浅谈线性规划与对偶问题

福建省福州第一中学 董克凡

摘 要

信息学竞赛中，有许多问题均能用线性规划表示。而解决这一类问题的方法十分灵活多变，如动态规划、最短路、网络流等。本文由线性规划的角度来探讨这一类问题的性质，然后通过对偶原理将多个线性规划问题归类总结。

1 引言

在信息学竞赛中，以网络流为代表的一系列线性规划问题的解法往往灵活多变，甚至有些“出人意料”，本文中通过将这些问题表示为线性规划的方式来更深层地理解这类问题。希望能给所有参加信息学竞赛的同学或多或少的启发。

本文第一部分着重介绍线性规划的定义，以及求解一般线性规划的单纯形方法，第二部分着重介绍对偶原理，并且列举了一些通过对偶原理高效求解的例题。

2 线性规划

2.1 定义

在最优化问题中，有一类问题满足它的限制条件以及目标函数都是自变量的线性函数，我们称这一类问题为线性规划问题。为了叙述方便，在这里首先给出一些定义。

定义2.1. 已知一组实数$a_1, a_2, \ldots, a_n$，以及一组变量$x_1, x_2, \ldots, x_n$，在这些变量上

的一个线性函数定义为

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

等式$f(x_1, x_2, \ldots, x_n) = b$，不等式$f(x_1, x_2, \ldots, x_n) \leq b, f(x_1, x_2, \ldots, x_n) \geq b$统称为线性约束。

线性规划问题要求最大化或最小化一个受限于一组有限的线性约束的线性函数。

称满足所有限制条件的解$x_1, x_2, \ldots, x_n$为可行解，使目标函数达到最优的可行解为最优解，所有可行解构成的区域为解空间。

2.2 线性规划的性质

为了描述线性规划的性质与算法，这里需要使用一种更规范化的形式来描述线性规划问题。

定义2.2. 标准型

标准型线性规划要求满足如下形式

$$\begin{aligned} &\text{最大化} && \sum_{j=1}^{n} c_j x_j \\ &\text{满足约束} && \sum_{j=1}^{n} a_{ij} x_j \leq b_i, && i = 1, 2, \ldots, m \\ & && x_j \geq 0, && j = 1, 2, \ldots, n \end{aligned}$$

容易发现，经过一些简单的变换，所有线性规划问题都可以用标准型描述。

若目标函数要求取最小值，那么可以对其取相反数变成取最大值。对于限制条件$f(x_1, x_2, \ldots, x_n) = b$，可以用两个不等式$f(x_1, x_2, \ldots, x_n) \leq b, -f(x_1, x_2, \ldots, x_n) \leq -b$描述，对于限制条件$f(x_1, x_2, \ldots, x_n) \geq b$，可以用不等式$-f(x_1, x_2, \ldots, x_n) \leq -b$描述。对于无限制的变量$x$，可以将其拆为两个非负变量$x_0, x_1$，使得$x = x_0 - x_1$。

标准型也可以用矩阵表示：

$$\begin{aligned} &\text{最大化} && \mathbf{c}^T\mathbf{x} \\ &\text{满足约束} && A\mathbf{x} \le \mathbf{b} \\ & && \mathbf{x} \ge 0 \end{aligned} \tag{2.2.1}$$

其中向量$\mathbf{x} \le \mathbf{y}$，当且仅当对于这两个向量的每一维$x_i, y_i$，都有$x_i \le y_i$

标准型的优点是简洁明了，而为了求解线性规划，需要所有的约束条件都是等式，这样可以避免在处理负系数时产生的不等号方向改变的问题，所以下面介绍线性规划的另一种表示方法——松弛型。

定义2.3. 松弛型

松弛型线性规划要求满足如下形式

$$\begin{aligned} &\text{最大化} && \sum_{j=1}^{n} c_j x_j \\ &\text{满足约束} && x_{i+n} = b_i - \sum_{j=1}^{n} a_{ij} x_j, && i = 1, 2, \ldots, m \\ & && x_j \ge 0, && j = 1, 2, \ldots, n+m \end{aligned}$$

将标准型转化为松弛型是容易的。对于在标准型中的一个限制$f_i(x_1, x_2, \ldots, x_n) \le b_i$，新增加一个变量$x_{i+n} = b_i - f_i(x_1, x_2, \ldots, x_n)$，那么原来不等式的限制条件就转化为如下的等式限制：$x_{i+n} = b_i - f_i(x_1, x_2, \ldots, x_n) \ge 0$，转化之后仍然满足所有变量的取值仍为非负实数。

考察一个线性规划的解空间，这个解空间是多个线性不等式解空间的交。由于每一个线性不等式的解空间都是一个凸形区域[1]，故可以由归纳证明得，线性规划的解空间是一个凸形区域。这里证明从略。

由于解空间是一个凸形区域，也就是说局部最优解的值有且只有一个[2]，并且等于全局最优解的值。那么在这个区域内进行贪心，使用一个类似于爬山的算法来求解最优值的时候，就不需要担心算法在一个局部最优解中止，由这个性质，便引出了求解线性规划的一个一般性方法——单纯形法。

[1]凸形区域的一个直观定义是，区域内的任意两点连线上的点都属于这个区域

[2]这里暂不考虑线性规划无界（最优解可以取到无穷）的情况；局部最优解可能不止一个，有可能是某个与目标函数平行的边界，但是它们的值一定相同

3 单纯形

3.1 算法描述

单纯形法的基本思想是，通过变量的代换，实现在解空间内沿着边界朝着目标函数增大的方向移动。其核心操作是*转轴*（pivot）操作，即变量的代换。

首先做一些定义：*基变量*：在松弛型等式左侧的所有变量。*非基变量*：在松弛型等式右侧的所有变量。一个线性规划的一组基变量和非基变量就隐含着一个*基本解*。即所有基变量的值为右侧的常数项[3]，所有非基变量的值为0，在单纯形算法中，我们只考虑基本解。

单纯形算法有两个主要的操作：转轴操作以及simplex操作。转轴操作的作用是，选择一个基变量x_B 以及一个非基变量x_N，将其互换（称这个非基变量为*换入变量*，基变量为*换出变量*），即用x_B以及其他非基变量代换x_N。具体来说，一开始有

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

那么

$$x_N = (b_i - \sum_{j\neq N} a_{ij} - x_B)/a_{iN}$$

将目标函数以及其余所有限制中的x_N 用这个等式替换，那么等式右侧就不会出现x_N这个变量，这就做到了互换基变量和非基变量。当然在选择x_N时要保证$a_{iN} \neq 0$

simplex操作即单纯形算法的主过程，从一个基本解出发，经过一系列的转轴操作，达到最优解。通过选择特定的换入变量以及换出变量，可以使得每一次转轴操作都能使目标函数增大，直到达到全局最优解。

下面用一个例子来说明单纯形的具体流程，考虑这一个线性规划：

$$\begin{aligned}
&\text{最大化} && 3x_1 + x_2 + 2x_3 && (3.1)\\
&\text{满足约束} && x_4 = 30 - x_1 - x_2 - 3x_3 &&\\
& && x_5 = 24 - 2x_1 - 2x_2 - 5x_3 &&\\
& && x_6 = 36 - 4x_1 - x_2 - 2x_3 &&\\
& && x_j \geq 0, \qquad j = 1, 2, \ldots, 6 &&
\end{aligned}$$

