计算机科学2002Vol.29№.7

水流启发式算法求解 Euclid TSP*)

A Water-Flow Heuristic Algorithm to Euclid TSP

郝志峰[1,2] 刘 海[1] 林智勇[1]

(华南理工大学应用数学系 广州510641)[1]

(中国科学院软件研究所计算机科学开放实验室 北京100080)[2]

Abstract This paper is enlightened by the surface tension in the process of water-flow, Firstly, it constructs a protruding polygon which passes through some points, and adds other points to the loop according to multifarious heuristic information one by one. The results show the algorithm can approach optimal answer well in a minute. Additionally, we obtain two optimal loop theorem of TSP under certain conditions, and prove them to provide the theoretical base of the algorithm.

Keywords TSP, Heuristic algorithm

1. 思想来源

旅行商问题(TSP)可以简单表述如下:给定一组N个城市和它们之间的两两距离,找出一个闭合的旅程,使得每个城市刚好经过一次且总的旅程距离最短。旅行商问题已经被证实是一个NP难解问题。虽然欧氏平面上的TSP有PTAS[4],但运算时间和精度呈指数函数关系,所以找一个快速的近似算法仍然具有很大的意义。近年来提出的逼近最优解的算法如遗传算法、模拟退火算法、神经网络算法本质上都是进行随机搜索,本文是用确定性的启发式算法求解TSP。

试考虑在一个光滑的平面上有N个小颗粒(城市),四周围有水慢慢地包围过来,首先水以一个类似凸多边形包住了这N个小颗粒,水的边界经过了其中的若干颗粒,而其他的颗粒还在该边界的内部;水继续往里收缩,这时两颗粒间距离较长的边比距离短的边更容易被水淹没……。水的边界经过的颗粒越来越多,当它经过所有的N个颗粒时,由于水的表面张力的作用,最终水的边界是否就是最优的TSP路径呢?

基于以上的想法,对Euclid平面上给出坐标的N个城市的TSP,先用一个最小的凸多边形包住所有的点,它必定经过其中的若干点(一般至少三点),而其余的点都在该凸多边形的内部。接下去的工作就是如何把剩余的点一一加入到这个多边型中,注意从加入的第一个点以后,该多边形很可能就不再是凸多边形了。加入点的策略有很多:按顺序把点加入到最小代价的边当中、要求每次加入的点使新的多边形包围了所有点、每次加入到某一条边的点和该边构成的角度是最大的、或者每次加入到某一条边的点是所有的边和点当中代价最小的……。本文对这几种策略都进行实例计算和比较。

2. 命题证明

以下四个命题的证明是为了得到在特殊条件下Euclid TSP的最优路径的定理,同时是建立算法的一些理论依据。

引理2.1 在Euclid平面上,由下图给出的四个点组成的城市(点4在点1、2、3组成的三角形的内部),不存在一个凸多边形恰好经过每一个城市。

证明:显然,穷举所给问题的4!条路径,可知不存在满足条件的凸多边形。

引理2.2 在Euclid平面上,只要一个TSP中城市的点集包含了满足引理2.1条件的四个点,则不存在一个凸多边形恰好经过每一个城市。

证明:(反证法)假设这四个点组成的集合是P,如果存在一个凸多边形恰好经过每一个城市的话,任选一个点 $m \in P$,与m相连的点假设是l、n,新增加一条边l-n,再把m点删去,由凸多边形的性质,所得到的回路仍然是个凸多边形,不断重复这一步骤,直到剩下的点都属于P为止,按理它也应该是个凸多边形,但这与引理2.1矛盾,所以假设不真,即引理2.2成立。

定理2.3 对Euclid平面上给出坐标的N个城市的TSP,如果存在一个凸多边形,它的每一个顶点都恰好分别对应一个城市,即该凸多边形经过所有的N个城市,则它就是该问题的最优路径,而且这样的凸多边形是唯一的。

