数据魔术师运筹优化及人工智能系列讲座
第2期

孙鹏 博士
Takeaway.com
(peng.sun@takeaway.com)

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

微信公众号"数据魔术师"

负责人：秦虎 教授
华中科技大学管理学院
微信号：43340630

扫码即可关注

目标：将算法和代码实现引入运筹学教学

适用人群：本科生、需入门运筹优化算法的硕士生、博士生，以及其他热爱运筹优化的初学者

文章形式：知识点的介绍，优化问题的描述，算法原理的介绍，程序源代码的介绍以及算例的介绍等等

涵盖的知识点：单纯形法，运输问题，网络流问题，分支定界，列生成算法，动态规划，Benders 分解，禁忌搜索，遗传算法，粒子群算法，自适应大领域搜索算法，蚁群算法，变邻域搜索算法，CPLEX的应用及建模等等，**并且内容在不断地增加ing**

公众号最大的特色：公开源代码，并且将代码详细注释，易于自主学习

分枝定价切平面算法 (Branch-Price & Cut) 及其在路径规划类问题中的研究与应用

孙鹏
Takeaway.com
(peng.sun@takeaway.com)

Technische Universiteit Eindhoven University of Technology

Where innovation starts

提纲

1. 背景介绍
2. 带时间窗的车辆路径规划问题介绍
3. 列生成以及定价问题的生成和求解
4. 分枝定界
5. 切平面生成
6. 常见的加速和改进技巧
7. 最新实验结果介绍
8. 个人实践
9. 总结

需要的知识储备

1. 理解线性规划的对偶理论 (如何构造对偶问题)*
2. 了解图论里的最短路径和流相关的算法原理

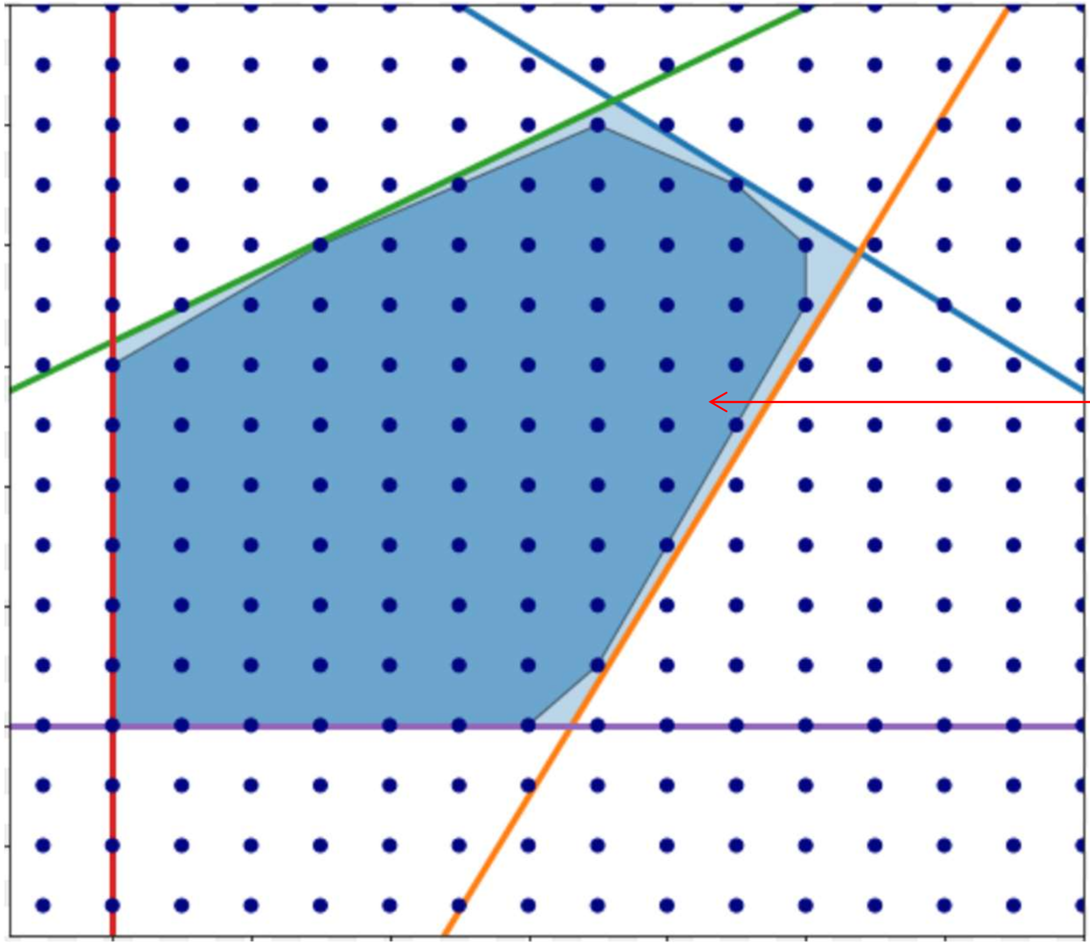
整数/线性规划

- 整数线性规划: 在线性规划基础上对变量增加整数要求;
- 纯整数线性规划: 所有决策变量都是整数;
- 混合整数线性规划: 只有一部分变量是整数;
- 0-1 整数线性规划: 所有变量都是0或者1.

整数线性规划的线性松弛: 决策变量整数约束进行松弛，得到一个线性规划

上界: (Upper bound)　　　　下界: (Lower bound)

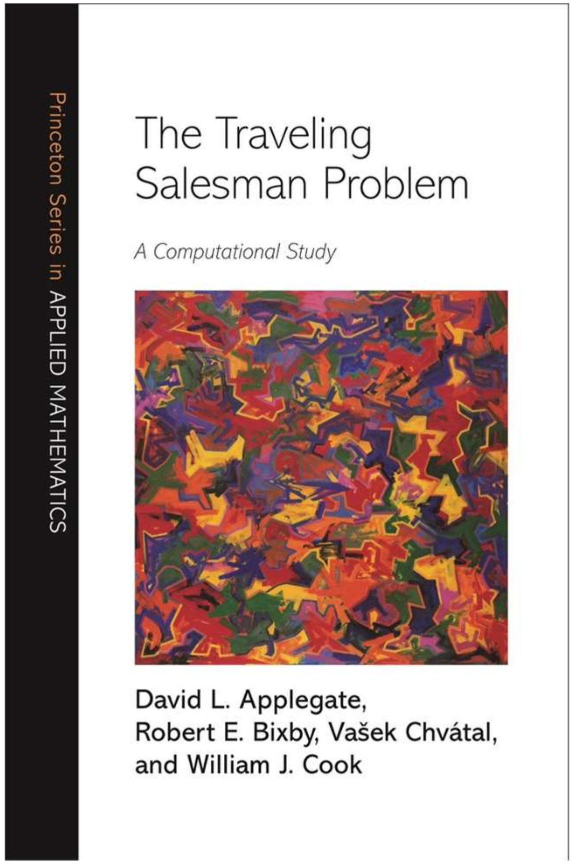
整数/线性规划



Convex hull
凸包

带时间窗的车辆路径规划问题 (VRPTW)

旅行商问题 (Traveling Salesman Problem, TSP)



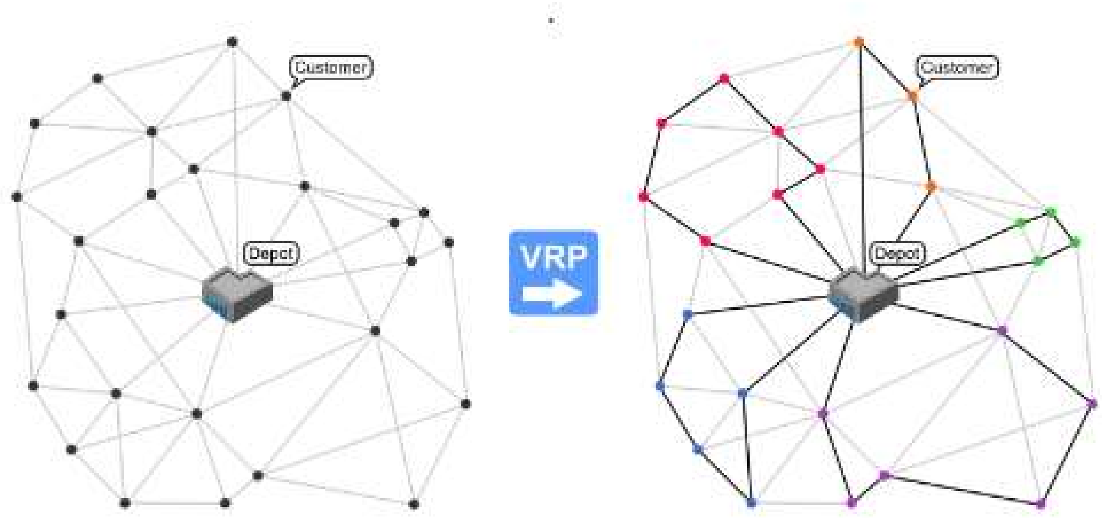
Technische Universiteit
Eindhoven
University of Technology

带时间窗的车辆路径规划问题 (VRPTW)

问题定义

输入： 一个配送中心 (Depot),
一组客户 (Customers),
多辆同型车 (Homogeneous Vehicles),
每个客户带一个服务时间窗 (Time window),
每辆车有一个容量限制 (Capacity limit)

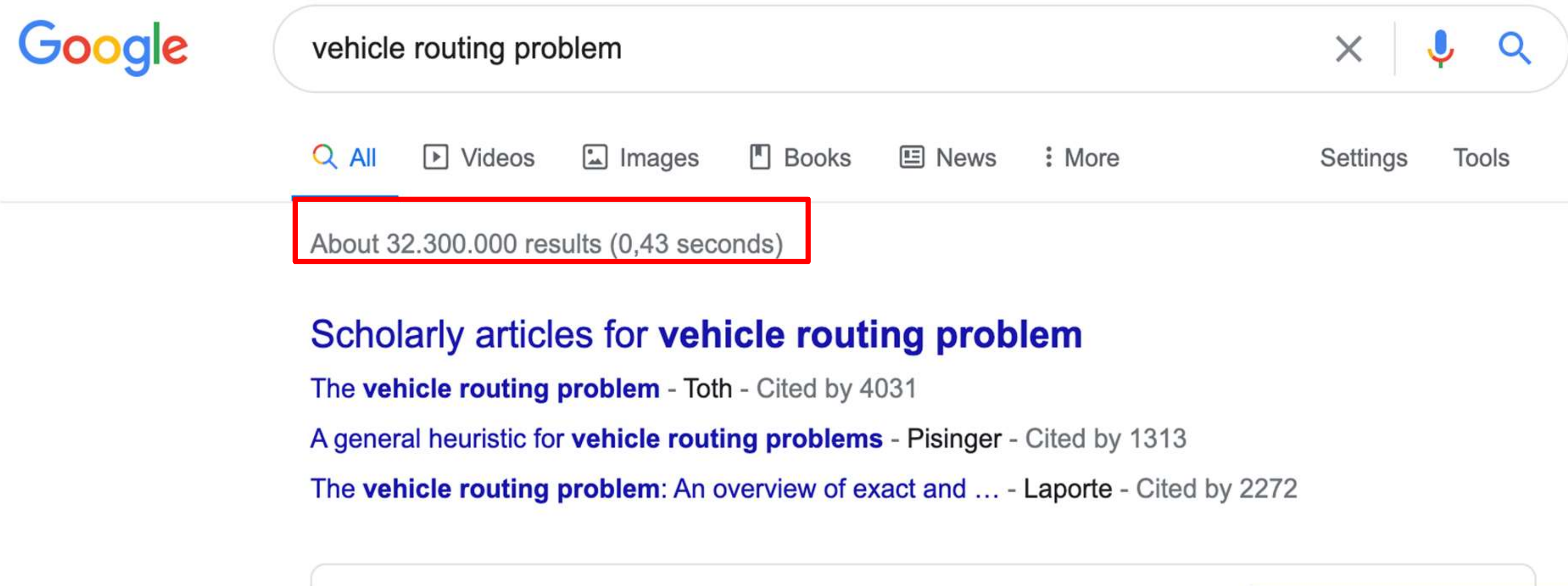
输出：
一组符合各项约束条件的车辆行驶路径，同时最小化车辆行驶的总里程。



An instance of a VRP (left) and its solution (right)

Technische Universiteit
Eindhoven
University of Technology

带时间窗的车辆路径规划问题 (VRPTW)



带时间窗的车辆路径规划问题 (VRPTW)

常量符号：

- $V = \{v_0, \ldots v_i, \ldots, v_j, \ldots\}$ 为点的集合。
- $(v_i, v_j) \in A$ 为边的集合。
- q_i 表示顾客 i 的需求量。
- s_i 表示顾客 i 所需要的服务时间。
- $[e_i, l_i]$ 表示顾客 i 的时间窗。
- Q 表示车辆容量。
- U 表示车辆集合。
- c_{ij} 和 t_{ij} 分别表示从顾客 i 到顾客 j 的行驶成本和行驶 时长。

决策变量符号：

- x_{ij}^u 0-1 变量。当边(v_i, v_j)被车辆 u 使用时为1， 否则为0。
- t_i^u 表示车辆 u 开始服务顾客 i 的时刻。

Technische Universiteit Eindhoven University of Technology

带时间窗的车辆路径规划问题 (VRPTW)

Arc-based Formulation

$$\text{minimize} \boxed{\sum_{1\le u\le U}} \sum_{(v_i,v_j)\in A} c_{ij}x_{ij}^u$$ → **最小化车辆行驶成本总和**

$$\text{subject to} \boxed{\sum_{1\le u\le U}} \sum_{\{v_j\in V|(v_i,v_j)\in A\}} x_{ij}^u \ge 1 \ \ (v_i \in V\backslash\{v_0\})$$ → **1). 每个顾客至少访问一次**

$$\sum_{\{v_j\in V|(v_i,v_j)\in A\}} x_{ij}^u - \sum_{\{v_j\in V|(v_i,v_j)\in A\}} x_{ji}^u = 0 \ \ (v_i\in V, 1\le u\le U)$$ → **2). 流守恒**

$$\sum_{\{v_j\in V|(v_0,v_j)\in A\}} x_{0j}^u \le 1 \ \ (1\le u\le U)$$ → **3). 车辆必须从v_0 出发**

$$\sum_{(v_i,v_j)\in A} q_i x_{ij}^u \le Q \ \ (1\le u\le U)$$ → **4). 容量约束**

$$t_i^u + s_i + t_{ij} - t_j^u + Mx_{ij}^u \le M\left((v_i,v_j\in A), v_j\ne v_0, (1\le u\le U)\right)$$

