线性规划与单纯形

孙雨奇，李佳蔚

主要内容

1. **线性规划问题 (*Linear Programming*)** 的简介与常见应用场景
2. 详解**单纯形 (*Simplex*)** 算法（正确性与时间复杂度证明）
3. 线性规划对偶性的直观证明，以及从一个更高的角度重新看待单纯形算法
4. 线性规划对偶性的一些应用（最短路，最大流最小割定理）
5. ***神秘附加内容？***

复习考试重点

- ✓ 本节课内容不适合出题，所以不用复习

- ❖ 这方面在 ACM 中最常见的应用是协助网络流建图，欢迎有兴趣的同学研究一下。

选题原因

1. 线性规划与单纯形算法大多数人在高中时期就已经学过，但是经过调查，很多人并不明白其背后的原理
2. 《算法导论》在这部分的内容讲的十分严谨，但是过于生硬，缺乏直观的例子与解释，令人望而生畏
3. 线性规划作为一经典问题，用途广泛；在 ACM 中虽然直接出现较少，但是很多问题都可以归结为线性规划模型，掌握好此部分内容对学习相关的算法与题目有很大的启发性

所以这节课希望能以简明直观的方式，使大家对线性规划有一个初步的了解，激起大家对该方面研究的兴趣。

一般线性规划问题

定义：线性规划问题，是在满足一系列**线性约束条件（*constrain set*）**的情况下，最大化（或最小化）一个**线性函数（目标函数 *objective function*）**。

形式化来说，给出一组变量 $x_1, x_2, \dots, x_n$，给出一个线性函数

$$f(x_1, \dots, x_n) = a_1x_1 + a_2x_2 + \cdots + a_nx_n = \sum_{i=1}^{n} a_ix_i$$

满足 m 个形如

$$f_j(x_1, \dots, x_n) = b_j,$$
$$f_j(x_1, \dots, x_n) \leq b_j,$$
$$f_j(x_1, \dots, x_n) \geq b_j.$$

的不等式约束，其中 f_j 为一个线性函数。**注意：线性规划中没有严格不等式！**

最终要求最大化或最小化目标函数 f.

一个简单例子

例1：最大化函数 $x_1 + x_2$

满足约束：

$$x_1 + 2x_2 \leq 4$$

$$4x_1 + 2x_2 \leq 12$$

$$-x_1 + x_2 \leq 1$$

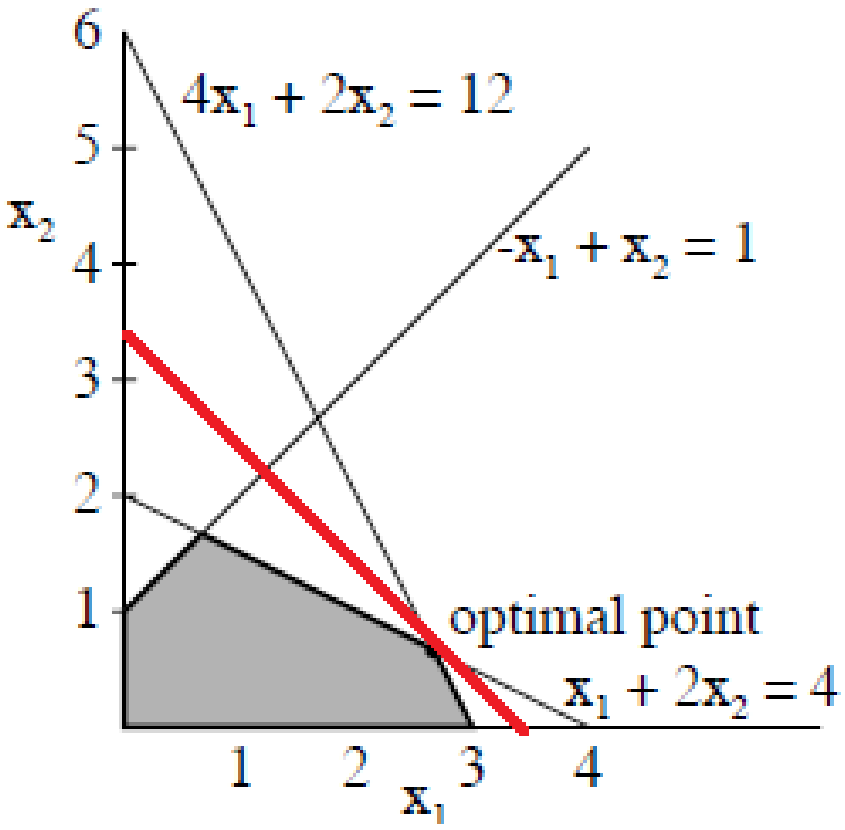
$$x_1, x_2 \geq 0$$

两个变量线性规划的高中解法

高中课本上讲过一种很直观的解决两个变量线性规划问题的解法。右图为例1的图解

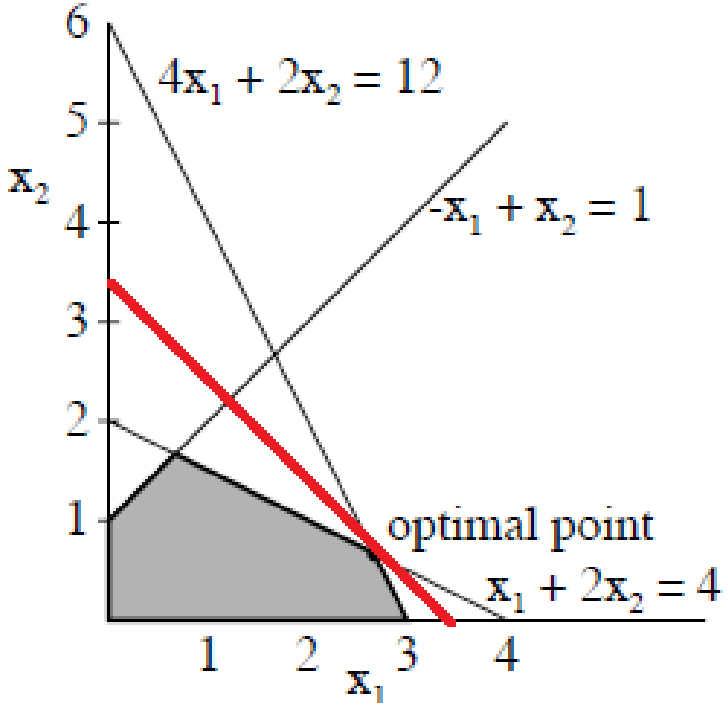
此做法在高维空间后就不好操作了，我们仍可以推测出 n 维空间中：一组约束表达一个半空间，所有半空间交集为一个凸集，将这个可行区域称为**单纯形。**

