

线性规划 && 单纯形算法 && 对偶原理 学习笔记

参考资料

[2016年国家集训队论文](#)

一些定义

线性函数

已知一组实数 a_1, a_2, \dots, a_n 和一组变量 x_1, x_2, \dots, x_n ，给出定义在这些变量上的一个**线性函数**形如：

$$f(x_1, x_2, \dots, x_n) = \sum_{i=1}^n a_i x_i$$

线性等式

形如：

$$f(x_1, x_2, \dots, x_n) = b$$

的等式

线性不等式

形如：

$$\begin{aligned} f(x_1, x_2, \dots, x_n) &\leq b \\ f(x_1, x_2, \dots, x_n) &\geq b \end{aligned}$$

的不等式

线性规划的一般形式

一个**线性规划**问题是一个线性函数最大化或最小化的问题，该线性函数服从一组有限个**线性约束**

线性规划中**不允许**不严格的不等式，即**所有约束都包含等号**

一些定义

称所有满足约束的变量 x_1, x_2, \dots, x_n 的取值为线性规划的一个**可行解**，不满足约束的称为**不可行解**

如果在坐标系中画出这些约束，则可以看到可行解的集合构成一个**凸区域**，称为**可行区域**

希望最大化的函数称为**目标函数**，目标函数上一个特定点的值称为**目标值**

若一个线性规划没有可行解，则称其为**不可行的**，否则称为**可行的**

若一个可行的线性规划不存在**最优目标值**，则称其为**无界的**

线性规划的标准型

在**标准型**中，我们已知 n 个实数 c_1, c_2, \dots, c_n ， m 个实数 b_1, b_2, \dots, b_m ，以及 mn 个实数 a_{ij} ，其中 $i \in [1, m]$, $j \in [1, n]$

我们希望找到 n 个实数 x_1, x_2, \dots, x_n ，最大化**目标函数**：

$$\sum_{j=1}^n c_j x_j$$

满足**约束**：

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \forall i \in [1, m]$$

和**非负约束**：

$$x_j \geq 0, \forall j \in [1, n]$$

写成矩阵形式

$$\begin{aligned} & \max c^T x \\ & Ax \leq b \\ & x \geq 0 \end{aligned}$$

化线性规划为标准型

- 目标函数为**最小化**，只需对目标函数中的**系数取负**（**对偶原理**）
- 有的变量 x_j 不具有**非负约束**，只需将其所有出现的地方换为 x'_j, x''_j ，并增加**非负约束** $x'_j, x''_j \geq 0$
- 存在**等式约束** $f(x_1, x_2, \dots, x_n) = b$ ，只需化为 $f(x_1, x_2, \dots, x_n) \leq b \wedge f(x_1, x_2, \dots, x_n) \geq b$
- 存在**不等式约束** $f(x_1, x_2, \dots, x_n) \geq b$ ，只需两边同乘 -1 ，化为 $f(x_1, x_2, \dots, x_n) \leq b$ 的形式

线性规划的松弛型 && 化标准型为松弛型

为了利用单纯性算法更高效地求解线性规划问题，我们更喜欢将其所有的**不等式约束变为等式约束**

设：

$$\sum_{j=1}^n a_{ij} x_j \leq b_i$$

是一个不等式约束，我们引入一个新的变量 s ，并重写不等式为两个约束：

$$\begin{aligned} s &= b_i - \sum_{j=1}^n a_{ij} x_j \\ s &\geq 0 \end{aligned}$$

我们称 s 为一个**松弛变量**，因为它度量了不等式左右之间的**松弛或差别**

现在，我们把每个等式约束写成一个变量在等式左边，其余变量都在等式右边的形式

并且每个等式右边拥有相同的变量集合，且这些变量也是在**目标函数中仅有的变量**

称等式左边的变量为**基本变量**，右边的为**非基本变量**

对满足这些条件的线性规划，我们有时会省略词语「最大化」和「满足约束」，以及明显的非负约束要求

我们也会使用变量 z 来表示目标函数值，称这样导出的形式为**松弛型**

用简洁的记号表示

用 N 表示非基本变量下标的集合， B 表示基本变量下标的集合

b_i, c_j, a_{ij} 表示常数项和系数， v 表示目标函数的一个可选常数项

则我们可以用一个 6 元组 (N, B, A, b, c, v) 来表示松弛型：

$$z = v + \sum_{j \in N} c_j x_j$$
$$x_i = b_i - \sum_{j \in N} a_{ij} x_j, i \in B$$

