第 27 卷第 9 期
2010 年 9 月

控 制 理 论 与 应 用
Control Theory & Applications

Vol. 27 No. 9
Sep. 2010

文章编号: 1000−8152(2010)09−1201−06

多优解更新信息素的混合行为蚁群算法

任志刚, 冯祖仁, 张兆军

(西安交通大学 系统工程研究所 制造系统工程国家重点实验室, 陕西 西安 710049)

摘要: 蚁群算法在优化领域, 尤其在组合优化问题中获得了较为成功的应用, 然而它存在易于早熟收敛、搜索时间长等不足. 针对该问题, 提出了一种改进算法. 该算法一方面在典型的状态转移规则中融合了一种随机选择策略, 保证算法始终具有一定的探索能力; 另一方面在搜索过程中保持一个优解池, 通过交替使用池中最优解和其它次优解更新信息素, 达到平衡算法强化搜索和分散搜索的目的. 文中讨论了相关参数的选取方法, 分析了所提算法的计算复杂度和收敛性, 并针对典型的旅行商问题进行了仿真实验, 结果表明该算法获得的解质量高于其他已有算法.

关键词: 蚁群算法; 早熟收敛; 状态转移规则

中图分类号: TP18　　**文献标识码:** A

Hybrid-behavior ant-colony optimization algorithm with pheromone updated by multiple good solutions

REN Zhi-gang, FENG Zu-ren, ZHANG Zhao-jun

(Systems Engineering Institute, State Key Laboratory for Manufacturing Systems Engineering, Xi'an Jiaotong University, Xi'an Shaanxi 710049, China)

Abstract: Ant-colony optimization algorithm(ACO) has been successfully applied to the optimization field, especially to combinatorial optimization problems. However, it may encounter premature convergence or costs an excessively long computation-time. To overcome these shortcomings, we present an improved ACO algorithm. This algorithm incorporates a random selection-strategy into the typical state transition rule, for ensuring its exploration ability. Meanwhile, the algorithm maintains a good-solution pool and alternately uses the optimal solution or the sub-optimal solution from the pool to update the pheromone. Thus, the intensification and the diversification of search are balanced. We also discuss the settings of related parameters, and analyze the computational complexity and convergence of the proposed algorithm. Additionally, simulation experiment is performed on typical traveling salesman problems. The results demonstrate that this algorithm generates higher quality solutions than existing algorithms.

Key words: ant-colony optimization algorithm; premature convergence; state transition rule

1 引言(Introduction)

蚁群算法(ACO)是一种基于群体的启发式仿生进化算法, 它是意大利学者Dorigo等人受自然界中真实蚁群觅食行为的启发而提出的[1]. 该算法具有分布式计算、易于与其他算法结合、鲁棒性强等优点, 已成功应用于旅行商(traveling salesman problem, TSP)[1~3]、二次分配[3]等复杂问题. 然而它存在易于早熟收敛、搜索时间长等不足[2, 3], 限制了其进一步的发展和应用. 为改善算法性能, 自首个蚁群算法——蚂蚁系统[1]提出以来, 国内外已有很多文献对其进行了改进研究, 主要分为以下3类方法:

1) 对算法模型进行改进. 典型的改进算法包括蚁群系统[2]、最大最小蚂蚁系统(max-min ant system, MMAS)[3]以及基于等级的蚂蚁系统[4]. 此外, 文献[5]为蚁群算法提出了一种超立方体框架, 进一步提高了算法的鲁棒性. 最近, 文献[6]提出了一种新型蚁群算法, 除了信息素模型, 该算法新引入了一种具有增强学习功能的构件. 在国内, 文献[7]提出了一种基于分布均匀度的自适应蚁群算法, 该算法根据优化过程中解的分布情况, 调整路径选择策略和信息素更新策略; 文献[8]提出了一种信息素扩散模型, 构建了不同于其他算法的蚁群协作方式; 文献[9]提出了有限级蚁群算法, 采用有限个级别表示信息素, 并证明了该算法具有线性收敛性.

2) 自适应调整算法参数. 该方法的主要思想是通过自适应地调整参数, 改善算法的全局搜索能力

收稿日期: 2009−01−12; 收修改稿日期: 2009−10−31.
基金项目: 国家自然科学基金资助项目(60875043); 国家重点基础研究发展计划(“973”计划)资助项目(2007CB311006).

