

改进蚁群算法求解 TSP^①

胡粼粼, 葛红

(华南师范大学 计算机学院, 广州 510631)

摘要: 针对蚁群算法存在易陷入局部寻优、收敛缓慢等缺陷, 提出一种基于邻接矩阵的两层搜索决策来选择转移路径的方法对蚁群算法进行改进, 求解 TSP 问题。通过实验及分析, 验证了该算法具有较好性能。

关键词: 改进蚁群算法; TSP; 路径寻优; 两层搜索决策; 全局收敛

Improved Ant Colony Algorithm for Solving TSP

HU Lin-Lin, GE Hong

(School of Computer Science, South China Normal University, Guangzhou 510631, China)

Abstract: Aiming at solving the problem of falling into local optimization easily and slow convergence, this paper proposes a method that adopts a bi-level searching decision based on adjacent matrix in choosing diverting path to improve ant colony algorithm and solve TSP problem. The experiment results show that the new algorithm has better performance.

Key words: improved ant colony algorithm; TSP; path optimization; bi-level searching decision; global convergence

蚁群算法(Ant Colony Optimization, ACO)又叫蚂蚁算法, 最早由意大利学者 Dorigo M^[1,2]等于 20 世纪 90 年代初提出。蚁群算法是对自然界中真实蚂蚁觅食的群体行为的一种模拟。蚂蚁在寻找食物的过程中会对经过的路径释放信息素, 通过对感知信息素浓度来选择转移路径, 最终所有蚂蚁都会走从巢穴到食物的最短路径。从提出起到目前, 国内外许多学者对该算法做了大量研究, 将其运用到诸多领域, 并取得了丰富的成果。典型应用如旅行商、车间作业调度、网络路由、车辆路径问题、二次指派等。

但是蚁群算法也有不足, 算法的收敛性和全局寻优性是制约算法被广泛运用的瓶颈。围绕这个问题, 不少学者提出了改进方法^[3-8]。本文在蚁群系统上, 重点研究新的搜索策略对蚁群算法进行改进, 并应用于求解 TSP 问题中, 通过实验结果及分析, 改进算法的收敛速度和全局寻优性都得到较好的提高。

1 蚁群算法

1.1 基本蚁群算法

TSP 的简单描述为: 给定个城市, 旅行商从某一

个城市出发, 求一条经过各城市且只经过一次, 最后回到原出发城市的最短路径。

m 为蚂蚁数目, $\tau_{ij}(t)$ 为 t 时刻路径 (i, j) 上的信息量。用禁忌表 $tabu_k$ 来记录蚂蚁 $k(k=1, 2, \dots, m)$ 当前走过的城市, 并作动态调整。蚂蚁 k 在搜索过程中, 根据各路径上的信息量及启发信息来计算状态转移概率。 $p_{ij}^k(t)$ 表示在 t 时刻蚂蚁 k 由站点 i 转移到站点 j 的状态转移概率:

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}(t)]^\beta}{\sum_{s \in allowed_k} [\tau_{is}(t)]^\alpha \cdot [\eta_{is}(t)]^\beta}, & \text{若 } j \in allowed_k \\ 0, & \text{否则} \end{cases} \quad (1)$$

式中, $allowed_k$ 表示蚂蚁 k 下一步允许选择的城市; α 表示信息启发式因子, 反映了蚂蚁在搜索路径时信息素所起作用强度, 其值越大, 则蚂蚁越倾向于选择其他蚂蚁经过的路径, 探索其他路径能力减弱; β 为期望启发式因子, 表示能见度, 其值越大, 则蚂蚁的转移概率越接近贪心规则; $\eta_{ij}(t)$ 表示启发函数, 表示蚂蚁从城市 i 转移到城市 j 的期望程度, 表达式如下:

$$\eta_{ij}(t) = \frac{1}{d_{ij}}$$

① 基金项目: 安徽省教育厅自然科学基金(2005KJ004ZD)

收稿时间: 2011-08-25; 收到修改稿时间: 2011-09-23

d_{ij} 为相邻两个城市之间的距离。

完成一次循环后, $(t+n)$ 时刻对路径 (i, j) 上的信息素更新为:

$$\tau_{ij}(t+n) = (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (2)$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^m \Delta\tau_{ij}^k(t) \quad (3)$$

$\rho(0 < \rho < 1)$ 表示信息素挥发系数, $(1-\rho)$ 表示信息素残留因子, 避免信息素的无限积累, $\Delta\tau_{ij}(t)$ 表示本次循环中路径 (i, j) 上的信息增量, 初始时刻 $\Delta\tau_{ij} = 0$, $\Delta\tau_{ij}^k(t)$ 表示蚂蚁 k 在本次循环中释放在路径 (i, j) 上的信息素量。Dorigo M 提出了三种不同的信息素更新策略, 其中 Ant-Cycle 模型为全局更新信息素, 求解 TSP 性能较好, 表达式如下:

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_k}, & \text{若蚂蚁 } k \text{ 经过路径 } (i, j) \\ 0, & \text{否则} \end{cases} \quad (4)$$

Q 表示信息素强度, 一定程度上影响算法的收敛速度, L_k 表示蚂蚁 k 在本次循环中路径长度。

1.2 群蚁系统

蚁群系统(Ant Colony System, ACS)是在基本蚂蚁算法上引进了新的转移规则和信息素更新策略。转移规则采用伪随机比例的方式。则位于城市 i 的蚂蚁 k 选择城市 j 的规则为:

$$j = \begin{cases} \arg \max \{ \tau_{ij}^\alpha \cdot \eta_{ij}^\beta \}, & \text{若 } q < q_0 \\ J, & \text{否则} \end{cases} \quad (5)$$

q 是均匀分布在 $[0,1]$ 上一个随机变量, $q_0(0 \leq q_0 \leq 1)$ 是一个算法参数。 J 为式(1)。蚂蚁 k 以 $q < q_0$ 的概率以确定城市转移, 以 $q \geq q_0$ 的概率去探索可以转移的城市。

同时, 信息素更新策略也不同。每次循环结束后, 只有至今最优的路径才可以进行全局信息素的更新, 更新公式为:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs} \quad (6)$$

而局部信息素更新公式为:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \tau_0 \quad (7)$$

$\xi(0 < \xi < 1)$ 是一个参数, τ_0 表示信息素的初始浓度。局部信息素更新规则, 使得路径 (i, j) 的信息素浓度降低, 则蚂蚁经过该路径的可能性降低, 这样就增加了探索未走过的路径概率, 使得算法不易陷入停滞状态。

2 改进蚁群算法

经研究发现, 在求解 TSP 问题的最优路径寻优搜索中, 下一个选择的总是偏向选择最近的一个城市^[4]。因此, 算法在城市数据结构中加入邻接矩阵, 该邻接矩阵为与该城市距离最近一些城市。邻接矩阵中城市个数与城市规模及相邻城市间的距离有关。蚁群算法中蚂蚁在搜索路径时, 从该城市邻接矩阵表中搜索下一个允许访问的城市, 而不是从整个剩下的城市中去搜索, 从而减少了算法解空间中劣质解量, 同时缩小了解空间的搜索范围, 这样就能加快算法收敛, 提高算法性能。

蚁群算法中蚂蚁在搜索路径时是根据路径上的信息素量来选择转移方向, 某路径信息素量过高或过低, 算法容易陷入局部寻优。针对以上问题, 本文采用两层搜索策略来引导蚂蚁在 t 时刻由城市 i 转移 j 。具体做法为, 蚂蚁 k 在搜索路径 (i, j) 时, 既考虑第一层路径 (i, j) 上的信息量及启发信息, 又考虑下一层路径 (j, r) 上的信息量及启发信息, 以增大蚂蚁感知信息素的层次范围。这样使算法扩大解的搜索范围, 能跳出局部信息素过高或过低的困境, 不易陷入局部寻优。则蚂蚁 k 在 t 时刻由城市 i 转移 j 的转移规则公式如下:

$$j = \begin{cases} \arg \max_{j \in allowed_{ki}} \{ [\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}(t)]^\beta \} \\ + \lambda \max_{r \in allowed_{kj}} \{ [\tau_{jr}(t)]^\alpha \cdot [\eta_{jr}(t)]^\beta \}, & \text{若 } q < q_0 \\ J, & \text{否则} \end{cases} \quad (8)$$

式中, $\lambda(0 \leq \lambda < 1)$ 为扩散因子, 表示第二层信息素扩散到第一层的浓度, λ 越大, 则蚂蚁 k 在 t 时刻由城市 i 转移 j 到的可能性越受由下一层路径 (j, r) 上的信息素浓度及启发函数影响; λ 越小, 则反之。 $allowed_{ki}$ 表示蚂蚁 k 下一步允许访问城市 i 的邻接城市列表。

每次循环结束后, 对信息素的更新采取式(6)及式(7)。同时对最优路径上城市的邻接城市进行额外增加信息素, 加强蚂蚁对邻接城市的访问的可能性, 以搜索出更优的路径。更新公式如下:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}, \quad j \in allowed_i \quad (9)$$

