一种改进的求解 TSP 问题的近似算法

刘艳娟，谢晓钢，陈胜达
（厦门大学 计算机科学系，福建 厦门 361005）
E-mail：jjj514@sohu.com

摘 要：旅行商问题（TSP）是典型的具有 NPC 复杂性的组合优化问题。在现有求解 TSP 问题的 2-近似算法 closest-point 算法基础上，通过对插入点的插入位置进行改进，提出了一种有效的近似算法最近点前后插入法（CPBOA），并采用 TSPLIB 中的一些典型实例对该算法进行了测试，同时与典型的常数近似比算法 MST-PRIM 算法和 closest-point 算法进行了比较。实验结果表明，该算法在求解质量上与 closest-point 和 MST-PRIM 算法相比都有很大的改进，而且速度也很快。

关键词：旅行商问题；NPC；closest-point；最近点前后插入法；近似算法

文章编号：1002-8331（2006）33-0071-03 **文献标识码**：A **中图分类号**：TP301.6

Improved Approximation Algorithm for TSP

LIU Yan-juan，XIE Xiao-gang，CHEN Sheng-da
（Department of Computer Science，Xiamen University，Xiamen，Fujian 361005，China）

Abstract：Traveling Salesman Problem（TSP） is a typical combinatorial optimization problem.It has been proved that TSP is of NPC complexity.This paper improves the position of inserted point based on 2-approximation algorithm closest-point for solving TSP，and proposes an efficient approximation algorithm CPBOA.Some typical instances in TSPLIB are used to test this algorithm，which are compared with MST-PRIM and closest-point.The computational results show that this algorithm can get better solution，and the time spending is less than closest-point.

Key words：TSP；NPC；closest-point；CPBOA；approximation algorithm

1 引言

TSP 问题是为人们所广泛研究的典型 NP-Hard 组合优化问题，广泛应用于 VLSI 芯片设计、电路板布局、机器人控制、车辆选路等领域。对该问题的综述参见文献[1-3]，其形式化描述为：给定加权图 $G=(V,E,w)$，V 为顶点集，$|V|=n$，E 为边集，$w:E\rightarrow R^{+}$为边权函数，G 中一个 TSP 环路就是一条不重复的访问 V 中所有顶点的哈密顿回路，记为

$$T=<v_1,v_2,\cdots,v_n>$$

其中 $\forall 1\leq i\neq j\leq n, v_i\neq v_j$ 且对 $1\leq i\leq n, \langle v_i, v_{(i \bmod n)+1}\rangle\in E$，记 TSP（G）为 G 中所有 TSP 环路的集合。定义 $w(T)=w(\langle v_1,v_n\rangle)+\sum_{1\leq i\leq n-1} w(\langle v_i,v_{i+1}\rangle)$，要求权最小的 TSP 环路 T^{*}，使得 $w(T^{*})=\min_{T\in TSP(G)} w(T)$。

TSP 问题是典型的 NP-Complete 问题，即 P≠NP 在的假设下，找不到一个算法能够保证在多项式时间内得到最优解。目前对于该问题的研究有两个方向：完全算法能够保证得到最优解，但是运行时间是呈指数复杂度的，因而难以适应大规模的实例；近似算法则只要求找到近似解，而在多项式时间内结束。在 TSP 问题上的近似算法分为两类：环路构造算法和环路改进算法。前者从某个非法解开始，通过某种增广策略逐步改变该解，直到得到一个合法解为止，这类算法虽然具有快速性，但求解的质量一般较差。这类算法包括最近邻算法[4]、closest-point 算法[5]、shortest-link 算法、MST-PRIM 算法[5]等。环路改进算法则在给定初始的合法解之后，使用某种策略来改进初始解。这些策略包括局部搜索、模拟退火、遗传算法和演化算法等，它们的求解质量比较好，但是往往需要的计算量很大，而且参数很难确定；其中最为简单和有效的方法为局部搜索，如 2-OPT[6]、3-OPT[7]、LK[8]、循环 LK[9]等。文献[11]中提出了一种改进的演化算法，虽然速度较快，但也只适用于 500 以下城市的规模。通常这两类算法结合起来使用，用环路构造法来构造初始解，而用环路改进算法改进这个初始解。Arora[10]对欧式空间的 TSP 问题提出了多项式时间的近似策略，这也是近年来在近似算法的研究中所取得的一项重要进展。

