现代管理科学方法
（第8讲）

郭仁拥 博士/教授/博导

讲授内容

1. 求解TSP的构造启发式方法
2. 求解TSP的改进启发式方法

1. 求解TSP的构造启发式方法

求解TSP的构造启发式方法：

- 最近邻域算法（Nearest Neighbour algorithm）
- 插入算法（Insertion algorithm）
- 修补/合并算法（Patching/Merger algorithm）
- 最小生成树算法（Minimum spanning tree algorithm）
- 克里斯托费兹算法（Christofides’ algorithm）

TSP启发式方法的执行保障

针对一般情形的坏消息

定理：设 A 是一个启发式算法。对于一个TSP算例 l，$A(l)$ 是由算法 A 得到的车辆旅行长度，$Opt(l)$ 是最优车辆旅行长度。给定一个常数 $1 \le r < \infty$，如果对于任一算例 l，有 $A(l) \le r \cdot Opt(l)$，则 $P = NP$。

这意味着：对于一般TSP情形，不能找到一个具有固定执行保障的启发式算法

对于满足三角不等式距离度量的STSP，该定理不成立

NP完全问题(NP-C问题)，是世界七大数学难题之一。

NP的英文全称是Non-deterministic Polynomial Complete的问题，即多项式复杂程度的非确定性问题。简单的写法是 NP=P？，问题就在这个问号上，到底是NP等于P，还是NP不等于P。

P类问题：所有可以在多项式时间内求解的判定问题构成P类问题。

NP类问题：所有的非确定性多项式时间可解的判定问题构成NP类问题。

最近邻域算法（NN）

从一个顶点v（例如仓库）开始，设置$k=1$和$U=\{v\}$（U是访问顶点集合）

当$k<n$

- 选择下一个访问顶点w，使得$c_{vw}=\min\limits_{h\notin U} c_{vh}$
- 更新$U=U\cup\{w\}$，设置当前顶点$v=w$和$k=k+1$

一类贪婪启发式算法

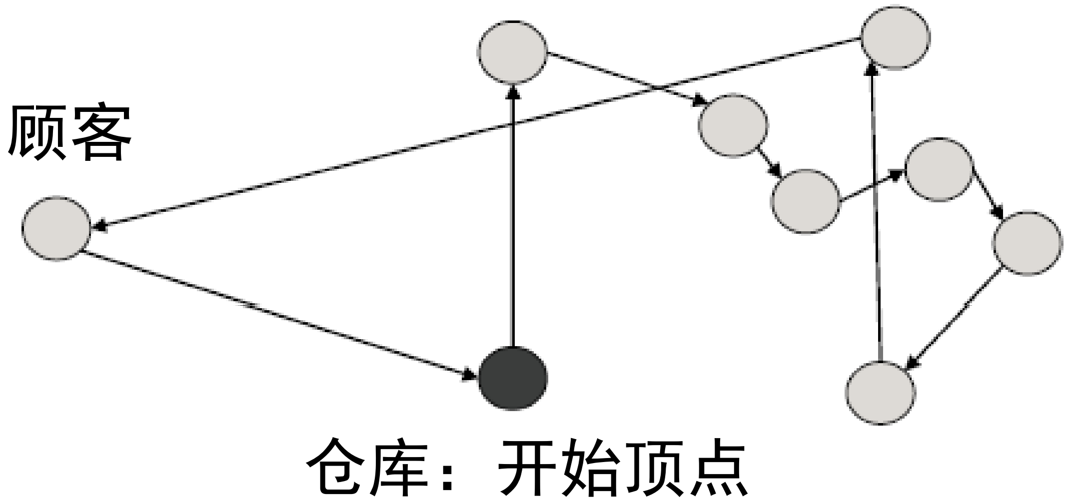
缺陷：算法仅仅在前面迭代中选择较短弧，后面迭代中的弧一般较长

算法解（平均意义上）比最优解大25%

计算复杂性：$O(n^2)$（对于超过1万个顶点的问题计算速度太慢）

该算法可以为改进启发式算法提供一个较好初始解

一个例子：欧式距离矩阵



一个关于NN算法的定理：

对于每个 $r>0$，存在 n 个顶点的算例 I（满足三角不等式），对于任意大的 n，有 $NN(I) \geq r \cdot Opt(I)$。

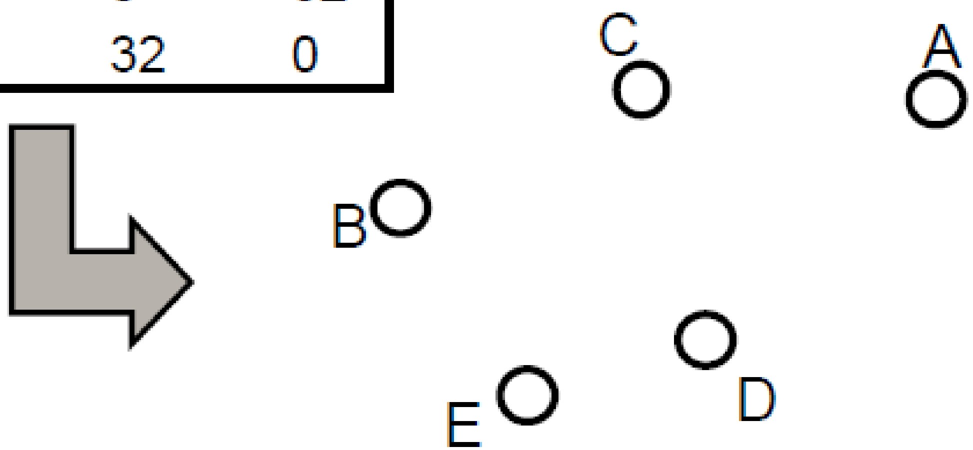
插入算法（IA）

- **基本算法：**从一个部分圈开始；在部分圈中插入新的顶点，直到所有顶点被考虑
- **插入规则（平均误差）：**最近顶点（20%）；最远顶点（10%）；随机（11%）

