计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(7):145−149 [doi: 10.15888/j.cnki.csa.006441]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

最大最小目标的多旅行商问题求解①

袁　志

(广州大学 华软软件学院, 广州 510990)
通讯作者: 袁　志, E-mail: mynameisyz@163.com

摘　要: 研究一类多旅行商问题, 对所有旅行商所走的环路, 寻求最大长度最小化. 设计了一种新的局部搜索算子, 该算子既能进行一条环路的优化, 也能对两条环路进行重组优化, 与现有的局部搜索算子相比, 在计算成本相当的条件下, 其寻优能力更好; 以该算子为基础, 提出了“搜索-选优-变异-搜索”的迭代策略, 按此策略设计了竞争搜索算法. 在公开的数据集上进行了实验, 与近期文献相比, 计算结果有所改进.

关键词: 多旅行商问题; 最小化最大值; 局部搜索; 进化算法; 竞争搜索算法

引用格式: 袁志.最大最小目标的多旅行商问题求解.计算机系统应用,2018,27(7):145–149. http://www.c-s-a.org.cn/1003-3254/6441.html

Solving Multiple Traveling Salesmen Problem with Minimal Maximum

YUAN Zhi

(South China Institute of Software Engineering, Guangzhou University, Guangzhou 510990, China)

Abstract: Research on one kind of multiple traveling salesmen problem requires minimizing the maximum length of the cycles traveled by all the salesmen. A new local search operator is designed, this operator not only optimizes one cycle, but also reorganizes and optimizes two cycles, its optimization ability is better than the existing local search operators under the equivalent calculation cost. Based on this operator, an iterative strategy named “search–select–mutate– search” is proposed, according to this strategy, the Competitive Search Algorithm (CSA) is designed. The experiments on the open data sets showed that CSA performs better than recent literatures.

Key words: multiple travelling salesmen problem; minimizing maximum; local search; evaluation algorithm; Competitive Search Algorithm (CSA)

1　概述

多旅行问题 (Multiple Traveling Salesmen Problem, MTSP) 是 TSP 的扩展. 给定一个中心城市和 n 个访问城市, 将访问城市分配给 m 个旅行商, 每个旅行商从中心城市出发巡游若干访问城市后回到中心城市, 要求所有旅行商经过的总路程 ($total$) 尽量小且其中最长环路的长度 (max) 尽量小. 事实上, 无法保证 $total$ 和 max 两个目标同时达到最小, 本文寻求最小化 max, 称之为 MinMax-MTSP. 这一类问题的算法可应用在工作均衡调度, 印刷机调度、卫星测量系统设计、机器人应急响应[1−3]等领域.

城市集合 $C=\{c_0,c_1,c_2,\cdots,c_n\}$, c_0 表示中心城市, c_1, $c_2,\cdots,c_n$ 表示访问城市. 距离矩阵 $D=\{d_{i,j}\}$, $d_{i,j}$ 为 c_i 到 c_j 的距离. 每条环路用城市号序列来编码, 例如 $cycle=(0,1,2)$ 表示 $c_0\rightarrow c_1\rightarrow c_2\rightarrow c_0$. 图 1 显示了 n=10, m=3 的 MinMax-MTSP 的一个解 (非最优解), 其中的 3 条环路分别是 (0,1,2), (0,5,4,3) 和 (0,6,7,8,9,10), 解表示为 S=(0,1,2,0,5,3,4,0,6,7,8,9,10). MinMax-MTSP 求解目标是寻找 S 的最佳序, 使 max 最小.