和收敛速度. 其中, 文献[10] 通过自适应地修改信息素挥发系数, 提高了算法性能; 文献[11] 根据寻优状态, 对算法参数和选择策略进行分阶段设计, 并证明了算法的全局收敛性.

3) 与其他算法进行融合. 该方法希望通过融合不同算法的思想或具体策略, 实现优势互补. 其中, 文献[12] 受神经网络和遗传算法的启发, 提出了一种二进制蚁群进化算法, 取得了良好的求解结果; 文献[13] 将分散搜索的思想融入蚁群算法, 在搜索过程中考虑解质量的同时, 兼顾考虑解的分散性, 提高了算法的全局搜索能力.

文献[14, 15] 对蚁群算法的相关理论和应用做了较为全面的综述. 综合来看, 现有算法主要采用当前迭代中获得的所有解或最优解更新信息素. 实际上, 蚁群算法在运行过程中会产生大量质量略低于最优解的次优解, 其邻域内可能含有质量更高的解, 可以为蚁群搜索提供较好的导向信息. 本文将探讨采用这些优解更新信息素的可行性. 此外, 为保障算法的探索能力, 在典型的状态转移规则中融合了一种随机选择策略, 提出了一种多优解更新信息素的混合行为蚁群算法, 理论分析了算法的复杂度和收敛性, 并通过仿真实验证明了其有效性.

2 蚁群算法的基本原理(Basic principles of ACO)

2.1 过程描述(Procedure description)

蚁群算法是一种并行构造型迭代式搜索算法, 本文以TSP为例说明其工作原理. TSP可以用完全图$G=(N,L)$表示, 其中N表示城市集合, L表示由连接任意两个城市i,j的弧段l_{ij}组成的集合. 对于$\forall l_{ij}\in L, d_{ij}$表示城市$i,j$间的距离.

蚁群算法使用信息素记录积累的经验信息. 当求解TSP时, $\tau_{ij}(t)$表示第t次迭代中弧段l_{ij}上的信息素值. 在每次迭代中, 各蚂蚁从某一随机选择的城市出发, 然后按照式(1)所示的状态转移规则, 不断地在尚未经历的城市间迁移, 直到获得一个完整路径.

$$P(s_{\text{next}}=j|i)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\eta_{ij}^{\beta}}{\sum\limits_{l\in A_{\text{cur}}}[\tau_{il}(t)]^{\alpha}\eta_{il}^{\beta}}, & \text{若}j\in A_{\text{cur}};\\ 0, & \text{其它}.\end{cases}\tag{1}$$

其中: $\eta_{ij}=1/d_{ij}$, 表示l_{ij}上的启发信息; α, β是两个正参数, 决定了信息素和启发信息的相对重要程度; A_{cur}表示蚂蚁尚未经历的城市集合.

在每次迭代中, 当各蚂蚁都获得一个完整路径后, 按照下式所示规则更新信息素:

$$\tau_{ij}(t+1)=\rho\tau_{ij}(t)+\Delta\tau_{ij}(t).\tag{2}$$

其中: $\rho(0\leqslant\rho<1)$表示信息素留存系数, $\Delta\tau_{ij}(t)$表示l_{ij}上的信息素增量. 令S_{ud}表示拟用以更新信息素的解, 那么$\Delta\tau_{ij}(t)$通常可以定义为:

$$\Delta\tau_{ij}(t)=\begin{cases}1/f(S_{\text{ud}}), & \text{若}l_{ij}\text{出现在}S_{\text{ud}}\text{中};\\ 0, & \text{其它}.\end{cases}\tag{3}$$

其中$f(S_{\text{ud}})$表示解S_{ud}对应的路径长度.

2.2 存在的问题(Existing problems)

现有研究表明, 采用到当前迭代为止获得的最优解(简记为S_{gd})或当前迭代中获得的最优解(简记为S_{id})更新信息素, 能够使得算法迅速集中在某一解区域内搜索, 快速找到一优解[2,3]. 然而由于蚁群算法的正反馈作用, 该方法依然可能导致算法搜索空间过小, 甚至陷入停滞状态. 因此, 如何使得算法在强化搜索优解邻域的同时, 保持适度的探索能力, 是挖掘蚁群算法潜能, 提高其性能的关键.

