一张一弛，解题之道

——“约制、放宽”方法在解题中的应用

广东省中山纪念中学 陈启峰

目录

一张一弛，解题之道.....1

——“约制、放宽”方法在解题中的应用.....1

目录.....2
【摘要】.....3
【关键字】.....3
“约制、放宽”方法的定义.....4
引言.....4
例题分析.....4
[例一]骑士.....4
【问题描述】.....4
【问题分析】.....5
【数学模型】.....5
【算法模型分析】.....5
【确定总算法和研究对象】.....5
【分析研究对象】.....6
【“约制”方法确定新任务】.....6
【寻找两个向量的规律】.....6
【“约制”方法解决新任务】.....8
【回到研究对象】.....10
【回到起点】.....11
【小结】.....11
[例二]友好的动物.....12
【问题描述】.....12
【问题分析】.....12
【数学模型】.....12
【原始算法的矛盾】.....13
【数据范围分析】.....13
【数据结构优化时的矛盾】.....13
【分析矛盾】.....14
【“放宽”方法化解矛盾】.....14
【小结】.....15
[例三]消防站.....16
【问题描述】.....16
【问题分析】.....16
【数学模型】.....16
【算法模型分析】.....16
【预处理】.....**Error! Bookmark not defined.**
【确定动态规划时的矛盾】.....17
【总结分析】.....17

【“放宽”方法转化限制】.....17
【进一步分析】.....18
【“约制”方法增添限制】.....18
【确定动态规划转移方程】.....19
【小结】.....20
总结.....20
【感谢】.....21

【摘要】

本文主要阐述了“约制、放宽”方法在解题中的应用。

第一部分解释了什么“约制、放宽”方法。第二部分论述了在解题中为什么需要“约制、放宽”方法。第三部分通过分析《骑士》、《友好的动物》和《消防站》分别介绍了“约制”方法、“放宽”方法和两者综合应用在解题中起的作用。最后作者结合自身经验谈谈在解题中如何灵活运用“约制、放宽”方法。

【关键字】

“约制”方法　“放宽”方法

过于宽松　过于繁杂

过于独立　过于严格

约制条件、限制

放宽条件、限制

正文

“约制、放宽”方法的定义

“约制”方法——添增一些约束的条件、限制，并保证在这些条件和限制下依然能找到解。

“放宽”方法——减除、放宽一些条件、限制，并保证在这些条件和限制下依然能找到解。

引言

在分析问题、设计算法的时候，我们常常会觉得条件、限制过于繁杂、严格或者过于宽松、独立以致无法下手。这时，不妨尝试“约制、放宽”方法。“约制”方法往往在我们迷茫时指出一条光明大道，“放宽”方法则常常在关系错综复杂时破除阻扰和荆棘。巧妙地运用这两种方法能使我们在解题中排除万难，直入脏腑。

例题分析

下面，本人从“约制”方法，“放宽”方法和两者的综合应用三个方面精心挑选了三个题目并作详细分析，希望这能起到抛砖引玉的作用。

[例一]骑士[1]

【问题描述】

有一骑士在一个无限大的棋盘上移动。它每次的移动都用一个整数对来描述——整数对（a,b）表示骑士能从位置（x,y）跳到位置(x+a,y+b)或者(x-a,y-b)。每个骑士有一系列的已确定的整数对，描述这骑士能进行哪些移动。我们保证每个骑士能到达的位置不在同一直线上。

当两个骑士以位置（0，0）为始点能到达的所有位置完全相同时（可能做很多次移动），我们就说这两个骑士是等价的。可以证明对于每一个骑士，都存在一个只有两个整数对的等价骑士。

[1] 选自 POI2005 的 STAGE 1 的《KNIGHTS》

任务：读入一个骑士的所有整数对，找出一个只有两个整数对的等价骑士。

【问题分析】

【数学模型】

令输入的整数对分别表示为向量(a_1,b_1),向量(a_2,b_2)……向量(a_n,b_n)，找出两个整数对——向量(c_1,d_1)和(c_2,d_2)与这 n 个向量等价,也就是

对于任意的整数序列$t_1,t_2,t_3.....t_n$，都存在两个整数e,f使得

$$\sum_{i=1}^{n} t_i \times (a_i,b_i) = e\times(c_1,d_1) + f\times(c_2,d_2)$$

并且对于任意两个整数e,f，都存在一个整数序列$t_1,t_2,t_3.....t_n$，使得

$$e\times(c_1,d_1) + f\times(c_2,d_2) = \sum_{i=1}^{n} t_i \times (a_i,b_i)$$

【算法模型分析】

看到这个题目，最容易想到的算法是枚举这两个向量和用宽度优先搜索判断是否等价。但是要寻找的这两个向量的范围是无限大的，并且棋盘也是无限大的。因此这个算法宛如大海捞针一般，极难在有限的时间内找到解。