将进化算法与局部搜索算法结合来求解 MTSP 是

① 基金项目: 广东省教育厅重大平台和科研项目 (2015KTSCX177)
Foundation item: Major Platform and Scientific Research Project of Bureau of Education of Guangdong Province (2015KTSCX177)
收稿时间: 2017-11-19; 修改时间: 2017-12-15; 采用时间: 2017-12-20; csa 在线出版时间: 2018-06-27

近年来的主要方法. 一类是遗传算法, Carter[4]提出了一种双基因编码 (城市基因和组基因) 的遗传算法, 并提出了 12 个测试例; Brown[5]提出一种分组遗传算法. Yuan[6]在分组遗传算法中引入了一种新的交叉算子 (TCX), 提出了 3 个新的测试例; Singh[7]改进了分组遗传算法 (GGA-SS), 用组的 *rk* 值 (*rk*=路径长度/城市数) 衡量一个组属于最优解的可能性, *rk* 值较大的组优先保留在子代中, 其余一些零散的城市用贪心算法插入到各个组, 使用 *2-opt* 局部搜索算子进行组内优化. 另一类是群体智能算法, Liu[8]用蚁群算法求解 MTSP; VENKATESH[9]提出了两种蜂群算法 (ABCFC、ABCVC) 和一种杂草入侵算法 (IWO), 同样用 2-opt 进行组内优化.

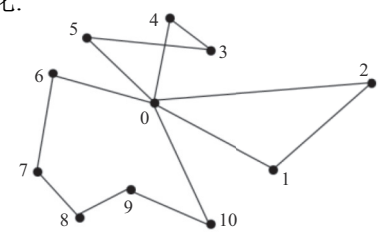


图 1　3 个旅行商 10 个访问城市的一个解

以上算法的一致性在于: 使用局部搜索算法来快速加速寻优过程, 使用群体进化算法不断累计优化结果. 这些算法采用 2-opt 作为局部搜索算子, 而且仅将 2-opt 作用于单条环路的优化. 实际上, 在 MTSP 中, 可以将局部搜索从单条环路的优化扩展为两条环路的重组优化, 加速算法的寻优过程 (第 2 节讨论). 这些算法主要依赖个体之间交换信息来完成迭代优化, 很少考虑到局部搜索算子自身的特点, 在进化算法中根据局部搜索算子自身的特点, 设计新的进化机制, 是本文关注的另一个重要内容 (在第 3 节讨论).

本文设计了一个新的局部搜索算子 reverse/ move (转置/移动), 该算子既能进行一条环路的优化, 也能重组优化两条环路, 即便在一条环路上进行优化, 其能力也明显好于 2-opt; 在分析 reverse/move 算子特点的基础上, 提出了“搜索-选优-变异-搜索”策略, 设计了竞争搜索算法 (Competitive Search Algorithm, CSA), 在文献[4]和文献[6]的 15 个测试例上进行实验, 与文献[6–9]进行比较, CSA 在计算结果上有明显的改进.

2　局部搜索算子 reverse/move

为了指导搜索过程, 需要对解有合适的评价方法. MinMax-MTSP 的目标是最小化 *max*, 但注意到, 将目标改为优先最小化 *max*, 其次最小化 *total*, 有助于最小化 *max*. 因为后者要求每一条环路自身次序最优, 在大多数情况下, 这有利于最小化 *max*.

以图 1 为例, 算法运行到当前步骤, (0,6,7,8,9,10) 是最长环路, 次序已经是最优; (0,5,3,4) 是另外一条稍短的环路, 次序不是最优. 此时对 (0,5,3,4) 进行优化得到 (0,3,4,5), 然后从 (0,6,7,8,9,10) 中移动“6”到 (0,3,4,5) 可以得到 (0,3,4,5,6), 这是一条新的最长环路, 而且比前一条最长环路更短.

因此, 将解的适应值设计为一个二元组 (max, $total$), 规定: 解 S_1 优于解 S_2, 当且仅当 (max_1<max_2) 或 (max_1=max_2 且 $total_1$<$total_2$).