单纯形算法

基本解

单纯形算法由若干轮迭代组成，每轮迭代关联一个**基本解**：将每个非基本变量设为 0

若一个基本解是可行的，则称其为**基本可行解**

转动 pivot

在每轮迭代中，我们会把一个松弛型改写成一个等价的松弛型，其关联的基本可行解的目标值不会小于上一轮

为了增大目标值，我们选择一个非基本变量，使得如果增加它的值，目标值也会增加

特别地，我们增加它，直至某基本变量变为 0，然后重写松弛型，交换此基本变量和**选定的**非基本变量

我们称此操作为一个**转动**，一个转动选择一个非基本变量 x_e （称为**替入变量**）和一个基本变量 x_l （称为**替出变量**），然后交换二者角色

如何判断最优

松弛型：

$$z = v + \sum_{j \in N} c_j x_j$$

是最优的，当且仅当 $\forall j \in N, c_j \leq 0$

证明：基本解代表 n 维空间中的原点，是可行域的边界

若 $\exists c_j \geq 0$ ，则我们可以增大 x_j ，此时可行解逐渐离开这个边界，且目标值会增大

如何选择替入变量

若存在多个 $c_j > 0$ ，我们应该选择最大的那个

如何选择替出变量

我们应该选择对当前非基本变量约束最「紧」的那个基本变量

终止条件

在单纯形算法执行的过程中，可能会在迭代过程中目标值不变的现象，称为**退化**

退化可能引起一种称为**循环**的现象，他会阻止单纯形算法的终止

根据**Bland 规则**，我们可以选择下标最小的变量，这样可以避免循环的出现

单纯型 simplex

现在我们已经知道，在基本可行解的前提下，如何找到线性规划的最优解

初始化

现在要找到一个基本可行解，若 $\forall i \in [1, m], b_i \geq 0$ 则可以跳过这一步，否则可以引入一个**辅助线性规划**：

$$\begin{aligned} \max z &= -x_0 \\ x_{i+n} &= b_i - \sum_{j=1}^n a_{ij}x_j + x_0 \end{aligned}$$

如果原线性规划存在一个可行解 $(\bar{x}_1, \bar{x}_2, \dots, \bar{x}_{n+m})$ ，则 $(0, \bar{x}_1, \bar{x}_2, \dots, \bar{x}_{n+m})$ 就是辅助线性规划的一个可行解

由于 $x_0 \geq 0$ ，故这个可行解就是辅助线性规划的最优解

另一方面，若求得辅助线性规划的最优解 $x_0 = 0$ ，则直接将 x_0 从这个线性规划中删去不会对约束产生影响

而辅助线性规划的初始解是容易构造的

由于 x_0 在每个约束中的系数都为 1，故我们把 x_0 作为替入变量

找到 b_i 的最小值 b_l ，把 x_{l+n} 作为替出变量执行一次转动操作

操作后，第 l 个约束变为：

$$x_0 = -b_l + \sum_{j=1}^n a_{lj}x_j + x_{l+n}$$

满足 $-b_l > 0$ ，其余约束变为：

$$x_{i+n} = -b_l + b_i + \sum_{j=1}^n (a_{lj} - a_{ij})x_j + x_{l+n} \quad (i \neq l)$$

由于 $b_l \leq b_i$ ，故 $-b_l + b_i \geq 0$ ，故此时为一个基本可行解

伪代码

时间复杂度

尽管单纯形算法的时间复杂度不是多项式的，但在实际应用中，他能较快地给出最优解

不过实践中，最好深入分析题目的性质，利用下面的对偶原理，将线性规划问题简化，从而利用其他算法解决，得到多项式的时间复杂度

代码实现

[Uoj #179. 线性规划](#)

