网络流

zzq

什么是最大流？

4.2 最大流问题

在最大流问题中，除源汇外的每一个节点都需要满足“流量守恒”，即流入流量等于流出流量。每一条边需要满足流量不超过容量。用$c(u,v)$表示(u,v)这条边的容量，$f(u,v)$表示流量。为了使问题更加简便，我们可以人为增加一条从汇到源的容量为∞的边，这样使得流量守恒对于每一个节点均成立，而且$f(t,s)$就是从源到汇的流量大小，用V表示点集，E表示边集，得到如下线性规划：

$$
\begin{aligned}
&\text{最大化} && f(t,s) && \\
&\text{满足约束} && f(u,v) \le c(u,v), && (u,v) \in E \\
& && \sum_v f(u,v) = \sum_v f(v,u), && u \in V \\
& && f(u,v) \ge 0, && (u,v) \in E \cup \{(t,s)\}
\end{aligned}
$$

什么是最小费用流？

4.3 最小费用流问题

在这个问题中，我们要求的只是最小费用，并不需要满足流量最大的限制，这时每条边有费用$w(u,v)$。故有如下线性规划：

$$\text{最小化} \quad \sum_{(u,v)\in E} f(u,v)w(u,v)$$

$$\text{满足约束} \quad f(u,v) \le c(u,v), \qquad (u,v) \in E$$

$$\sum_{v} f(u,v) - \sum_{v} f(v,u) = 0, \qquad u \in V - \{s,t\}$$

$$f(u,v) \ge 0, \qquad (u,v) \in E \cup (t,s)$$

可以发现，对于每一个变量$f(u,v)$，除了对变量本身的非负限制、容量限制之外，这个变量只出现在两个限制中，及u,v两个节点的流量守恒方程，且在这两个限制中$f(u,v)$的系数分别为$+1,-1$。

什么是最小费用流？

- 在最小费用流问题中，需要注意的是它可能和我们平常想的流不太一样，比如如果有一个孤零零的负权环，最小费用流可以直接将环流满，不需要和s、t连通。

Ford-Fulkerson

- 从s到t找一条每条边容量非0的路！
- 流min{当前容量}这么多！
- 把反向边加上！
- 复杂度 $O(Ef)$，其中f是流量。

Edmonds-Karp

- 从s到t找一条每条边容量非0的最短路！
- 流min{当前容量}这么多！
- 把反向边加上！
- 复杂度 $O(nm^2)$。

Dinic

- 将图按照从s开始的最短路分层，接下来使用dfs在这个dag上流。
- 当前弧优化。
- 时间复杂度是 $O(n^2m)$ 的。

```
int dfs(int x,int f)
{
    if(f<=0) return 0;
    if(x==T) return f;
    int ca=0;
    for(int& e=fst[x];e;e=nxt[e])
    {
        int b=vb[e];
        if(d[b]+1!=d[x]) continue;
        int w=dfs(b,(cap[e]<f-ca)?cap[e]:(f-ca));
        cap[e]-=w; cap[e^1]+=w; ca+=w;
        if(ca==f) break;
    }
    if(!ca) d[x]=-1;
    return ca;
}
```

Dinic

- Dinic的复杂度比较玄学，在大部分情况下都远远好于表面上的这一复杂度，我们举两个比较为人熟知的例子。
  - 如果容量只有1，时间复杂度不超过 $O\left(\min\left(n^{\frac{2}{3}}, m^{\frac{1}{2}}\right)m\right)$。
  - 如果除源点和汇点外，每个点只有一条容量为1的入边或只有一条容量为1的出边，那么时间复杂度不超过$O(\sqrt{n}m)$。

Successive Shortest Path

- 最小费用最大流。
- 从s到t找一条每条边容量非0的最短路，SPFA。
- 流min{当前容量}这么多！
- 把反向边加上！
- 复杂度 $O(SPFA(n,m)\times f)$，其中f是流量。

最大流最小割定理

- 最大流=最小割。
- 证明？线性规划对偶。这里就不证了。

最大流最小割定理

- 最大流大家都会输出方案，输出反向边的容量即可。如何输出最小割的方案呢？
- 在残余网络上从S开始dfs，dfs到的点就是S割集，其他点就是T割集。显然S割集里不会有T，否则说明没流满。
- 这是为什么呢？首先，将S割集到T割集的边全部割断一定是一个合法解。然后最大流=当前这个割，所以这个割就是最小割。

