控制与决策
Control and Decision

基于混合反馈机制的扩展蚁群算法

冯振辉, 肖人彬

引用本文:
冯振辉,肖人彬. 基于混合反馈机制的扩展蚁群算法[J]. 控制与决策, 2022, 37(12): 3160−3170.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2021.0846

基于混合反馈机制的扩展蚁群算法

冯振辉, 肖人彬†

(1. 华中科技大学 人工智能与自动化学院，武汉 430074；2. 华中科技大学 人工智能研究院，武汉 430074)

摘 要: 由于传统蚁群算法基于正反馈机制的单一搜索方式,导致其存在收敛速度慢、易陷入局部极值的缺点. 针对该问题提出一种基于混合反馈机制的扩展蚁群算法(MF-ACO),该算法在传统蚁群算法的基础上定义一种具有较强全局搜索能力的扩展型蚂蚁,帮助算法跳出局部极值;参考蚁群劳动分工行为,设计基于刺激-响应分工模型的负反馈平衡机制,动态平衡算法的收敛能力和全局搜索能力;最后依据分工模型对蚂蚁个体的信息素更新策略进行改进,进一步加快算法收敛速度. 以多个TSP实例作为测试对象进行仿真实验,实验结果验证了所提算法的优越性,并将该算法用于机器人路径规划问题,在实际应用中进一步验证了所提算法的有效性.

关键词: 正反馈；混合反馈机制；局部极值；劳动分工；刺激-响应分工模型

中图分类号: TP18 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2021.0846

引用格式: 冯振辉, 肖人彬. 基于混合反馈机制的扩展蚁群算法[J]. 控制与决策, 2022, 37(12): 3160-3170.

Extended ant colony algorithm based on mixed feedback mechanism

FENG Zhen-hui, XIAO Ren-bin†

(1. School of Artificial Intelligence and Automation，Huazhong University of Science and Technology，Wuhan 430074，China；2. Institute of Artificial Intelligence，Huazhong University of Science and Technology，Wuhan 430074，China)

Abstract: The traditional ant colony algorithm is based on positive feedback search way, leading to the existence of slow convergence speed and shortcoming of easily trapped in local minima. This paper proposes a kind of extended ant colony algorithm based on the mixed feedback mechanism (MF-ACO). On the basis of the traditional ant colony algorithm, the algorithm defines a kind of extension type ants, which have strong global search ability, to help the algorithm get out of local minima. In addition, the negative feedback balance mechanism based on stimulus-response model, the convergence ability and global search ability of dynamic balance algorithm are designed by referring to the labor division behavior of ant colony. Finally, on the basis of the labor division model, the individual ants pheromone update strategy is improved to further accelerate the algorithm convergence speed. Using a multiple TSP instance as the test object to conduct simulation experiments, the experimental results show the superiority of the proposed algorithm, and then this algorithm is applied to the robot path planning problem, to verify the effectiveness of the proposed algorithm in actual application.

Keywords: positive feedback；mixed feedback mechanism；local minima；labor division；stimulus-response model

0 引 言

蚁群优化算法(ant colony optimization, ACO)是由意大利学者Dorigo等提出的一种模拟蚂蚁觅食行为的仿生智能算法[1]. 蚂蚁在觅食过程中借助信息素进行信息的交流和传递,自主选择下一步移动方向,帮助种群寻找最佳觅食路径,具有正反馈特点[2]. 蚁群算法以信息素更新和概率转移为基本操作,与其他群智能优化算法(如粒子群算法、遗传算法、模拟退火算法、差分进化算法等)相比,蚁群算法具有正反馈并行机制、算法鲁棒性强和易与其他算法结合等优点,适合求解各种组合优化问题[3]. 目前已在多个领域获得了广泛应用,如旅行商(TSP)问题[4]、路由优化问题[5]、机器人路径规划[6]、车辆规划[7]、流程调度[8]、布局优化[9]领域等.

蚁群算法本质上是具有正反馈特性的启发式算法,在处理复杂问题时,算法通过正反馈的方式积累高质量路径的信息素浓度,往往可以得到较优解,然而如果当前较优解不是全局最优解,则算法会不可避免地陷入局部极值[10],因此正反馈特性保证了蚁群算法的稳定性和较好的搜索能力,但也导致了算法容易陷入局部极值. 针对这一缺点,国内外研究者进行了诸多研究. 文献[11]对蚁群算法进行权重参数改

收稿日期: 2021-05-14；**录用日期:** 2021-08-18.

基金项目: 科技创新2030——“新一代人工智能”重大项目(2018AAA0101200).

†通讯作者. E-mail: rbxiao@hust.edu.cn.

进,提出了一种基于信息素浓度和最优解的动态加权信息素更新机制;文献[12]引入启发信息递减系数,提出了初始信息素不均衡分配原则,动态调整蚁群算法的信息素累积速率,一定程度上降低了陷入局部极值的可能性;文献[13]根据所提出的信息素初始化策略,在算法陷入局部最优时重新初始化各路径的信息素浓度,以跳出当前极值;文献[14]利用问题对象的空间信息,采用k-均值聚类的方法对问题进行分解,定义了多类蚂蚁寻找最优路径. 上述文献都对算法进行了相关改进,一定程度上避免了算法陷入局部极值的缺点,但大多是引入动态参数调整机制或者信息素初始化机制,仍属单一正反馈搜索策略下进行的改进,依然存在陷入局部极值不易跳出的缺点. 对此本文参考群智能劳动分工策略,在传统蚁群算法基础上引入蚁群刺激-响应分工机制,在正反馈搜索的基础上补充负反馈分工加以平衡.

劳动分工作为群体智能方法的重要组成部分[15],是社会性昆虫族群中的一种普遍现象,表现为不同的个体执行不同的任务,在劳动分工作用下,个体执行的任务会随着环境动态调整,使群体表现出环境适应性. 文献[16]给出了蚁群等社会性昆虫的劳动分工行为描述,目前已证实的劳动分工机制主要有刺激-响应机制和激发-抑制机制两种[17],其中刺激-响应机制是一种用于描述蚁群系统劳动分工行为的内在机制;文献[18]详细分析了刺激-响应分工机制的作用原理,并给出了具有负反馈特性的数学表达模型. 近年来,随着群体智能的发展,群智能劳动分工策略也涌现出一系列成果,被广泛应用于各类分配问题,如群体利益分配[19]、空间分配[20]、时间分配[21]、任务分配[22]等.

此外,在传统的蚁群算法中,算法初期收敛速度较慢,计算时间较长. 根据这一缺点,目前已有的解决方式主要分为两种:一是改进信息素更新规则[23-24],二是改进或增加算法的启发式信息[25-26]. 以上方法均可有效地提高算法的收敛速度,但相比较而言,引入启发式信息可能会造成算法规模和复杂程度增加,故本文采用改进信息素更新机制的方法,根据引入的刺激-响应分工机制,依据精英策略,设计一种基于刺激-响应分工模型的信息素更新机制,以进一步加快算法收敛速度.