[3]常数项为负数的情况将在下一节中考虑

这个线性规划隐含的初始基本解为$(x_1, x_2, \ldots, x_6) = (0, 0, 0, 30, 24, 36)$，记目标函数为$z$，那么这时$z = 0$。

第一步选择x_1作为换入变量，因为在目标函数中x_1系数为正，增大x_1必然会增大z。由于所有变量非负的限制，x_1不可能无限制地增大，分别考虑三个限制条件，当x_1增大到30 以上，而x_2, x_3不变时，x_4就会变为负数，所以$x_1 \leq 30$。同样的，由后两个限制可以得到$x_1 \leq 12, x_1 \leq 9$。从中选择一个对x_1限制最紧的约束，即第三个约束，把第三个约束中的基本变量x_6作为换出变量。将限制三改写为$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$，代入目标函数以及其余约束，可以得到如下新形式的线性规划：

$$
\begin{aligned}
&\text{最大化} && 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4} \\
&\text{满足约束} && x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4} \\
& && x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2} \\
& && x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4} \\
& && x_j \geq 0, \qquad\qquad j = 1, 2, \ldots, 6
\end{aligned}
$$

这个新的线性规划仍然满足松弛型的形式，并且在这次迭代之后，目标函数增大至$z = 27$。

考虑转轴操作之后的基本解$(x_1, x_2, \ldots, x_6) = (9, 0, 0, 21, 6, 0)$与操作之前的基本解$(x_1, x_2, \ldots, x_6) = (0, 0, 0, 30, 24, 36)$，$x_4, x_5, x_6$在转轴操作前是基变量，它们的值的改变对目标函数的值没有影响，x_1在操作之后由非基变量变成了基变量，它的值由0增大为9，这就使得目标函数的值增大至27。虽然目标函数的值增大了，但是操作前后的两个线性规划是等价的。这时新的基本解为$(9, 0, 0, 21, 6, 0)$，如果将这个基本解代入(3.1)中，同样可以得到$z = 27$。

接下来，因为目标函数中仍然有变量的系数为正，那么也就是说目标函数可以继续被增大。这一次不妨选择x_3作为换入变量，那么换出变量就是x_5，因为x_5限制了$x_3 \leq \frac{3}{2}$，那么将第二个限制改写为$x_3 = \frac{3}{2} - \frac{3x_2}{8} - \frac{x_5}{4} + \frac{x_6}{8}$，代入目标函

数以及其余的约束，得：

$$\begin{aligned}
&\text{最大化} && \frac{111}{4}+\frac{x_2}{16}-\frac{x_5}{8}-\frac{11x_6}{16}\\
&\text{满足约束} && x_4=\frac{69}{4}+\frac{3x_2}{16}+\frac{5x_5}{8}-\frac{x_6}{16}\\
& && x_3=\frac{3}{2}-\frac{3x_2}{8}-\frac{x_5}{4}+\frac{x_6}{8}\\
& && x_1=\frac{33}{4}-\frac{x_2}{16}+\frac{x_5}{8}-\frac{5x_6}{16}\\
& && x_j\geq 0, \qquad j=1,2,\ldots,6
\end{aligned}$$

接下来，只能选择x_2作为换入变量，这是需要注意，由于在第一个限制条件中，x_2 的系数是正的，所以这一个限制条件并不限制x_2 增加到多少，故三个约束给出的上界分别为$\infty, 4, 132$。那么选择x_3作为换入变量，得：

$$\begin{aligned}
&\text{最大化} && 28-\frac{x_3}{6}-\frac{x_5}{6}-\frac{2x_6}{3}\\
&\text{满足约束} && x_4=18-\frac{x_3}{2}+\frac{x_5}{2}\\
& && x_2=4-\frac{8x_3}{3}-\frac{2x_5}{3}+\frac{x_6}{3}\\
& && x_1=8+\frac{x_3}{6}+\frac{x_5}{6}-\frac{x_6}{3}\\
& && x_j\geq 0, \qquad j=1,2,\ldots,6
\end{aligned}$$

此时对应的基本解为$(8,4,0,18,0,0)$。这时，目标函数中所有变量系数均为负，所以我们找到了一个局部最优点，由线性规划的性质，这就是全局最优解。下面正式地给出单纯形算法的伪代码：其中$(A,\mathbf{b},\mathbf{c})$意义与(2.2.1)同。

Algorithm 1 Simplex(A, b, c)

```
initiation(A, b, c)
while ∃e that c_e > 0 do
    find the index l that A_le > 0 and minimizes b_l/A_le
    if ∀l, A_le ≤ 0 then
        return Unbounded
    else
        pivot(A, b, c, l, e)
    end if
end while
```

其中，第4,5两行的作用是，若某一步选择的换入变量没有约束条件，那么这个线性规划就是无界的，否则，找到最紧的约束，执行转轴操作。initiation操作是初始化,即找到一组基本解，一般来说，若$b_i \geq 0, i = 1, \ldots, m$，那么可以略去这一操作，直接从$(0, 0, \ldots, 0, b_1, b_2, \ldots, b_m)$这一基本解开始。由于每一步转轴操作我们选择的是一个限制最紧的约束，所以在转轴操作之后，仍能保证$b_i \geq 0, i = 1, \ldots, m$。（证明从略）

初始化操作的基本思想是引入一个辅助线性规划:

$$\begin{aligned} &\text{最大化} && -x_0 && \\ &\text{满足约束} && x_{i+n} = b_i - \sum_{j=1}^{n} a_{ij}x_j + x_0, && i = 1, 2, \ldots, m \\ & && x_j \geq 0, && j = 0, 1, \ldots, n+m \end{aligned}$$

如果原线性规划存在一个可行解$(\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_{n+m})$，那么$(0, \overline{x}_1, \overline{x}_2, \ldots, \overline{x}_{n+m})$就是辅助线性规划的一个可行解。又因为$x_0 \geq 0$的限制，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最优解$x_0 = 0$，那么直接将x_0从这个线性规划中删去不会对约束产生影响，然后再将目标函数替换为原目标函数，就得到了一个满足$b_i \geq 0$的与原线性规划等价的线性规划，这样就完成了初始化操作。

而辅助线性规划的初始解是容易构造的。由于x_0在每一个约束中的系数都为+1，那么我们把x_0作为换入变量，找到b_i的最小值b_l，把x_{l+n}作为换出变量执行一次转轴操作。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$-b_l > 0$。对于其余约束变为

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

由于$b_l \leq b_i$，所以$-b_l + b_i \geq 0$，故操作之后就满足辅助线性规划有一个可行的初始基本解，然后对其执行simplex即可。

举一个例子，比如下面的线性规划：

$$\begin{aligned} &\text{最大化} && 2x_1 - x_2 && (3.2)\\ &\text{满足约束} && x_3 = 2 - 2x_1 + x_2 \\ & && x_4 = -4 - x_1 + 5x_2 \\ & && x_j \geq 0, && j = 1, 2, \ldots, 4 \end{aligned}$$

这个线性规划不存在一个可行的初始基本解，故对其进行初始化操作。引入辅助线性规划：

$$\begin{aligned} &\text{最大化} && -x_0 \\ &\text{满足约束} && x_3 = 2 - 2x_1 + x_2 + x_0 \\ & && x_4 = -4 - x_1 + 5x_2 + x_0 \\ & && x_j \geq 0, && j = 0, 1, \ldots, 4 \end{aligned}$$