目标函数取某固定值 z 的区域为 $n-1$ 维超平面。若该超平面与单纯形交集不为空，则存在目标值为 z 的可行解。



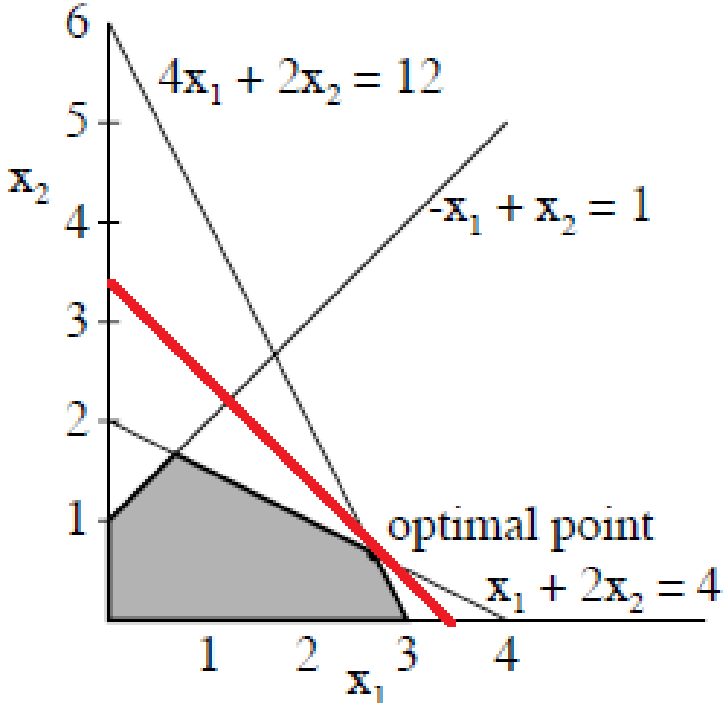
几个专业术语

- 满足所有约束式的变量 $x_1, \ldots, x_n$ 取值称为一个**可行解**，否则称为**不可行解**
- 可行解形成的区域（右图阴影部分）成为**可行区域**
- 目标函数在一特定点上的取值称为**目标值**
- 目标值最大的可行解是一个**最优解**



几个专业术语

- 一个线性规划若无可行解，则称此线性规划为**不可行的 (*infeasible*)**，否则称它为**可行的 (*feasible*)**
- 一个线性规划如果有可行解，但是没有有限的最优目标值，则称此线性规划是**无界的 (*unbounded*)**，否则称为**有界的 (*bounded*)**



线性规划的等价

为了对线性规划进行严谨地转化，这里定义线性规划**等价**的概念：

- 对两个最大化线性规划 L 和 L'，如果对 L 的每个目标值为 z 的可行解 $\bar{x}$，都存在一个对应的 L' 的目标值为 z 的可行解 $\bar{x}'$；且对 L' 的每个目标值为 z 的可行解 $\bar{x}'$，都存在一个对应的 L 的目标值为 z 的可行解 $\bar{x}$，则称 L 与 L' 是等价的。
- 对一个最大化线性规划和一个最小化线性规划，将以上定义中的 L' 的目标值替换为 $-z$ 即可

注意：该定义并不意味着可行解之间的一一对应关系

线性规划的两种规范形式

- 使用规范方式描述线性规划更为方便，这里用到两种形式：**标准型**与**松弛型**。
- 非正式地，标准型中的线性规划是满足线性**不等式**约束的一个线性函数最大化，而松弛类型的线性规划是满足线性**等式**约束的线性函数最大化。
- 通常使用标准型来表示线性规划，但在使用单纯形算法时，松弛型会更为方便。

标准型

在标准型中，我们已知 n 个实数 $c_1, c_2, \ldots, c_n$；m 个实数 $b_1, b_2, \ldots b_m$，以及 mn 个实数 a_{ij}，我们希望找到 n 个实数 $x_1, x_2, \ldots, x_n$，最大化：

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

满足约束条件：

$$\sum_{j=1}^{n} a_{ij} \leq b_i, \quad i = 1,2, \ldots m$$

$$x_j \geq 0, \quad j = 1,2, \ldots, n$$

标准型

最大化：

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

满足约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1,2,\dots m$$

$$x_j \geq 0, \quad j = 1,2,\dots,n$$

- 一共 $n+m$ 个约束，第二行的 n 个称为**非负约束**
- 一个任意线性规划不必有非负约束，但是标准型需要
- 可以用更紧凑的方式表示标准型：令

$$A = (a_{ij}), b = (b_i), c = (c_j), x = (x_j).$$

最大化：

$$c^T x$$

满足约束：

$$Ax \leq b$$

$$x \geq 0$$

于是可以用一个元组 (A, b, c) 来表示一个标准型线性规划

转化线性规划为标准型

最大化：

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

满足约束条件：

$$\sum_{j=1}^{n} a_{ij} \leq b_i, \quad i = 1,2,\dots m$$

$$x_j \geq 0, \quad j = 1,2,\dots,n$$

一个线性规划总可以转化为等价的标准型。一个线性规划可能有如下4个原因而不是标准型：

1. 目标函数可能是最小化，不是最大化
2. 可能有变量不具有非负约束
3. 可能有等式约束
4. 可能有不等式约束，但是是大于等于号

我们下面将逐一分析。

转化线性规划为标准型

1. 目标函数是最小化：将目标函数系数取反，然后将最小化改为最大化
2. 有变量不具有非负约束：假设变量 x_j 不具有非负约束，那么在每次 x_j 出现的地方都以 $x_j' - x_j''$ 替换，并增加非负约束 $x_j' \geq 0, x_j'' \geq 0$
3. 有等式约束：将每个等式约束换成一对不等式约束
4. 有大于等于号的约束：将该约束全部系数取反，然后变成小于等于号

