计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2024,33(3):158−169 [doi: 10.15888/j.cnki.csa.009418]

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

融合强化学习的多目标路径规划①

周　毅, 刘　俊

(武汉科技大学 信息科学与工程学院, 武汉 430081)
通信作者: 周　毅, E-mail: zhouyi83@wust.edu.cn

摘　要: 移动机器人路径规划问题的节点数量大、搜索空间广, 且对安全性和实时性有要求等因素, 针对移动机器人多目标路径规划问题, 提出一种新颖的融合强化学习的多目标智能优化算法. 首先, 该算法采用 NSGA-II 为基础框架, 利用强化学习的赋予个体学习能力, 设计一种 SARSA 算子提高算法的全局搜索效率. 其次, 为了加速算法的收敛速度和保证种群多样性, 增加自适应模拟二进制交叉算子 (tanh-SBX) 作为辅助算子, 并将种群分为两种性质不同的子种群: 精英种群和非精英种群. 最后, 设计了 4 种不同的策略, 通过模拟退火算法的 Metropolis 准则计算更新策略的概率, 让最合适的策略引导种群的优化方向, 以平衡探索和利用. 仿真实验表明, 该算法在不同复杂度的环境下均能找到最佳路径. 相比传统智能仿生算法, 在更加复杂的环境中, 所提出的算法能有效平衡优化目标, 找到更优的安全路径.

关键词: 多目标路径规划; 自然启发式算法; 强化学习; NSGA-II; 移动机器人

引用格式: 周毅,刘俊.融合强化学习的多目标路径规划.计算机系统应用,2024,33(3):158–169. http://www.c-s-a.org.cn/1003-3254/9418.html

Multi-objective Path Planning Based on Reinforcement Learning

ZHOU Yi, LIU Jun

(School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan 430081, China)

Abstract: The path planning problem for mobile robots involves a large number of nodes and a wide search space. It also considers factors such as safety and real-time requirements. To address the multi-objective path planning problem for mobile robots, this study proposes a novel multi-objective intelligent optimization algorithm that combines reinforcement learning. Firstly, the algorithm adopts NSGA-II as the base framework and equips individuals with learning capabilities by reinforcement learning. A SARSA operator is designed to improve the global search efficiency of the algorithm. Secondly, to accelerate the convergence speed and ensure population diversity, the study introduces an adaptive simulated binary crossover operator (tanh-SBX) as an auxiliary operator and divides the population into two sub-populations with different properties: elite and non-elite populations. Finally, the study designs four different strategies and calculates the probability of updating strategies using the Metropolis criterion of the simulated annealing algorithm. It allows the most suitable strategy to guide the population's optimization direction, balancing exploration and exploitation. Simulation experiments demonstrate that the proposed algorithm can find optimal paths in environments with different complexities. Compared to traditional intelligent biomimetic algorithms, the proposed algorithm effectively balances optimization objectives and discovers safer and better paths in more complex environments.

Key words: multi-objective path planning; natural heuristic algorithm; reinforcement learning; NSGA-II; mobile robot

① 基金项目: 国家自然科学基金 (62173259)
收稿时间: 2023-09-07; 修改时间: 2023-10-09; 采用时间: 2023-10-16; csa 在线出版时间: 2023-12-26
CNKI 网络首发时间: 2023-12-28

路径规划是一个典型的大规模全局优化问题. 由于其节点数量众多、搜索空间广阔, 传统的动态规划求解方法常受到求解时间的限制. 虽然智能优化方法能够在可接受的时间范围内找到次优解, 但是往往容易陷入局部最优或搜索停滞等问题[1]. 在机器人领域中, 路径规划是研究的热点问题之一[2]. 如何使移动机器人能够自主寻找长度最短、平稳性最佳的路径是实现自主导航的关键问题. 机器人能否成功地自主导航, 取决于其智能能力. 通过让机器人学习周围环境的信息, 并引导它运用智能技术从一个起始点安全地移动到指定地点, 完成避障、寻优等一系列子任务, 最终找到一条无碰撞的优化路径[3].

