模拟费用流

安师大附中 杨珑 & 余姚中学 陈扩文

2022 年 1 月 23 日

定义

这里给出最小费用流的线性规划形式：

定义

这里给出最小费用流的线性规划形式：
定义流网络 $G = (V, E)$。设 $c(u, v)$ 表示 u 到 v 的边的容量，$a(u, v)$ 表示 u 到 v 的费用，源点 s，汇点 t，$x_{u,v}$ 表示点 u 到 v 的流量，s 到 t 的总流量记为 V。

定义

这里给出最小费用流的线性规划形式：
定义流网络 $G=(V,E)$。设 $c(u,v)$ 表示 u 到 v 的边的容量，$a(u,v)$ 表示 u 到 v 的费用，源点 s，汇点 t，$x_{u,v}$ 表示点 u 到 v 的流量，s 到 t 的总流量记为 V。
则：

$$\min \sum_{(u,v)\in E} a(u,v)x_{u,v}$$

$$\forall u \in V-\{s,t\}, \sum_{(u,v)\in E} x_{u,v} = \sum_{(v,u)\in E} x_{v,u}$$

$$\forall (u,v)\in E, 0 \le x_{u,v} \le c(u,v)$$

$$\sum_{(s,u)\in E} x_{s,u} = \sum_{(u,t)\in E} x_{u,t} = V$$

定义

这里给出最小费用流的线性规划形式：
定义流网络 $G=(V,E)$。设 $c(u,v)$ 表示 u 到 v 的边的容量，$a(u,v)$ 表示 u 到 v 的费用，源点 s，汇点 t，$x_{u,v}$ 表示点 u 到 v 的流量，s 到 t 的总流量记为 V。
则：

$$\min \sum_{(u,v)\in E} a(u,v)x_{u,v}$$

$$\forall u \in V-\{s,t\}, \sum_{(u,v)\in E} x_{u,v} = \sum_{(v,u)\in E} x_{v,u}$$

$$\forall (u,v)\in E, 0 \le x_{u,v} \le c(u,v)$$

$$\sum_{(s,u)\in E} x_{s,u} = \sum_{(u,t)\in E} x_{u,t} = V$$

目前最常用的费用流算法为 SSP 算法，即每次找一条费用最小的增广路进行增广，其时间复杂度为 $O(nmf)$。

定义

这里给出最小费用流的线性规划形式：
定义流网络 $G=(V,E)$。设 $c(u,v)$ 表示 u 到 v 的边的容量，$a(u,v)$ 表示 u 到 v 的费用，源点 s，汇点 t，$x_{u,v}$ 表示点 u 到 v 的流量，s 到 t 的总流量记为 V。
则：

$$\min \sum_{(u,v)\in E} a(u,v)x_{u,v}$$

$$\forall u \in V-\{s,t\}, \sum_{(u,v)\in E} x_{u,v} = \sum_{(v,u)\in E} x_{v,u}$$

$$\forall (u,v)\in E, 0 \le x_{u,v} \le c(u,v)$$

$$\sum_{(s,u)\in E} x_{s,u} = \sum_{(u,t)\in E} x_{u,t} = V$$

目前最常用的费用流算法为 SSP 算法，即每次找一条费用最小的增广路进行增广，其时间复杂度为 $O(nmf)$。
但在很多特殊的图上，我们可以快速地寻找到增广路并快速进行增广。

定义

这里给出最小费用流的线性规划形式：
定义流网络 $G=(V,E)$。设 $c(u,v)$ 表示 u 到 v 的边的容量，$a(u,v)$ 表示 u 到 v 的费用，源点 s，汇点 t，$x_{u,v}$ 表示点 u 到 v 的流量，s 到 t 的总流量记为 V。
则：

$$\min \sum_{(u,v)\in E} a(u,v)x_{u,v}$$

$$\forall u \in V-\{s,t\}, \sum_{(u,v)\in E} x_{u,v} = \sum_{(v,u)\in E} x_{v,u}$$

$$\forall (u,v)\in E, 0 \le x_{u,v} \le c(u,v)$$

$$\sum_{(s,u)\in E} x_{s,u} = \sum_{(u,t)\in E} x_{u,t} = V$$

目前最常用的费用流算法为 SSP 算法，即每次找一条费用最小的增广路进行增广，其时间复杂度为 $O(nmf)$。
但在很多特殊的图上，我们可以快速地寻找到增广路并快速进行增广。
为了方便，以下用 (u,v,c,a) 表示一条从 u 到 v，容量为 c，费用为 a 的边。

寻找关键点

在某些图上，增广路长度普遍较小，而且存在少量节点决定了图的主要性质，且这些点之间的增广路的形式较为单一。我们取这些点作为关键点。我们往往只需要考察并维护这些关键点之间的最短路。

「AGC018C」Coins

有 n 个人，第 i 个有 A_i 枚金币，B_i 枚银币，C_i 枚铜币，要将人分成三组，人数分别是 X、Y、Z $(n = X + Y + Z)$，要最大化第一组人的金币个数之和 $+$ 第二组人的银币个数 $+$ 第三组人的铜币个数之和。