基于以上内容,本文提出基于混合反馈机制的扩展蚁群算法(extended ant colony optimization based on mixed feedback mechanism, MF-ACO),为验证算法性能,将算法应用于TSP和机器人路径规划问题的求解中. 使用TSPLIB中多个实例作为测试对象,与基本蚁群算法(ACO)、最大最小蚂蚁系统(MMAS)、蚁群系统(ACS)[27]3种经典算法以及近年来改进算法[14, 28-30]进行对比,并将复杂地图下的机器人路径规划问题作为实际案例进行求解并与其他算法比较.

1　基本蚁群算法

蚁群算法具有较强的鲁棒性和自组织特性,遵循正反馈机制. 一方面蚂蚁在构造路径时倾向于选择信息素浓度大的转移节点;另一方面通过信息素更新策略,使得蚂蚁在当前的最优路径上释放信息素. 算法分为概率转移和信息素更新两部分.

1.1　概率转移

在t时刻下,蚂蚁k在第i节点选择下一个节点j时的概率$P_{ij}(k)$的大小,是由信息素$\tau_{ij}(t)$和启发式信息$\eta_{ij}(t)$按照下式确定的:

$$P_{ij}(k)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\displaystyle\sum_{s\in A_k}[\tau_{is}(t)]^{\alpha}[\eta_{is}(t)]^{\beta}},\ j\in A_k;\\ 0,\ \text{otherwise.}\end{cases}\tag{1}$$

其中: $A_k(k=1,2,\ldots,m)$为蚂蚁k当前可选择的节点集合;α为信息素启发因子;β为期望启发式因子;$\eta_{ij}(t)$设定为(i,j)两节点间距离d_{ij}的倒数,即

$$\eta_{ij}(t)=1/d_{ij}.\tag{2}$$

1.2　信息素更新

蚂蚁经过的路径留下信息素,同时信息素会随时间挥发,因此需要对信息素进行更新. ACO算法在所有蚂蚁完成一次迭代后,对信息素τ进行更新:

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\rho\Delta\tau_{ij}(t).\tag{3}$$

$$\Delta\tau_{ij}(t)=\begin{cases}\dfrac{Q}{L_k},\ \text{蚂蚁}k\text{当前路径};\\ 0,\ \text{otherwise.}\end{cases}\tag{4}$$

其中: ρ为信息素挥发系数,$\Delta\tau_{ij}(t)$为蚂蚁k在t时刻后路径(i,j)上的信息素增量,Q为信息素强度系数.

2　刺激-响应分工机制

群智能劳动分工是一类自下而上的任务分配方法,特点是无需全局信息和中心控制,而且能够适应动态变化的环境. 本文引入的劳动分工机制为蚁群的刺激-响应机制.

2.1　分工机制

蚁群中个体对多个任务的选择分配是按照感知到的任务紧迫性程度划分的,每个任务都有一个环境刺激与其对应,刺激强度与任务的紧急程度成正比;同时蚂蚁个体对应每个任务也存在响应阈值,当一个任务的刺激强度超过了某个蚂蚁对应该任务的响应

阈值时,蚂蚁会以较大概率执行这项工作[31].

假设蚁群中某个蚂蚁有多个任务可以选择,令:S_i表示任务i的环境刺激量,描述执行该任务的外部驱动;θ_i表示个体对任务i的响应阈值,反映个体执行任务的内部倾向,此时个体执行任务i的概率Γ_i可表示为[18]

$$\Gamma_i = \frac{S_i^2}{S_i^2 + \theta_i^2}. \tag{5}$$

2.2 特性分析

由以上模型可知,执行任务i的概率与刺激S_i和阈值θ_i的大小有关,当$S_i \gg \theta_i$时,执行该任务的概率较高.此外在任务选择过程中,任务刺激S_i和个体阈值θ_i会随着族群需求和个体表现发生变化,使得个体可以灵活地执行任务. 例如S_i的变化取决于任务的执行情况,个体执行某任务,导致该任务的需求程度下降,刺激强度会随之降低,而刺激强度降低又使得个体执行任务的概率下降;反之,若未执行该任务,该任务需求未被满足,刺激强度随之升高,则增加任务的执行概率,显然以上变化呈现负反馈特性.

3 基于混合反馈机制的扩展蚁群算法设计

将刺激-响应分工机制与蚁群算法结合,利用负反馈分工特性对蚁群算法的单一正反馈搜索方式进行平衡,提出具有混合反馈机制的扩展蚁群算法.

3.1 扩展蚂蚁

在传统蚁群算法中,所有蚂蚁个体遵循相同的搜索规则,构造路径时依赖于信息素浓度. 本文在传统蚁群的基础上,除信息素搜索的常规蚂蚁外,另外引入扩展蚂蚁. 如图1所示,假设种群共有m只蚂蚁个体,其中常规蚂蚁数量为m_1,扩展蚂蚁个体数量为m_2,扩展蚂蚁不依赖信息素选择路径,而是选取目前已有路径进行组合交换,基于已知解的信息构造新的路径.

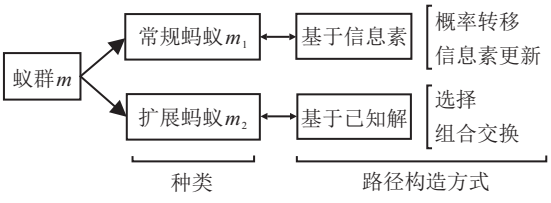


图 1 蚂蚁种群角色分类

3.1.1 选择策略

在每代蚂蚁种群开始搜索时,首先由常规蚂蚁构造m_1个当前路径作为已知解.随后算法依据正比选择(proportional selection)策略对当前路径进行选择,依次分配给每只扩展蚂蚁. 其中第i条路径被选择的概率P_i正比于该条路径的总长度L_i的倒数$1/L_i$,保证扩展蚂蚁优先选择长度越短的路径,即

$$P_i = \frac{1/L_i}{\sum_{i=1}^{m_1} 1/L_i}. \tag{6}$$

得到概率后,采用轮盘法旋转m_2轮,选出m_2个路径分配到每只扩展蚂蚁.为避免多个扩展蚂蚁重复选择到同一个当前路径,在选择时需剔除已被选择过的路径,也正是由于扩展蚂蚁基于常规蚂蚁的路径进行选择,在算法设计中扩展蚂蚁个体数量应小于等于常规蚂蚁的个体数量,即$m_2 \leqslant m_1$.

3.1.2 组合交换

选择完成后,每只扩展蚂蚁在除自身外其余扩展蚂蚁中随机挑选对象进行组合,组合交换方法的设计参考遗传算法中的两点交叉算子. 在遗传算法中,交叉算子得到的新个体差异性更强,可以产生的大量新个体使得算法具有较强的全局搜索能力[32].因此组合交换有利于跳出局部极值,保证全局寻优能力. 需注意路径在组合时节点首尾相连且不允许重复. 例如当第k只蚂蚁随机选中另一只个体g执行路径组合操作时,具体步骤如下:

1) 对个体k和g的节点进行index索引标注;

2) 在所有节点中随机选择两节点X、Y,如图2所示,随机选择$X=2$, $Y=5$;

3) 分别在X、Y两节点后剪开,得到交换对象k_{XY}、g_{XY};

4) 分别删除蚂蚁k和蚂蚁g的原有路径中关于g_{XY}、k_{XY}的重复节点;

5) 标注剩余节点索引,在X节点后插入交换对象的无重复节点,得到交换后的最终路径.