移动机器人的单/多目标路径规划问题可转化为多目标优化问题 (MOP) 进行研究. 近年来, 许多研究者已经认识到各种多目标进化算法 (multi-objective evolutionary algorithm, MOEA) 是解决 MOP 的有效途径.

目前, 已经发展出许多进化算法, 包括遗传算法 (genetic algorithm, GA)、粒子群算法 (particle swarm optimization, PSO)、蚁群算法 (ant colony optimization, ACO) 等一系列算法. 研究者们对这些算法进行了改进, 尝试解决移动机器人的单/多目标路径规划问题. 例如, Singh 等人[4]利用多目标 NSGA-II 启发式算法优化无人机的飞行轨迹, 开发的 NSGA-II 模型演变为最优的无人机飞行轨迹, 同时实现了无人机能耗最小化、节点能耗最小化、平均 RSSI 最大化的目标. Ajeil 等人[5]使用一种新颖的自然启发式算法实现的点生成, 该算法是将粒子群优化和改进频率蝙蝠算法 (PSO-MFB) 进行有效结合, PSO-MFB 生成并选择满足条件的点, 将不可行解转化为可行解, 最后通过避障算法形成一条无碰撞路径. Gul 等人[6]是通过灰狼优化器与粒子群优化算法 (PSO-GWO) 的结合来优化路径, 将 PSO-GWO 算法生成的所有最优可行解与局部搜索技术相结合, 将所有不可行解转换为可行解, 最后使用防撞检测和避障算法, 避免机器人碰撞障碍物. 上述研究者采用智能仿生算法让机器人智能化的躲避障碍物, 完成自主导航任务. 然而, 这些算法仍然存在一些缺陷. 例如, 当算法参数设置不合理时, 算法性能将会受到较大影响, 甚至无法找到一条有效路径; 当算法陷入局部最优时, 无法跳出; 算法的全局搜索与局部搜索平衡性的问题等.

随着深度强化学习 (deep reinforcement learning, DRL) 的兴起, 强化学习在多目标路径规划中也得到广泛应用. 在现实世界中, 移动机器人需要在考虑多个目标的情况下进行路径规划, 如最短路径、最小能耗、最大效率等. 多目标强化学习算法, 例如 NSGA-II 和 SPEA2 等的应用, 使得机器人能够在不同目标之间进行权衡和优化. 现实环境中的路径规划在某些情况下, 机器人可能处于部分可观测环境中, 即无法直接观测到完整的状态信息. 针对这种情况, 研究者们通过将历史观测信息纳入决策, 提高了路径规划的性能, 例如刘晓峰等人[7]提出一种基于记忆启发的强化学习方法, 不需要植入先验知识, 利用启发式回报函数改造 Q 学习方法, 提高搜索效率. 当涉及大规模的状态空间和复杂的动作空间, 这使得算法训练过程非常耗时, 王楷文等人[8]提出一种将深度强化学习与状态预测相结合的多智能体动态避碰路径规划方法, 将多障碍物避碰问题转换为时序单障碍物避碰问题, 提高了训练效率与避碰能力. 对于超参数难以设置的缺点, Kiran 等人[9]提出了一种分布式可变长度遗传算法框架, 可以系统地调整各种强化学习应用的超参数, 通过进化改进训练时间和架构的稳健性. 对于复杂的路径规划问题, 算法的收敛和稳定性也是一个难点, Dong 等人[10]提出一种自适应双记忆经验回放结构, 利用双记忆库结构拆分经验数据, 调整 HER 机制比例, 提高了算法的成功率, 并保证训练效率.

综上所述, 目前路径规划算法依然存在一些问题, 例如, 如何更好地避免陷入局部最优解, 如何更好地平衡全局搜索和局部搜索的效率问题等.

针对这些问题, 本文提出了一种融合强化学习的多目标路径规划算法 (RLAP-NSGA-II), 不仅提出了两种不同的交叉算子: SARSA 算子和 tanh-SBX 算子, 而且设计了一种基于 Metropolis 准则的 SA 进化策略, 平衡算法的探索与利用能力. 本文主要贡献如下.