然后尝试贪心、动态规划、图论等硬做的算法，但这些算法都在预料之中以失败告终。

最后，看来只有必经之路——数学方法才能可以解决这个问题。

【确定总算法和研究对象】

用数学方法解决等价转化等题目的方法还是不胜枚举的。例如有归纳法，有总体法，有解方程法，有数形结合。对于这个题目，我们应选取什么数学方法好呢？

注意到,如果任意的三个向量都可以与某两个向量等价,那么便可以从 n(n>2)个向量中任选三个向量出来，用与它们等价的两个向量代替它们，从而变成 n-1 个向量。不断重复上述的过程，直到只剩下两个向量为止，这时剩下的两个向量便是一个可行解。而由问题描述可以知道：**对于任意三个向量都存在两个向量与它们等价。**这便意味这种方法是可行的。

于是，就可以确定下总算法——**不断化三为二**，同时也产生了我们的研究对

象——**如何把三个向量等价替换为两个向量？**

【分析研究对象】

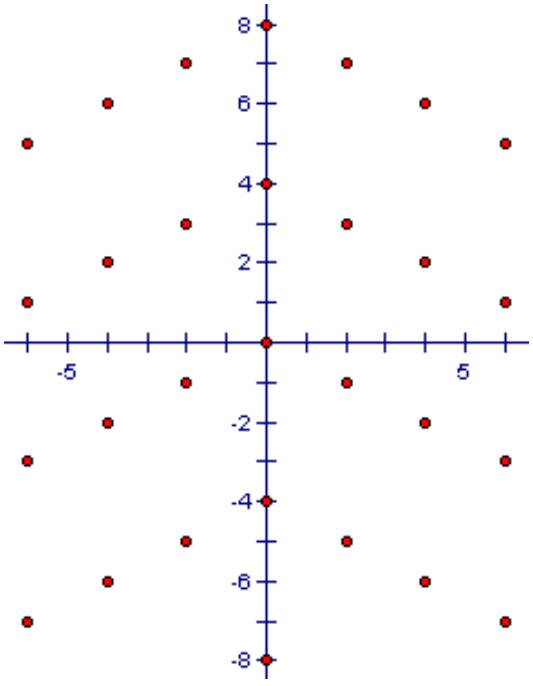
因为满足条件的解（等价的两个向量）是无限多个的，解的范围没有多少限制，而我们仅仅需要这样一个解，所以这时确定下来的研究对象虽然看起来简洁明了，可是要解决该问题的难度依然相当大。

【“约制”方法确定新任务】

既然研究对象的条件、限制还是那么宽松，因而可先且放下研究对象，而尝试做些更“细”的工作——**把两个向量等价转化为具有某些性质的两个向量——新任务**。

【寻找两个向量的规律】

为了更加容易地找到两个向量的一般性质，可以先从一些小数据着手，试着找出有用的规律。比如当两个向量分别为（2,3），（4,2）时，在直角坐标系中能被这两个向量表示的点的位置如下图：



由此我们可以观察很多重要的性质：

1、所有的点都在且只在直线 $x=2k(k\in Z)$ 和直线 $y=k(k\in Z)$ 上；

2、在 Y 轴上，所有的点都出现且只出现在 $(0,4k)(k\in Z)$ 上；

3、直线 $x=2k(k\in Z)$ 上的点都可通过平移 Y 轴而得到。

于是我们归纳出任意两个不在 Y 轴上的向量 (a_1,b_1) 和 (a_2,b_2) 能表示的点的一般性质：

① 所有的点分布在且只在直线 $x=\gcd(a_1,a_2)\times k(k\in Z)$ 和直线

$y=\gcd(b_1,b_2)\times k(k\in Z)$ 上；

②在 Y 轴上，所有的点都出现且只出现在 $(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)}k)(k\in Z)$ 上；

③直线 $x=\gcd(a_1,a_2)\times k(k\in Z)$ 上的点都可通过平移 Y 轴而得到。

性质②的证明：

1、充分性—— $(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)}k)(k\in Z)$ 位置上都有点的证明：

对于任意的 $k\in Z$，令 $t_1=\frac{a_2}{\gcd(a_1,a_2)}k$，$t_2=\frac{a_1}{\gcd(a_1,a_2)}k$，则

$$(a_1,b_1)t_1-(a_2,b_2)t_2=(0,b_1t_1-b_2t_2)=(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)}k)$$

所以 $(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)}k)(k\in Z)$ 位置上都有点。

证毕。

2、必要性——在 Y 轴上，所有的点都只出现在 $(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)}k)(k\in Z)$ 上的证明：