关于网络流的复杂度

- 网络流的复杂度比较玄学，一般情况下网络流的运行速度远远好于标识的复杂度。
- 虽然存在更优秀的网络流算法，一般情况下网络流题主要考查的是建模。

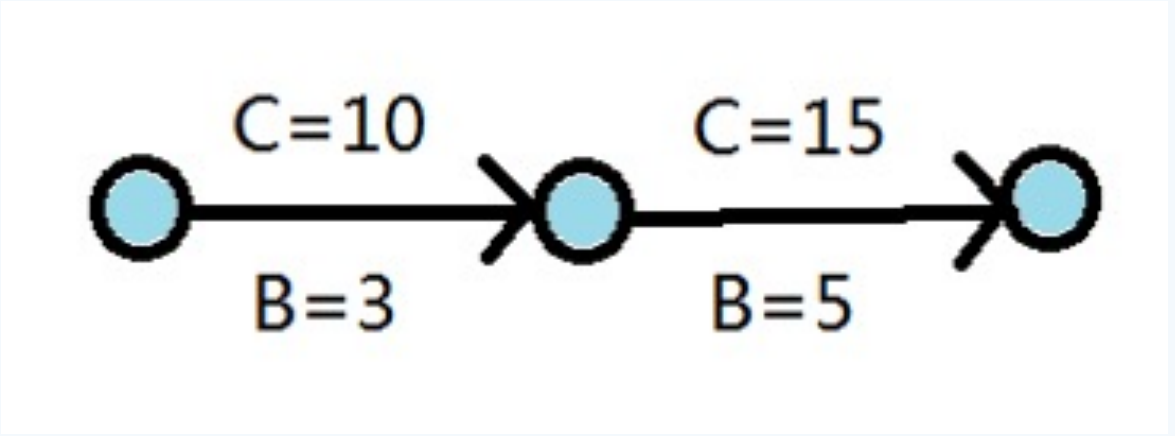
Paradox

- 解决哈密顿路！
- 每个点拆点，容量1费用1，边照常连。跑最小费用流。
- 最小费用流存在多项式时间做法。
- 图灵奖。

带上下界网络流

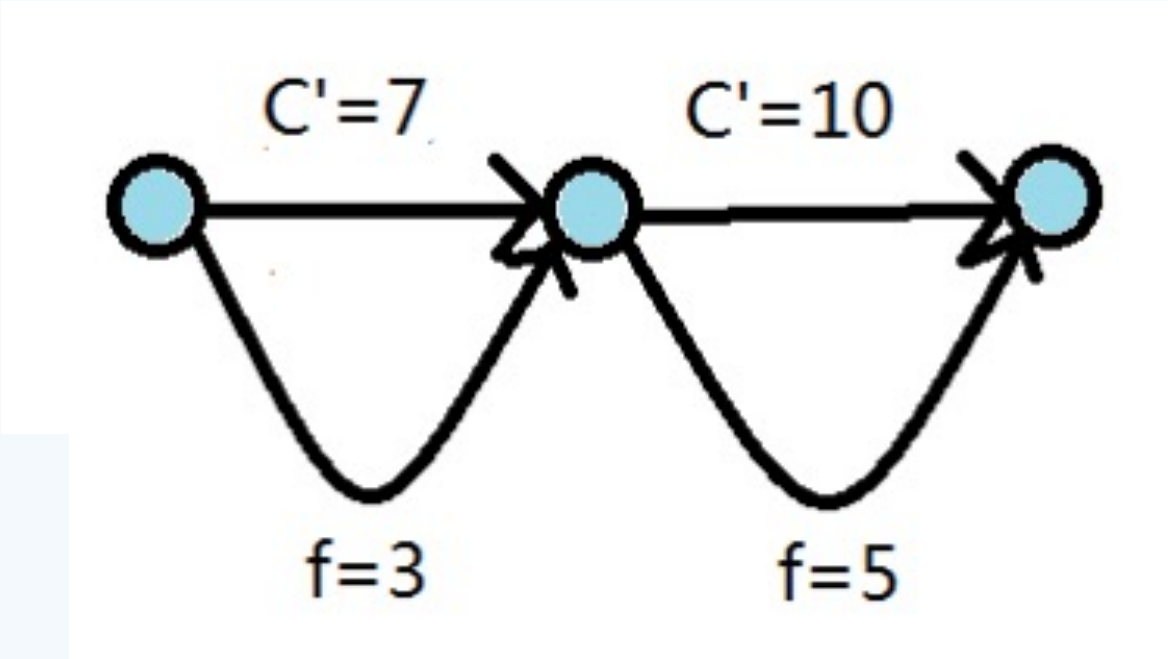
- 每一条边的流量除了上限不能超过容量外，还要求不能少于某个下限。
- 通常有以下四类问题：
  - 无源汇可行流
  - 有源汇可行流
  - 有源汇最大流
  - 有源汇最小流