一个例子（最远顶点规则）

对称距离矩阵

	A	B	C	D	E
A	0	85	47	57	87
B	85	0	43	52	38
C	47	43	0	48	58
D	57	52	48	0	32
E	87	38	58	32	0



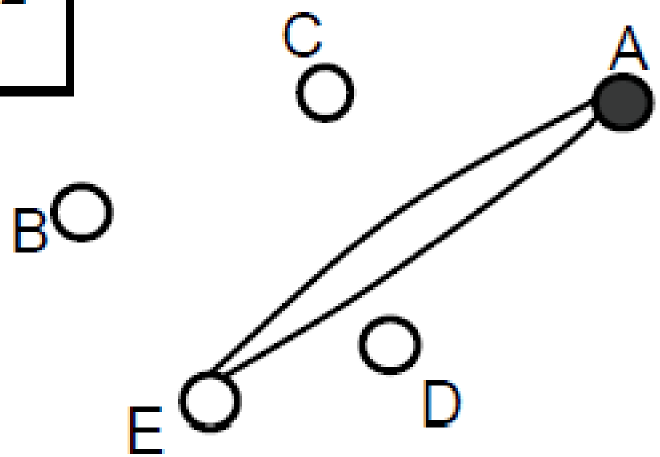
第1步：从顶点A（仓库）开始

对称距离矩阵

	A	B	C	D	E
A	0	85	47	57	87
B	85	0	43	52	38
C	47	43	0	48	58
D	57	52	48	0	32
E	87	38	58	32	0

⇦ the farthest

length(T)=174



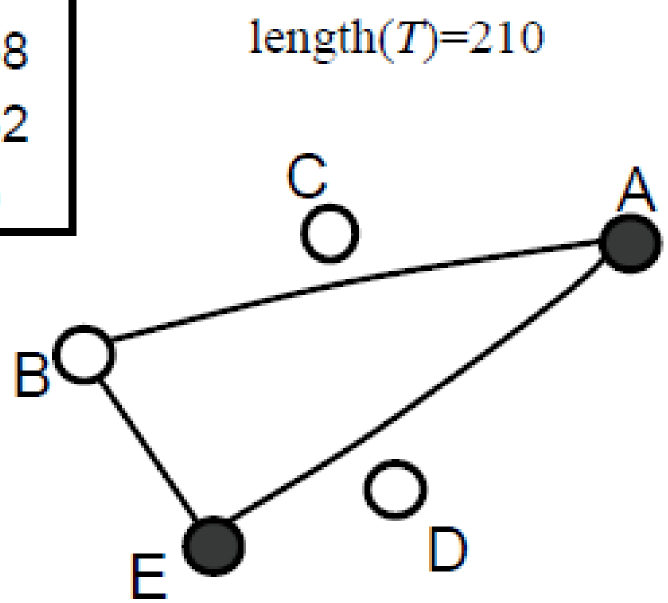
第2步：从A或E最远的顶点

对称距离矩阵

	A	B	C	D	E
A	0	85	47	57	87
B	85	0	43	52	38
C	47	43	0	48	58
D	57	52	48	0	32
E	87	38	58	32	0

$\mathrm{Tour}(T,k)$：选择旅行T中边(i,j)的过程，其中插入顶点k满足

$$\arg\min_{(i,j)}\left(c_{ik}+c_{kj}-c_{ij}\right)$$

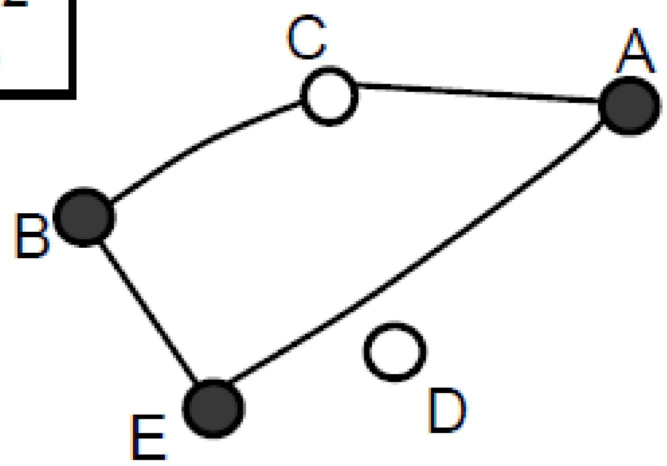


第3步：从A或E或B最远的顶点

对称距离矩阵

	A	B	C	D	E
A	0	85	47	57	87
B	85	0	43	52	38
C	47	43	0	48	58
D	57	52	48	0	32
E	87	38	58	32	0

length(T)=215



Tour(T, C):