设在 Y 轴上能用这两个向量表示的任意一点为 $(0,y)(y\in Z)$，则必存在两个整数 t_1 和 t_2，使得

$$\begin{cases}t_1a_1+t_2a_2=0 & ①\\ t_1b_1+t_2b_2=y & ②\end{cases}$$

由①式得

$$t_1a_1=-t_2a_2\Rightarrow\frac{t_1a_1}{a_2}=-t_2$$

因为 $-t_2$ 为整数，所以

$$a_2\mid t_1a_1\Rightarrow\frac{a_2}{\gcd(a_1,a_2)}\mid t_1\frac{a_1}{\gcd(a_1,a_2)}$$

因为 $\frac{a_2}{\gcd(a_1,a_2)}$ 与 $\frac{a_1}{\gcd(a_1,a_2)}$ 互质，所以

$$\frac{a_2}{\gcd(a_1,a_2)} \mid t_1$$

令 $t_1 = \frac{a_2}{\gcd(a_1,a_2)}k(k \in Z)$，则可以推出 $t_2 = -\frac{a_1}{\gcd(a_1,a_2)}k$,所以

$$y = t_1 b_1 + t_2 b_2 = \frac{a_1 b_2 - a_2 b_1}{\gcd(a_1,a_2)}k$$

所以必定存在一个整数 k 使得 $y = \frac{a_1 b_2 - a_2 b_1}{\gcd(a_1,a_2)}k$ ，也就是说在 Y 轴上, 所有的点都只出现在 $(0, \frac{a_1 b_2 - a_2 b_1}{\gcd(a_1,a_2)}k)(k \in Z)$ 上。

证毕。

综上所述，可以得出性质②。

证毕。

【“约制”方法解决新任务】

有了上面富有价值的性质之后，摆在眼前的问题便是，如何利用性质恰当地约束转化后的两个向量？虽然约束的方式不可胜数，但是绝大数对解题并没有什么任何作用。此时就要取其精华了。

注意到在每一条有点的竖直直线上，一个显眼的性质是，上面的点都是相隔定长出现的。这催人直觉上觉得，转化后的一个向量可以是一个竖直向量。这种感觉正确吗？正确！更重要的是，正因为这种“约制”方法，使解题的思路开始走出模糊、步入明晰。

下面给出转化后其中一个向量是竖直向量的转化方法：

设须要等价转化的两个向量分别为 (a_1,b_1) 和 (a_2,b_2) 。

1、如果 $a_1 = 0$ 或者 $a_2 = 0$ ，则可令转化后的两个向量为 (a_1,b_1) 和 (a_2,b_2) ；

2、如果 $a_1 \neq 0$ 或者 $a_2 \neq 0$ ，综合性质②和性质③，可以大胆地约制其中一个向量为 $(0, \frac{a_1 b_2 - a_2 b_1}{\gcd(a_1,a_2)})$, 因为每一条直线 $x = \gcd(a_1, a_2) \times k(k \in Z)$ 上的点都是相隔 $|\frac{a_1 b_2 - a_2 b_1}{\gcd(a_1,a_2)}|$ 个单位出现的。

此时，另一个向量应该是什么呢？由性质③可以推测，如果能找到一个向量，只用这个向量能且只能表示每一条直线

$x=\gcd(a_1,a_2)\times k(k\in Z)$ 上一个骑士能到达的点，那么问题就解决了。要找的这个向量的横坐标无疑可以是 $\gcd(a_1,a_2)$。为了表示出这个横坐标,顺理成章地想到用扩展的欧几里德算法快速地找出两个整数 p 和 q 使得

$$a_1p+a_2q=\gcd(a_1,a_2)$$

则向量 $(a_1,b_1)p+(a_2,b_2)q=(\gcd(a_1,a_2),b_1p+b_2q)$ 就是我们要找的另一个向量了。

下面证明 $(\gcd(a_1,a_2),b_1p+b_2q)$ 和 $(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})$ 等价于 (a_1,b_1) 和 (a_2,b_2)。

证明：

1、对于任意的整数 t_1 和 t_2，总存在两个整数 T_1 和 T_2 使得

$$(a_1,b_1)t_1+(a_2,b_2)t_2=(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2$$

的证明：

对于任意的整数 t_1 和 t_2，令整数 $T_1=\frac{a_1t_1+a_2t_2}{\gcd(a_1,a_2)}$。

因为骑士通过向量 (a_1,b_1) 和 (a_2,b_2) 能够到达位置 $(a_1,b_1)t_1+(a_2,b_2)t_2=(a_1t_1+a_2t_2,b_1t_2+b_2t_2)$ 和位置 $(\gcd(a_1,a_2),b_1p+b_2q)T_1=(a_1t_1+a_2t_2,b_1pT_1+b_2qT_1)$，所以骑士通过向量 (a_1,b_1) 和 (a_2,b_2) 也能到达位置 $(a_1t_1+a_2t_2,b_1t_2+b_2t_2)-(a_1t_1+a_2t_2,b_1pT_1+b_2qT_1)=(0,b_1t_2+b_2t_2-b_1pT_1-b_2qT_1)$。