按照上述流程进行组合交换,其中每只扩展蚂蚁进行组合交换N_C次,并选择其中距离最短的一种组合保留构造新路径.

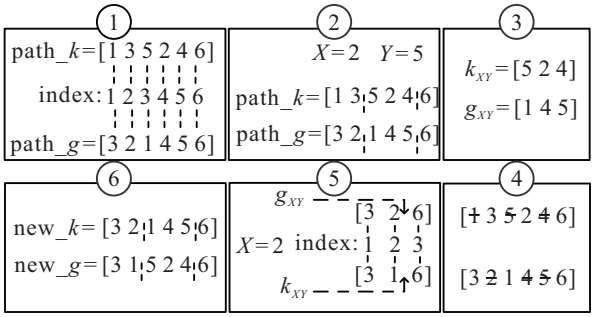


图 2 路径组合示意图

3.2 基于刺激-响应模型的角色平衡机制

如图3所示:常规蚂蚁参照信息素浓度构造个体路径,具有正反馈特性,保证算法有效收敛以及稳定性;扩展蚂蚁通过已有路径构造新路径,具有较强的全局搜索能力,保证种群多样性. 角色平衡机制则用

于动态调整蚁群中常规蚂蚁和扩展蚂蚁的数量占比,平衡算法的收敛能力和全局搜索能力. 该机制基于刺激-响应分工模型设计,本质上是一种任务负反馈模式下的调节方法.

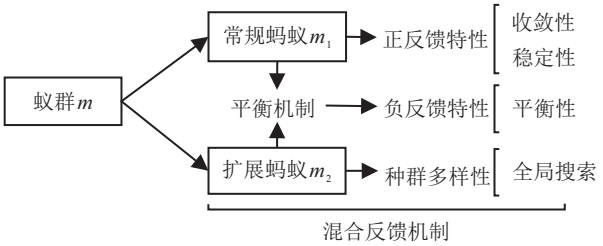


图 3　混合反馈机制示意图

3.2.1　负反馈调节模式

已知个体总数为m,常规蚂蚁为m_1,扩展蚂蚁为m_2. 在平衡机制运作过程中,需根据算法运行情况,对m_1和m_2数值动态调整,因此可将常规蚂蚁增加和扩展蚂蚁减少看作任务1($m_1\uparrow$ and $m_2\downarrow$);常规蚂蚁减少和扩展蚂蚁增加看作任务2($m_1\downarrow$ and $m_2\uparrow$). 当构造路径时,所有个体走过的路径越多、路径相似程度越弱,则种群多样性越强;反之,当种群逐渐聚焦于某些路径时,相似程度越强,种群多样性越弱.

如图4所示,当种群多样性减弱到某一程度时,个体均在相似路径上更新信息素,此时算法陷入极值点,按照平衡机制开始执行任务2,即扩展蚂蚁增加,常规蚂蚁减少,算法的全局搜索能力增强,进而增加种群多样性;进一步,当多样性增加至某一阈值时,个体的路径均有差异,虽然此时全局性很强,但算法整体呈现随机搜索状态,不利于快速收敛,因此按照平衡机制开始执行任务1,即常规蚂蚁增加,扩展蚂蚁减少,以增强信息素累积速率,增强算法正反馈特性,使算法快速收敛,但全局搜索能力下降.

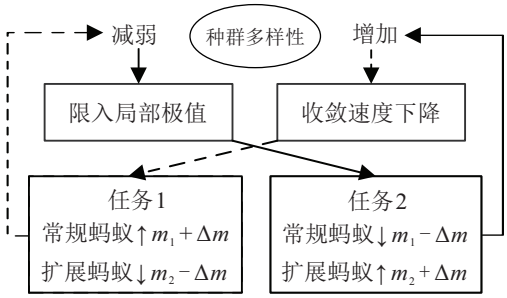


图 4　角色平衡机制负反馈示意图

算法运行过程中,可将每次增加或减少的蚂蚁数量设为Δm. 当Δm越大时,单步调整数量越大,负反馈响应速度越快,但易造成震荡,难以平衡算法的搜索能力和收敛速度,不利于算法精确求解;而当Δm越小时,单步调整数量越小,有利于算法找到两类蚂蚁之间的平衡点,有利于算法求得最优解,但负反馈速度较慢,所需时间较长.

综上所述,角色平衡机制是根据种群多样性情况对个体角色进行调整的一种负反馈调节方法,本文基于刺激-响应分工模型建立其数学模型.

3.2.2　数学模型

根据刺激-响应模型,需要通过环境刺激和响应阈值得到蚂蚁种群执行任务1和任务2的概率,首先设置外部环境刺激S和个体响应阈值θ.

环境刺激S描述了个体执行任务的外部驱动力,刺激越大,任务越紧急. 在常规蚂蚁和扩展蚂蚁两类角色中,二者数量之和为常数,角色之间属于竞争关系. 一般认为两类角色之间的平衡表现为:当种群多样性较好时,通过增加常规蚂蚁数量保证算法开采效率;当种群多样性较差时,通过增加扩展蚂蚁数量增加多样性,避免陷入局部极值,因此环境刺激与种群的多样性相关.

多样性表现为个体之间的差异,反映种群的分布情况. 本文的多样性度量方法参考文献[33]进行设计. 将多样性看成蚂蚁构造路径的差异性,这种差异可以由路径长度体现. 例如m只蚂蚁共得到的m条路径R_i,其长度分别为$L_i\ (i\in 1,2,\dots,m)$. 用第k只蚂蚁的路径长度L_k和种群平均长度L_{mean}之差体现这只蚂蚁到种群中心点的差距. 此时整体多样性度量方法如下:

$$\mathrm{diversity\_ant}=\frac{1}{m}\sum_{i=1}^{m}\sqrt{(L_i-L_{\mathrm{mean}})^2}. \tag{7}$$

其中:m为种群的蚂蚁数量,当所有蚂蚁到种群中心的差距之和越大时,认为种群多样性越大. 式(7)不依赖种群大小和空间范围,能较好地度量多样性大小. 给定角色转换任务的环境刺激如下:

$$S_{\mathrm{role}}=\mathrm{diversity\_ant}. \tag{8}$$

响应阈值θ描述了执行任务的内部倾向性. 阈值越小,倾向性越强. 不同算例的维度和求解空间大小不同,所以响应阈值应设置为可表征算例离散距离的参数. 在本文中θ与各节点间的平均距离D_{mean}有关,具体为

$$\theta_{\mathrm{role}}=n\cdot D_{\mathrm{mean}},$$
$$D_{\mathrm{mean}}=\frac{1}{(N-1)^2}\Big(\sum_{i=1}^{N}\sum_{j=1}^{N}d_{ij}\Big). \tag{9}$$

其中:d_{ij}为节点(i,j)间的距离,n为节点个数.

