运筹学通论I

胡晓东

应用数学研究所

中国科学院数学与系统科学研究院

http://www.amt.ac.cn/member/huxiaodong/index.html

5. 组合优化 - 算法设计技巧

精确算法

分而治之 (搜索、排大小序、旅行商)
动态规划 (最短路、三角剖分、背包)
分支定界 (整数线性规划、旅行商、工件排序)

近似算法或启发式算法

贪婪策略 (最小生成树、最大可满足、背包、
顶点覆盖、独立集、旅行商)
局部搜索(最大匹配、旅行商、最大割)
序贯法 (工件排序、装箱、顶点着色)
整数规划法 (顶点覆盖、最大可满足、最大割)
随机方法 (最小割、最大可满足、顶点覆盖)
在线算法 (页面调度、k-服务器、工件排序、装箱)
不可近似 (最大团、背包、旅行商、装箱、连通控制集等)

5.7 整数规划方法

整数规划是组合优化领域的一个非常重要的研究方向。一般来讲，绝大多数组合优化问题都可以比较容易地表述为等价的整数规划问题，有时候是线性的，有时候是非线性的。这样借助一些商业软件，就可以求解这些组合优化问题了。基于整数规划的求解组合优化问题的方法包括如下几个步骤:

- 将所考虑的组合优化问题转化为一个等价的整数规划问题
- 将整数约束条件松弛，得到一个容易求解的优化问题
- 用有效算法（在多项式时间内）求解松弛问题
- 将所求得的最优（非整数）解进行适当的舍入，得到原问题的一个可行（整数）解

5.7 最小权顶点覆盖

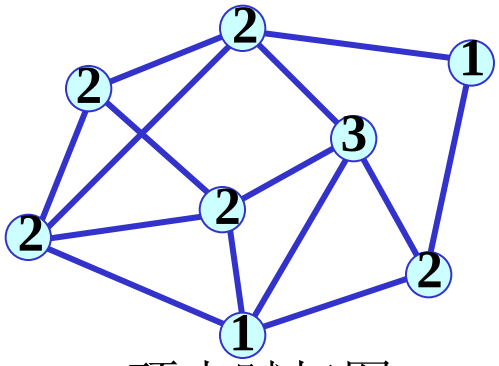
我们首先考虑最小权顶点覆盖问题。它是最小顶点覆盖问题的一般形式，而后者是一个著名的 **NP-难**问题。

问　题: **最小权顶点覆盖**

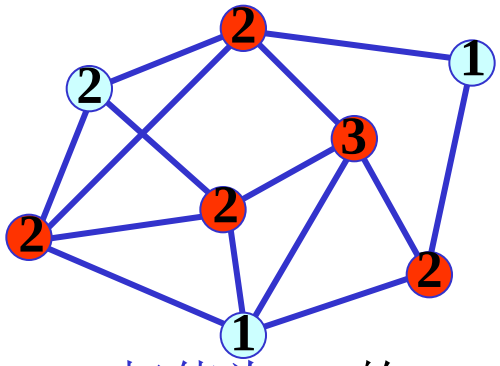
实　例: 图 $G=(V, E)$ 及顶点 $v_i \in V$ 的权值 w_i 。

可行解: 顶点覆盖 $C \subseteq V$。

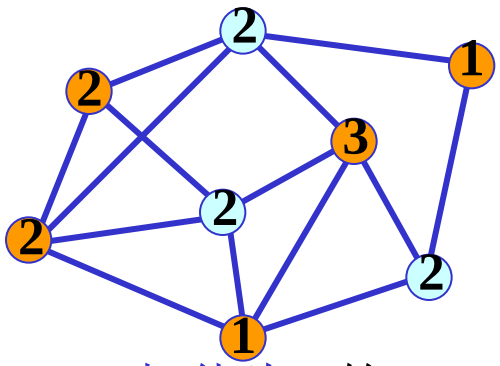
目　标: 最小化覆盖 C 中的顶点的权值之和 $\|C\|=\sum_{v_i \in C} w_i$