找到b_i的最小值是-4，那么以x_0为换入变量，x_4为换出变量执行转轴操作：

$$\begin{aligned} &\text{最大化} && -4 - x_1 + 5x_2 - x_4 \\ &\text{满足约束} && x_3 = 6 - x_1 - 4x_2 + x_4 \\ & && x_0 = 4 + x_1 - 5x_2 + x_4 \\ & && x_j \geq 0, && j = 0, 1, \ldots, 4 \end{aligned}$$

这时就找到了一个可行的基本解$(4, 0, 0, 6, 0)$，接下来对其进行simplex操作，将x_2换入，x_0换出：

$$\begin{aligned} &\text{最大化} && -x_0 \\ &\text{满足约束} && x_3 = \frac{14}{5} + \frac{4x_0}{5} - \frac{9x_1}{5} + \frac{x_4}{5} \\ & && x_2 = \frac{4}{5} - \frac{x_0}{5} + \frac{x_1}{5} + \frac{x_4}{5} \\ & && x_j \geq 0, && j = 0, 1, \ldots, 4 \end{aligned}$$

这个松弛型是辅助线性规划的最优解，$x_0 = 0$，故可以将x_0从辅助线性规划中移除，考虑原目标函数$2x_1 - x_2$，由于这时x_2是基变量，故需要把x_2用非基变量替换，故目标函数需要设定为

$$2x_1 - \left(\frac{4}{5} - \frac{x_0}{5} + \frac{x_1}{5} + \frac{x_4}{5}\right)$$

所以得到的线性规划为：

$$\begin{aligned}
&\text{最大化} && \frac{4}{5}+\frac{9x_1}{5}-\frac{x_4}{5}\\
&\text{满足约束} && x_3=\frac{14}{5}-\frac{9x_1}{5}+\frac{x_4}{5}\\
& && x_2=\frac{4}{5}+\frac{x_1}{5}+\frac{x_4}{5}\\
& && x_j\geq 0, \qquad\qquad j=1,2,3,4
\end{aligned}$$

这个线性规划就存在可行的初始基本解，可以将其返回给simplex过程。

进行初始化操作同样不会改变原线性规划的本质，这时的基本解$(x_1,x_2,x_3,x_4)=(0,\frac{4}{5},\frac{14}{5},0)$就满足(3.2)的所有约束。

3.2 单纯形法的时间复杂度

初始化操作与simplex操作的时间复杂度是同阶的，故只需考虑simplex的时间复杂度。

pivot操作即变量的代入及替换，单次操作的复杂度为$O(NM)$。但是pivot操作的执行次数并不是多项式的，不过在实际应用中，单纯形算法往往能十分快速地给出最优解。

需要注意的一点是，大部分的OI试题得出的限制条件都有一定的性质，而执行初始化操作会破坏这些性质，造成运行时间的无谓增加，且初始化操作较为繁琐，所以作者不建议大家在竞赛中使用，在本文的例题中我会介绍一些避免初始化操作的方法。

但这并不意味着线性规划问题不能在多项式时间内解决。椭球算法和内点算法均为解决线性规划的多项式时间算法。这里不加证明地给出这个定理：

定理3.1. 一般线性规划问题可以在多项式时间内解决。

但单纯形算法实现简单，并且实际运行时间并不明显比上述算法劣，不失为一种好的求解线性规划的方法。

4 将问题表示为线性规划

4.1 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：

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

4.2 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：

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

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$+1,-1$。

4.3 多物网络流问题

考虑这样一个问题：在最大流问题中，有n对源汇$(s_i,t_i), i = 1,2,\ldots,n$，每一对源汇对应的物品是不同的，但是它们共享同一个运输网络。对于每一对源

汇(s_i, t_i)，要求他们之间的流量不小于d_i，判断是否有可行解。

对于这个问题，要求所有节点对于每一种物品分别满足流量守恒，对于每一条边，所有物品的流量和不超过容量。由于只需要判断是否存在可行解，故目标函数为“空”：

$$
\begin{aligned}
&\text{最大化} && 0 && \\
&\text{满足约束} && \sum_i f_i(u,v) \le c(u,v), && (u,v) \in E, i = 1,2\ldots,n \\
& && \sum_v f_i(u,v) = \sum_v f_i(v,u), && u \in V, i = 1,2,\ldots,n \\
& && f_i(t_i, s_i) \ge d_i, , i = 1,2,\ldots,n && \\
& && f_i(u,v) \ge 0, && (u,v) \in E \cup \{(t_i, s_i) | 1 \le i \le n\}
\end{aligned}
$$

这个问题目前惟一已知的多项式解法是将其表示为线性规划，然后用多项式时间解决这个线性规划。

4.4 一道例题

例1. *precious stones*[4]

A、B两个人在洞穴中发现了n个石头，第i个石头对A来说价值为A_i，对B来说价值为B_i，石头是可以切割的，A、B认为石头的价值是均匀的（即石头的价值与其体积成正比）。现在A、B要分配这些石头，他们要求每个人得到的价值（一个人得到的价值是得到的石头在自己看来的价值）相同。求两人最大能得到的价值。

$n \le 50, 0 \le A_i, B_i \le 100$

设第i块石头有p_i分给了A，$1 - p_i$分给了B，那么两人得到的价值分别记为

$$V_a = \sum_{i=1}^{n} p_i A_i, V_b = \sum_{i=1}^{n} (1 - p_i) B_i$$

限制条件是$V_a = V_b$，要求最大化V_a。简单观察后，$V_a = V_b$这个限制可以放宽为$V_a \le V_b$，这是因为由于$A_i, B_i \ge 0$，所以当$V_a < V_b$时，可以将某部分B得到的

[4]试题来源：TCO08 round2 1000

石头分给A，A得到的价值不会减少，故增加（至少不减）目标函数的同时不违反限制条件，故得到下述线性规划：

$$\begin{aligned}
&\text{最大化} && \sum_{i=1}^{n} p_i A_i \\
&\text{满足约束} && \sum_{i=1}^{n} p_i(A_i + B_i) \le \sum_{i=1}^{n} B_i \\
& && p_i \le 1, && i = 1, 2, \ldots, n \\
& && p_i \ge 0, && i = 1, 2, \ldots, n
\end{aligned}$$

这样放宽之后，得到的线性规划便不需要执行初始化操作，接下来只需要直接运行单纯形算法即可。

5 整数线性规划

在线性规划问题中，有一类特殊的问题，它们的自变量取值范围是整数而非实数，这样的线性规划称为*整数线性规划*。

整数线性规划是一个NP-hard问题，所以在信息学竞赛中要求解一般整数线性规划是不实际的。但是有一些特殊的线性规划问题保证了当自变量取值范围为整数和实数得到的答案是一样的，例如容量限制均为整数的最大流问题以及最小费用流问题，虽然这些问题在问题描述中并没有要求流量必须是整数，但是由于问题的特殊性，它一定存在一个最优解满足所有边的流量是整数。这一点可以由针对网络流问题的算法的步骤得出。

但是有些问题并不能满足这一点，若此时直接使用单纯形法求解一个一般的线性规划，得到的结果往往是原问题的限制所不能达到的。下面给出一道作者认为应该存疑的例题：

例2. *志愿者招募加强版*[5]