在刺激-响应分工机制中,角色调整任务的环境刺激和响应阈值共同决定个体执行任务的概率Γ,如下所示:

$$\Gamma_{\mathrm{role\_1}}=\frac{S_{\mathrm{role}}^2}{S_{\mathrm{role}}^2+\theta_{\mathrm{role}}^2}, \tag{10}$$

$$\Gamma_{\text{role\_2}} = 1 - \frac{S_{\text{role}}^2}{S_{\text{role}}^2 + \theta_{\text{role}}^2}. \tag{11}$$

其中: $\Gamma_{\text{role_1}}$为执行任务1的概率, $\Gamma_{\text{role_2}}$为执行任务2的概率.

设计算法参数时,需对两类蚂蚁的初始数量占比进行设定.在算法初期,为了加快信息素的正反馈累加速度,常规蚂蚁个体数量需占绝对优势,以保证算法的收敛速度,因此本文设定常规蚂蚁初始比例为80 %,扩展蚂蚁初始比例为20 %,且$m_2 \leqslant m_1$.

3.3 基于刺激-响应模型的信息素更新机制

为加快收敛,本文引入精英策略,设计一种基于刺激-响应模型的信息素更新机制. 所有个体遵循此信息系更新机制. 信息素更新机制的设计思路如下:

1) 将是否更新当前路径的信息素看作任务;

2) 每个蚂蚁均有各自的信息素更新阈值,该阈值与蚂蚁个体对应的路径长度有关;

3) 外部环境有对应刺激,刺激值与种群的平均适应度函数值(即平均路径)有关;

4)) 蚂蚁个体更新对应路径信息素的概率大小与该蚂蚁的阈值和外部环境刺激值有关,遵循刺激-响应模型.

令S_{ph}表示信息素更新任务的环境刺激,θ_{i_ph}表示个体i的响应阈值,则个体执行任务的概率为

$$\Gamma_{i\_ph} = \frac{S_{ph}^2}{S_{ph}^2 + \theta_{i\_ph}^2}, \tag{12}$$

$$S_{ph} = L_{\text{mean}}, \tag{13}$$

$$\theta_{i\_ph} = L_i. \tag{14}$$

当前路径长度越大时,对应算法解的质量越差,根据上述模型,其信息素更新的概率也越小;当路径长度越小时,对应算法解的质量越好,其信息素更新的概率越大. 因此,在算法运行初始阶段,各可行解之间的质量差距较大,质量高的可行解会以较大概率更新信息素,保证了初始阶段的信息素的快速积累,加快了算法收敛速度.

3.4 算法流程

本文算法(MF-ACO)整体框架如图5所示,包括角色分配、路径构造和信息素更新3部分. 此外,为进一步加强算法的局部搜索能力,算法引入2-opt交换规则[34]帮精英个体进行局部调优,选取路径最短的5 %个体为精英进行2-opt交换.算法具体步骤如下:

step 1: 设置参数包括m、Δm、n、Q、ρ、α、β、N_{MAX}、N_C等;

step 2: 初始化两类蚂蚁个体m_1、m_2占比;

step 3: 常规蚂蚁依据信息素浓度构造m_1个路径(方法与基本蚁群算法ACO一致);

step 4: 根据式(6)和轮盘法,选择已有路径分配给扩展蚂蚁;

step 5: 扩展蚂蚁依据第3.1.2节的组合交换规则构造m_2个新路径;

step 6: 所有个体进行排序,选择路径最短的5 %个体执行2-opt交换;

step 7: 根据式(12)计算每个个体的更新概率,且每个个体生成(0 ~ 1)区间内的随机数,比较随机数和更新概率的大小,对本次迭代满足条件的个体依据式(3)进行信息素更新;

step 8: 根据式(10)、(11)计算种群的角色调整概率,然后生成随机数,比较随机数和更新概率的大小,判断选择执行任务1或任务2;

step 9: 记录最优值,判断是否满足算法终止条件,不满足则返回step 3,否则输出最优结果.

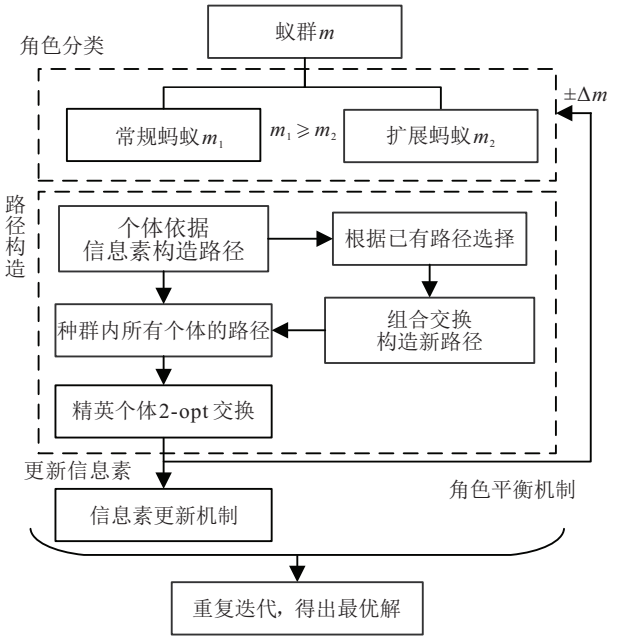


图5 基于混合反馈机制的扩展蚁群算法(MF-ACO)框架

4 仿真实验

为验证本文算法的改进效果,选取标准TSP实例进行仿真实验. 算法基于Matlab 2017实现,运行环境为Windows10; CPU为Intel(R) Core(TM) i7-7500U; 运行内存为8 GB.

4.1 参数设置

由前文已知Δm与负反馈调节作用的大小有关,本节主要讨论Δm对算法性能的影响.选取实例eli76和eil101进行仿真对比. 除Δm外其余参数设置如下: $\alpha = 2, \beta = 5, \rho = 0.6, Q = 100$,迭代次数$N_{\text{MAX}} = 500$,蚂蚁总数$m = 1.5n$($n$为路径节点数),初始常规蚂蚁个数$m_1 = 0.8m$,扩展蚂蚁$m_2 = 0.2m$,扩展蚂蚁组合循环次数$N_C = n$.

根据Δm的大小分成5组,每组进行20次独立实验,结果如表1所示. 已知解为当前所知的最短路径长度,来自于TSPLIB标准库;最优值为20次运算所得最短路径;平均迭代次数N_{ave}为达到最优值时的迭代次数求平均值后取整.

表 1　参数结果对比

实例	已知解	$\Delta m/m$	最优值	平均值	N_{ave}
eli76	538	0.02	547.24	569.07	447
		0.05	544.36	551.72	270
		0.1	546.83	554.21	233
		0.15	547.14	557.45	217
		0.20	555.72	574.67	211
eli101	629	0.02	658.76	667.41	483
		0.05	642.08	653.47	302
		0.1	644.93	654.86	293
		0.15	649.29	661.30	267
		0.20	651.37	665.27	271

由表1结果可知,当Δm取0.05倍的m时,算法效果最好,此时若减小Δm,则迭代次数明显增加,无法保证算法在500次内收敛;而若增大Δm,虽可以略微降低平均迭代次数,但最优值变差. 故算法设定最终$\Delta m = \text{ceil}(0.05m)$,其中ceil(·)为向上取整函数.