(A, B): $\Delta c = 47 + 43 - 85 = 5$

(B, E): $\Delta c = 43 + 58 - 38 = 63$

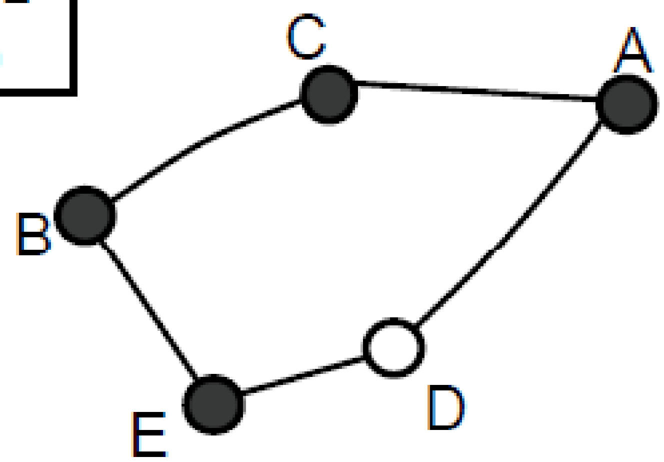
(E, A): $\Delta c = 58 + 47 - 87 = 18$

第4步：确定在哪插入D

对称距离矩阵

	A	B	C	D	E
A	0	85	47	57	87
B	85	0	43	52	38
C	47	43	0	48	58
D	57	52	48	0	32
E	87	38	58	32	0

length(T)=217



$\text{Tour}(T, E)$

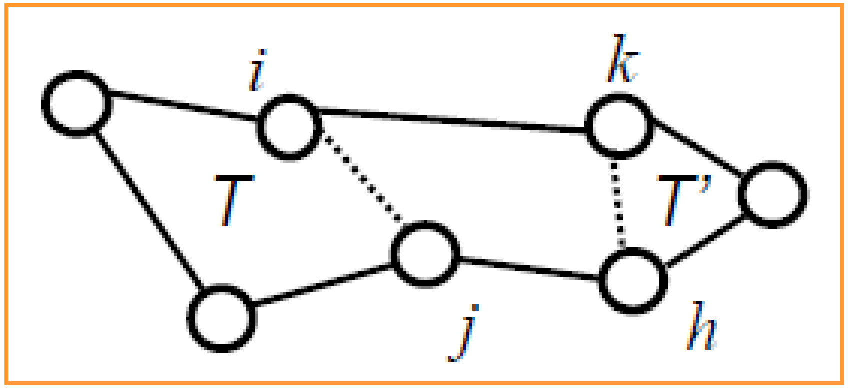
(A, C): $\Delta c = 57 + 48 - 47 = 58$

(C, B): $\Delta c = 48 + 52 - 43 = 57$

(B, E): $\Delta c = 52 + 32 - 38 = 46$

(E, A): $\Delta c = 32 + 57 - 87 = 2$

修补/合并算法（PMA）



合并

从 n 个单顶点旅行开始

当旅行数 >1

- 选择一对将被合并的旅行 (T,T')，使得

  $$\min\{c_{ij} : i \in T, j \in T'\}$$

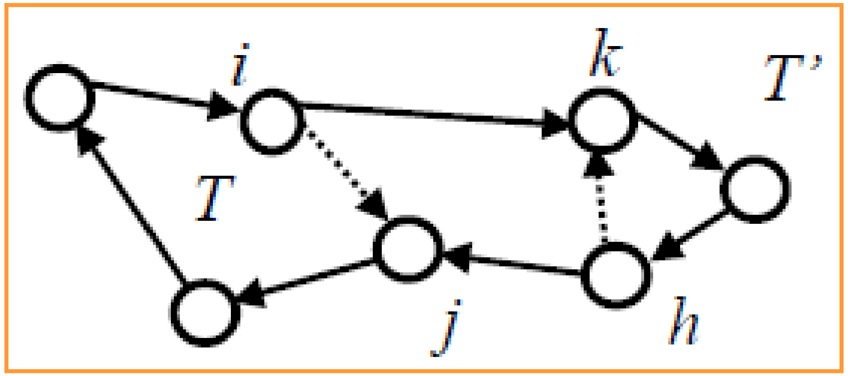
- 合并选定的旅行：如果 T 或 T' 是单顶点旅行，则Tour (T,k') 或Tour (T',k)；否则，确定 $(i,j) \in T$ 和 $(k,h) \in T'$，使得 $c_{ik} + c_{hj} - c_{ij} - c_{kh}$ 最小

修补

从ATSP的分配问题（AP）松弛所产生的子闭迹开始

当旅行数 >1

- 选择一对将被合并的旅行 (T,T')，确定 $(i,j)\in T$ 和 $(k,h)\in T'$，使得 $c_{ik}+c_{hj}-c_{ij}-c_{kh}$ 最小
- 合并选定的旅行



基于最小生成树（MST）的算法

基础

给定一个完全无向图 G，一个Hamiltonian路径（HP）是 G 中一个树，其费用不少于最小生成树（MST）费用 c

最小生成树的计算量是 $O(n^2)$

一个Hamiltonian圈（HC）是一个添加了一条边的HP

最小费用HC的费用表示为 $c(TSP)$，则有 $c(MST) \le c(TSP)$

一个圈（可能访问某一顶点多次）可由一个MST产生，通过一个翻倍所有MST边的过程

深度优先的遍历过程

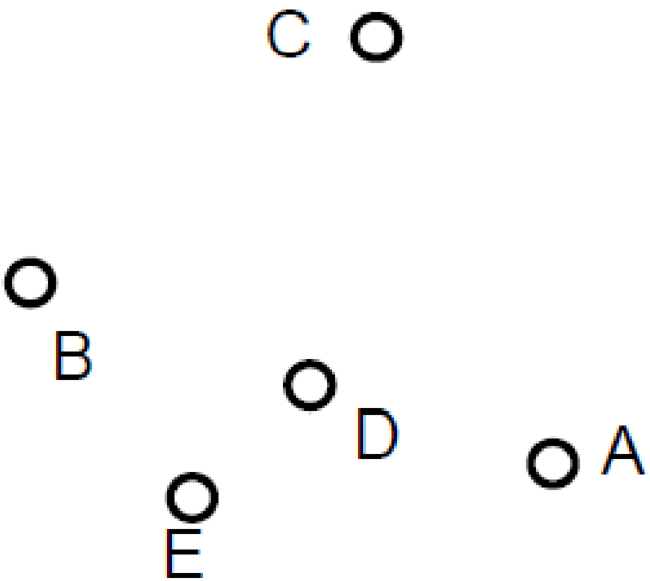
- 从一个叶顶点（当前顶点）开始
- 重复
- 如果存在与当前顶点相连的未遍历MST边，则沿着该边到达一个新的当前顶点
- 否则沿着之前访问的边，返回到单次遍历过的顶点
- 直到所有的MST边被遍历两次

该过程在重新访问到开始顶点处终止

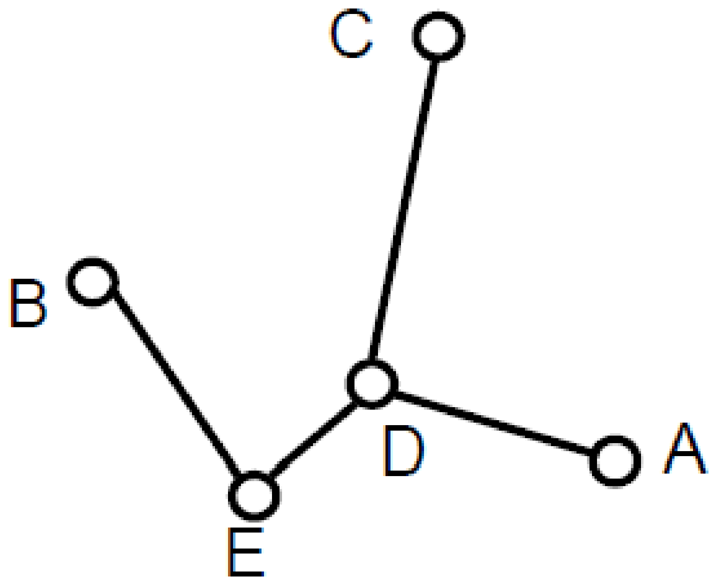
该圈的费用是$2 \cdot c(MST)$，则有

$$c(MST) \le c(TSP) \le 2 \cdot c(MST)$$

一个例子

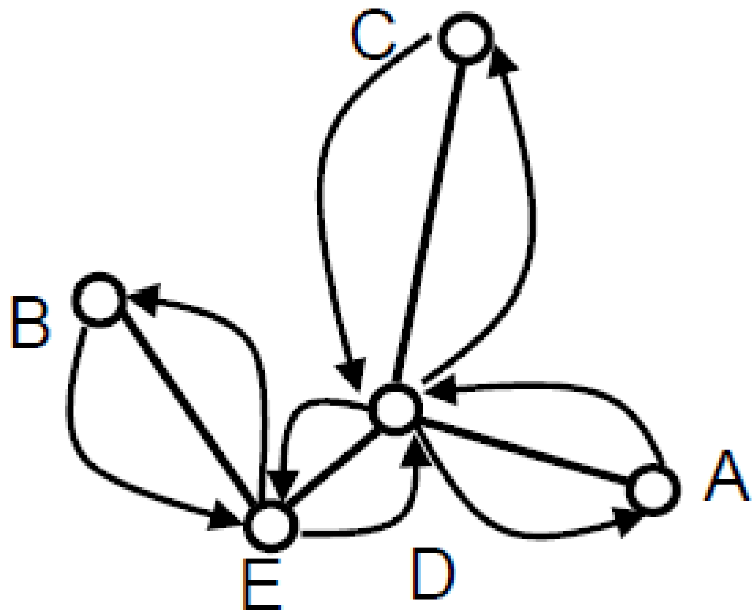


G （所有的边被忽略）



MST

一个例子



双MST（DMST）圈： B-E-D-A-D-C-D-E-B

如果三角不等式成立，则由引入最短割，DMST圈能被转换成一个具有更少费用的HC（$c(HC) \leq 2 \cdot c(MST)$）

最短割遍历（Shortcut traversal）：当深度优先遍历向后访问一个已经访问过的顶点时，跳过该遍历，直接访问下一个未被访问顶点（如果所有顶点已经被访问，则返回开始顶点）

