Flow

TQX

2024.8.12

最大流

感性理解

一张有向图，规定了源点、汇点，每条边有容量。
现在我们在源点灌水，水沿边流，每条边的流量不能超过容量，
水不能在中间结点堆积，求汇点最多能接到多少水?

理性理解

网络（network）是指一个特殊的有向图 $G = (V, E)$，其与一般有向图的不同之处在于有容量和源汇点。
E 中的每条边 (u, v) 都有一个被称为容量（capacity）的权值，记作 $c(u, v)$。当 $(u, v) \notin E$ 时，可以假定 $c(u, v) = 0$。
V 中有两个特殊的点：源点 (source)s 和汇点 (sink)$t(s \neq t)$。

理性理解

网络（network）是指一个特殊的有向图 $G=(V,E)$，其与一般有向图的不同之处在于有容量和源汇点。
E 中的每条边 (u,v) 都有一个被称为容量（capacity）的权值，记作 $c(u,v)$。当 $(u,v)\notin E$ 时，可以假定 $c(u,v)=0$。
V 中有两个特殊的点：源点 (source)s 和汇点 (sink)$t(s\neq t)$。

G 中的流是一个函数 $f\colon V\times V\to R$, 满足以下两个性质:

1. 容量限制：对所有 $u,v\in V$，有 $0\leq f(u,v)\leq c(u,v)$。
2. 流守恒性：对所有 $u\in V-\{s,t\}$，有 $\sum_{v\in V}f(u,v)=0$。

理性理解

网络（network）是指一个特殊的有向图 $G=(V,E)$，其与一般有向图的不同之处在于有容量和源汇点。
E 中的每条边 (u,v) 都有一个被称为容量（capacity）的权值，记作 $c(u,v)$。当 $(u,v)\notin E$ 时，可以假定 $c(u,v)=0$。
V 中有两个特殊的点：源点 (source)s 和汇点 (sink)$t(s\neq t)$。

G 中的流是一个函数 $f\colon V\times V\rightarrow R$, 满足以下两个性质：

1. 容量限制：对所有 $u,v\in V$，有 $0\leq f(u,v)\leq c(u,v)$。
2. 流守恒性：对所有 $u\in V-\{s,t\}$，有 $\sum_{v\in V}f(u,v)=0$。

最大流问题即求 G 上的流 f 以最大化整个网络的流量 $|f|=\sum_{x\in V}f(s,x)-\sum_{x\in V}f(x,s)$。

残量网络

流了一个流之后，有的边因为有流量，所以还能够的流的量就减少了，同时我们能够选择在这条边上反悔一定的流量，我们将这抽象成残量网络。
设 f 为流网络 $G=(V,E)$ 中的一个流，对于节点对 (u,v)，定义其残存容量如下：

$$c_f(u,v)=\begin{cases} c(u,v)-f(u,v) & (u,v)\in E \\ f(u,v) & (v,u)\in E \\ 0 & otherwise \end{cases}$$

我们将 G 中所有结点和剩余容量大于 0 的边构成的子图称为残量网络 G_f（Residual Network），即 $G_f=(V,E_f)$，其中 $E_f=\left\{(u,v)\mid c_f(u,v)>0\right\}$。
初始残量网络即为原网络，实现网络流算法时我们维护残量网络，那么流经原边的流在原图上体现为多流了一定流量，流经反边的流在原图上体现为少流了一定流量。

增广路

增广路是 G_f 上一条从源点 s 到汇点 t 的路径。对于一条增广路，我们给每一条边 (u,v) 都加上等量的流量 $c(p)=\min\{c_f(u,v)|(u,v)\in p\}$ 以令整个网络的流量增加，这一过程被称为增广（Augment）。由此，最大流的求解可以被视为若干次增广分别得到的流的叠加。

增广路

增广路是 G_f 上一条从源点 s 到汇点 t 的路径。对于一条增广路，我们给每一条边 (u, v) 都加上等量的流量 $c(p) = \min\{c_f(u, v) | (u, v) \in p\}$ 以令整个网络的流量增加，这一过程被称为增广（Augment）。由此，最大流的求解可以被视为若干次增广分别得到的流的叠加。

Edmonds–Karp 算法

通过 BFS 寻找增广路：每一轮先沿着流量不为 0 的边在残量网络上 BFS 一遍，求出 s 到每个点的距离（边权为 1），那么 s 到 t 就意味着有增广路。沿 s 到 t 的最短路增广，直到无法找到增广路为止。复杂度为 $\mathcal{O}(nm^2)$。

Dinic

每次只增广一条路太慢了。每次 BFS 完后，一次 DFS 同时增广多条路：即找到一条最短路后，回溯时将沿途的流量修改，并继续查看其他最短路径是否可以增广。

Dinic

每次只增广一条路太慢了。每次 BFS 完后，一次 DFS 同时增广多条路：即找到一条最短路后，回溯时将沿途的流量修改，并继续查看其他最短路径是否可以增广。
当前弧优化：枚举一个节点的一些边后，之后枚举时直接跳过之前枚举过的边。注意在下一轮 BFS 前还原记录的当前弧。
这样时间复杂度为 $\mathcal{O}(n^2m)$，但绝大多数情况跑得很快。

Dinic

每次只增广一条路太慢了。每次 BFS 完后，一次 DFS 同时增广多条路：即找到一条最短路后，回溯时将沿途的流量修改，并继续查看其他最短路径是否可以增广。
当前弧优化：枚举一个节点的一些边后，之后枚举时直接跳过之前枚举过的边。注意在下一轮 BFS 前还原记录的当前弧。
这样时间复杂度为 $\mathcal{O}(n^2m)$，但绝大多数情况跑得很快。