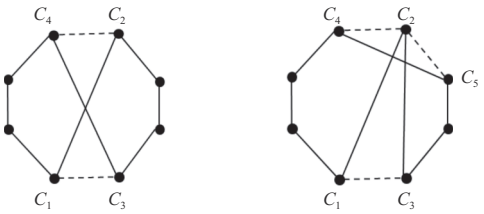
本文局部搜索算子通过依次检查 S 中的每一个位置来完成. 下文中, 将被检查位置上的城市标记为 c_1, 它在 S 中的后一个城市标记为 c_3, c_1 的邻域 $N(c_1)$ 中的城市的标记为 c_2, c_4 和 c_5 分别是 c_2 在 S 中的后一个城市和前一个城市. 按照 Lin-kernighan 算法的建议, $N(c_1)$ 取离 c_1 最近的 6 个城市.

我们的局部搜索方法是: 依次检查 S 的每一个位置, 对于该位置上的城市 c_1, 对 $N(c_1)$ 中每一个 c_2, 做两种尝试: 先尝试转置 c_2 与 c_3 之间 (包含 c_2 与 c_3) 的次序, 得到 S', 如果 S'优于 S, 则用 S'代替 S; 否则尝试将 c_2 移动到 c_1 和 c_3 之间, 得到 S', 如果 S'优于 S, 则用 S'代替 S. S 包含 $n+m$ 个位置, 如果连续检查 $n+m$ 个位置都无法优化 S, 算子停止. 转置和移动合称“reverse/move”.

c_2 和 c_3 可能位于 S 中的同一环路, 也可能位于两条不同的环路, 以下分别进行讨论.

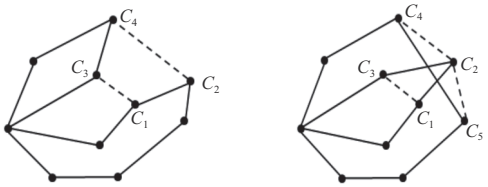
当 c_2 和 c_3 在同一环路中, 转置 c_2 和 c_3 之间的次序将删除两条边并新增两条边, 效果如图 2(a), 例如: $cycle$=(0,1,2,3,4,5,6), c_2=1, c_3=5, $reverse$ ($cycle$)=(0,5,4,3,2,1,6); 移动 c_2 将删除三条边并新增三条边, 效果如图 2(b), 例如: $cycle$=(0,1,2,3,4,5,6), c_2=1, c_3=5, $move$($cycle$)=(0,2,3,4,1,5,6).

当 c_2 和 c_3 属于分属两条环路. 转置 c_2 和 $c3$ 之间的次序, 将对两条环路进行重组, 效果如图 3(a), 例如: S=(0,1,2,0,5,3,4,0,6,7,8,9,10), c_2=1, c_3=7, $reverse$(S)=(0,7,6,0,4,3,5,0,2,1,8,9,10). 移动 c_2, 相当于将 c_2 从一条环路中移除, 插入到另一条环路, 效果如图 3(b), 例如: S=(0,1,2,0,5,3,4,0,6,7,8,9,10), c_2=1, c_3=7, $move$(s)=(0,2,0,5,3,4,0,6,1,7,8,9,10).



(a) 转置 (b) 移动

图 2 一条环路中的转置/移动



(a) 转置 (b) 移动

图 3 跨两条环路的转置/移动

我们在经典 TSP 的公开测试数据 TSPLIB 上比较 reverse/move 与 2-opt. 经典 TSP 问题相当于 m=1 的 MTSP 问题, 其上的测试结果能够反映局部搜索算子的寻优能力. 对每个测试例, 分别产生 800 个随机的初始解, 然后各用 reverse/move 与 2-opt 两种局部搜索进行优化, 得到 800 个最终解. 从两个方面进行比较: 1) 环路长度 (*length*) 的平均值, 2) 检查位置数 (*checked-points*) 的平均值, 结果如表 1.

表 1 reverse/move 与 2-opt 的搜索能力比较

TSPInstance	*length*		*checked-points*	
	2-opt	reverse/move	2-opt	reverse/move
EIL51	451	440	199	231
EIL101	680	661	570	609
A280	3041	2898	2774	2821
LIN318	50 023	48 224	4528	4406
Att532	102 744	97 770	8687	8887
Pr1002	339 131	318 357	22 054	22 580

表 1 显示, 对每一个测试例, 两种局部搜索算法在停止时, reverse/move 检查的位置数略大于 2-opt, 与此同时, 在搜索的结果上, reverse/move 的明显好于 2-opt.

