

Column Generation & BP

研究生课程

张树柱, sz.zhang@zufe.edu.cn

信息管理与人工智能学院



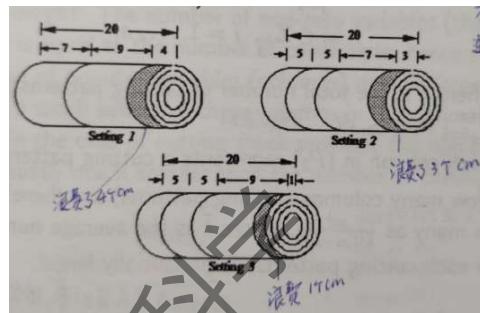
Q 运筹优化与数据科学

Outline

- Cutting stock problem
- Column generation
- Vehicle routing problem
- Branch-and-Price
- Branch-and-????

Cutting stock problem

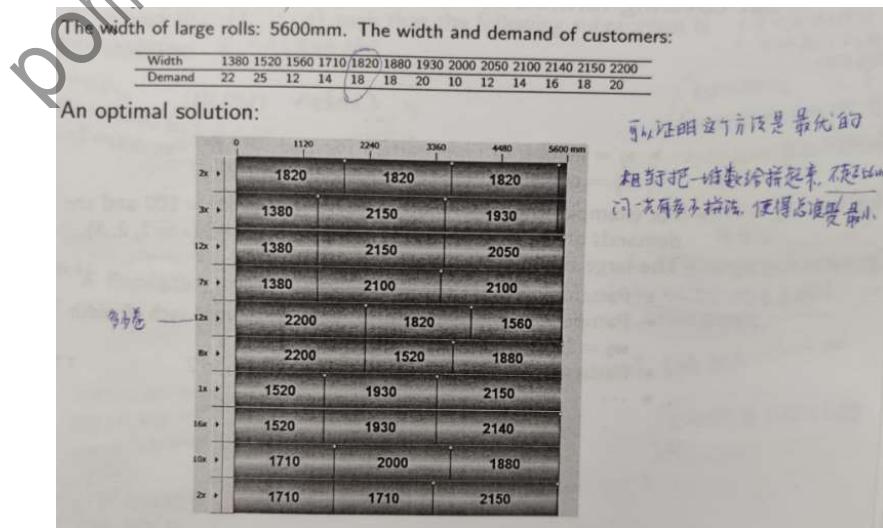
- 造纸厂里有很多大纸卷，每个大纸卷的长度是固定的。不同的顾客需要不同数量、不同规格的纸卷，需要在大的纸卷上进行切割，问怎么切割纸卷才能使得浪费最少？
- 这是个很常见的问题，比如造纸厂、铝厂、玻璃厂等等，都是生产标准尺寸的产品再进行切割。



3

Cutting-stock problem

- 纸卷长度5600mm
- 顾客*i*
- 需求规格 w_i ,
- 需求量 n_i



4

Cutting stock problem – 经典IP模型

- 假设现在一共有 K 个大纸卷，纸卷长度的 W ，顾客 i 需要长度为 w_i 的纸卷 n_i 卷。
- 问总共需要切割多少大纸卷才能满足需求，以及每个纸卷是怎么切割，才能使得总浪费最少？
- 简化处理，假设一个顾客只需要一种规格 w_i 的产品，需要 n_i 件。
 - 对于一个顾客需要多种规格的情况，把多种规格进行拆分，当作很多顾客来处理
 - 对于多个顾客需要同一种规格的情况，合并处理
- 决策变量：
 - $y_k \in \{0, 1\}, \forall k \in K$ 表示纸卷 k 是否被使用。
 - $x_i^k \in \mathbb{Z}_+, \forall i \in M, k \in K$ 表示规格 i 在纸卷 k 上切几个。

5

Cutting-stock problem – 经典IP模型

P1-classcial model	
$\min \sum_{k \in K} y_k$ $s.t.$ $\sum_{k \in K} x_i^k \geq n_i, \forall i \in M$ $\sum_{i \in M} w_i x_i^k \leq W y_k, \forall k \in K$ $x_i^k \in \mathbb{Z}_+, \forall i \in M$ $y_k \in \{0, 1\}, \forall k \in K$	K set of available rolls M set of customers $y_k = 1$ if roll k is cut, 0 otherwise x_i^k : number of times item i is cut on roll k 第一个约束是顾客的 demand 约束 第二个约束是纸卷的 width 约束

6

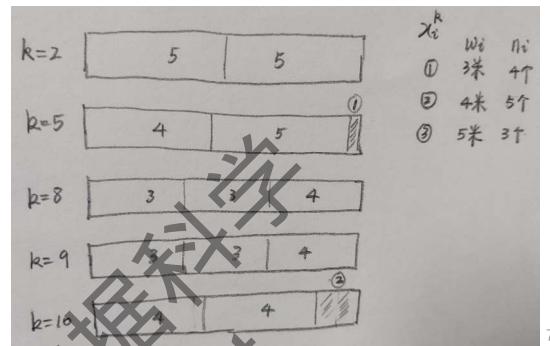
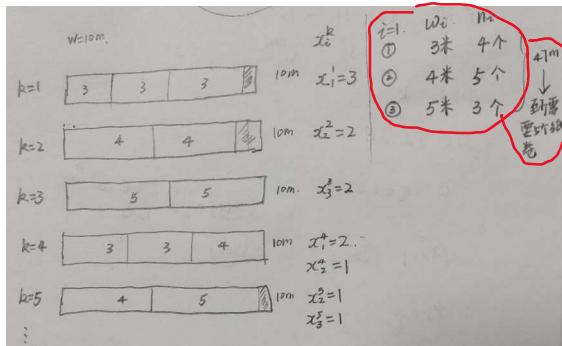
Cutting stock problem – 经典IP模型

- 假设10个纸卷，每个纸卷10米长
- 顾客需求3米规格的4个
- 4米规格的5个，5米规格的3个

```

status = Optimal
status = Optimal
objective value = 5.0
solving time = 0.01500000013969839
第2个纸卷被切割了,      y2=1.00,          切割方式为[-0.0, -0.0, 2.0]
第5个纸卷被切割了,      y5=1.00,          切割方式为[-0.0, 1.0, 1.0]
第8个纸卷被切割了,      y8=1.00,          切割方式为[2.0, 1.0, -0.0]
第9个纸卷被切割了,      y9=1.00,          切割方式为[2.0, 1.0, -0.0]
第10个纸卷被切割了,     y10=1.00,         切割方式为[-0.0, 2.0, -0.0]

```



Cutting-stock problem – 经典IP模型

- 这个模型是Kantorovich提出的，前苏联数学家，研究投入产出的，1975年与Koopman一起获得了诺贝尔经济学奖。
 - 这个模型中包含了所有的信息，需要多少卷，以及每个卷是怎么切的。**
- 这个模型的表达看起来很好，但是实际求解时不管是从计算的角度还是从理论研究的角度来看，效率都很低。
- 规模小的时候还勉强能算，规模稍微大一点，基本上就算不出来了。
- 比如有100个纸卷，20种规格要求时，CPLEX就需要计算很多天，甚至不一定能保证找到最优解。
- CPLEX解整数规划一般是B&C为主，加一些其他的启发式技巧。所以在算法方面是没有问题的，那一定是模型上有问题。

Cutting stock problem – 经典IP模型

- 这个模型之所以有问题，其中一个主要原因就是这个问题的线性松弛的界很差。

- 这个问题的线性松弛的界其实是

$$Z^{LP} = \sum_{k \in K} y_k \geq \sum_{k \in K} \sum_{i=1}^m \frac{w_i x_i^k}{W} = \sum_{i=1}^m w_i \sum_{k \in K} \frac{x_i^k}{W} = \sum_{i=1}^m w_i \frac{\sum_{k \in K} x_i^k}{W} \geq \sum_{i=1}^m w_i n_i$$

P1-classcial model

$$\begin{aligned} & \min \sum_{k \in K} y_k \\ \text{s.t.} \quad & \sum_{k \in K} x_i^k \geq n_i, \forall i \in M \\ & \sum_{i \in M} w_i x_i^k \leq W y_k, \forall k \in K \\ & x_i^k \in Z_+, \forall i \in M \\ & y_k \in \{0, 1\}, \forall k \in K \end{aligned}$$

- 中间的推理过程分别用到了P1的两个约束，即

$$\begin{aligned} \sum_{i=1}^m w_i x_i^k \leq W y_k \Rightarrow y_k \geq \frac{1}{W} \sum_{i=1}^m w_i x_i^k \Rightarrow \sum_{k \in K} y_k \geq \sum_{k \in K} \sum_{i=1}^m \frac{w_i x_i^k}{W} \\ \sum_{k \in K} x_i^k \geq n_i \end{aligned}$$

- 套用前面的例子， $Z^{LP} \geq 4.7$

- B&B或者B&C等都是建立在LP松弛的基础上，如果LP松弛的效果很差（松弛后的下界与最优解相差很大），那么B&B或者B&C的求解效果就会很差。

9

Cutting-stock problem – 经典IP模型

$$Z^{LP} \geq \sum_{i=1}^m \frac{w_i n_i}{W}$$

假设10个纸卷，每个纸卷10米长

顾客需求3米规格的4个

4米规格的5个，5米规格的3个

```

status = Optimal
status = Optimal
objective value = 5.0
solving time = 0.015000000013969839
第2个纸卷被切割了,      y2=1.00,          切割方式为[-0.0, -0.0, 2.0]
第5个纸卷被切割了,      y5=1.00,          切割方式为[-0.0, 1.0, 1.0]
第8个纸卷被切割了,      y8=1.00,          切割方式为[2.0, 1.0, -0.0]
第9个纸卷被切割了,      y9=1.00,          切割方式为[2.0, 1.0, -0.0]
第10个纸卷被切割了,     y10=1.00,         切割方式为[-0.0, 2.0, -0.0]

status = Optimal
status = Optimal
objective value = 4.7
solving time = 0.0
第1个纸卷被切割了,      y1=0.70,          切割方式为[2.3333333333333335, 0.0, 0.0]
第3个纸卷被切割了,     y3=1.00,          切割方式为[0.0, 2.5, 0.0]
第4个纸卷被切割了,     y4=1.00,          切割方式为[0.0, 1.25, 1.0]
第9个纸卷被切割了,     y9=1.00,          切割方式为[0.0, 0.0, 2.0]
第10个纸卷被切割了,    y10=1.00,         切割方式为[1.6666666666666665, 1.25, 0.0]
```

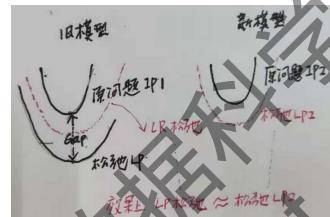
10

Cutting stock problem – 经典IP模型

- 按照之前的思路，用LR来当下界，效果不会弱于LP松弛，但是问题是CPLEX不会这么处理，如果自己手动编写程序，中间过程会很麻烦，因为涉及到解对偶问题，需要用到对偶搜索，次梯度等等。
- 所以我们换个角度去看待这个问题，尝试构造另外一個新模型，且新模型的LP松弛在效果上等价于原模型的拉格朗日松弛。**这样新模型的LP松弛就比旧模型的LP松弛要好。

- 基本思路是**

- 对于原IP问题模型，如果直接做LP松弛，容易实现，但是效果很差。如果做LR松弛，理论上讲效果不会弱于LP松弛，但是很难实现。**LR松弛更多的是一种指导原则。**
- 构造新IP问题模型，特点在于新的IP问题的LP松弛在效果上等价于原IP问题的LR松弛。**这样就既容易实现，又效果好。**



11

Cutting-stock problem – 集合覆盖模型

对于集合覆盖模型，首先有个pattern的概念，一个 pattern 就表示一种切法。[←]

比如对于宽度 10m 的纸卷，需求规格是 3 米，4 米，5 米，可能有以下 6 种切法[←]

- P1: 3 米, 3 米, 3 米[←] 剩余 1 米[←]
- P2: 3 米, 3 米, 4 米[←]
- P3: 3 米, 5 米[←] 剩余 2 米[←]
- P4: 4 米, 4 米[←] 剩余 2 米[←]
- P5: 4 米, 5 米[←] 剩余 1 米[←]
- P6: 5 米, 5 米[←]

注意当宽度很大，或者需求规格很多时，可能会有很多很多 pattern，几乎是不可能穷举出来的。[←]

[←]

这个其实是个组合问题，对于三种规格而言，至少会有 $c_3^1 + c_3^2 + c_3^3 = 7$.[←]

c_3^1 是说一个纸卷只切一种规格[←]

c_3^2 是说一个纸卷切两种规格[←]

c_3^3 是说一个纸卷切三种规格[←]

其实对于上面的三种情况，每种都可以再细分有很多种情况，除了每种规格之外，还有切几个的问题。

12

原来的切割问题其实就相当于把不同的数（需求规格）组合起来，不能超过规定的长度，**问一共有多少种组合方式**。这个数基本上也是不可能穷举出来的。[↓]

但是我们先假设这个 pattern 是可以全部穷举出来的，**至少可以先穷举出来一部分**。[↓]

比如，我们知道至少可以有这三种切割方式。[↓]

- P1: 3 米, 3 米, 3 米 剩余 1 米[↓]
- P4: 4 米, 4 米 剩余 2 米[↓]
- P6: 5 米, 5 米[↓]

我们定义如下的符号[↓]

- $x_j \in Z_+$ 表示 pattern j 被使用的次数[↓]
- $a_{ij} \in Z_+$ 表示 pattern j 中包含几个规格 i，也就是包含几个 w_i [↓]

[↓]

这里 x_j 是一个变量， a_{ij} 其实是一个参数[↓]

给定一个 pattern j，我们就知道这个 pattern 中具体包含什么信息，也就是 a_{ij} 的信息，而且这里有个隐含的条件就是 $\sum_{i=1}^m w_i a_{ij} \leq W$. 一个 pattern 中包含的 w_i 之和不能超过 W.[↓]

[↓]

结合上面的例子，我们有[↓]

- P1: 3 米, 3 米, 3 米 剩余 1 米 $\rightarrow a_{11} = 3, a_{21} = 0, a_{31} = 0$ [↓]
- P2: 3 米, 3 米, 4 米 $\rightarrow a_{12} = 2, a_{22} = 1, a_{32} = 0$ [↓]
- P3: 3 米, 5 米 剩余 2 米 $\rightarrow a_{13} = 1, a_{23} = 0, a_{33} = 1$ [↓]
- P4: 4 米, 4 米 剩余 2 米 $\rightarrow a_{14} = 0, a_{24} = 2, a_{34} = 0$ [↓]
- P5: 4 米, 5 米 剩余 1 米 $\rightarrow a_{15} = 0, a_{25} = 1, a_{35} = 1$ [↓]
- P6: 5 米, 5 米 $\rightarrow a_{16} = 0, a_{26} = 0, a_{36} = 2$ [↓]