(1) 为了提升算法在迭代前期的全局搜索能力, 提出了 SARSA 算子, 将强化学习中的 SARSA 算法融入 NSGA-II 的迭代过程中, 在线学习种群的基因特征与动作空间并不断更新策略, 引导子代种群的进化方向, 加快了算法在迭代前期的探索效率, 避免了较多无效路径的探索.

(2) 防止算法在逼近全局最优解的过程中, 出现振荡现象, 提出了 tanh-SBX 算子. 该算子通过引入 tanh 函数, 使其局部搜索具备动态调整能力, 快速逼近全局最优, 提高了算法在迭代后期的利用效率, 使优化后的

近似最优路径尽可能地逼近全局最优.

(3) 在平衡全局搜索与局部搜索的效率问题上, 提出了一种 SA 进化策略. 将种群分为两种不同性质的子种群: 精英种群和非精英种群, 不同的算子作用于不同性质的种群, 得到 4 种进化策略. 通过模拟退火算法 (SA) 的 Metropolis 准则计算策略更新概率, 设计一种衰减函数, 基于该函数调整 Metropolis 准则中的温度系数, 使概率不断趋于零. 最终, 更新策略的概率趋于零, 使得搜索过程逐渐稳定, 在平衡算法探索与利用效率的同时有效平衡了路径的长度和平稳度并实时更新策略, 找到更优的安全路径.

1 移动机器人路径优化的数学模型

1.1 障碍物环境模型

为了简化障碍物的外观, 将采用路径规划中常用简化环境的方法——栅格法, 它不仅可以快速建立环境模型, 而且简化的障碍物对模型的影响很小. 该方法是将环境以矩阵的形式分割成若干矩形块, 障碍物所在区域为不可行区域, 所在的矩形块用黑色填充, 在矩阵中对应的值为 1. 其余部分均为可行区域, 所在的矩形块用白色填充, 在矩阵中对应的值为 0. 同时结合矩阵坐标系和序号法, 建立障碍物环境模型. 图 1 为 20×20 障碍物环境模型, 将坐标 (1, 1) 的序号定为 0, 从左至右, 从下至上标号.

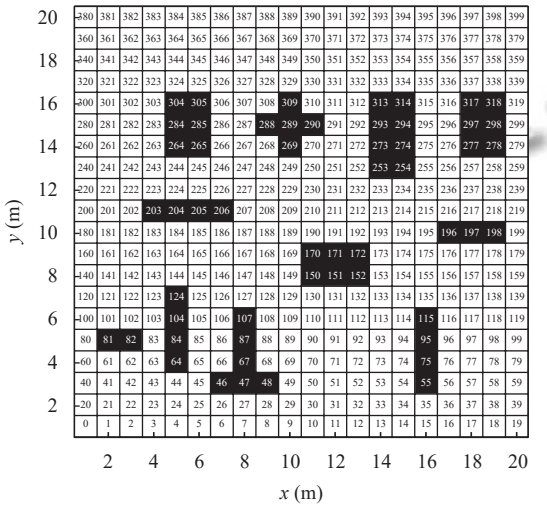


图 1 20×20 障碍物环境模型

该序号具有唯一性, 可通过式 (1) 计算序号对应的坐标值, 其中 x^n 表示序号 n 的横坐标, y^n 表示序号 n 的纵坐标, G^x 表示地图 G 的横坐标上限, G^y 表示地图 G 的纵坐标上限, $\mathrm{mod}(\cdot)$ 表示取模运算, $fix(\cdot)$ 表示取整运算, 即四舍五入为最近的整数.

$$\begin{cases} x^n = \mathrm{mod}\,(n, G^x) + 1 \\ y^n = fix\left(\dfrac{n}{G^y}\right) + 1 \end{cases} \tag{1}$$