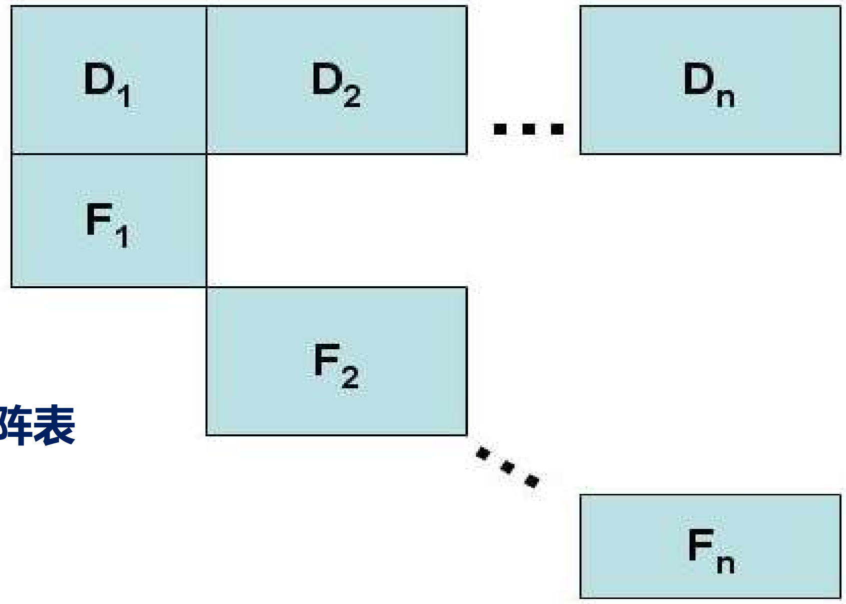
$$t_i^u + s_i + t_{i0} - l_0 + Mx_{i0}^u \le M\left((v_i,v_j\in A), (1\le u\le U)\right)$$ → **5). 6). 访问时间连续性**

$$e_i \le t_i^u \le l_i \ \ (v_i\in V, 1\le u\le U\,)$$ → **7). 时间窗约束**

$$x_{ij}^u \in \{0,1\} \left(v_i, v_j\in A, 1\le u\le U\right)$$ → **8). 0-1 变量**

列生成 (Column Generation)

Danzig-Wolfe Decomposition



George Dantzig 和 Phil Wolfe 在1960年创立，用于解决大规模的线性规划问题。

线性规划的约束矩阵具有特殊的形式，D 矩阵表示的是耦合约束关系而F 表示独立的子矩阵。

$$
\begin{array}{llllllllr}
\text{Minimize:} & \mathbf{c}_1'\mathbf{x}_1 & + & \mathbf{c}_2'\mathbf{x}_2 & + \cdots + & \mathbf{c}_l'\mathbf{x}_l & & & (1)\\
\text{Subject to:} & \mathbf{D}_1\mathbf{x}_1 & + & \mathbf{D}_2\mathbf{x}_2 & + \cdots + & \mathbf{D}_l\mathbf{x}_l & = & \mathbf{b}_0 & (2)\\
 & \boxed{\mathbf{F}_1\mathbf{x}_1} & & & & & = & \mathbf{b}_1 & (3)\\
 & & & \boxed{\mathbf{F}_2\mathbf{x}_2} & & & = & \mathbf{b}_2 & (4)\\
 & & & & & & \vdots & & \\
 & & & & & \boxed{\mathbf{F}_l\mathbf{x}_l} & = & \mathbf{b}_l, & (5)
\end{array}
$$

主问题

子问题

列生成 (Column Generation)

Danzig-Wolfe Decomposition

思路：

- **首先将原问题分解成主问题和若干个子问题；**
- **给每个子问题定义一个目标函数；**
- **求解每一个子问题 F_i ；**
- **将每个子问题的解提供给主问题；**
- **如果主问题的目标函数被优化更新子问题的目标函数，继续循环。**

列生成 (Column Generation)

子问题相同！！！

1. 不考虑 u
2. 找到所有满足下面所有约束条件的路径。

$$\sum_{\{v_j \in V|(v_i,v_j)\in A\}} x_{ij} - \sum_{\{v_j \in V|(v_i,v_j)\in A\}} x_{ji} = 0 \quad (v_i \in V)$$

$$\sum_{\{v_j \in V|(v_0,v_j)\in A\}} x_{0j} \le 1$$

$$\sum_{(v_i,v_j)\in A} q_i x_{ij} \le Q$$

$$t_i + s_i + t_{ij} - t_j + M x_{ij} \le M\left((v_i, v_j \in A), v_j \ne v_0\right)$$

$$t_i + s_i + t_{i0} - t_0 + M x_{i0} \le M\left((v_i, v_j \in A)\right)$$

$$e_i \le t_i \le l_i \quad (v_i \in V)$$

$$x_{ij} \in [0,1] \ (v_i, v_j \in A)$$

A feasible route set Ω

Technische Universiteit
Eindhoven
University of Technology

列生成 (Column Generation)

θ_k^u：当车辆 u 使用路径 $r_k \in \Omega$ 为1， 否则为0；

$b_{ij}^{\boldsymbol{k}}$：当路径 r_k 访问了边 (v_i, v_j) 为1，否则为0；

$$\sum_{r_k \in \Omega} \theta_k^u = 1 (1 \leq u \leq U)$$

$$\Downarrow$$

$$x_{ij}^u = b_{ij}^k \sum_{r_k \in \Omega} \theta_k^u = \sum_{r_k \in \Omega} b_{ij}^k \theta_k^u$$

列生成 (Column Generation)

$$\text{minimize} \sum_{1 \le u \le U} \sum_{(v_i, v_j) \in A} c_{ij} x_{ij}^u$$

$$\text{subject to} \quad \sum_{1 \le u \le U} \sum_{\{v_j \in V | (v_i, v_j) \in A\}} x_{ij}^u \ge 1 \ \ (v_i \in V \backslash \{v_0\})$$

$$x_{ij}^u \in \{0,1\} \ \left(v_i, v_j \in A, 1 \le u \le U\right)$$

$$\text{minimize} \sum_{1 \le u \le U} \sum_{(v_i, v_j) \in A} c_{ij} \sum_{r_k \in \Omega} b_{ij}^k \theta_k^u$$

$$\text{subject to} \quad \sum_{1 \le u \le U} \sum_{\{v_j \in V | (v_i, v_j) \in A\}} \left(\sum_{r_k \in \Omega} b_{ij}^k \theta_k^u \right) \ge 1 \ \ (v_i \in V \backslash \{v_0\})$$

$$\sum_{r_k \in \Omega} \theta_k^u = 1 (1 \le u \le U)$$

$$\theta_k^u \in \{0,1\} (r_k \in \Omega, 1 \le u \le U)$$

列生成 (Column Generation)

能否进一步简化主问题模型?

- c_k： **路径** $r_k \in \Omega$ **的行驶成本;**
- a_{ik}： **当路径** $r_k \in \Omega$ **访问了顾客** v_i **为1；否则为0** .

$$c_k \theta_k^u = \sum_{(v_i, v_j) \in A} c_{ij}\, b_{ij}^k \theta_k^u$$

$$
\begin{aligned}
&\text{minimize} \quad \sum_{1 \le u \le U} \sum_{r_k \in \Omega} c_k \theta_k^u \\
&\text{subject to} \quad \sum_{1 \le u \le U} \sum_{r_k \in \Omega} a_{ik} \theta_k^u \ge 1 \ \ (v_i \in V \backslash \{v_0\}) \\
&\qquad \sum_{r_k \in \Omega} \theta_k^u = 1 (1 \le u \le U) \\
&\qquad \theta_k^u \in \{0,1\} (r_k \in \Omega, 1 \le u \le U)
\end{aligned}
$$

$$a_{ik} \theta_k^u = \sum_{\{v_j \in V | (v_i, v_j) \in A\}} \left(\sum_{r_k \in \Omega} b_{ij}^k \theta_k^u \right)$$

列生成 (Column Generation)

不考虑 u

主问题:

Master Problem

Path-based Formulation

Linear Relaxation

根据线性规划的对偶理论 (Dual Theory)

$$\text{minimize} \sum_{r_k \in \Omega} c_k \theta_k$$

$$\text{subject to} \quad \sum_{r_k \in \Omega} a_{ik} \theta_k \geq 1 \ \ (v_i \in V \backslash \{v_0\}) \longrightarrow \lambda_i^*$$

$$\sum_{r_k \in \Omega} \theta_k \leq U \longrightarrow \lambda_0^*$$

$$\theta_k \cancel{\in \{0,1\}} \ (r_k \in \Omega)$$

主问题转化为一个 set partitioning problem !!!

列生成 (Column Generation)

Master Problem
(Linear Relaxation)

Ω: 所有可行路径的集合

Restricted Master Problem
(Linear Relaxation)

Ω': 可扩充的可行路径的集合

n	$\|\Omega\|$
10	6235301
20	4.18041131107144e+18
30	4.557791689620825e+32
…	…
100	1.603607401161831 4e+158

列生成 (Column Generation)

$\Omega \rightarrow \Omega'$ **Restricted Master Problem Linear Relaxation**

$$\text{minimize} \sum_{r_k \in \Omega'} c_k \theta_k$$

$$\text{subject to} \quad \sum_{r_k \in \Omega'} a_{ik} \theta_k \geq 1 \ \ (v_i \in V \setminus \{v_0\}) \longrightarrow \lambda_i$$

$$\sum_{r_k \in \Omega'} \theta_k \leq U \longrightarrow \lambda_0$$

$$0 \leq \theta_k \leq 1 (r_k \in \Omega')$$

列生成 (Column Generation)

对偶问题

Dual problem

$$\text{maximize} \sum_{v_i \in V \setminus \{v_0\}} \lambda_i + U\lambda_0$$

$$\text{subject to} \quad \boxed{\sum_{v_i \in V \setminus \{v_0\}} a_{ik}\lambda_i + \lambda_0 \le c_k \left(r_k \in \Omega', v_i \in V \setminus \{v_0\}\right)}$$

$$\lambda_i \ge 0 \ \left(v_i \in V \setminus \{v_0\}\right)$$

$$\lambda_0 \le 0$$

$$c_k - \sum_{v_i \in V \setminus \{v_0\}} a_{ik}\lambda_i - \lambda_0 \ge 0 \left(r_k \in \Omega', v_i \in V \setminus \{v_0\}\right)$$

Reduced Cost >= 0

列生成 (Column Generation)

子问题的目标：找出路径 $r_k \in \Omega \backslash \Omega'$**，使得**

$$c_k - \sum_{v_i \in V \backslash \{v_0\}} a_{ik}\lambda_i - \lambda_0 < 0 \quad \rightarrow \quad r_k \in \Omega'$$

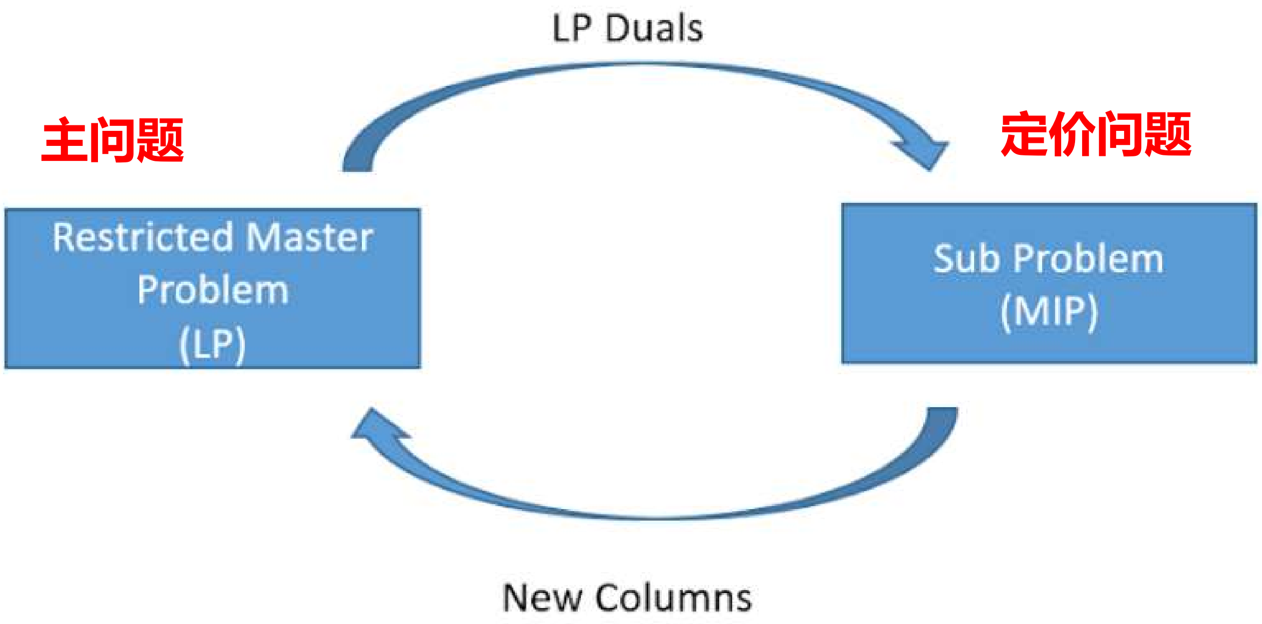
$$\sum_{(v_j, v_i) \in A} b_{ji}\left(c_{ji} - \lambda_i\right) < 0$$

$$min \sum_{(v_j, v_i) \in A} b_{ji}\boxed{\left(c_{ji} - \lambda_i\right)}$$

$$\lambda_i \rightarrow \lambda_i^* \qquad \lambda_0 \rightarrow \lambda_0^*$$

最短路径问题：

列生成 (Column Generation)