13

Cutting-stock problem – 集合覆盖模型

结合前面的例子[↓]

对于每个 pattern 而言[↓]

- P1: 3 米, 3 米, 3 米 剩余 1 米 $\rightarrow a_{11} = 3, a_{21} = 0, a_{31} = 0$ [↓]
- P2: 3 米, 3 米, 4 米 $\rightarrow a_{12} = 2, a_{22} = 1, a_{32} = 0$ [↓]
- P3: 3 米, 5 米 剩余 2 米 $\rightarrow a_{13} = 1, a_{23} = 0, a_{33} = 1$ [↓]
- P4: 4 米, 4 米 剩余 2 米 $\rightarrow a_{14} = 0, a_{24} = 2, a_{34} = 0$ [↓]
- P5: 4 米, 5 米 剩余 1 米 $\rightarrow a_{15} = 0, a_{25} = 1, a_{35} = 1$ [↓]
- P6: 5 米, 5 米 $\rightarrow a_{16} = 0, a_{26} = 0, a_{36} = 2$ [↓]

[↓]

只有一个约束条件，即对于每个顾客而言，顾客需求必须被满足[↓]

$$\sum_{j=1}^n a_{ij} x_j \geq n_i, \forall i = 1, \dots, m$$

顾客 1: $a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{15}x_5 + a_{16}x_6 \geq n_1$ [↓]

顾客 2: $a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + a_{25}x_5 + a_{26}x_6 \geq n_2$ [↓]

顾客 3: $a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 + a_{35}x_5 + a_{36}x_6 \geq n_3$ [↓]

$$\begin{pmatrix} 3 \\ 0 \end{pmatrix}x_1 + \begin{pmatrix} 2 \\ 1 \end{pmatrix}x_2 + \begin{pmatrix} 1 \\ 0 \end{pmatrix}x_3 + \begin{pmatrix} 0 \\ 2 \end{pmatrix}x_4 + \begin{pmatrix} 0 \\ 1 \end{pmatrix}x_5 + \begin{pmatrix} 0 \\ 2 \end{pmatrix}x_6 \geq \begin{pmatrix} n_1 \\ n_3 \end{pmatrix}$$

每一列对应着一个割法。[↓]

总共可能有多少列（也就是多少种割法），这个数字很大，大约为 $c(m, k) = \frac{m!}{k!(m-k)!}$ ， k 是一个割法种可能包含的规格数的平均数。这个数字因为涉及到阶乘运算，所以是个很大的数字，Exponentially large.[↓]

x_1	x_2	x_3	x_4	x_5	x_6	顾客
P_1	a_{11}	a_{21}	a_{31}			n_1
P_2	a_{12}	a_{22}	a_{32}			n_2
P_3	a_{13}	a_{23}	a_{33}			n_3
\vdots						
P_m	a_{1m}	a_{2m}	a_{3m}			n_m
						顾客

顾客
m是有限的
n是很大的数

14

Cutting stock problem – 集合覆盖模型

- 这个问题换个角度去理解，假设我们知道所有可能的割法，也就是所有的pattern，所有可能的割法肯定是有有限的，我们只需要从这些所有的割法中选择一些出来，能够覆盖我们的需求即可。
 - 例如，对于顾客*i*而言，只要 $\sum_{j=1}^n a_{ij}x_j \geq n_i$ 即可，就是从所有能够给顾客*i*提供服务的割法 x_j 中，选择一部分出来，满足顾客*i*的需求即可。
- 注意这里是假设我们知道所有可能的割法*n*，实际上我们是不知道的，因为这个数字可能大到几乎不可能穷举。
- 这个观念跟传统的集合覆盖模型非常类似，传统的集合覆盖模型是说需要建立一些急救中心来服务一定的区域，要求能够覆盖所有区域，且使得建造总成本最小。对每个区域而言，有一个子集，子集中的元素选择要能覆盖区域的需求。
- 两者都是从一个给定的集合中选择一部分元素出来，这部分元素要在能够满足我们需求的前提下，元素数量越少越好。

15

Set covering/partitioning/packing model

集合覆盖、划分、包装模型都是针对选址问题而言的，比如要建立救护站，emergency medical service, EMS 选址问题。这里有两点要注意的

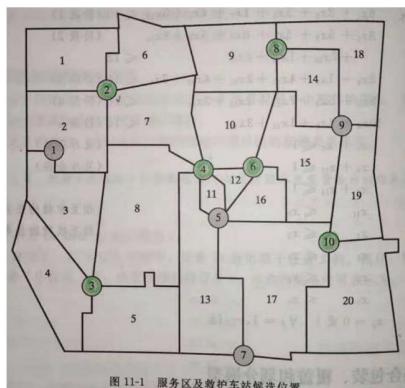
(1) 选址问题都是在给定的一组候选地点列表中选择一些点来进行选址，而不是确定新的坐标点。

(2) 每个备选地址与需要服务的区域之间的关联是确定的。

结合下面的例子，一共 20 个服务区域，10 个候选点。

比如候选点 2 就可以为区域 1, 2, 6, 7 提供服务。

相当于有个矩阵 $A = [a_{ij}]_{m \times n}$, $m = 20$ 个服务区域, $n = 10$ 个候选点, $a_{ij}, i = 1, \dots, m; j = 1, \dots, n$ 是给定的。

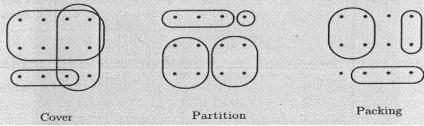


候选点	服务区域									
	1	2	3	4	5	6	7	8	9	10
1	0	1	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0
3	1	0	1	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0
7	0	1	0	1	0	0	0	0	0	0
8	0	0	1	1	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	0
10	0	0	0	1	0	1	0	0	0	0
11	0	0	0	1	1	0	0	0	0	0
12	0	0	0	1	1	1	0	0	0	0
13	0	0	0	1	1	0	1	0	0	0
14	0	0	0	0	0	0	0	0	1	1
15	0	0	0	0	0	1	0	0	1	0
16	0	0	0	0	0	1	1	0	0	0
17	0	0	0	0	1	0	1	0	0	1
18	0	0	0	0	0	0	0	1	1	0
19	0	0	0	0	0	0	0	0	1	1
20	0	0	0	0	0	0	0	0	0	1

16

从集合覆盖、划分、包装的角度来看

有至少一个站点 \Leftrightarrow 覆盖 \Leftrightarrow 集合中至少有一个元素为 1 $\Leftrightarrow \sum_j x_j \geq 1$
 恰好有一个站点 \Leftrightarrow 划分 \Leftrightarrow 集合中恰好有一个元素为 1 $\Leftrightarrow \sum_j x_j = 1$
 最多有一个站点 \Leftrightarrow 包装 \Leftrightarrow 集合中最多有一个元素为 1 $\Leftrightarrow \sum_j x_j \leq 1$



对每个服务区域而言，都有一些候选站点给它提供服务，这些候选站点构成一个“集合”。

这里的“集合覆盖”的概念，是指对每个服务区域而言，比如，

对服务区域 1 而言，它的集合是 $\{x_2\}$ $x_2 \geq 1$

对服务区域 2 而言，它的集合是 $\{x_1, x_2\}$ $x_1 + x_2 \geq 1$

对服务区域 3 而言，它的集合是 $\{x_1, x_3\}$ $x_1 + x_3 \geq 1$

...

对服务区域 7 而言，它的集合是 $\{x_2, x_4\}$ $x_2 + x_4 \geq 1$

...

对服务区域 12 而言，它的集合就是 $\{x_4, x_5, x_6\}$ $x_4 + x_5 + x_6 \geq 1$

...



服务区域	1	2	3	4	5	6	7	8	9	10
1	0	1			0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0
3	1	0	1	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0
7	0	1	0	1	0	0	0	0	0	0
8	0	0	1	1	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	0
10	0	0	0	1	0	1	0	0	0	0
11	0	0	0	1	1	0	0	0	0	0
12	0	0	0	1	1	1	0	0	0	0
13	0	0	0	1	1	0	1	0	0	0
14	0	0	0	0	0	0	0	1	1	0
15	0	0	0	0	0	1	0	0	1	0
16	0	0	0	0	1	1	0	0	0	0
17	0	0	0	0	1	0	1	0	0	1
18	0	0	0	0	0	0	0	1	1	0
19	0	0	0	0	0	0	0	0	1	1
20	0	0	0	0	0	0	0	0	0	1

集合覆盖的意思是说约束条件的形式是集合覆盖的样子，从一个集合中选择一些元素来覆盖特定的需求。

一般 J_{all} 表示所有可能的列构成的集合， $J \subseteq J_{all}$ 表示当前受限主问题包含的列的集合。

x_j, c_j 分别表示决策变量与其对应目标函数系数， aJ 表示约束条件系数矩阵的第 J 列， b 是右端项向量。

17

Cutting-stock problem – 集合覆盖模型

- 这个模型是 Gilmore and Gomory 提出的， n 是所有可能的割法 pattern，注意 n 可能是个非常大的数字。
- 而且每个可能的割法都默认隐含了宽度约束。有了割法 x_j ，也就有了 a_{ij} ，默认包含了

$$\sum_{i=1}^m w_i a_{ij} \leq W, j = 1, \dots, n$$

P2-SC model	Set covering model	
$\min \sum_{j=1}^n x_j$ $s.t.$ $\sum_{j=1}^n a_{ij} x_j \geq n_i, \forall i = 1, \dots, m$ $x_j \in \mathbb{Z}_+, \forall j = 1, \dots, n$	$\min \sum_{j=1}^n c_j x_j$ $s.t.$ $\sum_{j=1}^n a_{ij} x_j \geq 1, \forall i = 1, \dots, m$ $x_j \in \{0, 1\}, \forall j = 1, 2, \dots, n$	$i \in M = \{1, \dots, m\}$: region $j \in N = \{1, \dots, n\}$: service center c_j : fixed set-up cost $a_{ij} \in \{0, 1\}$ correlation

18

Cutting stock problem – 集合覆盖模型

P1-Classical model	P2-Set covering model
$\min \sum_{k \in K} y_k$ <p>s.t.</p> $\sum_{k \in K} x_i^k \geq n_i, \forall i \in M$ $\sum_{i \in M} w_i x_i^k \leq W \cdot y_k, \forall k \in K$ $y_k \in \{0, 1\}, \forall k \in K$ $x_i^k \in Z_+, \forall i \in M, k \in K$	$\min \sum_{j \in N} x_j$ <p>s.t.</p> $\sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M$ $\sum_{i \in M} w_i a_{ij} \leq W, \forall j \in N$ $x_j \in Z_+, \forall j \in N$
一共 K 个纸卷可用 一共 M 个顾客需求，每个顾客需求一种规格 $y_k \in \{0, 1\}$ 表示纸卷 k 是否被使用 $x_i^k \in Z_+$ 表示在纸卷 k 上切了几个规格 w_i	这里 N 表示所有可能的 pattern 的集合，有 n 个元素， n 是个很大的数字 x_j 是整数，表示切法 j 一共用了几次。 切法 j 的具体信息是通过 a_{ij} 表示出来的，是 x_j 对应的系数矩阵列。 第二个约束条件是个隐含约束，其中不包含决策变量，任何一个切法默认规定了不能超过纸卷的长度。

19

从 P2 到 LPM 问题，即对集合覆盖模型做线性松弛

P2-IP	Linear MP, LMP	Dual of LMP
$\min \sum_{j=1}^n x_j$ <p>s.t.</p> $\sum_{j=1}^n a_{ij} x_j \geq n_i, i = 1, \dots, m$ $x_j \in Z_+, j = 1, \dots, n$	$\min \sum_{j=1}^n x_j$ <p>s.t.</p> $\sum_{j=1}^n a_{ij} x_j \geq n_i, i = 1, \dots, m$ $x_j \in R_+, j = 1, \dots, n$	$\max \sum_{i=1}^m n_i \pi_i$ <p>s.t.</p> $\sum_{i=1}^m a_{ij} \pi_i \leq 1, j = 1, \dots, n$ $\pi_i \geq 0, i = 1, \dots, m$

这里只是把整数变量 $x_j \in Z_+$ 松弛成实数变量 $x_j \in R_+$ ，松弛后的问题叫做 LP 主问题 (LP master problem, LMP)。我们希望通过解这个 LMP 问题，得到一个分数解，然后通过向上取整操作，再得到一些整数可行解。注意这个向上取整的操作其实相当于启发式的方法，并不能保证是整数最优解，但是至少可以得到一个整数可行解。

实际上对于这个 LMP 问题，虽然这是个 LP 问题，但是实际上我们也不能直接求解，因为列数 n 太多，我们根本不可能把这个 LPM 问题完整地写出来。从集合覆盖的角度去看， n 太多，其实就是集合中的元素太多，不可能都一一写出来。所以我们要寻找一种方法，可以让我们在不需要把每一列都写出来的前提下就可以求解这个 LMP 问题。

注意这里共包含两个步骤，一是求解这个 LPM 问题，二是把 LPM 问题的解转变为 P2 的解。这两个步骤都是比较困难的，好在都有办法解决。

求解 LPM 问题是用列生成，从 LPM 到 P2 是用启发式（或者其他 branching 的技巧）。

20

P2-IP→LPM→Restricted LPM (RLPM)→Dual of RLPM

- 从 LPM 到 IP 的过程是启发式的，类似于向上取整（因为约束条件是 \geq ）。
- 从 LPM 到 RLPM 是先固定一部分列，得到一个受限的 LPM。
- 求解 LPM 整体上是用列生成的思想。

求解这个 LPM 问题的关键是，对于这个 LPM 问题而言，基变量的数量是固定的，等于约束条件的数量 m ，而非基变量的部分 $n-m$ 虽然可能是很大，但是我们并不需要知道所有的非基变量，事实上绝大多数非基变量都不发挥作用。**直观理解**，系数矩阵 $A_{m \times n}$, $m \ll n$ ，而决策变量 $x^T = (x_B^T, x_N^T) = (B^{-1}b, 0)$ ，我们真正需要的是 $B_{m \times m}$ 的一个矩阵，从结果上看其余的 $N_{m \times (n-m)}$ 是不发挥作用的。而构造 $B_{m \times m}$ 的过程是从 $A_{m \times n}$ 的 n 列中不断筛选的过程，直到选出来最优的 m 列为止。**具体怎么找列的过程**，是通过对偶问题来处理的。