1.2 优化问题的数学模型

移动机器人在进行路径规划任务时, 首先需要解决的问题是如何在避障的同时, 使路径最短和平稳度最佳. 将路径长度和平稳度作为优化目标, 路径长度即路径的长度, 平稳度使用角度改变量 (平滑度) 和转弯次数 (拐点数) 进行量化. 例如, A 由 n 个序号 s 组成的一条路径, 即 $A = \{s^1, s^2, \cdots, s^n\}$, 那么路径 A 的长度、平滑度和拐点数可用式 (2) 计算:

$$\begin{cases} L_A = \displaystyle\sum_{i=2}^{n} \left| s^i - s^{i-1} \right| \\ E_A = \displaystyle\sum_{i=3}^{n} e\left(s^i, s^{i-1}, s^{i-2}\right) \\ P_A = \displaystyle\sum_{i=3}^{n} p\left(s^i, s^{i-1}, s^{i-2}\right) \end{cases} \tag{2}$$

其中, L 表示路径 A 的长度, E 表示平滑度, P 表示拐点数, $e(\cdot)$ 是计算直线 (s^i, s^{i-1}) 与直线 (s^{i-1}, s^{i-2}) 组成的夹角, $p(\cdot)$ 是判断路径 (s^i, s^{i-1}, s^{i-2}) 是否存在拐点, 若存在, $p(s^i, s^{i-1}, s^{i-2}) = 1$, 反之为 0. 综上所述, 目标函数可用式 (3) 表示如下:

$$\begin{cases} \min L = \displaystyle\sum_{i=2}^{n} \left| s^i - s^{i-1} \right| \\ \min E = \displaystyle\sum_{i=3}^{n} e\left(s^i, s^{i-1}, s^{i-2}\right) \\ \min P = \displaystyle\sum_{i=3}^{n} p\left(s^i, s^{i-1}, s^{i-2}\right) \end{cases} \tag{3}$$

2 RLAP-NSGA-Ⅱ

2.1 tanh-SBX 算子

SBX 算子是 Deb 等人[11]提出的一种模拟单点二进制交叉的交叉算子, 该算子是主要针对解决实数编码方式在进行单点交叉时, 前后个体的基因信息差别较大, 无法体现出子代继承父代基因信息的特点. 例如, 当两个父代基因信息分别为 $X^a(x_1^a, \cdots, x_p^a, \cdots, x_n^a)$、$X^b(x_1^b, \cdots, x_p^b, \cdots, x_n^b)$, 若交叉位点在 p 处, 通过单点交叉算子得到两个子代 $X^c(x_1^a, \cdots, x_p^b, \cdots, x_n^b)$、$X^d(x_1^b, \cdots, x_p^a, \cdots,$

x_n^a), 由于编码方式为实数编码, 父代与子代相似度大幅减小, 导致子代与父代的适应度值变化过大, 并不是期待的结果. 为了使子代保留父代的部分基因信息, 父代可通过 SBX 算子得到子代 $X^e(x_1^e,\cdots,x_p^e,\cdots,x_n^e)$、$X^f(x_1^f,\cdots,x_p^f,\cdots,x_n^f)$, 可根据式 (4) 计算得到:

$$\begin{cases} x_i^e = 0.5\times\left[(1+\gamma)\cdot x_i^a + (1-\gamma)\cdot x_i^b\right] \\ x_i^f = 0.5\times\left[(1-\gamma)\cdot x_i^a + (1+\gamma)\cdot x_i^b\right] \end{cases} \tag{4}$$

其中, γ是根据交叉分布指数η依据式 (5) 随机生成得到:

$$\gamma = \begin{cases} (2\lambda)^{\frac{1}{1+\eta}}, & \lambda < 0.5 \\ \left(\dfrac{1}{2-2\lambda}\right)^{\frac{1}{1+\eta}}, & \text{else} \end{cases} \tag{5}$$

交叉分布指数η是用户定义的参数, η值越大则产生子代与父代相似度越高, 子代能保留部分父代基因信息, η值越小则产生子代与父代的差距越大, 建议$\eta = 20$, 可使算子达到较好的效果. SBX 算子在局部优化搜索上表现较佳, 同时也容易陷入局部最优, 由于η值是人为设置的固定参数, 当种群搜索至局部最优区域时, 为了保留父代基因信息, 生成子代均与局部最优点高度相似, 从而无法摆脱局部最优. 文献[12]指出让交叉分布指数η动态变化, 可使算法得到更优的解, 因此本文将设计一种具备自适应能力的 tanh-SBX 算子, 使种群在面临局部最优时能有一定能力跳出局部最优.