$X + Y + Z \leq 10^5, 1 \leq A_i, B_i, C_i \leq 10^9$。

「AGC018C」Coins

我们先建出费用流图：

「AGC018C」Coins

我们先建出费用流图：
对于我们对每个人建一个点 $P_1 \dots P_n$，对每一组各建一个点 F_1, F_2, F_3。

「AGC018C」Coins

我们先建出费用流图：
对于我们对每个人建一个点 $P_1 \dots P_n$，对每一组各建一个点 F_1, F_2, F_3。

我们连边 $(P_i, F_1, 1, -A_i)$，$(P_i, F_2, 1, -B_i)$,$(P_i, F_3, 1, -C_i)$ 表示这个人分配到哪一组。

「AGC018C」Coins

我们先建出费用流图：
对于我们对每个人建一个点 $P_1 \dots P_n$，对每一组各建一个点 F_1, F_2, F_3。

我们连边 $(P_i, F_1, 1, -A_i)$，$(P_i, F_2, 1, -B_i)$,$(P_i, F_3, 1, -C_i)$ 表示这个人分配到哪一组。
接着连接 $(s, P_i, 1, 0)$，$(F_1, t, X, 0)$，$(F_2, t, Y, 0)$，$(F_3, t, Z, 0)$。

「AGC018C」Coins

我们先建出费用流图：
对于我们对每个人建一个点 $P_1 \dots P_n$，对每一组各建一个点 F_1, F_2, F_3。

我们连边 $(P_i, F_1, 1, -A_i)$，$(P_i, F_2, 1, -B_i)$,$(P_i, F_3, 1, -C_i)$ 表示这个人分配到哪一组。
接着连接 $(s, P_i, 1, 0)$，$(F_1, t, X, 0)$，$(F_2, t, Y, 0)$，$(F_3, t, Z, 0)$。
我们选取 s, F_1, F_2, F_3, t 作为关键点。此时 s 到 F_1, F_2, F_3 的增广路是不可以走反向边的，F_1, F_2, F_3 之间的增广路只能经过最多一条反向边，其意义为把一个人从他所在的组改为另一组。

「AGC018C」Coins

我们先建出费用流图：
对于我们对每个人建一个点 $P_1 \dots P_n$，对每一组各建一个点 F_1, F_2, F_3。

我们连边 $(P_i, F_1, 1, -A_i)$，$(P_i, F_2, 1, -B_i)$,$(P_i, F_3, 1, -C_i)$ 表示这个人分配到哪一组。
接着连接 $(s, P_i, 1, 0)$，$(F_1, t, X, 0)$，$(F_2, t, Y, 0)$，$(F_3, t, Z, 0)$。
我们选取 s, F_1, F_2, F_3, t 作为关键点。此时 s 到 F_1, F_2, F_3 的增广路是不可以走反向边的，F_1, F_2, F_3 之间的增广路只能经过最多一条反向边，其意义为把一个人从他所在的组改为另一组。
因此我们可以用堆维护任意两个关键点之间的最短路，对于每两组之间我们维护一个堆表示这个人这样换组的收益。每次找关键点间最短路时我们取最大的就好。

「NOI2019」序列

给定非负整数 n, k, L 和两个长度为 n 的数组 $\{a_i\}$，$\{b_i\}$。

「NOI2019」序列

给定非负整数 n, k, L 和两个长度为 n 的数组 $\{a_i\}$，$\{b_i\}$。
要确定两个长度为 k 的序列 $\{c_i\}, \{d_i\}$，其中
$1 \leq c_1 < c_2 < \cdots < c_k \leq n, 1 \leq d_1 < d_2 < \cdots < d_k \leq n$。

「NOI2019」序列

给定非负整数 n, k, L 和两个长度为 n 的数组 $\{a_i\}$，$\{b_i\}$。
要确定两个长度为 k 的序列 $\{c_i\}, \{d_i\}$，其中
$1 \le c_1 < c_2 < \cdots < c_k \le n, 1 \le d_1 < d_2 < \cdots < d_k \le n$。
要求 $|\{c_i\} \cap \{d_i\}| \ge L$。

「NOI2019」序列

给定非负整数 n, k, L 和两个长度为 n 的数组 $\{a_i\}$，$\{b_i\}$。
要确定两个长度为 k 的序列 $\{c_i\}, \{d_i\}$，其中
$1 \le c_1 < c_2 < \cdots < c_k \le n, 1 \le d_1 < d_2 < \cdots < d_k \le n$。
要求 $|\{c_i\} \cap \{d_i\}| \ge L$。
最大化 $\sum_{i=1}^{k} a_{c_i} + b_{d_i}$。

「NOI2019」序列

考虑每一个下标可以分为四种：

「NOI2019」序列

考虑每一个下标可以分为四种：

1. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中；
2. 只出现在 $\{c_i\}$ 中；
3. 只出现在 $\{d_i\}$ 中；
4. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中。

「NOI2019」序列

考虑每一个下标可以分为四种：
1. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中；
2. 只出现在 $\{c_i\}$ 中；
3. 只出现在 $\{d_i\}$ 中；
4. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中。

这样我们可以建三个点 F_1, F_2, F_3 表示 $2, 3, 4$ 种情况，对下标 i 建一个点 P_i。

「NOI2019」序列

考虑每一个下标可以分为四种：
1. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中；
2. 只出现在 $\{c_i\}$ 中；
3. 只出现在 $\{d_i\}$ 中；
4. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中。