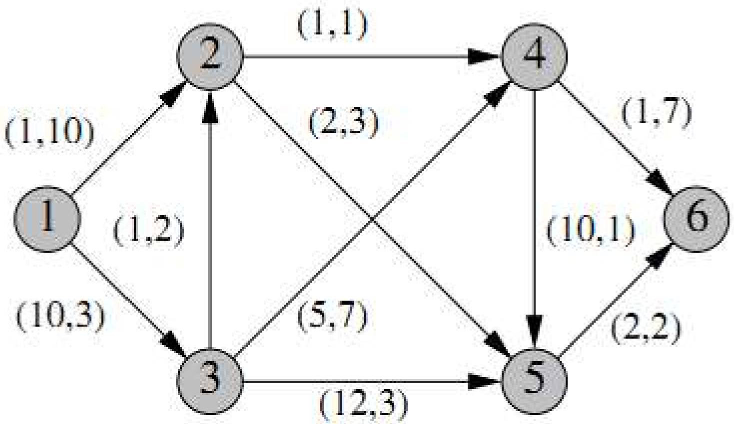


定价问题 (Pricing Problem)

带资源限制的最短路径问题：

Elementary Shortest Path Problem with Resource Constraints (NP-hard)

Shortest Path Problem with Resource Constraints (Pseudo-polynomial)



Time Constraint Shortest Path Problem (Ahuja et al. 1993)

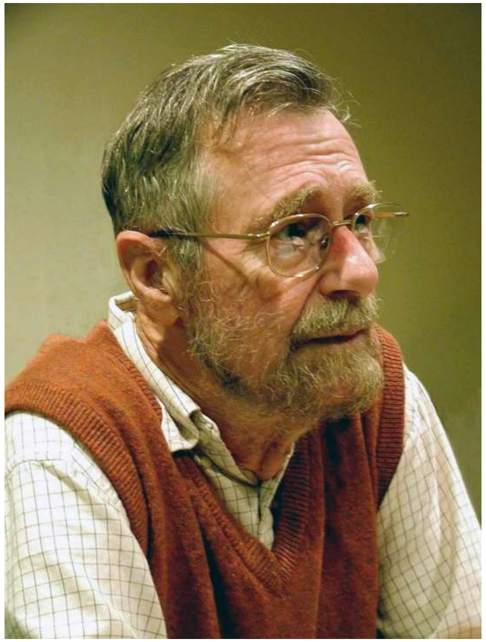
定价问题 (Pricing Problem)

常用算法:

- **Label setting algorithm/ Dynamic programming**

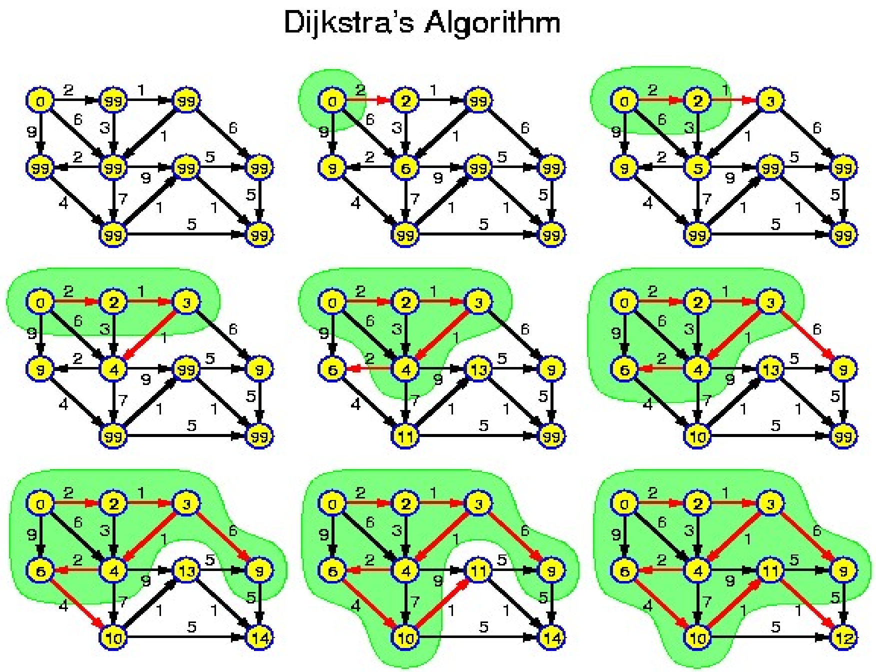
Dijkstra Algorithm

That is, once we scan a node, its labels are set permanently and never changed again.



Professor of Mathematics

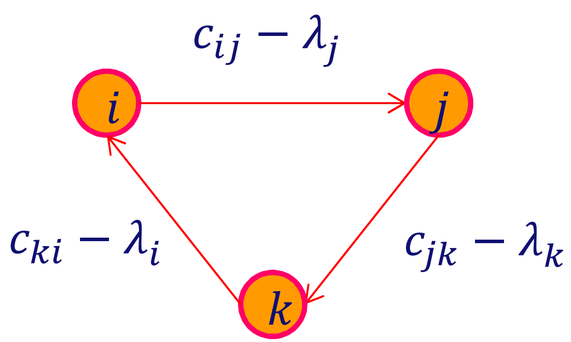
TU Eindhoven (1962-1984)



定价问题 (Pricing Problem)

Dijkstra Algorithm

无法解决带负环的问题
无法解决资源约束问题



Label

$$L_i = \{C_i, \overbrace{q_i, t_i, r_i}^{\textbf{Resources}}, v\}$$

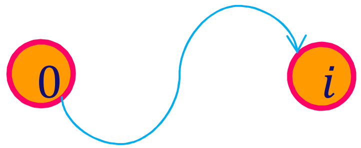
Cost **Position**

$$c_{ij} - \lambda_j + c_{jk} - \lambda_k + c_{ki} - \lambda_i \le 0$$

定价问题 (Pricing Problem)

标签定义：

$$L_i = \{C_i, q_i, t_i, r_i, v_i\}$$

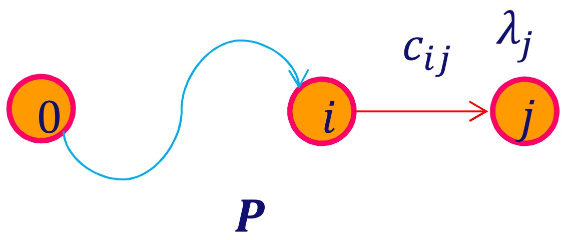


- C_i： **到达当前点** v_i **的 reduced cost；**
- q_i： **到达当前点** v_i **的累计消耗容量;**
- t_i： **到达当前点** v_i **的时间;**
- r_i： **对应路径上所有已访问的客户点集合;**
- v ： **对应路径当前所访问的客户点** v_i**。**

定价问题 (Pricing Problem)

标签扩展 (Label Extension)：

$$L_i \rightarrow L_j$$



- $C_j = C_i + c_{ij} - \lambda_j$
- $q_j = q_i + d_j$
- $t_j = \max(t_i + t_{ij} + s_i, e_j)$
- $r_i = r_i \cup v_j$
- $v \ = v_j$

$$L_i = \{C_i, \boxed{q_i, t_i, r_i}, v_i\} \Longrightarrow L(P) = \{C(P), R(P), V(P)\}$$

定价问题 (Pricing Problem)

择优准则(Dominance Criterion):

- **所有P2的有效扩展都是P1的有效扩展。**
- **P1某个扩展得到完整路径 (P1 $\oplus$ E) 优于P2关于该扩展得到的完整路径 (P2 $\oplus$ E)**

$$C(P1 \oplus E) \le C(P2 \oplus E)$$

- $V(P1) = V(P2)$
- $C(P1) \le C(P2)$
- $R(P1) \le R(P2)$

定价问题 (Pricing Problem)

择优准则(Dominance Criterion):

Less restrictive?

Stronger?

P1

P3

P2

- $V(P1) = V(P2) = V(P3)$
- $C(P1) \leq C(P2) \leq C(P3)$
- $R(P1) \leq R(P2)$? $R(P3)$

$R'(P1) \leq R'(P2) \leq R'(P3)$?

$R'(P1) \leq R'(P3)$?

定价问题 (Pricing Problem)

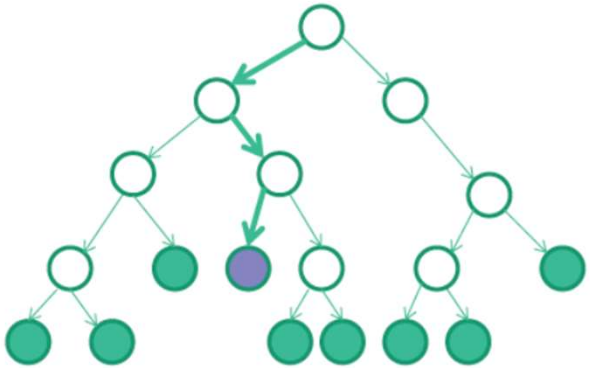
Algorithm 1 Generic label algorithm

GENSPPRC$(G, s, t, d, C, a, b, \tau, T)$
$L \leftarrow \{0, s, 0, 0, 0\}$
$PQ.ENQUEUE(L)$
$SOL \leftarrow \emptyset$
while $PQ.TOP() \neq \emptyset$ **do**
  $L \leftarrow PQ.DEQUEUE()$
  for $e(u, v) \in \delta^+(v(L))$ **do**
    $NewL \leftarrow EXTENDLABEL(L, v, d, C, a, b, \tau, T)$
    if $NewL = NIL$ **then**
      continue
    end if
    if $v = t$ **then**
      $SOL.ENQUEUE(Newl)$
    end if
    if $DOMINATE(NewL, \mathcal{L}_v)$ **then**
      continue
    end if
    $PQ.ENQUEUE(NewL)$
  end for
end while
return SOL

分枝定界(Branch and Bound)

- 由**A. H. Land and A. G. Doig** 在**1960**年首先提出，主要用于解决离散整数规划问题;
- 首先对整数规划问题的整数/**0-1**变量松弛求解;
- 然后将松弛后的线性规划问题分解成若干个子问题并分别求解;
- 同时，不断更新上界与下界，判断可否停止分枝。

Branch-and-Bound



Each node in branch-and-bound is a new MIP

分枝定界(Branch and Bound)

1、先不考虑整数约束，解 (M)IP 的松弛问题：LP ，可能得到以下情况一：

1) 若 LP 没有可行解，则 (M)IP 也没有可行解，停止计算。

2) 若 LP 有最优解，并符合 IP 的整数条件，则 LP 的最优解即为 (M)IP 的最优解，停止计算。

3) 若 LP 有最优解，但不符合 (M)IP 的整数条件，则进行分枝。

分枝定界(Branch and Bound)

2. 分枝：

在 **LP**的最优解 $\mathbf{X}^{(0)}$中，任选一个不符合整数条件的变量构造两个约束条件，并**将这两个约束条件分别加入问题 *(M)IP*，形成两个子问题 *(M)IP1* 和*(M)IP2*，再解这两个问题的松弛问题 *LP1* 和 *LP2*。**

例如 $x_r = 2.5$ 不为整数，构造两个约束条件: $x_r \leq 2$ 和 $x_r \geq 3$。

分枝定界(Branch and Bound)

3. 按照以下两点规则进行修改上、下界：

1) 在各分枝问题中，找出目标函数值最大(小)者作为新的上(下)界；

2) 从已符合整数条件的分枝中，找出目标函数值最大(小)者作为新的下(上)界。

4. 比较与剪枝：

各分枝的目标函数值中，若有小(大)于下(上)界者，则剪掉此枝，表明此子问题已经探清，不必再分枝了; 否则继续分枝。

分枝定界(Branch and Bound)

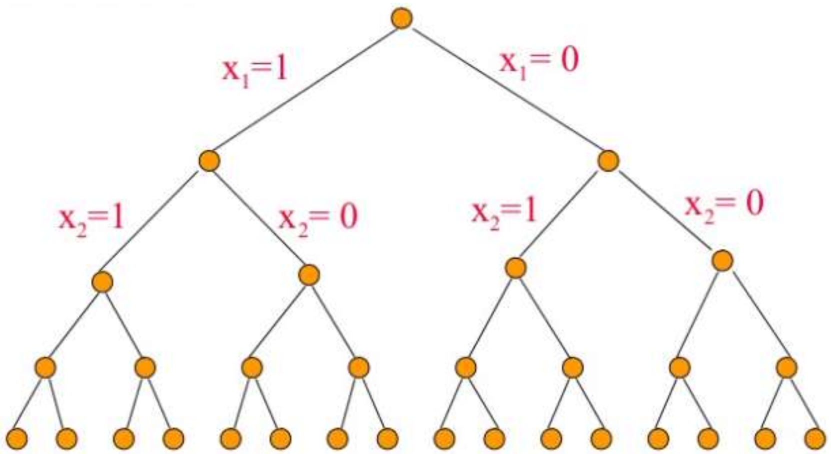
思考：为什么要定界?