为了使算子具备跳出局部最优的能力, 首先需要解决两个核心问题: 第一, 如何识别是否陷入局部最优. 第二, 陷入局部最优时, 如何跳出局部最优. 针对上述问题, 有两种解决思路: 第一, 可根据当代种群与上一代种群的目标函数值在迭代的过程中是否变化极小, 并且与全局最优解仍有较大差距判断是否陷入局部最优, 当陷入局部最优时, 可将交叉分布指数η设置一个较小的数, 使种群尽可能生成多样性较高且保留较少局部最优解基因信息的子代, 从而跳出局部最优. 第二, 由于局部最优的目标空间距离全局最优的目标空间较远, 初始种群与全局最优的目标空间也具有较远的距离, 因此在算法的迭代前期应该快速锁定全局最优的大致方位, 并朝向全局最优方向逼近, 即迭代前期尽可能生成多样性较高的子代, 去搜索全局最优的大致方位, 而不是不断保留父代的部分基因信息去局部搜索, 不仅花费过多的计算资源, 搜索效率也会大大降低; 迭代中期种群易陷入局部最优的原因是算子的局部搜索能力太强, 子代与父代差距较小且多样性较差, 无法跨越局部最优区域, 可将交叉分布指数η设置为一个较小的值削弱局部搜索能力, 提高子代的多样性; 迭代后期种群应该收敛至全局最优区域, 此时算法的收敛性依赖于全局最优区域的局部搜索能力, 为了提高收敛速度, 可将交叉分布指数η设置为一个较大的值, 增强局部搜索能力, 不断逼近全局最优. 综上所述, 可根据种群的迭代过程动态调整交叉分布指数η实现算法在不同阶段的搜索任务.

对于第 1 个解决思路, 需要已知全局最优的信息, 同时在搜索过程中消耗过多计算资源, 影响搜索效率. 因此采用第 2 个解决思路, 为了使算法更好地完成不同阶段的搜索任务, 交叉分布指数η将根据式 (6) 计算得到:

$$\eta(n) = \frac{\theta(\exp(n)-\exp(-n))}{2(\exp(n)+\exp(-n))} + \frac{\theta}{2} \tag{6}$$

式 (6) 是由 tanh 函数改进而来, 保留了 tanh 函数的饱和性, 其中θ是由用户定义的交叉分布指数η的上界值$(0<\eta<\theta)$, 定义域$n\in(-m,+m)$. 为了使 tanh 函数正常工作, 需要将种群的代数N根据式 (7) 映射到 tanh 函数的定义域中:

$$n = \frac{N}{N_{\max}}\times 2m \tag{7}$$

其中, $N_{\max}$表示最大迭代次数, m是根据用户需求设置定义域范围, 由于η值的微小变化对算法搜索效率影响不大, 因此m值没必要取的过大, 同时必须要覆盖η值的上下界, 建议设置$m=5$.

2.2 SARSA 算子

优化算子不仅主导着算法的优化方向, 而且决定着结果的质量, 因此大部分学者通过改进算子改善算法的性能, 文献[13]提出的一种自适应交叉变异算子, 通过自适应调节交叉概率和变异概率确保优质解的保存和对劣质解的破坏. 文献[14]利用指数函数对传统遗传算法的交叉、变异算子自调整公式进行改进, 使自适应策略更加符合实际情况, 同时能够避免算法陷入局部最优解. 文献[15]改进传统遗传算法的选择、交叉、变异算子, 采用分层法对种群个体进行选择操作, 采用单点交叉法对种群个体进行交叉操作, 采用八邻域单点变异法对变异算子种群个体进行变异操作, 提高了算法的收敛速度. 上述改进方法弥补了传统算法的一些缺陷, 但鲁棒性较弱, 原因在于算子缺乏自主学

习能力, 因此引入了 SARSA 算子. 在强化学习算法中, SARSA 算法是一种使用时序差分求解强化学习控制问题的方法, 该算法一直使用同一个策略π更新价值函数和选择新的动作. 该算法的名称是由S, A, R, S, A几个字母组成, 其中S代表状态 (state), A代表动作 (action), R代表奖励 (reward), 在迭代过程中, 该算法严格按照策略π选择两个动作, 但只执行第 1 个动作A, 通过第 3 个动作A'更新价值函数, 更新规则表示为式 (8):

$$q(s,a)=q(s,a)+\alpha\left[r+\gamma q(s',a')-q(s,a)\right] \tag{8}$$

其中, α表示学习率, γ表示折扣因子.