对于 TSP 问题而言，在求得 n-1 个城市的基础上进一步求解 n 个城市的解，将大大减少搜索量。这种求解方法即属于构造性方法，如 closest-point 算法[5]。本文在对 TSP 问题进行分析的基础上，对现有的 2-近似算法 closest-point 算法进行了改进，从而提出了新的近似算法最近点前后插入法。该算法在找下一个点时，方法与 closest-point 是一致的，不同的是对 closest-point 算法中离环最近的点的插入位置进行了改进，不是简单的直接插入到环中离该点最近的点的后面，而是通过判断插入到前面还是后面会使更新后的环的总权值更小（即增量更小），最后确定插入位置。这里用该算法测试了 TSPLIB 中的一些典型实例[12]，并与 MST-PRIM 算法和 closest-point 算法作了比较，实验表明，该算法在求解质量上较其有很大的改进，而且速度也很快。

作者简介：刘艳娟（1981-），女，硕士研究生，主要研究方向为计算智能；谢晓钢（1982-），男，硕士研究生，主要研究方向为分布式操作系统；陈胜达（1983-），男，硕士研究生，主要研究方向为计算智能。

2 closest-point 算法

closest-point 算法是一种求解 TSP 问题的并且已经被证明的 2-近似算法。该算法是在最小生成树算法 Prim 算法基础上改造而成：选取图中的任一点作为初始环，在以后的每一步中，找出离环最近的点 u，设环中距离 u 最近的点为 v，将 u 插到 v 后面以得到一个新的环；重复以上步骤直到所有点都在环中。该算法的时间复杂度为 O(ElogV)。Closest-point 算法具体描述如下：

```
for each u∈V[G]
    do key[u]←∞
       π[u]←NIL
key[r]←0
Q←V[G]
C←{r}// C 记录环中的顶点
while Q≠Φ
    do u←min(Q)    //min(Q)为找出离环最近的顶点
    v←π[u]
将 u 插入到环中 v 的后面，更新 C
for each w∈Adj[u]
    do if w∈Q and weight(u, w)<key[w]
       then π[w]←u
            key[w]←weight(u, w)
```

3 最近点前后插入法

该算法在 closest-point 算法基础上，对处理离环最近的点的插入问题的办法作了改进，并分情况讨论了插入前面还是后面能使总增量更小以确定插入位置。当找到离环最近的点 u 后，该算法主要分四种情况进行讨论，具体如下：

(1)如图 1，当环中只有一个点 v 时，u 直接插到 v 后面(v=π[u])。

(2)如图 2，设环 C={v, b, c}，当 u 的前驱(即环中离 u 最近的点)v=π[u]为环 C 中第一个点时，判断是否有 weight(v, b)+weight(u, c)<weight(u, b)+weight(v, c)成立(即判断环{u, v, b, c}和{v, u, b, c}哪个环总权值更小)，如果成立，则将 u 插在 v 前面，更新 C 使 C={u, v, b, c}；否则，将 u 插在 v 后面，更新 C 使 C={v, u, b, c}。

(3)如图 3，设环 C={a, b, c}，当 u 的前驱(即环中离 u 最近的点)v=π[u]为环 C 中最后一个点时，判断是否有 weight(b, u)+weight(a, v)<weight(b, v)+weight(a, u)成立(即判断环{a, b, u, v}和{a, b, v, u}哪个环总权值更小)，如果成立，则将 u 插在 v 前面，更新 C 使 C={a, b, u, v}；否则，将 u 插在 v 后面，更新 C 使 C={a, b, v, u}。

(4)如图 4，设环 C={a, v, c}，当 u 的前驱(即环中离 u 最近的点)v=π[u]为环 C 中的中间点时，判断是否有 weight(a, u)+weight(v, c)<weight(a, v)+weight(u, c)成立(即判断环{a, u, v, c}和{a, v, u, c}哪个环总权值更小)，如果成立，则将 u 插在 v 前面，更新 C 使 C={a, u, v, c}；否则，将 u 插在 v 后面，更新 C 使 C={a, v, u, c}。

CPBOA 整个算法具体描述如下：

```
for each u∈V[G]
    do key[u]←∞
    π[u]←NIL
key[r]←0
Q←V[G]
C←{r}        // C 记录环中的顶点
while Q≠Φ
    do u←min(Q) //min(Q)为找出离环最近的顶点
    v←π[u]
    if C 中只有一个点 v, then 直接将 u 插入 v 后面构成新的环C
    else 分别计算 u 插在 v 前面和 v 后面而构成的新环的总权值，如果前者小则插在前面，否则插在后面，更新 C
    for each w∈Adj[u]
        do if w∈Q and weight(u, w)<key[w]
           then π[w]←u
                key[w]←weight(u, w)
```

由上述算法的描述可以看出，该算法的时间复杂度为 O(ElogV)，与 closest-point 算法的时间复杂度相同。

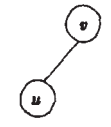

图 1 情况(1)

