不等式问题

great_influence

引言

引言

- 在OI中，存在一类问题。这类问题往往是由各种不等号连成的限制条件构成。

引言

- 在OI中，存在一类问题。这类问题往往是由各种不等号连成的限制条件构成。
- 为了解决这些不等式问题，存在大量算法来处理。

引言

- 在OI中，存在一类问题。这类问题往往是由各种不等号连成的限制条件构成。
- 为了解决这些不等式问题，存在大量算法来处理。
- 这里仅介绍其中的一部分。

导入:[SDOI2017]新生舞会

学校组织了一次新生舞会，Cathy作为经验丰富的老学姐，负责为同学们安排舞伴。

有 n 个男生和 n 个女生参加舞会买一个男生和一个女生一起跳舞，互为舞伴。

Cathy收集了这些同学之间的关系，比如两个人之前认识没计算得出 $a_{i,j}$

Cathy还需要考虑两个人一起跳舞是否方便，比如身高体重差别会不会太大，计算得出 $b_{i,j}$ ，表示第i个男生和第j个女生一起跳舞时的不协调程度。

当然，还需要考虑很多其他问题。

Cathy想先用一个程序通过 $a_{i,j}$ 和 $b_{i,j}$ 求出一种方案，再手动对方案进行微调。

Cathy找到你，希望你帮她写那个程序。

一个方案中有n对舞伴，假设没对舞伴的喜悦程度分别是 $a'_1, a'_2, \ldots, a'_n$ ，假设每对舞伴的不协调程度分别是 $b'_1, b'_2, \ldots, b'_n$ 。令

$$C = \frac{a'_1+a'_2+\ldots+a'_n}{b'_1+b'_2+\ldots+b'_n}$$

Cathy希望C值最大。

导入:[SDOI2017]新生舞会

- 先不考虑下面b的限制。

导入:[SDOI2017]新生舞会

- 先不考虑下面b的限制。
- 可以发现就是一个二分图最大权匹配问题。

导入:[SDOI2017]新生舞会

- 先不考虑下面b的限制。
- 可以发现就是一个二分图最大权匹配问题。
- 然而因为有b的限制，答案不一定最优。

导入:[SDOI2017]新生舞会

- 先不考虑下面b的限制。
- 可以发现就是一个二分图最大权匹配问题。
- 然而因为有b的限制，答案不一定最优。
- 这时需要考虑一种特殊的算法来解决。

分数规划

- 可以很简单地发现，答案具有单调性。

分数规划

- 可以很简单地发现，答案具有单调性。
- 我们尝试使用二分法来解决这个问题。

分数规划

- 可以很简单地发现，答案具有单调性。
- 我们尝试使用二分法来解决这个问题。
- 二分一个答案f，如果f比答案要小，则

分数规划

- 可以很简单地发现，答案具有单调性。
- 我们尝试使用二分法来解决这个问题。
- 二分一个答案f，如果f比答案要小，则

$$f \leq \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}$$

$$f \sum_{i=1}^{n} b_i \leq \sum_{i=1}^{n} a_i$$

$$\sum_{i=1}^{n} a_i - f b_i \geq 0$$

分数规划

- 这样二分后利用费用流直接计算答案就结束了。

分数规划

- 这样二分后利用费用流直接计算答案就结束了。
- 时间复杂度O[flow(n,m)logans]。

分数规划

- 这样二分后利用费用流直接计算答案就结束了。
- 时间复杂度O[flow(n,m)logans]。
- 这种利用二分法来解决带除法型最优化问题的方法就是01分数规划。