网络流做二分图匹配

从 s 向所有左部点连流量为 1 的边，原图的边变为左部点到右部点的流量为 1 的边，右部点向汇点连流量为 1 的边。
跑完后的残存网络中，若某条匹配中的边两端在不同强连通分量里面，这条边一定在最大匹配中。复杂度 $\mathcal{O}(m\sqrt{n})$。

HLPP

在求解最大流的过程中，我们想象每个节点处有一个水箱，允许让水箱存储一些水，直到求解完毕，水箱里的水最后都流到汇点或流回源点。也就是说对于 $u \in V - \{s\}$，我们允许

$$\sum_{v \in V} f(v, u) \geq \sum_{v \in V} f(u, v)$$

对于 $u \in V - \{s\}$，我们记

$$e(u) = \sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v) \geq 0$$

为进入节点 u 的超额流，若 $e(u) > 0$，称节点 u 溢出。

HLPP

我们还为每个节点记录其高度 $h(u)$，推送流的时候只将流推送到高度比当前节点小 1 的节点。
算法开始时，$h(s) = n$，其余节点 h 为 0。我们先从 s 推送尽可能多的流到相邻节点，然后不断通过“推送”和“重贴标签”两种操作来让这些超额流流到 t 或 s。

HLPP

我们还为每个节点记录其高度 $h(u)$，推送流的时候只将流推送到高度比当前节点小 1 的节点。
算法开始时，$h(s) = n$，其余节点 h 为 0。我们先从 s 推送尽可能多的流到相邻节点，然后不断通过“推送”和“重贴标签”两种操作来让这些超额流流到 t 或 s。

推送操作：对于溢出的节点 v，尝试将 $e(v)$ 通过 v 的出边推送到 v 的各个相邻的且高度符合要求的节点；

HLPP

我们还为每个节点记录其高度 $h(u)$，推送流的时候只将流推送到高度比当前节点小 1 的节点。
算法开始时，$h(s)=n$，其余节点 h 为 0。我们先从 s 推送尽可能多的流到相邻节点，然后不断通过“推送”和“重贴标签”两种操作来让这些超额流流到 t 或 s。

推送操作：对于溢出的节点 v，尝试将 $e(v)$ 通过 v 的出边推送到 v 的各个相邻的且高度符合要求的节点；

重贴标签操作：如果一个溢出的节点 v 在推送操作之后还是溢出的，就重新为它分配高度，将其高度设置为它所连向的所有节点的高度的最小值 $+1$，然后再尝试推送操作。

HLPP

我们还为每个节点记录其高度 $h(u)$，推送流的时候只将流推送到高度比当前节点小 1 的节点。
算法开始时，$h(s)=n$，其余节点 h 为 0。我们先从 s 推送尽可能多的流到相邻节点，然后不断通过“推送”和“重贴标签”两种操作来让这些超额流流到 t 或 s。

推送操作：对于溢出的节点 v，尝试将 $e(v)$ 通过 v 的出边推送到 v 的各个相邻的且高度符合要求的节点；

重贴标签操作：如果一个溢出的节点 v 在推送操作之后还是溢出的，就重新为它分配高度，将其高度设置为它所连向的所有节点的高度的最小值 $+1$，然后再尝试推送操作。

每次选择高度最高的溢出节点来做推送和重贴标签操作，复杂度为 $\mathcal{O}(n^2\sqrt{m})$，该复杂度较紧。

HLPP 的优化

BFS 优化：初始时将高度设为该节点到 t 的距离（s 除外）；

gap 优化：如果没有高度为 i 的节点了，那么高度大于 i 的节点就无法通过高度为 i 的节点推送到 t，它们的超额流只能推送回 s，故直接将这些节点的高度设为 $n+1$。

最小割

对于一个网络流图 $G=(V,E)$，其割的定义为一种点的划分方式：将所有的点划分为 S 和 $T=V-S$ 两个集合，其中源点 $s\in S$，汇点 $t\in T$。

最小割

对于一个网络流图 $G = (V, E)$，其割的定义为一种点的划分方式：将所有的点划分为 S 和 $T = V - S$ 两个集合，其中源点 $s \in S$，汇点 $t \in T$。

我们定义割 (S, T) 的容量 $c(S, T)$ 表示所有从 S 到 T 的边的容量之和，即 $c(S, T) = \sum_{u \in S, v \in T} c(u, v)$。当然我们也可以用 $c(s, t)$ 表示 $c(S, T)$。

最小割就是求得一个割 (S, T) 使得割的容量 $c(S, T)$ 最小。

最小割

对于一个网络流图 $G = (V, E)$，其割的定义为一种点的划分方式：将所有的点划分为 S 和 $T = V - S$ 两个集合，其中源点 $s \in S$，汇点 $t \in T$。

我们定义割 (S, T) 的容量 $c(S, T)$ 表示所有从 S 到 T 的边的容量之和，即 $c(S, T) = \sum_{u \in S, v \in T} c(u, v)$。当然我们也可以用 $c(s, t)$ 表示 $c(S, T)$。

最小割就是求得一个割 (S, T) 使得割的容量 $c(S, T)$ 最小。

最大流最小割定理

最大流等于最小割。

最小割

对于一个网络流图 $G=(V,E)$，其割的定义为一种点的划分方式：将所有的点划分为 S 和 $T=V-S$ 两个集合，其中源点 $s\in S$，汇点 $t\in T$。