顶点赋权图



权值为 11 的
顶点覆盖



权值为 9的
顶点覆盖

5.7 最小权顶点覆盖（续一）

在前面我们已经证明了，基于最大匹配的算法是求解最小顶点覆盖问题的一个**2**-近似算法，然而它无法处理该问题的一般情形。我们下面就说明如何用整数规划方法求解最小权顶点覆盖问题。为此，我们先将该问题化成一个整数（线性）规划问题。

(ILP) $\mathbf{Min} \sum_{v_i \in V} w_i x_i$

$\mathbf{s.t.}\ \ x_i + x_j \geq 1, (v_i, v_j) \in E,$

$x_i = 0$ 或者 $1,\ v_i \in V.$

(LP) $\mathbf{Min} \sum_{v_i \in V} w_i x_i$

$\mathbf{s.t.}\ \ x_i + x_j \geq 1, (v_i, v_j) \in E$

如果将上面（左侧）的整数约束用简单的约束 $x_i \geq 0$， 对所有 $v_i \in V$ 替换掉，我们就可以得到上面（右侧）的线性规划。

5.7 最小权顶点覆盖（续二）

应用求解线性规划的算法，我们即可得到上述松弛线性规划问题的一个最优解 $x^*_{LP}(G)$。注意，$x^*_{LP}(G)$ 的某些分量可能不是整数，因而它不一定是整数规划的一个可行解；但是它的目标函数值不会超过整数规划问题的最优解的目标函数值。

要得到一个整数规划的可行解，一个非常简单的方法就是将 $x^*_{LP}(G)$ 中足够大的分量都取整为**1**，而其他分量取整为**0**；亦即，如果一个顶点在 $x^*_{LP}(G)$ 中相应的分量至少是 **1/2**，则将该顶点包括在顶点覆盖中。

注意，求解最小权顶点覆盖问题的这一方案（可能）需要求解带有很多约束的一个线性规划问题；实际上，松弛线性规划问题的约束个数等于给定图的边的条数，因而这一方案可能比较费时（尽管仍然是多项式时间可完成的）。

5.7 最小权顶点覆盖（续三）

定理 1 任给最小权顶点覆盖问题的一个实例，基于取整方法的整数规划算法可以找的它的顶点覆盖，其权值不超过最优覆盖的权值的 **2** 倍。

证明. 设 C 为算法输出的一个解，$c_{DR}(G)$ 是 C 中所有顶点的权值之和。 反证法，假设 C 并没有覆盖边 (v_i, v_j)，则 $x_i^{LP}(G)$ 和 $x_j^{LP}(G)$ 一定都小于 **1/2**，这与 $x^{LP}(G)$ 是线性规划的可行解矛盾，因为它不满足约束条件 $x_i + x_j \geq 1$。

现另 $c_{opt}(G)$ 为整数规划的目标函数的最优值，$c_{opt}^{LP}(G)$ 为线性规划的目标函数的最优值。 则有 $c_{opt}(G) \geq c_{opt}^{LP}(G)$。因而

$$c_{DR}(G) = \sum_{v_i \in C} w_i \leq 2\sum_{v_i \in C} w_i x_i^{LP} \leq 2\sum_{v_i \in V} w_i x_i^{LP} = 2c_{opt}^{LP}(G) \leq 2c_{opt}(G)$$