分数规划

- 具体而言，分数规划针对问题的一般形式是:

$$\text{定义函数} f = \sum_{i=1}^{n} \frac{a_i x_i}{b_i x_i}\text{，最大化(最小化)f。}$$

分数规划

- 一般而言，解决分数规划的方法是二分答案，再化式子得到一种非除法规划来解决。

分数规划

- 一般而言，解决分数规划的方法是二分答案，再化式子得到一种非除法规划来解决。
- 时间复杂度为O(非除法规划*log ans)。

分数规划:[POJ2976]Dropping tests

有n个点，每个点都有两个权值a_i, b_i。你可以丢弃k个点，最大化

$$100\frac{\sum a_i}{\sum b_i}$$

分数规划:[POJ2976]Dropping tests

- 最基础的01分数规划。直接二分答案贪心求解即可。

分数规划:[POJ2976]Dropping tests

- 最基础的01分数规划。直接二分答案贪心求解即可。
- 时间复杂度O(n log n log ans)。

分数规划:[JSOI2016]最佳团体

JSOI 信息学代表队一共有 N 名候选人，这些候选人从 1 到 N 编号。方便起见，JYY 的编号是 0 号。每个候选人都由一位编号比他小的候选人 R_i 推荐。如果 $R_i = 0$，则说明这个候选人是 JYY 自己看上的。

为了保证团队的和谐，JYY 需要保证，如果招募了候选人 i，那么候选人 R_i 也一定需要在团队中。当然了，JYY 自己总是在团队里的。每一个候选人都有一个战斗值 P_i，也有一个招募费用 S_i。JYY 希望招募 K 个候选人（JYY 自己不算），组成一个性价比最高的团队。也就是，这 K 个被 JYY 选择的候选人的总战斗值与总招募费用的比值最大。

分数规划:[JSOI2016]最佳团体

- 明显的分数规划。

分数规划:[JSOI2016]最佳团体

- 明显的分数规划。
- 二分后跑树形依赖背包即可。

分数规划:[JSOI2016]最佳团体

- 明显的分数规划。
- 二分后跑树形依赖背包即可。
- 时间复杂度O(n^2log ans)。

分数规划

- 分数规划还有许多其他例题。

分数规划

- 分数规划还有许多其他例题。
- 基本都是二分后直接在上面套用其他的算法来求解。

分数规划

- 分数规划还有许多其他例题。
- 基本都是二分后直接在上面套用其他的算法来求解。
- 篇幅有限就不多举例了。

导入:[NOI2008]志愿者招募

申奥成功后，布布经过不懈努力，终于成为奥组委下属公司人力资源部门的主管。布布刚上任就遇到了一个难题：为即将启动的奥运新项目招募一批短期志愿者。经过估算，这个项目需要N 天才能完成，其中第i 天至少需要Ai 个人。 布布通过了解得知，一共有M 类志愿者可以招募。其中第i 类可以从第Si 天工作到第Ti 天，招募费用是每人Ci 元。新官上任三把火，为了出色地完成自己的工作，布布希望用尽量少的费用招募足够的志愿者，但这并不是他的特长！于是布布找到了你，希望你帮他设计一种最优的招募方案。

导入:[NOI2008]志愿者招募

- 考虑使用网络流。

导入:[NOI2008]志愿者招募

- 考虑使用网络流。

连边方法(以下描述中，用二元组 (f,w) 表示容量为 f 费用为 w 的边):

对于每一天向后一天连边 $(inf-a_i,0)$

对于每一种志愿者选择， s_i 向t_i+1 连边 (inf,c_i)

从超级源向第一天连边 $(inf,0)$

从最后一天+1向超级汇连边 $(inf,0)$

然后从超级源向超级汇跑费用流。

导入:[NOI2008]志愿者招募

- 是不是非常麻烦，根本想不到?

导入:[NOI2008]志愿者招募

- 是不是非常麻烦，根本想不到?
- 其实可以将这个问题想简单一点。

导入:[NOI2008]志愿者招募

- 这是一个明显的最优化问题。
- 题目可以写作:

导入:[NOI2008]志愿者招募

- 这是一个明显的最优化问题。
- 题目可以写作:

$$\forall p, \sum_{p\in(s_i,t_i)} k_i \geq a_p$$

$$\min\ f = \sum_{i=1}^{m} c_i k_i$$

导入:[NOI2008]志愿者招募

- 那么，有没有一种神奇的工具，可以解决这种多元一次不等式组的最优化问题呢?