在实现时，初始化可以不用上面的方法，若存在 $b_i < 0$ ，则找到一个 $a_{i,j} < 0$ ，然后 $\text{Pivot}(i, j)$ ，直到所有 $b_i > 0$

```
ci N=42;
const D eps=1e-8,inf=1e9;
D a[N][N],ans[N];
int n,m,tp,id[N];
mt19937 rd(time(0));
void Pivot(ci l,ci e){//转动
    swap(id[n+1],id[e]);
    D t=a[l][e];a[l][e]=1;
    for(int i=0;i<=n;++i)a[l][i]/=t;
    for(int i=0;i<=m;++i)
        if(i!=l&&fabs(a[i][e])>eps){
            t=a[i][e],a[i][e]=0;
            for(int j=0;j<=n;++j)a[i][j]-=a[l][j]*t;
        }
}
bool init(){//初始化
    while(1){
        int l=0,e=0;
        for(int i=1;i<=m;++i)if(a[0][i]<-eps&&(!l||rd()&1))l=i;
        if(!l)break;
        for(int i=1;i<=n;++i)if(a[l][i]<-eps&&(!e||rd()&1))e=i;
        if(!e){//若所有a[l][i]>=0，则无解
            puts("Infeasible");
            return 0;
        }
        Pivot(l,e);
    }
    return 1;
}
bool simplex(){
    while(1){
        int l=0,e=1;
        for(int i=2;i<=n;++i)if(a[0][i]>a[0][e])e=i;
        if(a[0][e]<eps)break;
        D mn=inf;
        for(int i=1;i<=m;++i)
            if(a[i][e]>eps&&a[i][0]/a[i][e]<mn)
                mn=a[i][0]/a[i][e],l=i;
    }
}
```

```

if(!l){//若没有限制，则无界
    puts("Unbounded");
    return 0;
}
Pivot(l,e);
}
return 1;
}

int main(){
scanf("%d%d%d",&n,&m,&tp);
for(int i=1;i<=n;++i)scanf("%lf",&a[0][i]);
for(int i=1;i<=m;++i){
    for(int j=1;j<=n;++j)scanf("%lf",&a[i][j]);
    scanf("%lf",&a[i][0]);
}
for(int i=1;i<=n;++i)id[i]=i;
if(init()&&Simplex()){
    printf("%.8lf\n",-a[0][0]);//注意这里是负数
    if(tp){//id[i]记录的是当前位置原来是哪个变量
        for(int i=1;i<=m;++i)ans[id[i+n]]=a[i][0];
        for(int i=1;i<=n;++i)printf("%.8lf ",ans[i]);
    }
}
return 0;
}

```

整数线性规划问题

整数线性规划问题要求 $x_i \in \mathbb{N}$ ，它是一个 NP 完全问题

但是许多整数线性规划问题保证了当自变量取值范围是整数和实数时是一样的，如：网络流问题

应用

最短路问题

$$\begin{aligned}
& \max d_t \\
d_v & \leq d_u + w(u, v) \quad (u, v) \in E \\
d_s & = 0
\end{aligned}$$

最大流问题

增加一条边 (t, s) ，使最大流变成循环流

$$\begin{aligned}
& \max f(t, s) \\
f(u, v) & \leq c(u, v) \quad (u, v) \in E \\
\sum_v f(u, v) & = \sum_v f(v, u) \quad u \in V \\
f(u, v) & \geq 0 \quad (u, v) \in E \cup \{(t, s)\}
\end{aligned}$$

最小费用流问题

这里并不要求流量最大，只要求费用最大

$$\begin{aligned} \min & \sum_{(u,v) \in E} f(u,v) w(u,v) \\ & f(u,v) \leq c(u,v) \quad (u,v) \in E \\ & \sum_v f(u,v) = \sum_v f(v,u) \quad u \in V - \{s,t\} \\ & f(u,v) \geq 0 \quad (u,v) \in E \cup \{(t,s)\} \end{aligned}$$

多物网络流问题

$$\begin{aligned} \max & 0 \\ & \sum_i f_i(u,v) \leq c(u,v) \quad (u,v) \in E \\ & \sum_v f_i(u,v) = \sum_v f_i(v,u) \quad u \in V, i \in [1, n] \\ & f_i(t_i, s_i) \geq d_i \quad i \in [1, n] \\ & f_i(u,v) \geq 0 \quad (u,v) \in E \cup \{(t_i, s_i) | 1 \leq i \leq n\} \end{aligned}$$