无源汇可行流



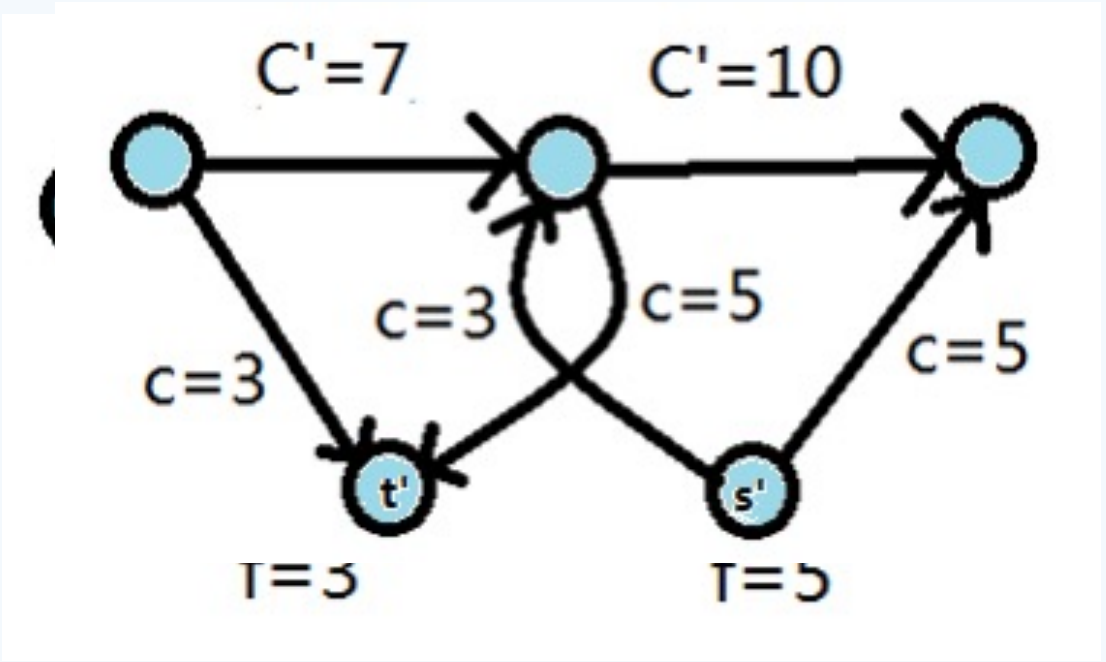
无源汇可行流

- 分离必要流和额外流。
- 要求必要流满流



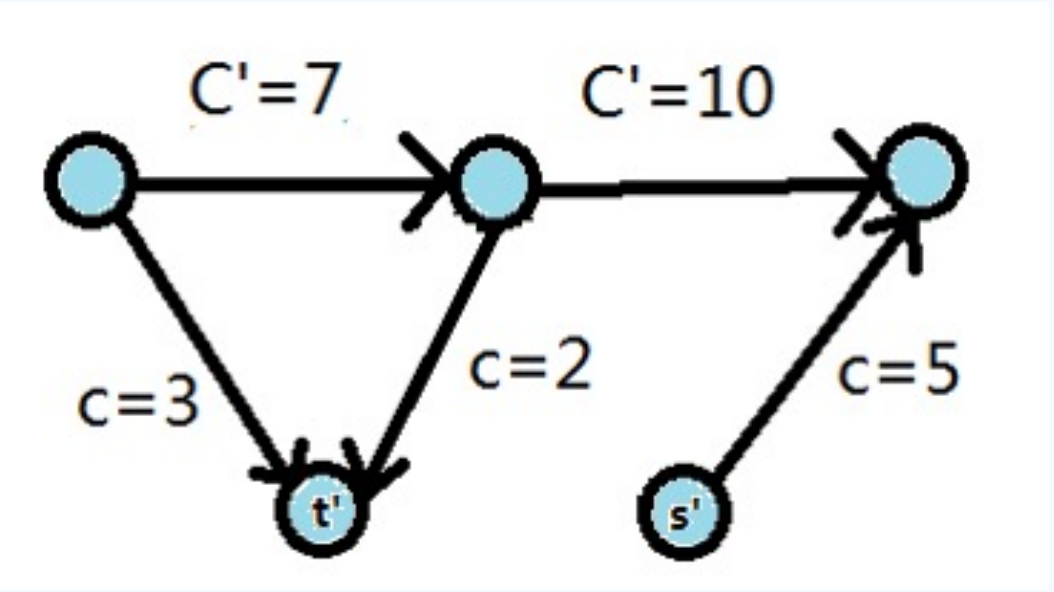
无源汇可行流

- 将必要流拆开
- 求新图最大流
- 当且仅当所有附加边满流时
- 原图存在可行流



无源汇可行流

- 化简连边



无源汇可行流

- 对于必要的流，考虑每个点的出入平衡关系，等效到超级源和超级汇。跑一个最大流。

有源汇可行流

- 连边 (T, S, ∞) 变为无源汇可行流。
- 若存在可行流，则原图的$S \to T$的流量=新图中 (T, S) 边的流量。

有源汇最大流

- 方法一：先判断原图是否有可行流。如果存在，再删去 (T,S) 边、附加边和附加点，在原图的残量网络中直接跑 $S \to T$ 的最大流，总的流量就是可行流流量+最大流流量。
- 方法二：连边 (T,S,∞,x) ，若 $x > ans$ 则不存在可行流，反之则存在。所以可以通过二分 x 来求解。

有源汇最小流

- 方法一：先判断原图是否有可行流。如果存在，再删去 (T,S) 边、附加边和附加点，在原图的残量网络中直接跑 $T \to S$ 的最大流，总的流量就是可行流流量-最大流流量。
- 方法二：连边 $(T,S,x,0)$ ，若 $x < ans$ 则不存在可行流，反之则存在。所以可以通过二分 x 来求解。

例0

- 有一张n个点的有向图，每个点有一个可以为正或负的点权。
- 对于每一个点，你可以选择或不选择，如果选择的话，价值会加上该点的点权。如果一个点被选择，那么它的所有后继也必须被选择。
- 计算可能取到的最大价值。