其中,

$$\Delta\tau_{ij}^{bs} = \frac{Q}{L^{bs}} \cdot \frac{1}{d_{ij}}, \quad (10)$$

L^{bs} 为最优路径长度。

3 算法步骤

步骤 1 城市数据读入, 计算路径 (i, j) 距离 d_{ij} , 邻接矩阵 $allowed_i$ 。

步骤 2 参数初始化。初始循环次数 $N_0 = 0$, 最大循环次数 N_{max} , 蚂蚁数 m , 城市数 n , 初始蚂蚁计数器 $k = 0$, 各路径上的初始信息量 $\tau_{(i,j)} = const$, 初始信息增量 $\Delta\tau_{(i,j)}(0) = 0$, 路径 (i, j) 距离 d_{ij} , 禁忌表 $tabu_k$, 参数 α, β, λ 。

步骤 3 循环次数 $N \leftarrow N + 1$ 。

步骤 4 蚂蚁数目 $k \leftarrow k + 1$, 将蚂蚁随机均匀置于各城市上。

步骤 5 随机产生数 $q, q_0 (0 < q, q_0 < 1)$ 。

步骤 6 若该蚂蚁的候选列表都已访问, 则转移到步骤 4, 否则转移到步骤 7。

步骤 7 若 $q < q_0$, 则蚂蚁以公式(8)中规则转移城市, 否则以概率转移城市。

步骤 8 修改禁忌表 $tabu_k$ 及候选列表 $allowed_k$,

$i \leftarrow j$ 。

步骤 9 若蚂蚁回到起点, 则转移到步骤 4, 否则转移到步骤 10。

步骤 10 若 $k < m$ 则转移到步骤 4, 否则转移到步骤 11。

步骤 11 按照公式(6),(7)和(9)更新各路径上的信息量。

步骤 12 若 $N < N_{max}$, 还原禁忌表 $tabu_k$ 及候选列表 $allowed_k$ 到初始化状态, 信息增量 $\Delta\tau_{(i,j)}(0) = 0$, 转移到步骤 3, 否则转移到步骤 13。

步骤 13 输出结果。

4 算例分析

以国际上通用的 TSP 测试库 TSPLIB 中 Oliver30 为例, 将本文改进算法与基本蚂蚁算法及蚁群系统进行比较, 来检验算法的性能。编程语言为 C++, 编译环境为 Microsoft Visual C++ 6.0。实验中参数设置分别为: $\alpha = 1, \beta = 3, \rho = 0.1, q_0 = 0.4, Q = 100, \lambda = 0.8$, 对各种算法测试 40 次。测试结果如表 1。

表 1 3 种算法结果对比

算法	平均迭代次数	平均解	最优解	最差解	相差	40 次运行总时间(S)
基本蚁群算法	137	430.238	425.820	438.243	12.423	46.648
蚁群系统算法	77	428.334	423.741	435.776	12.035	11.687
改进蚁群算法	42	423.813	423.741	424.646	0.723	7.407

图 1 是 Oliver30 问题中运行改进蚁群算法和蚁群系统算法程序画出收敛曲线的叠加图, 其中连接“□”的曲线表示改进蚁群算法的收敛曲线, 另一条则为蚁群系统算法的收敛曲线。从图中可以看出, 改进后的算法的收敛速度明显高于蚁群系统, 前期振荡也较小, 能在较少的迭代次数中找到最优解, 较好地避免局部寻优。

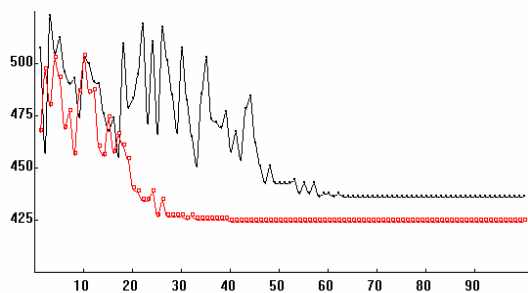


图 1 改进蚁群算法和蚁群系统算法收敛曲线

5 结语

本文在蚁群系统上, 提出了基于邻接列表的两层搜索决策对蚁群算法进行改进。通过 TSP 算例进行实验, 并对比了实验结果, 显示本文改进算法的收敛速度和稳定性有较明显的提高, 能较好克服了蚁群算法的缺点。由此证明了本文采取的改进方法是可行的。

参考文献

- Colomi A, Dorigo M, Maniezzo V, et al. Distributed optimization by ant colonies. Proc. of the 1st European Conference on Artificial Life. 1991: 134-142.
- Dorigo M. Optimization, learning and natural algorithms(in Italian). Dipartimento di Elettronica, Politecnico di Milano, IT, 1992.
- 赵吉东, 胡小兵, 刘好斌. 改进的蚁群算及其在 TSP 中的应用. 计算机工程与应用, 2010, 46(24): 51-52.