4.2　结果对比

4.2.1　与经典算法对比

将本文算法与ACO、MMAS、ACS三种经典蚁群算法进行对比,表2为对比结果. 共选取14个实例进行仿真,3种算法的参数参考文献[4, 23, 27]进行设置,迭代次数为500. 所有算法均独立运行20次.

如表2所示,在所有实例中,MF-ACO所得的最优解、最优偏差、平均值和平均偏差均优于3种经典算法. 此外由于在编程时以浮点数计算结果,所得最优解与已知最优解之间存在很小的偏差.

表 2　本文算法与经典算法对比结果

实例	已知解	算法	算法最优值		算法平均值	
			最优值	偏差/%	平均值	平均偏差/%
att48	33 522	ACO	34 379.70	2.56	35 523.26	5.97
		MMAS	34 643.96	3.35	35 637.24	6.31
		ACS	33 668.14	0.44	34 691.92	3.49
		MF-ACO	33 523.71	0.05	33 659.44	0.41
berlin52	7 542	ACO	7 854.79	4.15	7 986.17	5.89
		MMAS	7 854.79	4.15	7 941.10	5.29
		ACS	7 685.01	1.90	7 913.70	4.93
		MF-ACO	7 544.36	0.03	7 558.03	0.21
ch130	6 110	ACO	6 446.81	5.51	6 640.35	8.68
		MMAS	6 506.71	6.49	6 674.56	9.24
		ACS	6 324.68	3.51	6 447.88	5.53
		MF-ACO	6 110.86	0.01	6 193.86	1.37
eil51	426	ACO	449.12	5.42	465.19	9.20
		MMAS	452.41	6.20	463.66	8.84
		ACS	440.79	3.47	456.03	7.05
		MF-ACO	428.87	0.67	431.11	1.20
eil76	538	ACO	569.71	5.89	583.95	8.54
		MMAS	556.94	3.52	569.26	5.81
		ACS	554.87	3.13	573.72	6.64
		MF-ACO	544.36	1.18	551.72	2.55
eil101	629	ACO	685.85	9.03	698.94	11.12
		MMAS	677.10	7.65	685.67	9.01
		ACS	675.34	7.37	699.26	11.17
		MF-ACO	642.08	2.08	653.47	3.89
gr96	514	ACO	542.77	6.01	551.63	7.74
		MMAS	541.54	5.77	545.33	6.51
		ACS	539.57	5.32	548.66	7.16
		MF-ACO	512.31	0	525.94	2.72
kroA100	21 282	ACO	22 319.01	4.87	22 861.12	7.42
		MMAS	22 624.71	6.31	23 005.84	8.10
		ACS	22 024.38	3.48	22 744.07	6.87
		MF-ACO	21 285.44	0.02	21 763.75	2.26

表2(续)

实例	已知解	算法	算法最优值		算法平均值	
			最优值	偏差/%	平均值	平均偏差/%
lin105	14 379	ACO	14 807.32	2.98	15 355.33	6.79
		MMAS	14 553.67	1.21	15 000.17	4.32
		ACS	14 618.75	1.67	14 938.34	3.89
		MF-ACO	14 382.99	0.03	14 494.55	0.80
st70	675	ACO	701.47	3.92	727.58	7.79
		MMAS	721.75	6.93	747.16	10.69
		ACS	696.93	3.25	709.76	5.15
		MF-ACO	677.10	0.31	686.50	1.70
pr76	108 159	ACO	114 548.57	5.91	115 740.9	7.01
		MMAS	114 507.91	5.87	116 530.5	7.74
		ACS	112 712.53	4.21	114 367.3	5.74
		MF-ACO	108 159.44	0.00	108 664.30	0.47
pr107	44 303	ACO	46 798.14	5.63	47 701.04	7.67
		MMAS	46 682.07	5.37	48 121.92	8.62
		ACS	46 026.39	3.89	47 771.92	7.83
		MF-ACO	44 429.75	0.28	44 532.62	0.52
pr124	59 030	ACO	62 577.70	6.01	63 268.35	7.18
		MMAS	62 672.15	6.17	64 897.58	9.94
		ACS	61 603.71	4.36	62 949.59	6.64
		MF-ACO	59 074.8	0.07	59 349.62	0.54
rat99	1 211	ACO	1 293.86	6.84	1 310.54	8.22
		MMAS	1 275.89	5.36	1 308.01	8.01
		ACS	1 270.56	4.92	1 292.50	6.73
		MF-ACO	1 219.24	0.68	1 231.76	1.71

4.2.2 与改进算法对比

将算法与其他改进算法进行对比,首先与文献[14]提出的面向对象的多角色蚁群(OMACO)算法以及文献[28]提出的基于频率图的蚁群优化(FGACO)算法进行对比,对比结果见表3. 这两种改进算法均重点提高了蚁群算法的全局搜索能力. 根据原始文献,两种算法的参数设置如下:OMACO算法:$m = 1.5n, \alpha = 1, \beta = 5, \rho = 0.9, Q = 120, \varepsilon = 1.5, \xi = 6, N_{\mathrm{MAX}} = 1\,000$;FGACO算法:$m = 1.5n, \alpha = 3, \beta = 2, \rho = 0.05, Q = 100, N_{\mathrm{MAX}} = 1\,000$.

3种算法的对比结果如表3所示. 除lin105中算法的最优值相同之外,其余实例中其结果最优值均优于OMACO和FGACO算法. 可见本文算法较这两种改进算法的精确度和收敛性均有一定提高.

之后将本算法与基于虚拟蚂蚁的局部优化蚁群(VLACO)算法[29]和基于方向信息素协调的蚁群(AADC)算法[30]进行对比. 这两种算法通过改进启发信息来提高收敛速度力和搜索能力. VLACO算法参数为$m = n, \alpha = 2, \beta = 3, \rho = 0.328, Q =$

表3 本文算法与文献[14]和文献[28]算法对比

实例	已知解	算法	最优值	偏差
st70	675	OMACO	678.62	0.54
		FGACO	679.07	0.60
		MF-ACO	677.10	0.31
ch130	6 110	OMACO	6 127.39	0.27
		FGACO	6 256.24	2.39
		MF-ACO	6 110.86	0.01
lin105	14 379	OMACO	14 382.99	0.03
		FGACO	14 394.74	0.11
		MF-ACO	14 382.99	0.03
pr76	108 159	OMACO	108 202.14	0.04
		FGACO	108 700.32	0.50
		MF-ACO	108 159.4	0.00
pr107	44 303	OMACO	44 620.18	0.72
		FGACO	44 694.35	0.88
		MF-ACO	44 429.75	0.28

100; AADC 算法参数为 $m = n, \alpha = 1, \beta = 5, \rho = 0.9, Q = 100$.

