001)

- 左上角点为 (1,1), 右下角点为 (N,M)(上图中 N=4,M=5). 有以下三种类型的道路 1:(x,y)<==>(x+1,y) 2:(x,y)<==>(x,y+1) 3:(x,y)<==>(x+1,y+1) 道路上的权值表示这条路上最多能够通过的兔子数，道路是无向的．左上角和右下角为兔子的两个窝，开始时所有的兔子都聚集在左上角 (1,1) 的窝里，现在它们要跑到右下解 (N,M) 的窝中去，狼王开始伏击这些兔子．当然为了保险起见，如果一条道路上最多通过的兔子数为 K ，狼王需要安排同样数量的 K 只狼，才能完全封锁这条道路，你需要帮助狼王安排一个伏击方案，使得在将兔子一网打尽的前提下，参与的狼的数量要最小。因为狼还要去找喜羊羊麻烦．



3.3 平面图最小割

- 求平面图的对偶图
- 原图的一个割对应对偶图的一个环
- 求最小环即可

- 对偶图：
- 面对偶成点
- 两个面相邻则在两个对应的点之间连边

4.1 费用流

定义：

在网络 G 中，每条边不仅有容量上限，而且每通过 1 单位流量还需要一定的费用。一般需要求最小费用最大流或者最小费用流（费用可能是负数的情况）。

贪心算法：

仍然基于残量网络，若某条边的费用为 P ，则其反向弧的费用为 -P 。

每次在残量网络中寻找费用最小的增流路径，一般使用 spfa 算法。直到找不到为止。

定理：用这种算法找到的增流路径的费用是非严格递增的。

4.1 例题 网络扩容 (BZOJ1834)

网络流 + 费用流的裸题，建议作练习用

4.1 例题 修车 (BZOJ1070)

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。

- 要使平均等待时间最小即要使总等待时间最小
- 考虑每个工人修某一辆车的影响，若他之后还需要修 k 辆车，则需要让 (k+1) 个人等待 t 的时间 (t 为修当前车所需要的时间)
- 将工人拆点成 n 个点，第一个点表示这个工人修最后一辆车，第二个点表示修倒数第二辆...
- 接下来就是比较显然的费用流建图了

4.2 有上下界的网络流

构图转化为最大流模型。

谢谢！