我们定义割 (S,T) 的容量 $c(S,T)$ 表示所有从 S 到 T 的边的容量之和，即 $c(S,T)=\sum_{u\in S,v\in T}c(u,v)$。当然我们也可以用 $c(s,t)$ 表示 $c(S,T)$。

最小割就是求得一个割 (S,T) 使得割的容量 $c(S,T)$ 最小。

最大流最小割定理

最大流等于最小割。

感性理解：一根水管能流多快取决于最窄的地方。

最大流最小割定理

最大流 $\leq$ 最小割是显然的，因为所有流必须经过这些割边，所以最大流小于等于任何一个割的容量。取到等号当且仅当这些 $S \to T$ 的边全部满流，$T \to S$ 的边全部空流。

最大流最小割定理

最大流 $\leq$ 最小割是显然的，因为所有流必须经过这些割边，所以最大流小于等于任何一个割的容量。取到等号当且仅当这些 $S \to T$ 的边全部满流，$T \to S$ 的边全部空流。

增广结束后，残量网络上不存在 s 到 t 的路径，可以将 V 划分为 s 出发能到达的点集 S 与剩余部分 T。则 $\{S, T\}$ 是一个 G_f 的割。
在原图上 S 到 T 的边均满流，T 到 S 的边均空流，因此增广结束后最大流与此时 $\{S, T\}$ 的割相同。因此最大流等于最小割。

CF546E Soldier and Traveling

Description

有 n 个城市，其间可能有无向边，每个城市现在有 a_i 个人，还有一个目标人数 b_i。所有人都可以停留在其所在城市或沿某边走一步。问能否让每个城市达到目标人数。n 100 m 200。

CF546E Soldier and Traveling

Description

有 n 个城市，其间可能有无向边，每个城市现在有 a_i 个人，还有一个目标人数 b_i。所有人都可以停留在其所在城市或沿某边走一步。问能否让每个城市达到目标人数。$n\ 100\ m\ 200$。

为每个城市拆点，如下建边。

- s 连向 i，流量为 a_i，代表该城市原有 a_i 个人；
- i 连向 i'，流量为 ∞，代表留在原城市。
- i' 连向 t，流量为 b_I，代表城市最终有 b_i 人。
- 对于原图中的边 u, v，连边 $(u, v'), (v, u')$，流量为无穷，表示走到相邻城市。

满流则有解。

二分图题目

实际转化后为二分图最大匹配/最小点覆盖/最大独立集，DAG 最小路径覆盖等等。在二分图一章已充分叙述，实际实现时通常使用更快的网络流来实现，在此不多赘述。

最小割

一类经典网络流题目，利用最大流 $=$ 最小割转化为网络流，通常满足以下性质：

- 有若干元素，每个元素有两种状态（黑或白，选或不选…），要求选出其中一种，且选择某一种有相应收益。
- 有若干限制或额外收益，类似选择了 A 的 X 状态就不能选择 B 的 Y 的状态或者同时选择 A 的 X 状态和 B 的 Y 状态带来 w 的额外收益。
- 最后要求出收益最大值。

对于这类问题，可以先获得所有收益，然后减去构造出来的图的最小割（意义是不得不放弃的最小代价）（在此最小割中，在源点集合的选择一种状态，而在汇点集合的选择另外一种状态），即为最终答案。

典中典

luogu P4001 狼抓兔子

一张图有 $N \times M$ 个节点，排成网格状。横向、纵向、斜向（左上-右下）相邻节点均有边连接，边有边权。求边权和最小的一个边集，使得删除边集中的边后左上与右下不连通。
N, M 1000。

典中典

luogu P4001 狼抓兔子

一张图有 $N\times M$ 个节点，排成网格状。横向、纵向、斜向（左上-右下）相邻节点均有边连接，边有边权。求边权和最小的一个边集，使得删除边集中的边后左上与右下不连通。
N, M 1000。

最小割 $=$ 最大流。这道题可以充分证明网络流的玄学复杂度。
事实上利用平面图的最小割等于其对偶图的最短路，可以正确复杂图通过这道题。

洛谷 P1646 happiness

Description

一个 $n \times m$ 座位表，每个座位上有一个同学。对于每个同学，选择文科或理科各有一定喜悦值。同时，对于位置相邻的两个同学，如果他们同时选择文科或理科各会带来额外的喜悦值。求最大喜悦值。

$n, m \leq 100$

洛谷 P1646 happiness

Description

一个 $n \times m$ 座位表，每个座位上有一个同学。对于每个同学，选择文科或理科各有一定喜悦值。同时，对于位置相邻的两个同学，如果他们同时选择文科或理科各会带来额外的喜悦值。求最大喜悦值。

$n, m \leq 100$

用最小割表示不得不放弃的代价。s 和 t 分别代表文科和理科。对于每个同学分别建点，如果在最小割中属于源点集合则代表他选择文科，否则选择理科。

洛谷 P1646 happiness

Description

一个 $n \times m$ 座位表，每个座位上有一个同学。对于每个同学，选择文科或理科各有一定喜悦值。同时，对于位置相邻的两个同学，如果他们同时选择文科或理科各会带来额外的喜悦值。求最大喜悦值。
$n, m \leq 100$

用最小割表示不得不放弃的代价。s 和 t 分别代表文科和理科。对于每个同学分别建点，如果在最小割中属于源点集合则代表他选择文科，否则选择理科。

s 向每个同学连接容量为选择文科喜悦值的边，每个同学向 t 连接容量为选择理科喜悦值的边。这样割去源点的边代表放弃文科带来的权值，最终选择了理科；割去汇点的边同理。