- Source: 经典题，e.g. 网络流24题第二题

例0

- 这个模型叫最大权闭合子图。
- 考虑建图最小割。我们假设一开始选了所有点权为正的点，接下来把源点向正点权的点连边，边权为点权，如果割掉了说明不取这个点。我们把负点权的向汇点连边，边权为-点权，如果割掉了说明取这个点。
- 我们考虑不合法的情况，比如我们有一个点x到y有一条路径，x点权为正，y点权为负，取了x不取y就是不合法的。在图上路径大概是S->x->y->T，为了实现S->x或者y->T至少要割一边，我们把原有的边边权设为inf即可。

例1

- 给定两个长度为n的数组$\{A\},\{B\}$，有两个初始为空的集合S_1,S_2，每次可以选择两个数对$(i,j),(p,q)$，满足$(i,j)\notin S_1,(p,q)\notin S_2,B_j>A_i,B_p<A_q,GCD(A_i,B_j)\neq 1,GCD(A_q,B_p)\neq 1,GCD(A_i,B_j,A_q,B_p)\neq 1$，那么就把$(i,j)$放入$S_1$中，把$(p,q)$放入$S_2$中。问最多能进行多少次操作。
- $n\leq 400$，$1\leq A_i,B_i\leq 10^9$
- Source: CC GNUM

例1

- 一个朴素的二分图匹配的想法，把满足条件的(i,j)放左边，满足条件的(p,q)放右边，如果$(i,j),(p,q)$能进行一次操作则在它们之间连一条边。
- 我们先把二分图匹配转化为网络流。
- 优化1：对于同一侧的点，如果它们的GCD相同，那么它们的连边也必定相同。所以可以将它们合并，并修改S或T到它们的容量。
- 优化2：优化中间的连边。我们对于每个出现过的质数建立一个点，然后如果两侧的点是这个质数的倍数，那么就与这个质数连一条边。