对比结果如表 4 所示, MF-ACO 算法所得最优值和平均值最小, 优于文献 [29] 和文献 [30] 的算法. 此外, 在收敛性能上, 本文算法最优值的平均迭代次数 N_{ave} 远小于文献 [29] 算法, 与文献 [30] 相比也有一定提高.

表 4　本文算法与文献 [29] 和文献 [30] 算法对比

实例	算法	最优值	偏差	平均值	N_{ave}
att48 33 522	AADC	—	—	—	—
	VLACO	33 588.34	0.2	33 782.75	113
	MF-ACO	33 523.71	0.05	33 659.44	120
eil51 426	AADC	429.74	0.88	437.20	1078
	VLACO	428.87	0.67	432.71	348
	MF-ACO	428.87	0.67	431.11	184
eil76 538	AADC	555.61	3.27	557.43	970
	VLACO	546.83	1.64	557.07	279
	MF-ACO	544.36	1.18	551.72	270
st70 675	AADC	682.31	1.05	689.92	932
	VLACO	677.10	0.31	696.63	230
	MF-ACO	677.10	0.31	686.50	255
eil101 629	AADC	—	—	—	—
	VLACO	649.57	3.27	658.76	365
	MF-ACO	642.08	2.08	653.47	302

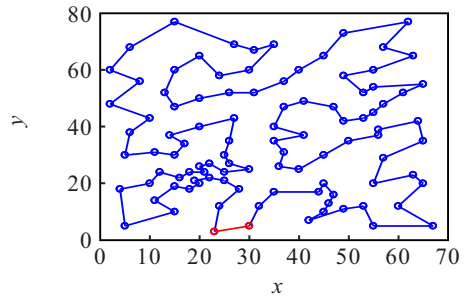
4.3　算法分析

4.3.1　平衡性分析

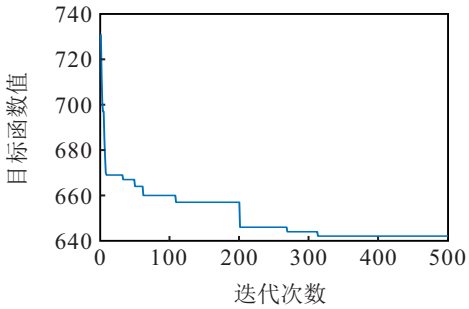
本文算法将种群分为常规蚂蚁和扩展蚂蚁两类, 以平衡机制动态调节两类蚂蚁的数量比例. 其中常规蚂蚁保证了算法的稳定性和收敛性, 而扩展蚂蚁则在算法陷入停滞时, 帮助算法跳出局部极值, 保持种群多样性. 由算法仿真对比可知, 本文提出的 MF-ACO 算法在精确度和收敛性上较以往的蚁群算法及相关改进算法均有较大提高.

图 6 为 MF-ACO 算法运行在 eil101 上得到的运行结果. 其中: 图 6(a) 为最短线路、图 6(b) 为收敛曲线、图 6(c) 为扩展蚂蚁数量变化曲线.

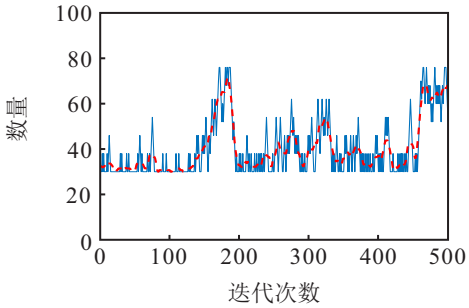
由图 6(b) 收敛曲线可知, 算法迭代 300 次左右达到最优值, 配合扩展蚂蚁数量变化曲线图 6(c) 进行分析: 在算法运行初期, 扩展蚂蚁个体数量处于较低水平, 常规蚂蚁个体数量占绝对优势, 此时从收敛曲线中可以看出, 算法处于快速收敛阶段; 而当算法迭代至 100 次左右时, 收敛速度急剧下降, 陷入局部极值, 此时扩展蚂蚁数量快速上升到达峰值, 种群多样性增加, 帮助算法跳出局部最优, 重新开始收敛; 之后扩展蚂蚁呈现周期性变化, 当种群多样性下降时, 扩展蚂蚁数量增加, 多样性上升后, 扩展蚂蚁数量减少, 这一变化规律符合算法设计原理: 初期常规蚂蚁保证算法的收敛速度, 当常规蚂蚁的正反馈搜索陷入局部极值时, 扩展蚂蚁个体增加, 利用组合交换规则跳出局部极值.



(a)　最短路径 (优化最短距离:642.083 4)



(b)　收敛曲线 (适应度进化)



(c)　扩展蚂蚁数量变化

图 6　MF-ACO 算法运行结果

4.3.2　复杂度分析

传统蚁群算法的复杂度主要与蚂蚁数量 m、路径节点个数 n、迭代次数 N_{MAX} 有关, 可以表示为 $O(N_{\text{MAX}} \cdot m \cdot n^2)$. 在本文算法中, 常规蚂蚁与基本蚁群算法的路径构造策略相同, 信息素更新则需要增加一次分工概率计算和判断, 增加部分的计算复杂度为 $O(N_{\text{MAX}} \cdot m_1)$, 对整体影响不大, 故常规蚂蚁的时间复杂度仍可近似看作 $O(N_{\text{MAX}} \cdot m_1 \cdot n^2)$. 扩展蚂蚁主要有选择和组合操作, 其中选择操作每代执行 m_2 次, 组合操作每代执行 $N_C \cdot m_2$ 次, N_C 等于 n, 故扩展型蚂蚁的时间复杂度为 $O(N_{\text{MAX}} \cdot m_2 + N_{\text{MAX}} \cdot n \cdot m_2)$, 约等于 $O(N_{\text{MAX}} \cdot n \cdot m_2)$. 此外, 算法引入了 2-opt 策略进行局部优化, 但执行个体所占整体的比例极少, 对复杂度的影响很小. 综上所述, MF-ACO 算法整体复

杂度为$O(N_{\mathrm{MAX}}\cdot m_1\cdot n^2+N_{\mathrm{MAX}}\cdot n\cdot m_2)$,与基本蚁群算法$O(N_{\mathrm{MAX}}\cdot m\cdot n^2)$差别不大,故本文算法的相关改进并未增加算法的复杂度.

5 机器人路径规划

为验证MF-ACO算法在实际问题上的有效性,将算法应用于机器人路径规划问题[12]进行求解.

5.1 基于路径规划问题的MF-ACO算法设计

本文MF-ACO算法是基于TSP问题设计的,为了适配于实际路径规划问题,需进行补充和修改的算法相关策略如下.

策略1:增加初始信息素不均衡分配策略.

使用蚁群算法求解路径规划问题时,会面临收敛性差、盲目搜索的问题,因此本文采用初始信息素不均衡分配策略[12],减少蚁群搜索的盲目性. 该策略主要思想为在起始节点与目标节点之间的所有节点,规定其初始信息素略大于其他节点.

策略2:修改扩展蚂蚁组合策略.

在路径规划问题中,两路径需有交叉节点时才可以进行交换操作,故组合策略需修改为如下形式:

1) 每只扩展蚂蚁随机挑选对象进行组合.