基于该思想, MMAS通过将信息素值限定在区间$[\tau_{\min},\tau_{\max}]$内, 一定程度上防止了停滞现象的发生, 但如何设置$\tau_{\min}$这一关键参数存在一定困难. 文献[3]通过给定重构S_{gd}的概率p_{gd}从而确定$\tau_{\min}$值, 然而在具体计算中忽略了启发信息对算法性能的影响, 并对候选城市个数做了近似处理; 另一方面, 实验表明单纯使用S_{gd} 或S_{id}更新信息素时, MMAS性能提高有限, 而采用S_{gd}和S_{id}混合更新策略以及信息素重新初始化机制时, MMAS性能卓越, 但如何合理调度S_{gd}和S_{id}, 以及确定重新初始化信息素的时机, 尚缺乏系统性规则.

3 多优解更新信息素的混合行为蚁群算法(Hybrid behavior ACO with pheromone updated by multiple good solutions)

3.1 混合行为描述(Description of hybrid behavior)

本文将蚂蚁在解空间内构造解的方式定义为一种搜索行为. 为获得一全局优解, 可能需要选择一些信息素值或局部启发信息值较小的节点. 为达到该目的, 本文算法在典型状态转移规则的基础上, 引入了一种随机选择行为, 保障算法始终具有一定的探索能力. 具体地, 按照下式所示规则构造解:

$$s_{\text{next}}=\begin{cases}\text{从}A_{\text{cur}}\text{中随机选择一城市}, & \text{若}q<q_0;\\ \text{按照式(1)选择一城市}, & \text{其它}.\end{cases}\tag{4}$$

其中: q为均匀分布于$[0,1]$内的随机数, $q_0(0<q_0<1)$是一参数. 按照该规则, 蚂蚁在构造解的每一步均以概率q_0从当前候选城市集合A_{cur}中随机选择一城市, 通过改变q_0可以调整算法探索新解的能力. 为防止搜索过度分散, q_0的取值较小. 下面具体讨论如何确定其取值.

定义事件$A_k(k=2,3,\cdots,|N|;|N|$表示N中的城市个数)为: 蚂蚁第k步选择的城市与前一步选择的城市构成的弧段出现在解$S_{\rm gd}$中. 令$p=1-q_0$, 考虑极限情况下, 信息素均集中在解$S_{\rm gd}$对应的弧段上, 其它弧段上的信息素值接近0, 那么

$$P(A_2)=p+\frac{2q_0}{|N|-1}, \tag{5}$$

$$P(A_k|A_{k-1}\ \cdots\ A_2)=p+\frac{q_0}{|N|-k+1},\ \text{若}k\geqslant 3, \tag{6}$$

$$\begin{aligned}&p_{\rm gd}=P(A_2\ A_3\ \cdots\ A_{|N|})=\\&P(A_2)P(A_3|A_2)\cdots P(A_{|N|}|A_{|N|-1}\ \cdots\ A_2)=\\&(p+\frac{2q_0}{|N|-1})\prod_{k=1}^{|N|-2}(p+\frac{q_0}{k})\approx\\&p^{|N|-1}+q_0p^{|N|-2}\sum_{k=1}^{|N|-2}\frac{1}{k}\approx\\&p^{|N|-1}+(\ln(|N|-2)+C)q_0p^{|N|-2}\approx\\&p^{|N|-1}=(1-q_0)^{|N|-1}.\end{aligned} \tag{7}$$

式(5)中右侧两项分别表示按照式(1)选择下一城市和完全随机选择城市, 使事件A_2成立的概率; 第2项分子中的“2”表示, 对于任意初始城市, 均有两个城市对应弧段的信息素最高. 由式(5)和(6), 可以进一步得到蚂蚁重构$S_{\rm gd}$的概率$p_{\rm gd}$, 具体如式(7)所示. q_0通常为一较小数(一般保持$q_0<0.005$), 因此式(7)中第3步到第4步的运算, 只保留了q_0的一阶项; 第5步中C为欧拉常数, 对于常规TSP($|N|<10000$), $(\ln(|N|-2)+C)q_0\ll p$成立, 故第2项可以忽略掉. 根据式(7), 对于给定的$p_{\rm gd}$, 可以按照下式获得q_0:

$$q_0=1-\sqrt[|N|-1]{p_{\rm gd}}. \tag{8}$$

由于$p_{\rm gd}$本身是一未知量, 所以无论是文献[3]中通过$p_{\rm gd}$求$\tau_{\min}$, 还是本文通过$p_{\rm gd}$求q_0, 似乎意义都不大. 实际上, $\tau_{\min}$, q_0的取值较敏感, 对于不同规模的问题难以确定其取值; 相反地, $p_{\rm gd}$的取值鲁棒性较好, 当它在一定范围内变化时, 算法性能变化不大.