比如在切割问题中，我们需要的规格种类往往比 pattern 数要少的多。也就是说我们并不需要知道所有的集合元素，事实上往往我们只需要很少一部分集合元素就能求解这个 LPM 问题。

$$\begin{array}{|c|c|} \hline \min c^T x & \\ \text{s.t.} & \\ Ax = b & \\ x \geq 0 & \\ \hline \end{array} \quad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}_{m \times n} = [B_{m \times m} \quad N_{m \times (n-m)}] \\ x^* = (x_B^*) - (B^{-1}b) \quad \boxed{0}$$

$$\begin{aligned} Ax = b \\ \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1m}x_m + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2m}x_m + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mm}x_m + \cdots + a_{mn}x_n = b_m \end{bmatrix} \\ \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} x_1 + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} x_2 + \begin{bmatrix} a_{13} \\ a_{23} \\ \vdots \\ a_{m3} \end{bmatrix} x_3 + \cdots + \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{mm} \end{bmatrix} x_m + \cdots + \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} x_n = b \\ A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} & a_{1,m+1} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2m} & a_{2,m+1} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} & a_{m,m+1} & \cdots & a_{mn} \end{bmatrix}_{m \times n} \end{aligned}$$

21

列生成法

- MP → LMP → RLMP → Dual of RLMP

Master problem, MP	Linear MP, LMP	Dual of LMP
$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in Z_+, \forall j \in N$	$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in R_+, \forall j \in N$	$\max \sum_{i \in M} n_i \pi_i$ s.t. $\sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in N$ $\pi_i \geq 0, \forall i \in M$

Restricted linear MP, RLMP	Dual of RLMP
$\min \sum_{j \in J_1} x_j$ s.t. $\sum_{j \in J_1} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \geq 0, \forall j \in J_1, J_1 \subseteq N$	$\max \sum_{i \in M} n_i \pi_i$ s.t. $\sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in J_1$ $\pi_i \geq 0, \forall i \in M$

22

Master problem, MP	Linear MP, LMP	Dual of LMP
$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in Z_+, \forall j \in N$	$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in R_+, \forall j \in N$	$\max \sum_{i \in M} n_i \pi_i$ s.t. $\sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in N$ $\pi_i \geq 0, \forall i \in M$
Restricted linear MP, RLMP	Dual of RLMP	
$\min \sum_{j \in J_1} x_j$ s.t. $\sum_{j \in J_1} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \geq 0, \forall j \in J_1, J_1 \subseteq N$	$\max \sum_{i \in M} n_i \pi_i$ s.t. $\sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in J_1$ $\pi_i \geq 0, \forall i \in M$	

23

列生成法

基本思路

- 注意这个LMP问题，虽然列数n太多，但是行数是固定的，共m行，从对偶的角度来看，它的对偶问题的变量个数是固定的。
- 解这个LMP问题的是从检验数是否满足最优化条件入手，而验证检验数是否满足最优化条件是从对偶可行性的角度来处理的。

Master problem, MP	Linear MP, LMP	Dual of LMP
$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in Z_+, \forall j \in N$	$\min \sum_{j \in N} x_j$ s.t. $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \in R_+, \forall j \in N$	$\max \sum_{i \in M} n_i \pi_i$ s.t. $\sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in N$ $\pi_i \geq 0, \forall i \in M$

$$\begin{bmatrix} c_1 & c_2 & c_3 & \dots \\ x_1: & x_2 & x_3 & \dots \\ \vdots & \vdots & \vdots & \vdots \\ a_{11} & a_{12} & a_{13} & \dots \\ a_{21} & a_{22} & a_{23} & \dots \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots \end{bmatrix}$$

↑ 寻找一个列满，高是寻找一列数
相当于要入基 入基要求非基变量≤0
↓ 因为偶可行性差处理

24

列生成法

$$\begin{array}{c|cc|c|c} M & \dots & \dots & & \\ \hline x_B & 1 & & & \\ & & B^{-1}N & B^{-1}b & \\ \hline & 0 & C_B^T - C_B^T B^{-1}A_j & & \end{array}$$

$$r_N^T = C_N^T - C_B^T B^{-1} N$$

$$r_j = c_j - C_B^T B^{-1} A_j$$

- 从单纯形表的角度来看，非基变量的检验数为， $r_N^T = c_N^T - C_B^T B^{-1} N$
 - 对于其中单个非基变量 x_j 而言是， $r_j = c_j - C_B^T B^{-1} A_j$ ，其中 A_j 是 x_j 在 A 中对应的列。
 - 如果存在 $r_j = c_j - C_B^T B^{-1} A_j < 0$ ，非基变量 x_j 检验数为负，说明当前解还可以改进。
 - 反之，如果所有的 $r_j = c_j - C_B^T B^{-1} A_j \geq 0$ ，则说明当前解已经是最优解了，且此时 $\pi^T = C_B^T B^{-1}$ 也是对偶最优解。
- 那么现在的问题就是，怎么验证 $r_N^T = c_N^T - C_B^T B^{-1} N \geq 0$ ，这里就需要从对偶问题的角度出发。
- 注意原始单纯形法的基本思路是原问题在保持可行解的前提下，去寻找最优解，在这个过程中，对偶问题对应的解也在移动，原问题最优时，对偶问题可行且最优。
 - 就是说原问题找到最优解时，对偶问题恰好是可行解，也就是说 $r_N^T = c_N^T - C_B^T B^{-1} N \geq 0$ 等价于 $c - A^T \pi \geq 0$ 。
 - 所以换个角度来看，可以用对偶约束来判断原问题的解是否是最优解。这个其实是对偶单纯形法的内容。

25

列生成法

非基变量检验数 ≥ 0 与对偶问题可行的对应关系

这里的证明技巧是直接定义了一个 π 的形式，即 $\pi^T = C_B^T B^{-1}$, $\pi = (B^{-1})^T c_B = (B^T)^{-1} c_B$ 这个定义是最关键的。因为按照这个定义，一些条件都会理所当然的满足。比如对偶可行性

$$c - A^T \pi = \begin{pmatrix} C_B \\ C_N \end{pmatrix} - \begin{pmatrix} B^T \\ N^T \end{pmatrix} \pi = \begin{pmatrix} C_B - B^T (B^T)^{-1} C_B \\ C_N - N^T (B^T)^{-1} C_B \end{pmatrix} = \begin{pmatrix} 0 \\ r_N \end{pmatrix} \geq 0$$

$$r_N = C_N - N^T (B^T)^{-1} C_B$$

$r_N^T = C_N^T - C_B^T B^{-1} N$ 就是非基变量的检验数。

当原问题 x 为 optimal 时， $r_N \geq 0$ ，这样就证明了 $A^T \pi \leq c$ ，即此时的 π 满足对偶可行性。

按照这个定义，在原始单纯形表中，非基变量的检验数 ≥ 0 ，就对应着对偶问题的可行解。这是个很巧妙的东西。直观理解，就是单纯形表中的非基变量检验数都 ≥ 0 时，意味着原问题是 optimal 的，同时对偶问题是 feasible 的。所以可以用对偶问题的解可行来判断原问题的非基变量检验数 ≥ 0 。

26

列生成法的详细说明

对于原来的 LMP 问题，因为原问题的行数 m 是固定的，我们可以先选择 m 列，构造一个确定的 LP 问题，称为受限的 LP 问题(RLMP)。

然后求解这个 RLMP 问题，因为是一个确定的 LP 问题，我们可以直接得到 RLMP 的最优解，和它的对偶问题的最优解。 $\mathbf{x}^* = \begin{pmatrix} \mathbf{x}_B \\ \mathbf{0} \end{pmatrix} = (\mathbf{B}^{-1}\mathbf{b})$, $(\boldsymbol{\pi})^T = \mathbf{c}_B^T \mathbf{B}^{-1}$

在 RLMP 问题上每增加一个变量，就是一种新的割法，相当于约束矩阵增加一列。

原问题增加一列，对偶问题增加一行。

原问题上增加一列能否起到改善作用？去检查这一列对应非基变量的检验数。

原问题新增列的检验数为 $\eta_j = c_j - \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j = c_j - \boldsymbol{\pi}^T \mathbf{A}_j$.

原问题上新增加的这一列的检验数可以通过对偶问题的行计算得到。

因为 n 的数量很大，理论上非基变量 $n-m$ 可以有很多，假设我们可以添加很多列，那么怎么去只保留那些起作用的列呢？起作用的列就是检验数 < 0 的列（要入基的列）。

这里我们用对偶解来验证新加入的某列 x_j 的检验数 $c_j - \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j = c_j - \boldsymbol{\pi}^T \mathbf{A}_j$.

如果能找到使得 $c_j - \boldsymbol{\pi}^T \mathbf{A}_j < 0$ 的 \mathbf{A}_j ，就说明 \mathbf{x}_B 是可以被改进的。

这时把 \mathbf{A}_j 这一列加入到原问题中，相当于原问题增加了一列。

这样每次加入一列就得到一个新问题，然后再求解这个新问题的对偶问题，继续验证检验数，看看是否可以找到其他的使得检验数小于 0 的列加入到原问题中，这样持续进行，直到找不到能使检验数小于 0 的列为止。这就是列生成法。

C_1	C_2	C_3	\dots	C_m	\dots	C_j
x_1	x_2	x_3	\dots	x_m	\dots	x_j

$$\left[\begin{array}{cccccc} a_{11} & a_{12} & a_{13} & \dots & a_{1m} & \dots \\ a_{21} & a_{22} & a_{23} & \dots & a_{2m} & \dots \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mm} & \dots \end{array} \right]_{m \times m}$$

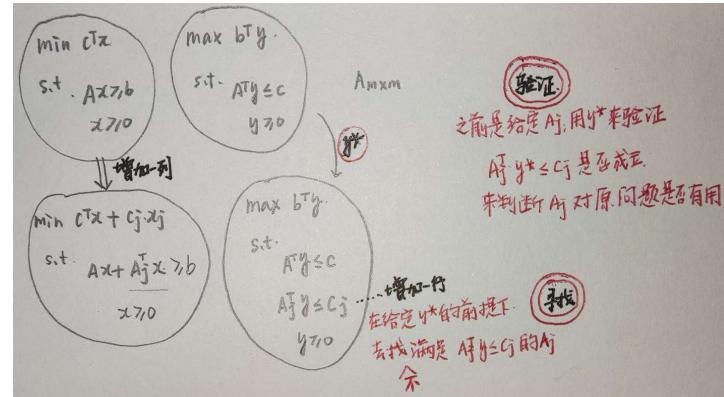
$$y_j = c_j - \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j = c_j - \boldsymbol{\pi}^T \mathbf{A}_j$$

再细致一点

- 首先这里在说给定一个新的割法 x_j 对应着一列 A_j ，这里只是在验证这一列的加入会不会对 RLPM 问题的最优解产生影响。还没有讲到怎么去寻找 A_j 。
- 其次是这个验证过程为什么是合理的？看下面的图，RLPM 上增加一列，Dual 上增加一行，然后用 $\boldsymbol{\pi}^*$ 来验证新增加的这一行是否可行，如果可行，表示原问题新增加的列的 $\eta_j \geq 0$ 。反过来，我们是希望用 $\boldsymbol{\pi}^*$ 来寻找 $\eta_j < 0$ 的列。

这里有几点需要说明

- 首先，对于给定 m 行 m 列的线性规划问题 RLPM，我们用 CPLEX 来解时，是可以同时得到原问题和对偶问题的解的，所以不需要我们再去写出对偶问题来求解。
- 其次是初始的 m 列如何获得，这里可以简单处理，假设一种割法只考虑一个顾客的需求，也就是只切一种规格，这样很容易就可以得到 m 个 pattern，也就 m 列。



- 接下来是如何寻找新的有效割法，即寻找有效的列。
- 这一列的检验数要<0，这样才能非基变量进基，并改善原问题的解
- (1) 其实是在给定对偶解的前提下，去寻找使 $A_j^T \pi \leq c_j$ 不成立的 A_j 。
 - (2) 注意 A_j 列对应的是一个割法，所以其实是一列非负数。如果把 A_j 当作一个变量 y ，要满足 $\sum_i w_i y_i \leq W$ 。
 - (3) 这时我们需要构造一个辅助问题来帮助我们找到 A_j ，使得 $c_j - \pi^T A_j < 0$ 或者是证明不存在这样的 A_j 。

总的来说，就是我们需要构造一个子问题，这个子问题能够在保持当前受限子问题对偶可行的前提下，寻找到一种新的割法 A_j ，使得原问题变得更好。

我们把 A_j 当作决策变量 y ，把 $c_j - \pi^T y$ 当作目标函数，把 $\sum_i w_i y_i \leq W$ 当作约束条件，得到下面的子问题。

29

列生成法

寻找 A_j 的过程就等价于解下面的辅助问题

如果 $\min(c_j - \pi^T y)$ 都大于 0 的话，说明不存在 A_j 能够使得 $c_j - \pi^T A_j < 0$ 。

否则，我们就能找到一个 y 满足， $c_j - \pi^T y < 0$ ，检验数 $r_j = c_j - c_B^T B^{-1} y < 0$ ，即满足检验数小于 0 的要求。

而上面的 min 问题等价于解后面的 max 问题，这个 max 问题包含一个约束条件，目标函数求最大，实际上这是个背包问题的变形。而背包问题虽然是个难题，却是可以用动态规划来求解的，其复杂度为 $O(nb)$ ，是个伪多项式时间算法。

30

子问题	子问题转化	去掉常数项 c_j	背包问题
$\min(c_j - \pi^T y)$ s.t. $\sum_{i=1}^m w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$	$c_j - \max \pi^T y$ s.t. $\sum_{i=1}^m w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$	$\max \pi^T y$ s.t. $\sum_{i=1}^m w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$	$\max \sum_{j=1}^n c_j x_j$ s.t. $\sum_{j=1}^n a_j x_j \leq b$ $x_j \in Z_+, j = 1, \dots, n$

下面是列生成法的具体算法说明

IP → LMP → RLMP → Dual of RLMP

然后使用 Dual 来构造子问题，不断修正 RLMP，直到 RLMP 不能改进为止，此时 RLMP 的解等价于 LMP 的解。之后是从 LMP 到 MP 的过程，可以采用某种取整的启发式方法。

第一步，构造受限的 LMP 问题(RLMP)，并求解，得到 RLMP 原问题与对偶问题的最优解。

The main idea of column generation is to start with a small subset $P \subset \{1, \dots, n\}$ such that the following subproblem is feasible. (RLPM).