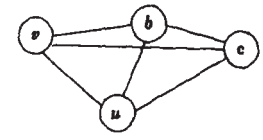

图 2 情况(2)

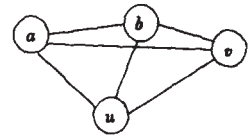

图 3 情况(3)

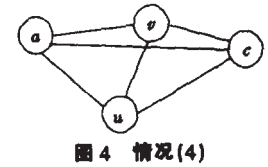

图 4 情况(4)

4 实验结果

本文实现了用 CPBOA 算法求解 TSP 问题，实验环境是 Dell GX270, P4 2.6 GHz 微处理机，512 MB 内存，操作系统为 Windows 2000，并对 TSPLIB 中的 12 个典型实例[12]进行了测试(各运行 15 次)。作为比较，本文在相同的实验环境下运行了 2-近似算法 MST-PRIM 算法和 closest-point 算法，并在相同的实例上测试了该算法，将三种算法的运行结果(包括平均解和平均运行时间)进行了比较，并分别计算了与最优解之间的误差，其结果见表 1、2。

表 1 测试结果对比

实例	最优解	平均计算结果/km			平均运行时间/s		
		MST-Prim	Closest-point	CPBOA	MST-Prim	Closest-point	CPBOA
kroA100	21 282	29 694	28 868	28 362	0.002	0.004	0.005
kroB100	22 141	28 949	29 418	27 760	0.002	0.005	0.005
pr136	96 772	143 480	143 296	129 636	0.000	0.012	0.008
pr144	58 537	82 078	83 645	81 764	0.003	0.010	0.008
kroA150	26 524	38 604	38 742	33 319	0.002	0.010	0.009
kroB150	26 130	35 793	37 257	34 281	0.002	0.010	0.009
ch150	6 528	9 171	9 272	8 356	0.002	0.014	0.009
pr226	80 369	115 635	118 203	110 450	0.003	0.031	0.030
pr439	107 217	144 087	147 966	137 748	0.009	0.091	0.085
rat575	6 773	9 511	9 939	8 612	0.016	0.169	0.161
u724	41 910	59 879	60 056	54 520	0.020	0.236	0.231
d1291	50 801	72 767	74 624	71 836	0.056	0.799	0.818

从表 1 可以看出，对于这 12 个实例而言，closest-point 算法得到的实验结果和时间较 MST-PRIM 都差很多，而对其改进的 CPBOA 算法较 closest-point 算法在求解质量上有很大的提高，时间上也有所提高；与 MST-PRIM 算法相比，CPBOA 求解质量上也有很大的改进。从表 2 可以看出：MST-PRIM 算法与最优解之间的误差为 40.55%，closest-point 算法的误差为 42.68%，而 CPBOA 算法的误差为 31.81%，相对前两者分别提高了 8.74%和 10.89%，有明显的改进。

表 2　误差对比

实例	最优解	与最优解之间的误差/%		
		MST-Prim	Closest-point	CPBOA
kroA100	21 282	39.53	35.65	33.27
kroB100	22 141	30.75	32.87	25.38
pr136	96 772	48.27	48.08	33.96
pr144	58 537	40.22	42.89	39.68
kroA150	26 524	45.54	46.06	25.62
kroB150	26 130	36.98	42.58	31.20
ch150	6 528	40.49	42.03	28.00
pr226	80 369	43.88	47.08	37.43
pr439	107 217	34.39	38.01	28.48
rat575	6 773	40.43	46.74	27.15
u724	41 910	42.88	43.30	30.09
d1291	50 801	43.24	46.89	41.41
平均误差	40.55%	42.68%	31.81%	

5　结论

本文在对 TSP 问题进行分析的基础上，针对现有的 2-近似算法 closest-point 算法，通过引入优化机制，判断离环最近的点插入在前面或者后面能使环的增量较小从而确定插入位置，进而提出了 CPBOA 算法。该算法一般不受规模大小的限制，实验表明，平均质量较 MST-PRIM 和 closest-point 都有大幅度改进，而且速度也很快，时间复杂度为 O(ElogV)。虽然离最优解还有一定的误差，但由于求解速度快，所以有较大的应用价值，也可以应用到遗传算法、模拟退火等启发式算法中作为初始解使用。希望能在以后的工作中找出求解 TSP 问题的更有效的近似算法并求出其近似比。(收稿日期：2006 年 2 月)

参考文献：

[1] JOHNSON D S, MCGEOCH L A.The traveling salesman problem: a case study in local optimization[C]//AARTS E H, LENSTRA J K. Local search in Combinatorial Optimization.New York: John Wiley and Sons, 1996.