例2

- 幼儿园里有n个小朋友打算通过投票来决定睡不睡午觉。对他们来说，这个问题并不是很重要，于是他们决定发扬谦让精神。虽然每个人都有自己的主见，但是为了照顾一下自己朋友的想法，他们也可以投和自己本来意愿相反的票。我们定义一次投票的冲突数为好朋友之间发生冲突的总数加上和所有和自己本来意愿发生冲突的人数。 我们的问题就是，每位小朋友应该怎样投票，才能使冲突数最小？
- $n \leq 300$
- Source: SHOI2007

例2

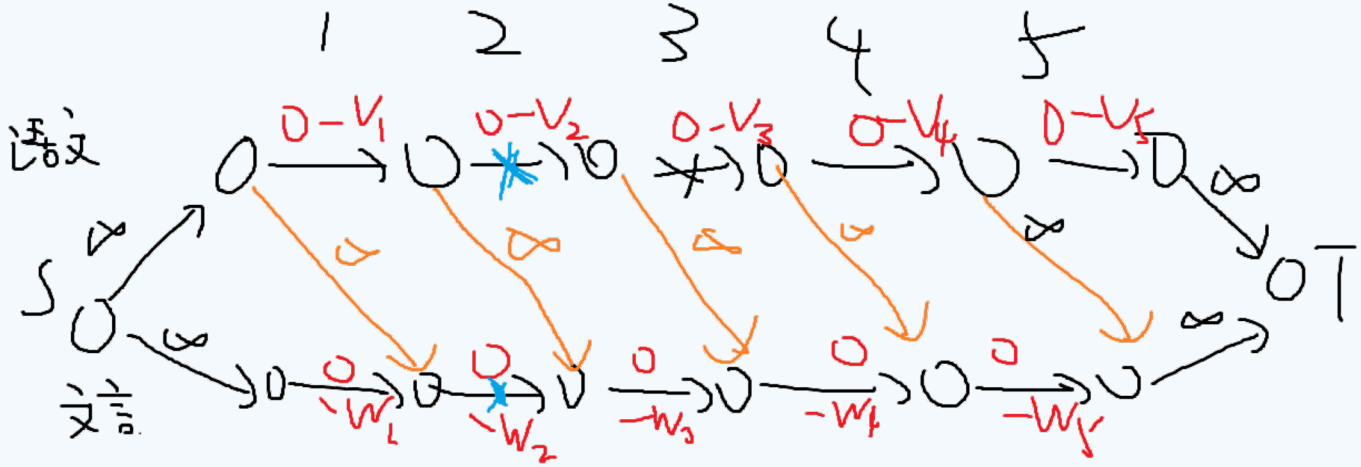
- 最小割建图。
- 本来是同意的，S->i（o），i->T（o+1）。
- 本来是反对的，S->i（o+1），i->T（o）。
- 好朋友a,b，a--b（1）。
- o取一个较大的数。o是为了保证不两边都割，割两边分别表示同意/不同意，如果割的不同侧就需要多割中间好朋友那条。

例3

- 有n门课程和m个学期。
- 有一些课程有先修课程，输入k个限制，形如a[i]是b[i]的先修课程。
- 每一门课程在每个学期可能获得不同的绩点。
- 你需要修完所有课，并输出可以获得的最大绩点，你的绩点就是每门课的绩点的平均值。
- Source: Codechef RIN

例3

- 把每个课程建成一条链，是一个o保证链上只取一个。
- 用inf边保证顺序，类似下图。



例4

- 给出一个$n \times n$的非负整数矩阵B和$1 \times n$的非负整数矩阵C。
- 求出一个$1 \times n$的01矩阵A。
- 最大化$D = (A * B - C) * A^T$。
- $n \leq 500$
- Source: TJOI2015

例4

- $D = A \times B \times A^T - C \times A^T$
- $D = \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j b_{i,j} - \sum_{i=1}^{n} a_i c_i$
- 那么问题就变成了n个物品，选第i个物品花费c_i，同时选第i个和第j个物品获得$B_{i,j}$的收益，最大化总收益-总代价。
- 那么把同时选看成是一个条件，那么就是选择该条件就必须选择两个物品，最大权闭合子图模型。