首先假设有个集合 $P \subset N = \{1, 2, \dots, n\}$ ，然后定义如下的问题，称为受限的 LPM 问题 (restricted linear master problem, RLMP)。

对于切割问题而言，选择 m 列的过程，相当于预先选择 m 种切法，因为有 m 个顾客需求规格，可以直接设置一种切割方法满足一种顾客需求规格，就可以很简单地得到 m 种切割方法。

LMP	RLMP	Dual of RLMP
$\min \sum_{j \in N} x_j$ <i>s.t.</i> $\sum_{j \in N} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \geq 0, \forall j \in N$	$\min \sum_{j \in P} x_j$ <i>s.t.</i> $\sum_{j \in P} a_{ij}x_j \geq n_i, \forall i \in M$ $x_j \geq 0, \forall j \in P, P \subseteq N$	$\max \sum_{i \in M} n_i \pi_i$ <i>s.t.</i> $\sum_{i \in M} a_{ij} \pi_i \leq 1, j \in P$ $\pi_i \geq 0, \forall i \in M$

31

第二步，生成列

接下来是从要从 $\{1, 2, \dots, n\} \setminus P$ 中寻找新的列(cut pattern)来改善当前 RLMP 的最优解。

因为 RLMP 对偶问题的解是已经的，假设是 π 。

RLMP 新加入一列 $j \in \{1, 2, \dots, n\} \setminus P$ ，其检验数就可以写成 $r_j = c_j - c_B^T B^{-1} A_j$

RLMP 新加入一列，相当于其对偶问题新加入一行

$$r_j = c_j - c_B^T B^{-1} A_j = c_j - \pi^T A_j = 1 - \sum_{i=1}^m a_{ij} \pi_i$$

直观理解上是在保持对偶可行的前提下，看看能不能找到一种割法使得原问题进行改进。

我们需要尝试去寻找检验数最小的非基变量

$$\min \left\{ 1 - \sum_{i=1}^m a_{ij} \pi_i \mid j \in \{1, 2, \dots, n\} \setminus P \right\}$$

理论上讲，我们需要检查所有在 $N \setminus P$ 中的列，但是这样做是不实际的，同样是因为我们不可能把所有的 n 都穷举出来。所以我们先只是寻找一列满足

子问题	背包问题形式	
$\min 1 - \sum_{i=1}^m y_i \pi_i = 1 - \max \sum_{i=1}^m y_i \pi_i$ <i>s.t.</i> $\sum_{i=1}^m w_i y_i \leq W$ $y_i \in Z_+, i = 1, \dots, m$	$\max \sum_{i=1}^m y_i \pi_i$ <i>s.t.</i> $\sum_{i=1}^m w_i y_i \leq W$ $y_i \in Z_+, i = 1, \dots, m$	$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix} = \mathbf{A}_j$, 是一种割法。 约束条件强制这个割法必须是可行割。 $\{\min -x\} = \{-\max x\}$

这个子问题其实是个整数背包问题 (an easy NP-hard problem)，整数背包问题是 NP-hard 问题，但是可以用动态规划来解，整体上这个子问题的复杂度为 $O(mW)$ 。

32

列生成算法伪码描述

Column generation algorithm	
Start with initial columns of \mathbf{A} of LPM.	
For instance, use the simple pattern to cut a large roll into $[W/w_i]$ small rolls of width w_i , \mathbf{A} is a diagonal matrix.	$\mathbf{A}_{m \times m} = \begin{bmatrix} W/w_1 & \cdots & \square \\ \vdots & W/w_i & \vdots \\ \square & \cdots & W/w_m \end{bmatrix}$
Repeat	
1. Solve the restricted linear master problem (RLMP). Let $\boldsymbol{\pi}$ be the optimal multiplier ($\boldsymbol{\pi} = \mathbf{c}_B^T \mathbf{B}^{-1}$). 2. Identify a new column by solving the knapsack subproblem with optimal value κ . 3. Add the new column to RLMP	$\kappa = \min 1 - \sum_{i \in M} y_i \pi_i$ s.t. $\sum_{i \in M} w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$
Until $\kappa \geq 0$	

到目前为止，还只是解决了 LMP 问题，这还是个线性规划问题，并不是最开始的整数规划问题。

用启发式的方法，向上取整，从 LMP 到 P2，约束仍然成立，这时才是个整数解。

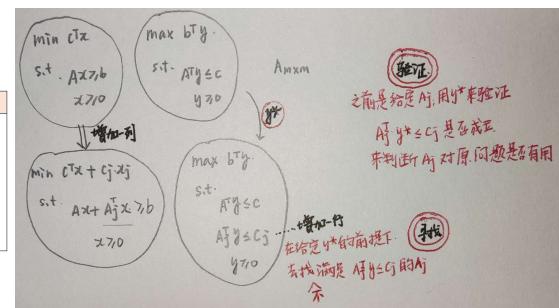
33

Specialty of cutting stock problem

对于切割问题而言，其特殊之处

- (1) 之所以能用列生成法，核心在于集合覆盖模型的对偶问题只有一个约束条件，所以可以用这个对偶问题的约束条件来验证与寻找原问题的新列。
- (2) 这个切割问题的目标函数中的决策变量的系数都为 1，这是个很重要的点，对应着子问题构造时，恰好是 $r_j = c_j - \mathbf{c}_B^T \mathbf{B}^{-1} \mathbf{A}_j = 1 - \boldsymbol{\pi}^T \mathbf{A}_j$ 的形式，可以忽略常数项的部分，这个点在 VRP 中就不具备。
- (3) 子问题恰好是个背包问题形式，这个子问题的约束是切割方法本身隐含的，任何切割方式都不能违反总长度。而背包问题虽然是难题，却可以用动态规划有效解决。

P2-SC model	Set covering model
$\min \sum_{j=1}^n x_j$ s.t. $\sum_{j=1}^n a_{ij} x_j \geq n_i, \forall i = 1, \dots, m$ $x_j \in \{0, 1\}, \forall j = 1, \dots, n$	$\min \sum_{j=1}^n c_j x_j$ s.t. $\sum_{j=1}^n a_{ij} x_j \geq 1, \forall i = 1, \dots, m$ $x_j \in \{0, 1\}, \forall j = 1, \dots, n$



34

列生成法

列生成算法

第 0 步, 初始化。

设迭代次数 $\ell \leftarrow 0$, 选择一个子集 $J_0 \subseteq J_{all}$, 使得对应的松弛受限主问题 $RLMP(J_0)$ 存在可行解。

第 1 步, 求解主问题。

求解松弛受限主问题 $RLMP(J_\ell)$, 得到相应的最优解 x^ℓ 和对偶问题的最优解 π^ℓ .

第 2 步, 列生成子问题。

考虑一个新的决策变量对应一个的列 a^g , 如果该列**满足所有列相关的复杂约束条件** $g \in J_{all}$. 并且

对最小化问题, 检验数 $\bar{c}_g < 0$, 则将该列添加到当前受限主问题 $RLMP(J_\ell)$ 的约束矩阵中。

对最大化问题, 检验数 $\bar{c}_g > 0$, 则将该列添加到当前受限主问题 $RLMP(J_\ell)$ 的约束矩阵中。

第 3 步, 判断算法终止条件。

如果在第 2 步中, 不能生成满足条件的列 g , 则终止算法, 此时解 x^ℓ 是原始模型的最优或近似最优解。

否则, 更新受限主问题约束矩阵的列集合 $J_{\ell+1} \leftarrow J_\ell \cup g$,

令 $\ell \leftarrow \ell + 1$, 返回第 1 步。

35

列生成法

通常来讲, 列生成算法最难实现的部分在于,

如何根据上一次迭代得到的对偶解 π^ℓ 和检验数 $\bar{c}_j \triangleq c_j - a^j \pi^\ell$, 生成一个新列 a^j 添加到限制主问题中。

对最小化问题,

若候选列对应的检验数 $\bar{c}_j < 0$, 则该列会进入到受限主问题中。

若对所有的列 $j \in J_{all}$, 检验数 $\bar{c}_j \geq 0$, 则当前主问题的解达到最优。

所以列生成算法有时会对所有可能的列 j , **选择检验数 \bar{c}_j 最小的列** (最小化问题)。

这种方法的好处在于它能够列生成算法第 3 步的结果, 得到明确的终止条件, 即如果最优的列 j 对应的检验数都不满足相应的正负符号规则, 则当前受限主问题已经找到了最优解, 所以终止算法。

在大多数实际应用中, 由于复杂约束和其他的约束条件给列生成算法带来的困难, 算法的第 2 步(求解子问题寻找列的部分)还可能会选择**启发式算法**实现, 只要该算法生成的列能够满足检验数的正负符号规则, 并且具有较好的效率即可。

36

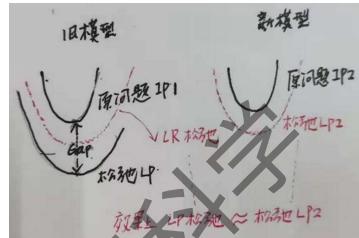
列生成法

对于最初的切割问题，我们可以构造两个模型，经典 IP 模型(P1)与集合覆盖模型(P2)。

- 对于前者，其实可以直接采用 B&B 或者 B&C 的方法来求解，只是效果不好。
- 对于后者，我们采用列生成的方法，相比而言，效果要比第一个模型要好。

之所以 P2 要比 P1 好，其实是因为 P2 的 LP 松弛等价与 P1 的拉格朗日松弛，所以整体上 P2 松弛后的界要比 P1 好。P1 的拉格朗日松弛的界等于 P2 的线性松弛的界。

P1-经典 IP 模型	P2-集合覆盖模型
$\min \sum_{k \in K} y_k$	$\min \sum_{j \in N} x_j$
s.t.	s.t.
$\sum_{k \in K} x_i^k \geq n_i, \forall i \in M$	$\sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M$
$\sum_{i \in M} w_i x_i^k \leq W \cdot y_k, \forall k \in K$	$x_j \in Z_+, \forall j \in N$
$x_i^k \in Z_+, \forall i \in M, k \in K$	
$y_k \in \{0, 1\}, \forall k \in K$	



37

Case – Cutting stock

- 钢铁生产厂，生产的钢条规格为218厘米，现在顾客需求44条规格为81厘米的钢条，3条规格为70厘米的钢条，48条规格为68厘米的钢条，问怎么切割才可能让总浪费最少？
- 两个模型比较
 - 经典模型中是一共有K根钢条可用， $y_k \in \{0,1\}$ 表示钢条k是否被使用， x_i^k 表示钢条k切割规格 w_i 切几个。两个需求分别是顾客需求必须被满足，钢条长度不能被超过。
 - 集合覆盖模型中， $x_j \in Z_+$ 是表示pattern j被使用的次数。 a_{ij} 是这个patter j本身包含几个 w_i .

Classical model – P1 \hookleftarrow	Set covering model – P2 \hookleftarrow
$\min \sum_{k \in K} y_k$	$\min \sum_{j \in N} x_j$
s.t. \hookleftarrow	s.t. \hookleftarrow
$\sum_{k \in K} x_i^k \geq n_i, \forall i \in M$	$\sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M$
$\sum_{i=1}^m w_i x_i^k \leq W y_k, \forall k \in K$	$x_j \in Z_+, \forall j \in N$
$x_i^k \in Z_+, \forall i \in M, k \in K$	
$y_k \in \{0, 1\}, \forall k \in K$	

38

Case

Classical model – P1 [←]	Set covering model – P2 [←]
$\begin{aligned} & \min \sum_{k \in K} y_k \\ & \text{s.t. } \\ & \sum_{k \in K} x_i^k \geq n_i, \forall i \in M^{\leftarrow} \\ & \sum_{i=1}^m w_i x_i^k \leq W y_k, \forall k \in K^{\leftarrow} \\ & x_i^k \in Z_+, \forall i \in M, k \in K^{\leftarrow} \\ & y_k \in \{0, 1\}, \forall k \in K^{\leftarrow} \end{aligned}$	$\begin{aligned} & \min \sum_{j \in N} x_j \\ & \text{s.t. } \\ & \sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M^{\leftarrow} \\ & x_j \in Z_+, \forall j \in N^{\leftarrow} \end{aligned}$

P2 → LMP [←]	RLMP [←]	Dual of RLMP [←]
$\begin{aligned} & \min \sum_{j \in N} x_j \\ & \text{s.t. } \\ & \sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M^{\leftarrow} \\ & x_j \geq 0, \forall j \in N^{\leftarrow} \end{aligned}$	$\begin{aligned} & \min \sum_{j \in P} x_j \\ & \text{s.t. } \\ & \sum_{j \in P} a_{ij} x_j \geq n_i, \forall i \in M^{\leftarrow} \\ & x_j \geq 0, j \in P, P \subseteq N^{\leftarrow} \end{aligned}$	$\begin{aligned} & \max \sum_{i \in M} n_i \pi_i \\ & \text{s.t. } \\ & \sum_{i \in M} a_{ij} \pi_i \leq 1, \forall j \in P^{\leftarrow} \\ & \pi_i \geq 0, \forall i \in M^{\leftarrow} \end{aligned}$

子问题 [←]	背包问题形式 [←]	[←]
$\begin{aligned} & \min 1 - \sum_{i=1}^m y_i \pi_i = 1 - \max \sum_{i=1}^m y_i \pi_i \\ & \text{s.t. } \sum_{i=1}^m w_i y_i \leq W \\ & y_i \in Z_+, i = 1, \dots, m^{\leftarrow} \end{aligned}$	$\begin{aligned} & \max \sum_{i=1}^m y_i \pi_i \\ & \text{s.t. } \sum_{i=1}^m w_i y_i \leq W \\ & y_i \in Z_+, i = 1, \dots, m^{\leftarrow} \end{aligned}$	$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix} = A_j, \text{ 是一种割法。}$ 约束条件强制这个割法必须是可行割。 $\{\min -x\} = \{-\max x\}$

39

首先是从 LMP 到 RLMP 的过程，即初始化选择 P 到的过程[←]

因为有 3 个顾客需求，即 $m=3$ ，我们需要先尝试构造 3 个初始 pattern。[←]

最简单的方法是一根钢条只切割一个产品，这样对三种产品会有三种割法。[←]

$$A_{m \times m} = (P1 \quad P2 \quad P3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

这样共需要 $44+3+48$ 条钢条。当然这样会造成巨大的浪费，但是这是一种最简单的可行的割法。而且适合做初始计算。[←]

以下另外三种割法，是一根钢条只切割一种规格，相比之下，浪费会少点。[←]

$$A_{m \times m} = (P1 \quad P2 \quad P3) = \begin{pmatrix} [218/81] & 0 & 0 \\ 0 & [218/70] & 0 \\ 0 & 0 & [218/68] \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

同样可以作为最开始的初始解，起步稍微好点，但是对整体影响不大。[←]

这样构造初始的 RLPM 问题[←]

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} x_3 \geq \begin{pmatrix} 44 \\ 3 \\ 48 \end{pmatrix}$$