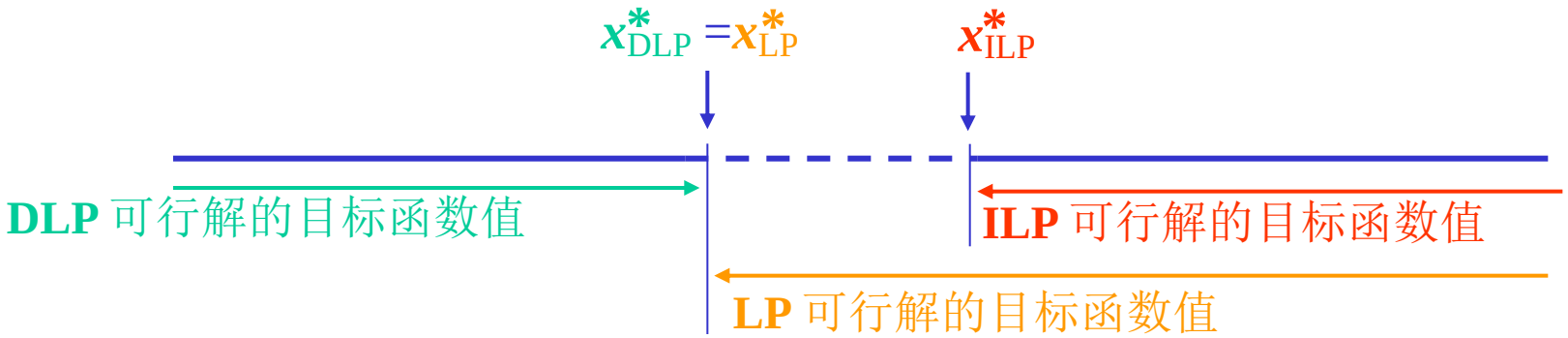
5.7 最小权顶点覆盖（续四）

我们下面解释应用整数规划求解组合优化问题的原始对偶技巧。它不是求松弛线性规划的最优解，而是求松弛线性规划的对偶问题的可行解。

回忆一下，给定一个（求最小值的）线性规划问题 LP，它的对偶问题 DLP 是一个求最大值的线性优化问题，其最优解 x^*_{DLP}（若存在）的目标函数值与线性规划问题 LP 的最优解 x^*_{LP} 的目标函数值相等。

因而，当我们将一个组合优化问题表述为求最小值的整数线性规划问题 ILP 时，其松弛就是一个线性规划问题 LP；对偶线性规划问题 DLP 的任意一个可行解的目标函数值都不会超过线性规划问题ILP 最优解的目标函数值， 而这个值又不会超过整数线性规划问题 ILP 的任意一个可行解的目标函数值；因此，当我们估计整数线性规划问题 ILP 的一个近似解的性能时，它可以做为该问题最优值的一个下界。

5.7 最小权顶点覆盖（续五）



给定最小权顶点覆盖问题的一个实例， 其松弛线性规划问题 LP 的对偶问题是如下求最大值线性规划问题：

$$\text{(DLP) Max} \sum_{(v_i,v_j)\in E} y_{ij}$$

$$\text{s.t.} \sum_{(v_i,v_j)\in E} y_{ij} \le w_i,\ v_i \in V;$$

$$y_{ij} \ge 0,\ (v_i, v_j)\in E$$

$$\text{(LP) Min} \sum_{v_i\in V} w_i x_i$$

$$\text{s.t.}\quad x_i + x_j \ge 1,\ (v_i, v_j)\in E$$

5.7 最小权顶点覆盖（续六）

注意，空集是原始整数线性规划问题 ILP 和其松弛线性规划问题 LP 的一个可行解，而与所有分量 y_{ij} 都为 **0** 所对应的解是线性规划问题 LP 的对偶规划问题 DLP 的一个可行解。因而，原始对偶算法可以从这样一对儿解开始迭代，$(y_{ij} = 0)$ 和 $(x_i = 0)$，通过寻找更好的对偶解来构造一个可行原始解。

ALGORITHM 11.2 *Primal-Dual Algorithm*

$y_{ij} := 0$, for $(v_i, v_j) \in E$,
$C := \emptyset$.
while C is not a cover **do**
    pick $(v_i, v_j) \in E$ such that $v_i, v_j \notin C$,
    increase y_{ij} until either i-th or j-th constraint of DLP becomes tight;
    if the i-th constraint is tight **then** $C := C \cup \{v_i\}$;
    else $C := C \cup \{v_j\}$.
end-while
return C.