这四步转化的等价性是比较显然的，这里不再赘述。

转化线性规划为标准型

例2: 将以下线性规划转化为标准型

最小化:

$$-2x_1 + 3x_2$$

满足约束:

$$x_1 + x_2 = 7$$

$$x_1 - 2x_2 \leq 4$$

$$x_1 \geq 0$$

转化线性规划为标准型

例2: 将以下线性规划转化为标准型

最小化:

$$-2x_1 + 3x_2$$

满足约束:

$$x_1 + x_2 = 7$$

$$x_1 - 2x_2 \leq 4$$

$$x_1 \geq 0$$

转化结果:

最大化:

$$2x_1 - 3x_2 + 3x_3$$

满足约束:

$$x_1 + x_2 - x_3 \leq 7$$

$$-x_1 - x_2 + x_3 \leq -7$$

$$x_1 - 2x_2 + 2x_3 \leq 4$$

$$x_1, x_2, x_3 \geq 0$$

松弛型

- 在使用单纯形算法时，将某些约束表示成等式约束更为方便。

- 更准确的说，松弛型是只有非负约束为不等式约束，而其他约束都是等式约束的形式

转化标准型到松弛型

使用一个巧妙的构造方式来转换：

对于不等式约束：

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

引入一个新的变量 x_{n+i}，并改写为两个约束：

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_j$$

$$x_{n+i} \geq 0$$

称 x_{n+i} 为一个**松弛变量**，它度量了原不等式左右之间的**松弛**或差别

转化标准型到松弛型

- 松弛型中通常会省略词语“最大化”和“满足约束”
- 同时使用变量 z 表示目标函数值，最终我们将松弛型表示成如下形式：

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

$$x_i = b_i - \sum_{j \in N} a_{ij} x_j$$

- 其中 v 为目标函数的一个常数项，方便计算。
- 因此松弛型也能用一个元祖 (N, B, A, b, c, v) 来表示

转化标准型到松弛型

例3：将例2中的标准型进行上述转化，会变成如下形式

最大化：

$$2x_1 - 3x_2 + 3x_3$$

满足约束：

$$x_4 = 7 - x_1 - x_2 + x_3$$

$$x_5 = -7 + x_1 + x_2 - x_3$$

$$x_6 = 4 - x_1 + 2x_2 - 2x_3$$

$$x_1, x_2, \ldots, x_6 \geq 0$$

- 在这种形式中，我们称在等式左边的变量为**基本变量**，等式右边的为**非基本变量**，出现在目标函数中的也只会为非基本变量。
- 确定了一组基本变量的值后就能解出非基本变量的值
- 在单纯形算法中，基本变量的集合和非基本变量的集合会不停变化（会进行一系列代数变换）。
- 一般将基本变量集合记为 B，将非基本变量集合记为 N。

线性规划算法

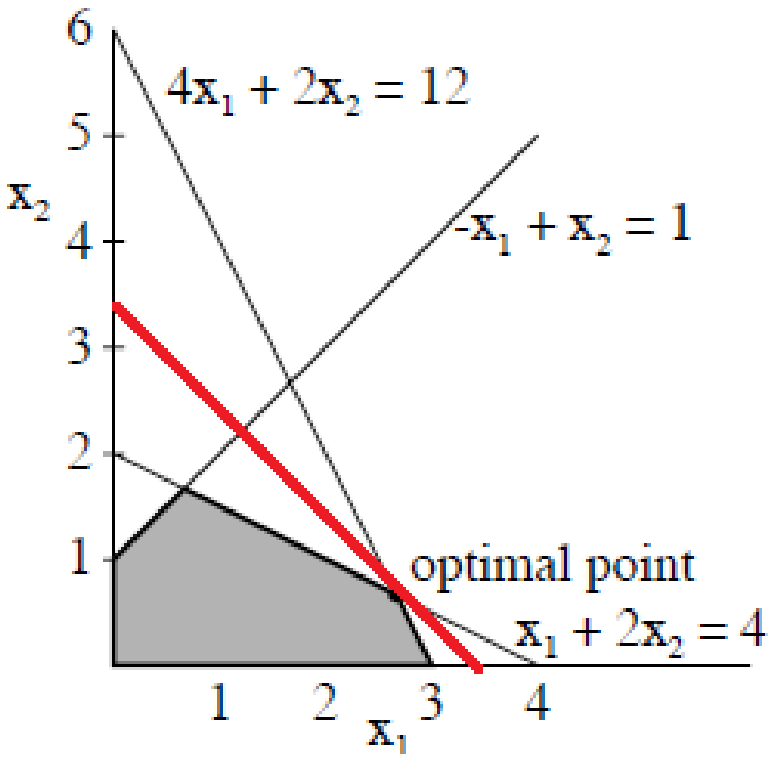
- 椭球算法：线性规划的第一个多项式时间算法，但实际运行缓慢
- 内点法：同样为多项式时间算法，输入较大时，性能与单纯形法相当，甚至更快
- 单纯形法：目前最常用，最好用的算法。一般情况下速度很快，但是极端情况下可以被卡到指数级。我们下面将重点介绍该算法

