Column Generation & Benders Decomposition
基于Gurobi 的列生成和 Benders 分解方法

刃之砺信息科技（上海）有限公司

课程大纲

(1) 列生成(Column Generation)

首先介绍一些列生成涉及到的基础知识，然后给出列生成算法的框架，最后通过Cutting Stock Problem 给出具体的求解步骤。

(2) Benders 分解(Benders Decomposition)

首先会描述一下Benders 分解算法的具体思路，并且给出经典Benders 分解算法的伪代码。最后通过数值案例详细展示Benders 分解算法的迭代过程。

列生成(Column Generation)

- Reduced Cost

$$\begin{aligned} & min\ \ c^T x \\ & st.\ Ax = b \\ & \qquad x \geq 0 \end{aligned}$$

令 $x = [x_B, x_N]$，其中x_B表示基变量，x_N表示非基变量。相应的 $A = [B, N]$， $c^T = [c_B^T, c_N^T]$

$Ax = b \iff Bx_B + Nx_N = b \iff x_B = B^{-1}b - B^{-1}Nx_N$ (非基变量$x_N = 0$， $x_B = B^{-1}b$)

$min\ \ c^T x \iff min\ \ c_B^T x_B + c_N^T x_N \iff min\ \ c_B^T(B^{-1}b - B^{-1}Nx_N) + c_N^T x_N \iff min\ \ c_B^T B^{-1} b + (\boldsymbol{c_N^T - c_B^T B^{-1} N}) x_N$

称 $\boldsymbol{c_N^T - c_B^T B^{-1} N}$ 为 **Reduced Cost**

(1) $\exists\, n \in N$，使得 $c_n^T - c_B^T B^{-1} n < 0$，因此可以适当增加非基变量$x_n$ 的值，从而降低目标值。

(2) $\forall\, n \in N$，都有 $c_n^T - c_B^T B^{-1} n \geq 0$，获得最优解。

列生成(Column Generation)

- Reduced Cost and Dual Variables

$$\begin{aligned} &min \;\; c^T x \\ &st. \;\; Ax = b \\ &\qquad\;\; x \ge 0 \end{aligned} \quad \xleftrightarrow{\textbf{Dual}} \quad \begin{aligned} &max \;\; y^T b \\ &st. \;\; y^T A \le c^T \end{aligned}$$

假设找到了原问题的最优解 $[x_B, 0] \Leftrightarrow \forall N$，$c_N^T - c_B^T B^{-1} N \ge 0$，即 $c_B^T B^{-1} N \le c_N^T$

$\boldsymbol{c_B^T B^{-1}} A = \boldsymbol{c_B^T B^{-1}}[B, N] = \left[c_B^T，\; \boldsymbol{c_B^T B^{-1}} N\right] \le [c_B^T，\; c_N^T] = \boldsymbol{c^T} \Leftrightarrow \boldsymbol{c_B^T B^{-1}} A \le \boldsymbol{c^T} \Leftrightarrow \boldsymbol{c_B^T B^{-1}}$是对偶问题可行解

令 $y^T = \boldsymbol{c_B^T B^{-1}}$，$y^T b = \boldsymbol{c_B^T B^{-1} b} = \boldsymbol{c_B^T} x_B = c^T x$，**根据对偶定理，** y^T **为对偶问题最优解。**

Reduced Cost $= \boldsymbol{c_N^T}$ − Dual Variables$*\boldsymbol{N}$

列生成(Column Generation)

- Column Generation 思路

$$
\begin{aligned}
& min\ \ c_1x_1 + c_2x_2 + \cdots \\
\text{s.t}\quad & a_{11}x_1 + a_{12}x_2 + \cdots \le b_1 \\
& a_{21}x_1 + a_{22}x_2 + \cdots \le b_2 \\
& \qquad\qquad \vdots \\
& a_{m1}x_1 + a_{m2}x_2 + \cdots \le b_m \\
& x_1, x_2 \ldots \text{Integer}
\end{aligned}
$$

对于一个整数规划模型,如果问题变量过多(**往往很难显性枚举出所有变量**),可以先枚举部分变量构造一个规模相对较小的模型，然后通过迭代不停地向较小模型里面添加新的变量(列),直到没有新的变量(列)添加为止。

列生成(Column Generation)