洛谷 P1646 happiness

Description

一个 $n \times m$ 座位表，每个座位上有一个同学。对于每个同学，选择文科或理科各有一定喜悦值。同时，对于位置相邻的两个同学，如果他们同时选择文科或理科各会带来额外的喜悦值。求最大喜悦值。

$n, m \leq 100$

用最小割表示不得不放弃的代价。s 和 t 分别代表文科和理科。对于每个同学分别建点，如果在最小割中属于源点集合则代表他选择文科，否则选择理科。

s 向每个同学连接容量为选择文科喜悦值的边，每个同学向 t 连接容量为选择理科喜悦值的边。这样割去源点的边代表放弃文科带来的权值，最终选择了理科；割去汇点的边同理。

而后考虑额外权值。仍然是套路地，对于每一个额外权值建立新点，如果这个权值和选择文科相关，则由 s 向它连接容量为权值的边，由它向对应同学连接容量为 ∞ 的边。这样如果其中某位同学最终属于汇点集合（相当于选择理科），那么由汇点连出的边必然会断去，代表放弃了这个额外权值。

最终答案为初始权值和减去最小割。

洛谷 P5458 [BJOI2016] 水晶

Description

太长了，自己看吧。

洛谷 P5458 [BJOI2016] 水晶

Description

太长了，自己看吧。

原题坐标和位置并不是一一对应的。于是可以这样转化 $(x, y, z) \to (x - z, y - z)$ 。注意到这样转化后仍然有 $x + y + z \equiv x - z + y - z \pmod 3$ 。
不难发现 a 共振和 b 共振一起相当于不能存在相邻的三个位置 $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ 满足

$$x_1 + y_1 \equiv 1 \pmod 3 \tag{1}$$

$$x_2 + y_2 \equiv 0 \pmod 3 \tag{2}$$

$$x_3 + y_3 \equiv 2 \pmod 3 \tag{3}$$

于是对于每个水晶先拆点，分为入点和出点，由入点向出点连容量为权值的边。
然后将水晶按照横纵坐标数值之和除以 3 的余数分类。模 3 为 1 的向相邻的模 3 为 0 的连边，模 3 为 0 的向相邻的模 3 为 2 的连边，容量均为 $+\infty$ 。
然后源点向模 3 为 1 的点连边，模 3 为 2 的点向汇点连边，容量均为 $+\infty$ 。
最终答案为总权值减取最小割。

洛谷 P6054 开门大吉

Description

有 n 个人， m 套题，第 i 个人去做第 j 套题会产生 $g_{i,j}$ 的权值。现在还有若干要求，每个要求形如 $i\ j\ k$ 表示第 i 个人做的题的编号至少比第 j 个人做的题目编号大 k 。现在要给每个人分配一套题使得总权值尽量小。

$n, m \leq 80, T \leq 50$

洛谷 P6054 开门大吉

Description

有 n 个人，m 套题，第 i 个人去做第 j 套题会产生 $g_{i,j}$ 的权值。现在还有若干要求，每个要求形如 $i\ j\ k$ 表示第 i 个人做的题的编号至少比第 j 个人做的题目编号大 k 。现在要给每个人分配一套题使得总权值尽量小。

$n, m \leq 80, T \leq 50$

总权值最小联想到最小割模型。
拆点。将每个人 i 拆为 $m+1$ 个点，连边 $(i,j) \rightarrow (i,j+1)$ 流量为 $g_{i,j}$ ，割去这条边表示第 i 个人做第 j 套题产生的代价。
然后源点向所有的 $(i,1)$ 连边，流量为 $+\infty$ ； $(i,m+1)$ 向汇点连边，流量为 $+\infty$ 。
现在的关键在于如何满足要求。
对于要求 (i,j,k) ，我们需要连边 $(j,x) \rightarrow (i,\min(x+k,m+1))$ 流量为 $+\infty$ 。如果选择在 $(j,x) \rightarrow (j,x+1)$ 处割掉，那么由于 $S \rightarrow (j,1) \rightarrow (j,x) \rightarrow (i,x+k) \rightarrow (i,m+1) \rightarrow T$ 这条路径的存在，则必须在 $\geq x+k$ 的地方割掉，满足了要求。

最大权闭合子图

对于一个点带权的 DAG 图，定义其闭合子图为一个点集 S 满足若 $u \in S$，则所有 u 能到达的点都 $\in S$，而最大权闭合子图就是点权和最大的闭合子图。

最大权闭合子图

对于一个点带权的 DAG 图，定义其闭合子图为一个点集 S 满足若 $u \in S$，则所有 u 能到达的点都 $\in S$，而最大权闭合子图就是点权和最大的闭合子图。

先选取所有的正权点的权值，然后建立最小割模型，由源点向正权点连接容量为其权值的边，负权点向汇点连接容量为其权值相反数的边，然后原图的边均保留，容量为 ∞。

最后答案为所有的正权点的权值减去最小割。在最小割中，没有被割的连向正权点的边表示选择了这个点，被割的由负权点连出的边表示选择了这个点，不难发现这样做一定是满足条件的。