又由刚才已经证明过了的性质②可以得出必有一个整数 T_2 使得

$$(0,b_1t_2+b_2t_2-b_1pT_1-b_2qT_1)=(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2。$$

所以 $(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2=$

$(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,b_1t_2+b_2t_2-b_1pT_1-b_2qT_1)=$

$(\gcd(a_1,a_2)T_1,b_1t_2+b_2t_2)=(a_1t_1+a_2t_2,b_1t_2+b_2t_2)=$

$=(a_1,b_1)t_1+(a_2,b_2)t_2$

证毕。

2、对于任意的整数 T_1 和 T_2，总存在两个整数 t_1 和 t_2 使得

$(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2=(a_1,b_1)t_1+(a_2,b_2)t_2$

的证明：

对于任意的整数 T_1 和 T_2，

$(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2=$

$(\gcd(a_1,a_2)T_1,b_1(T_1p-\frac{a_2T_2}{\gcd(a_1,a_2)})+b_2(T_1q+\frac{a_1T_2}{\gcd(a_1,a_2)}))$。

令 $t_1=T_1p-\frac{a_2T_2}{\gcd(a_1,a_2)},t_2=T_1q+\frac{a_1T_2}{\gcd(a_1,a_2)}$，则 t_1 和 t_2 恰好使得

$a_1t_1+a_2t_2=T_1a_1p+T_1a_2q-\frac{a_1a_2T_2}{\gcd(a_1,a_2)}+\frac{a_1a_2T_2}{\gcd(a_1,a_2)}=T_1(a_1p+a_2q)$

$=\gcd(a_1,a_2)T_1$。所以

$(\gcd(a_1,a_2),b_1p+b_2q)T_1+(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})T_2=(a_1,b_1)t_1+(a_2,b_2)t_2$

证毕。

证毕。

至此，“约制”方法已经很好地解决了新任务。下一步，就要利用新任务的结果来解决研究对象了。

【回到研究对象】

新任务被解决了，这为解决研究对象铺开了一条光明大道。相信聪明的读者已经知道下一步该怎么做了。设须要等价转化的三个向量分别为(a_1,b_1)、(a_2,b_2)和(a_3,b_3)，则把三个向量等价替换为两个向量的步骤如下：

步骤 1：先将(a_1,b_1)和(a_2,b_2)按上述方法等价转化为(a_1',b_1')和$(0,b_2')$；

步骤 2：再将(a_1',b_1')和(a_3,b_3)按上述方法等价转化为(a_1'',b_1'')和$(0,b_3')$；

步骤 3：最后，此时三个向量已等价转化为(a_1'',b_1'')、$(0,b_2')$和$(0,b_3')$。

如果$b_2'=0$或者$b_3'=0$，就可以把其中的向量$(0,0)$删去，这便剩下两个向量；否则，可以用向量$(0,\gcd(b_2',b_3'))$代替$(0,b_2')$和$(0,b_3')$，同样此时也只剩下两个向量。

当$b_2'\neq 0$并且$b_3'\neq 0$时，向量$(0,\gcd(b_2',b_3'))$之所以可以代替$(0,b_2')$和$(0,b_3')$是因为向量$(0,y)$能表示$(0,b_2')$和$(0,b_3')$的充分必要条件是$\gcd(b_2',b_3')\mid y$。

【回到起点】

经过一番深入研究探索之后，最终详细的解题方法已是不言而喻了。只须每次从剩下的向量中任取三个向量，按照上述的步骤将它们等价替换为两个向量。不断重复该过程直至只剩下两个向量，此时剩下的两个向量即为所求。

复杂度分析：每次把三个向量等价转为两个向量的时间复杂度为 O(logn)（求 gcd 所用的时间），总的时间复杂度为 O(nlogn)。空间复杂度为 O(1)。编程复杂度很低。

【小结】

“约制”方法的本质是为了缩小研究范围，增加约束性的条件限制，而保证能在该条件下找到解。在本题中，“约制”方法可谓是表现得淋漓尽致：在确定算法时，约制必须采取化三为二的算法；在把三个向量等价转化两个向量时，约制转化的方式必须为先将其中两个向量转化成特殊的两个向量；在确定特殊的两个向量时，约制特殊的含义为其中一个为竖直向量；在把两个非竖直向量转化为特殊的两个向量时，约制它们为$(\gcd(a_1,a_2),b_1p+b_2q)$和$(0,\frac{a_1b_2-a_2b_1}{\gcd(a_1,a_2)})$，在证明中多次约制某些变量为常量……。