导入:[NOI2008]志愿者招募

- 那么，有没有一种神奇的工具，可以解决这种多元一次不等式组的最优化问题呢?

单纯形法

单纯形法

- 单纯形法是用来解决多元一次不等式组的最优化问题的一种常用方法。

单纯形法

- 单纯形法是用来解决多元一次不等式组的最优化问题的一种常用方法。
- 在学习之前，先了解几个定义。

单纯形法

- 1.在单纯形法中，所有变量都非负。

单纯形法

- 1.在单纯形法中，所有变量都非负。
- 如果存在某个变量没有非负限制，那么将它拆成两个非负变量相减即可。

单纯形法

- 2.称最优化函数中系数不为0的变量为基变量，其余为非基变量。

单纯形法

- 3.称将一个基变量和一个非基变量相互替换的操作为转轴(pivot)。

单纯形法

- 4.最优化函数为max。可以知道，如果是求min的话，将所有的变量取反就可以转成max了。

单纯形法

- 有了前两条定义，可以将所有的不等式限制条件利用非基变量转为等式。

单纯形法

- 有了前两条定义，可以将所有的不等式限制条件利用非基变量转为等式。
- 转换方式如下:

$$x_1 + x_2 + x_3 \leq b \rightarrow x_1 + x_2 + x_3 + x_4 = b$$
$$x_1 + x_2 + x_3 \geq b \rightarrow x_1 + x_2 + x_3 - x_4 = b$$
$$x_1 + x_2 + x_3 = b \rightarrow$$
$$x_1 + x_2 + x_3 + x_4 = b, x_1 + x_2 + x_3 - x_5 = b$$

单纯形法

- 有了这些等式以后，我们可以发现，原来的不等式组变成了等式组。并且每一个限制条件可以用1个或者2个非基变量来代替。

单纯形法

- 有了这些等式以后，我们可以发现，原来的不等式组变成了等式组。并且每一个限制条件可以用1个或者2个非基变量来代替。
- 代替的方法就是转换一下式子:

$$x_1 + x_2 + x_3 + x_4 = b \rightarrow x_4 = b - x_1 - x_2 - x_3$$

单纯形法

- 我们先假设对于任意非基变量，其对应的b都为正数。非正数可以利用初始化来转成正数。

单纯形法

- 我们先假设对于任意非基变量，其对应的b都为正数。非正数可以利用初始化来转成正数。
- 那么可以发现，每个非基变量的取值范围下界都是0。

单纯形法

- 我们先假设对于任意非基变量，其对应的b都为正数。非正数可以利用初始化来转成正数。
- 那么可以发现，每个非基变量的取值范围上界都是对应的b。
- 那么，我们可以发现，函数f也是如此。

单纯形法

- 我们先假设对于任意非基变量，其对应的b都为正数。非正数可以利用初始化来转成正数。
- 那么可以发现，每个非基变量的取值范围上界都是对应的b。
- 那么，我们可以发现，函数f也是如此。
- 如果每个基变量的系数都为正数，那么此时所有基变量都取0时，答案自然最大。

单纯形法

- 我们先假设对于任意非基变量，其对应的b都为正数。非正数可以利用初始化来转成正数。
- 那么可以发现，每个非基变量的取值范围上界都是对应的b。
- 那么，我们可以发现，函数f也是如此。
- 如果每个基变量的系数都为正数，那么此时所有基变量都取0时，答案自然最大。
- 我们可以想办法构造一种算法，将所有基变量的系数变成正数。