假设：

- **我们每秒可以评估1百万个解，**
- **n 代表0-1变量的个数**

求解耗时：

- **n = 30，1 second**
- **n = 40，17 minutes**
- **n = 50，11.6 days**
- **n = 60，31 years**
- **n = 70，31,000 years**

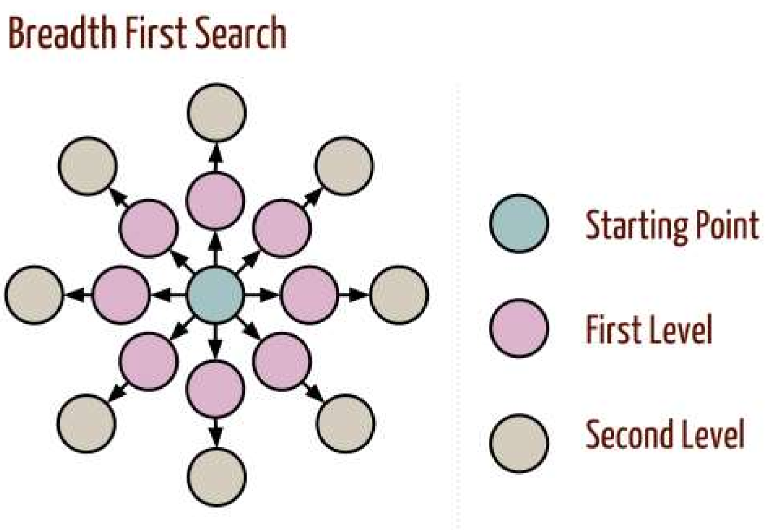


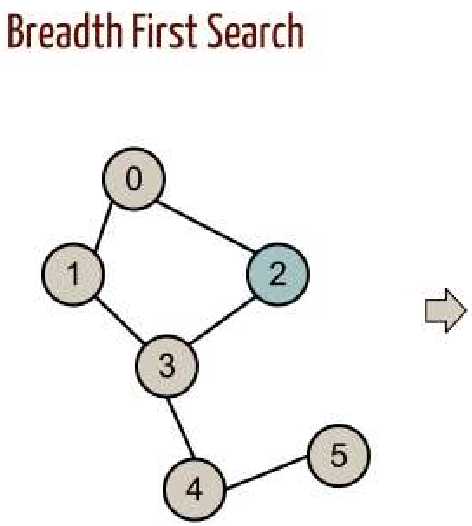
分枝定界(Branch and Bound)

分枝方法：

- **广度搜索 (Breadth-First)**
- **深度搜索 (Deep-First)**
- **最优搜索 (Best-First)**

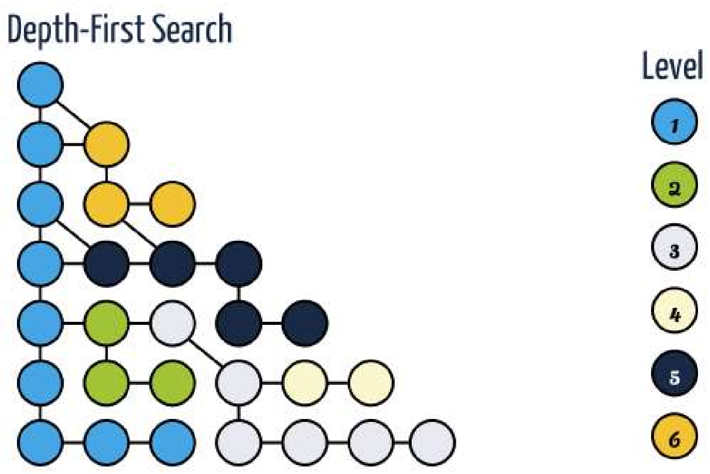
广度搜索(Breadth-First)

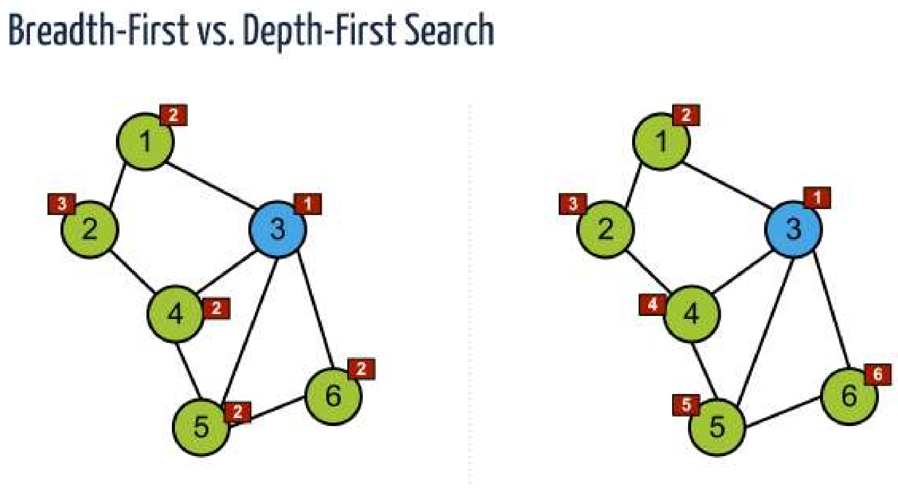





```
1. q = {}
2. q = {2}
3. q = {0, 3}
4. q = {1}
5. q = {4}
6. q = {5}
```

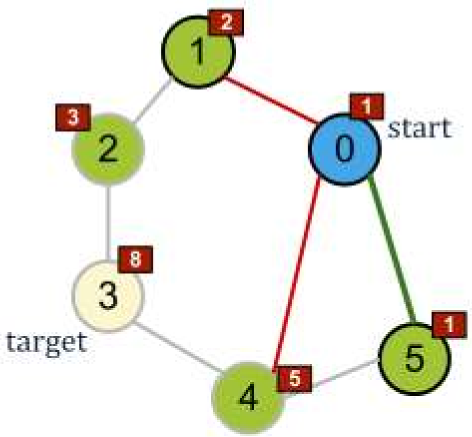
深度搜索(Deep-First)





最优搜索(Best-First)

Best-First Search



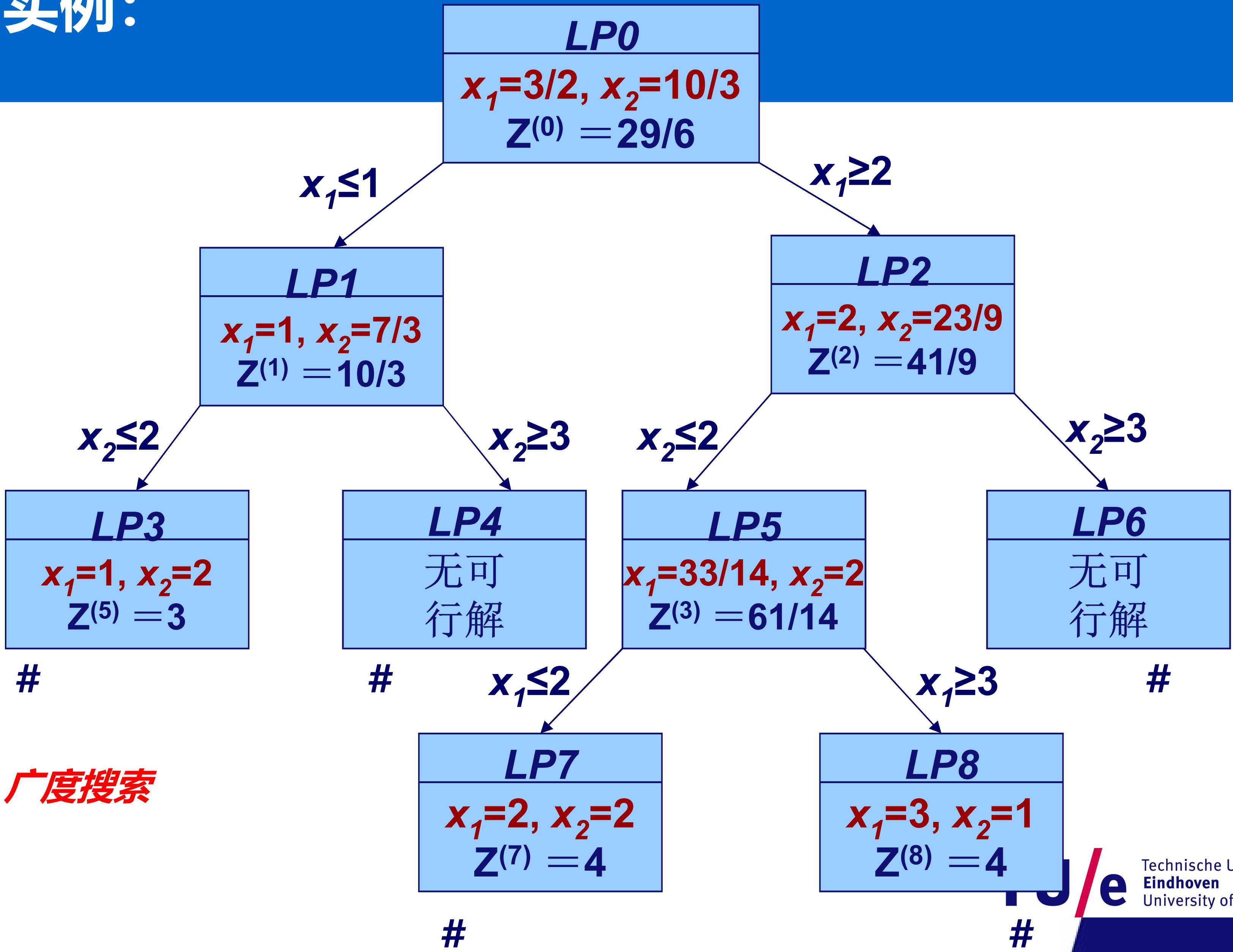

```
Adjacency Matrix
A[0]: [0, 2, 0, 0, 5, 1]
A[1]: [1, 0, 3, 0, 0, 0]
A[2]: [0, 2, 0, 8, 0, 0]
A[3]: [0, 0, 3, 0, 5, 0]
A[4]: [1, 0, 0, 8, 0, 1]
A[5]: [1, 0, 0, 0, 5, 0]
```

实例:

$$
\max Z = x_1 + x_2
$$

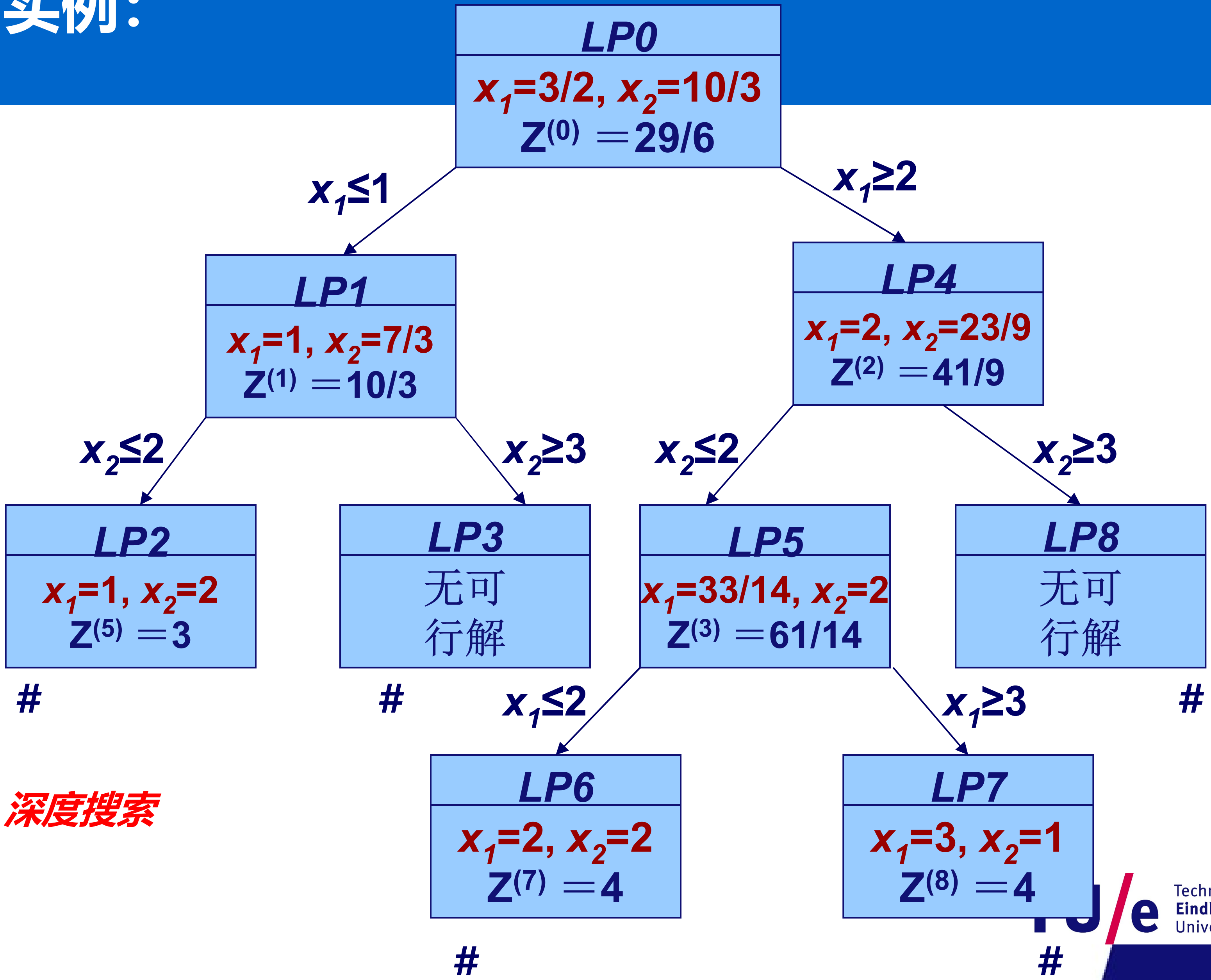
$$
\begin{cases}
14x_1 + 9x_2 \leq 51 \\
-6x_1 + 3x_2 \leq 1 \\
x_1, x_2 \geq 0 \cancel{\text{且全为整数}}
\end{cases}
$$

实例：


LP0
x1=3/2, x2=10/3
Z(0) ＝29/6
x1≤1
x1≥2
LP1
x1=1, x2=7/3
Z(1) ＝10/3
LP2
x1=2, x2=23/9
Z(2) ＝41/9
x2≤2
x2≥3
x2≤2
x2≥3
LP3
x1=1, x2=2
Z(5) ＝3
LP4
无可
行解
LP5
x1=33/14, x2=2
Z(3) ＝61/14
LP6
无可
行解
#
#
x1≤2
x1≥3
#
LP7
x1=2, x2=2
Z(7) ＝4
LP8
x1=3, x2=1
Z(8) ＝4
#
#