这样我们可以建三个点 F_1, F_2, F_3 表示 $2, 3, 4$ 种情况，对下标 i 建一个点 P_i。

考虑由于选 F_3 的点是大于等于 L 的，我们枚举这样的点数为 p。

「NOI2019」序列

考虑每一个下标可以分为四种：
1. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中；
2. 只出现在 $\{c_i\}$ 中；
3. 只出现在 $\{d_i\}$ 中；
4. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中。

这样我们可以建三个点 F_1, F_2, F_3 表示 $2, 3, 4$ 种情况，对下标 i 建一个点 P_i。

考虑由于选 F_3 的点是大于等于 L 的，我们枚举这样的点数为 p。

然后连边 $(s, P_i, 1, 0)$，$(P_i, F_1, 1, -a_i)$，$(P_i, F_2, 1, -b_i)$，$(P_i, F_3, 1, -a_i - b_i)$，$(F_1, t, k - p, 0)$，$(F_2, t, k - p, 0)$，$(F_3, t, p, 0)$。

「NOI2019」序列

考虑每一个下标可以分为四种：
1. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中；
2. 只出现在 $\{c_i\}$ 中；
3. 只出现在 $\{d_i\}$ 中；
4. 同时不出现在 $\{c_i\}$ 和 $\{d_i\}$ 中。

这样我们可以建三个点 F_1, F_2, F_3 表示 $2, 3, 4$ 种情况，对下标 i 建一个点 P_i。

考虑由于选 F_3 的点是大于等于 L 的，我们枚举这样的点数为 p。

然后连边 $(s, P_i, 1, 0)$，$(P_i, F_1, 1, -a_i)$，$(P_i, F_2, 1, -b_i)$，$(P_i, F_3, 1, -a_i - b_i)$，$(F_1, t, k - p, 0)$，$(F_2, t, k - p, 0)$，$(F_3, t, p, 0)$。

该图的结构与上一题一致，因此在 p 固定时我们可以套用上题的做法，而此题我们可以从小到大枚举 p，由于 F_1，F_2 到 t 的流量在减小，我们需要执行退流操作，这实质是寻找 t 到 s 的增广路。这样在 p 增大时，我们每次要找的增广路条数是 $O(1)$ 的。总复杂度依然为 $O(n \log n)$。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。
我们在对一条 s 到 t 的增广路进行增广之后，$dis(x)$ 不会减小。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。
我们在对一条 s 到 t 的增广路进行增广之后，$dis(x)$ 不会减小。
考虑如何证明，首先增广会有两种影响：删除原有边，加入反向边。前者不会增大 $dis(x)$，考虑后者。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。
我们在对一条 s 到 t 的增广路进行增广之后，$dis(x)$ 不会减小。
考虑如何证明，首先增广会有两种影响：删除原有边，加入反向边。前者不会增大 $dis(x)$，考虑后者。
对于最短路上的一条原边 (u, v)。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。
我们在对一条 s 到 t 的增广路进行增广之后，$dis(x)$ 不会减小。
考虑如何证明，首先增广会有两种影响：删除原有边，加入反向边。前者不会增大 $dis(x)$，考虑后者。
对于最短路上的一条原边 (u, v)。
因为这条边在最短路上，所以有 $dis(v) = dis(u) + a(u, v)$，加入反向边若产生松弛导致 $dis(u)$ 被更新，则有 $dis(u) > dis(v) - a(u, v)$，这二者是矛盾的。

凸性

对于无负环的残存网络，定义 $dis(x)$ 表示源点 s 到 x 的距离，如果无法到达 $dis(x) = +\infty$。
我们在对一条 s 到 t 的增广路进行增广之后，$dis(x)$ 不会减小。
考虑如何证明，首先增广会有两种影响：删除原有边，加入反向边。前者不会增大 $dis(x)$，考虑后者。
对于最短路上的一条原边 (u, v)。
因为这条边在最短路上，所以有 $dis(v) = dis(u) + a(u, v)$，加入反向边若产生松弛导致 $dis(u)$ 被更新，则有 $dis(u) > dis(v) - a(u, v)$，这二者是矛盾的。
定义 $cost(f)$ 表示流量为 f 时（这里 f 可以为负，表示 t 到 s 流），s 到 t 的最小费用，则该函数为凸函数。

「网络流 24 题」餐巾计划问题

一个餐厅在 n 天里，第 i 天需要用 r_i 块餐巾，用完后可以送到快洗部或慢洗部，前者洗一块需要 t_0 天，费用为 f_0，后者需要 t_1 天，费用为 f_1，购买新餐巾的代价是 p。

「网络流 24 题」餐巾计划问题

一个餐厅在 n 天里，第 i 天需要用 r_i 块餐巾，用完后可以送到快洗部或慢洗部，前者洗一块需要 t_0 天，费用为 f_0，后者需要 t_1 天，费用为 f_1，购买新餐巾的代价是 p。
求度过 n 天的最小花费。

「网络流 24 题」餐巾计划问题

一个餐厅在 n 天里，第 i 天需要用 r_i 块餐巾，用完后可以送到快洗部或慢洗部，前者洗一块需要 t_0 天，费用为 f_0，后者需要 t_1 天，费用为 f_1，购买新餐巾的代价是 p。
求度过 n 天的最小花费。
建出费用流图后，流量表示最开始买的餐巾数，由于费用关于流量的函数是凸的，因此我们可以采用三分来确定最初有多少餐巾，然后贪心得到答案。