[NOI2009] 植物大战僵尸

Description

太长了，自行查看题面。

[NOI2009] 植物大战僵尸

Description

太长了，自行查看题面。

每个植物看作一个点，点权为其收益，如果植物 a 保护植物 b，那么连边 $b \to a$ 表示将 a 吃掉后才能吃掉 b。

注意这样建出来的是有环的，而方才介绍的模型是不允许有环的。于是拓扑排序去除其中的环然后套用上面的模型即可。

最大密度子图

Description

n 个点 m 条边的无向图 $G=\{V,E\}$，求其子图 $G'=\{V',E'\}$，最大化 $\dfrac{|E'|}{|V'|}$。

最大密度子图

Description

n 个点 m 条边的无向图 $G=\{V,E\}$ ，求其子图 $G'=\{V',E'\}$ ，最大化 $\dfrac{|E'|}{|V'|}$ 。

不妨设最终答案为 t ，则必然有

$$\frac{|E'|}{|V'|}=t \Rightarrow |E'|-t|V'|=0 \tag{4}$$

考虑二分答案，不妨设当前二分值为 g 。对每条边建权值为 1 的点，向其原本连接的两个点连有向边。原图每个点的权值则定为 $-g$ ，然后跑最大闭合权子图，得到答案后依据其和 0 的相对大小关系即可继续二分。

杂题

其他一些有意思的杂题被放到了最后。

费用流

每条边不仅有容量 $c(u, v)$，还有费用 $w(u, v)$。从一条边流一些流需要花费流的大小乘以该边费用。整个流 f 的费用为 $\sum f(u, v)w(u, v)$。

通常要求在流最大的同时费用最小。

mcmf

建反向边时将反向边的费用设为原边费用的相反数。

mcmf

建反向边时将反向边的费用设为原边费用的相反数。

每次寻找一条以 w 为权值的 s 到 t 的最短路来增广。这里找最短路有负权，需要使用 SPFA 或者 johnson。
它的最坏时间复杂度是 $\mathcal{O}(nmf)$，与流量相关，因此是伪多项式的，参考 uoj 那个费用流模板题，通常情况下都是玄学的。

mcmf

建反向边时将反向边的费用设为原边费用的相反数。

每次寻找一条以 w 为权值的 s 到 t 的最短路来增广。这里找最短路有负权，需要使用 SPFA 或者 johnson。
它的最坏时间复杂度是 $\mathcal{O}(nmf)$，与流量相关，因此是伪多项式的，参考 uoj 那个费用流模板题，通常情况下都是玄学的。

可以 capacity scaling 优化到弱多项式复杂度。

mcmf

建反向边时将反向边的费用设为原边费用的相反数。

每次寻找一条以 w 为权值的 s 到 t 的最短路来增广。这里找最短路有负权，需要使用 SPFA 或者 johnson。
它的最坏时间复杂度是 $\mathcal{O}(nmf)$，与流量相关，因此是伪多项式的，参考 uoj 那个费用流模板题，通常情况下都是玄学的。

可以 capacity scaling 优化到弱多项式复杂度。

用费用流求二分图最优匹配

与二分图最大匹配接近，左右部点之间的边加上费用。

洛谷 P2517 [HAOI2010] 订货

Description

某公司估计市场在第 i 个月对某产品的需求量为 U_i，已知在第 i 月该产品的订货单价为 d_i，上个月月底未销完的单位产品要付存贮费用 m，假定第一月月初的库存量为零，第 n 月月底的库存量也为零，问如何安排这 n 个月订购计划，才能使成本最低？每月月初订购，订购后产品立即到货，进库并供应市场，于当月被售掉则不必付存贮费。假设仓库容量为 S。$n \leq 50$。

洛谷 P2517 [HAOI2010] 订货

Description

某公司估计市场在第 i 个月对某产品的需求量为 U_i，已知在第 i 月该产品的订货单价为 d_i，上个月月底未销完的单位产品要付存贮费用 m，假定第一月月初的库存量为零，第 n 月月底的库存量也为零，问如何安排这 n 个月订购计划，才能使成本最低？每月月初订购，订购后产品立即到货，进库并供应市场，于当月被售掉则不必付存贮费。假设仓库容量为 S。$n \leq 50$。

为每个月建点，按如下方式建边：

- s 连向每个月，容量为正无穷，费用为 d_i，代表进货；
- 每个月连向 t，容量为 U_i，费用为 0，代表售货；
- 每个月连向下一个月，容量为 S，费用为 m，代表存货。

最小费用最大流。

洛谷 P2053 [SCOI2007] 修车

Description

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。

$M \le 9, N \le 60$。

洛谷 P2053 [SCOI2007] 修车

Description

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。
$M \le 9, N \le 60$。

为每辆车建点，每名技术人员拆成 N 个点，分别代表第 i 个工人修的第 j 辆车。先修的车显然会被等很多次。按如下方式建边：

- s 到车，容量 1，费用 0;
- 工人到 t，容量 1，费用 0;
- 车到工人，容量 1，费用为这位工人修这辆车的时间 $\times i$。

最小费用最大流。

CF277E Binary Tree on Plane

Description

给平面上 n 个点，要求用这些点组成一个二叉树 (每个节点的儿子节点不超过两个)，定义每条边的权值为两个点之间的欧几里得距离。求一个权值和最小的二叉树，并输出这个权值。
点 i 可称为点 j 的父亲当且仅当 i 的 y 坐标大于 j 。
$n \leq 400$

CF277E Binary Tree on Plane

Description

给平面上 n 个点，要求用这些点组成一个二叉树 (每个节点的儿子节点不超过两个)，定义每条边的权值为两个点之间的欧几里得距离。求一个权值和最小的二叉树，并输出这个权值。
点 i 可称为点 j 的父亲当且仅当 i 的 y 坐标大于 j 。
$n \leq 400$

拆点。一个点拆为入点和出点。

- s 向所有入点连接容量为 2 费用为 0 的边代表该节点至多两个儿子。
- 所有出点向 t 连边，容量为 1 费用为 0 代表该节点至多有一个父亲。
- 对于每个点 i ，它的入点向所有可以作为它儿子 j 的出点连接容量为 1 ，费用为距离的边。