策略: 在遗传算法中, 将策略$\pi(a|s)$表示为个体s采取交叉动作a后, 根据子代的排序等级确定动作a被选中的概率, 具体表示为式 (9):

$$\pi(a|s)=\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|A(s)|}, & a=\arg\max_a rank\left(X_j,a\right)\\ \dfrac{\varepsilon}{|A(s)|}, & a\neq\arg\max_a rank\left(X_j,a\right)\end{cases} \tag{9}$$

种群初始化后, 将ε贪心策略与初始种群 (环境) 进行交互, 得到初始策略.

奖励机制: 在强化学习中, 除了策略以外, 还需要引入一些必不可少的概念, 强化学习的核心在于如何进行奖励, 合理的奖励机制会引导种群的交叉方向, 促进种群生成优良子代.

在遗传算法优化任务中, 采用ε贪心策略后, 假设父代$S_i\ (i=1,2,\cdots,m)$随机选定母代$S_r\ (r=randperm(m))$进行交叉, 选择的交叉位点 (动作) $A_j\ (j=1,2,3,\cdots,n)$不同, 将会产生不同的子代$X_j\ (j=1,2,3,\cdots,n)$, 其中n为个体基因段个数, 如图 2 所示,计算子代X_j在种群中的排序等级$rank(X_j)$来确定该交叉动作A_j的奖励R_j, 其中$X_{\text{merge}}=X_j\cup X_{rank=1}$, $X_{rank=1}$表示种群中的非支配解. 遗传算法的优化过程是可行解逐渐逼近理论pareto前沿(PF_{true})的过程, 因此只需要让可行解集不断向理论pareto前沿的方向进化即可, 交叉动作得到排序等级高的子代, 奖励R_j理应更大 (见图 3).

因此将$1/rank(X_j)$作为交叉动作A_j的奖励, 可表示为式 (10):

$$\begin{cases}rank(X_j)=g(X_j,X_{\text{merge}})\\ R_j=\dfrac{1}{rank(X_j)}\end{cases} \tag{10}$$

其中, $g(\cdot)$表示执行非支配算法的函数.

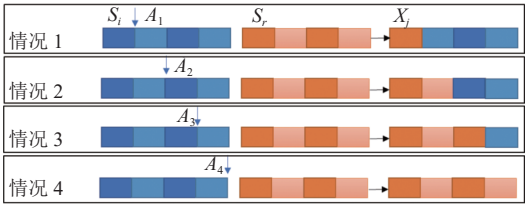


图 2 种群交叉情况图

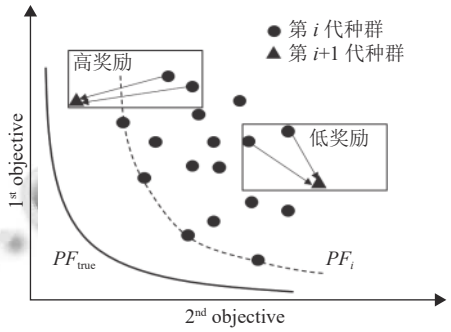


图 3 奖励机制示意图

回报: 当智能体采用策略π确定第 2 个动作a'后, 可根据式 (11) 计算回报的估计值.

$$U=R+\gamma q(s',a') \tag{11}$$

其中, γ表示折扣因子.

动作价值函数: 通过式 (11) 计算回报的估计值更新动作价值, 可用式 (12) 表示如下:

$$q(s,a)=q(s,a)+\alpha[U-q(s,a)] \tag{12}$$

其中, α表示学习率, 通过动作价值函数量化策略引导种群的优化效果, 学习率α可以缩小动作价值和实际回报之间的差距, 不断更新策略, 始终选择更优的策略引导种群迭代, 确保种群优化方向的准确性. 综上所述, SARSA 算子的伪代码如算法 1 所示.