维护凸函数

将 $V-\{s,t\}$ 分为两个集合 V_1,V_2，若两个集合之间仅有一条边相连，那么其中一侧产生的费用关于该边流量的函数也为凸函数。在一些特殊图中，我们往往用数据结构直接维护这个凸函数。

一些经典模型

对于如下的图：P_i 到 P_{i+1} 有边，s 到 $P_1 \dots P_n$ 有边，$P_1 \dots P_n$ 到 t 有边的图我们称为单向链。

一些经典模型

对于如下的图：P_i 到 P_{i+1} 有边，s 到 $P_1 \dots P_n$ 有边，$P_1 \dots P_n$ 到 t 有边的图我们称为单向链。
对于单向链来说，我们有如下可能的方法：

对于如下的图：P_i 到 P_{i+1} 有边，s 到 $P_1 \dots P_n$ 有边，$P_1 \dots P_n$ 到 t 有边的图我们称为单向链。
对于单向链来说，我们有如下可能的方法：
1. 用线段树维护每条边的流量和最优方案

一些经典模型

对于如下的图：P_i 到 P_{i+1} 有边，s 到 $P_1 \dots P_n$ 有边，$P_1 \dots P_n$ 到 t 有边的图我们称为单向链。

对于单向链来说，我们有如下可能的方法：

1. 用线段树维护每条边的流量和最优方案
2. 我们按照 P_1 到 P_n 的顺序加点。此时在加入 P_i 后由于我们不知道会有多少流量流入 P_{i+1}，设其为 x，此时前半部分产生的费用关于 x 是一个凸函数。用合适的数据结构维护这个凸函数即可。

「Lydsy1708 月赛」跳伞求生

n 个玩家，第 i 个战斗力为 a_i。

「Lydsy1708 月赛」跳伞求生

n 个玩家，第 i 个战斗力为 a_i。
m 个敌人，第 i 个战斗力为 b_i，赏金为 c_i。

「Lydsy1708 月赛」跳伞求生

n 个玩家，第 i 个战斗力为 a_i。
m 个敌人，第 i 个战斗力为 b_i，赏金为 c_i。
每个玩家可以消灭一个敌人也可以什么都不做。玩家 i 能消灭敌人 j 的条件是 $a_i > b_j$，消灭后能获得 $a_i - b_j + c_j$ 的收益。最大化总收益。

首先我们按 b 从小到大排序，对每个敌人建一个点 $P_1 \dots P_n$，然后我们加边 $(P_i, t, 1, b_i - c_i)$，$(P_i, P_{i-1}, \infty, b_i)$。对于每一个玩家，我们二分出他应该的位置 j，连边 $(s, P_j, 1, -a_i)$。

首先我们按 b 从小到大排序，对每个敌人建一个点 $P_1 \dots P_n$，然后我们加边 $(P_i, t, 1, b_i - c_i)$，$(P_i, P_{i-1}, \infty, b_i)$。对于每一个玩家，我们二分出他应该的位置 j，连边 $(s, P_j, 1, -a_i)$。
考虑线段树维护每条边的流量，我们先加入所有与 t 相连的边和链上的边，然后从 P_1 到 P_n 加入与 s 相关的边。

首先我们按 b 从小到大排序，对每个敌人建一个点 $P_1 \dots P_n$，然后我们加边 $(P_i, t, 1, b_i - c_i)$，$(P_i, P_{i-1}, \infty, b_i)$。对于每一个玩家，我们二分出他应该的位置 j，连边 $(s, P_j, 1, -a_i)$。

考虑线段树维护每条边的流量，我们先加入所有与 t 相连的边和链上的边，然后从 P_1 到 P_n 加入与 s 相关的边。

此时你加入的边可能会导致负环，但我们可以将 $(s, P_j, 1, -a_i)$ 这种已经满流的边改成 $(P_j, t, 1, a_i)$，反之同理，容易发现这样就没有负环了。也可以不改，这时我们需要考虑消圈。这二者本质相同。

「Lydsy1708 月赛」跳伞求生

首先我们按 b 从小到大排序，对每个敌人建一个点 $P_1 \dots P_n$，然后我们加边 $(P_i, t, 1, b_i - c_i)$，$(P_i, P_{i-1}, \infty, b_i)$。对于每一个玩家，我们二分出他应该的位置 j，连边 $(s, P_j, 1, -a_i)$。

考虑线段树维护每条边的流量，我们先加入所有与 t 相连的边和链上的边，然后从 P_1 到 P_n 加入与 s 相关的边。

此时你加入的边可能会导致负环，但我们可以将 $(s, P_j, 1, -a_i)$ 这种已经满流的边改成 $(P_j, t, 1, a_i)$，反之同理，容易发现这样就没有负环了。也可以不改，这时我们需要考虑消圈。这二者本质相同。

由于中间的链上的容量是无穷，每个点能到的点集一定是这个点的前缀。我们可以直接用堆维护前缀的边的情况（如果中间链的容量有限那么我们就需要使用线段树来维护容量来确定每个点能到的点集了）。

考虑另一种维护凸包的做法。

「Lydsy1708 月赛」跳伞求生

考虑另一种维护凸包的做法。

我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。

「Lydsy1708 月赛」跳伞求生

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。

「Lydsy1708 月赛」跳伞求生

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。
若新加入的边表示一个费用为 x 的玩家，则有转移方程：