广度搜索

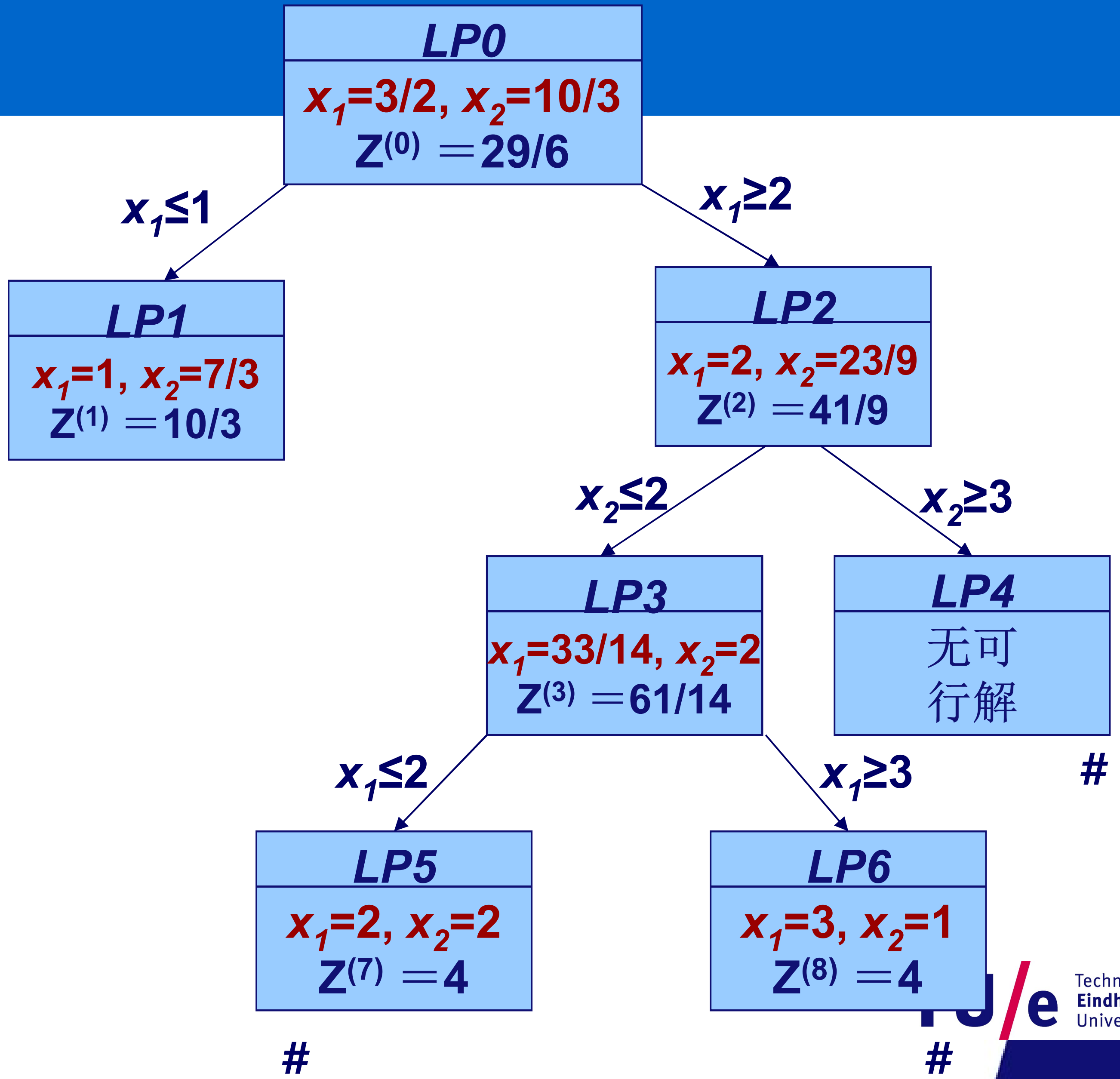
实例：



LP0
$x_1=3/2,\ x_2=10/3$
$Z^{(0)}=29/6$
$x_1\le 1$
$x_1\ge 2$
LP1
$x_1=1,\ x_2=7/3$
$Z^{(1)}=10/3$
LP4
$x_1=2,\ x_2=23/9$
$Z^{(2)}=41/9$
$x_2\le 2$
$x_2\ge 3$
$x_2\le 2$
$x_2\ge 3$
LP2
$x_1=1,\ x_2=2$
$Z^{(5)}=3$
LP3
无可行解
LP5
$x_1=33/14,\ x_2=2$
$Z^{(3)}=61/14$
LP8
无可行解
#
#
$x_1\le 2$
$x_1\ge 3$
#
LP6
$x_1=2,\ x_2=2$
$Z^{(7)}=4$
LP7
$x_1=3,\ x_2=1$
$Z^{(8)}=4$
#
#

深度搜索

实例:


LP0
$x_1=3/2,\ x_2=10/3$
$Z^{(0)} = 29/6$
$x_1 \le 1$
$x_1 \ge 2$
LP1
$x_1=1,\ x_2=7/3$
$Z^{(1)} = 10/3$
LP2
$x_1=2,\ x_2=23/9$
$Z^{(2)} = 41/9$
$x_2 \le 2$
$x_2 \ge 3$
LP3
$x_1=33/14,\ x_2=2$
$Z^{(3)} = 61/14$
LP4
无可行解
#
$x_1 \le 2$
$x_1 \ge 3$
LP5
$x_1=2,\ x_2=2$
$Z^{(7)} = 4$
#
LP6
$x_1=3,\ x_2=1$
$Z^{(8)} = 4$
#

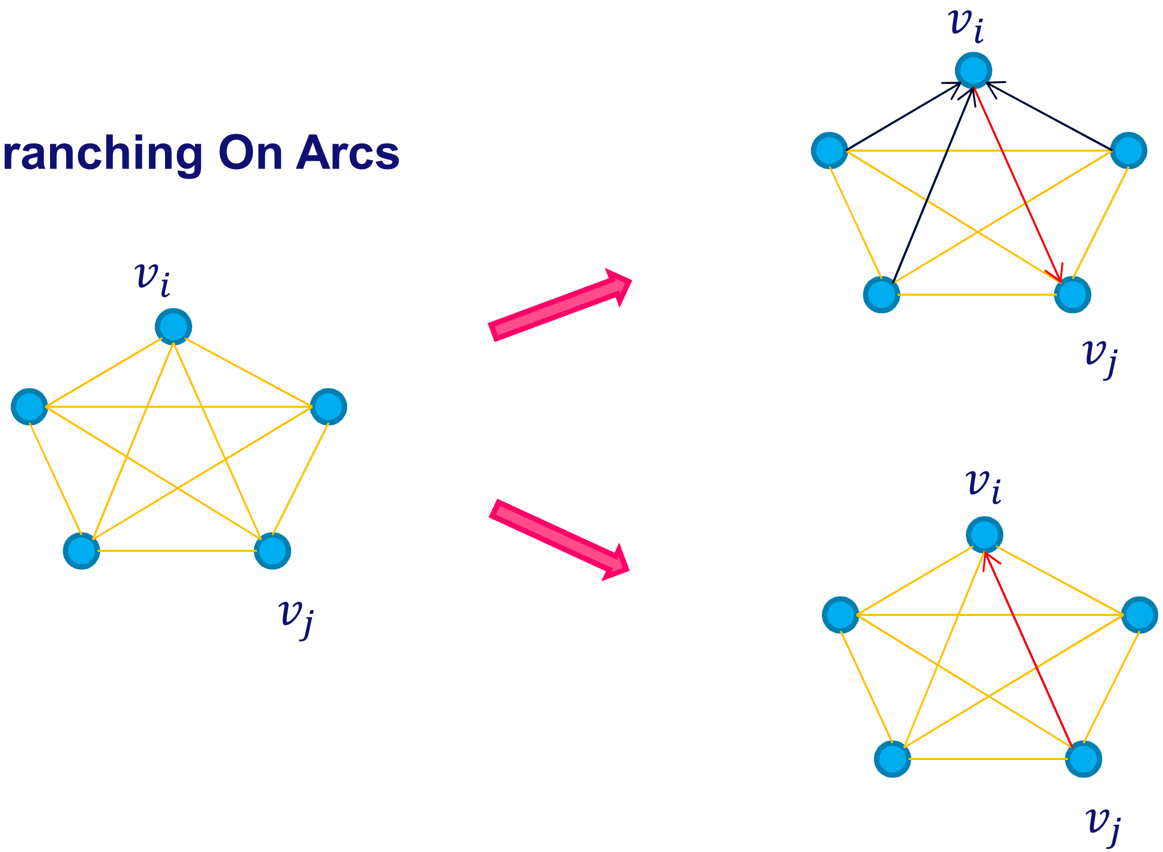
最优搜索

分枝策略(Branching)

Random branching：随机选择一个取值为分数的变量进行分枝。

Most infeasible branching：选择取值最接近0.5的变量作为分枝变量。

Branching On Arcs



$(v_i, v_j) = 1$

$(v_i, v_j) = 0$

分枝策略(Branching)

注意分枝树的平衡！！

Branching On Vehicle numbers

$$\theta_k = 0$$ **Difficult!**

$$\theta_k = 1$$ **Easy**

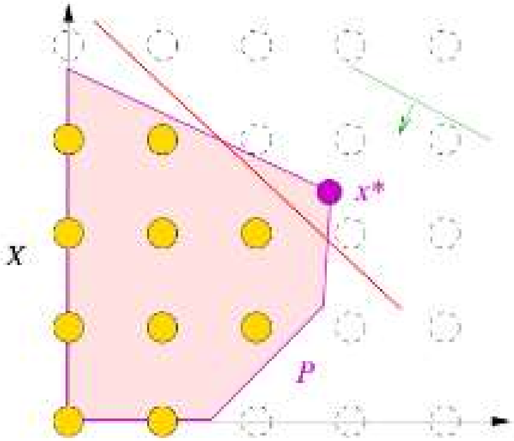
Branching On Vehicle numbers

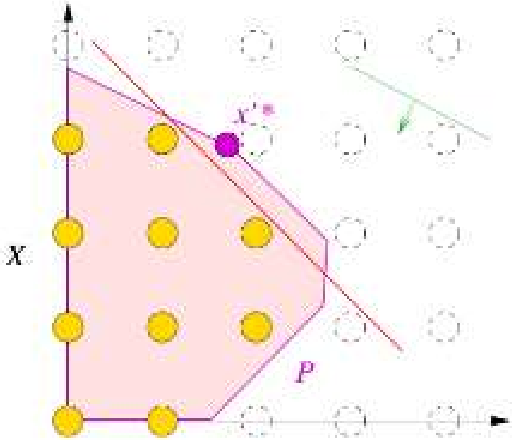
$$\sum_{r_k \in \Omega'} \theta_k \leq S$$

$$\sum_{r_k \in \Omega'} \theta_k \geq S + 1$$

Technische Universiteit
Eindhoven
University of Technology

切平面生成 (Cutting plane generation)





切平面生成 (Cutting plane generation)

$$
\begin{aligned}
&\text{maximize } Z = 4x_1 + 8x_2 + 3x_3 + 10x_4 - x_5 \\
&\text{s.t.} \quad \boxed{3x_1 + 4x_2 + 2x_3 + 5x_4 \leq 7} \\
&\qquad -x_1 + 5x_2 - x_3 + 3x_4 - 2x_5 \leq 4 \\
&\qquad 2x_2 - x_4 - x_5 \leq 1 \\
&\qquad x_1, x_2, x_3, x_4, x_5 \in \{0,1\}
\end{aligned}
$$

Cutting Planes:

$$x_1 + x_4 \leq 1$$

$$x_2 + x_4 \leq 1$$

$$x_1 + x_2 + x_3 \leq 2$$

切平面生成 (Cutting plane generation)

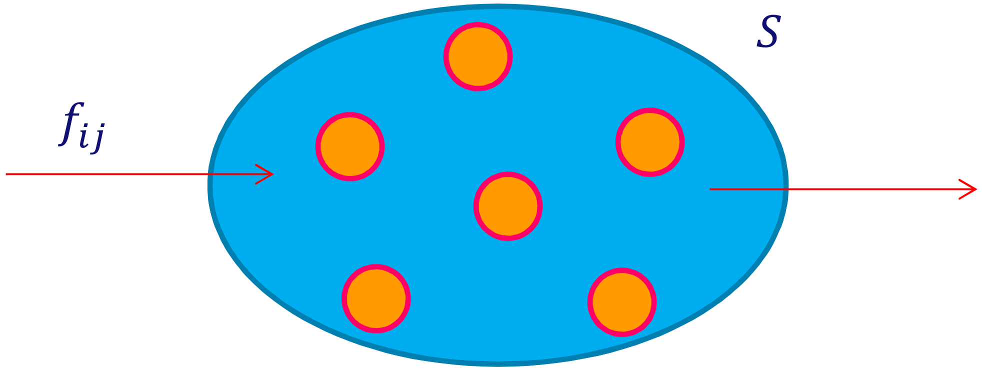
k-path inequalities (Robust cut)

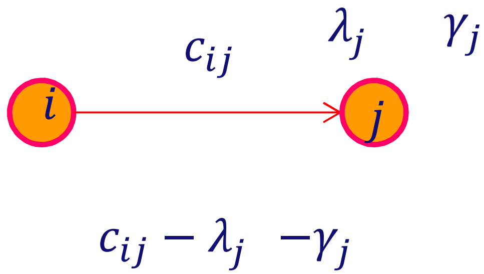
S：A set of vertex set whose demand cannot be satisfied with $k-1$ vehicles;

k：An integer ($1 < k \leq |S|$).

$$\sum_{v_i \in V \setminus S} \sum_{v_j \in S} f_{ij} \geq k \longrightarrow \gamma_j$$

2-path inequalities: $k = 2$





Traveling Salesman Problem with Time Windows (TSPTW)

切平面生成 (Cutting plane generation)

Sub-row inequalities (Non-Robust cut)