- 案例 Cutting Stock Problem

假设需要长度为3,7,9,16米的钢管各25,30,14,8根,目前只有长度为20米的钢管若干,请安排合理地切割方案,使得消耗的钢管数量最少。

可以很容易给出一种切割方案:每根钢管只切割一种长度,

3m的钢管 （需求数量25） 需要20米的钢管5根;

7m的钢管 （需求数量30） 需要20米的钢管15根;

9m的钢管 （需求数量14） 需要20米的钢管7根;

16m的钢管 （需求数量8） 需要20米的钢管8根。

上面的切割方案虽然能够满足需求,但浪费的钢管较多。例如20米的钢管只切7m长度会剩下6m,余料完全可以用来切3m长度。

列生成(Column Generation)

- 第一种建模方案(常规思路)

参数：	I	所需钢管种类集合；
	K	目前未切割的钢管集合；
	D_i	第i种长度钢管的需求数量；
	l_i	第i种长度钢管的长度；
	L_k	第k根未切割的钢管的长度。
变量：	x_{ik}	表示第k根钢管切割第i种长度的数量；
	y_k	表示第k根钢管是否使用。

$$\min \sum_k y_k$$

满足需求	$\sum_k x_{ik} \geq D_i, \forall i \epsilon I$
钢管切割的长度不能超过本身长度	$\sum_i l_i x_{ik} \leq L_k y_k, \forall k \epsilon K$

$$x_{ik} \in N, y_k \in \{0,1\}, \forall i \epsilon I, k \in K$$

列生成(Column Generation)

- 第二种建模方案(列生成思路)

参数： C_{ip} 表示在第p种切割模式下，切出第i种长度的钢管数量；

P 表示切割模式集合。

变量： z_p 表示第p种切割模式使用的次数。

$$min \sum_p z_p$$

满足需求 $$\sum_p C_{ip} z_p \geq D_i, \forall i \epsilon I$$

$$z_p \text{ Integer}, \forall p \epsilon P$$

切割模式组合非常多，因此枚举所有z_p几乎是不可行的，也没有必要。因为并不是所有的切割模式都会用到，例如，每根钢管只切割一种长度浪费较多。**那么如何去寻找好的切割模式？**

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem步骤

首先，给定几种初始的切割模式 $\bar{\bar{P}} \in P$ 带入原列生成模型中，并将变量松弛为连续型：

$$min \sum_{p} z_p$$

$$\sum_{p} C_{ip} z_p \geq D_i, \forall i \epsilon I$$

$$\boldsymbol{z_p \geq 0, \forall p \epsilon \bar{\bar{P}}}$$

求解上述模型获得对应的对偶变量取值，记为 λ_i 。

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem步骤

然后，思考是否存在新的切割模式p_{new}加到模型进而降低目标函数值？假设存在，必然会有变量$z_{p_{new}}$的Reduced Cost = $1-\sum_i \lambda_i C_{ip_{new}} < 0$。

$$min \quad (\sum_p z_p) + z_{p_{new}}$$

$$(\sum_p C_{ip} z_p) + C_{ip_{new}} z_{p_{new}} \geq D_i, \forall i \epsilon I$$

$$z_p \geq 0, \forall p \epsilon \overline{\overline{P}} \cup p_{new}$$

为了获取更优的目标值，往往会选择Reduced Cost 最小的切割模式加入模型中。那么如何确定Reduced Cost 最小的切割模式？

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem步骤

接着,通过一个子问题确定Reduced Cost 最小的切割模式。对于任意的切割模式,都需要满足钢管长度限制。因此可以构造如下模型(省略下标p_{new}),其中变量C_i 表示该切割模式切出第i种长度的钢管数量;

$$min \quad 1-\sum_i \lambda_i C_i$$

$$\sum_i C_i l_i \leq L, \forall i \epsilon I$$

$$C_i \text{ Integer}, \forall i \epsilon I$$

如果子问题最优目标值小于0,那么意味着发现更好的切割模式,将新切割模式加入到模型中重复以上步骤直到没有更好的切割模式出现。

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem 步骤

初始给出四种切割模式:

模式	第一种	第二种	第三种	第四种
3m	6	0	0	0
7m	0	2	0	0
9m	0	0	2	0
16m	0	0	0	1

$$
\begin{aligned}
& min \quad z_1 + z_2 + z_3 + z_4 \\
& 6z_1 + 0z_2 + 0z_3 + 0z_4 \geq 25 \\
& 0z_1 + 2z_2 + 0z_3 + 0z_4 \geq 30 \\
& 0z_1 + 0z_2 + 2z_3 + 0z_4 \geq 14 \\
& 0z_1 + 0z_2 + 0z_3 + 1z_4 \geq 8 \\
& z_1, z_2, z_3, z_4 \geq 0
\end{aligned}
$$

求解模型，获得对应的对偶变量值 $\lambda =$ [0.16666666666666666, 0.5, 0.5, 1.0]

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem 步骤

通过子问题确定Reduced Cost最小的切割模式:

$$\min\ 1 - 0.16666666666666666 c_1 - 0.5c_2 - 0.5c_3 - c_4$$
$$3c_1 + 7c_2 + 9c_3 + 16c_4 \leq 20$$
$$c_1, c_2, c_3, c_4\ \text{Integer}$$

Reduced Cost最小的值为 −0.33333333333333333<0, 因此发现更好的切割模式:

$$(c_1, c_2, c_3, c_4) = (2, 2, 0, 0)$$

将新的切割模式加入到模型中,继续迭代。

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem步骤

$$
\begin{aligned}
\min \quad & z_1 + z_2 + z_3 + z_4 + \mathbf{z_{new}} \\
& 6z_1 + 0z_2 + 0z_3 + 0z_4 + \mathbf{2z_{new}} \geq 25 \\
& 0z_1 + 2z_2 + 0z_3 + 0z_4 + \mathbf{2z_{new}} \geq 30 \\
& 0z_1 + 0z_2 + 2z_3 + 0z_4 + \mathbf{0z_{new}} \geq 14 \\
& 0z_1 + 0z_2 + 0z_3 + 1z_4 + \mathbf{0z_{new}} \geq 8 \\
& z_1, z_2, z_3, z_4, \mathbf{z_{new}} \geq 0
\end{aligned}
$$

求解模型，获得对应的对偶变量值λ＝[0，0.5，0.5，1.0]

列生成(Column Generation)

- 列生成算法求解 Cutting Stock Problem 步骤

继续通过子问题确定Reduced Cost最小的切割模式:

$$\min\ 1 -\ 0c_1 - 0.5c_2 - 0.5c_3 - c_4$$
$$3c_1 + 7c_2 + 9c_3 + 16c_4 \le 20$$
$$c_1, c_2, c_3, c_4\ \text{Integer}$$

Reduced Cost最小的值为 0, 因此没能发现更好的切割模式, 终止迭代。列生成完后的模型:

$$\min\ \ z_1 + z_2 + z_3 + z_4 + z_{new}$$
$$6z_1 + 0z_2 + 0z_3 + 0z_4 + 2z_{new} \ge 25$$
$$0z_1 + 2z_2 + 0z_3 + 0z_4 + 2z_{new} \ge 30$$
$$0z_1 + 0z_2 + 2z_3 + 0z_4 + 0z_{new} \ge 14$$
$$0z_1 + 0z_2 + 0z_3 + 1z_4 + 0z_{new} \ge 8$$
$$z_1, z_2, z_3, z_4, z_{new} \ge 0$$

列生成(Column Generation)

- 完整案例代码

```
from gurobipy import *
TypesDemand = [3, 7, 9, 16]                  # 需求长度
QuantityDemand = [25, 30, 14, 8]             # 需求的量
LengthUsable = 20                            # 钢管长度
try:
    MainProbRelax = Model()                  # 松弛后的列生成主问题
    SubProb = Model()                        # 子问题

    # 构造主问题模型，选择的初始切割方案 每根钢管只切一种长度
    # 添加变量
    Zp = MainProbRelax.addVars(len(TypesDemand), obj=1.0, vtype=GRB.CONTINUOUS, name = 'z')
    # 添加约束
    ColumnIndex = MainProbRelax.addConstrs(quicksum(Zp[p] * (LengthUsable//TypesDemand[i]) \
    for p in range(len(TypesDemand)) if p==i) >= QuantityDemand[i] for i in range(len(TypesDemand)))
    MainProbRelax.optimize() # 求解
```

列生成(Column Generation)