单纯形法

- 考虑定义3中提到的pivot。

单纯形法

- 考虑定义3中提到的pivot。
- 找到一个算法，将某个基变量和非基变量互换。

单纯形法

- 考虑定义3中提到的pivot。
- 找到一个算法，将某个基变量和非基变量互换。
- 这个比较容易推出。

单纯形法

- 设

$$x_i = b_i + \sum_{j!=i} a_{i,j} x_j$$

单纯形法

- 设

$$x_i = b_i + \sum_{j!=i} a_{i,j} x_j$$

- 如果aik为负数，则可以将非基变量xi和基变量xk互转(称为pivot(i,k))，即

$$x_k = -\frac{b_i}{a_{i,k}} - \sum_{j!=i,j!=k} \frac{a_{i,j}}{a_{i,k}} + \frac{1}{a_{i,k}} x_i$$

单纯形法

- 可以发现，这样转后，常数项b的系数仍然是正数，且xk换出来的xi系数为负数，加到最优化函数中就变成了正数。

单纯形法

- 可以发现，这样转后，常数项b的系数仍然是正数，且xk换出来的xi系数为负数，加到最优化函数中就变成了正数。
- 这样每转一次，一定能将一个负系数基变量转为正系数基变量。

单纯形法

- 可以发现，这样转后，常数项b的系数仍然是正数，且xk换出来的xi系数为负数，加到最优化函数中就变成了正数。
- 这样每转一次，一定能将一个负系数基变量转为正系数基变量。
- 无限地转移后，将系数全部转成正数，就可以直接出解了。

单纯形法

- 但是，pivot可能带来新的负系数基变量。

单纯形法

- 但是，pivot可能带来新的负系数基变量。
- 此时引入一个贪心的思想。

单纯形法

- 但是，pivot可能带来新的负系数基变量。
- 此时引入一个贪心的思想。
- 每次pivot的时候，找到限制这个基变量最紧的非基变量来转轴。

单纯形法

- 但是，pivot可能带来新的负系数基变量。
- 此时引入一个贪心的思想。
- 每次pivot的时候，找到限制这个基变量最紧的非基变量来转轴。
- 也就是在所有的aik<0的非基变量ai中，找到-bk/aik最小的一个变量，用它来替换ak。

单纯形法

- 但是，pivot可能带来新的负系数基变量。
- 此时引入一个贪心的思想。
- 每次pivot的时候，找到限制这个基变量最紧的非基变量来转轴。
- 也就是在所有的aik<0的非基变量ai中，找到-bk/aik最小的一个变量，用它来替换ak。
- 这样替换的话，对其他系数影响相对较小。可以优化一下复杂度，避免出现死循环。

单纯形法

- 下面给出pivot的具体实现。

单纯形法

- 下面给出pivot的过程。
- 1.利用公式 $x_k = -\frac{b_i}{a_{i,k}} - \sum_{j!=i,j!=k} \frac{a_{i,j}}{a_{i,k}} + \frac{1}{a_{i,k}} x_i$ ， 先计算出新的系数。

单纯形法

- 下面给出pivot的过程。
- 1.利用公式 $x_k = -\frac{b_i}{a_{i,k}} - \sum_{j!=i,j!=k} \frac{a_{i,j}}{a_{i,k}} + \frac{1}{a_{i,k}} x_i$ ，先计算出新的系数。
- 2.扫一遍所有的等式和最优化函数，将里面出现的所有的xk全部换成上方等式右边。

单纯形法

- 下面给出pivot的过程。
- 1.利用公式 $x_k = -\frac{b_i}{a_{i,k}} - \sum_{j!=i,j!=k} \frac{a_{i,j}}{a_{i,k}} + \frac{1}{a_{i,k}} x_i$ ，先计算出新的系数。
- 2.扫一遍所有的等式和最优化函数，将里面出现的所有的xk全部换成上方等式右边。
- 3.没了。

单纯形法

- 下面给出具体代码。

单纯形法

- 下面给出具体代码。

```
inline void pivot(int x,int y)
{
    b[y]=-b[x]/a[x][y];
    Rep(i,1,n+m)if(i^y)a[y][i]=-a[x][i]/a[x][y];
    a[y][x]=1.0/a[x][y];
    Rep(i,1,n+m)a[x][i]=0.0;b[x]=0.0;
    Rep(i,0,n+m)if(a[i][y])plu(i,y,a[i][y]),a[i][y]=0.0;
}
```