跑最小费用最大流。注意只有根没有父亲，所以当且仅当最大流为 $n-1$ 时才有解。

CF884F Anti-Palindromize

Description

对于一个字串 a，若其长度 m 为偶数，且对于 $\forall i \in [1, m]$ ，有 $a_i \neq a_{m-i+1}$ ，则将其称为反回文串。
Ivan 有一个由 n 个小写拉丁字母构成的字串 s ，且 n 为偶数。他想将 s 的字符重新排列构成反回文串 t ，称 t 的美丽值 $Ans = \sum_{i=1}^{strlen(s)} b_i[s_i = t_i]$ 。
请帮 Ivan 确定 Ans 的最大值。$n \leq 100, b_i \leq 100$

CF884F Anti-Palindromize

Description

对于一个字串 a，若其长度 m 为偶数，且对于 $\forall i \in [1, m]$ ，有 $a_i \neq a_{m-i+1}$ ，则将其称为反回文串。
Ivan 有一个由 n 个小写拉丁字母构成的字串 s ，且 n 为偶数。他想将 s 的字符重新排列构成反回文串 t ，称 t 的美丽值 $Ans = \sum_{i=1}^{strlen(s)} b_i[s_i = t_i]$ 。
请帮 Ivan 确定 Ans 的最大值。$n \leq 100, b_i \leq 100$

对每个字符 c 建点，源点向其连接流量为 c 在 s 中出现次数的边。再对每个位置建点，向汇点连接流量为 1 的边，表示每个位置仅有一个字符。对每个字符 c 新建 $\frac{n}{2}$ 个点，其中第 i 个代表 $i, len(s) - i + 1$ 这两个位置。c 向位置 i 连流量为 1 的边，它再向对应的两个位置分别连接流量为 1 费用依据题意的边。跑最大费用最大流即可。

无源汇有上下界可行流

普通的网络流，我们只限制了每条边的流量，也就是它的上界，但有些特殊题目，还会恶心地限制一下每条边流量的下界，即有上下界网络流。

无源汇有上下界可行流

普通的网络流，我们只限制了每条边的流量，也就是它的上界，但有些特殊题目，还会恶心地限制一下每条边流量的下界，即有上下界网络流。

loj 115. 无源汇有上下界可行流

可行流是指一个网络流中存在的一条满足除源点汇点之外，所有点的流入量等于流出量(即流量守恒）的流。无源汇意味着个图中所有点都必须满足流入量等于流出量，只需要找到任意一条可行流即可。

无源汇有上下界可行流

首先让所有边流量强行达到下界，称为初始流，初始流不一定满足流量守恒，需要通过适当在一些边上增加流量来使它满足流量守恒，新增加的流量合起来也是一条流，称为附加流。

无源汇有上下界可行流

首先让所有边流量强行达到下界，称为初始流，初始流不一定满足流量守恒，需要通过适当在一些边上增加流量来使它满足流量守恒，新增加的流量合起来也是一条流，称为附加流。

附加流首先要满足不会让任意一条边流量超过上界，因此每条边的最大流量为上界-下界，同时还要弥补初始流中的问题。对于点 i，令 w_i 表示初始流中 i 的流入量-流出量，对 w_i 分情况讨论：

- $w_i = 0$，初始流流量守恒，不需要对 i 点操作，让附加流满足流量守恒即可。
- $w_i > 0$，附加流在满足 i 点流量守恒之后，还要让它多流出 w_i 的流量，于是从附加流的源点 S 向 i 连一条流量为 w_i 的边。
- $w_i < 0$，附加流要让 i 点额外流入 $-w_i$ 的流量，于是从 i 向附加流的汇点 T 连一条流量为 $-w_i$ 的边。

无源汇有上下界可行流

首先让所有边流量强行达到下界，称为初始流，初始流不一定满足流量守恒，需要通过适当在一些边上增加流量来使它满足流量守恒，新增加的流量合起来也是一条流，称为附加流。

附加流首先要满足不会让任意一条边流量超过上界，因此每条边的最大流量为上界-下界，同时还要弥补初始流中的问题。对于点 i，令 w_i 表示初始流中 i 的流入量-流出量，对 w_i 分情况讨论：

- $w_i = 0$，初始流流量守恒，不需要对 i 点操作，让附加流满足流量守恒即可。
- $w_i > 0$，附加流在满足 i 点流量守恒之后，还要让它多流出 w_i 的流量，于是从附加流的源点 S 向 i 连一条流量为 w_i 的边。
- $w_i < 0$，附加流要让 i 点额外流入 $-w_i$ 的流量，于是从 i 向附加流的汇点 T 连一条流量为 $-w_i$ 的边。

为了满足流量守恒，额外连的这些边必须满流，于是在新图上跑最大流，只要流量 $= \sum w_i(w_i > 0 \ = \sum -w_i(w_i < 0)$ 则说明有解，否则无解。注意到第二个等号一定满足，所以只用考虑第一个等号即可。

无源汇有上下界可行流

首先让所有边流量强行达到下界，称为初始流，初始流不一定满足流量守恒，需要通过适当在一些边上增加流量来使它满足流量守恒，新增加的流量合起来也是一条流，称为附加流。

附加流首先要满足不会让任意一条边流量超过上界，因此每条边的最大流量为上界-下界，同时还要弥补初始流中的问题。对于点 i，令 w_i 表示初始流中 i 的流入量-流出量，对 w_i 分情况讨论：