- 完整案例代码

```
# 构造子问题模型
# 获得对偶值
Dualsolution = MainProbRelax.getAttr(GRB.Attr.Pi, MainProbRelax.getConstrs())
# 添加变量
Ci = SubProb.addVars(len(TypesDemand), obj=Dualsolution, vtype=GRB.INTEGER, name = 'c')
# 添加约束
SubProb.addConstr(quicksum(Ci[i] * TypesDemand[i] for i in range(len(TypesDemand))) <= LengthUsable)
SubProb.setAttr(GRB.Attr.ModelSense, -1)        # 设定优化方向
SubProb.optimize()                              # 求解
```

列生成(Column Generation)

- 完整案例代码

```
# 判断Reduced Cost是否小于零
while SubProb.objval > 1:
    # 获取变量取值
    columnCoeff = SubProb.getAttr("X", SubProb.getVars())
    column = Column(columnCoeff, MainProbRelax.getConstrs())
    # 添加变量
    MainProbRelax.addVar(obj=1.0, vtype=GRB.CONTINUOUS, name="CG", column=column)
    MainProbRelax.optimize()                    # 求解
    # 修改目标函数系数
    for i in range(len(TypesDemand)):
        Ci[i].obj = ColumnIndex[i].pi
    SubProb.optimize()
```

列生成(Column Generation)

- 完整案例代码

```
    # 将CG后的模型转为整数，并求解
    for v in MainProbRelax.getVars():
        v.setAttr("VType", GRB.INTEGER)
    MainProbRelax.optimize()
    for v in MainProbRelax.getVars():
        if v.X != 0.0:
            print('%s %g' % (v.VarName, v.X))

except GurobiError as e:
    print('Error code ' + str(e.errno) + ": " + str(e))

except AttributeError:
    print('Encountered an attribute error')
```

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

$$\min \ c^T x + f^T \mathbf{y}$$
$$s.t. \ Ax + B\mathbf{y} = b \qquad (1)$$
$$x \geq 0, \mathbf{y} \in Y$$

假设变量 $\mathbf{y}$ 是一个比较复杂的变量，如果该变量的值可以提前确定，剩下的模型往往会比较容易求解。因此通过变量的区分，可以将原模型拆解为两个相对较小的模型，然后通过间接迭代求解小模型到达求解原模型的目的。

$$\min \ f^T \mathbf{y} + g(y)$$
$$\mathbf{y} \in Y \qquad (2)$$

$$g(\bar{y}) = \min \ c^T x$$
$$Ax = b - B\bar{y} \qquad (3)$$
$$x \geq 0$$

如果模型(3)无界 $\Rightarrow$ 模型(2)无界 $\Rightarrow$ 原模型(1)无界。

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

如果模型(3)有界,写出模型(3)的对偶模型:

$$\max\ \lambda^T(b - B\bar{y})$$
$$A^T\lambda \le c \qquad (4)$$
$$\lambda\ free$$

可以看到**模型(4)的可行域与变量 y 没有关系**,变量 y 的值只会影响其目标函数值。如果模型(4)的可行域为空集,那么根据对偶性可知模型(3)无界或者为空。

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

假设模型(4)非空，可以枚举出所有的extreme points $(\alpha_p^1, \alpha_p^2, \alpha_p^3, \ldots \alpha_p^I)$和 extreme rays $(\alpha_r^1, \alpha_r^2, \alpha_r^3, \ldots \alpha_r^J)$。因此求解模型(4)可以等价为：

- **最优：** 寻找 α_p^i 最大化 ${\alpha_p^i}^T(b - B\bar{y})$。
- **排除不可行：** 检测 ${\alpha_r^j}^T(b - B\bar{y}) \leq 0$ 是否成立。如果${\alpha_r^j}^T(b - B\bar{y}) > 0$，模型(4)无界，进而模型(3)不可行。

因此模型(4)可以等价转化为：

$$\max_{i \in \{1,2,\ldots,I\}} {\alpha_p^i}^T(b - B\bar{y})$$
$${\alpha_r^j}^T(b - B\bar{y}) \leq 0, \quad \forall j \in \{1,2,\ldots,J\}$$

$$\Longrightarrow$$

$$\begin{aligned} & \min\ z \\ & {\alpha_p^i}^T(b - B\bar{y}) \leq z, \quad \forall i \in \{1,2,\ldots,I\} \\ & {\alpha_r^j}^T(b - B\bar{y}) \leq 0, \quad \forall j \in \{1,2,\ldots,J\} \\ & z\ free \end{aligned} \tag{5}$$