[2] JUNGER M, REINELT G, RINALDI G.The traveling salesman problem[C]//BALL M, MAGNANTI T, MONMA C L.Handbook on Operations Research and Management Science: Networks.NorthHolland, 1995: 225-330.

[3] BURKARD R E, DEINEKO V G, DAL R V, et al.Well-solvable special cases of the traveling salesman problem: a survey[J].SIAM Review, 1998, 40(3): 496-546.

[4] 陈荣秋.排序的理论与方法[M].武汉：华中理工大学出版社, 1987: 58-59.

[5] CORMEN T H, LEISERSON C E, RIVEST R L.Introduction to algorithms[M].2nd ed.北京：高等教育出版社, 2002: 1027-1033.

[6] CROES G A.A method for solving traveling salesman problems[J]. Operations Research, 1958, 6: 791-812.

[7] LIN S.Computer solutions to the traveling salesman problem[J].Bell System Technical Journal, 1965, 44(10): 2245-2269.

[8] LIN S, KERNIGHAN B W.An effective heuristic algorithm for the traveling salesman problem[J].Operations Research, 1973, 21: 498-516.

[9] JOHNSON D S.Local optimization and the traveling salesman problem[C]//Proceedings of the 17th Colloquium on Automata, Language, and Programming.Lecture Notes in Computer Science 443. Berlin: Springer-Verlag, 1990: 446-461.

[10] ARORA S.Polynomial time approximation schemes for Euclidean TSP and other geometric problems[C]//Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 1996: 2-11.

[11] 蔡之华，彭锦国.一种改进的求解 TSP 问题的演化算法[J].计算机学报, 2005, 28(5): 824-828.

[12] TSPLIB[EB/OL].http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

(上接 42 页)

明显的增强，同时 MPSO 相对 CLPSO 而言，解的收敛能力又有了明显的提升。

从以上两个实验中可以看出，MPSO 算法的性能相对于其他算法有了明显的提高，同时，解的稳定性和收敛性也有了一定的提高。此外，从实验二中可以发现，新算法对于高维优化问题中全局最优解相对搜索空间位置的鲁棒性得到了明显的提高，对比表 1 和表 2 可以发现其余三个算法的性能在实验一和实验二中有着相当大的变化，即使是在一般情况下，性能最好的隐匿墙作为边界条件的 CLPSO 算法，其性能也有了较大的变化，而 MPSO 算法在实验二中依然取得了相对较好的效果。所以，本文提出的算法对于那些全局最优解相对搜索空间位置较偏的优化问题也有比较好的性能。

5　结语

本文提出一种改进的粒子群算法，以 CLPSO 算法的思想为基础，在参数的设置中结合高斯分布的概念，并且引入了选择墙的概念，结合了三种边界条件的特点，提高了算法的鲁棒性。实验结果表明，改进后的粒子群算法防止陷入局部最优的能力有了明显的增强，解收敛的稳定性也有了一定的提高。特别是，相对于其他方法，算法使高维优化问题中全局最优解相对搜索空间位置的鲁棒性得到了明显的提高。

(收稿日期：2006 年 8 月)

参考文献：

[1] KENNEDY J, EBERHART R C.Particle swarm optimization[C]//Proc IEEE International Conference on Neural Networks.USA: IEEE Press, 1995, 4: 1942-1948.

[2] LIANG J J, QIN A K, SUGANTHAN P M, et al.Particle swarm optimization algorithms with novel learning strategies[C]//2004 IEEE International Conference on Systems, Man and Cybernetics, SMC 2004, Oct 10-13 2004, The Hague, Netherlands, c2004: 3659-3664.

[3] SHI Y, EBERHART R C.A modified particle swarm optimizer[C]//1998 IEEE International Conference on Evolutionary Computation Proceedings.IEEE World Congress on Computational Intelligence, 4-9 May 1998, Anchorage, AK, USA, c1998: 69-73.

[4] EBERHART R C, SHI Y.Comparing inertia weights and constriction factors in particle swarm optimization[C]//Proceedings of the IEEE Conference on Evolutionary Computation.USA[s. n.], 2000, 1: 84-88.

[5] KROHLING R A.Gaussian swarm: a novel particle swarm optimization algorithm[C]//2004 IEEE Conference on Cybernetics and Intelligent Systems.Singapore, 2004: 372-376.

[6] ROBINSON J, RAHMAT-SAMII Y.Particle swarm optimization in electromagnetics[J].IEEE Transactions on Antennas and Propagation, 2004, 52(2): 397-407.

[7] EBERHART R C, SHI Y.Empirical study of particle swarm optimization[C]//Proceedings of the 1999 Congress on Evolutionary Computation.USA, 1999, 3: 1945-1950.