例5

- 在$n * n$的格子上，每个格子要么是带非负权的靶子，要么是一个固定方向的炮台（方向是平行于坐标轴的）。
- 保证炮台的方向上没有其他炮台，每个炮台只能选择不超过一个目标攻击，且炮台的攻击路径不能相交。
- 求最大收益。
- $n \leq 50$
- Source: SRM627 LaserTowers

例5

- 将一个炮台能攻击到的靶子链成一条链，边权为o-靶子权值。
- 如果两个炮台会冲突，则在它们的链之间加边。
- 在链之间加边(i_x, j_y)的意思是如果$i \geq x$则$j \geq y$。所以需要将纵向的链翻转。

例6

- 有一个长度为n的正整数序列A[i]，选一个子序列，使得原序列的任意一个长度为m的连续子序列中，被选出的元素个数不超过k个。
- 最大化选出的子序列中的元素和。
- $n \leq 1000, m, k \leq 100$。

例6

- 转换问题，不是选择1次，而是选择k次，每次长度为m的连续段中至多取一个。
- 接下来我们进行建图。对每个元素建立一个点，S->1->2->...->n->T，容量为k，费用为0；i往i+m连容量为1，费用-A[i]；i往T连容量为1，费用-A[i]（i+m>n）。
- 每一个流量相当于一个子序列。

例7

- 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。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。
- $m \leq 9, n \leq 60$
- Source: SCOI2007

例7

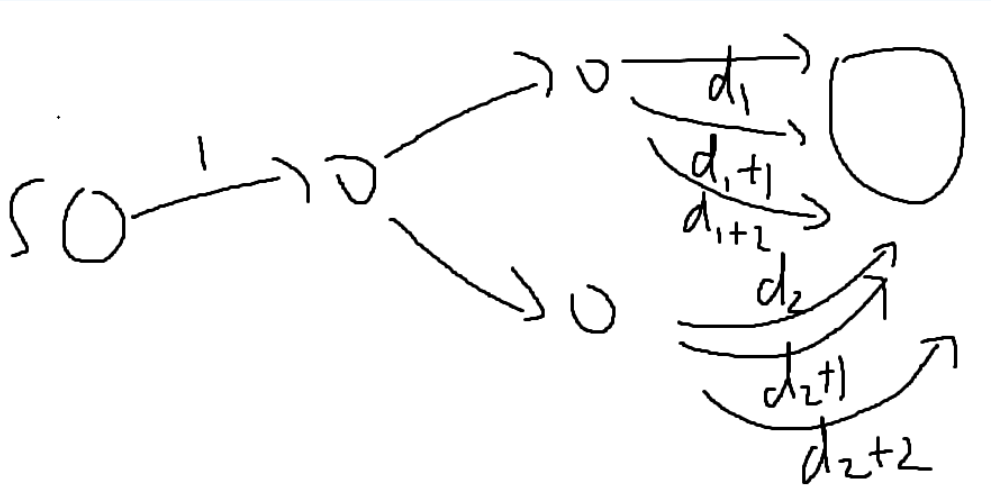
- 将每个工人拆成n个点，表示这辆车是这个工人倒数第1,2...n个修的，那么这辆车就会耽误后面的每个车主修车时长。给每一辆车配上这样一个点即可。
- 费用流建个图。

例8

- 在一些一对一游戏的比赛（如下棋、乒乓球和羽毛球的单打）中，我们经常会遇到A胜过B，B胜过C而C又胜过A的有趣情况，不妨形象的称之为剪刀石头布情况。有的时候，无聊的人们会津津乐道于统计有多少这样的剪刀石头布情况发生，即有多少对无序三元组(A, B, C)，满足其中的一个人在比赛中赢了另一个人，另一个人赢了第三个人而第三个人又胜过了第一个人。注意这里无序的意思是说三元组中元素的顺序并不重要，将(A, B, C)、(A, C, B)、(B, A, C)、(B, C, A)、(C, A, B)和(C, B, A)视为相同的情况。
- 有N个人参加一场这样的游戏的比赛，赛程规定任意两个人之间都要进行一场比赛：这样总共有N*(N-1)/2场比赛。比赛已经进行了一部分，我们想知道在极端情况下，比赛结束后最多会发生多少剪刀石头布情况。即给出已经发生的比赛结果，而你可以任意安排剩下的比赛的结果，以得到尽量多的剪刀石头布情况。
- $n \leq 100$
- Source: WC2007