证明:假设存在一个这样的凸多边形Q经过所有的N个城市,它对应的周长是s1,并且该TSP的最优路径R的总长度是s2。

反证:如果s1>s2,即路径R和路径Q不同。则Q中至少有一条边在R中没出现,假设该边是A-B,进一步假设在R中的边是A-B1。

因为Q是凸多边形,所以所有的点都在A-B的同一面。又因为A-B1出现在R中,则A-B1把N个城市分成两个部分,把其中包含B的那一部分记为R1,另一部分记为R2。

在回路R中,因为B和B1连通、B和A也连通,所以必有两条路径:B-m1-m2-…-mk-B1和B-n1-n2-…-nj-A,而且有

集合$\{mk\} \cap R2 \neq \Phi$或集合$\{nj\} \cap R2 \neq \Phi$

(若不然,则在R中,点A、B1和点集R1就已经包含了一个环,矛盾。)

换句话说,在R中必定存在一条边C-D,其中C∈R1且D∈R2,而且C-D和A-B1相交。

*)国家自然科学基金(19901009)、广东省自然科学基金(970472、000463)、中国科学院软件研究所计算机科学开放实验室(SYSKF0105)资助项目。

又因为如前所述，点 C 和点 D 分别在边 A-B1的两边，如果 C-D 和 A-B1不相交，则或者点 A 就在三角形 C-D-B1的内部、或者点 B1在三角形 A-C-D 的内部。这就满足了引理2.1的条件，由引理2.2可知不存在一个凸多边形恰好经过每一个城市。这与假设矛盾，所以 C-D 和 A-B1相交。

这样在 R 中就至少有一对相交的边，由2-opt 算法只要把这对交叉的边改为不交叉的边就可以使总距离变短。如图2，则该 TSP 的最优路径是凸多边形 Q，而且由证明过程知道它是唯一的。

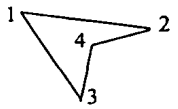

图 1

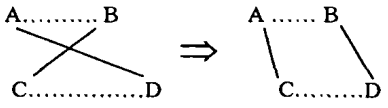

图 2

引理2.4 已知一条由几个点组成的回路 A_1-A_2-…-A_n-A_1和不在回路上的一点 k，如果有条边 A_i-A_j 使得 $d(k,A_i)+d(k,A_j)-d(A_i,A_j)$最短（$j=(i+1)\%n, i=1,2,\cdots,n$），则新回路 A_1-A_2-…-A_{i-k}-A_j-…-A_n-A_1是所有回路 A_1-A_2-…-A_{l-k}-A_m-…-A_n-A_1中最短的。

证明：不妨假设 $i>1$，则回路 A_1-A_2-…-A_{i-k}-A_j-…-A_n-A_1的总距离是：

$$\sum_{p=1}^{i-1} d(A_p,A_{p+1})+d(A_i,k)+d(k,A_j)+\sum_{p=j}^{l-1} d(A_p,A_{p+1})+d(A_l,A_m)+\sum_{p=m}^{n-1} d(A_p,A_{p+1})+d(A_n,A_1)$$

而回路 A_1-A_2-…-A_{l-k}-A_m-…-A_n-A_1的总距离是：

$$\sum_{p=1}^{i-1} d(A_p,A_{p+1})+d(A_i,A_j)+\sum_{p=j}^{l-1} d(A_p,A_{p+1})+d(A_l,k)+d(k,A_m)+\sum_{p=m}^{n-1} d(A_p,A_{p+1})+d(A_n,A_1)$$

上式减下式得：

$$(d(A_i,k)+d(k,A_j)-d(A_i,A_j))-(d(A_l,k)+d(k,A_m)-d(A_l,A_m))$$

由条件可知它小于0。

对 $i<1$ 的情况同理可证，则引理2.4成立。

定理2.5 对 Euclid 平面上给出坐标的 N 个城市的 TSP，如果存在一个凸多边形，它包围了所有的城市，并且其顶点恰好对应其中的 N－1个城市，即剩余的一个城市在该凸多边形的内部；则利用引理2.4计算得到的最小的 TSP 路径，就是该问题的最优解。