3 竞争搜索算法

当 reverse/move 算子停止搜索时, 对所得到的解 S, 选择其中一个片段, 转置其次序, 然后再执行 reverse/move, 可能得到更优的解. 以图 4 为例, 图 4(a) 是 reverse/move 得到的一个解, 图 4(b) 是一次转置的结果, 图 4(c) 和图 4(d) 是再次执行 reverse/move 的中间过程和结果.

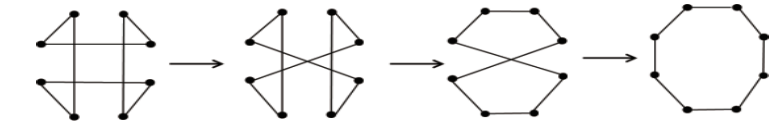

(a) 算子停止时的结果 (b) 转置一个片段 (c) 再次搜索的中间解 (d) 再次停止时的结果

图 4 转置一个片段后再次执行 reverse/move

用一对城市号 (c_{start}, c_{end}) 来标记 S 中的一个片段, 这样的对共有 $(n+m)\times(n+m-1)/2$ 个, 构成 S 的候选对集. 用随机方式来选择一对城市, 转置其所标记的片段, 然后再次执行 reverse/move, 可能有三种结果: 1) 得到更好的解, 2) 恢复到转置之前的解, 3) 得到变差的解. 每选出一对城市, 就从 S 的候选对集中将其删除, 直到候选对集为空.

根据 reverse/move 的以上特点, 我们提出一种群体迭代策略: “搜索-选优-变异-搜索”, 用图 5 说明. 图 5 中的横坐标是解空间, 纵坐标是解的适应值. 如果一组解经过局部搜索得到相同的局部最优解, 则将这组解集中在一个区域, 该区域中适应值最小的解就是该区域的局部最优解. 选择最好的若干个局部最优解, 对其进行变异 (随机的片段转置), 得到的一部分新的解将到达新的区域, 经再次搜索得到新的局部最优解. 如此迭代, 逐步提高局部最优解的质量, 直至最好的若干个局部最优解的候选对集全部为空.

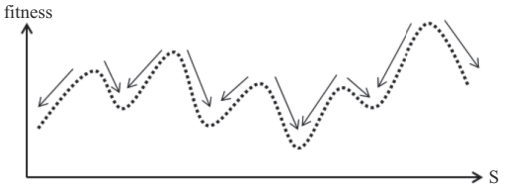


图 5 多个解经过局部搜索得到相同的局部最优解

基于这一策略, 我们设计了竞争搜索算法 (CSA), 如算法 1.

算法 1. CSA 算法

1) 设定种群规模 p 和选择比例 θ(θ 建议取 0.2), 随机生成 p 个初始解, 对每个解执行 reverse/move.
2) 对所有解按适应值从小到大排序, 保留 $\theta \times p$ 个排名靠前且互异的解.
3) 对保留的解, 从其候选对集中随机选 $(1-\theta)/\theta$ 个城市对, 并从候选对集中删除这些对; 对保留解, 分别转置这些片段产生新的解.
4) 在新的解上执行 reverse/move, 然后加入种群.
5) 循环执行 2) 到 4), 直至所有保留解的候选集为空.
6) 输出群体中的最优解.

CSA 算法与保留精英的遗传算法相似, 其特异性在于: 用局部搜索 reverse/move 取代杂交操作; 用最优且互异的若干个体作为父代个体, 取代概率性选择; 用固定的小尺度变异 (转置一个片段) 替代概率性变异.