3.2 信息素更新策略(Pheromone update strategy)

本文算法构建了一优解池(good solution pool, GSP), 将算法在运行过程中获得的最优解和目标函数值与最优解差异不大的次优解添加到该池中, 并拟采用这些优解更新信息素. 令$D(S_i,S_j)$表示任意两个解S_i和S_j之间的距离. 对于TSP, $D(S_i,S_j)$等于$|N|$减去S_i和S_j中相同的弧段个数. 给定算法到当前迭代为止获得的最优解$S_{\rm gd}$和一正参数ε, 对于任一新构造的解$S_{\rm n}$, 当且仅当满足下式所示两个条件时, 才将其添加到GSP中.

$$\begin{cases}a)\ f(S_{\rm n})\leqslant(1+\varepsilon)f(S_{\rm gd});\\ b)\ \forall S\in\text{GSP}, \text{若}f(S_{\rm n})=f(S), D(S_{\rm n},S)>0.\end{cases} \tag{9}$$

参数ε指定了GSP中允许次优解偏离最优解的最大程度. 特殊情况下, 若$\varepsilon=0$, 那么GSP中只保存$S_{\rm gd}$. 式(9)中条件b)保证了GSP中不包含重复解. 值得注意的是, 当$S_{\rm gd}$更新后, 需要剔除GSP中不满足条件a)的解.

具体到信息素更新策略, 有以下两种可选方式:

1) 每次迭代中, 选取GSP中全部或部分解更新信息素;

2) 每次迭代中, 从GSP中只选取一个解更新信息素.

若采用方式(1)更新信息素, 当GSP规模较大时, 操作较为复杂, 而且被选中优解的公共弧段有可能被过分加强, 导致算法搜索空间限制在较小范围内. 基于以上原因, 主要考虑采用方式2)更新信息素, 具体遵循以下3个原则:

1) 从GSP中选定一优解$S_{\rm ud}$后, 持续采用该解更新信息素, 直到在连续$T_{\rm ud}$次迭代中GSP均没有更新, 才重新选择另一优解更新信息素.

2) 交替使用GSP中最优解和其它次优解更新信息素. 即若当前$S_{\rm ud}=S_{\rm gd}$, 根据原则1), 当$S_{\rm ud}$需要更新时, 赋予它一次优解.

3) 当需要用次优解更新$S_{\rm ud}$时, 从GSP中所有尚未被用以更新信息素的次优解中随机选择一个解. 若所有次优解都已经使用过, 则将它们重新标识为“尚未使用”.

原则1)的目的是持续强化在选定优解的邻域内进行搜索. 若在连续若干次迭代中GSP均没有更新, 那么当前$S_{\rm ud}$很可能为局部优解, 此时重新选择一优解更新$S_{\rm ud}$, 将蚁群引导到一新区域内进行搜索. $T_{\rm ud}$的取值较为鲁棒, 一般设定其取值范围为$10\sim 20$. 实际上, 当单纯采用$S_{\rm gd}$更新信息素时, 通常也需要连续若干次迭代才能发现更优解. 该方法的缺点是算法一旦陷入局部优解, 缺乏切换到另一区域进行搜索的机制. 原则2)的目的是突出GSP中最优解$S_{\rm gd}$的作用. 与GSP中其它次优解相比, $S_{\rm gd}$代表着最有潜力的搜索方向. 假若将$S_{\rm gd}$和其它次优解同等处理, 当允许的迭代次数较少时, 新获取的$S_{\rm gd}$可能没有机会更新信息素. 原则3) 的主要目的是当$S_{\rm ud}$需要更新时, 避免重复选择某一次优解, 尽可能扩大蚁群的搜索空间.

保持搜索强化性和分散性之间的合理平衡是提高蚁群算法性能的关键. 然而在某一时刻, 很难确定

一个平衡点同时满足算法强化搜索和分散搜索的需求. 本文算法在连续若干次迭代中采用同一优解更新信息素, 实际上是在该解邻域内进行精搜索的过程, 由此满足算法强化搜索的需求; 算法提供的交替使用不同优解更新信息素的方式, 使得搜索能够在不同优解邻域内切换, 满足算法分散搜索的需求. 此外, 搜索时间长一直是制约蚁群算法在实际中应用的瓶颈, 而并行执行是解决该问题的有效途径之一[16], 典型的并行策略是在单次迭代中分别使用一个处理器执行单只蚂蚁的搜索行为. 在本文算法中, 对于每个选定的优解$S_{\rm ud}$, 至少会在$T_{\rm ud}$次迭代中连续使用, 那么可以将不同迭代中需要的信息素矩阵预先计算好, 并且在$T_{\rm ud}$组处理器中并行执行原来需要$T_{\rm ud}$次迭代才能完成的搜索任务, 从而缩短算法的运行时间.