单纯形算法的几何直观

联想之前二维线性规划的解法，算导给出了单纯形算法的一个几何直观解释：

- 单纯形算法从单纯形的某个顶点开始，执行顺序迭代。
- 每次迭代中，它沿着单纯形的一条边从当前顶点移动到一个目标值不小于当前顶点的点。
- 最终会到达一个局部最优值，由于单纯形是凸的，所以局部最优就是全局最优。

由于人类难以理解高维空间，这个解释可能并不是很直观，等学完代数视角的单纯形算法后，可以回头再来品味一下这个几何的解释



单纯形算法

- 单纯形算法在代数上和高斯消元有点类似，都是从一个不容易看出解的状态开始，不停的进行等价转换，最终变成一个“形式很好”，能直接看出来许多信息的形式

- 单纯形算法流程非常简单，我们下面从一个例子看起。

单纯形算法

例4: 求如下松弛型最优值

$$z = 3x_1 + x_2 + 2x_3$$

$$x_4 = 30 - x_1 - x_2 - 3x_3$$

$$x_5 = 24 - 2x_1 - 2x_2 - 5x_3$$

$$x_6 = 36 - 4x_1 - x_2 - 2x_3$$

- **基本解**： 将所有非基本变量（等式右边的）设为 0，然后计算出基本变量（等式左边的）的值。若该基本解可行，则称其为一个**可行基本解**。我们之后将重点关注基本解。
- 例5中，基本解为一组可行解 $(0,0,0,30,24,36)$，目标函数值为 0

转动 (PIVOT)

例4: 求如下松弛型最优值

$$z = 3x_1 + x_2 + 2x_3$$

$$x_4 = 30 - x_1 - x_2 - 3x_3$$

$$x_5 = 24 - 2x_1 - 2x_2 - 5x_3$$

$$x_6 = 36 - 4x_1 - x_2 - 2x_3$$

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- 从基本解开始，想要增大 z 的值，第一直觉就是增大目标函数中系数为正的 x_1, x_2, x_3 的值。现在我们每次考虑增加一个变量。
- 比如想要增加 x_1 的值，并且贪心的想让 x_1 增加尽量多。但是假设我们令 x_1 取值为 10，则根据第三个等式，解得 $x_6 = -4$，违反了非负约束!
- 需要找到一个等式，它是限制 x_1 继续增大的瓶颈，称这个等式约束是最**紧**的。
- 对于 x_1，第三个等式是最紧的，它限制了 $x_1 \leq 9$。为了得到一个 $x_1 = 9$ 的基本解，我们把 x_1 与 x_6 位置互换。

转动 (PIVOT)

- 然后我们再把目标函数中和其它等式中的 x_1 用等式右侧的内容替换掉

例4: 求如下松弛型最优值

$$z = 3x_1 + x_2 + 2x_3$$

$$x_4 = 30 - x_1 - x_2 - 3x_3$$

$$x_5 = 24 - 2x_1 - 2x_2 - 5x_3$$

$$x_6 = 36 - 4x_1 - x_2 - 2x_3$$

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

转动 (PIVOT)

例4: 求如下松弛型最优值

$$z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4}$$

$$x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4}$$

$$x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2}$$

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- 然后我们再把目标函数中和其它等式中的 x_1 用等式右侧的内容替换掉
- 于是我们形成了一个新的等价松弛型，可以观察到以下事实:

1. 新松弛型的基本解依然合法，因为之前取的是最紧的约束
2. 新松弛型的基本解的目标值更优了
3. x_1 变成了基本变量，称它为**替入变量**，x_6 变成了非基本变量，称它为**替出变量**

- 我们将这称为一次**转动 (*Pivot*)**操作

转动 (PIVOT)

例4: 求如下松弛型最优值

$$z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4}$$

$$x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4}$$

$$x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2}$$

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- “称 x_{n+i} 为一个**松弛变量**，它度量了原不等式左右之间的**松弛**或差别”？
- **松弛**型这“松弛”两字?
- 现在是不是有了更深的理解?

单纯形算法

例4: 求如下松弛型最优值

$$z = 27 + \frac{x_2}{4} + \frac{x_3}{2} - \frac{3x_6}{4}$$

$$x_4 = 21 - \frac{3x_2}{4} - \frac{5x_3}{2} + \frac{x_6}{4}$$

$$x_5 = 6 - \frac{3x_2}{2} - 4x_3 + \frac{x_6}{2}$$

$$x_1 = 9 - \frac{x_2}{4} - \frac{x_3}{2} - \frac{x_6}{4}$$

- 观察到目标函数中还有系数为正的变量，所以我们重复之前的转动操作，直到目标函数中的系数全部为负。

单纯形算法

例4: 求如下松弛型最优值

$$z = 28 - \frac{x_3}{6} - \frac{x_5}{6} - \frac{2x_6}{3}$$

$$x_1 = 8 + \frac{x_3}{6} + \frac{x_5}{6} - \frac{x_6}{3}$$

$$x_2 = 4 - \frac{8x_3}{3} - \frac{2x_5}{3} + \frac{x_6}{3}$$

$$x_4 = 18 - \frac{x_3}{2} + \frac{x_5}{2}$$

- 观察到目标函数中还有系数为正的变量，所以我们重复之前的转动操作，直到目标函数中的系数全部为负。
- 如左侧方程组目标函数全部为负，此时算法结束
- 显然该松弛型的基本解就是最优解 $(8,4,0,18,0,0)$，最优值为 28
- 算导引入线性规划对偶性来证明此基本解为最优解。但我的理解是该最终松弛型与原线性规划是**等价**的，所以该松弛型的最优解就是原线性规划的最优解。

