摘　要

整数规划和线性规划有类似的地方, 解整数规划技巧性更高一些. 下面粗略得叙述了一下整数规划的部分知识, 并用GLPK软件给了两个例子.

仍主要以卢开澄的书的内容为主.

1 整数规划问题概述 [1]

若线性规划问题中的变量不在实数域$\mathbb{R}$ 上取, 而都在整数域上取, 则称这样的问题为整数规划问题.

若有部分变量去实数, 部分取整数, 则称混合规划问题.

可将规划问题表示为

$$\begin{aligned} max \quad & z = \boldsymbol{Cx} + \boldsymbol{Dy} \\ s.t. \quad & \boldsymbol{Ax} + \boldsymbol{Dy} \leqslant \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \boldsymbol{0} \\ & \boldsymbol{y} \geqslant \boldsymbol{0} \\ & \boldsymbol{x}\text{为整数向量} \end{aligned} \tag{1.1}$$

1.1 整数规划问题的解法

对一些很特殊的问题, 系数矩阵为幺模矩阵. 对有些问题.整数规划的解与线性规划相同或在线性规划解的附近, 但一般来说整数规划的解会和线性规划不太一样.

对于有界问题, 可行解中的整点个数是有限的, 所以最基本的办法就是用穷举法. 显然穷举法的效率非常低. 为了表示问题, 可以将解适当编码, 化归为状态树上的搜索(树的遍历, DFS, BFS) , 通过适当剪枝来加速搜索.

1.2 分支定界法

分支定界法是解决搜索问题的强大工具. 可解决流动推销员问题, 最佳匹配问题等很多问题.《单目标、多目标与整数规划》[1] 中还介绍了用分支定界法解混合规划的问题.

分支定界的基本思想是对搜索空间进行分割, 在分割的状态空间上估计最大值(若问题求最大值). 若最值小于已知可行解. 则最大值肯定不在此子空间上取到. 将此搜索分支剪去, 在其他分支上搜索. 分支定界的关键就是好的估界函数, 这样就能减少搜索次数, 更快得到最优解.

就实现分支定界算法而言, 由于是在状态空间中搜索, 肯定要生成一棵搜索树, 这里必然用到树的遍历等算法, 这些算法对熟悉计算机程序设计的人来说是很平常的, 对一般人来说却的确有些困难. 这里就不详述了, 可以看看一些例子, 里面用到了搜索算法. 对建模竞赛的问题, 我建议用GLPK等软件解决, 方便快捷, 可以避免陷入编程细节(毕竟不是ACM程序设计竞赛).

这里给出用分支定界法解整数规划的流程图1 (顺变试一下做流程图的方法).

2 问题二的解

Problem 2.1. 解下列整数规划问题.

$$\begin{cases} max \quad & z = 3x_1 + 2x_2 \\ s.t. \quad & x_1 + x_2 \leqslant 3 \\ & x_1 \geqslant 0 \\ & x_2 \geqslant 0 \\ & x_1, x_2 \geqslant 0, \quad x_i \in \mathbb{N}, i = 1, 2 \end{cases} \tag{2.1}$$

