第六章 组合优化问题的有效算法

6.1 几个重要算法的计算复杂度讨论

一、线性规划问题的单纯形算法不是多项式时间算法

Klee 和 Minty 1972 给出了第一这样的例子。

首先要有一个有界多面体，它有指数个顶点。例如，一个立方体：

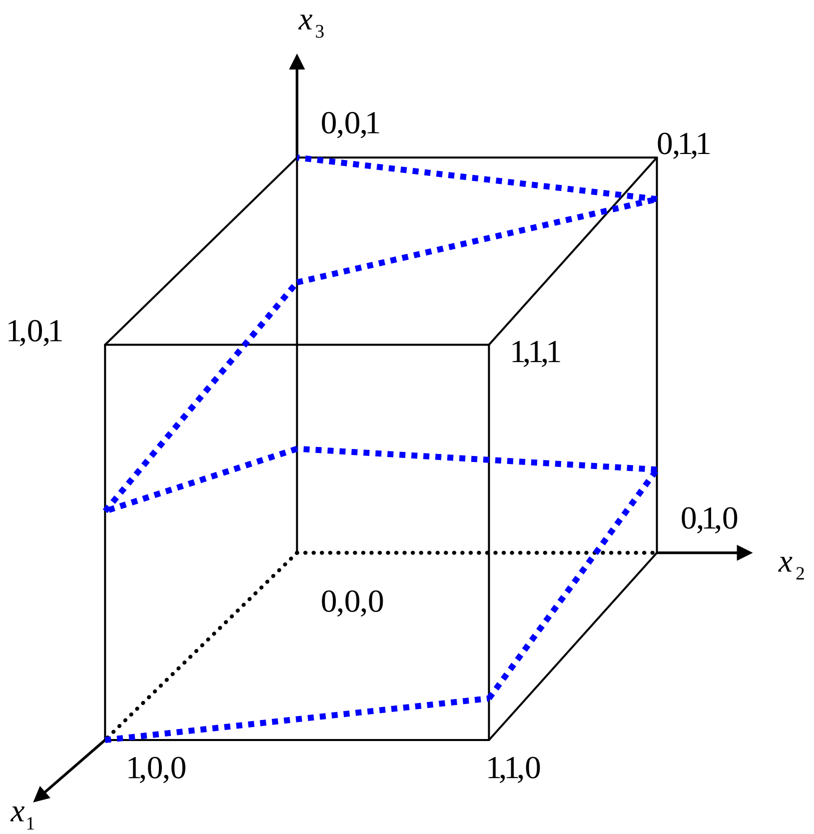
$$0 \le x_j \le 1,\ j = 1,2,3$$

有 2^3＝8 个顶点。

d 一维立方体：

$$0 \le x_j \le 1,\ j = 1,2,\cdots,d$$

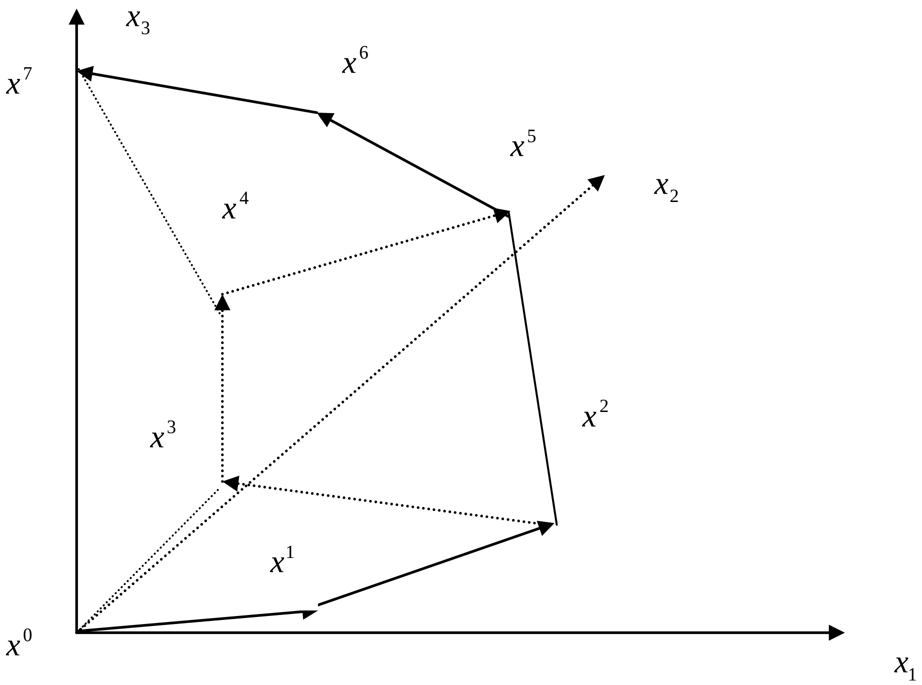
有 2^d 个顶点。每一个顶点对应于 $\{x_1, x_2, \cdots, x_d\}$ 的一个子集，使得该子集中的元素等于 1，其余为 0。



对于$0<\varepsilon<\frac{1}{2}$，构造立方体：

$$\varepsilon \le x_1 \le 1,$$
$$\varepsilon x_{j-1} \le x_j \le 1-\varepsilon x_{j-1},\ j=1,2,3,\cdots,d$$