单纯形算法

- 以上就是一次单纯形算法的流程，其实非常简单
- 例4的性质非常好，其实这个算法还有许多细节问题：

1. 初始基本解不可行怎么办？怎么判定一个线性规划是否可行？
2. 如何判断无界的情况？
3. 算法是否会终止？
4. 算法一定停止在最优解？
5. 如何选择替入和替出变量？

- 在解决这些问题之前，先看一下单纯形的具体代码实现。

单纯形算法

- 代码中的变量命名与之前的命名相同
- bool 数组 basic 表示一个变量是否为基本变量。e 表示替入变量，l 表示替出变量。
- initialize_simplex 用于生成一组基本可行解

```
double rel;
int simplex(bool is_init)
//结果存到 rel 里，函数返回 -1 表示无解， 返回 0 表示正常，返回 1 表示无界
{
    if (!is_init) //假如是从is_init里调用的该函数，则不用判断合法性
        if (!initialize_simplex())
            return -1;
    int i, e, l;
    while (e = get_pivot_index())
    {
        double del = 1e12;
        for (i = 1; i <= k; i++)
            if (basic[i])
                if (dcmp(a[i][e]) == 1)
                    if (b[i] / a[i][e] < del)
                        l = i, del = b[i] / a[i][e];
        if (del == 1e12)
            return 1; //无界
        else
            pivot(l, e);
    }
    rel = v;
    return 0;
}
```

单纯形算法的终止性

- 迭代时有可能保持目标值不变，这种情况称为**退化**，退化可能会阻止单纯形算法终止，称为**循环**。
- 设 (A, b, c) 是一个线性规划的标准形式。给定基本变量的一个集合 B，那么关联的松弛型是唯一确定的
- 循环是 SIMPLEX 可能不会终止的唯一原因（《算法导论》引理29.3,29.4)）
- 如果 SIMPLEX 在至多 $\binom{n+m}{m}$ 次迭代内不会终止，那么它是循环的
- 证明：基本变量集合 B 确定了唯一一个松弛型。总共 $n+m$ 个变量，$|B| = m$，所以至多有 $\binom{n+m}{m}$ 种选择 B 的方式

单纯形算法的终止性

- 如何解决循环问题？循环在理论上是可能的但实际上非常罕见，可以通过选择替入替出变量的策略解决。一种十分常用的策略是 Bland Rule
- Bland Rule：总是选择下标最小的变量来打破相等的目标值假如有多个可选的替入变量，选择序号最小的；假如有多个可选的替出变量（此时这些不等式一样紧），同样选择编号最小的。
- 证明详见参考文献，这里不再赘述

构造初始可行解

- 在存在 $b_i < 0$ 的情况时，基本解是不合法的
- 引入辅助变量 x_0，构造辅助线性规划 L_{aux}：

最大化：

$$-x_0$$

满足约束：

$$\sum_{j=1}^{n} a_{ij} - x_0 \leq b_i, \quad i = 1,2,\dots m$$

$$x_j \geq 0, \quad j = 0,1,2,\dots,n$$

对于辅助线性规划：

1. 以 x_0 为替入变量，选择基本解中负值最大的基本变量作为替出变量，做一次 pivot 后即可构造出一组基本可行解
2. 对当前可行解重复调用 pivot，直到得到辅助线性规划最优解。

若最优解 $-x_0 = 0$，则初始问题是可行的，同时我们获得了一组基本可行解。否则初始问题不可行。

- 这个巧妙构造的正确性比较显然，这里不再赘述

线性规划的对偶

- 线性规划的**对偶 (*duality*)** 是一个非常神奇的存在
- 可以利用对偶来证明一些很强的性质，比如最大流最小割定理， ***The Equilibrium Theorem*** （平衡定理？）等。理解了对偶将会对理解线性规划有很大的帮助
- 对偶线性规划一般是相对于一个标准型的线性规划而言的。
- 当表示一个对偶线性规划时，我们称初始的线性规划为**原始 (*primal*)** 线性规划

对偶的定义

- 原始线性规划：

最大化：

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

满足约束条件：

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1,2,\dots m$$

$$x_j \geq 0, \quad j = 1,2,\dots,n$$

- 对偶线性规划：

最小化：

$$\sum_{i=1}^{m} b_i y_i$$

满足约束条件：

$$\sum_{i=1}^{m} a_{ij} y_i \geq c_i, \ j = 1,2,\dots,n$$

$$y_i \geq 0, \ i = 1,2,\dots m$$

对偶的定义

- 原始线性规划：

最大化：

$$c^T x$$

满足约束：

$$Ax \leq b$$

$$x \geq 0$$

- 对偶线性规划：

最小化：

$$yb^T$$

满足约束：

$$y^T A \geq c$$

$$y \geq 0$$

对偶的定义

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\leq b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\leq b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\leq b_m$
	$\geq c_1$	$\geq c_2$	$\cdots$	$\geq c_n$	

线性规划弱对偶性

- 对于原始线性规划的任意一组可行解 x，和对偶线性规划的一组可行解 y，都有：

$$c^T x \leq y^T b.$$

原始线性规划的任意一个可行解的目标值一定小于等于对偶线性规划的任意一个可行解的目标值。

- 证明：

$$c^T x \leq y^T A x \leq y^T b.$$

从右图中可以较为直观地看出来

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\leq b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\leq b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\leq b_m$
	$\geq c_1$	$\geq c_2$	$\cdots$	$\geq c_n$	

线性规划弱对偶性的简单推论

- 若能找到原始线性规划的一组可行解 x，和对偶线性规划的一组可行解 y，有：

$$c^T x = y^T b.$$

则 x, y 分别为两组线性规划的最优解

- 那么是否能从原始线性规划的一组“最优解”（单纯形算法给出的解），构造出目标值相等的对偶线性规划的一组解呢？假如能构造出来，我们就可以证明单纯形算法的正确性了。