5.7 最小权顶点覆盖（续七）

定理 2 给定最小权顶点覆盖问题的一个实例，上述算法可以输出一个顶点覆盖 C，其权值最多不超过最优覆盖权值的 2 倍。

证明 根据算法的规则可知，C 一定是一个覆盖。将 C 的权值记为 $c_{PD}(G)$。为了估计 $c_{PD}(G)$ 的上界，我们可以注意到，对每一个顶点 $v_i \in C$，都有 $w_i = \sum_{(v_i, v_j) \in E} y_{ij}$。因而可得

$$c_{PD}(G) = \sum_{v_i \in C} w_i = \sum_{v_i \in C} \sum_{(v_i, v_j) \in E} y_{ij}$$

$$\leq \sum_{v_i \in V} \sum_{(v_i, v_j) \in E} y_{ij} = 2 \sum_{(v_i, v_j) \in E} y_{ij}.$$

因为，DLP 的对偶可行解的目标函数肯定不会超过LP最优原始解的目标函数值，故有

$\sum_{v_i \in V} \sum_{(v_i, v_j) \in E} y_{ij} \leq c_{opt}(G)$，因此有 $c_{PD}(G) \leq 2c_{opt}(G)$。

5.7 最大权可满足

我们现在考虑**最大权可满足问题**：给定一组子句，其中每一个子句都赋有一个权值，目标是布尔分量赋值使得它们满足的子句的权值之和最大。显然，该问题是前面讨论过的最大可满足问题的一般形式。

问　题： **最大权可满足问题**

实　例：n 个布尔变量 $X=\{x_1, \ldots, x_n\}$，
m 个合取子句$C=\{c_1, \ldots, c_m\}$，每个子句c_i 赋有权值w_i。

可行解：n 个布尔变量的赋值 $f: X \rightarrow \{真, 伪\}$。

目　标：C 中被满足的子句的权值之和最大。

前面已经证明， 对该问题的无权形式（即所有权值w_i 都相等），一个简单的贪婪算法可以给出一种布尔变量的赋值，它满足的子句个数不会比最优赋值满足的子句的个数的**1/2** 要少。

5.7 最大权可满足（续一）

我们来说明，如何用整数线性规划方法处理该问题的一般形式（即所有权值 w_j 不一定都相等）。为此，我们先将该问题表述为一个整数线性规划问题。用 c_j^+ 表示变量 x_i 在子句c_j 中以 x_i 出现的指标集合， 用 c_j^- 表示变量 x_i 在子句 c_j 中以 $\bar{x}_i$ 的非出现的指标集合。

(ILP) $\mathbf{Max} \sum_{c_j \in C} w_j z_j$

$$\text{s.t.} \quad \sum_{i \in c_j^+} y_i + \sum_{i \in c_j^-} (1 - y_i) \geq z_j, \text{ 所有 } c_j \in C;$$

$$y_i = 0 \text{ 或 } 1, \ 1 \leq i \leq n,$$

$$z_j = 0 \text{ 或 } 1, \ 1 \leq j \leq m.$$

练习 验证布尔变量 X 的赋值 f 与整数线性规划问题 ILP 之间存在这样一个一一对应：$y_i=1$ 当且仅当变量 x_i 赋值为真且 $z_j=1$ 当且仅当子句 c_j 被满足。

5.7 最大权可满足（续二）

现在将整数约束替换为如下约束

$$0 \le y_i \le 1,\ 1 \le i \le n,$$

$$0 \le z_j \le 1,\ 1 \le j \le m,$$

即可得到一个线性规划问题 LP，另设

$$y^*=(y_1^*, y_2^*, \ldots, y_n^*),\ z^*=(z_1^*, z_2^*, \ldots, z_m^*),$$

是该线性规划问题 LP 的一个最优解。

我们可以应用整数规划方法设计求解最大权可满足问题的如下算法：给定问题的一个实例 **I**，求出其松弛线性规划问题 LP 的一个最优解 (y^*, z^*)。然后，对每一个 $i = 1, 2, \ldots, n$，计算概率 $p_i = g_i(y^*)$，并以此概率给变量 x_i，赋值“真”，其中函数 g 可以以多种方式定义。