[例二]友好的动物[1]

【问题描述】

W 星球是一个和地球一样气候适宜、物种聚集的星球。经过多年的研究，外星生物学家们已经发现了数万种生物，而且这个数字还在不断增大。

W 星球上的生物很有趣，有些生物之间很友好，朝夕相伴，形影不离；但有些却很敌对，一见面就难免发生战斗。为了能够更好地了解它们之间的友好程度，外星生物学家希望进行一些量化的计算。他们发现，两种生物之间的友好程度和它们的 $k(k \le 5)$ 种属性有关，暂且将它们编号为属性 1、属性 2、……、属性 k，这些属性都是可以进行量化的。外星生物学家研究发现，如果前 k-1 种属性的差别越大，这两种生物就越友好；但是属性 k 与众不同，这种属性差别越小的两种生物越友好。

因此他们猜想是不是可以用这样一个公式量化两种生物之间的友好程度：

$Friendliness = (\sum_{i=1}^{k-1} C_i \times$属性i的差别$) - C_k \times$属性k的差别，其中 C_i 是非负常数。如果知道了每种生物的各种属性，利用上述公式就很容易算出它们之间的友好程度了。现在，外星生物学家们想问一问：在目前发现的这些生物当中，关系最友好的那对生物是哪一对呢？它们之间的友好程度是多少？

【问题分析】

【数学模型】

首先，把题目的任务转化为简明的数学语言：

令 $A_{i,j} = C_i \times$第i个动物的属性j,求出两种动物的编号：

$$1<=a<b<=n$$

使得目标函数

$$Friendliness(a,b) = (\sum_{i=1}^{k-1} |A_{a,i} - A_{b,i}|) - |A_{a,k} - A_{b,k}| \qquad ①$$

最大化。

[1] 选自 NOI 2005 冬令营的《友好的动物》

【原始算法的矛盾】

显然最原始的算法是枚举所有可能的 a 和 b，求出最大的 $Friendliness$。但这样做程序的时间复杂度是 O（$n^2 \times k$），显然无法通过所有的数据。

【数据范围分析】

一个很显眼的条件 k≤5，便令人不禁会想到时间复杂度为 O（$n \times 2^k$）、O（$n \times k!$），O（$n\log^k n$）之类的指数级算法,并促使人向该方面的算法思考。

【数据结构优化时的矛盾】

考虑从数据结构上优化原始算法。
首先，把函数①中的绝对值去掉，目标函数变成

$$Friendliness(a,b)=(\sum_{i=1}^{K-1}\pm A_{a,i}\mp A_{b,i})-(\pm A_{a,k}\mp A_{b,k}) \qquad ②$$

其中小括号内前面是加号的数不小于前面是减号的数

这样就有 2^k 种情况需要考察。

然后，对于每一种动物 a,考察 a 的属性在函数②中前面的符号的情况，用 0 到 2^k-1 一个二进制 BN 表示正负情况：如果 BN 的第 i 位为 1，则 $A_{a,i}$ 前面的符号为正，令 SIGN(BN,i)=1,否则为负，令 SIGN(BN,i)=－1。定义

$$F_{a,BN}=(\sum_{i=1}^{k-1}A_{a,i}\times SIGN(BN,i))-A_{a,k}\times SIGN(BN,k)$$

表示正负情况为 BN 动物 a 的值总和。
则 Friendliness 的最大值为

$$\max\{F_{a,BN}+F_{b,2^k-1-BN}\} \qquad ③$$

其中 a≠b,并且对于任意的 i∈[1,k],都有

$$A_{a,i}\times SIGN(BN,i)+A_{b,i}\times SIGN(2^k-1-BN,i)\geq 0 \qquad ④$$

实现时，只要先枚举 a 和 BN，然后查找满足性质④并且 $F_{b,2^k-1-BN}$ 值最大的 b(a<b)。这时的查找依懒于 2^k 棵第 k 层以 max{F}为元素、第 1 到第 k-1 层以线段树为元素的线段树。

复杂度分析：每次查找和插入的时间复杂度都为 O（$\log^k$ n），总的时间复杂度为 O ($n\times 2^k \times \log^k n$),空间复杂度为 O($n \log^{k-1} n$)，编程复杂度高。