对偶原理

对偶线性规划

给定一个原始线性规划：

$$\begin{aligned} \max z &= \sum_{j=1}^n c_j x_j \\ & \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad \forall i \in [1, m] \\ & x_j \geq 0, \quad \forall i \in [1, n] \end{aligned}$$

这是标准型，我们定义它的对偶线性规划为：

$$\begin{aligned} \min z &= \sum_{i=1}^m b_i y_i \\ & \sum_{i=1}^m a_{ij} y_i \geq c_j, \quad \forall j \in [1, n] \\ & y_i \geq 0, \quad \forall i \in [1, m] \end{aligned}$$

即：

- 改最大化为最小化
- 交换右边系数和目标函数系数
- 改 \leq 为 \geq

原始线性规划的每个约束在对偶线性规划中对应一个变量

对偶线性规划的每个约束在原始线性规划中对应一个变量

可以看做将矩阵 a 转置变成 a^T

线性规划弱对偶性

原始线性规划的任意一个可行解的目标值都不超过对偶线性规划任意一个可行解的目标值，即：

$$\sum_{j=1}^n c_j \bar{x}_j \leq \sum_{i=1}^m b_i \bar{y}_i$$

证明：

$$\begin{aligned}\sum_{j=1}^n c_j \bar{x}_j &\leq \sum_{j=1}^n \left(\sum_{i=1}^m a_{ij} \bar{y}_i \right) \bar{x}_j \\&= \sum_{i=1}^m \left(\sum_{j=1}^n a_{ij} \bar{x}_j \right) \bar{y}_i \\&\leq \sum_{i=1}^m b_i \bar{y}_i\end{aligned}$$

线性规划对偶性

若：

$$\sum_{j=1}^n c_j \bar{x}_j = \sum_{i=1}^m b_i \bar{y}_i$$

则 \bar{x}, \bar{y} 分别是原始线性规划和对偶线性规划的最优解

特别的，原始线性规划的无解对应对偶线性规划的无界，原始线性规划的无界对应对偶线性规划的无解

互松弛定理

若 \bar{x}, \bar{y} 分别是原始线性规划和对偶线性规划的可行解，则它们是最优解当且仅当下面两个条件同时满足：

- $\forall 1 \leq j \leq n$ ，有 $x_j = 0$ 或 $\sum_{i=1}^m a_{i,j} y_i = c_j$
- $\forall 1 \leq i \leq m$ ，有 $y_i = 0$ 或 $\sum_{j=1}^n a_{i,j} x_j = b_i$

这是使线性规划弱对偶性证明中两个不等式取到等号的条件

这可以用于求原始线性规划的方案

应用

化最小线性规划为最大线性规划

CF375E Red and Black Tree

题意：给定一棵 n 个节点的树，每个节点为红色或黑色，可以交换两个节点的颜色

最少交换多少次使得每个点都存在一个与它距离 $\leq d$ 的黑点。 $n \leq 500$

思路：

$$\begin{aligned}
& \min \sum_{i=1}^n red_i x_i \\
& \sum_{j=1}^n A_{i,j} x_j \geq 1 \quad \forall i \in [1, n] \\
& \sum_{i=1}^n x_i \geq Black \\
& - \sum_{i=1}^n x_i \geq -Black \\
& x_i \in \{0, 1\} \quad i \in [1, n]
\end{aligned}$$

我们需要证明其对应的普通线性规划的解不会更优

- 若存在 $x_i \geq 1$, 则我们可以把它的值分配给其它原来是黑色的点, 并使其变为 1
-

故我们不需考虑 $x_i \in \{0, 1\}$ 的限制, 对偶后得到:

$$\begin{aligned}
& \max \sum_{i=1}^n y_i + B_1 \times Black - B_2 \times Black \\
& \sum_{i=1}^n A_{i,j} y_i + B_1 - B_2 \leq red_j \quad \forall j \in [1, n]
\end{aligned}$$

然后就可以用单纯形算法解决了

最大流问题

最大流问题的对偶问题是最小割问题

因此可以证明最大流最小割定理

二分图最大权匹配问题

$$\begin{aligned}
& \max \sum_{(u,v) \in E} c_{u,v} d_{u,v} \\
& \sum_{v \in Y} d_{u,v} \leq 1 \quad u \in X \\
& \sum_{u \in X} d_{u,v} \leq 1 \quad v \in Y \\
& d_{u,v} \in \{0, 1\}
\end{aligned}$$