RLMP [←]	1th RLMP [←]	Dual of 1th RLPM [←]
$\begin{aligned} & \min \sum_{j \in P} x_j \\ & \text{s.t. } \\ & \sum_{j \in P} a_{ij} x_j \geq n_i, \forall i \in M^{\leftarrow} \\ & x_j \geq 0, \forall j \in P^{\leftarrow} \end{aligned}$	$\begin{aligned} & \min x_1 + x_2 + x_3 \\ & \text{s.t. } \\ & x_1 \geq 44 \\ & x_2 \geq 3 \\ & x_3 \geq 48 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$	$\begin{aligned} & \max 44\pi_1 + 3\pi_2 + 48\pi_3 \\ & \text{s.t. } \\ & \pi_1 \leq 1 \\ & \pi_2 \leq 1 \\ & \pi_3 \leq 1 \\ & \pi_1, \pi_2, \pi_3 \geq 0 \end{aligned}$
$\begin{aligned} P &= \{P1 \quad P2 \quad P3\}^{\leftarrow} \\ A &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}^{\leftarrow} \end{aligned}$	$\begin{aligned} x^* &= (44, 3, 48)^T \\ \pi^* &= (1, 1, 1)^T \\ f^* &= 95 \end{aligned}$	

40

这样构造初始的 RLPM 问题 ^①		
$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} x_3 \geq \begin{pmatrix} 44 \\ 3 \\ 48 \end{pmatrix}$		
RLMP^②	1th RLMP^③	Dual of 1th RLPM^④
$\min \sum_{j \in P} x_j$ $s.t.$ $\sum_{j \in P} a_{ij} x_j \geq n_i, \forall i \in M$ $x_j \geq 0, \forall j \in P$	$\min x_1 + x_2 + x_3$ $s.t.$ $x_1 \geq 44$ $x_2 \geq 3$ $x_3 \geq 48$ $x_1, x_2, x_3 \geq 0$	$\max 44\pi_1 + 3\pi_2 + 48\pi_3$ $s.t.$ $\pi_1 \leq 1$ $\pi_2 \leq 1$ $\pi_3 \leq 1$ $\pi_1, \pi_2, \pi_3 \geq 0$
$P = \{P1 \quad P2 \quad P3\}$ $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ $f^* = 95$	$x^* = (44, 3, 48)^T$ $\pi^* = (1, 1, 1)^T$ $\kappa = -2$	\Leftrightarrow
构造子问题^⑤	背包问题形式^⑥	1th subproblem^⑦
$\min 1 - \sum_{i \in M} \pi_i y_i$ $= 1 - \max \sum_{i \in M} \pi_i y_i$ $s.t.$ $\sum_{i \in M} w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$	$\max \sum_{i \in M} \pi_i y_i$ $s.t.$ $\sum_{i \in M} w_i y_i \leq W$ $y_i \in Z_+, \forall i \in M$	$\max y_1 + y_2 + y_3$ $s.t.$ $81y_1 + 70y_2 + 68y_3 \leq 218$ $y \in Z_+^3$
\Leftrightarrow $y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{pmatrix} = A_j$ 是一种割法, 约束条件强制这个割法必须是可行割。	$\text{给定 } \pi^* = (1, 1, 1)^T$ $f = 3$ $\kappa = 1 - 3 = -2 < 0$	$y = (0, 0, 3)^T$ $f = 3$ $\kappa = 1 - 3 = -2 < 0$ $\text{对于这个背包问题, } y = (0, 0, 3)^T, \kappa = 1 - 3 = -2 < 0.$ $\text{意味着非基变量检验数为负, 所以我们需要增加新的列}(0, 0, 3)^T. \text{ 变成如下形式:}$ $A_{3 \times 4} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 3 \end{pmatrix}$

2nd RLMP^⑧	2nd knapsack problem^⑨
$\min x_1 + x_2 + x_3 + x_4$ $s.t.$ $x_1 \geq 44$ $x_2 \geq 3$ $x_3 + 3x_4 \geq 48$ $x = (44, 3, 0, 16)^T$ $\pi = (1.0, 1.0, 0.33)^T$ $f^* = 63$	$\max y_1 + y_2 + 0.33y_3$ $s.t.$ $81y_1 + 70y_2 + 68y_3 \leq 218$ $y \in Z_+^3$
继续增加一列 ^⑩	
$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} x_3 + \begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix} x_4 + \begin{pmatrix} 44 \\ 3 \\ 0 \end{pmatrix} x_5 \geq \begin{pmatrix} 44 \\ 3 \\ 48 \end{pmatrix}$ $A_{3 \times 5} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 3 \\ 0 & 0 & 1 & 3 & 0 \end{pmatrix}$	3rd RLMP^⑪
继续增加一列 ^⑩	3rd knapsack problem^⑫
	$\max y_1 + 0.33y_2 + 0.33y_3$ $s.t.$ $81y_1 + 70y_2 + 68y_3 \leq 218$ $y \in Z_+^3$
$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} x_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} x_3 + \begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix} x_4 + \begin{pmatrix} 44 \\ 3 \\ 0 \end{pmatrix} x_5 + \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} x_6 \geq \begin{pmatrix} 44 \\ 3 \\ 48 \end{pmatrix}$ $A_{3 \times 6} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 3 & 0 \\ 0 & 0 & 1 & 3 & 0 & 0 \end{pmatrix}$	$\max y = (2, 0, 0)^T$ $f = 2$ $\kappa = 1 - 2 = -1 < 0$

4th RLMP	4th knapsack problem
$\min x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leftarrow$ <i>s.t.</i> \leftarrow $x_1 + 2x_6 \geq 44 \leftarrow$ $x_2 + 3x_5 \geq 3 \leftarrow$ $x_3 + 3x_4 \geq 48 \leftarrow$ $x = (0, 0, 0, 16, 1, 22)^T \leftarrow$ $\pi = (0.5, 0.33, 0.33)^T \leftarrow$ $f^* = 39 \leftarrow$	$\max 0.5y_1 + 0.33y_2 + 0.33y_3 \leftarrow$ <i>s.t.</i> \leftarrow $81y_1 + 70y_2 + 68y_3 \leq 218 \leftarrow$ $y \in Z_+^3 \leftarrow$ $y = (1, 0, 2)^T \leftarrow$ $f = 1.17 \leftarrow$ $\kappa = 1 - 1.17 = -0.17 < 0 \leftarrow$
5th RLMP	5th knapsack problem
$\min x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leftarrow$ <i>s.t.</i> \leftarrow $x_1 + 2x_6 + x_7 \geq 44 \leftarrow$ $x_2 + 3x_5 \geq 3 \leftarrow$ $x_3 + 3x_4 + 2x_7 \geq 48 \leftarrow$ $x = (0, 0, 0, 0, 1, 10, 24)^T \leftarrow$ $\pi = (0.5, 0.33, 0.25)^T \leftarrow$ $f^* = 35 \leftarrow$	$\max 0.5y_1 + 0.33y_2 + 0.25y_3 \leftarrow$ <i>s.t.</i> \leftarrow $81y_1 + 70y_2 + 68y_3 \leq 218 \leftarrow$ $y \in Z_+^3 \leftarrow$ $y = (1, 0, 2)^T \leftarrow$ $f = 1 \leftarrow$ $\kappa = 1 - 1 = 0 \leftarrow$
<p style="color: red;">直到 $\kappa \geq 0$, 已经不能再找到更好的列了。 \leftarrow</p> <pre>===== final objective value = 35.0 solution = [0, 0, 0, 0, 1, 10, 24] cutting pattern is [1, 0, 0, 0, 0, 2, 1] [0, 1, 0, 0, 3, 0, 0] [0, 0, 1, 3, 0, 0, 2]</pre>	
<p style="color: green;">此时, RLMP 等价于 LMP, 最后从 LMP 到 P2 的转变 \leftarrow 对 $x'_j = \lceil x_j \rceil$ 向上取整, 可以保证这个 x'_j 是 P2 的可行解。 \leftarrow</p> $\sum_{j=1}^n a_{ij} x'_j \geq \sum_{j=1}^n a_{ij} x_j \geq n_i \leftarrow$ <p style="color: green;">实际上当我们求解了 LPM, 得到了 x_j, 也就是求得了 P2 问题的下界。 在此基础上, 可以用那些 B&B 或者 B&C 去求 P2 问题的最优解。 \leftarrow 注意向上取整只能保证是可行解。 \leftarrow</p> <p style="color: green;">上面的例子中 $x = (0, 0, 0, 0, 1, 10, 24)^T$, 恰好都是整数, 表示 \leftarrow 第 5 中割法用了 1 次, 第 6 种割法用了 10 次, 第 7 种割法用了 24 次, 最少用 35 根钢条。 \leftarrow</p>	

43

Vehicle-Routing Problem

对比经典模型 P1 与集合覆盖模型 P2

- P1 模型中把所有可能的切割方案都显式地写进了模型中,
 $y_k \in \{0, 1\}, x_i^k \in Z_+, \forall k \in K, i \in M, \sum_{i=1}^m w_i x_i^k \leq W y_k, k \in K$
- P2 模型并没有显式地把所有的 cut 方案都写进模型中, 只是每次产生一个 pattern, 而且 x_j 对应着每个 pattern 用多少次, $\sum_j a_{ij} x_j \geq n_i, \forall i \in M$ 。而每个 pattern 里默认隐含了具体的切法 $\sum_j a_{ij} w_j \leq W$.

现在的问题是这样的 idea 能否推广到其他问题上?

有没有其他的问题也可以这样建立两个 model,

- 其中一个模型显式地包含了所有可能的解,
- 而另外一个模型只是把所有可行的解先求出一部分来, 然后一个一个加进去, 相当于隐式地知道所有的解。

网络中路的问题一般可能具有这样的特点, 可能会有很多很多路径组合, 但是并不是所有可能的路径都是我们需要的, 往往我们的最优路径只需要少量的子路径组合起来就可以了。

比如对于 10 个顾客而言, 可能有 1 个人一条路, 2 个人一条路, 3 个人一条路 ...,

对应的路径数量就是 $c_{10}^1 + c_{10}^2 + \dots + c_{10}^{10}$, 而且其中对于每个组合的具体路径序列, 其实又是个排列问题, 所以说是个很大的数字。

VRP 问题中的一条路径就相当于切割问题中的一个 pattern.

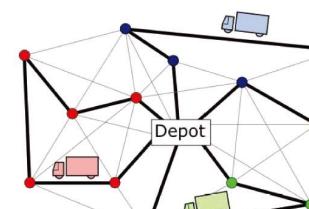


Figure: Vehicle routing problem

44

VRP – P1

我们考虑一个最基本的 VRP 问题，假设就一辆车，有容量约束，这辆车给一部分顾客送完货之后，就返回仓库装车，然后再去服务另外的顾客。要求总距离最短。因为就一辆车，所以不存在车辆索引，这个问题与 TSP 问题的唯一区别在于车辆容量导致需要出发很多次。

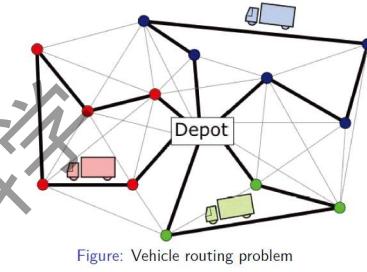
我们首先建立 P1 模型，基于 arc flow.

$$G = (V, A), V = \{0\} \cup N, A = \{(i, j) | i, j \in V, i \neq j\}$$

决策变量

- $x_{ij} \in \{0, 1\}, \forall i, j \in V$ 表示路径序列
- $y_i \in Z_+, \forall i \in V$ 表示车辆离开节点 i 时的车载量

经典模型 VRP-af (arc-flow) P1	
$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$ <p>s.t.</p> $\sum_{j \in V} x_{ij} = 1, \forall i \in N$ $\sum_{j \in V} x_{ij} = \sum_{j \in V} x_{ji}, \forall i \in N$ $y_i \geq y_j + d_j - M * (1 - x_{ij}), \forall i, j \in N$ $y_i + d_i \leq Q, \forall i \in V$ $x_{ij} \in \{0, 1\}, \forall i, j \in V$	$d_i, \forall i \in N$ demand of customer i . c_{ij} : transportation cost (distance) from i to j . Q : vehicle capacity $x_{ij} \in \{0, 1\}$: $x_{ij} = 1$ if a vehicle goes directly from i to j , $x_{ij} = 0$ otherwise $y_i \geq 0$: load of vehicle when leaving node i .



45

VRP – P2

然后我们来考虑 P2 集合覆盖模型。

首先，集合覆盖模型一定的是说有个很大的集合，我们只需要从中选择一部分出来，能够覆盖我们的需求就可以了。这里假设我们知道所有可能的路，用 R 来表示所有可能的路的集合。

其次，VRP 问题的解一定是从这个集合 R 中选择一部分出来组合起来就行了，因此，我们会有一个变量 $y_r \in \{0, 1\}, \forall r \in R$ 来标记路 r 是否在 VRP 的解中，然后把所有 $y_r = 1$ 的路组合起来就行。

我们的目标是这种组合方式能使得总距离最短。注意这里只是假设我们知道所有可能的路，实际上我们并不知道。开始的时候我们只是知道一部分，然后我们一个个往里加，直到找到一个路的组合能使得总距离最短。

关于路的一些说明

这个路一定是从 depot 出发，访问某些顾客，并返回 depot。也就是说路本身必须是符合要求，从仓库出发，服务一定数量的顾客（每个顾客只服务一次），并返回仓库。

给定了一条路 $r \in R$ 之后，我们就可以计算这条路上的费用或者距离 c_r 。而且我们也知道顾客 i 是否在这条路上， $a_{ir} = 1$ 表示顾客 i 在路 r 上。

注意这里还有一个隐含条件是这条路上的顾客需求不能超过车的容量。

就是说一条可行的路本身是包含了一些约束条件的，类似于切割问题中的一种割法也是默认包含了一些约束条件。

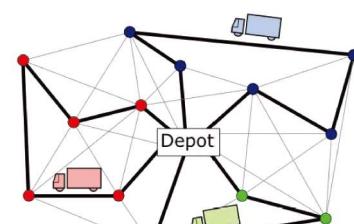


Figure: Vehicle routing problem

46

VRP – P2

集合覆盖模型 VRP-sc (set-covering) P2	
$\min \sum_{r \in R} c_r y_r$	R: sets of feasible routes
s.t.	c_r : transportation cost of route $r \in R$
$\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$	$a_{ir} = 1$ if node i is served on route r . $a_{ir} = 0$ otherwise
$y_r \in \{0,1\}, \forall r \in R$	$y_r = 1$ if route r is in the solution, $y_r = 0$ otherwise