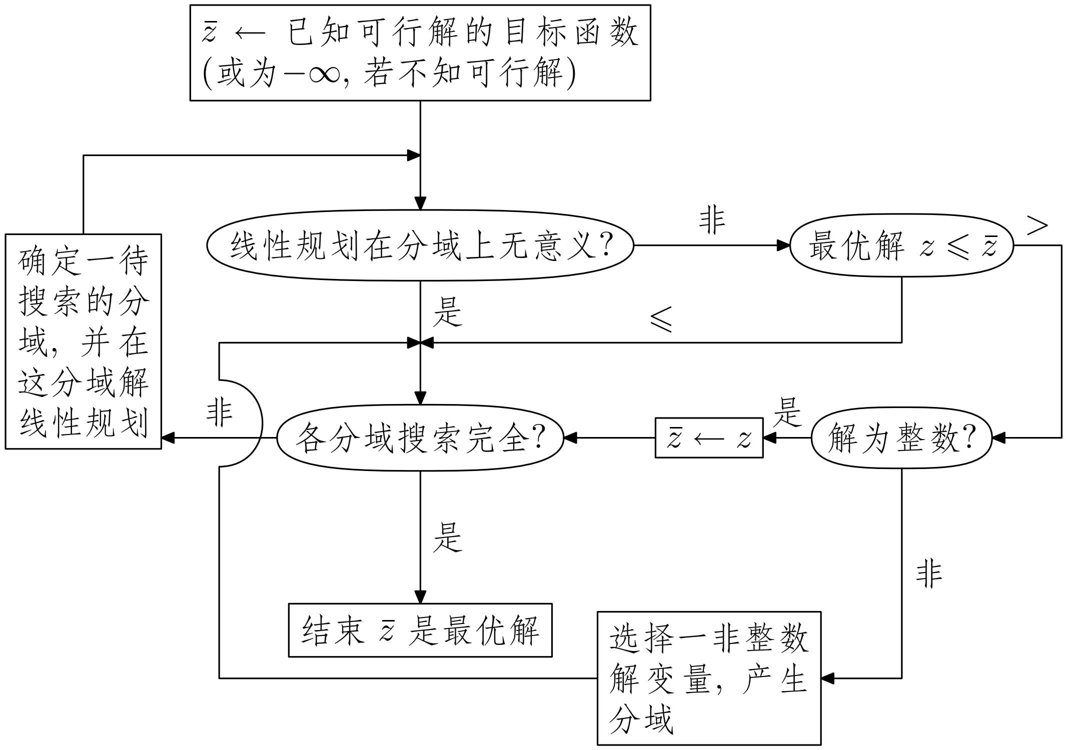

图 1: 分支定界法解整数规划的流程图

Answer 2.1.1. *用CPX LP 格式描述问题:*

```
\* Problem: p1 *\
max
z : 3 x1 + 2 x2
s.t.
a:  x1 + x2 <=3

bound
   x1 <=2
   x2 <=2
integer
   x1
   x2
end
```

用glpk求解结果:

```
G:\mydoc\my text\建模\modeling\05sum\integerp>cat inter1.sol
Problem:
Rows:        1
Columns:     2 (2 integer, 0 binary)
Non-zeros:   2
Status:      INTEGER OPTIMAL
```

```
Objective:  z = 8 (MAXimum) 8 (LP)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 a                         3                             3

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 x1           *            2             0             2
     2 x2           *            1             0             2

End of output
```

最优解为$x_1 = 2, x_2 = 1$，此时z取最大值8

3 钻井问题

Problem 3.1. *某钻井队要从以下十个可供选择的井位: $s_1, s_2, \ldots, s_{10}$ ，中确定5个钻井探油，使总的钻探费用为最小。十个井位的钻探费用估计分别为$c_1, c_2, \ldots, c_{10}$ 且井位选择必须满足如下限制条件:*

- *或选择s_1 和s_7 ，或选择s_8 ;*
- *选择了s_3 或s_4 ，就不能选s_5 ，或反过来也一样;*
- *在s_5, s_6, s_7, s_8 中最多只能选两个，*

试建立数学模型求解。

Answer 3.1.1. *用MathProg语言描述问题.*

```
set N:= 1..10;

/* 数组c保存每个钻井的代价 */
param c{N};
/* cost */

/* 01变量 s[i] 表示是否选取这个钻井 */
var s{N} binary;

/* 目标 代价最小 */
minimize cost: sum{i in N} c[i] * s[i];

/* 约束，选取5个钻井*/
/* TO */
s.t. all5: sum{i in N} s[i] = 5;
```

```
/* 要么选1和7, 要么选8 */
/* T1 */
s.t. s1e7: s[1]-s[7] = 0;
s.t. s1or8: s[1]+s[8] = 1;

/* 3,4 不能与5同选 */
/* T2 */
s.t. s3n5: s[3] + s[5], <= 1;
s.t. s4n5: s[4] + s[5], <= 1;

/* 5,6,7,8, 最多选两个 */
/* T3 */
s.t. bin5678: s[5] + s[6] + s[7] + s[8] , <=2 ;

solve;

/* printf{i in N} "c[%d] = %g\n",i, c[i]; */

printf "minimize cost = %g\n", cost;
for{i in N: s[i] =1}
{
   printf "s[%d]\t", i;
}

data;
/*  代价的数据 */
param :c:=
1   1
2   5
3   6
4   9
5   12
6   8
7   1
8   6
9   4
10  3
;
end;
```

这里随便选了*10*个测试数据代表每个井的代价. 用*glpk* 解得(截取输出片断):

```
...
minimize cost = 14
s[1]    s[2]    s[7]    s[9]    s[10]
...
```

即最小总代价为14, 选s_1, s_2, s_7, s_9, s_10 这五口井.

参考文献

[1] 清华大学出版社,《单目标、多目标与整数规划》, 卢开澄, 1999

[2] 汇集了一些问题的程序解法, 使用python语言, http://wiki.woodpecker.org.cn/moin/PyPorgramGames

[3] GLPK (GNU Linear Programming Kit) , http://www.gnu.org/software/glpk/glpk.html , 2005

[4] 中文的GLPK介绍下载页, http://wiki.woodpecker.org.cn/moin/GLPK , 2005