其中 $d_{u,v}$ 表示 (u, v) 是否匹配

由于必定存在一个取值为整数的解使得目标函数值等于最优解, 故去掉取值为整数的限制并不影响答案

对偶后 (将 $\sum_{v \in Y} d_{u,v}, \sum_{u \in X} d_{u,v}$ 看做变量 x_u, x_v) 可得:

$$\begin{aligned}
& \min \sum_{u \in X} p_u + \sum_{v \in Y} p_v \\
& p_u + p_v \geq c_{u,v} \quad u \in X, v \in Y \\
& p_u, p_v \geq 0
\end{aligned}$$

从而得到以下定理: 在一张带权二分图中, **最大权匹配等于最小顶标和**

推论：二分图最小点覆盖等于最大匹配

P4412 [SHOI2004] 最小生成树

题意：给定一个无向图 G 以及图上的一棵生成树 T ，要求修改边权，使得修改后 T 为 G 的最小生成树

修改一条边的代价为边权改变量的绝对值。 $n \leq 50, m \leq 800$

显然树边必定减小边权，非树边必定增加边权

考虑非树边 e 连接的 (u, v) 路径上的所有边 t ，必有

$$w(e) + \Delta e \geq w(t) - \Delta t \Rightarrow \Delta t + \Delta e \geq w(t) - w(e)$$

按照边是否在 T 中出现过分类，把 Δ 看做顶标，则这就是一个二分图最小顶标和问题

化线性规划为半平面交

Equations

题意：给定三个长度为 n 的数组 a_i, b_i, c_i ，每次求下面线性规划的最优目标值：

$$\begin{aligned} & \max \sum_{i=1}^n c_i x_i \\ & \sum_{i=1}^n a_i x_i \leq s \\ & \sum_{i=1}^n b_i x_i \leq t \\ & x_i \geq 0 \end{aligned}$$

$$n \leq 10^5, m \leq 10^4, 1 \leq a_i, b_i, c_i, s, t \leq 10^4$$

思路：只有两个约束，对偶得到：

$$\begin{aligned} & \min sx + ty \\ & a_i x + b_i y \geq c_i \\ & x, y \in R \end{aligned}$$

这样，每个限制 $a_i x + b_i y \geq c_i$ 就对应了一个半平面，它们与询问无关

可以预处理半平面交，然后二分斜率求解询问即可，时间复杂度 $O((n+m)\log n)$

化线性规划为网络流问题

若在线性规划问题中，系数矩阵每一行有且仅有两个非零位置，且它们的值分别为 ± 1

则可以把每个约束看做一个节点，每个变量看做一条边，变量的值看做边的流量

则一个线性等式约束的限制就相当于一个节点的流量守恒限制，这显然是一个网络流模型

对于最大化目标函数的要求，可以用费用流解决

Chefbook

题意：给定一个无向图，每条边 (u, v) 有边权 $L_{u,v}$ ，可以执行以下两种操作：

- 将一个点 u 所有入边的权值减去 Q_u ，其中 Q_u 为任意正数
- 将一个点 u 所有出边的权值加上 P_u ，其中 P_u 为任意正数

要求最终每条边的边权 $L'_{u,v}$ 满足 $S_{u,v} \leq L'_{u,v} \leq T_{u,v}$ ，最大化 $\sum L'_{u,v}$

输出方案, $n \leq 100, m \leq n^2$

思路：先表示成线性规划问题：

$$\begin{aligned} \max & \sum_{u,v} L_{u,v} + P_u - Q_v && (u,v) \in E \\ S_{u,v} & \leq L_{u,v} + P_u - Q_v \leq T_{u,v} && (u,v) \in E \\ P_u, Q_u & \geq 0 && u \in V \end{aligned}$$

目标函数的常数项先忽略不计，令 u 点的入度、出度分别为 $in(u), out(u)$ ，则有：

$$\begin{aligned} \max & \sum_{u \in V} P_u out(u) - Q_u in(u) \\ P_u - Q_v & \leq T_{u,v} - L_{u,v} && (u,v) \in E \\ -P_u + Q_v & \leq L_{u,v} - S_{u,v} && (u,v) \in E \\ P_u, Q_u & \geq 0 && u \in V \end{aligned}$$