4 实验

我们用文献[4]的 12 个测试例和文献[6]的 3 个测试例作为实验数据. 文献[4] 的 12 个测试例中的城市数据是二维平面坐标, 包括 MTSP-51 的 3 个问题 (m=3, m=5, m=10), MTSP-100 的 4 个问题 (m=3, m=5, m=10, m=20) 和 MTSP-150 的 5 个问题 (m=3,m=5, m=10, m=20, m=30), 用第一个城市作为中心城市. 文献[6]的 3 个测试例中的数据是 128 个城市的经纬度和距离矩阵, 包括 sgb128(m=10, m=15,m=30), 用第一个城市作为出发城市.

我们在 2.8 GHz, 2 Core, 4 G RAM 的 Windows 8.1 系统上实现了 CSA, 在实验中种群规模 p 设置为 50. 表 2 给出了 CSA 在 15 个测试例上得到的最小的最长环路长度 (*best max*)、迭代次数 (*iterations*) 和计算时间 (*time*).

表 2 CSA 算法的结果、迭代次数和时间

instances	m	*best max*	*iterations*	*time*(s)
MTSP-51	3	159	4	0.23
	5	118	5	0.36
	10	112	8	0.81
MTSP-100	3	8507	13	2.7
	5	6772	50	10.8
	10	6358	14	6.8
	20	6358	9	7.0
MTSP-150	3	13 039	36	25.3
	5	8416	32	24.7
	10	5593	15	21.6
	20	5246	26	34.3
	30	5246	15	24.5
Sgb128	10	2748	28	12.8
	15	2273	32	13.5
	30	2082	24	10.4

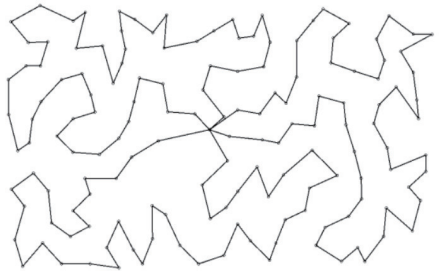
表 2 说明, CSA 能在较短时间内终止, 满足实际应用的需求.

对于文献[4]的 12 个测试例, CSA 所得的 *best max* 与文献[6–9]的结果比较如表 3.

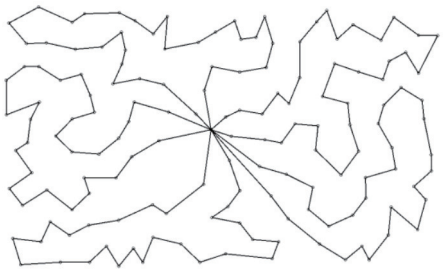
表 3 几种算法在文献[4]测试例上的 *best max* 比较

算法	m	TCX	ACO	GGA-SS	IWO	**CSA**
MSTP-51	3	203	160	161	160	**160**
	5	154	118	119	118	**118**
	10	113	112	112	112	**112**
MTSP-100	3	12 726	8817	8542	8509	**8507**
	5	10 086	6964	6852	6767	**6767**
	10	7064	6363	6370	6358	**6358**
	20	6402	6356	6359	6358	**6358**
MTSP-150	3	18 019	13 885	13 268	13 168	**13 038**
	5	12 619	9270	8660	8479	**8417**
	10	8054	6132	5875	5594	**5594**
	20	5673	5250	5252	5246	**5246**
	30	6402	5246	5247	5246	**5246**

由表 3, 对文献[4]的 12 个问题, 在所有算法中, CSA 的结果都优于或等于现有算法的最好结果, 其中两个测试例 MTSP-150 (m=3, m=5) 的解展示在图 6 中.



(a) MTSP-150, m=3, *max*=13038



(b) MTSP-150, m=5, *max*=8417

图 6 CSA 在文献[6]问题上的两个解

对于文献[6]的 3 个测试例, CSA 所得的 *best max* 与文献[6–9]的结果比较如表 4.

由表 4, 对文献[6]的 3 个测试例, CSA 的结果有大幅度改进. 由于文献[6]提供的 128 城市的坐标用经纬

度来表示, 在二维平面较难展示, 下面给出 m=10, max=2748 的解, 其中每一对括号包含的是一条环路.