此时矛盾重重，时间复杂度依然很高，空间复杂度太大，编程复杂度令人难以忍受。

【分析矛盾】

不难发现，产生这些矛盾的根本原因是性质④的要求太苛刻了。因此，此时须要***放宽限制。***

【“放宽”方法化解矛盾】

正所谓退一步海阔天空，试着放宽性质④的要求。

尝试将苛刻的要求去掉，但这样编出来的程序和用原始算法编出来的程序的运行结果是不正确的。因为这样做实在太武断了。

先来对性质④做个详细分析：注意到一个特殊性：$|A_{a,k}-A_{b,k}|$越大，Friendliness 的值越小，其余的$|A_{a,i}-A_{b,i}|$（i<k）越大，Friendliness 的值越小。因此，试着把性质④**放宽**为只要保证

$$A_{a,k}\times SIGN(BN,i)+A_{b,k}\times SIGN(2^k-1-BN,i)\geq 0 \qquad ⑤$$

结果，这样编出来的程序和用原始算法编出来的程序的运行结果是完全一样的。那么，这做法是否一定可以保证正确呢？下面的证明肯定了这一做法的正确性。

证明：

[定理]对于任意的 A,B∈R,都有|A-B|≥A-B，|A-B|≥B-A。

[推论]对于任意的 A,B∈[1,n]，BN∈[0,2^k-1]都有

$$(\sum_{i=1}^{K-1}|A_{a,i}-A_{b,i}|)-|A_{a,k}-A_{b,k}|\geq$$ **满足性质⑤的** $F_{a,BN}+F_{b,2^k-1-BN}$

也就是

满足性质④的 $F_{a,BN1}+F_{b,2^k-1-BN1}\geq$ **满足性质⑤的** $F_{a,BN2}+F_{b,2^k-1-BN2}$

设满足性质④的(a,b,BN)所构成的集合为S1,满足性质⑤的所构成的集合为

S2，MAX1= $\max\{F_{a,BN}+F_{b,2^k-1-BN} \mid (a,b,BN)\in S1\}$

MAX2= $\max\{F_{a,BN}+F_{b,2^k-1-BN} \mid (a,b,BN)\in S2\}$

∵性质④包含性质⑤

∴S1⊆S2

∴MAX1≤MAX2

由推论可以得到：**任意的（a,b,BN2）∈S2 对应的 $F_{a,BN2}+F_{b,2^k-1-BN2}$，都有一个(a,b,BN1) ∈S1 对应的 $F_{a,BN1}+F_{b,2^k-1-BN1}$ 大于或等于这个值。**

∴MAX1≥MAX2

∴MAX1=MAX2

所以只要保证性质⑤，求出来的函数③的最大值就是 Friendliness 的最大值。

证毕。

实现时，为了保证满足性质⑤，只要先以 $A_{?,k}$ 为关键字将 A 从小到大排序，令 $Best_{i,BN}$ 为 $\max\{F_{a,BN} \mid a\le i\}$，这只要 O (n×$2^k$)的时间就可以把 $Best$ 计算出来。然后，枚举 i 和第 k 位为 1 的 BN，以 $Best_{i-1,2^k-1-BN}+F_{i,BN}$ 更新最大值。最后输出最大值。

复杂度分析：时间复杂度为 O(nlogn+n×2^k)。空间复杂度为 O(n×2^k)。

到此，利用“放宽”方法已经很好地解决了此题。

【小结】

归纳上面解题的步骤：可以得出用“放宽”方法解决最大化（最小化）的一种方法：

1、把任务转化成数学模型；
2、找出原始算法；
3、用数据结构优化原始算法；
4、分析优化后的苛刻条件，将其转化为宽松的条件。

[例三]消防站[1]

【问题描述】

Z 国有 n 个城市，从 1 到 n 给这些城市编号。城市之间连着高速公路，并且每两个城市之间有且只有一条通路。不同的高速公路可能有不同的长度。最近 Z 国经常发生火灾，所以当地政府决定在某些城市修建一些消防站。在城市 k 修建一个消防站须要花费大小为$W(k)$的费用。函数 W 对于不同的城市可能有不同的取值。如果在城市 k 没有消防站，那么它到离它最近的消防站的距离不能超过$D(k)$。每个城市在不超过距离 D(这个城市）的前提下，必须选择最近的消防站作为负责站。函数 D 对于不同的城市可能有不同的取值。为了节省钱，当地政府希望你用最少的总费用修建一些消防站，并且使得这些消防站满足上述的要求。