有一个比赛持续N天，共有M种志愿者，每种志愿者的工作时间是连续的几段，第i种志愿者可以工作k_i段时间，每段时间从第s_{ij}天开始到第t_{ij}天结束，招

[5]试题来源：http://www.lydsy.com/JudgeOnline/problem.php?id=3265

募一个这种志愿者的花费为c_i，比赛要求第i天工作的志愿者数量不小于a_i，求最少花费。

$N \leq 1000, M \leq 10000$

显然题目要求招募的志愿者数量必须是整数，所以这是一个整数线性规划问题。这个问题的原问题是志愿者招募[6]这一题，原题每个志愿者的工作时间是一段连续的区间，经过转化后可以转为网络流建模，故使用单纯性算法解决该问题时可以保证求出的定义在实数上的解等于原问题的解。但是本题并没有这个保证，下面给出一个例子：

比赛共有三天，有三种志愿者：第一种志愿者工作时间为$[1,1],[2,2]$两段时间，招募代价为1。第二种志愿者工作时间为$[1,1],[3,3]$两段时间，招募代价为1。第三种志愿者工作时间为$[2,2],[3,3]$两段时间，招募代价为1。要求每天工作的志愿者数量不少于1。

显然这个问题的最优解为招募任意两种志愿者各一个，代价为2，但是当我们把变量取值范围放宽到实数时，最优解为三种志愿者各招募0.5个，代价为1.5，这确实满足了每天工作的志愿者数目为1，但并不符合原问题的描述。

我并没有好的方法解决这个问题，但是从通过情况上看，绝大部分人都是使用单纯形算法通过了此题，对此作者表示不解。

6 对偶问题

6.1 引言

考虑下面的线性规划：

$$\begin{aligned} &\text{最小化} && 7x_1 + x_2 + 5x_3 \\ &\text{满足约束} && x_1 - x_2 + 3x_3 \geq 10 \\ & && 5x_1 + 2x_2 - x_3 \geq 6 \\ & && x_i \geq 0, \qquad i = 1,2,3 \end{aligned}$$

[6]试题来源：NOI2008

由于每一个变量x_i非负，那么容易得到$7x_1 + x_2 + 5x_3 \geq x_1 - x_2 + 3x_3 \geq 10$，这一点可以通过逐项比较变量的系数得到。也就是说，目标函数的值至少是10。

那么，用同样的方法，能否再得到更“紧”的目标函数的下界呢？同时考虑两个限制条件，可以发现，$7x_1+x_2+5x_3 \geq (x_1-x_2+3x_3)+(5x_1+2x_2-x_3) \geq 10+6$。能否做到更好呢？我们考虑如何通过这个方法最大化目标函数的下界。通过这个方法找到的下界一定是如下形式：

$$\begin{aligned} f(y_1, y_2) &= y_1(x_1 - x_2 + 3x_3) + y_2(5x_1 + 2x_2 - x_3) \\ &= (y_1 + 5y_2)x_1 + (-y_1 + 2y_2)x_2 + (3y_1 - y_2)x_3, \qquad y_1, y_2 \geq 0 \end{aligned}$$

同时由于$(x_1 - x_2 + 3x_3) \geq 10, (5x_1 + 2x_2 - x_3) \geq 6$，故$f(y_1, y_2) \geq 10y_1 + 6y_2$。如果对于每一个变量$x_i$，满足$x_i$在目标函数中的系数要不小于它在$f(y_1, y_2)$中的系数（例如对于$x_1$，满足$y_1 + 5y_2 \leq 7$），那么由变量的非负性可以保证

$$\begin{aligned} 7x_1 + x_2 + 5x_3 &\geq (y_1 + 5y_2)x_1 + (-y_1 + 2y_2)x_2 + (3y_1 - y_2)x_3 \\ &= f(y_1, y_2) \geq 10y_1 + 6y_2 \end{aligned}$$

故在满足这个要求的同时我们要求最大化$10y_1 + 6y_2$，可以惊喜地发现，这个问题同样能表述为一个线性规划问题：

$$\begin{aligned} &\text{最大化} && 10y_1 + 6y_2 \\ &\text{满足约束} && y_1 + 5y_2 \leq 7 \\ & && -y_1 + 2y_2 \leq 1 \\ & && 3y_1 - y_2 \leq 5 \\ & && y_i \geq 0, \qquad i = 1, 2 \end{aligned}$$

我们称初始的线性规划为原问题，新得到的这个线性规划为对偶问题。如果对对偶问题再进行一次对偶操作，就又回到了原问题，也就是说，对偶过程是相互的，一个最大化问题可以对偶为一个最小化问题，一个最小化问题也同样可以对偶为一个最大化问题。下面给出对偶问题的形式化的定义：

定义6.1. 对偶问题

给定一个原始线性规划：

$$\begin{aligned} &\text{最小化} && \sum_{j=1}^{n} c_j x_j \\ &\text{满足约束} && \sum_{j=1}^{n} a_{ij} x_j \geq b_i, && i = 1, 2, \ldots, m \\ & && x_j \geq 0, && j = 1, 2, \ldots, n \end{aligned}$$

定义它的对偶线性规划为：

$$\begin{aligned} &\text{最大化} && \sum_{i=1}^{m} b_i y_i \\ &\text{满足约束} && \sum_{i=1}^{m} a_{ij} y_i \leq c_j, && j = 1, 2, \ldots, n \\ & && y_i \geq 0, && i = 1, 2, \ldots, m \end{aligned}$$

用矩阵可以更形象地表示为：

$$\begin{array}{lll} \text{最小化} & \mathbf{c}^T\mathbf{x} & \\ \text{满足约束} & A\mathbf{x} \geq \mathbf{b} & \text{互为对偶} \\ & \mathbf{x} \geq 0 & \end{array} \qquad \begin{array}{lll} \text{最大化} & \mathbf{b}^T\mathbf{y} & \\ \text{满足约束} & A^T\mathbf{y} \leq \mathbf{c} & \\ & \mathbf{y} \geq 0 & \end{array}$$

上述两个问题互为对偶问题，其中A^T表示矩阵A的转置。

6.2 对偶原理

首先形式化地给出上一节得到的结论：

定理6.1. （线性规划弱对偶性） 若$\mathbf{x} = (x_1, \ldots, x_n)$是原问题的一个可行解，$\mathbf{y} = (y_1, \ldots, y_m)$是对偶问题的可行解，那么

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

证明. 由于$\mathbf{y}$是对偶问题的一个可行解，那么

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

由于$x_i \geq 0$，所以

$$\sum_{j=1}^{n} c_j x_j \geq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j$$

同理，由于$\mathbf{x}$是原问题的可行解，$y_i \geq 0$，所以

$$\sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i \geq \sum_{i=1}^{m} b_i y_i$$

原式成立是因为如下等式显然成立

$$\sum_{j=1}^{n} \left(\sum_{i=1}^{m} a_{ij} y_i \right) x_j = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij} x_j \right) y_i$$

□

这只说明了我们通过对偶问题找到了原问题最优解的一个下界，而下面的定理说明了通过对偶问题求出的下界一定可以达到。

定理6.2. （线性规划对偶性） 若$\mathbf{x}^* = (x_1^*, \ldots, x_n^*)$与$\mathbf{y}^* = (y_1^*, \ldots, y_m^*)$分别是原问题及对偶问题的最优解，那么

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

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。

观察上述的两个定理，若$\mathbf{x}^*, \mathbf{y}^*$分别为原问题与对偶问题的最优解，那么在定理6.1中，等号必然成立。这就引出了下面的定理：

定理6.3. （互松弛定理） 若$\mathbf{x}, \mathbf{y}$分别是原问题及对偶问题的可行解，那么$\mathbf{x}, \mathbf{y}$都是最优解**当且仅当**下列两个条件被同时满足：