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

进一步将模型(5)带入到模型(2)中，可以得到原模型的一个等价模型(6)：

$$\min\ f^T y + z$$

$$\alpha_p^{i\,T}(b - By) \le z, \qquad \forall i \in \{1,2,\ldots,I\} \qquad (6)$$

$$\alpha_r^{j\,T}(b - By) \le 0, \qquad \forall j \in \{1,2,\ldots,J\}$$

$$y \in Y, z\ free$$

模型(6)包含变量 y 和 z，以及大量的约束，这些约束在实际求解释很难全部枚举，因此很难直接求解模型(6)。

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

因此Benders 分解方法不直接求解模型(6), 具体迭代思路如下:

首先, 找到一组候选解 (y^*, z^*), 将y^*带入到模型(3)中, 获得如下模型(被称为Benders subproblem, SP)

$$g(y^*) = min\ c^T x$$

$$Ax = b - By^* \qquad \text{(SP)}$$

$$x \geq 0$$

求解SP模型可能有三种结果:

- 最优解且 $g(y^*) = z^*$, 发现原问题最优解, 停止迭代。
- 最优解且 $g(y^*) > z^*$, 可以构造一个形如${\alpha_p}^T(b - By) \leq z$ (benders optimality cut) 加入到 Benders master problem, MP 。
- 不可行, 可以构造一个形如${\alpha_r}^T(b - By) \leq 0$ (benders feasibility cut) 加入到 Benders master problem, MP 。

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 思路

进而得到一个随着迭代规模不断增加的Benders master problem Model, MP。

$$\min\ f^T y + z$$

$$\alpha_p{}^T(b - By) \le z \ \text{(benders optimality cut)} \qquad \text{(MP)}$$

$$\alpha_r{}^T(b - By) \le 0 \ \text{(benders feasibility cut)}$$

$$\mathrm{y} \in Y, z\ \ free$$

接着求解MP, 获得一组**新的**$(\boldsymbol{y}^*, \boldsymbol{z}^*)$, 然后重复上述迭代过程直到 $g(y^*) = z^*$。

Benders 分解(Benders Decomposition)

- Classic Benders Decomposition 算法结构

```
Classic Benders Decomposition
Identify a MP and an easy SP(y)
repeat
    solve MP obtaining the solution (y*, z*)
    solve SP(y*)
    if SP(y*) is feasible
        if obj(SP(y*)) = z*
            STOP
        else
            add to MP the benders optimality cut
    else
        add to MP the benders feasibility cut.
until (end condition)
```

Benders 分解(Benders Decomposition)

- Benders 分解数值案例

$$min\ \ x_1 + x_2 + x_3 + x_4 + x_5 + 7y_1 + 7y_2 + 7y_3 + 7y_4 + 7y_5$$

$$x_1 + x_4 + x_5 = 8$$

$$x_2 + x_5 = 3$$

$$x_3 + x_4 = 5$$

$$x_1 \le 8y_1$$

$$x_2 \le 3y_2$$

$$x_3 \le 5y_3$$

$$x_4 \le 5y_4$$

$$x_5 \le 3y_5$$

$$x_1, x_2, x_3, x_4, x_5 \ge 0,\ \ y_1, y_2, y_3, y_4, y_5 \in \{0,1\}$$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

首先，确定合适的MP和SP，针对本数值案例，变量 x,y 分别为连续变量和{0, 1}变量，所以变量 y 相对复杂，因此构建MP模型如下：

$$min\ \ 7y_1+7y_2+7y_3+7y_4+7y_5+z$$

$$z\geq 0,\ \ y_1,y_2,y_3,y_4,y_5\epsilon\{0,1\}$$

求解MP模型，获得最优解 $y^*=(0,0,0,0,0)$，$z^*=0$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

将第一组候选解(y^*, z^*)，其中$y^* = (0,0,0,0,0)$，$z^* = 0$ 带入SP模型：

$$min \quad x_1 + x_2 + x_3 + x_4 + x_5 + 7*0 + 7*0 + 7*0 + 7*0 + 7*0$$

$$x_1 + x_4 + x_5 = 8$$

$$x_2 + x_5 = 3$$

$$x_3 + x_4 = 5$$

$$x_1 \le 8*0,\ x_2 \le 3*0,\ x_3 \le 5*0,\ x_4 \le 5*0,\ x_5 \le 3*0$$

$$x_1, x_2, x_3, x_4, x_5 \ge 0$$

显然SP模型不可行，Dual-SP无界，可以获得Dual-SP的一个extreme ray(1.0, 0.0, 0.0, -1.0, 0.0, 0.0, -1.0, -1.0)