5.7 最大权可满足（续三）

注意，如果可以在多项式时间内计算出函数 g 的值，那么上述算法就是一个多项式时间的算法。此外，该算法输出的解的性能也取决于函数 g 的选取。最简单的一个选取方法是，令 $g(y_i) := y_i^*$，$i = 1, 2, \ldots, n$，亦即，以概率 y_i^* 独立地将每一个变量 x_i 赋值“真”。

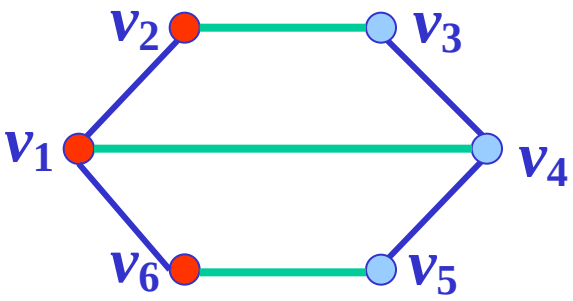
我们可以证明：如果给定的最大权可满足问题的实例中的每一个子句最多含有 k 个“字”时，那么算法输出的解的权值与最优解的权值之比的期望值至少是 $3/4$ 当 $k \leq 2$，或者 $1 - 1/e$ 当 $k \geq 3$。

5.7 最大割

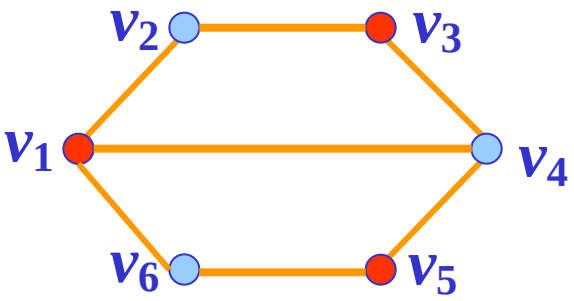
我们前面讲述了，如何先通过松弛整数线性规划问题 ILP 的整数约束条件，再对松弛线性规划问题 LP 的最优解进行取整，来找的最小权顶点覆盖问题的一个好的近似解。事实上，这一技巧还可以应用于处理其他许多组合优化问题。

下面我们针对若干特殊的组合优化问题说明，当我们可以找到适当的松弛方法时，结合使用松弛和取整技巧可以取得很好的效果。为此，我们考虑最大割问题，它是**最大 *k*-割问题**的一个特殊情形（$k = 2$）；我们在前面讨论了如何用局部搜索技巧处理这个问题。