- 对于所有$1 \leq j \leq n$：满足$x_j = 0$或$\sum_{i=1}^{m} a_{ij} y_i = c_j$
- 对于所有$1 \leq i \leq m$：满足$y_i = 0$或$\sum_{j=1}^{n} a_{ij} x_j = b_i$

对偶原理将一个最大化问题与一个与之对应的最小化问题联系起来，在对偶过程中，原问题的每一个限制条件（即原矩阵的每一行）在对偶问题中产生了一个变量（即转置矩阵的每一列），而原问题的每个变量在对偶问题中就变成

了一个限制条件。定理6.2告诉我们这两个问题的最优解相等，而定理6.3则提供了一种原问题最优解与对偶问题最优解之间的关系。

现在考虑一些含有不那么“标准”的线性约束的线性规划对偶问题。若原问题中有约束$\sum_{j=1}^{m} a_{rj}x_j = b_r$，那么将这个约束拆为$\sum_{j=1}^{m} a_{rj}x_j \le b_r, -\sum_{j=1}^{m} a_{rj}x_j \le -b_r$，假设这两个约束对偶之后对应的变量为$y'_r, y''_r$，它们应该满足$y'_r, y''_r \ge 0$，在对偶之后的第$j$个约束为

$$\sum_{1\le i\le n, i\ne r} a_{ij}y_i + a_{rj}y'_r - a_{rj}y''_r = \sum_{1\le i\le n, i\ne r} a_{ij}y_i + a_{rj}(y'_r - y''_r)$$

如果令$y_r = y'_r - y''_r$，那么$y_r \in \mathbb{R}$，也就是说，一个等式约束在对偶之后变成了一个无限制的变量。

使用相同的方法可以继续考虑限制$\sum_{j=1}^{m} a_{rj}x_j \ge b_r$，它对偶之后得到了一个非正变量。

6.3 一些经典问题的对偶

6.3.1 最大流问题

接触过网络流的选手应该都知道，最大流的对偶问题是最小割问题，接下来我们来证明这一点。

最大流问题的线性规划描述在4.1中已经给出：

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

最小割问题可以表述为如下的线性规划：

$$\begin{aligned}
&\text{最小化} && \sum_{u,v\in E} c_{uv}d_{uv} && \\
&\text{满足约束} && d_{uv} - p_u + p_v \ge 0, && (u,v) \in E \\
& && p_s - p_t \ge 1 && \\
& && p_u, d_{uv} \in \{0,1\} &&
\end{aligned}$$

这是因为，一方面，一个割将所有点分为了两个不相交的集合S,T，其中$s \in S, t \in T$，这个割有代价$|cut(s,t)| = \sum_{u\in S,v\in T} c_{uv}$。令$p_u = [u \in S]$[7]，$d_{uv} = \max\{p_u - p_v, 0\}$，那么这个割就对应了这个线性规划的一个可行解。另一方面，要最小化$\sum_{u,v\in E} c_{uv}d_{uv}$，就要求当$u \in S, t \in T$时，$d_{uv} = 1$，否则$d_{uv} = 0$，那么这个线性规划的最优解就对应了一个割，故最小割问题可以用这个线性规划解决。

接下来考虑最大流的对偶问题。记由(u,v)的容量限制产生的变量为d_{uv}，由点u的流量守恒产生的变量为p_u，那么对偶之后的问题就是：

$$\begin{aligned}
&\text{最小化} && \sum_{(u,v)\in E} c_{uv}d_{uv} && \\
&\text{满足约束} && d_{uv} - p_u + p_v \geq 0, && (u,v) \in E \\
& && p_s - p_t \geq 1 && \\
& && d_{uv} \geq 0, && (u,v) \in E
\end{aligned}$$

这个形式已经非常接近最小割的模型了，但是限制条件$p_u \in \{0,1\}$没有被满足。不过可以证明，一定存在一个最优解满足这个限制。这一部分的证明与主题无关，此处略去，在附录中给出。这里直接给出结论：

定理6.4. （最大流最小割定理）给定一个源为s，汇为t的流网络，则s,t之间的最大流等于s,t之间的最小割。

6.3.2 二分图最大权匹配问题

二分图最大权匹配问题可以表示为下述线性规划：

$$\begin{aligned}
&\text{最大化} && \sum_{u,v\in E} c_{uv}d_{uv} && \\
&\text{满足约束} && \sum_{v\in Y} d_{uv} \leq 1, && u \in X \\
& && \sum_{u\in X} d_{uv} \leq 1, && v \in Y \\
& && d_{uv} \in \{0,1\} &&
\end{aligned}$$

[7]方括号内的式子是一个布尔表达式，这个式子表示当$u \in S$时，$p_u = 1$，否则$p_u = 0$

其中用d_{uv}表示点u,v是否匹配，c_{uv}表示匹配代价，X,Y分别为二分图左右部点集。类似最小割问题的证明，在这里去掉取值为整数的限制并不影响答案[8]，考虑这个问题的对偶问题。我们用p_u,p_v来表示上述线性规划中两类限制对偶之后对应的变量。对偶之后可得：

$$\begin{aligned} &\text{最小化} && \sum_{u\in X} p_u + \sum_{v\in Y} p_v && \\ &\text{满足约束} && p_u + p_v \geq c_{uv} && u\in X, v\in Y \\ & && p_u, p_v \geq 0 && \end{aligned}$$

这个问题有如下组合解释：给出一张带权二分图，要求给每一个节点定一个“顶标”，对于二分图中的每一条边要求满足这条边连接的两个顶点的顶标之和不小于边权，且所有顶标非负，求最小的顶标之和。我们称其为最小顶标和问题，这一张图与原图相同，故得到以下结论：

定理6.5. *在一张带权二分图中，最大权匹配等于最小顶标和。*

考虑这个问题的弱化版，若二分图中每条边的边权都为1，那么原问题就是二分图最大匹配问题。如果限制对偶之后的问题自变量取值为整数，那么对偶之后的限制就变成了对于有边相连的两个节点，至少有一个节点的顶标为1，也就是说对于一条边连接的两个顶点，至少要选择一个顶点来“覆盖”这条边。那么有以下结论：

定理6.6. *二分图中，最大匹配数等于最小点覆盖数。*

二分图的最大权匹配问题可以用KM算法或者直接套用最小费用最大流的算法高效求解。一般来说，我们倾向于将最小顶标和问题转化为最大权匹配问题解决。

例3. *MST*最小生成树[9]

给出一张带权无向图G以及这张图上的一棵生成树T，你可以修改一条边的边权，代价为权值改变量的绝对值。要求经过修改之后T是G的一棵最小生成树，要求最小化代价。

用n,m分别表示图G的点数以及边数，则$n \leq 50, m \leq 800$

[8]这是因为可以找到一个取值为整数的解得到的目标函数值等于最优解

[9]题目来源：Shoi2004 http://www.lydsy.com/JudgeOnline/problem.php?id=1937

由于目标是让T称为一棵最小生成树，那么对于在树上的边，进行的操作一定是减小它的边权，对于非树边就只能增加边权。考虑一条非树边e，记e连接的两个节点为u,v，那么如果T是最小生成树就要满足e的边权不小于树中u,v路径上的任意一条边的边权。也就是说若边t在树中u,v两点间的路径上，那么就要求$V(e)+\Delta e \geq V(t)-\Delta t$，即$\Delta t+\Delta e \geq V(t)-V(e)$。把原图中的每条边当做一个节点，按照是否出现在$T$中分类，那么这就是一个二分图，把$\Delta t$看做顶标，则限制$\Delta t+\Delta e \geq V(t)-V(e)$就是要求顶标和不小于$V(t)-V(e)$。那么这就是一个最小顶标和问题，可以转化为二分图最大权匹配问题解决。