单纯形法

- 有了基本思想后，给出单纯形法的基本思路。

单纯形法

- 有了基本思想后，给出单纯形法的基本思路。
- 1.在基变量中寻找一个负系数基变量，找不到即结束。

单纯形法

- 有了基本思想后，给出单纯形法的基本思路。
- 1.在基变量中寻找一个负系数基变量，找不到即结束。
- 2.找到对该变量约束最紧的负系数非基变量。找不到则答案无穷大。

单纯形法

- 有了基本思想后，给出单纯形法的基本思路。
- 1.在基变量中寻找一个负系数基变量，找不到即结束。
- 2.找到对该变量约束最紧的负系数非基变量。找不到则答案无穷大。
- 3.利用pivot更替变量，返回1。

单纯形法

- 下面放出代码。

单纯形法

- 下面放出代码。

```
static int now,ps;
static double mn;
while(true)
{
    now=0;
    Rep(i,1,n+m)if(a[0][i]>eps){now=i;break;}
    if(!now)break;
    mn=1e20;ps=0;
    Rep(i,1,n+m)if(a[i][now]<-eps&&-b[i]/a[i][now]<mn)
    {mn=-b[i]/a[i][now];ps=i;}
    if(!ps)return (void)puts("Unbounded");
    pivot(ps,now);
}
```

单纯形法

- 开始考虑初始化问题。

单纯形法

- 开始考虑初始化问题。
- 可以发现，初始化的复杂度难以保证，没有好的初始化方法。

单纯形法

- 开始考虑初始化问题。
- 可以发现，初始化的复杂度难以保证，没有好的初始化方法。
- 考虑随机初始化。

单纯形法

- 开始考虑初始化问题。
- 可以发现，初始化的复杂度难以保证，没有好的初始化方法。
- 考虑随机初始化。
- 每次找到一个b为负数的非基变量(随机选择)，在找到一个负系数非基变量(随机选择)，将他们pivot一下。

单纯形法

- 开始考虑初始化问题。
- 可以发现，初始化的复杂度难以保证，没有好的初始化方法。
- 考虑随机初始化。
- 每次找到一个b为负数的非基变量(随机选择)，在找到一个负系数基变量(随机选择)，将他们pivot一下。
- 这样不停换，知道所有b全部为正为止。

单纯形法

- 下面放出代码。

单纯形法

- 下面放出代码。

```
inline bool init()
{
    static int now,ps;
    while(true)
    {
        now=ps=0;
        Rep(i,1,n+m)if(b[i]<-eps&&(!now||rand()&1))now=i;
        if(!now)return false;
        Rep(i,1,n+m)if(-a[now][i]<-eps&&(!ps||rand()&1))ps=i;
        if(!ps)return true;
        pivot(now,ps);
    }
}
```

单纯形法

- 有了单纯形法，很多的多元一次方程组最优化问题似乎都可以迎刃而解。

单纯形法

- 有了单纯形法，很多的多元一次方程组最优化问题似乎都可以迎刃而解。
- 比如引入([NOI2008]志愿者招募)，发现就是一个模板式的线性规划。直接单纯性求解即可。

单纯形法

- 有了单纯形法，很多的多元一次方程组最优化问题似乎都可以迎刃而解。
- 比如引入([NOI2008]志愿者招募)，发现就是一个模板式的线性规划。直接单纯性求解即可。
- 其它的题目，例如最大流、最小割、货物分配、差分约束等一大波算法都可以用线性规划替代。