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。
若新加入的边表示一个费用为 x 的玩家，则有转移方程：

$$f_{i,j} = \begin{cases} \min(f_{i-1,j-1} + x, f_{i-1,j}) & (j > 0) \\ f_{i-1,j} & (j = 0) \end{cases}$$

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。
若新加入的边表示一个费用为 x 的玩家，则有转移方程：

$$f_{i,j}=\begin{cases}\min(f_{i-1,j-1}+x,f_{i-1,j}) & (j>0)\\ f_{i-1,j} & (j=0)\end{cases}$$

设 $g_{i,j}=f_{i,j}-f_{i,j-1}(j>0)$，因为 $f_{i,j}$ 关于 j 是凸的，所以 $g_{i,j}$ 单调不降。则可以列出 g 的转移方程：

「Lydsy1708 月赛」跳伞求生

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。
若新加入的边表示一个费用为 x 的玩家，则有转移方程：

$$f_{i,j}=\begin{cases}\min(f_{i-1,j-1}+x,f_{i-1,j}) & (j>0)\\ f_{i-1,j} & (j=0)\end{cases}$$

设 $g_{i,j}=f_{i,j}-f_{i,j-1}(j>0)$，因为 $f_{i,j}$ 关于 j 是凸的，所以 $g_{i,j}$ 单调不降。则可以列出 g 的转移方程：

$$g_{i,j}=\begin{cases}g_{i-1,j-1} & (j>1\ \wedge\ x<g_{i-1,j-1})\\ x & ((j=1\ \vee\ g_{i-1,j-1}\le x)\ \wedge\ x<g_{i-1,j})\\ g_{i-1,j} & (x\ge g_{i-1,j})\end{cases}$$

「Lydsy1708 月赛」跳伞求生

考虑另一种维护凸包的做法。
我们从前往后加入与 s 和 t 相关的边。在加到第 i 个点的时候，我们维护费用关于多余的流量的函数。
设 $f_{i,j}$ 表示加入了前 i 条边，剩余流量为 j 时的最小费用。
若新加入的边表示一个费用为 x 的玩家，则有转移方程：

$$f_{i,j}=\begin{cases}\min(f_{i-1,j-1}+x,f_{i-1,j}) & (j>0)\\ f_{i-1,j} & (j=0)\end{cases}$$

设 $g_{i,j}=f_{i,j}-f_{i,j-1}(j>0)$，因为 $f_{i,j}$ 关于 j 是凸的，所以 $g_{i,j}$ 单调不降。则可以列出 g 的转移方程：

$$g_{i,j}=\begin{cases}g_{i-1,j-1} & (j>1\ \wedge\ x<g_{i-1,j-1})\\ x & ((j=1\ \vee\ g_{i-1,j-1}\le x)\ \wedge\ x<g_{i-1,j})\\ g_{i-1,j} & (x\ge g_{i-1,j})\end{cases}$$

这相当于插入 x 再给 g 排序。

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若 $-x < g_{i-1,1}$，则 f 不变。

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若 $-x < g_{i-1,1}$，则 f 不变。
若 $-x \geq g_{i-1,1}$，则

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若 $-x < g_{i-1,1}$，则 f 不变。
若 $-x \geq g_{i-1,1}$，则

$$f_{i,0} = f_{i-1,0} + x + g_{i-1,1}$$

$$g_{i,j} = \begin{cases} g_{i-1,j+1} & (-x \geq g_{i-1,j+1}) \\ -x & (g_{i-1,j} \leq -x < g_{i-1,j+1}) \\ g_{i-1,j} & (-x < g_{i-1,j}) \end{cases}$$

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若 $-x < g_{i-1,1}$，则 f 不变。
若 $-x \geq g_{i-1,1}$，则

$$f_{i,0} = f_{i-1,0} + x + g_{i-1,1}$$

$$g_{i,j} = \begin{cases} g_{i-1,j+1} & (-x \geq g_{i-1,j+1}) \\ -x & (g_{i-1,j} \leq -x < g_{i-1,j+1}) \\ g_{i-1,j} & (-x < g_{i-1,j}) \end{cases}$$

容易发现这实质是删除 $g_{i-1,1}$ 再插入 $-x$ 然后排序。

若新加入的边表示一个费用为 x 的敌人，同样我们可以得到：

$$f_{i,j} = \min(f_{i-1,j}, f_{i-1,j+1} + x)$$

若 $-x < g_{i-1,1}$，则 f 不变。
若 $-x \geq g_{i-1,1}$，则

$$f_{i,0} = f_{i-1,0} + x + g_{i-1,1}$$

$$g_{i,j} = \begin{cases} g_{i-1,j+1} & (-x \geq g_{i-1,j+1}) \\ -x & (g_{i-1,j} \leq -x < g_{i-1,j+1}) \\ g_{i-1,j} & (-x < g_{i-1,j}) \end{cases}$$

容易发现这实质是删除 $g_{i-1,1}$ 再插入 $-x$ 然后排序。
因此我们只需要用堆来维护 g 即可。

「全国統一プログラミング王決定戦本戦」Homework Scheduling

有 n 个任务，第 i 个任务有一个截止日期 a_i。在截止日期前完成可以获得 x_i 的收益，否则可以获得 y_i 的收益。你每天最多完成一个任务。

「全国統一プログラミング王決定戦本戦」Homework Scheduling

有 n 个任务，第 i 个任务有一个截止日期 a_i。在截止日期前完成可以获得 x_i 的收益，否则可以获得 y_i 的收益。你每天最多完成一个任务。
对于每一个 $1 \leq t \leq n$，求出只在前 t 天工作的最大收益。