可以向MP模型中添加*benders feasibility cut*

$$8y_1 + 5y_4 + 3y_5 \ge 8$$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

更新MP模型：

$$min \quad 7y_1 + 7y_2 + 7y_3 + 7y_4 + 7y_5 + z$$

$$8y_1 + 5y_4 + 3y_5 \geq 8$$

$$z \geq 0, \quad y_1, y_2, y_3, y_4, y_5 \epsilon \{0,1\}$$

求解该模型获得最优解为$y^* = (1,0,0,0,0)$，$z^* = 0$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

将第二组候选解(y^*, z^*)，其中$y^* = (1,0,0,0,0)$，$z^* = 0$带入SP模型：

$$min \;\; x_1 + x_2 + x_3 + x_4 + x_5 + 7*1 + 7*0 + 7*0 + 7*0 + 7*0$$

$$x_1 + x_4 + x_5 = 8$$

$$x_2 + x_5 = 3$$

$$x_3 + x_4 = 5$$

$$x_1 \le 8*1,\; x_2 \le 3*0,\; x_3 \le 5*0,\; x_4 \le 5*0,\; x_5 \le 3*0$$

$$x_1, x_2, x_3, x_4, x_5 \ge 0$$

显然SP模型仍然不可行，Dual-SP的一个extreme ray（0.0,1.0,1.0,0.0,−1.0,−1.0,−1.0,−1.0）。向MP模型再次添加*benders feasibility cut*

$$3y_2 + 5y_3 + 5y_4 + 3y_5 \ge 8$$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

继续更新MP模型：

$$\min \;\; 7y_1+7y_2+7y_3+7y_4+7y_5+z$$

$$8y_1+5y_4+3y_5\geq 8$$

$$3y_2+5y_3+5y_4+3y_5\geq 8$$

$$z\geq 0,\;\; y_1,y_2,y_3,y_4,y_5\in\{0,1\}$$

求解该模型获得最优解为$y^*=(0,0,0,1,1)$，$z^*=0$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

将第三组候选解(y^*, z^*), 其中$y^* = (0,0,0,1,1)$, $z^* = 0$带入SP模型:

$$min \;\; x_1 + x_2 + x_3 + x_4 + x_5 + 7*0 + 7*0 + 7*0 + 7*1 + 7*1$$

$$x_1 + x_4 + x_5 = 8$$

$$x_2 + x_5 = 3$$

$$x_3 + x_4 = 5$$

$$x_1 \le 8*0,\;\; x_2 \le 3*0,\;\; x_3 \le 5*0,\;\; x_4 \le 5*1,\;\; x_5 \le 3*1$$

$$x_1, x_2, x_3, x_4, x_5 \ge 0$$

SP模型有最优解 $obj = 22 > 0$, 因此可以向MP模型添加*benders optimality cut*

$$5y_4 + 3y_5 + z \ge 16$$

Benders 分解(Benders Decomposition)

- Benders 分解数值案例具体步骤

继续更新MP模型：

$$min \quad 7y_1 + 7y_2 + 7y_3 + 7y_4 + 7y_5 + z$$

$$8y_1 + 5y_4 + 3y_5 \geq 8$$

$$3y_2 + 5y_3 + 5y_4 + 3y_5 \geq 8$$

$$5y_4 + 3y_5 + z \geq 16$$

$$z \geq 0,\ y_1, y_2, y_3, y_4, y_5 \epsilon \{0,1\}$$

求解该模型获得最优解为$y^* = (0,0,0,1,1)$，$z^* = 8$, 因为y^*与上一步值相同, 因此带入SP模型最优解 $obj = 22$。发现了原问题的最优解，停止迭代过程。

原问题最优解$obj = 22$，$x = (0,0,0,5,3)$，$y = (0,0,0,1,1)$

Benders 分解(Benders Decomposition)