3.3 算法步骤及复杂度分析(Algorithm procedure and complexity analysis)

本文算法遵循蚁群算法的一般步骤, 但在解构造和信息素更新阶段变化较大. 图1给出了算法的流程图, 其具体步骤如下:

Step 1 初始化参数α, β, ρ, ε, $T_{\rm ud}$, $p_{\rm gd}$, $N_{\rm a}$(蚂蚁个数), 以及信息素矩阵;

Step 2 对每只蚂蚁进行如下操作: 随机选择一初始节点, 然后按照式(4)(1)所示规则选择下一节点, 直到获得一完整解;

Step 3 统计到当前迭代为止获得的最优解$S_{\rm gd}$, 根据式(9), 删除GSP中不满足条件的解, 并将当前迭代中获得的满足条件的解添加到GSP中;

Step 4 根据3.2节原则(1)判断$S_{\rm ud}$是否需要更新, 若需要, 则根据原则2)和3)从GSP中重新选择一优解更新$S_{\rm ud}$, 否则, 转下一步;

Step 5 根据式(2)(3)更新信息素矩阵;

Step 6 判断是否满足终止条件, 若满足, 则退出, 否则, 转Step 2.

记N_C为允许的最大迭代次数. Step 1, 2, 5的时间复杂度分别为$O(|N|^2)$, $O(|N|^2N_{\rm a})$, $O(|N|^2)$[1]. 记K为GSP中解的个数, Step 3中统计$S_{\rm gd}$, 删除GSP中不满足条件的解, 以及添加新解的时间复杂度分别为$O(N_{\rm a})$, $O(K)$, $O(N_{\rm a})$; Step 4的最大时间复杂度为$O(K)$. 通常情况下, $K \ll |N|^2N_{\rm a}$, 那么总体上看, 本文算法和蚂蚁系统具有相同的时间复杂度, 为$O(|N|^2N_{\rm a}N_C)$[1].

3.4 收敛性分析(Convergence analysis)

收敛性分析是随机优化算法理论研究的重要内容之一. 对于本文算法, 本节给出如下定理:

定理 1 令$P^*(t)$表示算法在前t次迭代中至少一次找到最优解的概率. 那么对于一个任意小的正数ζ, 存在一个充分大的数t使以下不等式成立:

$$P^*(t) \geqslant 1-\zeta, \text{ 且 } \lim_{t\to\infty} P^*(t) = 1.$$

证 根据状态转移规则(4)和(1), 任意时刻, 任一可行弧段被选择到的概率具有一个平凡下界$p_{\min} = q_0/|N| > 0$, 那么获得任一可行解(包括最优解)的概率P满足: $P \geqslant p_{\min}^{|N|-1} > 0$. 进一步, 算法在前$t$次迭代中至少一次找到一最优解的概率$P^*(t)$满足: $P^*(t) \geqslant 1-(1-P)^t$. 通过选择一个足够大的$t$, 可以获得比任意$1-\zeta$都要大的$P^*(t)$, 因此可得$\lim\limits_{t\to\infty} P^*(t) = 1$. 证毕.