「全国統一プログラミング王決定戦本戦」Homework Scheduling

此时凸包做法是非常不可做的，我们考虑用线段树。

此时凸包做法是非常不可做的，我们考虑用线段树。
我们考虑对每一种任务建一个点 $X_1, \ldots, X_n$，对每一天也建点 $P_1, \ldots, P_n$，然后建边 $(P_i, P_{i-1}, +\infty, 0)$，$(X_i, P_{a_i}, 1, -x_i)$，$(X_i, P_n, 1, -y_i)$，$(s, X_i, 1, 0)$。

此时凸包做法是非常不可做的，我们考虑用线段树。
我们考虑对每一种任务建一个点 $X_1, \ldots, X_n$，对每一天也建点 $P_1, \ldots, P_n$，然后建边 $(P_i, P_{i-1}, +\infty, 0)$，$(X_i, P_{a_i}, 1, -x_i)$，$(X_i, P_n, 1, -y_i)$，$(s, X_i, 1, 0)$。
然后我们考虑一个一个把 $(P_i, t, 1, 0)$ 往图中加入并维护最大收益。此时新加入的边并不会导致负环，因此我们只考虑 s 到 t 的增广路。

「全国統一プログラミング王決定戦本戦」Homework Scheduling

此时凸包做法是非常不可做的，我们考虑用线段树。
我们考虑对每一种任务建一个点 $X_1, \ldots, X_n$，对每一天也建点 $P_1, \ldots, P_n$，然后建边 $(P_i, P_{i-1}, +\infty, 0)$，$(X_i, P_{a_i}, 1, -x_i)$，$(X_i, P_n, 1, -y_i)$，$(s, X_i, 1, 0)$。
然后我们考虑一个一个把 $(P_i, t, 1, 0)$ 往图中加入并维护最大收益。此时新加入的边并不会导致负环，因此我们只考虑 s 到 t 的增广路。
同时我们可以根据最短路无环得出如下性质：每一次我们最多只会把一个 X_i 从选择 x_i 变为选择 y_i，同时已经选择 y_i 的 X_i 不会再变动。

此时凸包做法是非常不可做的，我们考虑用线段树。
我们考虑对每一种任务建一个点 $X_1, \ldots, X_n$，对每一天也建点 $P_1, \ldots, P_n$，然后建边 $(P_i, P_{i-1}, +\infty, 0)$，$(X_i, P_{a_i}, 1, -x_i)$，$(X_i, P_n, 1, -y_i)$，$(s, X_i, 1, 0)$。
然后我们考虑一个一个把 $(P_i, t, 1, 0)$ 往图中加入并维护最大收益。此时新加入的边并不会导致负环，因此我们只考虑 s 到 t 的增广路。
同时我们可以根据最短路无环得出如下性质：每一次我们最多只会把一个 X_i 从选择 x_i 变为选择 y_i，同时已经选择 y_i 的 X_i 不会再变动。
这时我们就可以用线段树维护每一个 X_i 到每个 P_i 的最小费用了。我们可以求出每一条边的流量，增广时相当于区间加/减。查询 s 到这个 P_i 的最短距离可以全部通过线段树来完成。

在单向链的基础上，我们增加 P_{i+1} 到 P_i 的边，此时中间的链就变成双向的。在该图上我们可以继续考虑之前的两种做法。

「CTSC2010」产品销售

n 个季度，第 i 个季度最多可以生产 U_i 件产品，每件成本为 P_i。

「CTSC2010」产品销售

n 个季度，第 i 个季度最多可以生产 U_i 件产品，每件成本为 P_i。
一件产品从第 i 个季度保存到第 $i+1$ 个季度需要花费 M_i 的存储费用。
第 i 个季度订购量为 D_i，对于第 i 个季度的订购量如果没能完成需要对每件产品赔偿 C_i 的费用并将剩余的订购量推迟到下一季度（到了下个季度可以继续推迟）。

「CTSC2010」产品销售

n 个季度，第 i 个季度最多可以生产 U_i 件产品，每件成本为 P_i。
一件产品从第 i 个季度保存到第 $i+1$ 个季度需要花费 M_i 的存储费用。
第 i 个季度订购量为 D_i，对于第 i 个季度的订购量如果没能完成需要对每件产品赔偿 C_i 的费用并将剩余的订购量推迟到下一季度（到了下个季度可以继续推迟）。
求满足所有订购需求的最小花费。

「CTSC2010」产品销售

首先我们可以建图，我们为每一天建一个点 $X_1 \dots X_n$，然后连边 (s, X_i, U_i, P_i)，$(X_i, X_{i+1}, +\infty, M_i)$，$(X_{i+1}, X_i, +\infty, C_i)$，$(X_i, n, +\infty, -\infty)$。

「CTSC2010」产品销售

首先我们可以建图，我们为每一天建一个点 $X_1 \dots X_n$，然后连边 (s, X_i, U_i, P_i)，$(X_i, X_{i+1}, +\infty, M_i)$，$(X_{i+1}, X_i, +\infty, C_i)$，$(X_i, n, +\infty, -\infty)$。