证明：（一）假设经过 N－1个点的凸多边形形成的回路是 A_1-A_2-…-A_n-A_1，而在其内部的一个点 k，则该 TSP 的最优路径一定是类似于：A_1-A_2-…-A_{i-k}-A_j-…-A_n-A_1。

若不然，假设该 TSP 的最优路径中和点 k 相连的点是 A_p 和 A_q，而且边 A_p-A_q 不是原凸多边形的边。

这样 A_{p-k}-A_q 就把 N 个城市分成两个部分 R1和 R2，因为 A_p-A_q 不是原凸多边形的边，所以 $R1\neq\Phi$ 而且 $R2\neq\Phi$。

类似于定理2.3的证明，必存在 $A_i\in R1$ 且 $A_j\in R2$ 使得：

边 A_i-A_j 和边 A_{p-k} 相交或者边 A_i-A_j 和边 A_k-A_q 相交。

这样由2-opt 算法只要把这对交叉的边改为不交叉的边就可以使总距离变短，所以该 TSP 的最优路径一定是类似于：A_1-A_2-…-A_{i-k}-A_j-…-A_n-A_1，其中边 A_i-A_j 是原凸多边形的一条边。

再由引理2.4可知，由它计算得到的最小的 TSP 路径，就是该问题的最优解。

3. 算法步骤

对 Euclid 平面上给出坐标的 N 个城市的 TSP 问题，算法步骤如下：

（一）首先用一个最小的凸多边形包住所有的点

1）确定以下特殊的四个点：横坐标最大、最小的点和纵坐标最大、最小的点（它们之间可能有重复）。

2）从横坐标最小的点开始，选择纵坐标大于它并且和它的连线的斜率最大的点，在这两点之间连线。

3）从步骤2选定的点开始，再重复步骤2，直到选定的点是纵坐标最大的点为止。

4）从纵坐标最大的点开始，选择横坐标大于它并且和它的连线的斜率的绝对值最小的点，在这两点之间连线。

5）从步骤4选定的点开始，再重复步骤4，直到选定的点是横坐标最大的点为止。

6）从横坐标最大的点开始，选择纵坐标小于它并且和它的连线的斜率最大的点，在这两点之间连线。

7）从步骤6选定的点开始，再重复步骤6，直到选定的点是纵坐标最小的点为止。

8）从纵坐标最小的点开始，选择横坐标小于它并且和它的连线的斜率的绝对值最小的点，在这两点之间连线。

9）从步骤8选定的点开始，再重复步骤8，直到选定的点是横坐标最小的点为止。

（二）确定选择策略，把剩余的点依次加入到 TSP 回路中

策略一 对剩余的点按顺序把它们加入到命题2.4代价最小的边当中去。

1）按顺序（比如从城市1到城市 N）选定一个目前不在回路上的点，由命题2.4的计算方法找出对应的代价最小的边，完成插入工作。

2）重复步骤1直到所有的点都插入完毕。

策略二 在策略一的基础上并满足：该点插入后使得所有剩余的城市都在新生成的多边形的内部；即要把点 A 插入到边 B1-B2时，不允许有其他剩余的点在三角形 A-B1-B2的内部。

1）用策略一找出当前要插入点的最小代价边。

2）判断是否有其他未插入的点在当前点和边所组成的三角形的内部（可以通过判断该点是否和三角形的一个顶点同在这个顶点对边的一面来确定）。

3）如果步骤2是真，则找下一个未插入的点，返回步骤1。

4）如果步骤2是假，则把当前点插入到最小代价边中。

5）如果没有剩余的未插入点，插入完毕；否则找下一个未插入的点，返回步骤1。

策略三 找出与每一个剩余点构成角度最大的边，最终从这些点和边中挑出构成角度最大的点和边完成插入。

1）按顺序（比如从城市1到城市 N）选定一个目前不在回路上的点，计算出该点插入到当前每一条边时所构成的角度，并记下该点最大角度所对应的边。

2）继续找下一个未插入的点，重复步骤1，如果得到的新

（下转第145页）

参 考 文 献

1 Booch G, Rumbaugh J, Jacobsson I. Unified Modeling Language. Notation Guide version 1.0 Rational Software Corporation, 1997