算法（MSTS）

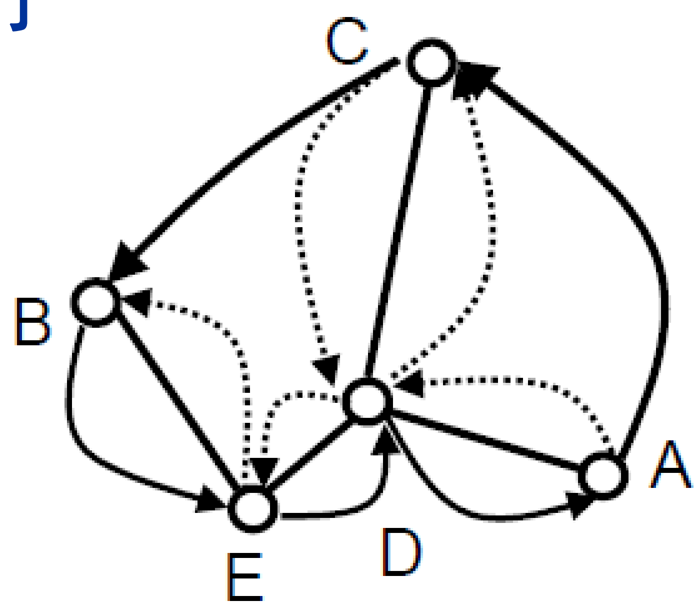
- 找出 G 的一个MST
- 由一个具有最短割的深度优先遍历来构造一个HC

HC费用是下有界和上有界的

$$c(MST) \le c(TSP) \le c(HC) \le 2 \cdot c(MST)$$

$$\mathrm{MSTS}(I) \le 2 \cdot \mathrm{Opt}(I)$$

一个例子



最短割遍历：B-E-D-A-C-B

注意：跟随一个不同的深度优先遍历B-E-D-C-D-A-D-E-B和引入最短割，最终得到的HC更长一些

克里斯托费兹算法（Christofides' algorithm）

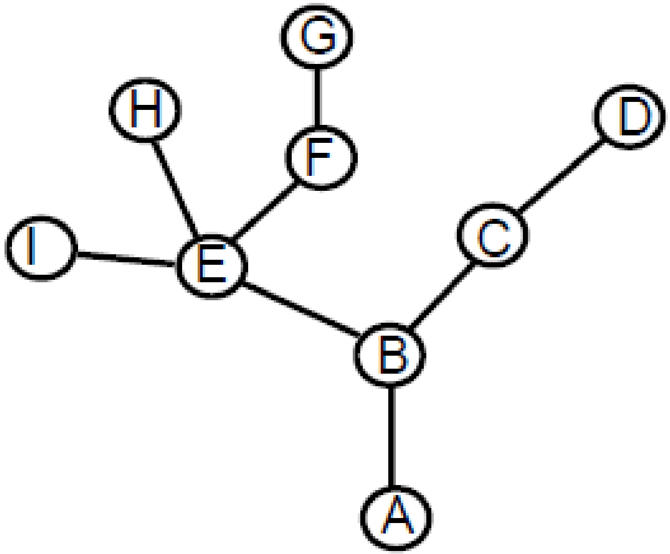
算法（CA）

- 找出 G 的一个MST
- 由引入奇数度顶点对间的最小费用边（最小权重匹配问题），转换一个欧拉图中的MST
- 由一个最短割欧拉遍历来构造一个HC

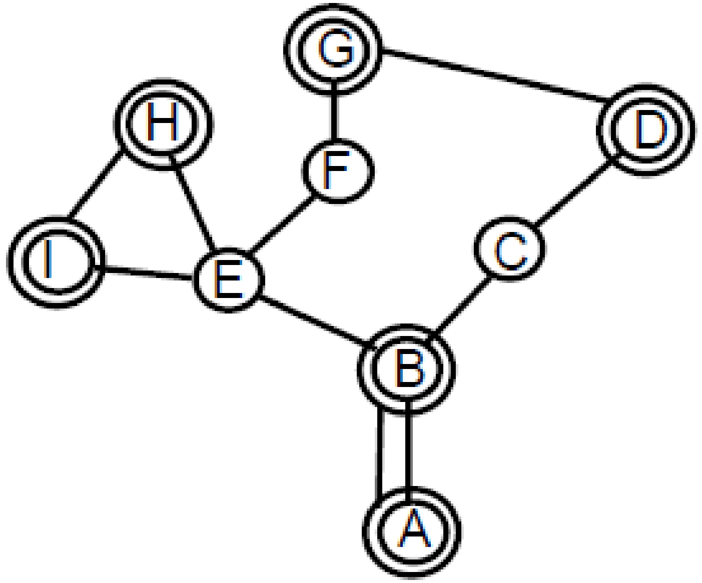
注意：

- 一个图中的奇数度顶点数量总是偶数
- 最小权重匹配问题（The minimum weight matching problem）能在多项式时间（$O(n^3)$）内被求解
- 可以证明$C(I) \le 3/2 \cdot \mathrm{Opt}(I)$