考虑线段树的做法，同样我们从 X_1 到 X_n 加入 s 的出边，我们使用维护每一条边的流量以及当前点到每个点的费用，其中一个点往后能到的点的费用是好算的，而往前的费用取决与之中的反向边。

「CTSC2010」产品销售

首先我们可以建图，我们为每一天建一个点 $X_1 \dots X_n$，然后连边 (s, X_i, U_i, P_i)，$(X_i, X_{i+1}, +\infty, M_i)$，$(X_{i+1}, X_i, +\infty, C_i)$，$(X_i, n, +\infty, -\infty)$。

考虑线段树的做法，同样我们从 X_1 到 X_n 加入 s 的出边，我们使用维护每一条边的流量以及当前点到每个点的费用，其中一个点往后能到的点的费用是好算的，而往前的费用取决与之中的反向边。

容易发现一条向右的边只有在加入一个前缀的 X 的时候才可能被增广，产生向左的反向边，而该反向边仅在继续加入这个 X 的后缀的时候才可能继续被增广直到消失。因此只需要在一个向右的边最初被增广以及反向边被增广完的时候处理前缀的每个点的费用即可。

「CTSC2010」产品销售

首先我们可以建图，我们为每一天建一个点 $X_1 \dots X_n$，然后连边 (s, X_i, U_i, P_i)，$(X_i, X_{i+1}, +\infty, M_i)$，$(X_{i+1}, X_i, +\infty, C_i)$，$(X_i, n, +\infty, -\infty)$。

考虑线段树的做法，同样我们从 X_1 到 X_n 加入 s 的出边，我们使用维护每一条边的流量以及当前点到每个点的费用，其中一个点往后能到的点的费用是好算的，而往前的费用取决与之中的反向边。

容易发现一条向右的边只有在加入一个前缀的 X 的时候才可能被增广，产生向左的反向边，而该反向边仅在继续加入这个 X 的后缀的时候才可能继续被增广直到消失。因此只需要在一个向右的边最初被增广以及反向边被增广完的时候处理前缀的每个点的费用即可。

加边时可能会产生负环，我们可以采取与单向链完全相同的处理方法。

整个过程均可用线段树维护。

「CTSC2010」产品销售

考虑维护凸包的做法，我们设 f_j 表示剩余流量为 j（如果 $j<0$ 则说明后面的点有流量流过来）时大小为 i 的前缀的最小费用（此时 i 到 $i+1$ 之间的费用也要计算。

「CTSC2010」产品销售

考虑维护凸包的做法，我们设 f_j 表示剩余流量为 j（如果 $j<0$ 则说明后面的点有流量流过来）时大小为 i 的前缀的最小费用（此时 i 到 $i+1$ 之间的费用也要计算。
此时我们在这里设一个数组 $h_j=-([j<0]C_i+[j>0]M_i)j$，则我们发现 f_j+h_j 关于 j 的函数也是凸的，因为这种操作相当于把中间边的费用变成 0。

「CTSC2010」产品销售

考虑如果我们的流量是一个一个加入的，由于入边出边是对称的，这里只考虑加入边的情况。

「CTSC2010」产品销售

考虑如果我们的流量是一个一个加入的，由于入边出边是对称的，这里只考虑加入边的情况。
我们考虑加一个入边后，f 会更新为 f'。则我们可以得到转移方程：

「CTSC2010」产品销售

考虑如果我们的流量是一个一个加入的，由于入边出边是对称的，这里只考虑加入边的情况。
我们考虑加一个入边后，f 会更新为 f'。则我们可以得到转移方程：

$$f'_j = \begin{cases} \min(f_j, f_{j-1} + P_i + M_i) & (j > 0) \\ \min(f_j, f_{j-1} + P_i - C_i) & (j \leq 0) \end{cases}$$

「CTSC2010」产品销售

考虑如果我们的流量是一个一个加入的，由于入边出边是对称的，这里只考虑加入边的情况。
我们考虑加一个入边后，f 会更新为 f'。则我们可以得到转移方程：

$$f'_j = \begin{cases} \min(f_j, f_{j-1} + P_i + M_i) & (j > 0) \\ \min(f_j, f_{j-1} + P_i - C_i) & (j \leq 0) \end{cases}$$

同时我们定义差分数组 g，设

「CTSC2010」产品销售

考虑如果我们的流量是一个一个加入的，由于入边出边是对称的，这里只考虑加入边的情况。
我们考虑加一个入边后，f 会更新为 f'。则我们可以得到转移方程：

$$f'_j = \begin{cases} \min(f_j, f_{j-1} + P_i + M_i) & (j > 0) \\ \min(f_j, f_{j-1} + P_i - C_i) & (j \le 0) \end{cases}$$

同时我们定义差分数组 g，设

$$g_j = \begin{cases} f_j - f_{j-1} & (j > 0) \\ 0 & (j = 0) \\ f_j - f_{j+1} & (j < 0) \end{cases}$$