线性规划对偶性

- 假设 $x = (x_1, x_2, \ldots, x_n)$ 是 SIMPLEX 在原始线性规划 (A, b, c) 上返回的最优解。令 N, B 分别为最终松弛型非基本变量和基本变量的集合，令 c' 表示最终松弛型中的系数。令

$$y_i = \begin{cases} -c'_{n+i} & \text{若 } (n+i) \in N \\ 0 & \text{其他} \end{cases}$$

则 x 是原始线性规划的一组最优解，y 是对偶线性规划的一组最优解，以及

$$c^T x = y^T b$$

线性规划对偶性

- 我有一个非常优雅的证明，可惜这里地方太小写不下了。

- 提示：从这张图中发现玄机

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\leq b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\leq b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\leq b_m$
	$\geq c_1$	$\geq c_2$	$\cdots$	$\geq c_n$	

线性规划对偶性

- 引理：若某个线性规划满足：

$$b_i \geq 0, c_j \leq 0, \forall i, j$$

则此时取所有 x_j, y_i 为 0，显然是两组线性规划的最优解

- 正确性是显然的

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\leq b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\leq b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\leq b_m$
	$\geq c_1$	$\geq c_2$	$\cdots$	$\geq c_n$	

线性规划基本定理

- 任何以标准型给出的线性规划 L 可能是如下情形:

1. 有一个有限目标值的最优解
2. 不可行
3. 无界

如果 L 是不可行的，SIMPLEX 返回“不可行”；如果 L 是无界的，SIMPLEX 返回“无界”；否则，SIMPLEX 返回一个有限目标值的最优解

最短路的线性规划模型

- 已知有向图 $G=(V,E)$，设原点为 s，终点为 t，$w(u,v)$ 表示 u 到 v 的路径长度，d_i 表示到达 i 的最短距离，则我们可以列出如下的线性规划模型：
- 最大化：

$$d_t$$

满足约束：

$$d_v \le d_u + w(u,v), (u,v) \in E$$

$$d_s = 0$$

最短路的线性规划模型

- 最大化：

$$d_t$$

满足约束：

$$d_v \leq d_u + w(u,v), (u,v) \in E$$

$$d_s = 0$$

- 要求最短路径，为什么是最大化 d_t？

- 把该标准型转成对偶后，又可以看出什么？

网络流的线性规划模型

- 已知有向图 $G=(V,E)$，其中每条边 $(u,v)\in E$ 有一个非负的容量 $c(u,v)\geq 0$，以及两个特别的顶点，源点 s 和汇点 t。则有如下模型：
- 最大化

$$\sum_{v\in V} f_{sv} - \sum_{v\in V} f_{vs}$$

满足约束

流量限制： $f_{uv} \leq c(u,v)$

流量守恒： $\sum_{v\in V} f_{uv} = \sum_{v\in V} f_{vu}, \forall u \in V - \{s,t\}$

非负限制： $f_{uv} \geq 0$

最大流与最小割

原始线性规划

最大化 $\sum X_p$

满足约束：

$$\sum_{P:e\in p} X_p \le C_e$$

$$X_p \ge 0$$

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\le b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\le b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\le b_m$
	$\ge c_1$	$\ge c_2$	$\cdots$	$\ge c_n$	MaxFlow