例8

- 不是剪刀石头布的情况，就是有一个人赢了剩下两个，所以设每个人的出度是d_i，那么答案就是 $C_n^3 - \sum_{i=1}^{n} C_{d_i}^2$ ，我们要最小化 $\sum_{i=1}^{n} C_{d_i}^2$。
- 对于每一个比赛，我们要选择一侧将d加一。我们会发现，$C_x^2 - C_{x-1}^2 = x - 1$，对答案的贡献是越来越大的，因此直接建若干条费用递增的边即可。



例9

- JYY在玩galgame，有n个剧情点，在第i个剧情点时，有k[i]种可能的选择，到达k[i]段不同的支线剧情。看每段支线剧情需要一定时间，会到达另一个剧情点。JYY一开始处在1号剧情点，也就是游戏的开始。显然任何一个剧情点都是从1号剧情点可达的。此外，随着游戏的进行，剧情是不可逆的。所以游戏保证从任意剧情点出发，都不能再回到这个剧情点。
- 由于JYY过度使用修改器，导致游戏的“存档”和“读档”功能损坏了，所以JYY要想回到之前的剧情点，唯一的方法就是退出当前游戏，并开始新的游戏，也就是回到1号剧情点。JYY可以在任何时刻退出游戏并重新开始。不断开始新的游戏重复观看已经看过的剧情是很痛苦，JYY希望花费最少的时间，看完所有不同的支线剧情。
- $n \leq 300, \sum_i k_i \leq 5000$
- Source: AHOI2014

例9

- 相当于给定一张拓扑图，每条边有边权，每次只能从第一个点出发沿着拓扑图走一条路径，求遍历所有边所需要的最小边权和。
- 对于每一条边x->y，x->y连一条费用为-inf，容量1的边，再连一条费用z，容量inf的边。对于每一个点x，x->T连一条费用为0，容量为inf的边。S->1也连费用0，容量inf的边。

例10

- 有一个n*m的棋盘，每个格子上有一个数。每次可以选择两个相邻格子，把两个数都+1。
- 求出需要花多少步能把棋盘上的数变成同一个数，如果永远不行则输出-1。
- $n, m \leq 40$
- Source: SCOI2012

例10

- 黑白染色，假设黑格有n1个和为s1，白格有n2个和为s2。
- 如果n1=n2，必须要有s1=s2，并且答案有可二分性。（行或列是偶数）
- 如果n1!=n2，假设最后每个是s，就要有s*n1-s1=s*n2-s2，所以可以直接算出s。
- 如何check x是否合法？相当于二分图匹配，S->黑（x-原来的值），白->T（x-原来的值），相邻的黑->白（inf）。

例11

- 文理分科是一件很纠结的事情！（虽然看到这个题目的人肯定都没有纠结过）
- 小P所在的班级要进行文理分科。他的班级可以用一个n*m的矩阵进行描述，每个格子代表一个同学的座位。每位同学必须从文科和理科中选择一科。同学们在选择科目的时候会获得一个满意值。满意值按如下的方式得到：
  - 1．如果第i行第j列的同学选择了文科，则他将获得art[i][j]的满意值，如果选择理科，将得到science[i][j]的满意值。
  - 2．如果第i行第j列的同学选择了文科，并且他相邻（两个格子相邻当且仅当它们拥有一条相同的边）的同学全部选择了文科，则他会更开心，所以会增加same_art[i][j]的满意值。
  - 3．如果第i行第j列的同学选择了理科，并且他相邻的同学全部选择了理科，则增加same_science[i][j]的满意值。
- 小P想知道，大家应该如何选择，才能使所有人的满意值之和最大。请告诉他这个最大值。
- Source: bzoj3894

例11

- 对于每个人，S->这个人，为选理的满意度+o，这个人->T，为选文的满意度+o，就是说割S一侧表示选理，否则表示选文。
- 如果若干个人都学理会获得满意度，建立个点，把这些人往这个点连inf，把这个点往T连满意度，就如果有人选了文就必须割掉这个满意度。文同理。
- 此外还有一个神必做法，可以参见 http://hzwer.com/2422.html