2) 挑选完成后,判断被选蚂蚁的路径与自身路径是否有相交节点:若有相交节点则以节点为交换点进行交换;若不相交,则返回重新选取,直至选择NC次后停止.

3) 选择距离最短的一种组合保留,构造新路径.

策略3:删除2-opt交换策略.

在TSP问题中,算法利用2-opt策略对少数精英个体进行局部寻优,但在路径规划问题中,该策略违背路径规划逻辑,因此删除此条策略.

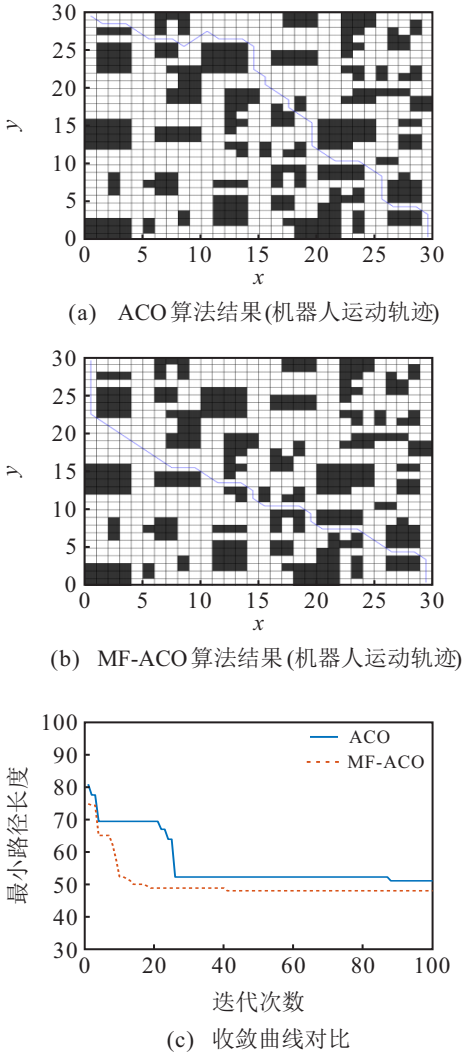
5.2 实验与结果

采用栅格法设计20 × 20、30 × 30地图各一个,先将本文算法与ACO算法对比,两种算法蚂蚁个数为100,迭代次数为100,其余参数与第4节相同,仿真结果如表5所示,30 × 30仿真示意图如图7所示.

表5 本文算法与ACO算法的结果对比

算法	20 × 20地图		30 × 30地图	
	路径长度	迭代次数	路径长度	迭代次数
ACO	32.98	60	51.11	89
MF-ACO	31.55	27	48.04	41

在20 × 20环境下,ACO算法用60次迭代收敛到32.98,本文算法只用了27次迭代便收敛于最优路径31.55. 在30 × 30环境下,ACO算法用89次迭代搜索到最优路径51.11,本文算法用41次迭代便收敛于最优路径48.04. 因此本文提出的MF-ACO算法相比ACO算法,其搜索到的路径更短,收敛速度也更快,在收敛性和全局搜索能力上均得到较大提高,也验证了MF-ACO算法在解决实际问题时的有效性.



(a) ACO算法结果(机器人运动轨迹)

(b) MF-ACO算法结果(机器人运动轨迹)

(c) 收敛曲线对比

图7 30 × 30地图仿真结果对比

扩大对比范围,将本文算法与常用的最短路径A*算法[35]和Dijkstra算法[36]进行对比,算法对比结果如表6所示.

表6 本文算法与最短路径算法的结果对比

算法	20 × 20地图	30 × 30地图
MF-ACO	31.55	48.04
Dijkstra	31.55	48.04
A*	31.55	48.87

由表6可知,20×20地图环境下,MF-ACO算法与A*以及Dijkstra算法均可以找到31.55长度的最短路径. 在30×30地图环境下,与MF-ACO算法和Dijkstra算法相比,A*算法由于加入部分启发式策略,在障碍地图环境下未能找到最短路径. 此外,与A*以及

Dijkstra等全局规划算法相比,MF-ACO算法属于局部路径规划算法,即算法在进行每一步搜索时,仅需对周围的节点信息进行探索,而不需要知道全局信息,可以用来处理动态环境下的路径规划问题.

6 结 论

本文针对传统蚁群算法初期收敛速度慢、易陷入局部最优值的缺点,通过引入刺激-响应分工机制,提出了一种具有混合反馈机制的扩展蚁群算法. 本文算法既包含基于信息素的正反馈式搜索,也包含具有负反馈特性的动态平衡机制,解决了单一信息素正反馈搜索的缺点,保证了算法的全局搜索能力;此外,为进一步加快算法收敛,本文基于刺激-响应模型和精英化思想设计了一种新的信息素更新机制. 所提出的算法在TSP和路径规划问题实例上与其他算法进行了仿真对比,并得到了较好效果,验证了MF-ACO算法的有效性和优越性.

从实验结果看,MF-ACO算法相比于已有算法有一定提高,但仍有不足: 1) 本文引入改进策略的同时也引入了新的参数,在应用中需要调节更多参数,某种程度上增加了参数敏感性; 2) 本文算法属于局部搜索算法,未用到全局信息,在某些实际可以得到全局信息的场景中,若加入全局信息,则求解效率可以得到进一步提高,因此在实际应用时,可将本文算法与全局算法结合使用.

参考文献(References)

[1] Dorigo M, Maniezzo V, Colorni A. Ant system: Optimization by a colony of cooperating agents[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 1996, 26(1): 29-41.

[2] Bonabeau E, Dorigo M, Theraulaz G. Inspiration for optimization from social insect behaviour[J]. Nature, 2000, 406(6791): 39-42.

[3] 汪定伟, 王俊伟, 王洪峰. 智能优化方法[M]. 北京: 高等教育出版社, 2007: 198-205.
(Wang D W, Wang J W, Wang H F. Intelligent optimization methods[M]. Beijing: Higher Education Press, 2007: 198-205.)

[4] Mavrovouniotis M, Müller F M, Yang S X. Ant colony optimization with local search for dynamic traveling salesman problems[J]. IEEE Transactions on Cybernetics, 2017, 47(7): 1743-1756.

[5] Wang P, Lin H T, Wang T S. An improved ant colony system algorithm for solving the IP traceback problem[J]. Information Sciences, 2016, 326: 172-187.

[6] Wu X X, Wei G L, Song Y, et al. Improved ACO-based path planning with rollback and death strategies[J]. Systems Science & Control Engineering, 2018, 6(1): 102-107.

[7] Lai M Y, Tong X J. A metaheuristic method for vehicle routing problem based on improved ant colony optimization and Tabu search[J]. Journal of Industrial & Management Optimization, 2012, 8(2): 469-484.

[8] Et al G Joel Sunny Deol. Hadoop job scheduling using improvised ant colony optimization[J]. Turkish Journal of Computer and Mathematics Education: TURCOMAT, 2021, 12(2): 3417-3424.