最大流与最小割

对偶线性规划

最小化 $\sum c_e y_e$

满足约束：

$$\sum_{e \in p} y_e \geq 1$$

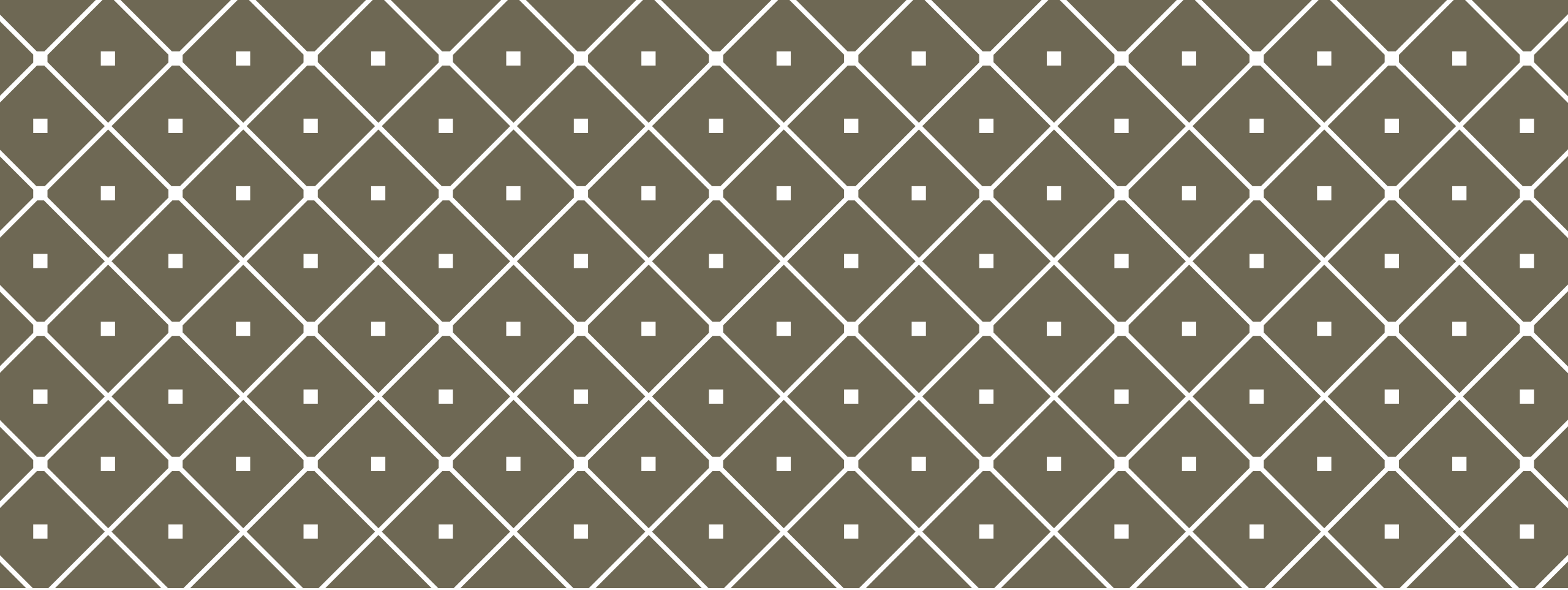
$$y_e \geq 0$$

	x_1	x_2	$\cdots$	x_n	
y_1	a_{11}	a_{12}	$\cdots$	a_{1n}	$\leq b_1$
y_2	a_{21}	a_{22}	$\cdots$	a_{2n}	$\leq b_2$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
y_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}	$\leq b_m$
	$\geq c_1$	$\geq c_2$	$\cdots$	$\geq c_n$	MinCut

遗留的问题

由于时间问题，我们在之前的证明中遗留了几个定理的证明，现列举如下，希望有兴趣的同学可以课后研究一下

1. 用于选择替入、替出变量的 Bland 法则
2. PPT 第 32 页的有关终止性的定理，在算导514页有证明



零和博弈与线性规划

Tetris AI 技术分享

李佳蔚，梁家硕，吴克文

真·选题原因

- Tetris AI 不应该使用最大最小搜索。为了解决这个问题，构建了一个博弈模型。
- 看过《美丽心灵》这部电影，所以一开始认为是纳什均衡。当然此问题也确实可使用纳什均衡解决。
- 但其实零和博弈有特殊性，所以可以转化为线性规划求解。学了一波线性规划之后，发现很有意思，并决定分享一下

TETRIS AI 建模

- 文字不易描述，请期待上课讲解

零和博弈

- **零和博弈**（***Zero-Sum Game***），又称**零和游戏**或**零和赛局**，与非零和博弈相对，是博弈论的一个概念，属非合作博弈。
- 零和博弈表示所有博弈方的利益之和为零或一个常数，即一方有所得，其他方必有所失
- 之后使用 I, II 指代二人零和游戏中的 Player1, Player2
- 常用**正则形式**（***Strategic Form***）来表示一个二人零和博弈，即一个三元组 (X, Y, A)，其中：

1. X 是一个非空集合，表示 I 的策略集
2. Y 是一个非空集合，表示 II 的策略集
3. A 是一个定义在 $X \times Y$ 上的实矩阵，

$$A(x, y),\ x \in X, y \in Y$$

表示若 I 选择策略 x，II 选择策略 y，则 I 从 II 处赢得了 $A(x, y)$ 的价值。

例题：ODD OR EVEN

- I, II 同时说出数字 1 或者 2，若数字之和为奇数，则 I 赢，否则 II 赢。
- 这里 $X = \{1,2\}$, $Y = \{1,2\}$，A 由右图给出：
- 这个游戏中有一个玩家有优势，猜猜是哪个?
- 怎样又算“有优势”呢?

$$
\begin{array}{lcc}
 & \text{II (even)} & y \\
 & & \begin{array}{cc} 1 & 2 \end{array} \\
\text{I (odd)} \quad x & \begin{array}{c} 1 \\ 2 \end{array} & \begin{pmatrix} -2 & +3 \\ +3 & -4 \end{pmatrix}
\end{array}
$$

$$
A(x, y) = \text{I's winnings} = \text{II's losses.}
$$

例题：ODD OR EVEN

- 若 I 可以采取某种策略策略，使得无论 II 使用怎样的策略，I 的收入期望恒正(稳赚不赔)，则我们称“I 有优势”。
- 考虑 I 在选择数字 1 的概率为 p则数字 2 的概率为 $1-p$。要想在 II 无论选 1 还是 2，收益都相同，则可列出等式：

$$-2p+3(1-p)=3p-4(1-p)$$

解得 $p=\frac{7}{12}$。此时无论 II 如何“勾心斗角”，I 都是每回合期望收益 $\frac{1}{12}$

$$\begin{array}{cccc} & & \text{II (even)} & y \\ & & & 1 \quad 2 \\ \text{I (odd)} & x & 1 & \begin{pmatrix} -2 & +3 \\ +3 & -4 \end{pmatrix} \\ & & 2 & \end{array}$$

$$A(x,y) = \text{I's winnings} = \text{II's losses.}$$

例题：ODD OR EVEN

- 我们把 *I* 的这种策略称为**平衡策略**（***equalizing strategy***）
- 看起来这个平衡策略每次期望最多也只能赚 $\frac{1}{12}$，有没有赚的更多的策略?
- 答案是没有，假如 *II* “会玩”的话。

$$
\begin{array}{lll}
 & \text{II (even)} & y \\
 & & \begin{array}{cc} 1 & 2 \end{array} \\
\text{I (odd)} \quad x & \begin{array}{c} 1 \\ 2 \end{array} & \begin{pmatrix} -2 & +3 \\ +3 & -4 \end{pmatrix}
\end{array}
$$

$$A(x, y) = \text{I's winnings} = \text{II's losses}.$$

例题：ODD OR EVEN

- 对于 II，设他选择 1 的概率为 q，则同样可以用平衡策略的思路列出方程：

$$2q - 3(1-q) = -3(1-q) + 4q$$

解得 $q = \frac{5}{12}$，代入后可算出 II 在此策略下，无论 I 采取怎样的策略，II 都期望每局损失 $\frac{1}{12}$。不会更多或更少

- 可以发现 I 能通过采取适当策略，使得收益至少为 $\frac{1}{12}$，而 II 可以采取策略使得损失最大为 $\frac{1}{12}$。

- 所以我们称 $\frac{1}{12}$ 为该游戏的 ***value***（值？），两人采取的策略称为极大极小策略（***Minimax***），或最优策略