- $w_i = 0$，初始流流量守恒，不需要对 i 点操作，让附加流满足流量守恒即可。
- $w_i > 0$，附加流在满足 i 点流量守恒之后，还要让它多流出 w_i 的流量，于是从附加流的源点 S 向 i 连一条流量为 w_i 的边。
- $w_i < 0$，附加流要让 i 点额外流入 $-w_i$ 的流量，于是从 i 向附加流的汇点 T 连一条流量为 $-w_i$ 的边。

为了满足流量守恒，额外连的这些边必须满流，于是在新图上跑最大流，只要流量 $= \sum w_i(w_i > 0 \;= \sum -w_i(w_i < 0)$ 则说明有解，否则无解。注意到第二个等号一定满足，所以只用考虑第一个等号即可。

最后求可行流，则将附加流中每条原图中边的流量加上其下界即可。

有源汇

有源汇有上下界可行流

增加了源点 s 和汇点 t（注意与上文中的 S 与 T 区分），源点和汇点不需要满足流量守恒，源点可以任意流出，汇点可以任意流入。

有源汇

有源汇有上下界可行流

增加了源点 s 和汇点 t (注意与上文中的 S 与 T 区分)，源点和汇点不需要满足流量守恒，源点可以任意流出，汇点可以任意流入。

直接增加一条从 t 连向 s 的流量上限为 inf，无下限的边，然后按照无源汇的做即可，同时这种情况下，可行流对应的原图流量，可以直接通过 $t \to s$ 这条边的流量得到。

有源汇

有源汇

loj 116./洛谷 P5192 有源汇有上下界最大流

在可行流的基础上，要求原图流量最大。

有源汇

loj 116./洛谷 P5192 有源汇有上下界最大流

在可行流的基础上，要求原图流量最大。

先解出可行流，然后在残量网络上跑最大流，叠加到可行流上。

有源汇

loj 116./洛谷 P5192 有源汇有上下界最大流

在可行流的基础上，要求原图流量最大。

先解出可行流，然后在残量网络上跑最大流，叠加到可行流上。

loj 117. 有源汇上下界最小流

如题。

先跑出一个可行流，然后在残量网络上跑 $t \to s$ 的最大流，将这最大流的路径上所有道路的流量都撤回，依然得到一个可行流。

洛谷 P4043 [AHOI2014] 支线剧情

Description

有一个拓扑图，保证 1 号点可以到达所有点。每条边有花费时间 v_i。每一步可以花一定时间沿一条边走，或返回到 1 号点。问至少经过多少时间后才能遍历完所有边。$n \leq 300$

洛谷 P4043 [AHOI2014] 支线剧情

Description

有一个拓扑图，保证 1 号点可以到达所有点。每条边有花费时间 v_i。每一步可以花一定时间沿一条边走，或返回到 1 号点。问至少经过多少时间后才能遍历完所有边。$n \leq 300$

以 1 为 s。每条边的下界定为 1，上界定为正无穷，费用定为 v_i。再从每个点连向 t，下界为 0，上界为正无穷，费用为 0。跑最小费用可行流。

UOJ575 光伏元件

Description

给出 $n \times n$ 的 $0/1$ 矩阵 A ，其中 $A_{i,j} = 1$ 表示位置 (i,j) 有光伏元件，而 0 表示没有元件。设第 i 行的元件个数为 $c_{0,i}$ ，第 i 列的元件个数为 $c_{1,i}$ 。对于每个 i ，给出 dl_i, dr_i, k_i ，要求 $|c_{0,i} - c_{1,i}| \le k_i$ 且 $c_{0,i}, c_{1,i} \in [dl_i, dr_i]$ 。给出 $n \times n$ 的矩阵 C ，以 $C_{i,j}$ 表示改变位置 (i,j) 上元件有/无状态的代价；特别地，若 $C_{i,j} = -1$ ，则表示 (i,j) 位置的状态不可改变。
询问最少代价。
$n \le 100$

UOJ575 光伏元件

Description

给出 $n \times n$ 的 $0/1$ 矩阵 A ，其中 $A_{i,j}=1$ 表示位置 (i,j) 有光伏元件，而 0 表示没有元件。设第 i 行的元件个数为 $c_{0,i}$ ，第 i 列的元件个数为 $c_{1,i}$ 。对于每个 i ，给出 dl_i, dr_i, k_i ，要求 $|c_{0,i}-c_{1,i}| \le k_i$ 且 $c_{0,i}, c_{1,i} \in [dl_i, dr_i]$ 。给出 $n \times n$ 的矩阵 C ，以 $C_{i,j}$ 表示改变位置 (i,j) 上元件有/无状态的代价；特别地，若 $C_{i,j}=-1$ ，则表示 (i,j) 位置的状态不可改变。
询问最少代价。
$n \le 100$

每行每列建一个点，行向列连边，有流量代表放了。对于不可改变的位置，建边是平凡的。对于 $a_{i,j}=0$ 的位置，连边 $(i,j',0,1,c_{i,j})$ ；对于 $a_{i,j}=1$ 的位置，连边 $(i,j',1,1,0)$ 表示强制放且无代价，再连 $(j',i,0,1,c_{i,j})$ ，表示可换为不放。
对 $k=0$ ，对每个 i 连边 $(i,i',dl_i,dr_i,0)$ ，然后跑无源汇最小费用可行流即可。
回到 $k \ne 0$ 的情况，对每个 i 建立虚点 D_i ，连边 $(i,D_i,dl_i,dr_i,0),(D_i,i',dl_i,dr_i,0)$ ，而后建立超级源汇 S,T ，连边 $(S,D_i,0,k,0),(D_i,T,0,k,0)$ 。跑有源汇最小费用可行流。