[9] 孙凯, 刘祥. 基于蚁群-遗传混合算法的设备布局优化方法[J]. 系统工程理论与实践, 2019, 39(10): 2581-2589.
(Sun K, Liu X. A method of workshop equipment layout based on ant colony——genetic hybrid algorithm[J]. Systems Engineering—Theory & Practice, 2019, 39(10): 2581-2589.)

[10] 覃远年, 梁仲华. 蚁群算法研究与应用的新进展[J]. 计算机工程与科学, 2019, 41(1): 173-184.
(Qin Y N, Liang Z H. New progress of the ant colony algorithm in research and applications[J]. Computer Engineering & Science, 2019, 41(1): 173-184.)

[11] Liu G Q, He D X. An improved Ant colony algorithm based on dynamic weight of pheromone updating[C]. 2013 International Conference on Natural Computation (ICNC). Shenyang, 2013: 496-500.

[12] 王晓燕, 杨乐, 张宇, 等. 基于改进势场蚁群算法的机器人路径规划[J]. 控制与决策, 2018, 33(10): 1775-1781.
(Wang X Y, Yang L, Zhang Y, et al. Robot path planning based on improved ant colony algorithm with potential field heuristic[J]. Control and Decision, 2018, 33(10): 1775-1781.)

[13] Ning J X, Zhang Q, Zhang C S, et al. A best-path-updating information-guided ant colony optimization algorithm[J]. Information Sciences, 2018, 433/434: 142-162.

[14] 杜鹏桢, 唐振民, 孙研. 一种面向对象的多角色蚁群算法及其TSP问题求解[J]. 控制与决策, 2014, 29(10): 1729-1736.
(Du P Z, Tang Z M, Sun Y. An object-oriented multi-role ant colony optimization algorithm for solving TSP problem[J]. Control and Decision, 2014, 29(10): 1729-1736.)

[15] Hinchey M G, Sterritt R, Rouff C. Swarms and swarm intelligence[J]. Computer, 2007, 40(4): 111-113.

[16] Bonabeau E W, Sobkowski A, Theraulaz G, et al. Adaptive task allocation inspired by a model of division of labor in social insects[J]. Bio-Computation and Emergent Computing, 1997(8): 36-45.

[17] Wang Y C, Xiao R B. Labour division in swarm intelligence for allocation problems: A survey[J]. International Journal of Bio-Inspired Computation, 2018, 12(2): 71-86.

[18] Theraulaz G, Bonabeau E, Deneubourg J. Response threshold reinforcements and division of labour in insect societies[J]. The Royal Society B Biological Sciences, 1998, 265(1393): 327-332.

[19] 肖人彬, 王英聪. 面向群体利益分配的蚁群劳动分工建模与仿真[J]. 管理科学学报, 2016, 19(10): 1-15.
(Xiao R B, Wang Y C. Modeling and simulation of ant colony's labor division for interest allocation of social groups[J]. Journal of Management Sciences in China, 2016, 19(10): 1-15.)

[20] Xiao R B, Wu H S, Hu L, et al. A swarm intelligence labour division approach to solving complex area coverage problems of swarm robots[J]. International Journal of Bio-Inspired Computation, 2020, 15(4): 224-238.

[21] 肖人彬, 王英聪. 一种面向时间分配问题的群智能劳动分工新方法[J]. 智能系统学报, 2019, 14(3): 438-448.
(Xiao R B, Wang Y C. A new approach to labor division in swarm intelligence for time allocation problem[J]. CAAI Transactions on Intelligent Systems, 2019, 14(3): 438-448.)

[22] Zahadat P, Schmickl T. Division of labor in a swarm of autonomous underwater robots by improved partitioning social inhibition[J]. Adaptive Behavior, 2016, 24(2): 87-101.

[23] Stützle T, Hoos H H. MAX-MIN ant system[J]. Future Generation Computer Systems, 2000, 16(8): 889-914.

[24] Wang Z Y, Xing H L, Li T R, et al. A modified ant colony optimization algorithm for network coding resource minimization[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(3): 325-342.

[25] Che H L, Wu Z Z, Kang R X, et al. Global path planning for explosion-proof robot based on improved ant colony optimization[C]. 2016 Asia-Pacific Conference on Intelligent Robot Systems (ACIRS). Tokyo , 2016: 36-40.

[26] Bellaachia A, Alathel D. A local pheromone initialization approach for ant colony optimization algorithm[C]. 2014 IEEE International Conference on Progress in Informatics and Computing. Shanghai, 2014: 133-138.

[27] Dorigo M, Gambardella L M. Ant colony system: A cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66.

[28] Wang Y, Wu Y W. Frequency graphs for travelling salesman problem based on ant colony optimization[J]. International Journal of Computational Intelligence and Applications, 2019, 18(3): 1950016.

[29] 李俊, 周虎, 李波. 基于虚拟蚂蚁的局部优化蚁群算法[J]. 控制与决策, 2019, 34(11): 2459-2468.
(Li J, Zhou H, Li B. Local optimization ACO based on virtual ant colony algorithm[J]. Control and Decision, 2019, 34(11): 2459-2468.)

[30] 孟祥萍, 片兆宇, 沈中玉, 等. 基于方向信息素协调的蚁群算法[J]. 控制与决策, 2013, 28(5): 782-786.
(Meng X P, Pian Z Y, Shen Z Y, et al. Ant algorithm based on direction-coordinating[J]. Control and Decision, 2013, 28(5): 782-786.)

[31] 肖人彬, 王英聪. 群智能自组织劳动分工研究进展[J]. 信息与控制, 2019, 48(2): 129-139.
(Xiao R B, Wang Y C. Research progress of self-organized labor division in swarm intelligence[J]. Information and Control, 2019, 48(2): 129-139.)

[32] Wang C L, Shi H Y, Zuo X Q. A multi-objective genetic algorithm based approach for dynamical bus vehicles scheduling under traffic congestion[J]. Swarm and Evolutionary Computation, 2020, 54: 100667.

[33] Ursem R K. Diversity-guided evolutionary algorithms[C]. Parallel Problem Solving from Nature—PPSN VII. Berlin: Springer Berlin Heidelberg, 2002: 462-471.

[34] Lin S, Kernighan B W. An effective heuristic algorithm for the traveling-salesman problem[J]. Operations Research, 1973, 21(2): 498-516.

[35] Fu B, Chen L, Zhou Y T, et al. An improved A* algorithm for the industrial robot path planning with high success rate and short length[J]. Robotics and Autonomous Systems, 2018, 106: 26-37.

[36] Enayattabar M, Ebrahimnejad A, Motameni H. Dijkstra algorithm for shortest path problem under interval-valued Pythagorean fuzzy environment[J]. Complex & Intelligent Systems, 2019, 5(2): 93-100.

作者简介

冯振辉(1992−), 男, 博士生, 从事群智能优化、劳动分工的研究, E-mail: feng_zh@hust.edu.cn;

肖人彬(1965−), 男, 教授, 博士生导师, 从事群智能、涌现计算、复杂系统建模与仿真等研究, E-mail: rbxiao@hust.edu.cn.

(责任编辑: 闫 妍)