6.4 对偶原理在信息学竞赛中的应用

6.4.1 寻找初始基本解

例4. *Flight Distance* [10]

给出一张n个节点的带权无向图，边(u,v)的边权为w_{uv}，可以随意增加或者减少一条边的边权，代价是边权的变化量。要求这张图在进行操作之后满足任意两点间直接相连的边长度不超过两点之间的最短路长度。求最小代价，以分数形式输出。

$n \leq 10, w_{uv} \leq 20$

记边的集合为E。对于每一条边$(u,v) \in E$建立两个变量t^+_{uv}, t^-_{uv}，表示这条边的变化量，那么在操作之后的边权为$w_{uv}+t^+_{uv}-t^-_{uv}$。另d_{uv}为u,v两点在操作之后之间的最短路的长度，那么由于题目要求连接两点的边的长度不超过两点之间最短路长度，同时这两点之间的最短路长度也不能小于直接相连的边的边权。所以得到$d_{uv}=w_{uv}+t^+_{uv}-t^-_{uv}$。另外，最短路之间需要满足$d_{uv} \leq w_{ui}+t^+_{ui}-t^-_{ui}+d_{iv}(u \neq v,(u,i) \in E)$，否则$d_{uv}$就可以被继续松弛，得到一条更短的路径，这个条件保证了d_{uv}为u,v之间的最短路长度。所以可以得到如下线

[10]试题来源：https://www.codechef.com/FEB12/problems/FLYDIST

性规划：

$$
\begin{aligned}
&\text{最小化} && \sum_{(u,v)\in E} t_{uv}^{+} + t_{uv}^{-} && \\
&\text{满足约束} && d_{uv} - t_{uv}^{+} + t_{uv}^{-} \geq w_{uv} && (u,v) \in E \\
& && -d_{uv} + t_{uv}^{+} - t_{uv}^{-} \geq -w_{uv} && (u,v) \in E \\
& && t_{ui}^{+} - t_{ui}^{-} + d_{iv} - d_{uv} \geq -w_{ui} && u \neq v, (u,i) \in E \\
& && d_{uv}, t_{uv}^{+}, t_{uv}^{-} \geq 0 &&
\end{aligned}
$$

在目标函数中，所有的系数都是非负的，也就是说对偶之后所有的约束条件的常数项非负，故可以不必进行初始化操作，所以可以对偶之后执行单纯形算法。

因为答案要求输出分数形式，所以需要实现一个分数类。

6.4.2 将线性规划模型转化为网络流问题

在最小费用流问题中，线性规划矩阵满足每一列有且仅有两个非零的位置，且它们的值分别为±1。这样每一个变量就可以看为一条边的流量，每一行的限制可以看成一个节点的流量守恒限制。具体的构图方法在下面的例题中详细说明。

一般来说，经过网络流构图之后使用解决网络流问题的算法求解这一类线性规划问题要比直接用单纯形算法求解快，这是因为解决网络流问题的算法都利用了这些线性规划问题的特殊性，而单纯形算法直接忽略了这些特殊性。这也就是我们考虑将线性规划问题转化为网络流问题的原因。

例5. *Chefbook* [11]

给出一张n个点m条边的有向图，每一条边(u,v)有边权L_{uv}，以及限制S_{uv}, T_{uv}。每次操作可以选择：*1.*将一个点u的所有出边权值加上P_u，其中P_u是任意正数。*2.*将一个点u的所有入边权值减去Q_u，其中Q_u是任意正数。每种操作在一个点只能执行一次，两种操作可以在一个点都被执行。可以任意执行操作，记最终每条边边权为L_{uv}^{*}，要求$S_{uv} \leq L_{uv}^{*} \leq T_{uv}$的同时最大化$\sum_{uv} L_{uv}^{*}$，**并输出方案**。

$n \leq 100, m \leq n^2$

[11]试题来源：https://www.codechef.com/JUNE15/problems/CHEFBOOK

简单观察后发现，这个问题可以被表示为线性规划模型，每一个限制可以表示为$S_{uv} \le L_{uv} + P_u - Q_v \le T_{uv}((u,v) \in E)$，要求最大化$\sum_{uv} L_{uv} + P_u - Q_v((u,v) \in E)$。计算过程中可以忽略目标函数的常数项，只需在最后加上忽略的常数即可。接下来合并同类项，P_u的系数为$\sum_{uv} 1((u,v) \in E)$，这就等于点u的出度，同理Q_v的系数就等于点v的入度的相反数。记点u的入度为$deg_{in}[u]$，出度为$deg_{out}[u]$，即可得到如下线性规划：

$$\begin{aligned}
&\text{最大化} && \sum_u (P_u deg_{out}[u] - Q_u deg_{in}[u]) \\
&\text{满足约束} && P_u - Q_v \le T_{uv} - L_{uv}, && (u,v) \in E \\
& && -P_u + Q_v \le L_{uv} - S_{uv}, && (u,v) \in E \\
& && P_u, Q_u \ge 0, && u \in V
\end{aligned}$$

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，所以考虑将其对偶，用x_{uv}表示前一种限制对偶后对应的变量，y_{uv}表示后一种限制对偶后对应的变量，那么可以得到：

$$\begin{aligned}
&\text{最小化} && \sum_u ((T_{uv} - L_{uv})x_{uv} + (L_{uv} - S_{uv})y_{uv}) \\
&\text{满足约束} && \sum_v (x_{uv} - y_{uv}) \ge deg_{out}[u], && u \in V \\
& && \sum_u (-x_{uv} + y_{uv}) \ge -deg_{in}[v], && v \in V \\
& && x_{uv}, y_{uv} \ge 0, && (u,v) \in E
\end{aligned}$$

考虑限制

$$\sum_v (x_{uv} - y_{uv}) \ge deg_{out}[u]$$

为了将其转化为最小费用流问题，故所有的限制应该为等式，一种解决方法是在此式中增加辅助变量$a_u \ge 0$，即可得到

$$\sum_v (x_{uv} - y_{uv}) - deg_{out}[u] - a_u = 0, u \in V$$

同理，对于第二类限制，增加辅助变量b_v，可得：

$$\sum_u (-x_{uv} + y_{uv}) + deg_{in}[v] - b_v = 0, v \in V$$

满足上述条件同时，最小化

$$\sum_u ((T_{uv} - L_{uv})x_{uv} + (L_{uv} - S_{uv})y_{uv})$$

将每一个变量看为一条边，变量的值是边的流量，这条边的费用就是这个变量在目标函数中的系数；每一个限制看为一个点，那么在限制u中若变量x_{uv}的系数为+1，就相当于变量x_{uv}所对应的边流入点u，若系数为−1，就相当于这条边从点u流出，那么每一个限制就相当于一个节点的流量守恒方程，也就是说这一个线性规划与一个费用流模型等价，接下来考虑如何构造出这个费用流图。