一个例子

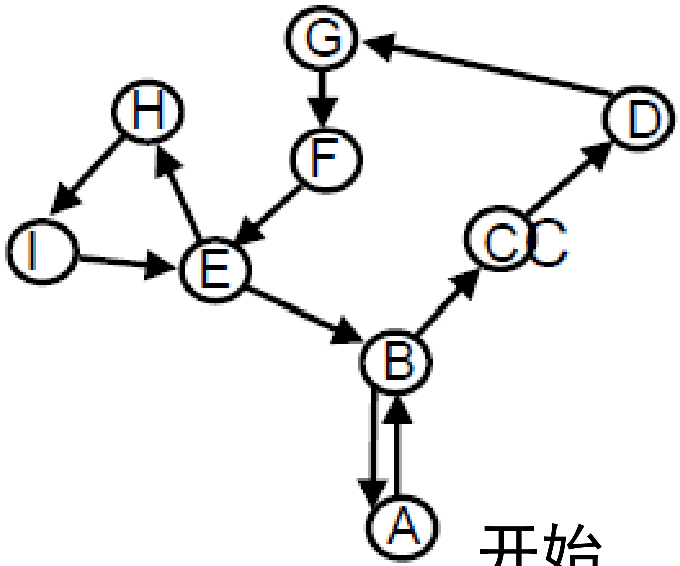


MST

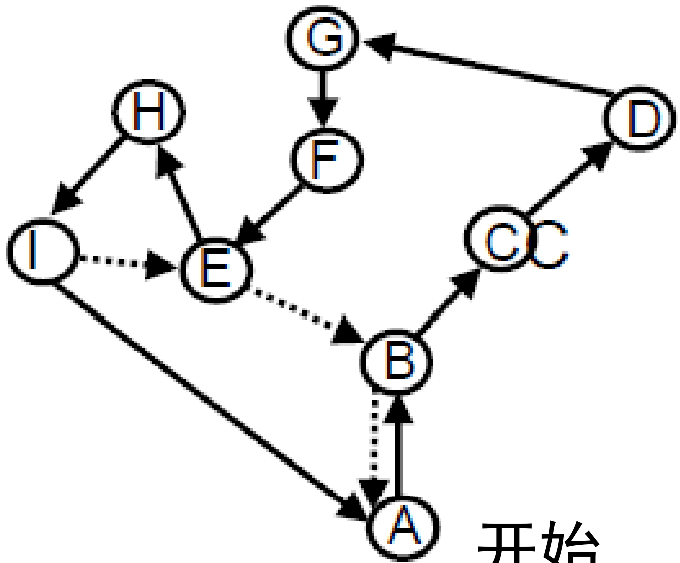


由最小权重匹配
得到的欧拉图

一个例子



图的欧拉遍历



HC（最短割）

注意：由MSTS算法产生的解是不同的ABCDEFGHI，是更长的

2. 求解TSP的改进启发式方法

求解TSP的改进启发式方法：

- 基于2-OPT（两元素优化）的局部搜索
- 基于3-OPT（三元素优化）的局部搜索
- Lin-Kernighan（LK）算法

基本局部搜索（LS）算法

初始化：

- 产生一个初始解 x
- 设置当前解和优化目标（$x_c = x$ 和 $Z_c = Z(x)$）

重复

- 设置 $x_b = x_c$ 和 $Z_b = Z_c$
- 对于每个候选解 $x \in N(x_b)$，如果 $Z(x) < Z_c$，则 $x_c = x$ 和 $Z_c = Z(x)$