(下转第 208 页)

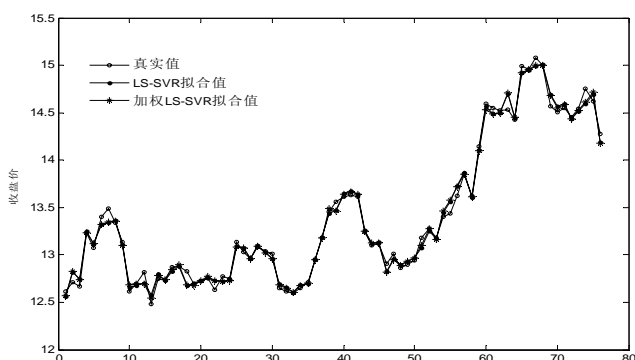


图 1 LS-SVR 与加权 LS-SVR 对训练样本的拟合

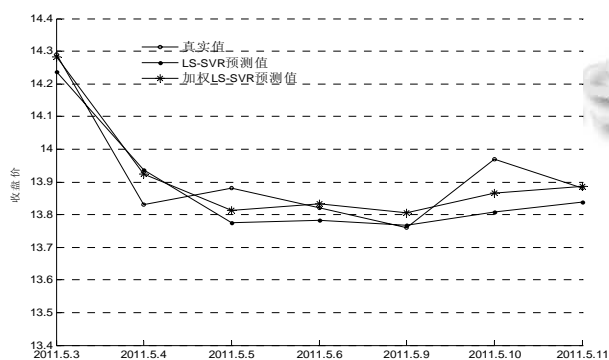


图 2 LS-SVR 与加权 LS-SVR 对测试样本的预测

5 总结

本文考虑到不同特征对于所预测问题的相关程度不同,提出了对于不同的特征指数施加不同的权重,同时根据参数的选取对于预测效果的影响,提出了基于特征指数加权的二乘支持向量回归机的模型,

解决了样本特征对回归预测结果相关程度的差异性和减少参数选取对预测效果的影响性以及特征加权方法的有效性等问题。同时为验证该算法的有效性,选取实际股票价格数据利用加权 LS-SVR 算法与传统 LS-SVR 算法进行实验验证,并对拟合与预测的结果进行比较,实验结果表明加权 LS-SVR 较 LS-SVR 有更好的回归预测能力,效果基本令人满意,说明该算法具有一定的实用价值。但同时本文还存在一定不足,比如对于特征指数权重的确定,对于测试样本的预测时间长短等,因此下一步的主要任务是进一步的探讨研究如果能更好的确定特征权重,使得特征加权支持向量回归机算法得到更好的改进,以便能有效的应用于实际中。

参考文献

- 1 金凌霄,张国基.基于特征加权的支 持向量回归机研究.计算机工程与应用,2007,43(6):42-44.
- 2 邓乃杨,田英杰.支持向量机理论、算法与拓展.北京:科学出版社,2009.100-105.
- 3 邓乃杨,田英杰.数据挖掘中的新方法-支持向量机.北京:科学出版社,2004.226-228.
- 4 Vapnik V. Statistical learning theory. New York:Wiley, 1998.
- 5 王定成.支持向量机建模预测与控制.北京:气象出版社,2009.96-97.
- 6 罗党.灰色决策问题分析方法.郑州:黄河水利出版社,2005.29-31.
- 7 <http://www.esat.kuleuven.be/sista/lssvml>.

(上接第 200 页)

- 4 侯文静,马永杰,张燕,石玉军.求解 TSP 的改进蚁群算法.计算机应用研究,2010,27(6):2087-2089.
- 5 黄永.改进蚁群算法及其在公交线网优化中的应用[硕士学位论文].上海:华东师范大学,2009.
- 6 Qi CM. An ant colony system hybridized with randomized algorithm for TSP. Proc. of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. 2007.

- 461-465.
- 7 吴斌,史忠植.一种基于蚁群算法的 TSP 问题分段求解算法.计算机学报,2001,24(12):1328-1333.
- 8 于滨,杨忠振,程春田.并行蚁群算法在公交线网优化中应用.大连理工大学学报,2007,47(3):211-214.
- 9 段海滨.蚁群算法原理及其应用.北京:科学出版社,2005.
- 10 马良,朱刚,宁爱兵.蚁群优化算法.北京:科学出版社,2008.