2 González M, Klein M, Lehoczky J. Fixed priority scheduling of periodic tasks with varying execution priorities. Real Time Systems Symposium, 1991

3 Z. 100, ITU (1994). ITU recommendation Z. 100. Specification and Description Language (SDL)

4 Burns A, Wellings A. HRT-HOOD: A Structured Design Method for Hard Time Ada Systems. Real-Time Safety Critical Systems. Elsevier. 1995, 3

5 Alvarez J M, Diaz M, et al. Integrating Schedulability Analysis and SDL in an Object-Oriented Methodology for Embedded Real-Time Systems. SDL Forum 1999

6 Gresser K. An even model for deadline verification of hard real-time systems. In: Proc. Fifth Euromicro Workshop on Real Time Systems. Oulu, Finland, 1993. 118～123

7 Anders Ek. The SOMT Method. Telelogic. Sep. 1995

(上接第141页)

角度比上一个点对应的最大角度要大，记住该新角度和它对应的点和边。

3)重复步骤2直到所有的未插入点都计算完毕，把最终得到的点插入到其对应的边中。

表1

计算结果和时间	10个城市	20个城市	CHN144
策略一	2.691(<1s)	24.522(<1s)	32738.100(<1s)
策略二	2.691(<1s)	24.753(<1s)	34933.981(<1s)
策略三	2.691(<1s)	24.522(<1s)	34156.440(3s)
策略四	2.691(<1s)	24.522(<1s)	34519.159(7s)
目前最优结果	2.691(<1s) 霍氏神经网络算法	24.38(10s) 模拟退火算法	30566(644.55s) 模拟退火算法

4)如果没有剩余的未插入点，插入完毕；否则找下一个未插入的点，返回步骤1。

策略四 找出每一个剩余点对应的代价最小的边，最终从这些点和边中挑出代价最小的点和边完成插入。

1)按顺序(比如从城市1到城市N)选定一个目前不在回路上的点，由命题2.4的计算方法找出对应的代价最小的边，并记下插入代价。

2)继续找下一个未插入的点，重复步骤1，如果该代价比上一个点的代价小，记住该点和其对应的边。

3)重复步骤2直到所有的未插入点都计算完毕，把最终得到的点插入到其对应的边中。

4)如果没有剩余的未插入点，插入完毕；否则找下一个未插入的点，返回步骤1。

4. 实例计算和分析

本文对规模N=10、20以及CHN144的TSP都进行了计算，结果如表1。

从计算结果上看，在这三个不同规模大小TSP中，策略一的表现最好，策略三次之，其次是策略四和策略二；当N=10时他们都找出了最优解，当N取其它值也较好地逼近了最优解。

从运行时间上看，策略一和策略二大致相同，而策略三和策略四对N较大时花费的时间大于策略一和策略二；但它们和目前的最优算法相比，在N较大时都大大地减少了运算的时间。

上述算法的搜索空间复杂度分析：策略一的空间复杂度是$O[N^2/2]$；策略二和策略三的空间复杂度都是界于$O[N^2/2]$和$O[N^3/2]$之间；策略四的空间复杂度是$O[N^3/2]$。

参 考 文 献

1 康立山，等. 非数值并行计算. 科学出版社，2000

2 Hou N A E，李军，等译. 用于最优化的计算智能. 清华大学出版社，1999

3 Z. 米凯利维茨. 演化程序. 科学出版社，2000

4 Arora S. Polynomial-time approximation schemes for Euclidean TSP and other geometric problems. In: Proceeding of 37th IEEES symposium on Foundations of Computer Science, 1996. 2～12