直到 $Z_c < Z_b$

$N(x)$ 是解 x 的邻域

- 一个 x 的邻域—改进 x 所产生的解构成

如果 $x_a \in N(x_b)$，则 $x_b \in N(x_a)$

2-OPT（3-OPT）— $N(x)$ 是由2-exchange（3-exchange）移动产生

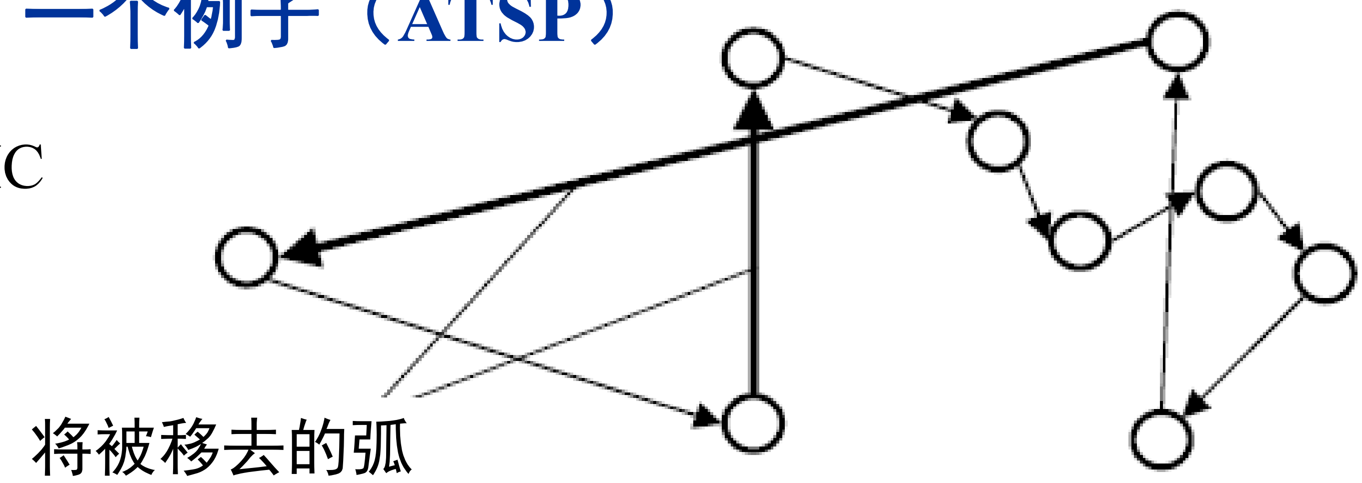
- 当前解中的2（3）条边被移去，由2（3）条其他边替代，产生一个不同的HC

优点：好的执行效果（算法计算结果与最优解比较，具有较小偏差）

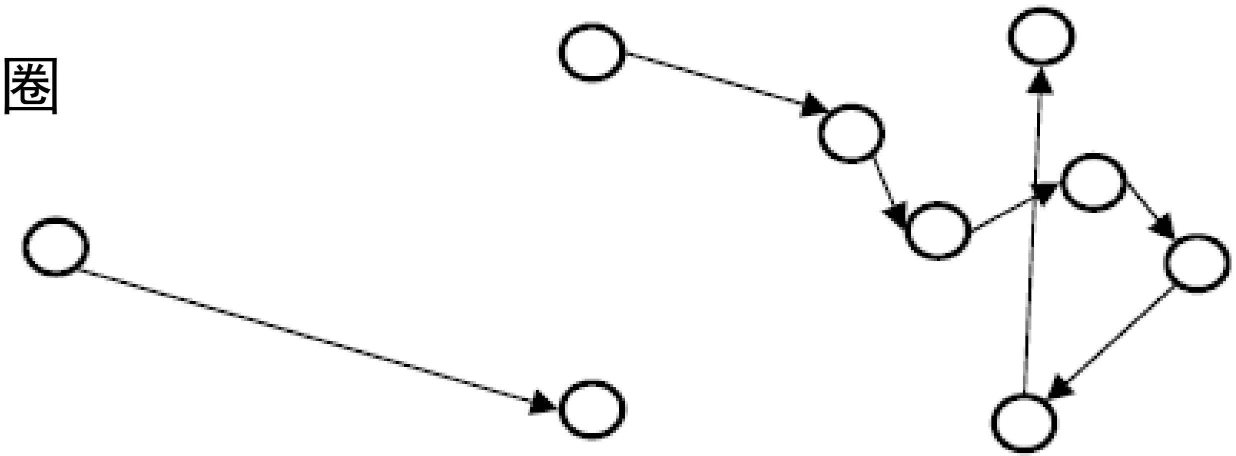
如果所有可能的移动都被检验，计算时间量太大

2-OPT：一个例子（ATSP）

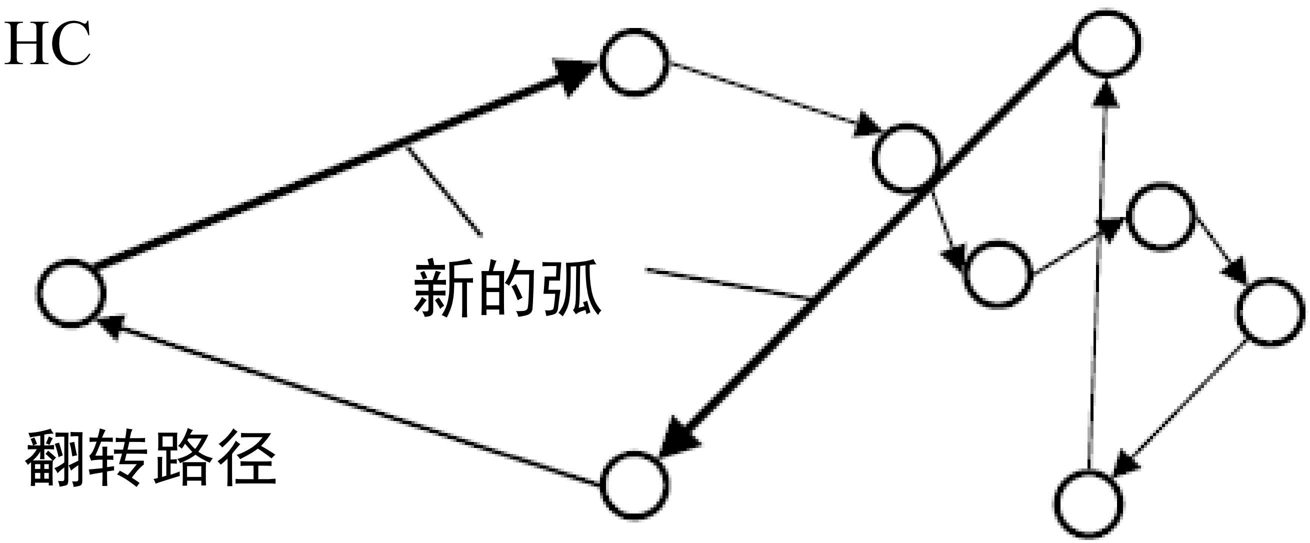
初始HC



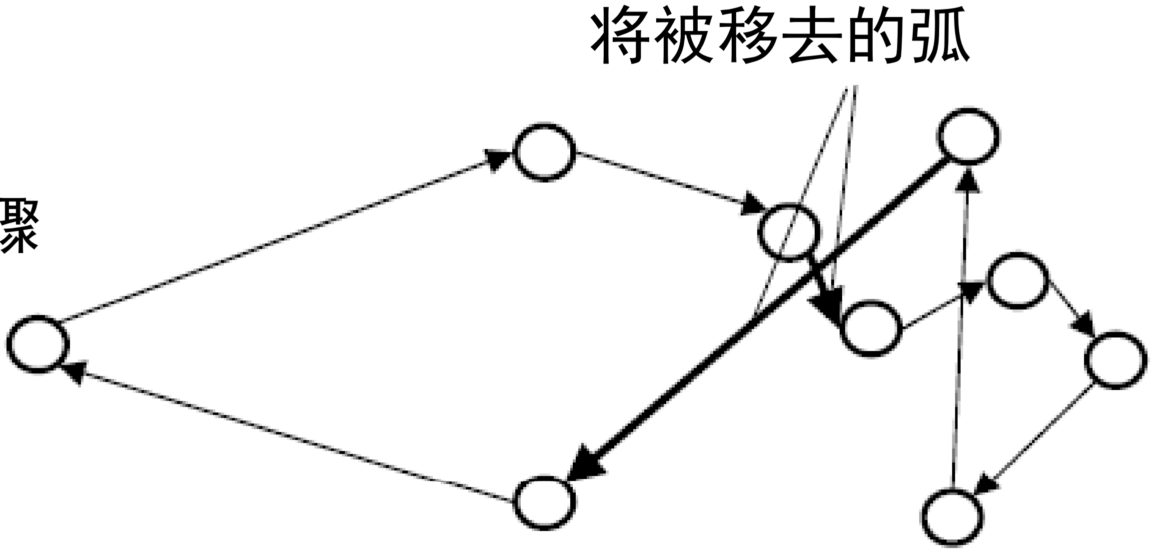
部分闭圈

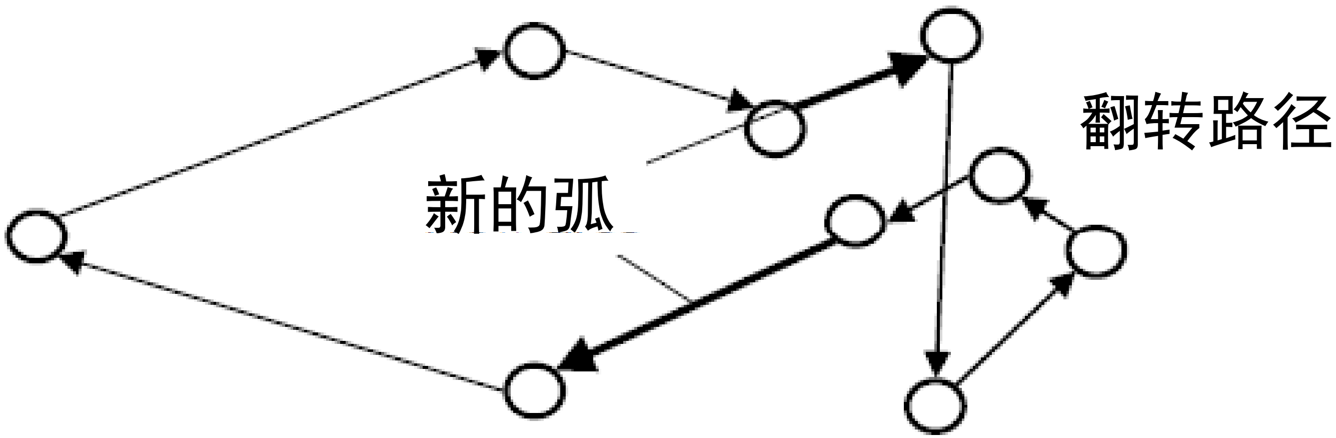


第1步的HC

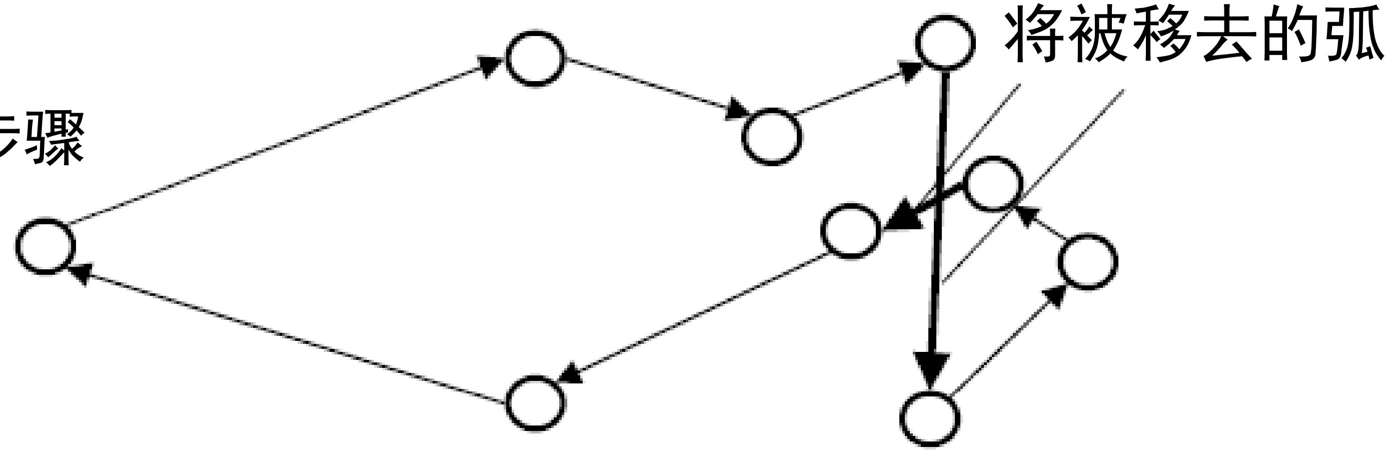


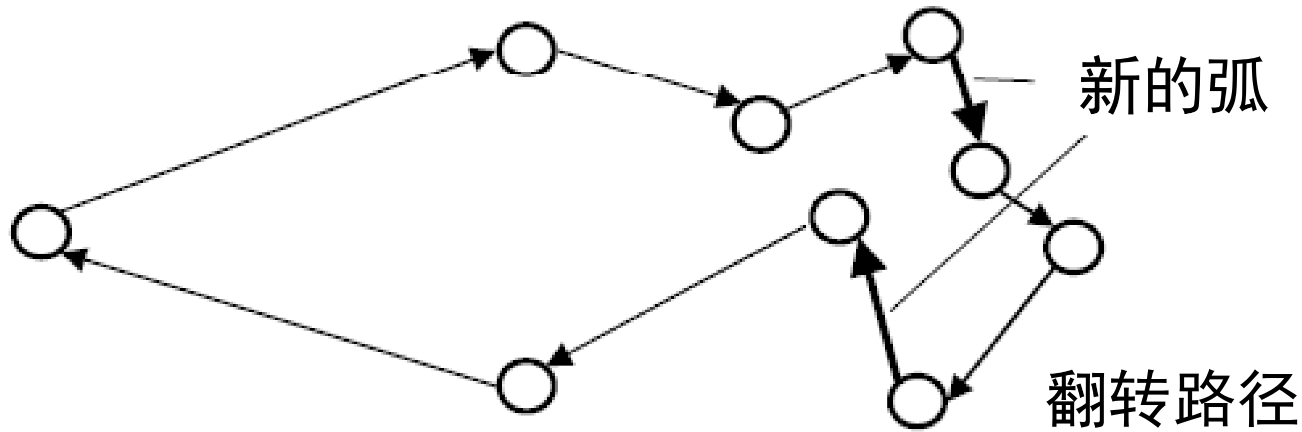