这里的唯一一个约束是说每个顾客都必须在某条路上，也就是顾客需求必须被满足。

这里可能稍微细一点

对于顾客节点而言，应该是 $\sum_{r \in R} a_{ir} y_r = 1, \forall i \in N$

对于仓库节点而言，应该是 $\sum_{r \in R} a_{ir} y_r \geq 1, i = 0$

所以合起来就是 $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$

这里的 R 是一个很大数(exponentially large)，实际上我们是不可能把所有的 $r \in R$ 都写出来的。

所以我们尝试用列生成的方法来处理，每次生成一个可行的路。

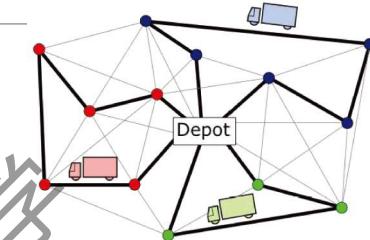


Figure: Vehicle routing problem

47

VRP – P2

VRP-sc, P2	切割问题的 P2	传统的集合覆盖模型
$\min \sum_{r \in R} c_r y_r$	$\min \sum_{j \in N} 1 * x_j$	$\min \sum_{j \in N} c_j x_j$
s.t.	s.t.	s.t.
$\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$	$\sum_{j \in N} a_{ij} x_j \geq n_i, \forall i \in M$	$\sum_{j \in N} a_{ij} x_j \geq 1, \forall i \in M$
$y_r \in \{0,1\}, \forall r \in R$	$x_j \in Z_+, \forall j \in N$	$x_j \in \{0,1\}, \forall j \in N$

对比这个 VRP-sc 与切割问题的 sc 模型，有三个区别

- 第一个问题：可以发现 VRP-sc 在目标函数部分多了个 c_r ，这个需要怎么处理？
- 第二个问题，VRP-sc 线性松弛是怎么松弛？好像与 CSP-p2 的松弛方式不一样？
- 第三个问题：我们怎么去构造一个子问题，使得这个子问题的解对应着 VRP-sc 的一列，且 reduced cost 小于 0。

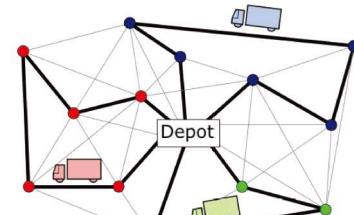


Figure: Vehicle routing problem

48

VRP – P2

VRP-sc	LMP	RLPM	Dual of RLPM
$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \in \{0,1\}, \forall r \in R$	$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in R$	$\min \sum_{r \in P} c_r y_r$ s.t. $\sum_{r \in P} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in P \subseteq R$	$\max \sum_{i \in V} \pi_i$ s.t. $\sum_{i \in V} a_{ir} \pi_i \leq c_r, \forall r \in P$ $\pi_i \geq 0, \forall i \in V$

按照前面列生成法的步骤,

- (1) 这里肯定也是先把 VRP-sc 做线性松弛, 构造 LMP, 也就是线性主问题。
- (2) 然后这个 LMP 的约束的行数其实也是固定的, $n+1$ 行, 对应着 n 个顾客和 1 个仓库。所以这里再构造 RLMP, 受限的线性主问题, 选择 $n+1$ 列, 约束条件是一个 $(n+1) \times (n+1)$ 的矩阵。
- (3) 然后求这个 RLMP 的对偶问题, 用对偶问题的解反过来验证非基变量的检验数是否为负值。
- (4) 构造子问题, 寻找是否能找到符合条件的列, 如果能找到, 就在 RLMP 上新加一列, 重新执行上面的计算过程。每一列表示一条可行路。然后肯定是一列一列的往里加, 加的时候要确保非基变量的检验数是负值才行。
- (5) 直到不能再插入新的列为止, 此时 RLMP=LMP.
- (6) 求解完 LMP 问题后, 再通过启发式方法, 求解原问题的整数解。比如向上取整, 或者结合 B&B 处理。

49

VRP – P2

VRP-sc	LMP	RLPM	Dual of RLPM
$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \in \{0,1\}, \forall r \in R$	$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in R$	$\min \sum_{r \in P} c_r y_r$ s.t. $\sum_{r \in P} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in P \subseteq R$	$\max \sum_{i \in V} \pi_i$ s.t. $\sum_{i \in V} a_{ir} \pi_i \leq c_r, \forall r \in P$ $\pi_i \geq 0, \forall i \in V$

构造 RLMP, 对于初始集合的选择, 最简单的情况 P 选择 $n+1$ 列, 相当于初始化时每个顾客单独一条路。

假设 RLMP 对偶问题的解为 π . 用对偶问题的约束条件来验证某一条路的检验数, 即

$$\bar{c}_r = c_r - \sum_{i \in V} a_{ir} \pi_i$$

我们的目标是找到 a_{ir} 能使得 $\bar{c}_r < 0$.

这样相当于在保持当前对偶解的前提下, 找到了一列数 a_{ir} 能够添加到 RLMP 中, 使得 RLMP 改进。

怎么寻找 a_{ir} 的过程就是子问题的构造过程。

50

VRP – P2

VRP-sc	LMP	RLPM	Dual of RLPM
$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \in \{0,1\}, \forall r \in R$	$\min \sum_{r \in R} c_r y_r$ s.t. $\sum_{r \in R} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in R$	$\min \sum_{r \in P} c_r y_r$ s.t. $\sum_{r \in P} a_{ir} y_r \geq 1, \forall i \in V$ $y_r \geq 0, \forall r \in P \subseteq R$	$\max \sum_{i \in V} \pi_i$ s.t. $\sum_{i \in V} a_{ir} \pi_i \leq c_r, \forall r \in P$ $\pi_i \geq 0, \forall i \in V$

对于某一条路 r 而言, $a_{ir} = 1$ 表示 i 在路径 r 上, 其实等价于 $\sum_{j \in N} x_{ij} = 1$, 表示 i 在某条弧上。

而 c_r 是一条路上的弧 (i, j) 的成本之和, 即 $c_r = \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$, 这里 $x_{ij} \in \{0, 1\}$ 表示弧 (i, j) 在路上, 前提是 (i, j) 在路 r 上。

$r_j = c_j - c_B^T B^{-1} A_j = c_j - \pi^T A_j$ 用对偶解来验证非基变量的检验数

所以有下面的转变

$$\bar{c}_r = c_r - \sum_{i \in V} a_{ir} \pi_i = \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} - \sum_{i \in V} \sum_{j \in V} x_{ij} \pi_i = \sum_{i \in V} \sum_{j \in V} (c_{ij} - \pi_i) x_{ij} = \sum_{i \in V} \sum_{j \in V} \bar{c}_{ij} x_{ij}$$

$$\bar{c}_{ij} \stackrel{\text{def}}{=} c_{ij} - \pi_i$$

所以说检验数的符号最终取决于 $\sum_{i \in N} \sum_{j \in N} (c_{ij} - \pi_i) x_{ij}$.

51

在此基础上, 我们构造列生成法的子问题。

Sub-problem	经典模型 VRP-af
$\min \sum_{i \in N} \sum_{j \in N} (c_{ij} - \pi_i) x_{ij}$ s.t. $\sum_{j \in N} x_{0j} = 1$ $\sum_{i \in N} x_{ij} = \sum_{k \in N} x_{jk}, \forall j \in N$ $y_i \geq y_j + d_j - M(1 - x_{ij}), \forall i \in N, j \in N \setminus \{1\}$ $y_i + d_i \leq Q, \forall i \in N$ $x_{ij} \in \{0, 1\}, \forall i, j \in N$	$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$ s.t. $\sum_{j \in N} x_{ij} = 1, \forall i \in N$ $\sum_{i \in N} x_{ij} = \sum_{k \in N} x_{jk}, \forall j \in N$ $y_i \geq y_j + d_j - M(1 - x_{ij}), \forall i \in N, j \in N \setminus \{1\}$ $y_i + d_i \leq Q, \forall i \in N$ $x_{ij} \in \{0, 1\}, \forall i, j \in N$

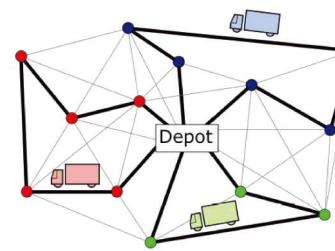


Figure: Vehicle routing problem

注意这个子问题的变量是 x_{ij} , 我们是为了寻找一条从仓库出发的路。

子问题是寻找一条路, 原问题 VRP-af 是为了寻找所有的路。

这个子问题与 VRP-af 的区别在于第一个约束条件和目标函数。

- 子问题因为只是寻找一条路, 所以第一个约束是 x_{0j} 而不是 x_{ij} .
- 在目标函数方面, $(c_{ij} - \pi_i)$ 可能是负值, 只有在 $(c_{ij} - \pi_i)$ 有负值的情况下, 子问题的目标函数值才有可能是负数, 对应着非基变量的检验数小于 0, 只有在这种情况下我们才会把 x_{ij} 构成的这条新路, 作为新的一列加入到 RLPM 中。

这个子问题的约束条件相当于一条可行路本身隐含的信息, 必须从仓库出发, 不能超过车容量, 不能有子回路。 $\sum_{j \in N} x_{0j} = 1$ 相当于限定了只能从仓库节点出发 1 次。

解这个子问题其实相当于又绕回到了解一个 VRP-af 问题, 但是这里只是形式上类似。

对于这个子问题, 其实是一个有资源约束的最短路问题 ESPPRC, 可以用改进的 Dijkstra 算法或者改进的动态规划方法求解, 就是标签算法那一套东西来求解, 所以这个子问题是很快求解的。

52

在此基础上，我们构造列生成法的子问题

Sub-problem	经典模型 VRP-af
$\min \sum_{i \in N} \sum_{j \in N} (c_{ij} - \pi_i) x_{ij}$	$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$
s.t.	s.t.
$\sum_{j \in N} x_{0j} = 1$	$\sum_{j \in N} x_{ij} = 1, \forall i \in N$
$\sum_{i \in N} x_{ij} = \sum_{k \in N} x_{jk}, \forall j \in N$	$\sum_{i \in N} x_{ij} = \sum_{k \in N} x_{jk}, \forall j \in N$
$y_i \geq y_j + d_j - M(1 - x_{ij}), \forall i \in N, j \in N \setminus \{i\}$	$y_i \geq y_j + d_j - M(1 - x_{ij}), \forall i \in N, j \in N \setminus \{i\}$
$y_i + d_i \leq Q, \forall i \in N$	$y_i + d_i \leq Q, \forall i \in N$
$x_{ij} \in \{0, 1\}, \forall i, j \in N$	$x_{ij} \in \{0, 1\}, \forall i, j \in N$

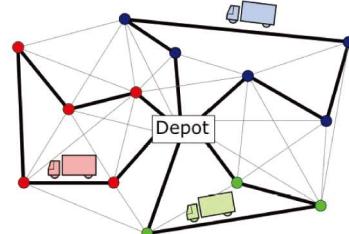
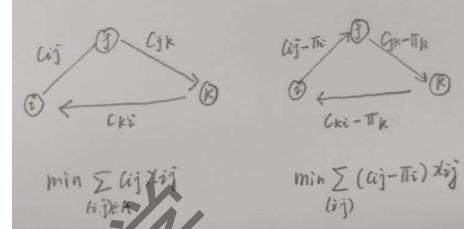


Figure: Vehicle routing problem



所以 VRP 通过子问题寻找一条路其实是在寻找一个负权回路。

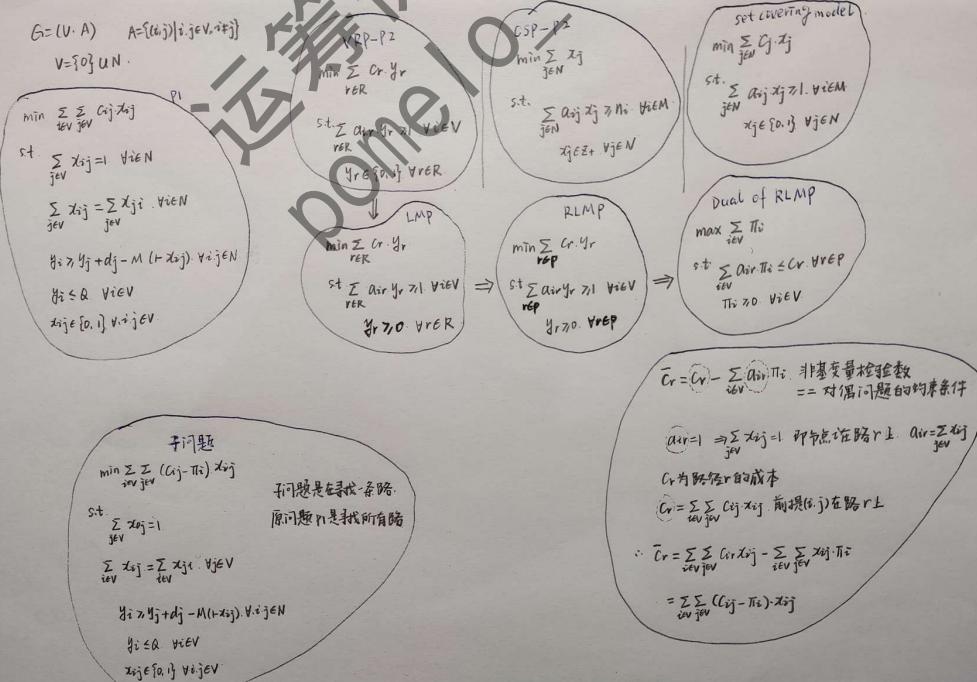
原来弧的权是 c_{ij} , 修改后的权是 $c_{ij} - \pi_i$, 其中 π_i 是节点 i 的对偶变量。

Recall Floyd-Warshall 最短路算法, 原来是在无负权回路的图中寻找多对多的最短路, 当图中有负权回路时, 这个算法也能找到这个负权回路 (无资源限制版本)

各种求解最短路问题的图论算法, 其本质都是基于动态规划的标签算法。

这里其实是 labelling algorithm for ESPPRC

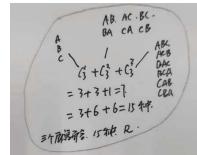
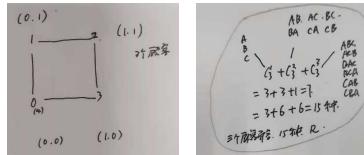
53



54

Case – VRP

- 对于这个vrp问题的初始可行路集合，可以很简单的直接给每个顾客安排一条路即可。
- 里面对于下面的例子而言，3个顾客，可能的路的集合有15种。
- 初始化时我们选择3条路，每个顾客一条路，相当于3列，然后一列一列往里添加。