将第一类制对应的点记作u，第二类限制对应的点记作$v+n$，那么上述线性规划的对应建图应该为：对于每一条边$(u,v) \in E$，新建一条$v+n \to u$的边，费用为$T_{uv} - L_{uv}$，容量为∞（对应变量x_{uv}），以及一条$u \to v+n$的边，费用为$L_{uv} - S_{uv}$，容量为∞（对应变量y_{uv}）。由于变量a_u只出现了一次，那么就建立一条$u \to t$的边，容量为∞，费用为0(其中t表示网络流图中的汇，对于节点t没有流量守恒限制)，b_v类似。对于限制中的常数项$-deg_{out}[u]$，建立一条$u \to t$的边，容量为$deg_{out}[u]$，并且强制这条边满流（也就是说这条边的流量下界是$deg_{out}[u]$）。对于常数项$deg_{in}[v]$，建立一条$s \to v+n$的边，容量为$deg_{in}[v]$，流量下界为$deg_{in}[v]$。

至此，我们已经用一个网络流模型描述了整个线性规划，那么接下来就只需使用有下界的最大费用流算法解决。

上述建模是一个比较一般的建模方法，首先将所有限制变为等式限制，然后分别处理变量、辅助变量以及常数项，将模型转化为一个有下界的费用流问题。不过此题中还有更为巧妙的处理方法：

考虑任意一个可行解，这个解必然同时满足

$$\sum_v (x_{uv} - y_{uv}) \geq deg_{out}[u], \quad \sum_u (-x_{uv} + y_{uv}) \geq -deg_{in}[v]$$

如果把这两类共$2n$个限制相加，则不等式右侧为$\sum_u deg_{out}[u] - \sum_v deg_{in}[v] = 0$，左侧为$\sum_u \sum_v (x_{uv} - y_{uv}) + \sum_v \sum_u (-x_{uv} + y_{uv}) = 0$，所以得到$0 \geq 0$，由于$0 = 0$，那么要使这个等号成立，约束条件中的等号必然全部成立（否则将所有限制相加之后就得到$0 > 0$），也就是说，对于一个可行解，需要满足

$$\sum_v (x_{uv} - y_{uv}) = deg_{out}[u], \quad \sum_u (-x_{uv} + y_{uv}) = -deg_{in}[v]$$

如此便不需要添加辅助变量，更进一步，由于s的所有出边的容量和为$\sum_u deg_{out}[u]$，t的所有入边容量和为$\sum_v deg_{in}[v]$，而$\sum_u deg_{out}[u] = \sum_v deg_{in}[v]$，且除此之外图中不存在其它的可行流量，故只需要执行**最小费用最大流**算法即可保证所有等式成立，不需要对边加入流量下界的限制。

至此我们已经可以求出最终的答案，但是输出方案的问题仍没有解决。在这个问题中，我们已经知道了对偶问题的一个最优解的方案（每个变量的取值就等于费用流中对应边的流量），那么，由定理6.3(互松弛定理)可以得到：若$x_{uv} > 0$，那么原问题的最优解必然满足$P_u - Q_v = T_{uv} - L_{uv}$，若$y_{uv} > 0$，那么$-P_u + Q_v = L_{uv} - S_{uv}$，除此之外还应该满足原线性规划的限制条件，即$P_u - Q_v \leq T_{uv} - L_{uv}, -P_u + Q_v \leq L_{uv} - S_{uv}$，那么这就是一个差分约束系统，可以用最短路算法解决，对应的构图如下：对于限制$P_u - Q_v \leq T_{uv} - L_{uv}$，建立一条$v + n \to u$的长度为$T_{uv} - L_{uv}$的边，限制$-P_u + Q_v \leq L_{uv} - S_{uv}$，建立一条$u \to v + n$的长度为$L_{uv} - S_{uv}$的边，对于等式限制可以将其拆为两个不等式限制再连边。然后随意选择一个起点执行最短路算法，最后每个点的距离顶标就对应着一组最优解。

为了保证求出的最优解P, Q非负，可以将所有的P, Q均加上一个大常数，可以发现这样并不影响答案以及线性规划的限制。至此整个问题得以解决，时间复杂度为$O(\text{mincostflow}(2n, 2(n+m)))$

以这个问题为例，我测试了单纯形与网络流的运行时间时间差别，一共运行了$n = 50, m = 1000$的测试数据十组，使用费用流算法运行时间约为$60ms$，而使用单纯形算法运行时间约为$1630ms$。足以体现问题的独特性质对算法运行时间的影响。

例6. *Orz the MST*[12]

给出一个带权的连通无向图，图中有n个点m条边，对于其中的每条边i，在原来边权的基础上，其边权每增加1需要付出的代价为a_i，边权每减少1需要付出的代价为b_i，现在指定该图的一棵生成树，求通过修改边权，使得该生成树成为图的一棵最小生成树，需要付出的最少总代价。

$n \leq 300, m \leq 1000$

类似例3，把每一条边按照是否为树边分类，记树边集合为T，那么非树边集合为$E - T$，设第i条树边减小了权值x_i，第j条非树边增加了权值y_j，若i被j覆

[12]试题来源：http://www.lydsy.com/JudgeOnline/problem.php?id=3118

盖[13]，那么要求修改后的i的边权不超过j的边权，所以$w_i - x_i \le w_j + y_j \Leftrightarrow x_i + y_j \ge w_i - w_j$。定义函数$cover(i,j)$表示$i$是否被$j$覆盖，那么可以得到如下线性规划：

$$\begin{aligned}
&\text{最小化} && \sum_{i\in T} b_i x_i + \sum_{j\in E-T} a_j x_j \\
&\text{满足约束} && x_i + y_j \ge w_i - w_j, && i \in T, j \in E \setminus T, cover(i,j) = 1 \\
& && x_i, y_j \ge 0
\end{aligned}$$

这类似“增强版”的最小顶标和问题，考虑将其对偶，用s_{ij}表示对偶后的对应变量，得到：

$$\begin{aligned}
&\text{最大化} && \sum_{\substack{i\in T, j\in E\setminus T \\ cover(i,j)=1}} (w_i - w_j) s_{ij} \\
&\text{满足约束} && \sum_{\substack{j\in E\setminus T \\ cover(i,j)=1}} s_{ij} \le b_i, && i \in T \\
& && \sum_{\substack{i\in T \\ cover(i,j)=1}} s_{ij} \le a_i, && j \in E - T \\
& && s_{ij} \ge 0
\end{aligned}$$

由于$T \cap (E \setminus T) = \emptyset$，所以可以将第一类限制看做对左部节点的出度限制，第二类限制看做对右部节点的入度限制，对于左部节点i，最多只能流出b_i的流量，所以建立一条$s \to i$，容量为b_i，费用为0的边，对于右部节点j，建立一条$j \to t$，容量为a_i，费用为0的边，若i被j覆盖，那么由左部的i向右部的j连一条费用为$w_i - w_j$，容量为∞的边，然后执行最大费用流算法。

这个对偶后的模型就是二分图最优多重匹配问题。

例7. 战线防守[14]

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，在i号位置上建一座塔有c_i的花费，且一个位置可以建任意多的塔。有m限制，每个限制形如区间$[l_i, r_i]$中至少有d_i个塔。求满足所有限制的最少花费。

[13]不妨设j所连接的两个点为u_j, v_j，那么i被j覆盖即i在u_j, v_j两点间树上的路径上

[14]试题来源：ZJOI2013

$n \le 1000, m \le 10000$

由于每一个限制是对一段区间中的数量和的限制，所以为了能使用网络流构图，需要使用一点小技巧：记$s_i = \sum_{j \le i} x_j$，即对x作前缀和，那么每个限制就变成了两个变量相减的形式，于是可以得到如下线性规划：

$$\begin{aligned} &\text{最小化} && \sum_{j=1}^{n}(s_j - s_{j-1})c_j \\ &\text{满足约束} && s_{r_i} - s_{l_i-1} \ge d_i, && i = 1,2,\ldots,m \\ & && s_j - s_{j-1} \ge 0, && j = 1,2,\ldots,n \\ & && s_j \ge 0, && j = 0,1,\ldots,n \end{aligned}$$