接下来的步骤





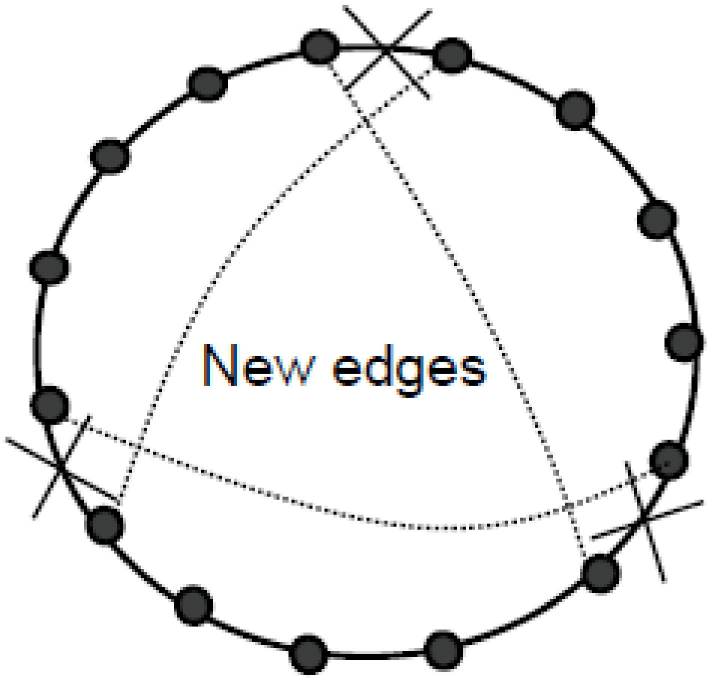
接下来的步骤

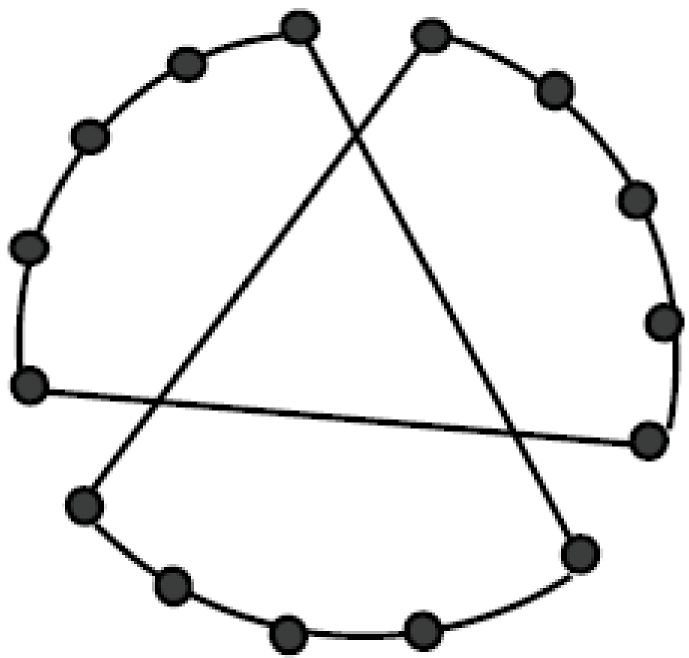




3-OPT：一个例子（ATSP）

将被移去的边





Lin-Kerningham算法（可变k-OPT）

k-exchange移动提供了k-最优解：一个旅行被称为k-最优（k-OPT），如果满足：通过替代任意k条路段均不能得到一个更短的旅行

任意一个k-OPT旅行也是k'-OPT，这里$k' \le k$

一个旅行是最优的，如果它是n-OPT

k值越大，对应解越可能是最优的

增加k值会导致算法计算时间快速增加（计算量是$O(n^k)$，k一般取2或3）

缺点：k值必须提前设定

Lin-Kernighan算法

- 最好的有效启发式方法之一
- 可变k-OPT
- 平均运行时间$O(n^{2.2})$（原始实施）
- 不容易实施⇒广泛的研究
- 在算法实施过程中k的值是变化的，在每次迭代需要先确定一个k值

基本的LK启发式（LKH）

- 在每次迭代，如果k条边的互换可以得到一个更短的旅行，则检验k的递增值（给定r条边的交换被考虑，一系列测试需要被执行，为了确定是否$r+1$条边的交换应该被考虑）