单纯形法

- 有了单纯形法，很多的多元一次方程组最优化问题似乎都可以迎刃而解。
- 比如引入([NOI2008]志愿者招募)，发现就是一个模板式的线性规划。直接单纯性求解即可。
- 其它的题目，例如最大流、最小割、货物分配、差分约束等一大波算法都可以用线性规划替代。
- 然而单纯形法为非多项式算法，尽管操作次数一般不多，单次操作也高达O(nm)。使用的时候需要慎重。

单纯形法:例题

- 先做这道题吧。http://uoj.ac/problem/179(UOJ神题，至今无人AK)

单纯形法:例题

- 因为单纯形法的编程复杂度较低重复性高，所以题目就不放代码了。只放了一个模板供参考。

[ZJOI2013]防守战线

- 战线可以看作一个长度为n 的序列，现在需要在这个序列上建塔来防守敌兵，在序列第i 号位置上建一座塔有Ci 的花费，且一个位置可以建任意多的塔，费用累加计算。有m 个区间[L1, R1], [L2, R2], …, [Lm, Rm]，在第i 个区间的范围内要建至少Di 座塔。求最少花费。

[ZJOI2013]防守战线

- 战线可以看作一个长度为n 的序列，现在需要在这个序列上建塔来防守敌兵，在序列第i 号位置上建一座塔有Ci 的花费，且一个位置可以建任意多的塔，费用累加计算。有m 个区间[L1, R1], [L2, R2], …, [Lm, Rm]，在第i 个区间的范围内要建至少Di 座塔。求最少花费。

$$\sum_{j\in(l_i,r_i)} k_j \geq d_i$$

$$\min f = \sum_{i=1}^{n} c_i k_i$$

[ONTAK2010]Vacation

- 有3N个数，你需要选出一些数，首先保证任意长度为N的区间中选出的数的个数<=K个，其次要保证选出的数的个数最大。

[ONTAK2010]Vacation

- 有3N个数，你需要选出一些数，首先保证任意长度为N的区间中选出的数的个数<=K个，其次要保证选出的数的个数最大。

$$\sum_{j\in(i,i+n-1)} k_j \le K$$

$$\forall i, 0 \le k_i \le 1$$

$$\max f = \sum_{i=1}^{n} k_i$$

[网络流24题]最长k可重区间集问题

- 对于给定的开区间集合 I 和正整数 k，计算开区间集合 I 的最长 k可重区间集的长度。

[网络流24题]最长k可重区间集问题

- 对于给定的开区间集合 I 和正整数 k，计算开区间集合 I 的最长 k可重区间集的长度。

$$\sum_{l_z<i<r_z} z_i p_i \le k$$

$$\max f = \sum_{z \in S} z_i p_i$$

线性规划:对偶

给定一个原始线性规划:

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

$$\sum_{j=1}^{n} a_{i,j} x_j \geq b_i$$

$$x_j \geq 0$$

线性规划:对偶

定义它的对偶线性规划为:

$$\max \sum_{i=1}^{m} b_j y_j$$

$$\sum_{i=1}^{m} a_{i,j} y_i \geq c_j$$

$$y_i \geq 0$$

线性规划:对偶

- 可以证明，这2个问题的最优解是相等的。详见2016国家队论文《浅谈线性规划与对偶问题》。

线性规划:对偶

- 可以证明，这2个问题的最优解是相等的。详见2016国家队论文《浅谈线性规划与对偶问题》。
- 因此，可以通过对偶将线性规划转成更方便求解的形式。

线性规划:对偶

- 可以证明，这2个问题的最优解是相等的。详见2016国家队论文《浅谈线性规划与对偶问题》。
- 因此，可以通过对偶将线性规划转成更方便求解的形式。
- 例如，如果一个线性规划只有2个变量，显然可以用半平面交做到优于单纯形的复杂度。或者它只有2个限制，则可以利用对偶转成2个变量问题，用半平面交求解。