然后用类似处理chefbook一题的方法进行费用流建图即可，这里不再赘述。

6.4.3 将线性规划模型转化为半平面交

当一个线性规划只有两个变量的时候，每一个限制条件就将解集限制在了二维平面上的一个半平面内，所以线性规划问题的解空间就是所有限制对应的半平面的交。

例8. *Equations*[15]

给出三个长度为n的数组a_i, b_i, c_i以及m次询问，每次询问给出两个参数s, t，求一组非负实数x_i，满足

$$\sum_{i=1}^{n} a_i x_i = s, \sum_{i=1}^{n} b_i x_i = t$$

的同时最大化$\sum_{i=1}^{n} c_i x_i$，对于每次询问输出这个最大值，或者判断无解。

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

[15]试题来源：http://poj.org/problem?id=3689

对于每一个询问，写出线性规划模型：

$$\begin{aligned}
&\text{最大化} && \sum_{i=1}^{n} c_i x_i \\
&\text{满足约束} && \sum_{i=1}^{n} a_i x_i = s \\
& && \sum_{i=1}^{n} b_i x_i = t \\
& && x_i \geq 0, \qquad i = 1, 2, \ldots, n
\end{aligned}$$

这个模型只有两个限制条件，故对偶之后只有两个变量，不妨记这两个变量为x, y，那么对偶之后得到如下线性规划：

$$\begin{aligned}
&\text{最小化} && sx + ty \\
&\text{满足约束} && a_i x + b_i y \geq c_i, \qquad i = 1, 2, \ldots, n \\
& && x, y \in \mathbb{R}
\end{aligned}$$

那么，每一个限制$a_i x + b_i y \geq c_i$就对应了一个半平面，满足所有限制的解x, y就在所有半平面的交内，这些半平面与询问无关，可以预处理出来。那么每一个询问就是要求出这个半平面交内使$sx + ty$最小的点，可以二分求解。当且仅当$sx + ty$无界时，原问题无解。所以总的时间复杂度为$O((n + m)log_2 n)$

6.4.4 小结

要高效解决一些线性规划问题最重要的环节是对特定的线性规划问题的特殊性的运用，例如只有两个变量的线性规划问题可以用半平面交算法解决，每个变量只在两个限制中出现，且系数和为0的线性规划问题可以用网络流算法解决等。对偶原理便是一种运用这些性质的手段。

7 总结

从线性规划的角度看待信息学竞赛中的问题，使得这些问题的描述更加清晰，同时这也带来了一种新的解决问题的方法。但是将问题一般化之后变不免损失了问题的特殊性，难以更高效地解决问题。

通过对具有特殊性的问题（例如网络流问题）的线性规划表示的总结，我们能够将一些具有特殊性的线性规划问题转化为其他问题，从而可以使用针对这些问题的算法来高效解决具有某些特殊性的线性规划问题，这就使我们在用一般性的角度看待问题的同时又能很好地利用问题的特殊性，达到高效解题的目标。

感谢

感谢父母的养育之恩。

感谢陈颖老师，余林韵教练，黄豪硕学长，张瑞喆学长的指导，感谢周聿浩等同学对本文提供意见与帮助。

感谢CCF提供学习和交流的机会。

附录

最大流对偶线性规划的最优解与割的对应关系

最大流的对偶线性规划如下：

$$\begin{aligned} &\text{最小化} && \sum_{uv} c_{uv}d_{uv} \\ &\text{满足约束} && d_{uv} - p_u + p_v \geq 0, && (u,v) \in E \\ & && p_s - p_t \geq 1 \\ & && d_{uv} \geq 0 \end{aligned}$$

记这个线性规划的最优解为(d^*_{uv}, p^*_u)，最优解的值为$z^* = \sum_{uv} c_{uv}d^*_{uv}$。为了证明存在一个割，使得$cut(s,t) = z^*$，首先需要证明该线性规划存在一个解$(d_{uv}, p_u)$使得$\sum_{uv} c_{uv}d_{uv} = \sum_{uv} c_{uv}d^*_{uv}$且满足$0 \leq p_u \leq 1$。

由于在目标函数中没有出现p_u，且在约束条件中p_u总是以差值$p_u - p_v$的形式出现，那么不妨假设$p^*_t = 0$。考虑解$(d_{uv}, p_u) = (d^*_{uv}, \min\{p^*_u, 1\})$，它得到的目标函数的值与最优解得到的值相同，下面证明这个解仍满足所有限制：

对于限制$d_{uv} - p_u + p_v \geq 0$，若$p^*_u > 1, p^*_v > 1$，那么这个限制等价于$d_{uv} \geq 0$，显然成立。

若$p_u^* > 1, p_v^* \le 1$，由于$p_u^* > p_u$，所以$d_{uv} - p_u + p_v > d_{uv}^* - p_u^* + p_v^* \ge 0$。

若$p_u^* \le 1, p_v^* > 1$，由于$p_u \le p_v$，故$p_u - p_v \le 0$，所以$d_{uv} \ge 0 \ge p_u - p_v$。

对于剩余限制显然成立，故解(d_{uv}, p_u)为一个可行解，更进一步，解(d_{uv}, p_u)得到的目标函数的值与最优解得到的值相同，故这是一个最优解。同理可证存在一个最优解满足$p_u \ge 0$。

下面证明一定存在一个割$cut(s,t)$，使得$cut(s,t) \le z^*$。

一个割可以用一个集合S表示，要求满足$s \in S, t \notin S$，表示将点集分割为$S, \complement_V S$ 两部分。设x是一个在$(0,1]$中均匀取值的随机变量，对于一个x，我们定义一个与之对应集合$S_x = \{u | p_u \ge x\}$，这个集合对应了一个割$cut_x(s,t)$，接下来我们考虑$|cut_x(s,t)|$的数学期望。

由于期望的线性性质，可以分别考虑每一条边对期望的贡献。对于边(u,v)，当且仅当$u \in S_x, v \notin S_x$时才会计算为割的代价。若$p_u \ge p_v$，那么这条边在割中的期望为$E_{uv} = c_{uv}(p_u - p_v)$，否则为0。由于在最小割的线性规划模型中限制了$d_{uv} \ge \max\{0, p_u - p_v\}$，故$c_{uv} d_{uv} \ge E_{uv}$，所以$E(|cut_x(s,t)|) = \sum_{uv} E_{uv} \le \sum_{uv} c_{uv} d_{uv} = z^*$，由于期望值是所有方案的平均值，所以一定存在$x \in (0,1]$，使得$|cut_x(s,t)| \le z^*$。证毕。

参考文献

[1] Thomas H.Cormen, Charles E Leiserson, Ronald L Rivest, Clifford Stein. Introduction to Algorithms. Second Edition.

[2] 胡伯涛，《最小割模型在信息学竞赛中的应用》

[3] 曹钦翔，《线性规划与网络流》

[4] Luca Trevisan. Stanford University-CS261: Optimization, Handout 15

[5] Vijay V. Vazirani. Approximation Algorithms: Chapter 10, Introduction to LP-Duality, Chapter 13, The primal-dual schema