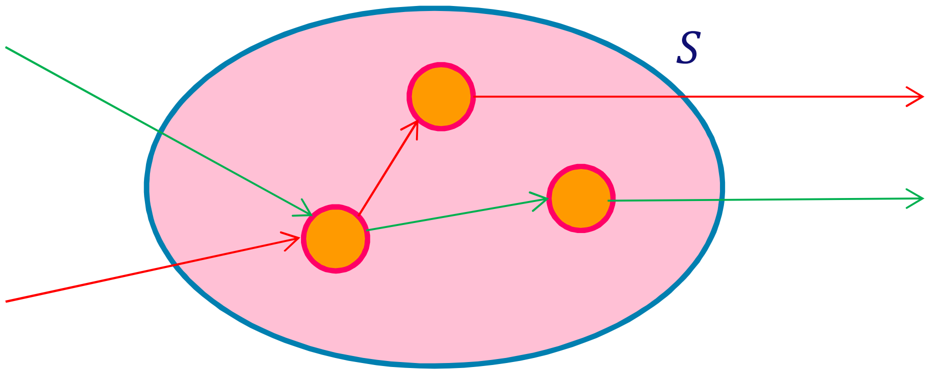
$$\sum_{r\in R}\left\lfloor \frac{1}{k}\sum_{v_i\in S}\alpha_{ir}\right\rfloor \theta_r \geq \left\lfloor \frac{|S|}{k} \right\rfloor \quad \Longrightarrow \quad \sum_{r\in R}\left\lfloor \frac{1}{2}\sum_{v_i\in S}\alpha_{ir}\right\rfloor \theta_r \geq 1$$

φ_j

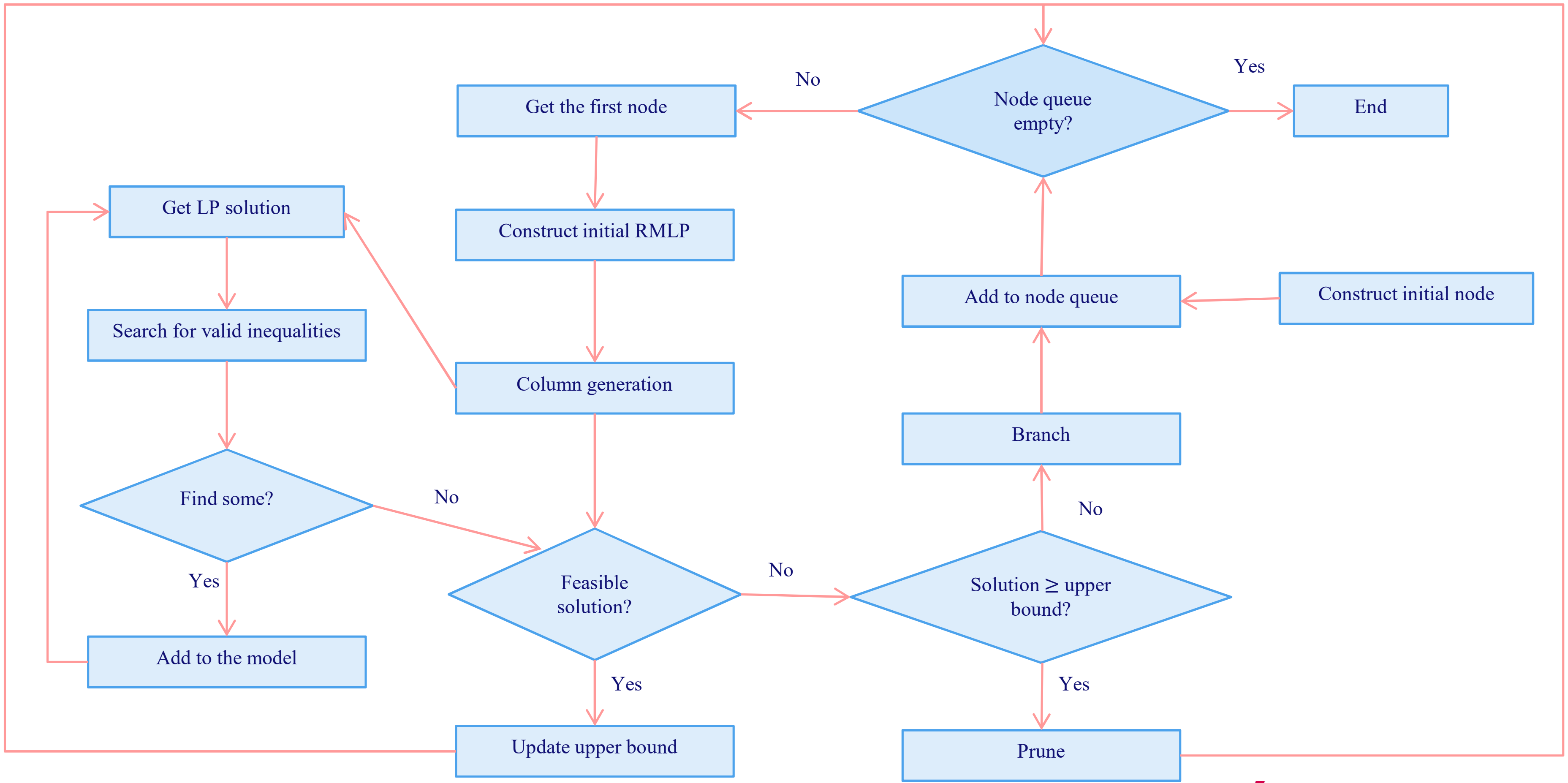
S	A set of vertices	$\|S\|=3$
k	An integer ($1 < k \leq \|S\|$)	$k=2$

$$L_i = \{C_i, q_i, t_i, r_i, v_i, w_i, \dots\}$$

确保所得解里面访问 $|S|$ 中超过1个点的路径不超过一条



小结



Node queue empty?
No
Yes
Get the first node
End
Get LP solution
Construct initial RMLP
Add to node queue
Construct initial node
Search for valid inequalities
Column generation
Branch
Find some?
No
No
Yes
Feasible solution?
No
Solution ≥ upper bound?
Add to the model
Yes
Yes
Update upper bound
Prune

(H. Qin et al.)

常见的加速和改进
(Acceleration & Improvements)

定价问题的可能改进:

- **Q-routes relaxation:**

  只考虑车辆容量限制 Q, 通过相应的Dominance 规则，最多只需要保存 n (Q+1) 个标签。算法复杂度 O(n^2*Q) 。

Arc-load formulation

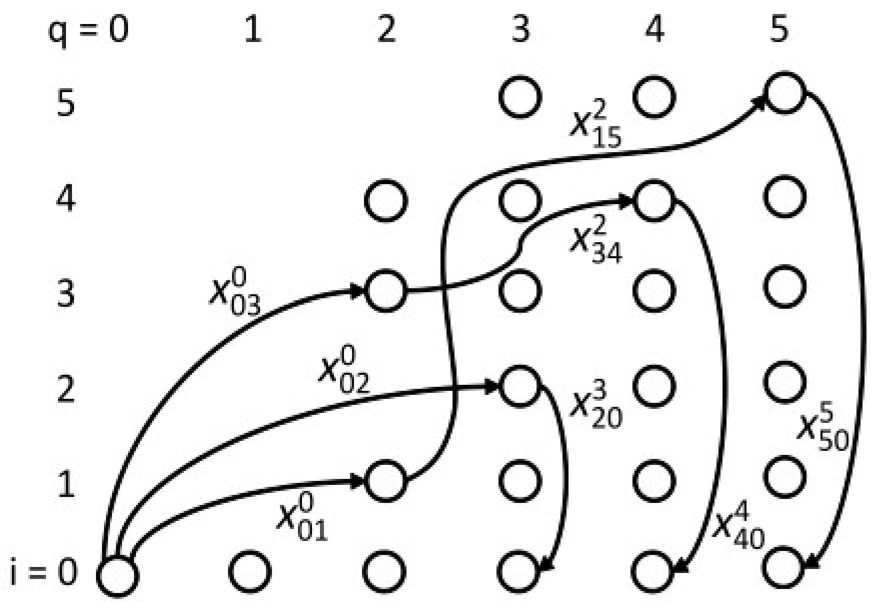
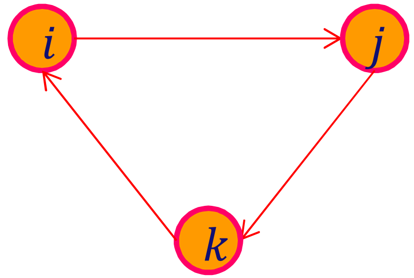


Figure: $n = Q = 5$, $d_1 = d_3 = d_4 = 2$, $d_2 = d_5 = 3$; routes 0-1-5-0, 0-2-0, 0-3-4-0 are shown

客户需求是个整数且不可拆分，车辆的容量消耗是递增非减的变量

常见的加速和改进
(Acceleration & Improvements)



定价问题的可能改进:

- **Q-routes with k-Cycle Elimination:**

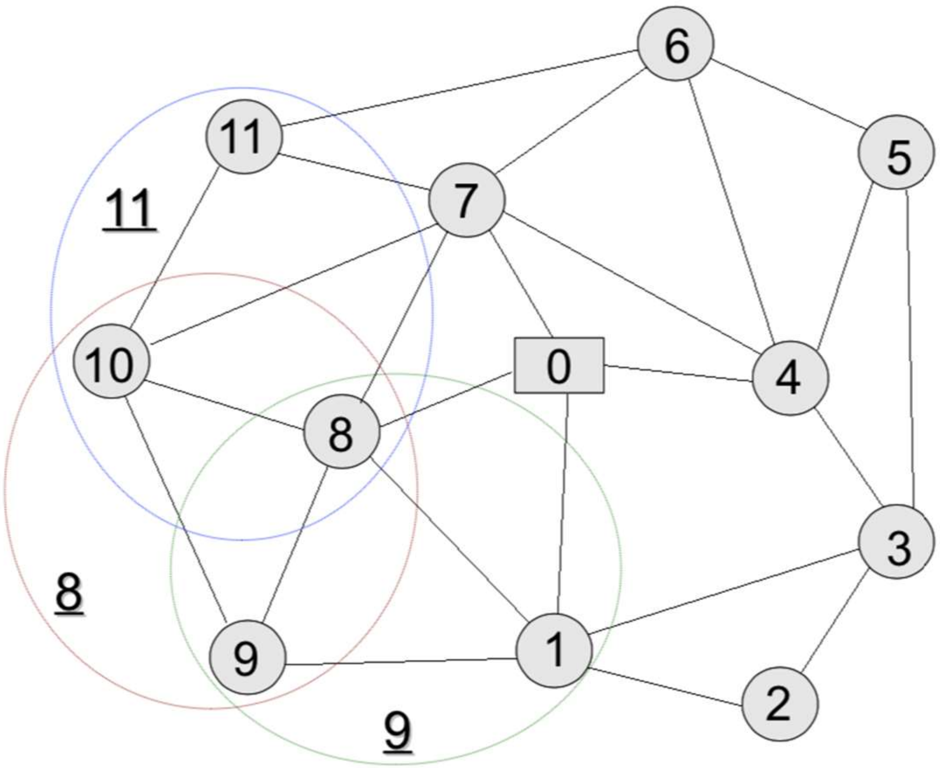
  如果在Q-routes relaxation的基础上增加，每个点只允许访问一次的限制，算法复杂度变为O(2^n*Q)。所以只考虑k-cycle Elimination. 每个点 i 可以被重复访问，但是之间必须至少访问了k 个其他的点。
  (k<=4, Fukasawa et al.)

- **Ng-routes (Baldacci et al.):**

  给每个点 i 建立一个Ng-set. Ng-route 只允许存在关于i 的cycle, 当且仅当cycle 里面访问的点的Ng-sets 不包含 i.

常见的加速和改进 (Acceleration & Improvements)

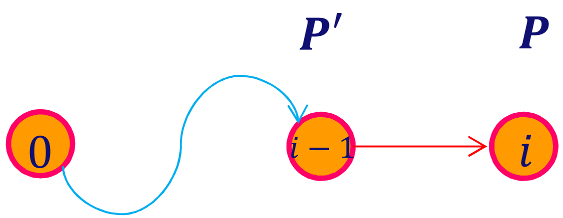
ng-Sets



常见的加速和改进 (Acceleration & Improvements)

- **给定一条路径 P = (v_0, . . . , v_i , . . . , v_p), 用 V(P) 表示 P 所有访问过的节点集合。用 Π(P) 来表示路径 P 不可以扩展的节点集合。**

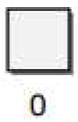
$$\Pi(P) = \{v_i \in V(P): v_i \in \bigcap_{s=i+1}^{p} N_s, i = 1, \dots, p-1\} \cup \{v_p\}$$



$$\Pi(P) = \Pi(P') \cap N_i \cup \{v_p\}$$

常见的加速和改进
(Acceleration & Improvements)

$$N_1 = \{1, 2\},\ N_2 = \{2, 1\},\ N_3 = \{3, 1\}$$



$$\pi_0 = \{\}$$

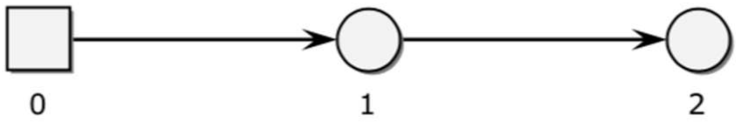
常见的加速和改进 (Acceleration & Improvements)

$$N_1 = \{1,2\},\ N_2 = \{2,1\},\ N_3 = \{3,1\}$$

$$\pi_0 = \{\}$$
$$\pi_1 = \pi_0 \cap N_1 \cup \{1\}$$
$$\pi_1 = \{1\}$$

常见的加速和改进 (Acceleration & Improvements)

$$N_1 = \{1, 2\},\ N_2 = \{2, 1\},\ N_3 = \{3, 2\}$$



$$\pi_1 = \{1\}$$
$$\pi_2 = \pi_1 \cap N_2 \cup \{2\}$$
$$\pi_2 = \{1, 2\}$$

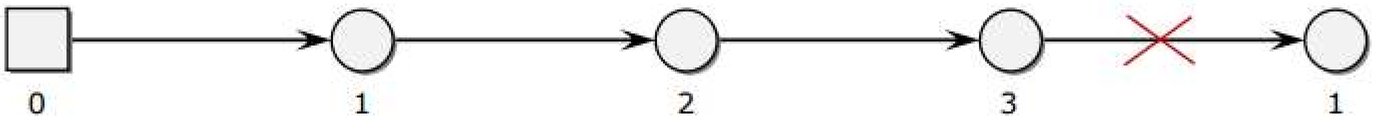
常见的加速和改进 (Acceleration & Improvements)

$$N_1 = \{1,2\},\ N_2 = \{2,1\},\ N_3 = \{3,1\}$$

$$\pi_2 = \{1,2\}$$
$$\pi_3 = \pi_2 \cap N_3 \cup \{3\}$$
$$\pi_3 = \{1,3\}$$

常见的加速和改进 (Acceleration & Improvements)

$$N_1 = \{1, 2\},\ N_2 = \{2, 1\},\ N_3 = \{3, 1\}$$



$$\pi_3 = \{1, 3\}$$

The extension is **not** allowed!