- 案例代码

```
from gurobipy import *
def addBendersCuts(SP_Dual_obj, x):
   if SP_Dual.status == GRB.Status.UNBOUNDED:                    #模型无界
      ray = SP_Dual.UnbdRay
      MP.addConstr(8*ray[0] + 3*ray[1] + 5*ray[2] + 8*ray[3]*y[0] + 3*ray[4]*y[1] + 5*ray[5]*y[2] + 5*ray[6]*y[3] + 3*ray[7]*y[4]<= 0)
   elif SP_Dual.status == GRB.Status.OPTIMAL:                    #发现最优解
      MP.addConstr(8*Vdual1[0].x + 3*Vdual1[1].x + 5*Vdual1[2].x + 8*Vdual2[0].x*y[0] + \
                      3*Vdual2[1].x*y[1] +   5*Vdual2[2].x*y[2] + 5*Vdual2[3].x*y[3] + 3*Vdual2[4].x*y[4] <= z)
      SP_Dual_obj[0] = SP_Dual.ObjVal                           #获取最优解
      x.append(x1.pi)                                           #获取SP模型解
      x.append(x2.pi)
      x.append(x3.pi)
      x.append(x4.pi)
      x.append(x5.pi)
   else:                                                        #其他状态
      print (SP_Dual.status)
```

Benders 分解(Benders Decomposition)

- 案例代码

```
try:
    MP = Model()                          #Benders Master Problem
    SP_Dual = Model()                     #dual of Benders SubProblem
    y= MP.addVars(5, obj=7, vtype=GRB.BINARY, name='y')
    z= MP.addVar(obj=1, vtype=GRB.CONTINUOUS, name='z')
    Vdual1 = SP_Dual.addVars(3, lb=-GRB.INFINITY, vtype=GRB.CONTINUOUS, name='Vdual1')
    Vdual2 = SP_Dual.addVars(5, lb=-GRB.INFINITY,ub=0, vtype=GRB.CONTINUOUS, name='Vdual2')
    x1 = SP_Dual.addConstr(Vdual1[0] + Vdual2[0] <=1)
    x2 = SP_Dual.addConstr(Vdual1[1] + Vdual2[1] <=1)
    x3 = SP_Dual.addConstr(Vdual1[2] + Vdual2[2] <=1)
    x4 = SP_Dual.addConstr(Vdual1[0] + Vdual1[2] + Vdual2[3] <=1)
    x5 = SP_Dual.addConstr(Vdual1[0] + Vdual1[1] + Vdual2[4] <=1)
    SP_Dual.Params.InfUnbdInfo = 1        #设置参数 InfUnbdInfo
    iteration = 0
    SP_Dual_obj = [9999]
    x = []
    MP.optimize()
```

Benders 分解(Benders Decomposition)

- 案例代码

```
while z.x < SP_Dual_obj[0]:                    #迭代主循环
    if iteration == 0:
        SP_Dual.setObjective(8*Vdual1[0] + 3*Vdual1[1] + 5*Vdual1[2] + \
                        8*Vdual2[0]*y[0].x + 3*Vdual2[1]*y[1].x + 5*Vdual2[2]*y[2].x + 5*Vdual2[3]*y[3].x + 3*Vdual2[4]*y[4].x, GRB.MAXIMIZE)
        SP_Dual.optimize()
        addBendersCuts(SP_Dual_obj, x)         #add Benders Cuts
        iteration = 1
    else:
        Vdual2[0].obj = 8*y[0].x               #更新dual -SP
        Vdual2[1].obj = 3*y[1].x
        Vdual2[2].obj = 5*y[2].x
        Vdual2[3].obj = 5*y[3].x
        Vdual2[4].obj = 3*y[4].x
        SP_Dual.optimize()
        addBendersCuts(SP_Dual_obj, x)         #add Benders Cuts
        iteration = iteration + 1
    MP.optimize()
```

Benders 分解(Benders Decomposition)

- 案例代码

```
    for i in range(5):                          #获取变量x的取值
        print('x[%d] = %f'%(i, x[i]))

    for i in range(5):                          #获取变量y的取值
        print('y[%d] = %d'%(i, y[i].x))

except GurobiError as e:
    print('Error code ' + str(e.errno) + ": " + str(e))

except AttributeError:
    print('Encountered an attribute error')
```

谢谢各位对 Gurobi 中文网络课程的支持。

我们会不断推出专题培训,敬请关注

www.gurobi.cn