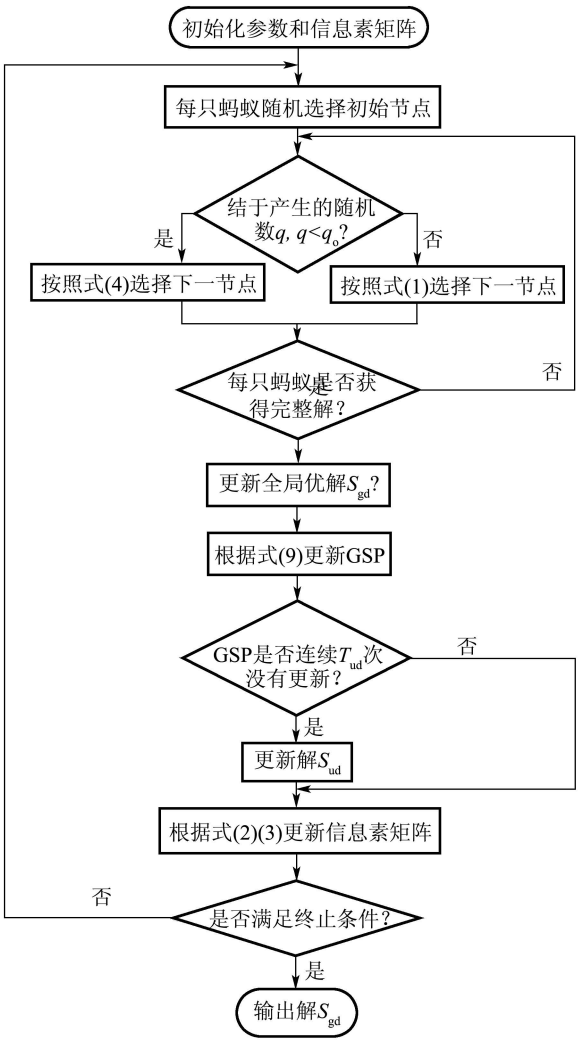


图 1 算法流程图

Fig. 1 Flow chart of the algorithm

4 实验与分析(Experiment and analysis)

为考察本文算法的有效性, 用VC++6.0实现了MMAS和本文算法, 运行环境为PIV 2.0 GHz CPU,

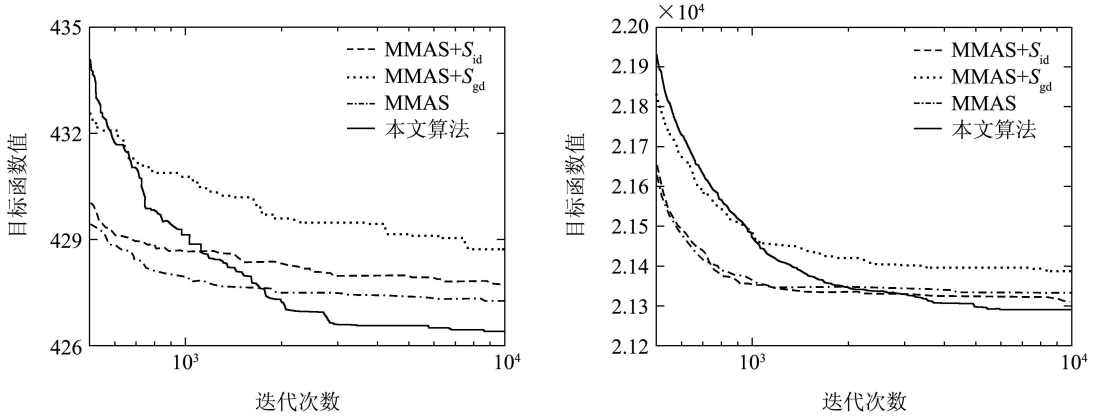
1 GB RAM, Windows XP操作系统. 测试算例均选自TSPLIB, 算例名称中的数字表示城市数目. 为保证比较的公平性, α, β, ρ, N_a与文献[3]中MMAS取相同的值, 依次为1.0, 2.0, 0.98, $|N|$, 允许的最大迭代次数为10000; MMAS中没有涉及到的参数取值为$p_{gd} = 0.80$, $\varepsilon = 0.005$, $T_{ud} = 10$. 实验中对每个算例均独立测试25次, 实验结果如表1所示. 表中MMAS+S_{id}和MMAS+S_{gd}分别表示采用S_{id}和S_{gd}更新信息素的MMAS算法, 在标准MMAS中每隔10次迭代采用S_{gd}更新一次信息素, 在其它迭代中均采用S_{id}更新信息素; 表中后两项分别表示各算法首次获得其最终优解所需要的平均时间和平均迭代次数. 值得指出的是, 本实验的主要目的是考察比较各算法本身的寻优能力, 因此没有使用局部搜索改进算法性能.