例12

- n*m的棋盘，每个点有选择代价和控制收益。
- 一个点被控制当且仅当它被选择或者它的邻居都被选择。
- Source: Topcoder SRM 558

例12

- 考虑分解成以下条件：
  - 该点被选择。
  - 该点未被选择且四周的点均被选择。
- 黑白染色后成为文理分科模型。

例13

- n*m的棋盘，有些格子有棋子。
- 每次可以拿走同行或同列若干连续棋子，求至少需要多少次操作可以拿走所有棋子。
- Source: Topcoder SRM 578

例13

- 相当于每个格子选择横着取还是竖着取，如果横着相邻两个棋子都是横着取可以省1，竖着相邻两个棋子都是竖着取可以省1。
- 每个点i：S->i，边权o，i->T，边权o，割左边是横着，割右边是竖着。
- 接下来还是类似文理分科，对于横着相邻的格子x和y，我们连x->y（1），y->x（1），建立新点w，x->w,y->w（inf），w->T（1）。

对偶

- $\max\{c^T x | Ax \leq b\} = \min\{b^T y | A^T y \geq c\}$
- 原问题：你是工厂主，每个产品的收益是c，每个产品所需要的原料是A，第j个产品要A[i][j]个原料i，每种原料的数量是b，决策生产个数x最大化利润。
- 对偶问题：你是材料商，你要把数量是b的材料卖给工厂主，决策原材料的价格（“影子价格”），至少要定价多少才能赚不了钱。
- 左边<=右边

- “编一个下界”

例14

- 对于一条边，可以用a_i的代价给边权+1，或者用b_i的代价-1。
- 要求修改边权使得制定的生成树变成最小生成树，求最小代价。
- $n \leq 300$
- Source：BZOJ 3118

例14

- 显然只会给树边减，给非树边加。那么对于一组树边和非树边有$x_i + y_j \geq w_{i,j}$。
- 对偶后得到$\sum_j z_{i,j} \leq \mathrm{a_i}, \sum_i z_{i,j} \leq b_j$，要求最大化$\sum z_{i,j} * w_{i,j}$。

例15

- 鼹鼠们在底下开凿了n个洞，由n-1条隧道连接，对于任意的i>1，第i个洞都会和第i/2（取下整）个洞间有一条隧道，第i个洞内还有ci个食物能供最多ci只鼹鼠吃。
- 一共有m只鼹鼠，第i只鼹鼠住在第pi个洞内。一天早晨，前k只鼹鼠醒来了，而后n-k只鼹鼠均在睡觉，前k只鼹鼠就开始觅食，最终他们都会到达某一个洞，使得所有洞的ci均大于等于该洞内醒着的鼹鼠个数，而且要求鼹鼠行动路径总长度最小。现对于所有的1<=k<=m，输出最小的鼹鼠行动路径的总长度，保证一定存在某种合法方案。
- $n, m \leq 10^5$
- Source：NEERC2016

例15

- 容易看出费用流建图，相当于是每次灌一个新的容量进去，这时候我们就是要找一条最短路增广。
- 模拟费用流：记一下每条边当前的正/反容量，f[i]表示i到子树里某个能流出的点的最小费用和在哪，每次枚举lca开始流。

例17

- 给出一个$n * m$的网格，有一些位置是障碍。
- 蛇是一条至少包含两个格子的折线，蛇不能相交，并且蛇的两个端点必须满足在网格的边界上或者相邻即蛇形成环。
- 现在要用一些蛇覆盖网络，要求每个非障碍格子恰好被一条蛇覆盖。同时要求最小化不构成环的蛇的数量。
- $n, m \leq 12$

例17

- 可以发现除了非环的蛇的两端的度数是1以外，其他的点的度数都将是2。
- 先经典的黑白染色，然后S向白点和黑点向T连$(b=2,c=2)$的边，表示所有的点度数都应该为2。接着白点向相邻黑点连$(b=0,c=1)$的边。
- 为了能够表示出非环的蛇，S向边界黑点，边界白点向T连 $(b=0,c=1,cost=1)$的边，若这条边满流则表示直接缺少1的度数，也就是这是一个非环蛇的端点。
- 跑最小费用流即可，答案就是费用/2。


谢谢大家