常见的加速和改进 (Acceleration & Improvements)

定价问题的可能改进:

- Stronger Dominance Rules;
- Bi-directional Search;
- Completion Bounds;
- Decremental State Space Relaxation (DSSR)

常见的加速和改进 (Acceleration & Improvements)

列生成的可能改进:

- Dual stabilization (Du Merle et al.);
- Faster pricing heuristics first;
- Dynamically update and control the Column pool;
- Routes enumeration.

常见的加速和改进
(Acceleration & Improvements)

分支方案的可能改进:

- **强分支 (Strong Branching)**
  - A simpler branching over individual edges (Fractional variables);
  - Quick evaluation of 30 candidates and produce ranking. Evaluate the best one fully. Other candidates with good ranking are better evaluated.
  - Collect the previous evaluations of a variable, which could be a good predictors of future evaluations.

常见的加速和改进 (Acceleration & Improvements)

根据问题特性寻找更有效的切平面:

- **Limited memory sub-set cuts (Non-robust cuts)**
- **Benders decomposition?**

实验结果 (D. Pecin et. al)

Class	NP	BMR11 Unsolved	BMR11 Gap	BMR11 Time	Con12 Unsolved	Con12 Gap	Con12 Time	Rop12 Unsolved	Rop12 Gap	Rop12 Time
A	22	0	0.13	30	0	0.07	59	0	0.57	53
B	20	0	0.06	67	0	0.05	89	0	0.25	208
E-M	12	3	0.49	303	2	0.30	2807	2	0.96	44295
F	3	1	0.11	164	1	0.06	3	0	0.25	2163
P	24	0	0.23	85	0	0.13	43	0	0.69	280
Total	81	4			3			2		
Machine		Xeon X7350 2.93GHz			Xeon E5462 2.8GHz			Core i7-2620M 2.7GHz		

Class	NP	This BCP Unsolved	This BCP Gap	This BCP Time
A	22	0	0.03	7
B	20	0	0.04	8
E-M	12	0	0.14	8395
F	3	0	0.28	390
P	24	0	0.08	24
Total	81	0		
Machine		Core i7-3770 3.4GHz		

实验结果 (D. Pecin et. al)

M-n151-k12

Algo	Machine	Root LB	Final LB	Total Time
BMR11	X7350 2.93GHz	1004.3	1004.3	380
Contardo12	E5462 2.8GHz	1012.5	**1015**	19699
Ropke12	i7-2620M 2.7GHz	1001.5	**1015**	417146
This BCP	i7-3770 3.4GHz	1013.7	**1015**	349

实验结果 (D. Pecin et. al)

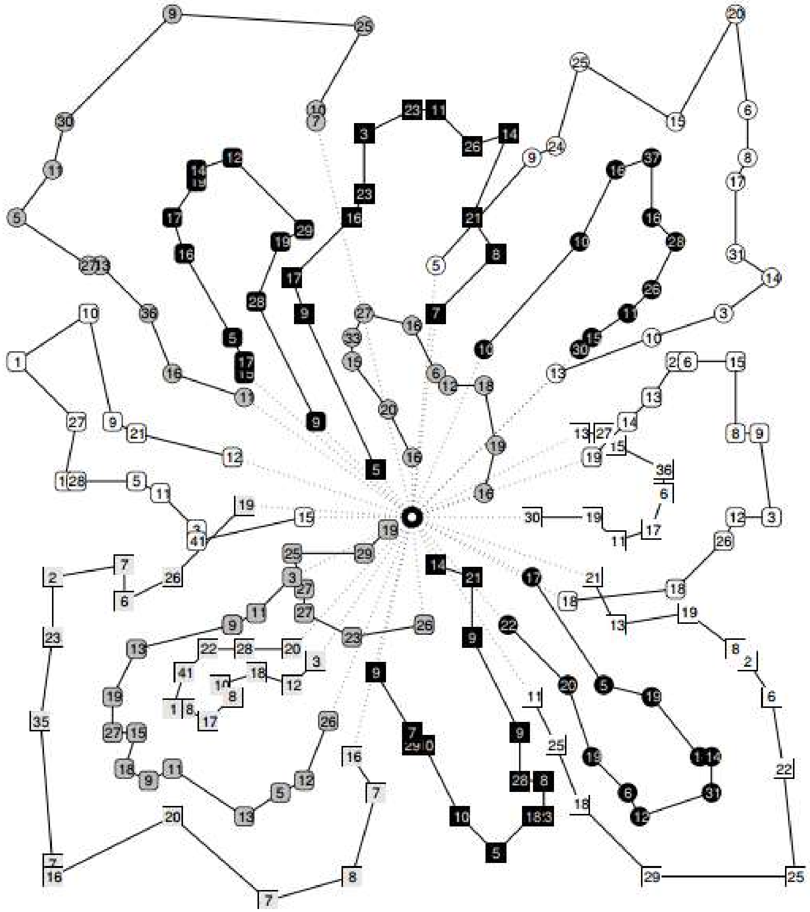
M-n200-k16

Algo	Machine	Root LB	Final LB	Total Time
BMR11	X7350 2.93GHz	1256.6	1256.6	319
Contardo12	E5462 2.8GHz	1263.0	1263.0	265588
Ropke12	i7-2620M 2.7GHz	1253.0	1258.2	7200
This BCP	i7-3770 3.4GHz	1266.0	**1274**	89772

Previous upper bound: 1278.

实验结果 (Diego Pecin et. al)

M-n200-k16

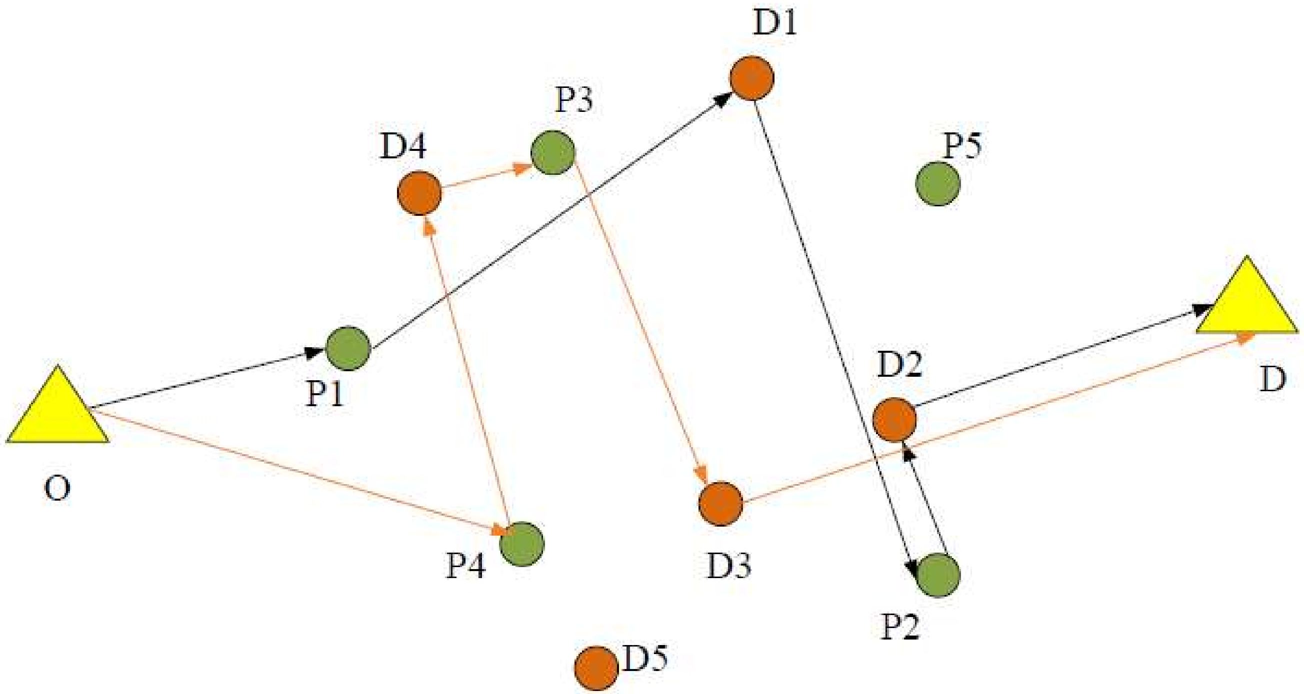


TU/e Technische Universiteit Eindhoven University of Technology

个人实践

Time-dependent Pickup and Delivery Problem with Time Windows

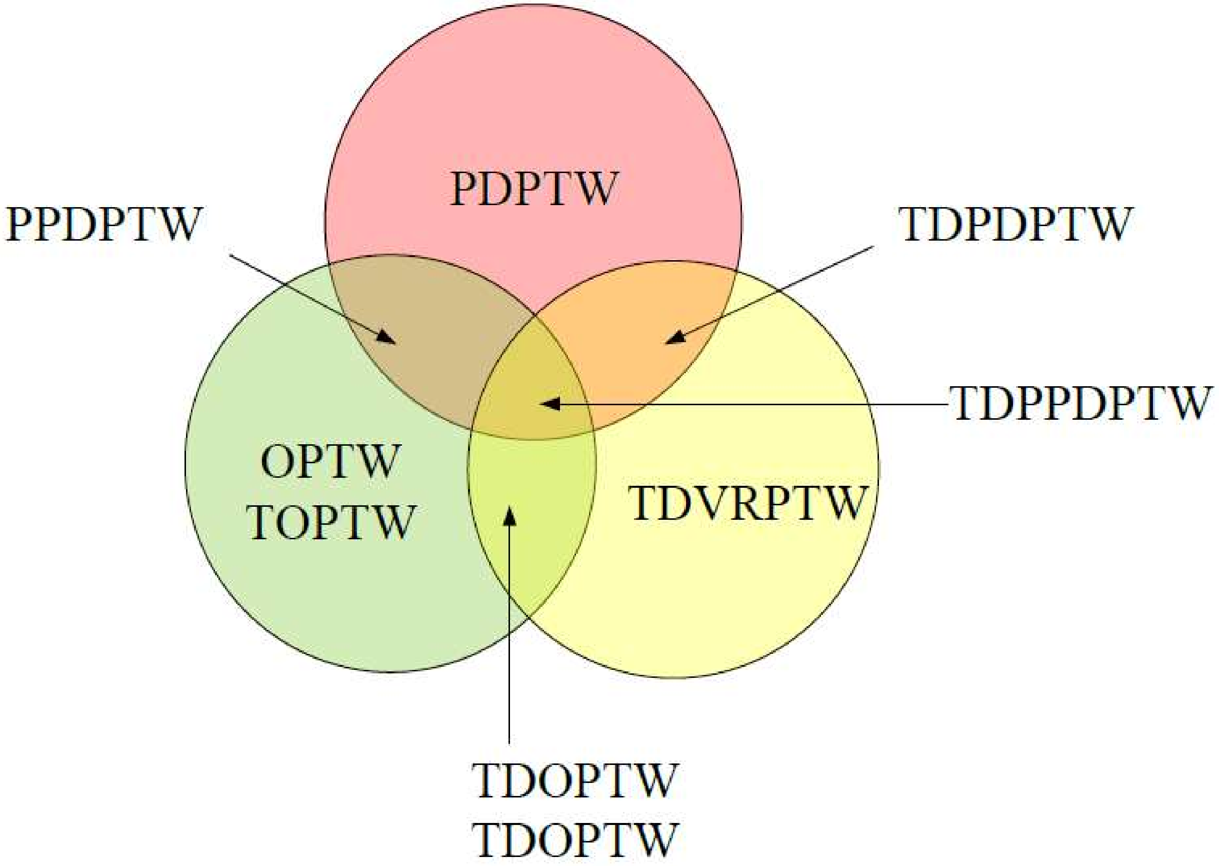
- **Select a set of routes which maximize the difference between the total collected profits and the traveling cost.**



研究意义

- Traffic congestion on the highways or crowed urban traffic situations.
- Time-dependent travel times in daily routing problem.