【问题分析】

【数学模型】

首先，以 n 个城市为结点、高速公路为边，高速公路长为边权构造成一个图。由性质“每两个城市之间有且只有一条通路”可知这个图是一棵树。

令 $dis(i, j)$ 为结点 i 和结点 j 之间的距离。任务是找出一个 01 序列 $X_1, X_2, X_3.....X_n$，使得对于$1 \le i \le n$,都有

$$\min\{dis(i, j) \mid X_j = 1\} \le D(i)$$

并且使得目标函数

$$Z = \sum_{i=1}^{n} X_i \times W(i)$$

最小化。

【算法模型分析】

由于这题涉及到距离和图论等方面，便可猜想这是一道用图论算法解决的问题。可是在尝试过许多图论算法之后却发现这种猜想是走不通的。

[1] 选自 POJ2152 的《Fire》 Author,Lou Tiancheng

这时就要充分地利用问题的特殊性。我们知道这图是一棵树，并且这题是求目标函数最小化的问题。根据这些特性，我们基本上可以肯定这题的算法是树型动态规划。

【确定动态规划时的矛盾】

用动态规划算法解题首先要做的是确定好状态，这应该是不容置疑的，因为状态表示是动态规划中的重中之重。一般地，树型动态规划的状态中会有一个参数 *Root* , *Root* 表示此状态的研究对象是以 *Root* 为根的子树。

但是，如果仅用 F_{Root} 表示在以 *Root* 为根的子树中，修建符合要求(子树中的所有结点到最近消防站的距离不超过其对应的函数 D 值)的消防站的最小费用——即状态只用上述的一个参数，那么状态转移方程是无法找到的。因为这种状态表示无法反映出在哪里修建了消防站、离 *Root* 最近的消防站的详细情况。

为了解决这种情况，我们通常会增加一个参数，可称作增加一维。这时应该增加的参数既可以是 *Root* 到最近消防站的距离，又可以是 *Root* 的最近消防站的编号，也可以是树内的最近消防站的编号，同样可以是树外的最近消防站的编号。到底更加哪个参数是可行的呢？

可是事与愿违！所有的这些状态表示都难以找到好的转移方程。难道状态还要增加一个参数吗？还是这题本身是 NP 完全性问题、而不是用动态规划题目？别急，先来做个分析吧。

【初步分析】

分析上面难以找到转移方程的原因，便会发现产生这些矛盾主要是因为在状态转移时不能保证 *Root* 到最近消防站的距离或编号与定义的一致——换句话说，就是状态的定义太严格了——再换句话说，题目的要求太严格了。

所以，此时当务之急是放宽题目的要求。

【“放宽”方法转化限制】

现在面对的主要障碍无疑是，“每个城市在不超过距离 *D*(这个城市）的前提下，必须选择最近的消防站作为负责站”这一严格限制在状态转移中起着干扰作用。其实，我们并不须要知道最近的消防站是哪个，而只要保证在距离 *D*(这个城市）内至少有一个消防站就足够了。于是可以尝试放宽这个限制：把这个限制转化为“每个城市在不超过距离 *D*(这个城市）的前提下，可以选择任意一个消防站作为负责站”。

转化后，求出的最优解与转化前的是一样的。原因在于在转化后，必定存在

一个最优解满足性质“每个城市在不超过距离 D(这个城市）的前提下，必须选择最近的消防站作为负责站”。

现在每个城市都享有一定的“自由权”了，可以在自己的活动范围内自由地选择消防站作为负责站。此时就有必要把状态表示重新定义一下——令 $F_{i,j}$ 表示

① 在以 i 为根的子树里修建一些消防站；

② 在结点 j 必须修建一个消防站；

③ 以 i 为根的子树内的每个结点在不超过距离 D(这个结点）的前提下,选择一个在子树内或结点 j 上的消防站作为负责站；

④ 结点 i 必须选择结点 j 上的消防站作为负责站；

的最少总费用（如果 j 在树外则不算在 $F_{i,j}$ 内）。自然而然地“最近的消防站”这几个字在定义中消失了，这为以后确定动态规划转移方程提供了很大的方便。

【进一步分析】

经过“放宽”方法放宽限制后，状态表示基本上已经定下来了。进而要做的是确定动态规划转移方程。但是此时要确定下转移方程还是遇到了一点困难，总觉得欠缺一些性质、关系之类的。相信聪明的读者已经挖掘出原因了，那就是此时的限制过于宽松。

【“约制”方法增添限制】

动态规划算法讲求拓扑顺序和无后效性。然而现在每个城市对负责站的选取是任意的，于是就不妨对策略选取增添限制——假设城市 P_1 选取城市 P_m 的上消防站作为负责站，令 P_1 到 P_m 的路径为 $P_1\ P_2\ P_3 \ldots\ldots P_m$，那么对于任意 $i \in [1,m]$ 都有 P_i 的负责站为 P_m。如果我们证明总是在一个最优解满足上述的性质，那么此限制就能被增添了。下面将证明必有一个最优解满足上述的性质。