从表1中可以看出, 在25次测试中, MMAS+S_{id}获得的平均解质量优于MMAS+S_{gd}, 而标准MMAS性能最高, 这与文献[3]中的结论是一致的. 与上述三种算法相比, 无论是最优解、最差解, 还是平均解, 本文算法均显示出一定优势. 就标准差而言, 对于算例d198, 本文算法的标准差值略大于MMAS+S_{id}, 但小于其他两种算法; 而在其余两个算例中, 其标准差性能均高于另外3种算法, 由此说明了本文算法性能较为稳定. 同时可以注意到, 虽然在多次测试中, MMAS+S_{gd}获得的平均解质量较低, 标准差较大, 但是它所获得的最优解质量不低于其他两种MMAS算法. 这说明S_{gd}可以为蚁群搜索提供较好的导向信息, 然而MMAS+S_{gd}的性能对初始时刻搜索质量的依赖性较大, 算法稳定性差. 本文算法在突出S_{gd}导向作用的同时, 能够始终保持一定的探索能力, 提高了算法的搜索性能和稳定性. 正是由于这种探索性, 与其他3种算法相比, 本文算法通常需要较长的搜索时间, 才能获得其最终优解. 图2给出了4种算法针对3个算例的搜索进化曲线. 从图中可以看出, MMAS+S_{gd}对应的曲线中“水平线段”较多且偏长, 说明该算法获得的优解长时间不能更新, 算法的探索能力偏弱; MMAS+S_{id}和标准MMAS对应的曲线较为相近, 且始终处于MMAS+S_{gd}对应曲线的下方; 在整个进化过程中, 本文算法对应曲线的“台阶”较多, 说明算法保持了较强的探索能力. 虽然在搜索前期, 算法获得的解质量较低, 然而在搜索的中后期, 其获得的解质量均超过了其他3种算法.

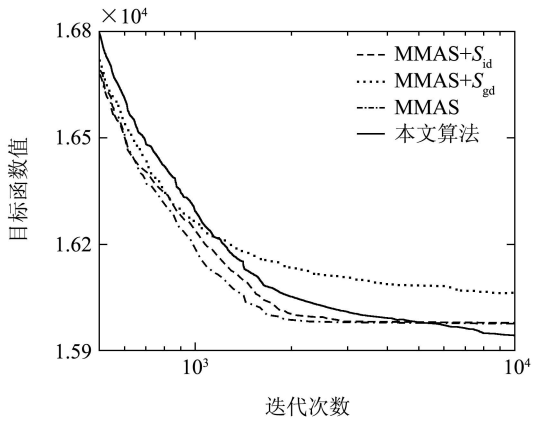
表 1　实验结果比较

Table 1　Comparison of experimental results

算例(已知最优解)	算法	最优解	最差解	平均解	标准差	首遇时间/s	首遇迭代次数
eil51 (426)	MMAS+S_{id}	427	431	427.72	0.8426	4.8162	2257.3
	MMAS+S_{gd}	426	435	428.72	2.7917	5.2100	2450.8
	MMAS	426	428	427.28	0.7371	3.9656	1854.2
	本文算法	426	427	426.40	0.5000	4.4068	2053.3
kroA100 (21282)	MMAS+S_{id}	21282	21389	21310.00	44.9347	22.0118	2546.9
	MMAS+S_{gd}	21282	21967	21389.00	158.0291	18.6932	2162.2
	MMAS	21282	21379	21333.00	48.6911	11.6025	1335.4
	本文算法	21282	21305	21290.00	11.2677	26.1382	2985.4
d198 (15780)	MMAS+S_{id}	15934	16007	15975.00	16.1426	132.4200	3782.8
	MMAS+S_{gd}	15876	16320	16061.00	104.4860	197.7313	5647.8
	MMAS	15923	16028	15973.00	24.1666	116.3348	3323.6
	本文算法	15853	16005	15944.00	21.9551	205.3364	5836.2



(a) ei151　　(b) kroA100



(c) d198

图 2 4种算法的进化曲线

Fig. 2 Evolution curves of four algorithms

5 结论(Conclusions)

本文从状态转移规则和信息素更新规则两方面, 对蚁群算法提出了改进策略. 在典型的状态转移规则中融合了一种随机选择策略, 使得蚁群能够始终保持一定的探索能力; 通过交替使用优解池中不同优解更新信息素, 即强化了蚁群在优解邻域内的搜索, 又使得算法能够在不同解区域间切换搜索, 防止陷入局部优解. 最后通过仿真实验验证了该算法的有效性. 本文算法虽然是以TSP为例进行描述的, 但在设计具体策略时, 并没有使用TSP的特有信息, 因此具有一定的适用性.

参考文献(References):

[1] DORIGO M, MANIEZZO V, COLORNI A. The ant system: optimization by a colony of cooperating agents[J]. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 1996, 26(1): 29 – 41.