洛谷 P5038 奇怪的游戏

Description

一个 $n*m$ 的棋盘，每个格子有一个数。每次可选择相邻的两个格子，使这两个格子上的数各 $+1$ 。
请求出最少操作次数使得棋盘上的数变成同一个数。若无解输出 -1 。
$n, m \leq 40$

洛谷 P5038 奇怪的游戏

Description

一个 $n * m$ 的棋盘，每个格子有一个数。每次可选择相邻的两个格子，使这两个格子上的数各 $+1$ 。
请求出最少操作次数使得棋盘上的数变成同一个数。若无解输出 -1 。
$n, m \leq 40$

黑白染色。假设黑色格子有 cb 个，权值和为 sb ，白色格子有 cw 个，权值和为 sw 。每次操作一定是黑色白色权值各 $+1$ 。
假设最终的同一个数为 X ，那么就有 $sb - sw = X(cb - cw)$
$cb - cw \neq 0$ 当且仅当 n, m 均为奇数。此时可以计算出 X 的值，然后最大流随便 $check$ 一下就行。
否则 n, m 中至少一个为偶数。此时倘若 X 可行，那么 $X + 1$ 必然也可行。于是二分答案，还是用最大流 $check$ 。
$check$ 类似于二分图匹配的过程，最后判断是否满流即可。

洛谷 P3358 最长 k 可重区间集问题

Description

给定实直线 L 上 n 个开区间组成的集合 I，和一个正整数 k，试设计一个算法，从开区间集合 I 中选取出开区间集合 $\mathbf{S} \subseteq I$，使得在实直线 L 上的任意一点 x，S 中包含 x 的开区间个数不超过 k，且 $\sum_{z\in S}|z|$ 达到最大（$|z|$ 表示开区间 z 的长度）。求其最大值。
$n \leq 500, 1 \leq k \leq 3$

洛谷 P3358 最长 k 可重区间集问题

Description

给定实直线 L 上 n 个开区间组成的集合 I，和一个正整数 k，试设计一个算法，从开区间集合 I 中选取出开区间集合 $\mathbf{S} \subseteq I$，使得在实直线 L 上的任意一点 x，S 中包含 x 的开区间个数不超过 k，且 $\sum_{z \in S} |z|$ 达到最大（$|z|$ 表示开区间 z 的长度）。求其最大值。
$n \leq 500, 1 \leq k \leq 3$

建立超级源点向数轴上 0 的位置连边，流量为 k 费用为 0 。然后对于数轴上每个点 i 向 $i+1$ 连边，流量为 k 费用为 0 。然后从数轴上最大的点向超级汇点连边。最后对于每个区间，从 l_i 向 r_i 连边，流量为 1 费用为 $w_i = r_i - l_i$ 。跑出最大费用可行流即为答案。值域比较大时可以先离散化。

AGC038F Two Permutations

Description

给两个长为 n 的排列 p, q ，构造两个排列 a, b 。要求 a_i 要么等于 p_i 要么等于 i ， b_i 要么等于 q_i 要么等于 i ，最大化 $\sum_{i=1}^{n}[a_i \neq b_i]$ 。$n \leq 10^5$

AGC038F Two Permutations

Description

给两个长为 n 的排列 p, q ，构造两个排列 a, b 。要求 a_i 要么等于 p_i 要么等于 i ， b_i 要么等于 q_i 要么等于 i ，最大化 $\sum_{i=1}^{n}[a_i \neq b_i]$ 。 $n \leq 10^5$

将排列看作环。容易发现一个环要么保留，要么全部拆成自环。设 P_i 表示 p_i 所在的环，Q_i 表示 q_i 所在的环，做以下分类讨论：

- $p_i = q_i = i$ ，那么有 $a_i = b_i$ 。
- $p_i = i, q_i \neq i$ ，当且仅当 Q_i 被拆后有 $a_i = b_i$ 。
- $p_i \neq i, q_i = i$ ，当且仅当 P_i 被拆后有 $a_i = b_i$ 。
- $p_i \neq q_i, p_i \neq i, q_i \neq i$ ，当且仅当 P_i, Q_i 都被拆后有 $a_i = b_i$ 。
- $p_i = q_i, p_i \neq i, q_i \neq i$ ，当且仅当 P_i, Q_i 均被拆或均未被拆有 $a_i = b_i$ 。

AGC038F Two Permutations

对每个环分别建点。P 在 S 中表示 P 被拆，Q 在 T 中表示 Q 被拆。考虑最小割模型。

- $p_i = q_i = i$，耗费 1 的代价。
- $p_i = i, q_i \neq i$， Q_i 在 T 中时要消耗 1 的代价，故 S 向 Q_i 连流量为 1 的边。
- $p_i \neq i, q_i = i$， P_i 在 S 中时要消耗 1 的代价，故 P_i 向 T 连流量为 1 的边。
- $p_i \neq q_i, p_i \neq i, q_i \neq i$， P_i 在 S 中且 Q_i 在 T 中时要消耗 1 的代价，故 P_i 向 Q_i 连流量为 1 的边。
- $p_i = q_i, p_i \neq i, q_i \neq i$，（$P_i$ 在 S 中且 Q_i 在 T 中）或（P_i 在 T 中且 Q_i 在 S 中）时要消耗 1 的代价，故 P_i 和 Q_i 相互连流量为 1 的边。

完结撒花

网络流的建模非常多样，还是需要读者的自行练习与总结。希望大家能获得一点收获。