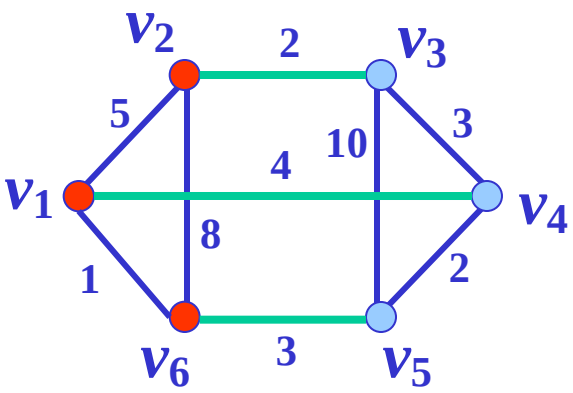
5.7 最大割（续一）



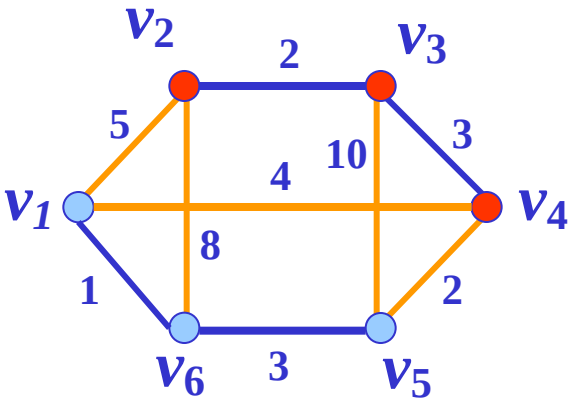
非最优 2-割



最优 2-割



权值为9 的 2-割



权值为 29 的 2-割

5.7 最大割（续二）

我们可以用如下方法，将最大割问题表示为一个整数二次规划问题（IQP）。首先，将每一对顶点 $v_i, v_j \in V$ 都赋值 $w_{ij} := w(v_i, v_j)$ 当 $(v_i, v_j) \in E$，或 $w_{ij} := 0$ 当 $(v_i, v_j) \notin E$。这样即有

$$\text{(IQP)} \quad \text{Max} \ \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{j-1} w_{ij}(1 - x_i x_j)$$

$$\text{s.t.} \qquad x_i = 1 \text{ 或 } -1, \text{ 所有 } v_i \in V.$$

注意，对变量 x_i 的赋值相当将对顶点集合 V 划分为两个不相交的子集 V_1 和 V_2，由此产出的割的值等于整数二次规划问题 IQP 的目标函数值。

事实上，考虑两个相邻的顶点 v_i 和 v_j。如果或者 $v_i, v_j \in V_1$ 或者 $v_i, v_j \in V_2$，亦即 $x_i = x_j$，那么 $1 - x_i x_j = 0$；另一方面，如果 v_i 和 v_j 不属于 V_1 或 V_2 中的同一个集合，亦即 $x_i \neq x_j$，那么有 $1 - x_i x_j = 2$。

5.7 最大割（续三）

注意，我们可以将每一个变量 x_i 视为**1-**维空间上单位模的向量。因而我们可以这样对整数二次规划问题 IQP 松弛：将每一个变量 x_i 替换为一个**2-**维空间中单位模向量 y_i，亦即属于**2-**维单位球面 S_2 的向量。 据此整数二次规划问题 IQP 就可以化为一个二次规划问题：

$$\text{(QP)}\quad \text{Max}\ \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{j-1} w_{ij}(1 - x_i x_j)$$

$$\text{s.t.}\qquad x_i \in S_2, \text{所有 } v_i \in V.$$

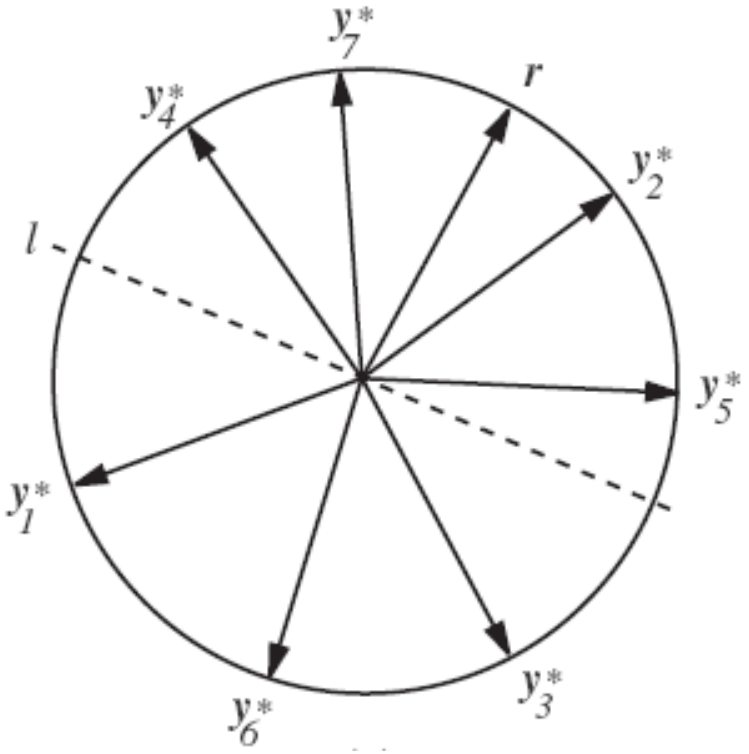
其中 $x_i\, x_j$ 表示向量 x_i 和 x_j 的内积。显然，给定一个可行解 $x = (x_1, x_2, \ldots, x_n)$，我们可以得到二次规划问题 QP 的一个可行解 $X = (x_1, x_2, \ldots, x_n)$，其中 $x_i = (x_i, 0)$，且两个解 x 和 X 的目标函数值是一样的。因而二次规划问题 QP 就是整数二次规划问题 IQP 的一个松弛问题。