证明：

令某个最优解对应的 01 序列为 $X_1, X_2, X_3 \ldots\ldots X_n$。

构造：在 01 序列 $X_1, X_2, X_3 \ldots\ldots X_n$ 的布局下，首先增加一个结点 s，在 s 和有消防站的结点之间连一条权值为 0 的边。然后以 s 为源点做一次 Dijkstra，并

记录下前驱结点。对于每个结点，如果结点有消防站则选择其上的消防站为负责站，否则选择前驱的负责站为其负责站。

满足上述性质和必要限制：

1、 设任意一个结点到源点的路径为 $P_1\ P_2\ P_3 \ldots\ldots P_m\ s$，易知任意 $i \in [2,m]$ 都有 P_i 为 P_{i-1} 的前驱，而 P_m 的负责站为 $P_m \Rightarrow P_{m-1}$ 的负责站为 $P_m \ldots\ldots \Rightarrow P_1$ 的负责站为 P_m，所以任意 $i \in [1,m]$ 都有 P_i 的负责站为 P_m。

2、 由于每个结点都选择最近的消防站，所以它与负责站的距离不超过 D(这个结点）。

3、 而构造选取的消防站与最优解是一样的，所以总费用是最少的。

综上所述，总是在一个最优解(构造出来的方案)满足上述的性质。

证毕。

如今，上述的限制终于可以被正确地增添上了。

【确定动态规划转移方程】

经过两番转化后，动态规划转移方程已经可以被确定下来了。为了转移方便，先定义一个简单的辅助状态 $Best$，$Best_i$ 表示在以 i 为根的子树中，修建合符要求(子树中所有结点到其树内的负责站的距离不超过其对应的函数 D 值)的消防站的最小费用。明显地

$$Best_i = \min\{F_{i,j} \mid j在以i为根的子树中\}$$

下面对 F 进行分析：

①当 $dis(i,j) > D(i)$ 时，$F_{i,j} = +\infty$，这表示不存在状态 $F_{i,j}$；

②当 $dis(i,j) \le D(i)$ 时，

⑴当 j 在以 i 为根的子树外时，对于 i 的每个儿子 k 都有两种选择：选择以 k 为根的子树内或外的消防站为负责站。当选择以 k 为根的子树内的消防站为负责站时，其子树所需的最少费用为 $Best_k$，当选择以 k 为根的子树外的消防站为负责站时，根据新添的限制易知 k 只可以选择 j 上的消防站作为负责站，此时其子树所需的最少费用为 $F_{k,j}$。综上得到

$$F_{i,j}=\sum_{k\text{为}i\text{的儿子}}\min\{Best_k,F_{k,j}\}$$

⑵当$i=j$时，i的每个儿子的选择情况与⑴中的一样。此时还要加上修建j上的消防站的费用。因此

$$F_{i,j}=W(j)+\sum_{k\text{为}i\text{的儿子}}\min\{Best_k,F_{k,j}\}$$

⑶当$i\neq j$并且j在以i为根的子树内时，此时j必定在i的某个儿子$child$的子树里。对于i的每个不是$child$的儿子其选择情况与⑴中的一样，而对于$child$，根据新添的限制它只能选择j作为负责站。综上得到

$$F_{i,j}=F_{child,j}+\sum_{\substack{k\text{为}i\text{的儿子}\\ \text{并且}k\neq child}}\min\{Best_k,F_{k,j}\}$$

复杂度分析：时间复杂度为$O(n^2)$，空间复杂度为$O(n^2)$。

【小结】

“放宽”方法和“约制”方法不总是互相排斥、矛盾的，它们往往会互相补充。它们各自可以在需要它们的方面发挥特长——应用“放宽”方法确定状态；应用“约制”方法确定状态转移方程。在保证能找到答案的前提下，对于过于严格而阻挠前进的条件、限制，我们对它进行“放宽”；对于过于宽松而茫无头绪的条件、限制，我们对它进行“约制”——这就是所谓的一张一弛了。一张一弛不仅是文武之道，更是解题之道。

总结

不管是“约制”方法还是“放宽”方法，其目的都是要简化问题。

在应用这两种方法的时候，首先要摸清这两者的适用范围、所起的作用和效果。一般而言，当条件、限制过于宽松时就用“约制”方法来约制它们，当条件、限制过于繁杂时就用“放宽”方法来放宽它们。

一张一弛作为一种解题方法，是须要在思索、做题中慢慢形成的。除了实践外，还有几点是须要注意的：

敢于创新、

敢于猜想、

敢于类比、

敢于拓展。

其中敢于创新显得尤为重要,因为只有不断创新和实践，才能“拨得云开见月明”。

【感谢】

衷心感谢向老师给我的指导和意见。
衷心感谢长郡中学的郭化阳给我解答了例 1。
衷心感谢周戈林、郭化阳、杨沐给我提出宝贵意见。