几个名词的补充

- **纯策略 (*pure strategy*)**：X, Y 中的元素称为纯策略
- **混合策略 (*mixed strategy*)**：按照一些比例在 X, Y 中**随机**选择策略。Odd or Even 中 I, II 的平衡策略就属于混合策略。纯策略 $x \in X$ 也可以认为是选择 x 的概率为 1 的混合策略。
- **有限游戏 (*Finite Game*)**：若一个双人零和游戏 (X, Y, A) 中的 X, Y 有限集合，则我们称这个游戏为**有限游戏**，有时我们也称其为**矩阵游戏**，Odd or Even 和我们的 Tetris AI 的模型都是有限游戏。

❖ 是否所有有限游戏都像 Odd or Even 那样，存在达到平衡的 value，和相应的极大极小策略?

MINIMAX THEOREM

- ***Minimax Theorem***：对于任意一个有限游戏，都有：

1. 存在一个数 V，表示这个游戏的值 (value)
2. I 有一个混合策略，使得 I 期望至少收益 V，无论 II 采取怎样的策略
3. II 有一个混合策略，使得 II 期望最多损失 V，无论 I 采取怎样的策略

➢ 该定理由**约翰·冯·诺伊曼 (*John von Neumann*)** 在 1928 年的一篇关于社会博弈论文中证明，这篇论文宣告了博弈论的正式诞生。该定理被称为博弈论的基本原理。冯·诺依曼也因此被称为博弈论之父

一个重要的引理

- 对于一个有限游戏 (X, Y, A)，$|X| = m, |Y| = n$。I 的一个混合策略可以表示为向量

$$p = (p_1, p_2, \dots p_m)^T, \sum_1^m p_i = 1$$

II 的一个混合策略表示为

$$q = (q_1, q_2, \dots, q_n)^T, \sum_1^n q_i = 1.$$

I 的收益可以表示为 $p^T A\, q.$

- 对于 II 的任意一种混合策略 q，I 能使得收益最大策略中，一定包括至少一种纯策略。

矩阵游戏的线性规划模型

❖利用极大极小策略的思想和之前的引理，从 I 的视角，可列出以下模型：

最大化：

$$\min_{1\le j\le n} \sum_{i=1}^{m} p_i a_{ij}$$

满足约束：

$$p_1 + p_2 + \ldots + p_m = 1$$

$$p_i \ge 0,\ i = 1,2,\ldots m$$

❖ 目标函数不是一个线性函数？可以做个巧妙的转化，新加一个变量v：

最大化：

$$v$$

满足约束：

$$v \le \sum_{i=1}^{n} p_i a_{ij},\ j = 1,2,\ldots n$$

$$p_1 + p_2 + \ldots + p_m = 1$$

$$p_i \ge 0,\ i = 1,2,\ldots m$$

矩阵游戏的线性规划模型

❖ I 的视角：

最大化：

$$v$$

满足约束：

$$v \le \sum_{i=1}^{m} p_i a_{ij}\,,\ \ j = 1,2,\dots n$$

$$p_1 + p_2 + \dots + p_m = 1$$

$$p_i \ge 0\,,\ \ i = 1,2,\dots m$$

❖ II 的视角：

最小化：

$$v$$

满足约束：

$$v \ge \sum_{j=1}^{n} q_j a_{ij}\,,\ \ i = 1,2,\dots m$$

$$q_1 + q_2 + \dots + q_n = 1$$

$$q_j \ge 0\,,\ \ i = 1,2,\dots n$$

矩阵游戏的线性规划模型

- 可以发现从 I, II 两个视角列出的线性规划模型是对偶的，利用线性规划对偶性，我们就证明了 *Minimax Theorem*！

- 对于矩阵游戏，现在有了一个完美的解决方案！

双矩阵游戏

- 两人零和博弈的一个简单扩展，就是两人非零和博弈，这类博弈在实际生活中也经常出现。

- 这里不再给形式的定义了，可以简单的理解为零和博弈中的矩阵 A ，变成两个矩阵 A, B ， $A_{i,j}, B_{i,j}$ 分别表示 I, II 的收益。

囚徒困境

- 两个共谋犯罪的人被关入监狱，不能互相沟通情况。
- 如果两个人都不揭发对方，则由于证据不确定，每个人都坐牢一年；若一人揭发，而另一人沉默，则揭发者因为立功而立即获释，沉默者因不合作而入狱四年；若互相揭发，则因证据确实，二者都判刑三年

$$\begin{array}{c} \\ \text{cooperate} \\ \text{defect} \end{array} \begin{array}{c} \begin{array}{cc} \text{cooperate} & \text{defect} \end{array} \\ \begin{pmatrix} (3,3) & (0,4) \\ (4,0) & (1,1) \end{pmatrix} \end{array}$$

纳什均衡

- 纳什均衡 **(*Nash equilibria*)**：任何表示成正则表示 (*strategic form*) 的有限 n 人游戏，都至少有一个策略平衡点

- 1951 年首次被 John Nash 提出，是 *Minimax Theorem* 的推广

- 找平衡点是一个非常困难的问题，目前没有好的算法

提问环节

- 欢迎与我联系交流，右边是我的联系方式，或者

QQ : 1360230697

邮箱： 1360230697@qq.com

- PS: 目前这部分博弈内容题目不多，所以我考虑往各类比赛平台上投一些这类题目。

参考资料

1. *算法导论，第29章*
2. *“Linear Programming: A Concise Introduction”, Thomas S. Ferguson* (强烈推荐)
3. *曹钦翔 线性规划与网络流*
4. *“Linear Programming and Extensions”, George B. Dantzig*
5. *“GAME THEORY : Part II. Two-Person Zero-Sum Games” Thomas S. Ferguson*