5.7 最大割（续四）

我们求解最大割问题的基本思想如下： 首先，求出二次规划问题 QP 的一个最优解 $X^* = (x_1^*, x_2^*, \ldots, x_n^*)$，在2-维球面上选取一个向量 $r \in S_2$，然后生成原问题的一个解 (V_1, V_2) 使得一个顶点属于 V_1 或者 V_2 取决于相应的向量 x_i^* 是在 r 的垂直线的上方还是下方。

ALGORITHM 11.4 *Randomized Algorithm*

find an optimal solution $(\mathrm{x}_1^*, \mathrm{x}_2^*, \cdots, \mathrm{x}_n^*)$ of QP
randomly choose a vector $\mathbf{r} \in S_2$.
$V_1 := \{v_i \in V \mid \mathrm{x}_i^* \cdot \mathbf{r} \geq 0\}$;
$V_2 := V \setminus V_1$.
return V_1 and V_2.



5.7 最大割（续五）

注意，上述算法的性能取决于我们如何选取向量 ***r***。在运行算法的过程中，我们可以任意取一个向量 ***r***。这样的算法可视为一个随机算法，即使对同一个实例，它所输出的解也可能不一样。（我们在后面会对随机算法做一个简单的介绍）

定理 3 给定最大割问题的一个实例 **I**，假定向量 ***r*** 是在**2-**维单位球面 S_2 上随机选取的，则上述算法输出的割的期望值最多不超过最优割的值的 **0.8785** 倍。

5.7 最大割（续六）

最后，我们讨论一下该算法的时间复杂度。显然，它是一个多项式时间算法当且仅当二次规划问题 QP 可以在多项式时间内求解。遗憾的是，人们还知道是否存在这样的求解算法。不过，为了有效地求解二次规划问题 QP， 我们可以将它做少许修改：将变量 x_i 视为 n-维单位球面 S_n 上的一个向量，而不是 2-维单位球面 S_2 上的一个向量。这个修改过的问题可以用**半定规划**的理论和技巧求解。

回忆一下，前面介绍过的局部搜索算法可以找到最大权割问题的一个 **0.5**-近似解，然而其时间复杂度可能不是多项式时间的。因此，上述算法不仅可以得到一个更好的近似解，而且所用的时间也更少（尽管算法更复杂一些）。

5.7 练习

练习. 考虑优化问题： $\min z = f_1(x_1) + f_2(x_2)$，其中

$f_1(x_1) = 20 + 5x_1$，若 $x_1 > 0$；否则 $x_1=0$，$f_1(x_1) = 0$；

$f_2(x_2) = 12 + 6x_2$，若 $x_2 > 0$；否则 $x_2=0$，$f_2(x_2) = 0$。

针对以下约束条件，分别给出相应的混合整数规划问题。

（1）或者 $x_1 \geq 0$ 或者 $x_2 \geq 0$。

（2）下面三个不等式至少有一个成立

$2x_1 + x_2 \geq 15$；

$x_1 + x_2 \geq 15$；

$x_1 + 2x_2 \geq 15$；

（3）$|x_1 + x_2| = 0$ 或者 5 或者 10。

（4）$x_1 \geq 0, x_2 \geq 0$。