三个顾客			
	C ₁	C ₂	C ₃
T ₁	1	0	0
C ₂	0	1	0
C ₃	0	0	1
T ₄	0	0	0

	2	2.82	2	3.41	2.41	4	3.41
	y ₁	y ₂	y ₃	y ₄	y ₅	y ₆	y ₇
顾客1	1	0	0	0	0	0	0
顾客2	0	1	0	0	0	0	0
顾客3	0	0	1	0	0	0	0
总和	1	1	1	1	1	1	1

55

```
[0, 1, 1, 1, 0] -----RLPM-----
[0, 0, 1, 1, 1] RLMP status = Optimal
[0, 1, 0, 1, 1] objective = 5.41
[0, 1, 1, 0, 1] route choice = [0.0, 0.0, 1.0, 1.0, 0.0]
[0, 0, 0, 0, 0] duals = [-0.0, 2.0, 1.4100000000000001, 2.0, -0.0]
-----sub-problem-----
status = Optimal
objective value / reduced cost = -1.4100000000000001
new route: 0->1 1-->2 2-->3 3-->4
route cost = 4.0
要插入的新列: [1, 1, 1, 1, 1]

=====第 1 次=====
-----RLPM-----
RLMP status = Optimal
objective = 4.0
route choice = [0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
duals = [-0.0, 2.0, -0.0, 2.0, -0.0]
-----sub-problem-----
status = Optimal
objective value / reduced cost = -0.5857864376269049
new route: 0->3 1-->4 3-->1
route cost = 3.414213562373095
要插入的新列: [1, 1, 0, 1, 1]

=====第 2 次=====
-----RLPM-----
RLMP status = Optimal
objective = 4.0
route choice = [0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0]
duals = [-0.0, 1.414213562373095, 0.5857864376269051, 2.0, -0.0]
-----sub-problem-----
status = Optimal
objective value / reduced cost = 0.0
new route: 0->3 3-->4
route cost = 2.0
要插入的新列: [1, 0, 0, 1, 1]
新列为[1, 0, 0, 1, 1]
检验数为 0.0
已经找不到新的列了!
```

56

列生成法总结

列生成法的适用范围，一定某个问题有两种不同的表达方式。

- P1 是经典模型，能在模型中显式地列出所有的可行方案，比如这里的切割问题，或者 VRP 问题。
- P2 是集合覆盖模型，都是在满足某个 demand 的前提下求最小。变量一般是 0-1 变量，表示某个方案是取还是不取，而这个方案到底是什么样子，我们其实并不知道。这个方案并没有显式表达，而是用了一个隐式（隐含了某些要求）。这个隐式的概念也是这种方法的优点所在。
- P1 是可以直接求解的，只是求解时间可能太长。B&B 是可以求精确解，但是 B&B 本身是部分枚举，并不是个有效的算法。
- P2 的列是未知的，所以不能直接用软件求解。但是可以用列生成的方法，先找到一些列，然后一列一列的往里加，使得每次加入一列都能改善当前的解。

Cut 问题是整数变量，表示某个方案用几次。VRP 问题是 0-1 变量，表示是否采用某个方案。

每一列代表一个可行方案，对于 cut 问题而言，是一个可行的切割方案，对 vrp 问题而言，是一条可行的路。

Cut-SC 的最终子问题是背包形式的问题 Knapsack Problem，所以可以用动态规划来求解

VRP-SC 的最终子问题是带资源约束的最短路径问题 ESPPRC，有专门的启发式方法，基于动态规划的标签算法。

57

Branch-and-Price

- $P2 \rightarrow LMP \rightarrow RLMP \rightarrow \text{Dual of RLMP}$
- 当使用列生成法解决实际问题时，我们希望主问题是个线性规划问题，这样可以同时得到主问题的解与对偶解，我们希望把整数变量放到子问题中去解决。
 - 对于前面的切割问题与路径问题，我们用列生成法得到 LMP 问题的解之后，从 $LMP \rightarrow P2$ 的转变过程是对 LMP 的解进行了简单的向上取整，这里是因为约束条件比较简单，所以可以这样操作。
 - 但是，在许多其他实际问题中，由于决策变量是整数变量（尤其是 0-1 变量），受限主问题的求解将会变得非常复杂。即便采用向上取整的方法也不能保证得到近似最优解，所以需要采用枚举方法。**分支定价算法(Branch and Price, B&P)**正是解决上述问题的一种方法。
 - 分支定价算法通过根据检验数不断生成新列的方法来提高**分支定界算法**求解线性松弛问题的效率。

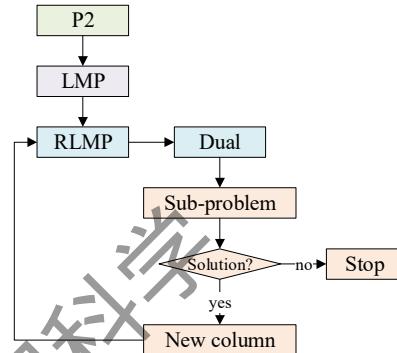
58

Branch-and-Price

- 基本认识

- 列生成其实是处理线性规划的，因为需要用到对偶问题，其处理的阶段是RLMP \rightarrow LMP。
- 寻找列的过程往往是寻找一列非负整数，因此子问题是整数规划问题 (IP)，往往用动态规划来处理，具体用什么方法取决于子问题的结构与性质。
- 而从LMP \rightarrow P2的过程，虽然是集合覆盖约束往往都是 \geq 的形式，但是简单的取整操作并不意味着精确求解，这部分要用B&B的框架来处理，尤其是0-1规划问题，这样才是精确求解，这就是Branch-and-Price，其中Price是求解子问题的代价。

$P_2 \rightarrow LMP \rightarrow RLMP \rightarrow \text{Dual of RLMP}$
 $\Rightarrow \text{sub-problem (price)}$
 $\rightarrow RLMP \rightarrow \text{Dual of RLMP}$
 $\Rightarrow \text{sub-problem (price)}$
 $\rightarrow RLMP \rightarrow \text{Dual of RLMP}$
...



59

Branch-and-Price

(1) 初造化，在根节点处，从可能大量的列中选择四个列构成 RLMP 的约束，对应的线性松弛问题的最优解是 $\bar{x}^0 = (0, 1, 1, 0)$ ，目标函数值为 $\bar{v} = 136$ 。因为各个元素都满足整数约束，所以 $\hat{x} \leftarrow \bar{x}^0$ ，是第一个原问题当前的最优解， $\hat{v} = \bar{v} = 136$ 。

尽管算法当前找到了一个整数解，但是由于可能通过生成新的列而找到更优的解，所以并不能类似分支定界算法马上停止。这里需要通过列生成来进行处理，直到不能找到新的列为止。

例如

在节点 1 处，新加入的第 5 列使得线性松弛问题的最优目标函数值减小到 129.7。

在节点 2 处，新加入的第 6 列将目标函数值进一步减小到 124.56。

(2) 假定在节点 2 处不能再生成以改进相应线性松弛问题目标函数值的列，那么 \bar{x}^2 就是该节点子问题的线性松弛最优解。然而，由于 \bar{x}^2 是分数解，并且目标函数仍比原问题当前最优目标函数值小，所以并不能马上终止该分支。分支，选择第二个元素 $x_2 = 0.3$ 进行分支。

(3) 对于 $x_2 = 0$ 的分支节点 3，线性松弛问题是不可行的，并且不可能生成新的列，所以基于最优解可行性的条件，终止这个分支。

(4) 对于 $x_2 = 1$ 的分支节点 4，得到一个线性松弛问题的最优解，其目标函数值是 $\bar{v} = 130.55$ ，最优解中仍然含有不满足整数约束的元素。但是，第 7 列由于对应的检验数满足条件可以进入限制主问题中，得到节点 5 处的新问题，其线性松弛问题的最优目标函数值为 $\bar{v} = 125.2$ ，最优解中仍含有不满足整数约束的元素，并且不存在可以继续添加的新列。

(5) 对节点 5 而言，虽然 \bar{x}^5 可能是线性松弛问题的最优解，但是算法还需要对其中不符合整数条件的元素进行分支，选择 $x_7 = 0.6$ 。

(6) 对 $x_7 = 1$ 对应的分支节点 6，得到的线性松弛问题最优解为 $\bar{x}^6 = (0, 1, 1, 0, 0, 0, 1)$ ，目标函数值为 126。由于 \bar{x}^6 满足所有整数约束，而且比原问题的当前最优解更好，因此更新原问题的当前最优解。又因为不可能生成改进该最优解的列，所以基于线性松弛问题整数解的判断条件终止这个分支。

(7) 对于 $x_7 = 0$ 的分支节点 7，得到的线性松弛问题最优目标函数值为 129.10，最优解中仍然包含不满足整数约束的元素。由于不可能生成改进该最优解的列，并且 $129.10 > \bar{v} = 126$ ，所以基于目标函数值上下界条件判断，终止这个分支。

(8) 最后，原问题的当前最优解就是原问题的最优解。

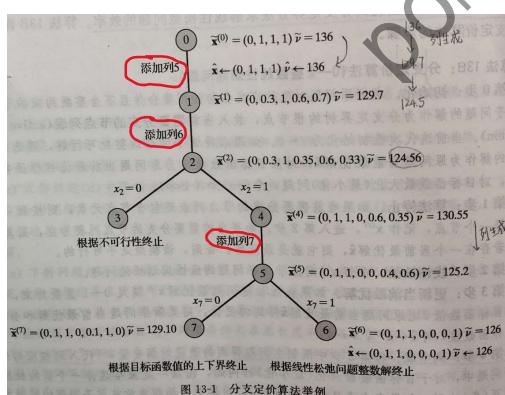
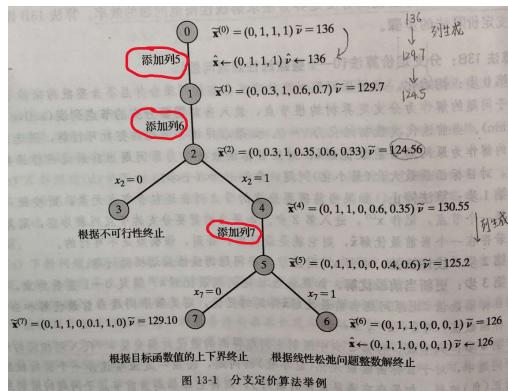


图 13-1 分支定价算法举例

60

Branch-and-Price

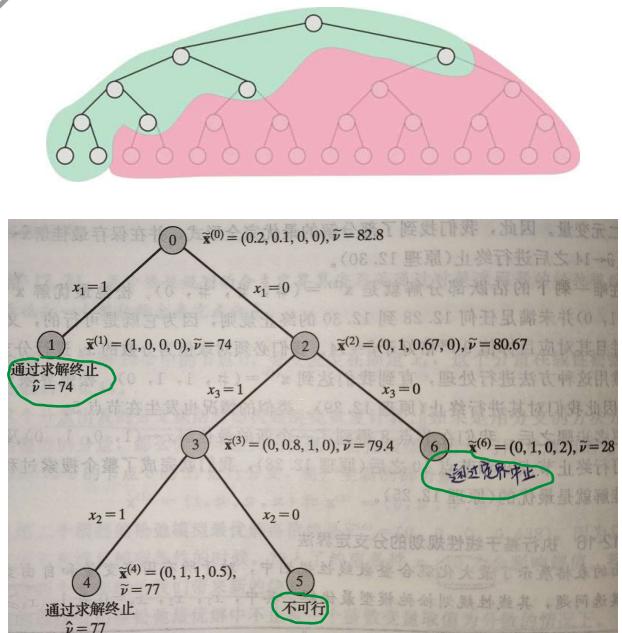


61

Branch-and-Bound

关键点

- 优先处理哪个活跃节点, search strategy
- 优先处理哪个非整分量 (branch)
- 通过不可行来终止
- 通过定界来终止 (bound)
- 通过整数解来终止 (relaxation)

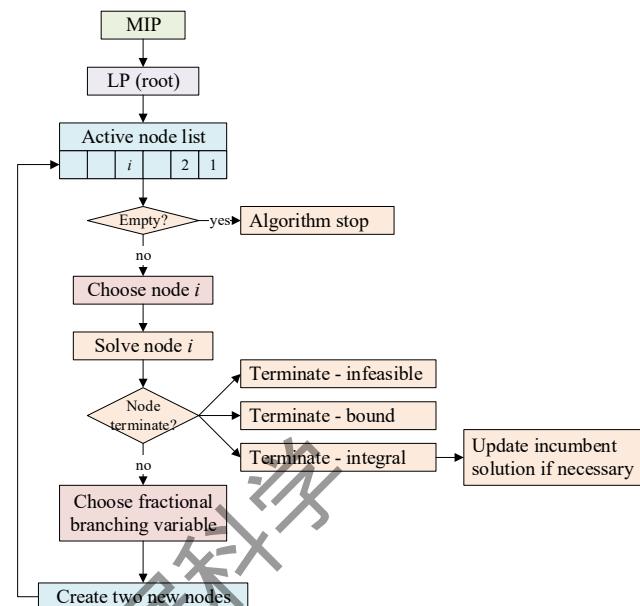


62

Branch-and-Bound

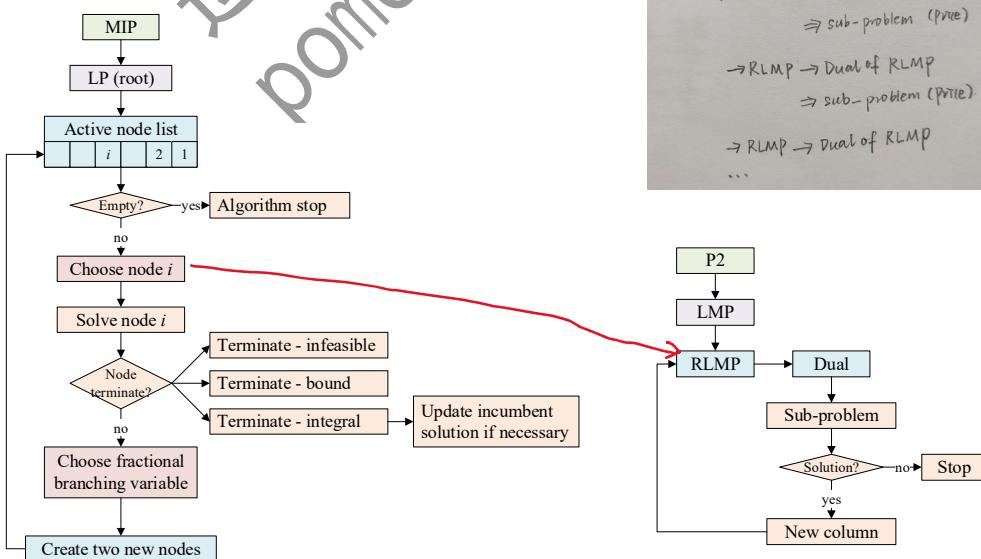
- 关键点

- 优先处理哪个活跃节点, search strategy
- 优先处理哪个非整分量 (branch)
- 通过不可行来终止
- 通过定界来终止 (bound)
- 通过整数解来终止 (relaxation)