「CTSC2010」产品销售

接着我们讨论两种情况：

「CTSC2010」产品销售

接着我们讨论两种情况：

1. $f_0 \leq f_{-1} + P_i - C_i$

此时对于 $j < 0$ 的 g_j 是不变的。

对于 $j > 0$ 的部分，该部分类似单向链，即插入 $P_i + M_i$ 并排序。

「CTSC2010」产品销售

接着我们讨论两种情况：

1. $f_0 \le f_{-1} + P_i - C_i$

此时对于 $j < 0$ 的 g_j 是不变的。

对于 $j > 0$ 的部分，该部分类似单向链，即插入 $P_i + M_i$ 并排序。

2. $f_0 > f_{-1} + P_i - C_i$，等价于 $g_{-1} < C_i - P_i$。

此时对于 $j < 0$ 的部分类似单向链，即插入 $C_i - P_i$ 并删除最小值后排序。

此时对于 $j > 0$ 的部分，由于 f_0 会被 f_{-1} 修改，

这时有：$f'_0 = f_0 + g_{-1} + P_i - C_i$，由 $f + h$ 是凸函数的性质，我们可以得到：$f_0 - f_{-1} - C_i \le f_1 - f_0 - M_i$。

所以 $f_1 \ge f_0 + M_i - C_i - g_{-1} > f_0 + M_i + P_i$，因此 $f'_1 = f_0 + P_i + M_i$。

所以 $g'_1 = f'_1 - f'_0 = -g_{-1} - C_i - M_i$，而对于 $j > 0$，有 $g_j \ge g_1 \ge M_i + P_i$。

因此我们相当于加入 $-g_{-1} - M_i - C_i$ 并将 g 大于 0 的部分排序。

「CTSC2010」产品销售

接着我们讨论两种情况：

1. $f_0 \le f_{-1} + P_i - C_i$

此时对于 $j < 0$ 的 g_j 是不变的。

对于 $j > 0$ 的部分，该部分类似单向链，即插入 $P_i + M_i$ 并排序。

2. $f_0 > f_{-1} + P_i - C_i$，等价于 $g_{-1} < C_i - P_i$。

此时对于 $j < 0$ 的部分类似单向链，即插入 $C_i - P_i$ 并删除最小值后排序。

此时对于 $j > 0$ 的部分，由于 f_0 会被 f_{-1} 修改，

这时有：$f'_0 = f_0 + g_{-1} + P_i - C_i$，由 $f + h$ 是凸函数的性质，我们可以得到：$f_0 - f_{-1} - C_i \le f_1 - f_0 - M_i$。

所以 $f_1 \ge f_0 + M_i - C_i - g_{-1} > f_0 + M_i + P_i$，因此 $f'_1 = f_0 + P_i + M_i$。

所以 $g'_1 = f'_1 - f'_0 = -g_{-1} - C_i - M_i$，而对于 $j > 0$，有 $g_j \ge g_1 \ge M_i + P_i$。

因此我们相当于加入 $-g_{-1} - M_i - C_i$ 并将 g 大于 0 的部分排序。

当 i 增加时，我们的操作就相当于给差分数组大于 0 和小于 0 的部分分别全体加。

「CTSC2010」产品销售

接着我们讨论两种情况：

1. $f_0 \le f_{-1} + P_i - C_i$

此时对于 $j < 0$ 的 g_j 是不变的。

对于 $j > 0$ 的部分，该部分类似单向链，即插入 $P_i + M_i$ 并排序。

2. $f_0 > f_{-1} + P_i - C_i$，等价于 $g_{-1} < C_i - P_i$。

此时对于 $j < 0$ 的部分类似单向链，即插入 $C_i - P_i$ 并删除最小值后排序。

此时对于 $j > 0$ 的部分，由于 f_0 会被 f_{-1} 修改，

这时有：$f'_0 = f_0 + g_{-1} + P_i - C_i$，由 $f + h$ 是凸函数的性质，我们可以得到：$f_0 - f_{-1} - C_i \le f_1 - f_0 - M_i$。

所以 $f_1 \ge f_0 + M_i - C_i - g_{-1} > f_0 + M_i + P_i$，因此 $f'_1 = f_0 + P_i + M_i$。

所以 $g'_1 = f'_1 - f'_0 = -g_{-1} - C_i - M_i$，而对于 $j > 0$，有 $g_j \ge g_1 \ge M_i + P_i$。

因此我们相当于加入 $-g_{-1} - M_i - C_i$ 并将 g 大于 0 的部分排序。

当 i 增加时，我们的操作就相当于给差分数组大于 0 和小于 0 的部分分别全体加。

因为需要批量加入，我们可以用平衡树维护过程。

我们可以考虑问题放到树上的情况，我们可以继续沿用之前的两种做法。

「ICPC World Finals 2018」征服世界

一棵由无向边连接的树，初始时第 i 个点有 x_i 个军队，一个军队沿着一条代价为 $c(c > 0)$ 的边移动到另一个点花费的代价为 c。最终要对于每个点 i，其军队个数不小于 y_i。求最小代价。

「ICPC World Finals 2018」征服世界

容易建出费用流图，我们考虑两种解决的办法。

容易建出费用流图，我们考虑两种解决的办法。
1. 我们可以按照 dfs 序来加入与源点相连的边，按照这种顺序可以使得反向边变化总次数为 $O(1)$。用树剖等办法维护流量即可。

容易建出费用流图，我们考虑两种解决的办法。
1. 我们可以按照 dfs 序来加入与源点相连的边，按照这种顺序可以使得反向边变化总次数为 $O(1)$。用树剖等办法维护流量即可。
2. 考虑对每一个点维护到父亲节点剩余流量为 j 时的最小代价 f_j，合并子树的转移方程形如 $f_j = \min f'_k + f''_{j-k}$，实质相当于做闵可夫斯基和。加入边与出边的方式与上文类似，可用可并堆维护。