表 4 几种算法在文献[6]测试例上的 best max 比较

	m=10	m=15	m=30
TCX	5912	5295	4003
ABC(FC)	5172	3819	3456
ABC(VC)	4660	3958	3811
IWO	4450	3665	3494
CSA	**2748**	**2273**	**2082**

cycle1: (0,95,65,22,81,44,78,23,61,122,101,46,82,107,4,93)

cycle2: (0,89,127,111,74,114,126,80,8,98,7,15,14)

cycle3: (0,73,59,109,39,60,116,92,3,76,35,105,40,66,62,17,53)

cycle4: (0,21,16,52,77,28,10,11,34,125,106,102,30,90)

cycle5: (0,88,108,54,118,112,25,85,97,87,27,42,86,41)

cycle6: (0,100,13,83,36,67,1,75,9)

cycle7: (0,47,70,68,120,50,99,84,31,55,63,123,12,58,32)

cycle8: (0,57,48,49,2,56,96,117,124,113,104)

cycle9: (0,26,18,45,115,94,64,43,91,29,37)

cycle10: (0,130,71,69,79,33,72,103,51,20,110,19,24,121,6,38,5,119)

5 结论

为了求解最大最小目标的多旅行商问题, 在对现有文献进行研究的基础上, 提出了竞争搜索算法(CSA), 与近期文献中的相比, 明显提高了解的质量. 局部搜索算子和变异算子是 CSA 算法中的两个关键算子. 我们曾采用多种不同的变异算子, 包括: 随机移动一个城市、随机转置一个片段、随机转置两个或多个片段以及这些操作的组合, 我们观察到, 采用不同的变异算子, 在收敛速度和最终解质量上有明显差异, 其中, 随机转置一个片段的变异方法明显好于其他方法. 改进变异方法, 可能进一步提高 CSA 算法性能.

参考文献

1 Malmborg CJ. A genetic algorithm for service level based vehicle scheduling. European Journal of Operational Research, 1996, 93(1): 121–134. [doi: 10.1016/0377-2217(95)00185-9]

2 Saleh HA, Chelouah R. The design of the global navigation satellite system surveying networks using genetic algorithms. Engineering Applications of Artificial Intelligence, 2004, 17(1): 111–122. [doi: 10.1016/j.engappai.2003.11.001]

3 Trigui S, Koubâa A, Cheikhrouhou O, *et al*. A clustering market-based approach for multi-robot emergency response applications. 2016 International Conference on Autonomous Robot Systems and Competitions. Braganca, Portugal, 2016: 137–143. [doi: 10.1109/ICARSC.2016.14]

4 Carter AE, Ragsdale CT. A new approach to solving the multiple traveling salesperson problem using genetic algorithms. European Journal of Operational Research, 2006, 175(1): 246–257. [doi: 10.1016/j.ejor.2005.04.027]

5 Brown EC, Ragsdale CT, Carter AE. A grouping genetic algorithm for the multiple traveling salesperson problem. International Journal of Information Technology & Decision Making, 2007, 6(2): 333–347. [doi: 10.1142/S0219622007002447]

6 Yuan S, Skinner B, Huang SD, *et al*. A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms. European Journal of Operational Research, 2013, 228(1): 72–82. [doi: 10.1016/j.ejor.2013.01.043]

7 Singh A, Baghel AS. A new grouping genetic algorithm approach to the multiple traveling salesperson problem. Soft Computing, 2009, 13(1): 95–101. [doi: 10.1007/s00500-008-0312-1]

8 Liu WM, Li SJ, Zhao FG, *et al*. An ant colony optimization algorithm for the Multiple Traveling Salesmen Problem. Proceedings of the 4th IEEE Conference on Industrial Electronics and Applications. Xi'an, China, 2009: 1533–1537. [doi: 10.1109/ICIEA.2009.5138451]

9 Venkatesh P, Singh A. Two metaheuristic approaches for the multiple traveling salesperson problem. Applied Soft Computing, 2015, (26): 74–89. [doi: 10.1016/j.asoc.2014.09.029]