[2] 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.

[3] STUTZLE T, HOOS H H. Max-min ant system[J]. *Future Generation Computer Systems*, 2000, 16(9): 889 – 914.

[4] BULLNHEIMER B, HARTL R F, STRAUSS C. A new rank-based version of the ant system: a computational study[J]. *Central European Journal for Operations Research and Economics*, 1999, 7(1): 25 – 38.

[5] BLUM C, DORIGO M. The hyper-cube framework for ant colony optimization[J]. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 2004, 34(2): 1161 – 1172.

[6] BORKAR V S, DAS D. A novel ACO algorithm for optimization via reinforcement and initial bias[J]. *Swarm Intelligence*, 2009, 3(1): 3 – 34.

[7] 陈峻, 沈洁, 秦玲, 等. 基于分布均匀度的自适应蚁群算法[J]. 软件学报, 2003, 14(8): 1379 – 1387.
(CHEN Ling, SHEN Jie, QIN Ling, et al. An adaptive ant colony algorithm based on equilibrium of distribution[J]. *Journal of Software*, 2003, 14(8): 1379 – 1387.)

[8] 黄国锐, 曹先彬, 王煦法. 基于信息素扩散的蚁群算法[J]. 电子学报, 2004, 32(5): 865 – 868.
(HUANG Guorui, CAO Xianbin, WANG Xufa. An ant colony optimization algorithm based on pheromone diffusion[J]. *Acta Electronica Sinica*, 2004, 32(5): 865 – 868.)

[9] 柯良军, 冯祖仁, 冯远静. 有限级信息素蚁群算法[J]. 自动化学报, 2006, 32(2): 296 – 303.
(KE Liangjun, FENG Zuren, FENG Yuanjing. Ant colony optimization algorithm with finite grade pheromone[J]. *Acta Automatica Sinica*, 2006, 32(2): 296 – 303.)

[10] 冯远静, 冯祖仁, 彭勤科. 一类自适应蚁群算法及其收敛性分析[J]. 控制理论与应用, 2005, 22(5): 713 – 717.
(FENG Yuanjing, FENG Zuren, PENG Qinke. Adaptive ant colony optimization algorithms and its convergence[J]. *Control Theory & Applications*, 2005, 22(5): 713 – 717.)

[11] 邢桂华, 于盛林. 动态分阶段蚁群算法及其收敛性分析[J]. 控制与决策, 2007, 22(6): 685 – 688.
(XING Guihua, YU Shenglin. Dynamic stage ant colony algorithm and its convergence[J]. *Control and Decision*, 2007, 22(6): 685 – 688.)

[12] 熊伟清, 魏平. 二进制蚁群进化算法[J]. 自动化学报, 2007, 33(3): 259 – 264.
(XIONG Weiqing, WEI Ping. Binary ant colony evolutionary algorithm[J]. *Acta Automatica Sinica*, 2007, 33(3): 259 – 264.)

[13] 张晓霞, 唐立新. 一种求解TSP问题的ACO & SS 算法设计[J]. 控制与决策, 2008, 23(7): 762 – 766.
(ZHANG Xiaoxia, TANG Lixin. An ACO & SS algorithm for traveling salesman problem[J]. *Control and Decision*, 2008, 23(7): 762 – 766.)

[14] DORIGO M, BIRATTARI M, STUTZLE T. Ant colony optimization: artificial ants as a computational intelligence technique[J]. *IEEE Computational Intelligence Magazine*, 2006, 1(4): 28 – 39.

[15] 段海滨. 蚁群算法理论及应用[M]. 北京: 科学出版社, 2005.
(DUAN Haibin. *Ant Colony Algorithms: Theory and Applications*[M]. Beijing: Science Press, 2005.)

[16] TALBI E G, ROUX O, FONLUPT C, et al. Parallel ant colonies for combinatorial optimization problems[J]. *Lecture Notes in Computer Science*, 1999, 1586(1): 239 – 247.

作者简介:

任志刚 (1982—), 男, 博士研究生, 主要从事智能优化算法、数据挖掘等方面的研究, E-mail: whrzg5258@163.com;

冯祖仁 (1953—), 男, 博士研究生, 教授, 博士生导师, 研究领域包括机器人控制、机器人视觉导航、智能信息处理等;

张兆军 (1981—), 男, 博士生, 主要从事智能优化算法、分布式计算等方面的研究.