线性规划:对偶

- 可以证明，这2个问题的最优解是相等的。详见2016国家队论文《浅谈线性规划与对偶问题》。
- 因此，可以通过对偶将线性规划转成更方便求解的形式。
- 例如，如果一个线性规划只有2个变量，显然可以用半平面交做到优于单纯形的复杂度。或者它只有2个限制，则可以利用对偶转成2个变量问题，用半平面交求解。
- 还有更多应用，比如化成网络流、费用流等。再次就不再多展开了。

线性规划:对偶

- 可以证明，这2个问题的最优解是相等的。详见2016国家队论文《浅谈线性规划与对偶问题》。
- 因此，可以通过对偶将线性规划转成更方便求解的形式。
- 例如，如果一个线性规划只有2个变量，显然可以用半平面交做到优于单纯形的复杂度。或者它只有2个限制，则可以利用对偶转成2个变量问题，用半平面交求解。
- 还有更多应用，比如化成网络流、费用流等。再次就不再多展开了。
- 因此，可以利用线性规划的对偶来扩展网络流建模的思路，也可以算是一种作用吧。

线性规划:对偶

- 如果有需要的话，在这里将证明放出。

线性规划:对偶

- 定理1:如果向量X是原问题的可行解，Y是对偶问题的可行解，则

$$\sum_{j=1}^{n} c_j x_j \geq \sum_{i=1}^{m} b_i y_i$$

线性规划:对偶

- 证明:

线性规划:对偶

- 证明:
- 因为Y是对偶问题一个可行解，所以 $c_j \geq \sum_{i=1}^{m} a_{i,j} y_i$

线性规划:对偶

- 证明:
- 因为Y是对偶问题一个可行解，所以 $c_j \geq \sum_{i=1}^{m} a_{i,j} y_i$
- 由于xi>=0,所以 $\sum_{j=1}^{n} c_j x_i \geq \sum_{j=1}^{n} \sum_{i=1}^{m} a_{i,j} y_i x_j$

线性规划:对偶

- 证明:
- 因为Y是对偶问题一个可行解，所以 $c_j \geq \sum_{i=1}^{m} a_{i,j} y_i$
- 由于xi>=0,所以 $\sum_{j=1}^{n} c_j x_i \geq \sum_{j=1}^{n} \sum_{i=1}^{m} a_{i,j} y_i x_j$
- 同理，$\sum_{j=1}^{n} \sum_{i=1}^{m} a_{i,j} y_i x_j \geq \sum_{i=1}^{m} b_i y_i$

线性规划:对偶

- 证明:
- 因为Y是对偶问题一个可行解，所以 $c_j \geq \sum_{i=1}^{m} a_{i,j} y_i$
- 由于xi>=0,所以 $\sum_{j=1}^{n} c_j x_i \geq \sum_{j=1}^{n} \sum_{i=1}^{m} a_{i,j} y_i x_j$
- 同理，$\sum_{j=1}^{n} \sum_{i=1}^{m} a_{i,j} y_i x_j \geq \sum_{i=1}^{m} b_i y_i$
- 因此，$\sum_{j=1}^{n} c_j x_j \geq \sum_{i=1}^{m} b_i y_i$

线性规划:对偶

- 定理2:若X*为原问题最优解，Y*为对偶问题最优解，则

$$\sum_{j=1}^{n} c_j x_j^* = \sum_{i=1}^{m} b_i y_i^*$$

线性规划:对偶

- 证明连论文上都没写:

这个定理的证明较为繁琐，此处略去，参考文献[1]中有详细证明。

线性规划:对偶

- 证明连论文上都没写:

这个定理的证明较为繁琐，此处略去，参考文献[1]中有详细证明。

- 反正形象理解一下，就是定理1求出来的下界一定可达。

01规划

- 基本就不举例了。

01规划

- 基本就不举例了。
- 但是线性规划好像只能解决实数域问题啊？

01规划

- 基本就不举例了。
- 但是线性规划好像只能解决实数域问题啊？
- 有没有一种方法，能够解决这种整数规划问题呢？

01规划

- 基本就不举例了。
- 但是线性规划好像只能解决实数域问题啊？
- 有没有一种方法，能够解决这种整数规划问题呢？

01规划

- 先考虑线性规划的几何意义。

01规划

- 先考虑线性规划的几何意义。
- 线性规划其实就是在一个多维平面上找到一个最优点。

01规划

- 先考虑线性规划的几何意义。
- 线性规划其实就是在一个多维平面上找到一个最优点。
- 可以证明，这个最优点一定是该平面的某个顶点。

01规划

- 先考虑线性规划的几何意义。
- 线性规划其实就是在一个多维平面上找到一个最优点。
- 可以证明，这个最优点一定是该平面的某个顶点。
- 运用反证法证明。如果最优点不在某个顶点上，一定存在一个维度k,该维度可以任意变大变小。那么向着让最优化函数更优的方向移动，一定不会更劣，与假设矛盾。

01规划

- 如果线性规划的最优点坐标正好全是整数，那么跑出来的解一定合法。

01规划

- 如果线性规划的最优点坐标正好全是整数，那么跑出来的解一定合法。
- 如果不合法，就要考虑别的方法了。

01规划

- 如果线性规划的最优点坐标正好全是整数，那么跑出来的解一定合法。
- 如果不合法，就要考虑别的方法了。
- 对于一般的整数规划，可以无视整数限制直接做线性规划(一般不会有问题)，也可以”将中间的变量全部换成整数”，或者暴力枚举。

01规划

- 1.变量较少时(<10)直接枚举所有情况即可。

01规划

- 1.变量较少时(<10)，直接枚举所有情况即可。
- 2.变量较多但不是特别多时，考虑存储最大值，增加一个限制条件f>max，再依次判断。这种枚举方法被称为隐枚举法。

01规划

- 1.变量较少时(<10)，直接枚举所有情况即可。
- 2.变量较多但不是特别多时，考虑存储最大值，增加一个限制条件f>max，再依次判断。这种枚举方法被称为隐枚举法。
- 3.变量非常多，但是能够找到一种网络流或者费用流等整数算法替代，那就直接替代。

01规划

- 1.变量较少时(<10)，直接枚举所有情况即可。
- 2.变量较多但不是特别多时，考虑存储最大值，增加一个限制条件f>max，再依次判断。这种枚举方法被称为隐枚举法。
- 3.变量非常多，但是能够找到一种网络流或者费用流等整数算法替代，那就直接替代。
- 4.否则就暴力枚举吧。

01规划

- 在这些整数规划问题中，存在一类特殊的整数规划。它的所有系数全部都是0、1、-1，且如果将限制条件的系数作为一个矩阵时，它任意一个子方阵的行列式均为0,1,-1。

01规划

- 在这些整数规划问题中，存在一类特殊的整数规划。它的所有系数全部都是0、1、-1，且如果将限制条件的系数作为一个矩阵时，它任意一个子方阵的行列式均为0,1,-1。
- 可以证明，这类线性规划的所有顶点全部都是整数，可以直接套用单纯形法解决。

01规划

- 在这些整数规划问题中，存在一类特殊的整数规划。它的所有系数全部都是0、1、-1，且如果将限制条件的系数作为一个矩阵时，它任意一个子方阵的行列式均为0,1,-1。
- 可以证明，这类线性规划的所有顶点全部都是整数，可以直接套用单纯形法解决。
- (然而大部分时候没这么好，不过不仔细卡的话，单纯形法能够得到相当高的分数)

题目推荐

- 01分数规划:Poj2728Desert King、Poj3621Sightseeing Cows、[Scoi2014]方伯伯运椰子、[HNOI2009]最小圈、POJ3155 Hard Life。
- 线性规划:志愿者招募加强版、POJ1275 Cashier Employment、Uva10498Happiness、BZOJ1937Flight Distance、POJ3689Equations
- 01规划:[ZJOI2015]幻想乡Wifi搭建计划(70pts)、[清华集训2014]文学、[CF275E]Red and Black Tree