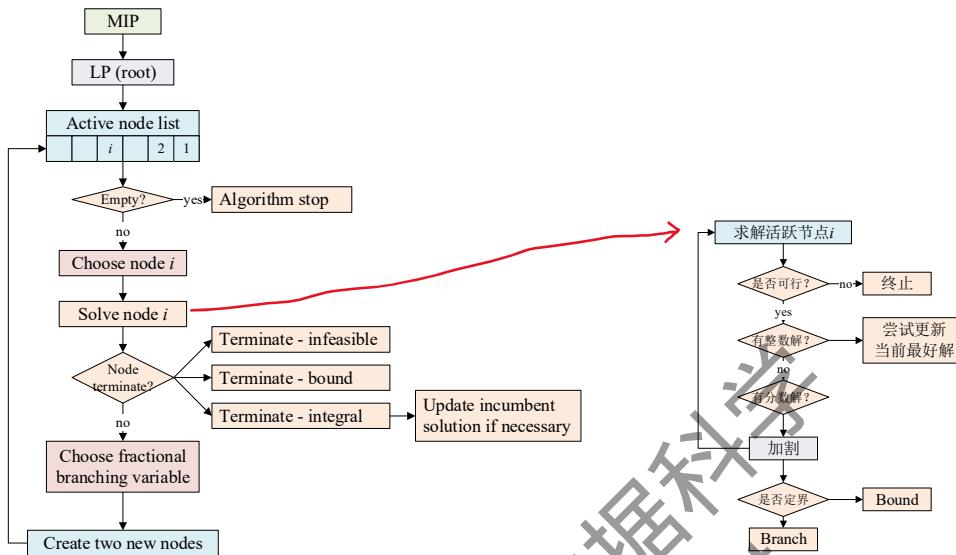
63

Branch-and-Price



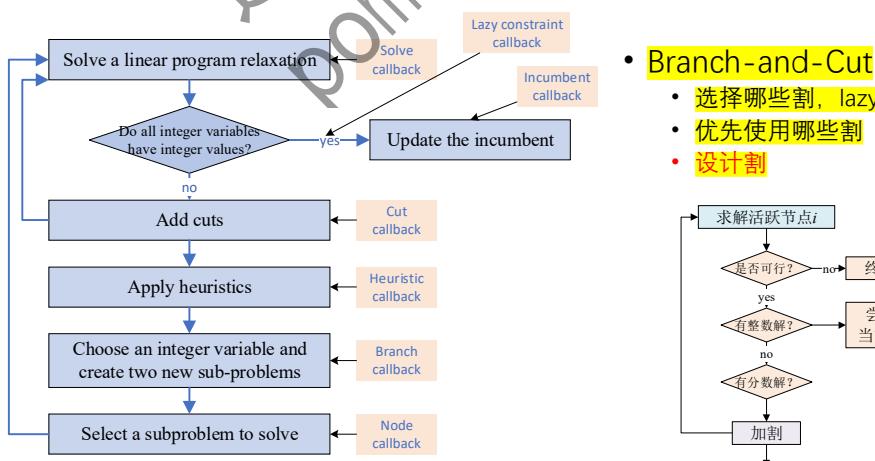
64

Branch-and-Cut



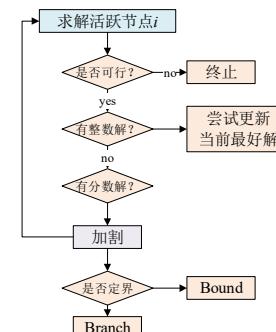
65

Branch-and-Cut



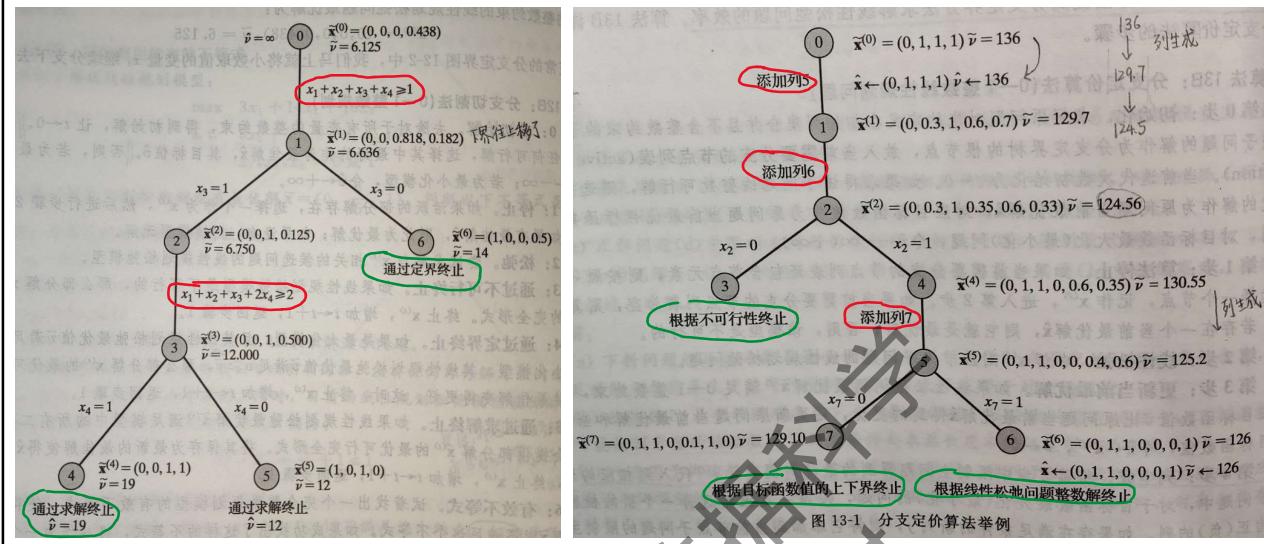
- Branch-and-Cut

- 选择哪些割, lazy cut and user cut
- 优先使用哪些割
- 设计割



66

B&C vs B&P



Branch-and-XX

总结

Branch-and-Bound, B&B

Branch-and-linear relaxation

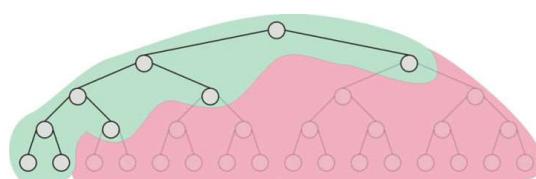
Branch-and-Lagrangian relaxation

Branch-and-Cut, B&C

Branch-and-Benders cut?

Branch-and-Price, B&P

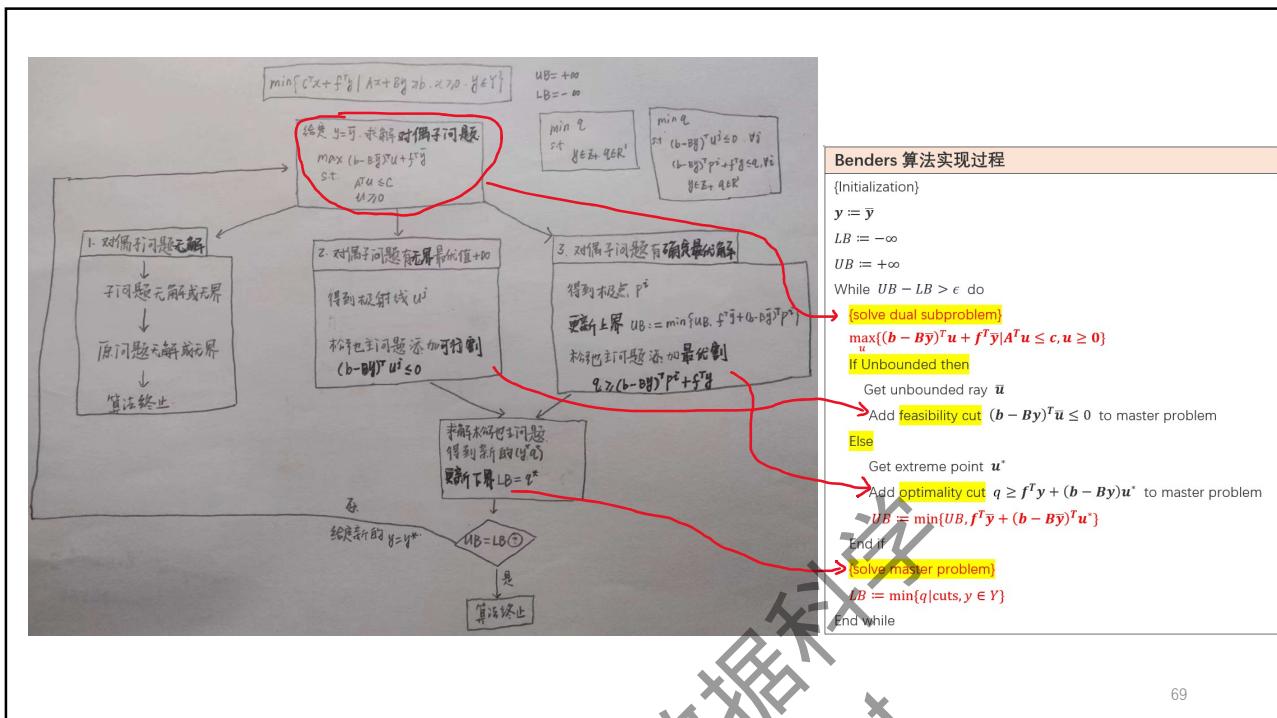
Branch-and-Price-and-Cut, B&P&C



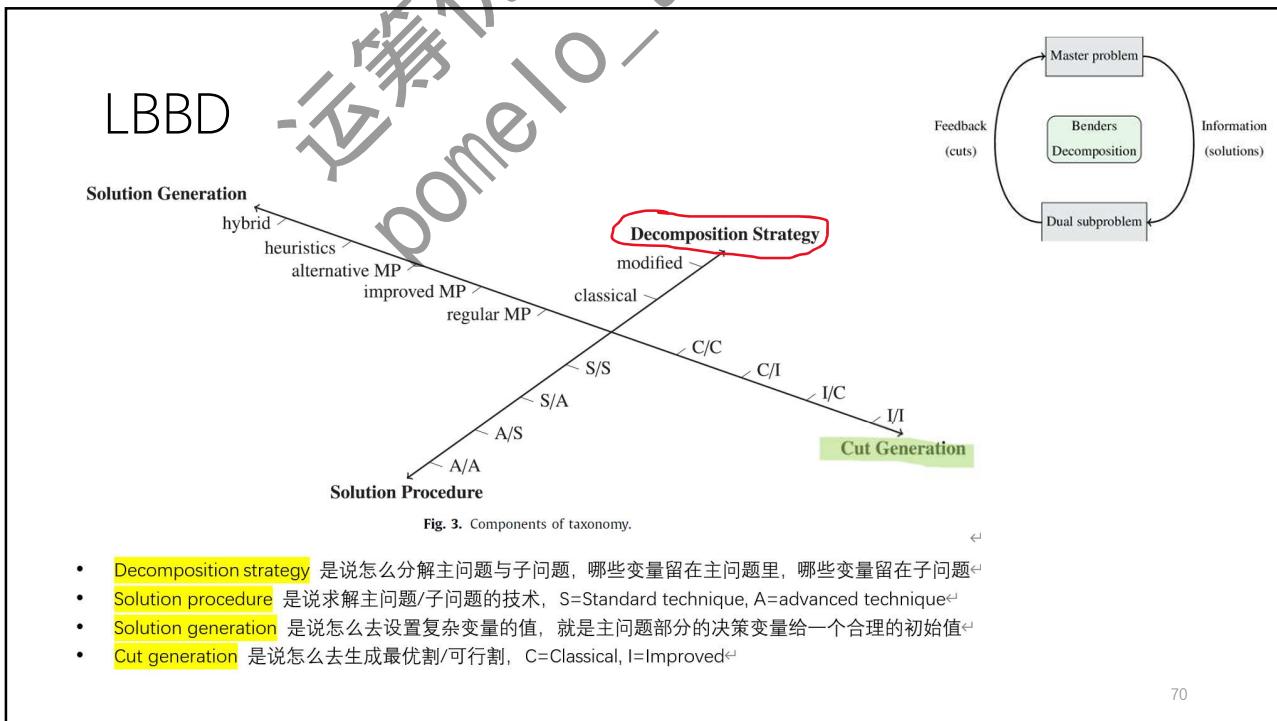
目的都是为了减少分支定界树的搜索空间，尽量剪枝。

- Upper bound, UB 越小越好，UB 对应的解是原问题的可行解，最好的 UB 对应着 incumbent solution.
尽早找到尽可能好的 UB，以便于剪枝。这个寻找往往需要用到一些启发式的技巧。
- 加 cut 或者加 column 的初衷都是为了改善 Lower bound, LB, 下界，下界越大越好。
对于一个 IP 问题的松弛问题而言，我们希望松弛后的模型越紧越好， $(v^* - \bar{v})$ 的 gap 越小越好。
- $LB = \bar{v} \leq v^* \leq \hat{v} = UB$, $[\bar{v}, \hat{v}]$ 的区间 gap 越小，越有助于快速定位 v^* .

很经常地，B&B 用来处理 IP 主问题，DP 用来处理某个子问题。



69



70

Branch-and-Benders

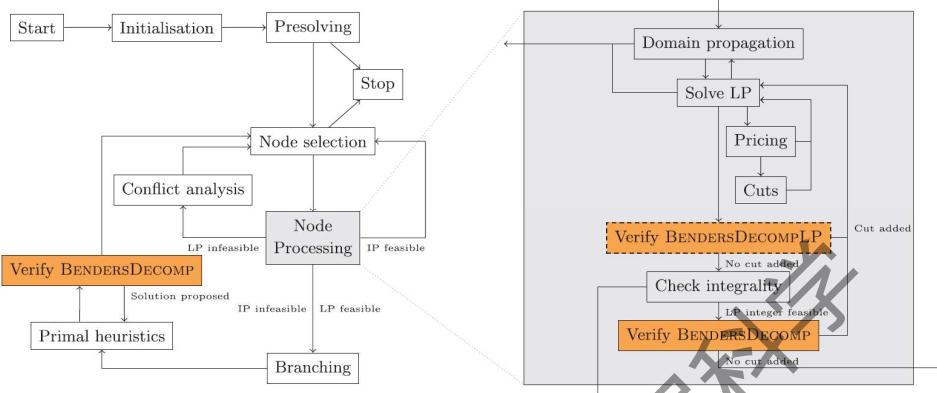


Fig. 1. The solving process of SCIP including the verification of the BD constraint handlers

71