这个线性规划的矩阵满足每一行有且仅有两个系数非零，且分别为 ± 1 ，故考虑将其对偶

用 $x_{u,v}$ 表示前一种限制对偶后对应的变量， $y_{u,v}$ 表示后一种变量对偶后所对应的变量，则：

$$\begin{aligned} \min & \sum (T_{u,v} - L_{u,v})x_{u,v} + (L_{u,v} - S_{u,v})y_{u,v} \\ \sum_v (x_{u,v} - y_{u,v}) & \geq out(u) && u \in V \\ \sum_u (-x_{u,v} + y_{u,v}) & \geq -in(v) && v \in V \\ x_{u,v}, y_{u,v} & \geq 0 && (u,v) \in E \end{aligned}$$

考虑添加辅助变量，化不等式为等式：

$$\begin{aligned} \min & \sum (T_{u,v} - L_{u,v})x_{u,v} + (L_{u,v} - S_{u,v})y_{u,v} \\ \sum_v (x_{u,v} - y_{u,v}) - out(u) - a_u & = 0 && u \in V \\ \sum_u (-x_{u,v} + y_{u,v}) + in(v) - b_v & = 0 && v \in V \\ x_{u,v}, y_{u,v}, a_u, b_v & \geq 0 && (u,v) \in E, u, v \in V \end{aligned}$$

现在考虑如何构造费用流图

即约束 1 对应的点为 u ，约束 2 对应的点为 $v+n$

记 (u, v, w, c) 表示一条从 u 到 v ，费用为 w ，流量为 c 的边

对变量 x, y , $\forall (u, v) \in E$, 连 $(v+n, u, T_{u,v} - L_{u,v}, \infty), (u, v+n, L_{u,v} - S_{u,v}, \infty)$

对变量 a_u , 连 $(u, t, 0, \infty)$, 对变量 b_v , 连 $(v+n, t, 0, \infty)$

对常数项 $-out(u)$, 连 $u \rightarrow t$, 费用为 0, 但强制流量为 $out(u)$ 的边

对常数项 $in(v)$, 连 $s \rightarrow v+n$, 费用为 0, 但强制流量为 $in(v)$ 的边

如何只需要从 s 到 t 的最小费用可行流即可, s, t 不要求流量守恒

这是一个比较一般的建模方法：首先将所有约束变成等式，如何分别处理变量、辅助变量和常数项，将模型转化成费用流问题

不过由本题的特殊性还有一种巧妙的方法

将对偶线性规划的约束全部相加，得到：

$$\sum_u \sum_v (x_{u,v} - y_{u,v}) + \sum_v \sum_u (-x_{u,v} + y_{u,v}) \geq \sum_u \text{out}(u) - \sum_v \text{in}(v)$$

显然左右都为 0，故要使这个 $[0 = 0]$ 成立，约束中的所有等号都必须成立

然后按照上面的方法建模，而且由于建出后 s 的出边容量和与 t 的入边容量和相等

故不需要下界的限制，只需求最大流，这样就可以保证等号成立

现在的问题是如何输出方案

考虑利用互松弛定理，原始线性规划的可行解的约数形如若干个二元不等式

由于我们求出的是对偶线性规划的最优解，其也是一个可行解

故我们只需再满足互松弛定理的条件就可以求出原始线性规划的最优解

而我们发现所有的条件都是二元不等式，故可以使用差分约束系统求解

Orz the MST

题意：将 [P4412 \[SHOI2004\] 最小生成树](#) 加强，将边权 $+1$ 的代价为 a_i ， -1 的代价为 b_i 。

$n \leq 300, m \leq 1000$

思路：

$$\begin{aligned} \min & \sum_{i \in T} b_i x_i + \sum_{j \in E-T} a_j x_j \\ x_i + y_j & \geq w_i - w_j \quad i \in T, j \in E-T, \text{cover}(i,j) = 1 \\ x_i, y_j & \geq 0 \end{aligned}$$