相关研究


PDPTW
PPDPTW
TDPDPTW
TDPPDPTW
OPTW
TOPTW
TDVRPTW
TDOPTW
TDOPTW

四个问题

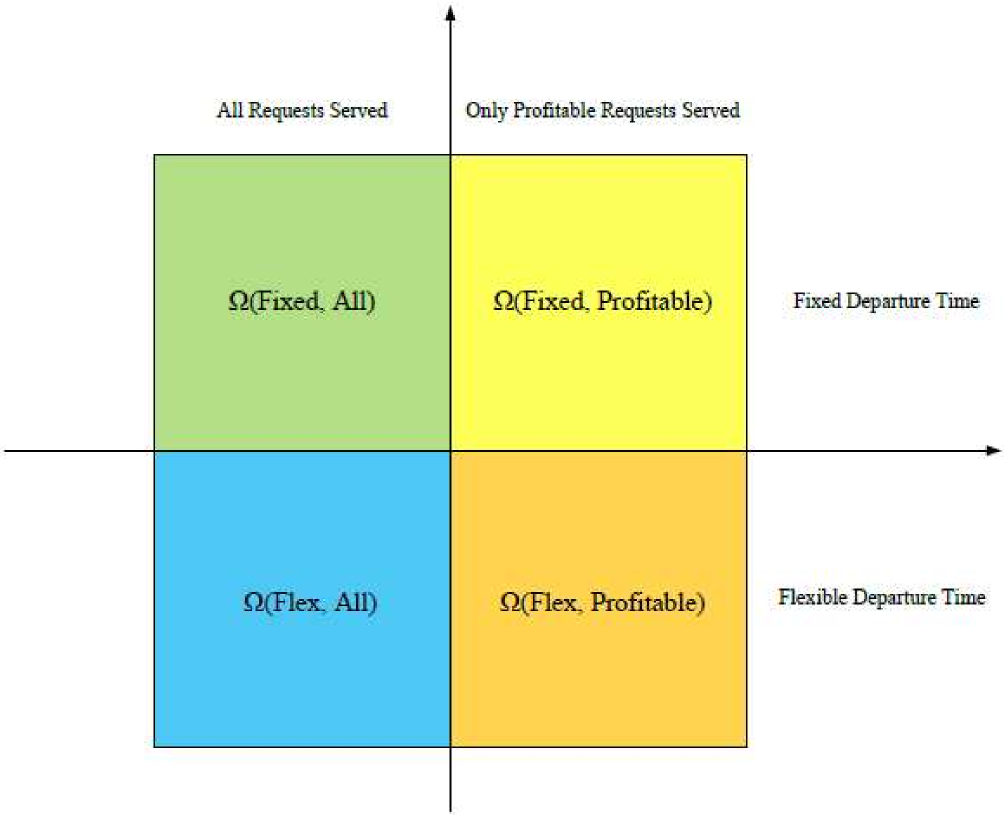


Figure 1 An illustration of the time-dependent pickup and delivery problem with time windows and its variants

时变性 (Time-dependent)

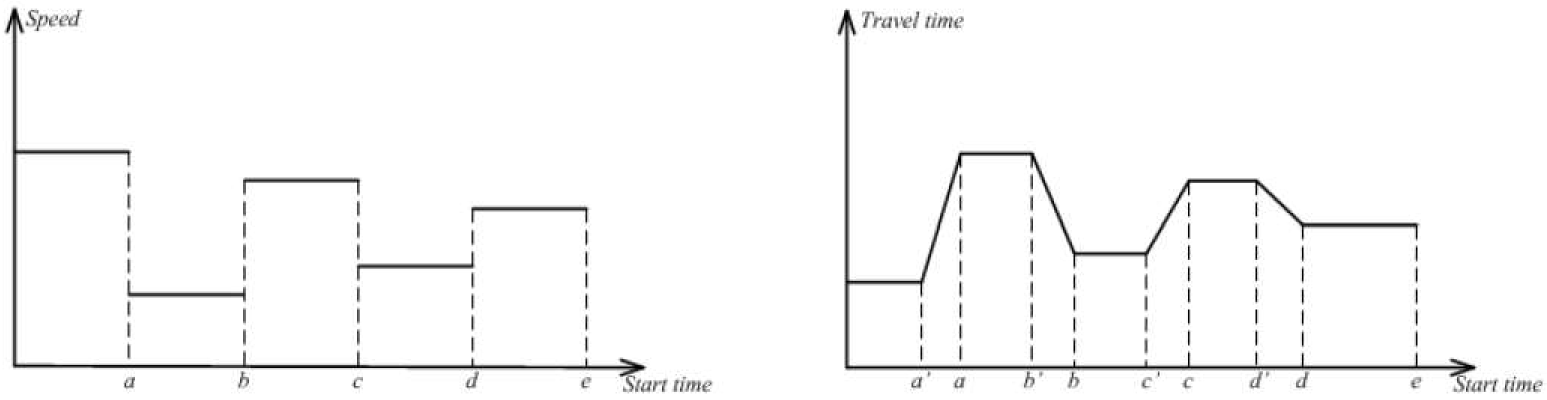


Figure 1 Speed and travel time functions.

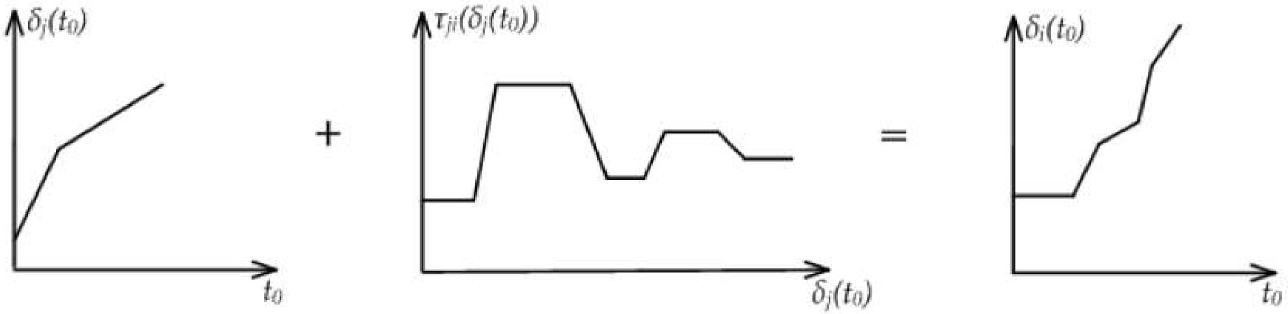
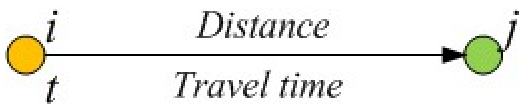


Figure 2 Arrival time functions.



TU/e Technische Universiteit Eindhoven University of Technology

使用的加速技巧

- **Stronger & less restrictive dominance criterion**

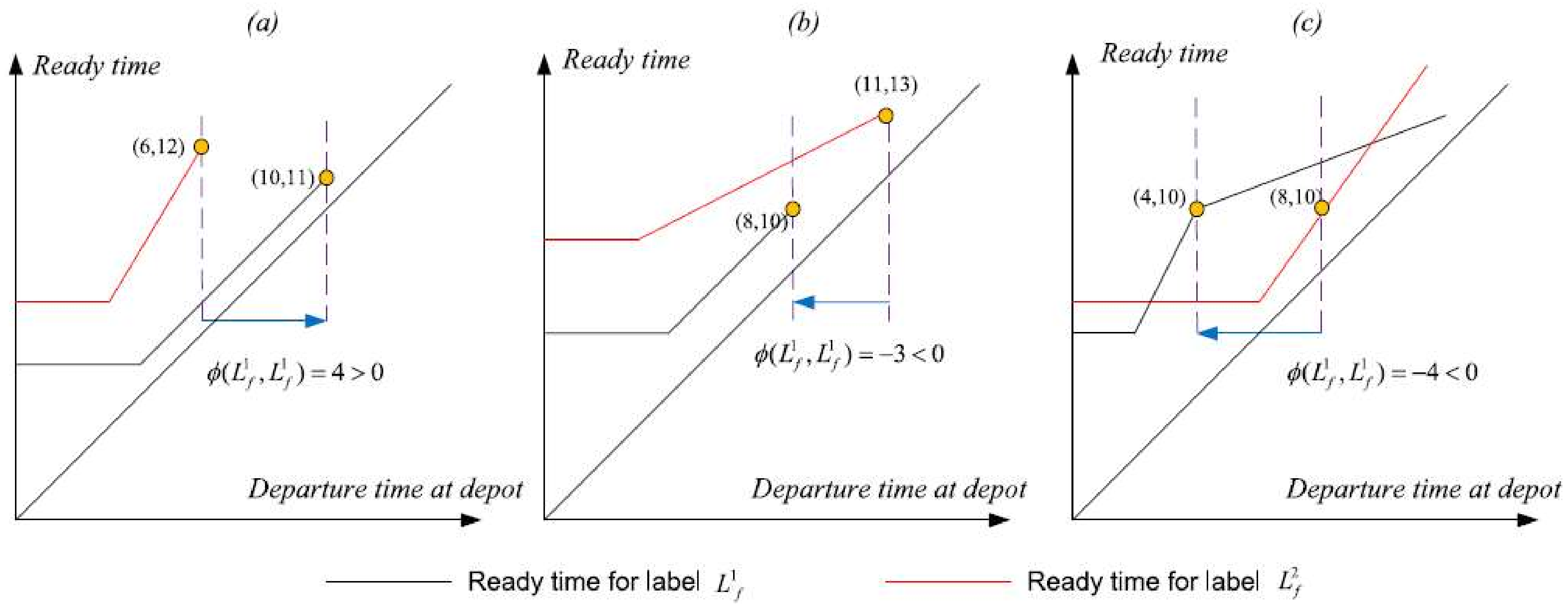


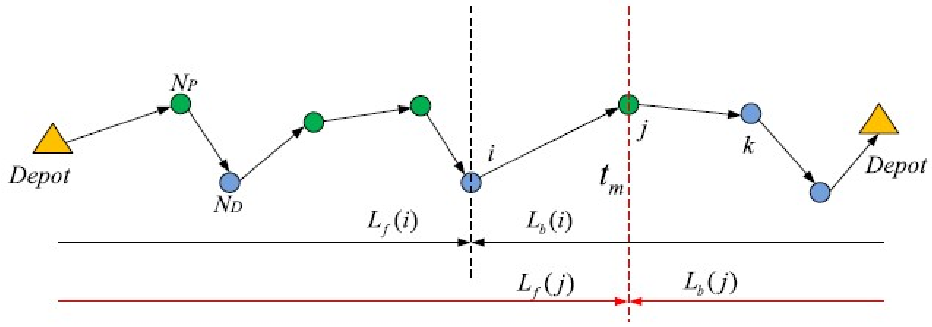
Fig. 3. Illustration of $\phi(L^1_f, L^2_f)$.

TU/e Technische Universiteit Eindhoven University of Technology

使用的加速技巧

- **Bi-directional label search**

Instance	Optimal value	Requests served	Processing time (seconds) Monodirectional	Bidirectional
AA30	266.72	16	2.37	2.12
AA35	278.14	20	4.68	5.04
AA40	341.74	22	7.94	5.82
AA45	361.57	21	23.54	5.6
AA50	430.41	23	22.40	9.53
AA55	436.05	27	32.79	18.66
AA60	505.02	26	87.11	24.71
AA65	576.39	29	182.09	34.80
AA70	606.68	29	180.01	39.14
AA75	715.53	31	1038.67	257.60
BB30	274.45	21	2.36	2.78
BB35	337.55	22	1.30	1.03
BB40	326.82	23	8.30	9.95
BB45	378.90	23	34.88	27.94
BB50	432.10	25	31.81	37.41
BB55	530.84	27	40.14	59.02
BB60	558.02	28	164.38	119.2
BB65	547.84	25	353.15	332.13
BB70	558.29	27	225.79	351.57
BB75	613.35	27	775.58	511.37
CC30	316.39	20	32.51	27.64
CC35	386.26	24	133.15	104.18
CC40	464.28	25	2934.28	942.27
CC45	497.68	26	20641.34	3422.21
CC50	519.60	26	45532.40	13913.43
CC55	581.50	28	156770.32	28429.25
CC60	624.95	30	≥ 2 weeks	52801.65
CC65	663.31	31	≥ 2 weeks	699831.05
DD30	343.65	21	396.23	80.42
DD35	410.19	22	2815.31	338.16
DD40	490.92	25	53402.91	4618.11
DD45	540.17	27	67293.53	7667.97
DD50	610.07	30	186697.41	21889.86
DD55	639.75	29	≥ 2 weeks	823755.21



P. Sun, L. P. Veelenturf, S. Dabia, T. V. Woensel, The Time-Dependent Capacitated Profitable Tour Problem with Time windows and Precedence Constraints. European Journal Operational Research, 264(3): 1058-1073, 2018.

Technische Universiteit Eindhoven University of Technology

使用的加速技巧

- **Route enumeration**
- **Limited-memory subset-row cuts**

Table 5
Performance of exact framework with and without enhancements.

	Ω(*Fix, All*)				Ω(*Fix, Prof*)			
Options	*#Better*	*#Equal*	*#Worse*	Overall	*#Better*	*#Equal*	*#Worse*	Overall
BP + 1	1	7	5	−4	0	1	4	−4
BP + 2	4	5	4	0	0	0	5	−5
BP + 1 + 2	5	5	3	2	0	0	5	−5
	Ω(*Flex, All*)				Ω(*Flex, Prof*)			
Options	*#Better*	*#Equal*	*#Worse*	Overall	*#Better*	*#Equal*	*#Worse*	Overall
BP + 1	2	3	8	−6	1	4	6	−5
BP + 2	6	3	4	2	2	4	5	−3
BP + 1 + 2	9	3	1	8	4	3	4	0

1: Route enumeration. 2: Limited-memory subset-row cuts.

P. Sun, L. P. Veelenturf, M. Hewitt, T. V. Woensel, The time-dependent pickup and delivery problems with time windows. Transportation Research Part B: Methodological, 116: 1-24, 2018.

Technische Universiteit
Eindhoven
University of Technology

总结

- Danzig-Wolfe Decomposition?
- 构造有效快速的(最短路径)算法求解定价问题;
- 构造有效的分枝技巧;
- 增加强而有效的切平面生成。

延伸阅读：

- Bertsimas, D. and Tsitsiklis, J.N., 1997. ***Introduction to linear optimization*** (Vol. 6, pp. 479-530). Belmont, MA: Athena Scientific.

- Wolsey, L.A., 1998. ***Integer programming*** (Vol. 52). John Wiley & Sons.

- Desaulniers, G., Desrosiers, J. and Solomon, M.M. eds., 2006. ***Column generation*** (Vol. 5). Springer Science & Business Media.

- Toth, P. and Vigo, D. eds., 2014. ***Vehicle routing: problems, methods, and applications***. Society for Industrial and Applied Mathematics.

- Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W. and Vance, P.H., 1998. ***Branch-and-price: Column generation for solving huge integer programs***. *Operations research*, *46*(3), pp.316-329.

谢谢！

& 提问？

Technische Universiteit
Eindhoven
University of Technology