- 迭代继续，直到某停止条件被满足

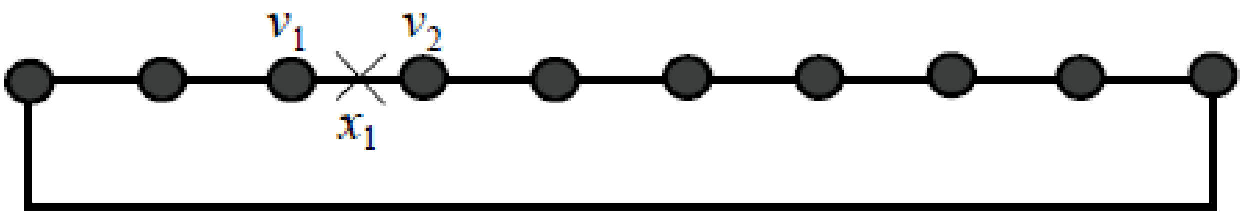
在每一步，LKH考虑一个潜在交换增长集（从 $r=2$ 开始）交换被选择，使得在该过程的每一次迭代，一个可行的旅行被形成

如果在一个迭代中找到一个新的更短的旅行，则当前的旅行被替代

设 X 是将被移去的 r 条边的集合，Y 是将被添加的 r 条边的集合（初始为空）；(v_i, v_{i+1}) 是一条边，x 是 X 中一条边，y 是 Y 中的一条边；$G = c(x_i) - c(y_i)$ 是由 y_i 替代 x_i 的损益

一个LKH迭代的例子

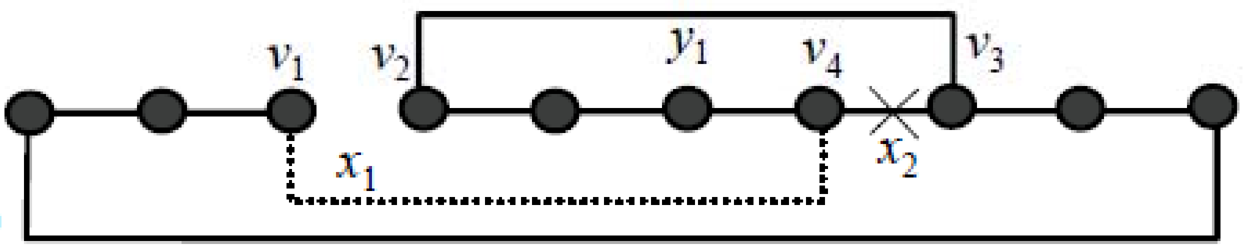
初始旅行T



$X=\{x_1, x_2\}$

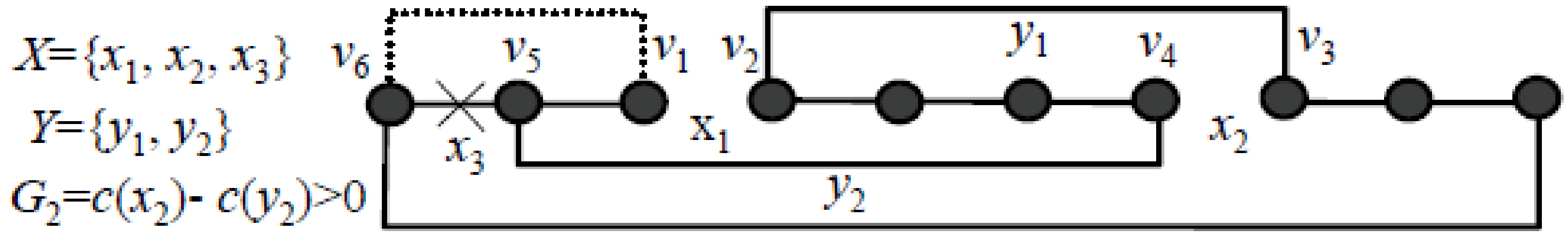
$Y=\{y_1\}$

$G_1=c(x_1)-c(y_1)>0$

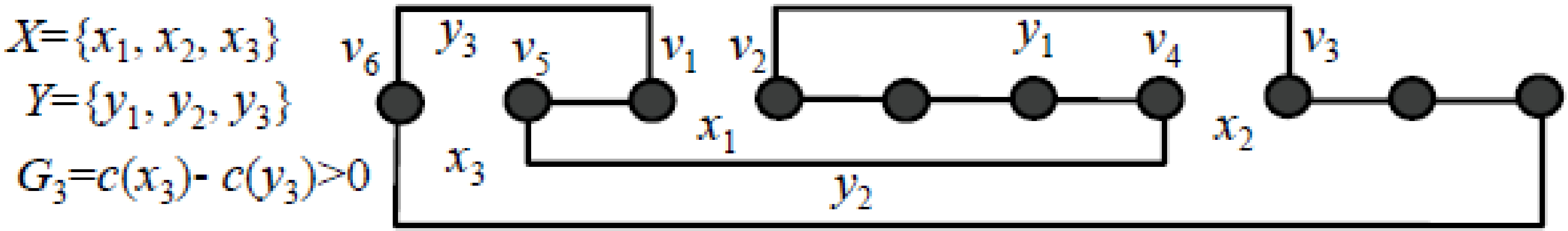


一个旅行必须以边(v_4,v_1)来闭合

这样的边不被选择因为$G<0$



一个旅行能以边(v_6,v_1)来闭合



该步执行一个k=3 exchange