这个有界多面体是d－维立方体的一个摄动，当$\varepsilon \to 0$时，它趋于d－维立方体。



为了把上述有界多面体化为标准形式，引进d个松弛变量：

$$s_1, s_2, \cdots, s_d$$

和d个剩余变量：

$$r_1, r_2, \cdots, r_d$$

因此，有$m=2d$个方程，$n=3d$个变量，最大化x_d，就构造出如下的线性规划问题：

对于$0<\varepsilon<\frac{1}{2}$，在$d$维空间中构造$2d$个约束方程，$3d$个变量的线性规划问题：

$$
\begin{aligned}
&\min -x_d \\
&x_1 - r_1 = \varepsilon \\
&x_1 + s_1 = 1 \\
&x_j - \varepsilon x_{j-1} - r_j = 0 \\
&x_j + \varepsilon x_{j-1} + s_j = 1 \quad j = 2,3,\cdots,d \\
&x_j, r_j, s_j \geq 0, \quad j = 1,2,\cdots,d
\end{aligned}
\qquad \text{(LP)}
$$

定理 5.1 对于每个 $d>1$，存在线性规划问题，它有 $2d$ 个约束方程，$3d$ 个变量，并且它的系数的绝对值为不超过 4 的整数。当用单纯形算法解这个线性规划问题时，其迭代步数可以为 2^d-1 步。

对单纯形算法的进一步讨论：

1．单纯形算法的变形可以改变选入或退出规则（转轴规则）以避免经过每一个顶点。但是，对于不同的变形的算法，可以找到不同的反例。这使得我们相信单纯形算法及其变形都不是多项式算法。

2．这种怀的例子在真实问题中很少发生。过去几十年的观察表明，对于中等规模的实际问题，单纯形算法需要 $4m$ 至 $6m$ 的迭代步数来完成两阶段的计算。据推测，当 n 相对于 m 来说较大时，迭代次数预计为 αm，其中

$$
e^{\alpha} < \log_2\left(2 + \frac{n}{m}\right)
$$

类似的结果已经用人工产生的概率分布的蒙特卡洛实验给予了证实。因此，单纯形算法可以期望的计算量为 $O(m^2 n)$。

二、组合优化问题原始－对偶算法的计算复杂度

1. 最短路问题的原始－对偶算法——Dijkstra 标号算法实现

多项式算法，计算复杂度为 $O(n^2)$。

2. 最短路问题的 Floyd-Washall 算法

不是原始－对偶算法，是多项式算法，计算复杂度为 $O(n^3)$。

3. 最大流问题的原始－对偶算法——Ford-Fulkerson 标号算法实现

流网络 $N=(s,t,V,A,b)$ 的 Ford-Fulkerson 标号算法可能永远不停止。设所有弧的容量为整数，那么必有限步终止：

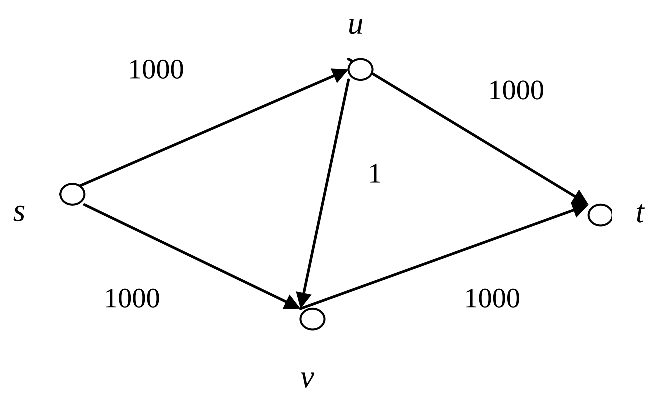
(1) 每次迭代所需要的计算复杂度为 $O(|A|)$：需要对节点进行检查和标号，$N=(s,t,V,A,b)$ 的每一条弧 (u,v) 最多检查两次，一次检查 v，另一次检查 u，因此一次标号所需要的算术运算步数为 $O(|A|)$；另一方面，标号返回需 $O(p)$ 步完成，其中 p 是发现增广路的长度(弧的条数)。由于增广路上的节点不会重复出现，所以 $p\le|V|$，所以每次迭代所需要的计算量为 $O(|V|+|A|)=O(|A|)$。

(2) 算法的迭代步数 $O(S)$，其中 S 为流的增广次数：由于所有弧的容量为整数以及算法每次迭代所得到的流也是整数，所以每次迭代流的值至少增加 1。所以，若最大流的值为 v，则 $S\le v$。不能用问题的解值去估计算法的计算复杂度，而必须用例子的输入来表示。因此，我们用 $\sum_{(x,y)\in A} b(x,y)$ 来代替 v，这是因为 $v\le\sum_{(x,y)\in A} b(x,y)$。

(3) 整个算法的计算复杂度为 $O\left(\left(\sum_{(x,y)\in A} b(x,y)\right)\cdot|A|\right)$。

因此 Ford-Fulkerson 标号算法不是多项式算法，是拟（伪）多项式算法。因为算法的迭代步数 S 可能会达到最大流的值 v，而最大流的值 v 可以是例子输入的指数倍。

例如：



最大流值为 2000。从 0 流开始，应用标号算法：

第 1 次迭代，得到增广路(s,u,v,t)，流增加 1，流值变为 1；

第 2 次迭代，得到增广路(s,v,u,t)，流增加 1，流值变为 2；

第 3 次迭代，得到增广路(s,u,v,t)，流增加 1，流值变为 3；

第 4 次迭代，得到增广路(s,v,u,t)，流增加 1，流值变为 4；

:

:

第 2000 次迭代，得到增广路(s,v,u,t)，流增加 1，流值变为 2000；

算法经过 2000 次迭代后终止。若将 1000 改为M，则 Ford-Fulkerson 标号算法的迭代步数为$2M$。所以，在最坏情况下，Ford-Fulkerson 标号算法所需要的迭代步数是指数的。

4. 考虑 Hitchcock 问题的$\alpha\beta$算法的计算复杂度。

三、其它一些常见组合优化问题的有效算法

- 最小生成树问题算法
- 匹配问题的算法
- **……**