考虑将其对偶，用 $s_{i,j}$ 表示对偶后的变量，有：

$$\begin{aligned} \max & \sum_{\substack{i \in T \\ j \in E-T \\ \text{cover}(i,j)=1}} (w_i - w_j) s_{i,j} \\ \sum_{\substack{j \in E-T \\ \text{cover}(i,j)=1}} s_{i,j} & \leq b_i \quad i \in T \\ \sum_{\substack{i \in T \\ \text{cover}(i,j)=1}} s_{i,j} & \leq a_j \quad j \in E-T \\ s_{i,j} & \geq 0 \end{aligned}$$

由于 $T \cap (E \setminus T) = \emptyset$ ，故可以把 $T, E \setminus T$ 分成左右两个点集

把 \leq 看做度数的限制，这样就转化成了最大费用可行流问题

[P3980 \[NOI2008\] 志愿者招募](#)/[P3337 \[ZJOI2013\] 防守战线](#)

利用差分即可

[CF1307G Cow and Exercise](#)

题意：给定一个有向图，每条边有边权 w_i ，每次询问给定一个数 x

可以将一条边修改成 $w_i + a_i$ ，但要保证 $a_i \geq 0, \sum a_i \leq x$

求修改后，从 1 到 n 的最短路径的最长长度。 $n \leq 50, m \leq n(n-1), q \leq 10^5$

思路：

$$\begin{aligned}
 & \max d_n - d_1 \\
 d_v - d_u & \leq w_{u,v} + a_{u,v} \quad (u, v) \in E \\
 \sum_{(u,v) \in E} a_{u,v} & \leq x \\
 d_i, a_{u,v} & \geq 0 \quad i \in V, (u, v) \in E
 \end{aligned}$$

对偶得到 (α, β 对应约束) :

$$\begin{aligned}
 & \min \beta x + \sum_{(u,v) \in E} \alpha_{u,v} w_{u,v} \\
 \sum_{u \in V} \alpha_{u,i} - \sum_{v \in V} \alpha_{i,v} & \geq f(i) \quad i \in V \\
 \beta - \alpha_{u,v} & \geq 0 \quad (u, v) \in E
 \end{aligned}$$

其中：

$$f(i) = \begin{cases} -1 & (i = 1) \\ 1 & (i = n) \\ 0 & (i \neq 1 \wedge i \neq n) \end{cases}$$

这里我们已经看出费用流的影子了，类似上面的题目，将所有约束相加，可以得到：

$$\begin{aligned}
 & \min \beta x + \sum_{(u,v) \in E} \alpha_{u,v} w_{u,v} \\
 \sum_{u \in V} \alpha_{u,i} - \sum_{v \in V} \alpha_{i,v} & = f(i) \quad i \in V \\
 \beta & \geq \alpha_{u,v} \quad (u, v) \in E
 \end{aligned}$$

发现最后那个约束不允许我们用上面的模型，考虑将所有数除 β ，令 $flow = 1/\beta$ ，得到：

$$\begin{aligned}
 & \min \frac{x + \sum_{(u,v) \in E} \alpha_{u,v} w_{u,v}}{flow} \\
 \sum_{u \in V} \alpha_{u,i} - \sum_{v \in V} \alpha_{i,v} & = 0 \quad i \in V - \{1, n\} \\
 \sum_{1 \in V} \alpha_{1,i} - \sum_{v \in V} \alpha_{1,v} & = -flow \\
 \sum_{n \in V} \alpha_{n,i} - \sum_{v \in V} \alpha_{n,v} & = flow \\
 0 \leq \alpha_{u,v} & \leq 1
 \end{aligned}$$

把 1 看做汇点， n 看做源点，这显然是一个费用流的模型，其中 $flow$ 为流量， $cost$ 为最小费用

要求最小化 $(x + cost)/flow$ ，但是我们不可能遍历所有的 $flow$ ，但是我们知道 $flow < n$

又由于在费用流算法中， $flow$ 是递增的，根据它的贪心性， $cost/flow$ 是不减的，故每次增广后的 $(flow, cost)$ 组构成了凸壳的所有位置

从而我们只需建图后跑费用流，记录凸壳上的点，查询时枚举所有位置取最小值即可

时间复杂度 $O(n^2m + qn)$

不过，当 x 为定值时， $(x + cost)/flow$ 是关于 $flow$ 的凸函数，可以三分（或者二分斜率）做到 $O(n^2m + q \log n)$