算法 1. SARSA 算子伪代码

输入: 迭代次数T, 状态集 (种群基因) S, 动作集A, 学习率α, 折扣因子γ, 探索率 ϵ.

输出: 最优策略$\pi(a|s)$和最优动作价值函数$q(s,a)$.

(1) 随机初始化动作价值$q(s,a)$, 终止状态的q值初始化为 0.

(2) for $i=1:T$

1) 初始化个体s作为当前状态序列的第 1 个状态, 根据策略π确定动作a.

2) if $s'\neq s_{\text{end}}$

① 执行动作a, 得到奖励R和新种群s'

② 采用策略π确定动作a'

③ 计算回报的估计值$U=R+\gamma q(s',a')$

④ 更新动作价值函数$q(s,a)=q(s,a)+\alpha[U-q(s,a)]$

⑤ 根据$q(s,a)$修改策略$\pi(a|s)$的 ϵ 值

⑥ $s=s'$, $a=a'$

3) else 转到步骤②

2.3 SA 进化策略

启发式算法在求解问题的最优解时, 可以将算法的搜索行为分为: 探索与利用. 如何平衡探索与利用是应该解决的核心问题, 文献[16]表明在 AGLDPSO 中, 采用主从多亚种群分布模型, 将整个种群划分为多个亚种群, 这些亚种群共同进化, 最后实验证明与其他单种群进化或集中式机制的大规模优化算法相比, 多亚种群分布式协同进化机制将充分交换不同亚种群之间的进化信息, 进一步增强种群多样性. 受到该文献的启发, 将种群划分为两个子种群: 精英种群和非精英种群, 并设计了 4 种策略, 最后通过模拟退火算法的 Metropolis 准则更新概率.

在每个迭代步骤中, 根据 Metropolis 准则计算更新策略的概率, 根据衰减函数来调整 Metropolis 准则中的温度参数, 以使概率随着时间的推移而逐渐降低. 最终, 更新策略的概率趋于零, 使得搜索过程逐渐趋于稳定, 伪代码如算法 2.

算法 2. SA 进化策略伪代码

输入: 策略集 M, 初始种群 pop, 初始温度 T_0, 终止温度 T_f, 降温速度 $K\in(0,1)$.

输出: 当前种群 pop'.

(1) 令 $T=T_0$, 任取初始策略 m_1

(2) if $T>T_f$, 对当前温度 T, 执行以下步骤:

1) 采用当前的策略 m_1 进化种群 pop, 根据精英率 r_1 评估当前策略的优劣, $r=N_{\text{elite}}/N_{pop}$, 其中 N_{elite} 为精英个体数量, N_{pop} 为总个体数量.

2) 对当前策略进行随机扰动, 得到另一个策略 m_2

3) 根据 m_2 策略进化种群, 计算精英率 r_2

4) 计算精英率的增量 $\mathrm{d}r=r_2-r_1$

5) 根据 Metropolis 准则进行判断, if $\mathrm{d}r>0$

接受策略 m_2 作为新的当前策略, $m_1=m_2$.

6) else

① 计算 m_2 接受的概率 $p=\exp(-\mathrm{d}r/T)$

② if $\exp(-\mathrm{d}r/T)>rand(0,1)$, 接受策略 m_2, $m_1=m_2$.

③ else 保留策略 m_1

④ 根据策略 m_1 进化种群 pop, 得到子代种群 pop'

7) 根据衰减函数进行降温: $T=T\times K$, 转到步骤 (2)

(3) else 输出当前种群

全局探索策略 m_1: 全局探索策略旨在探索全局空间, 让种群在有限的迭代次数中尽可能逼近全局最优区域. 由于传统全局搜索算子的效率十分缓慢, 将采用 SARSA 算子解决这个问题. SARSA 算子是一个在全局搜索的效率和范围上进行折中的全局搜索算子. 该算子对精英种群和非精英种群的基因信息进行学习, 得到一个最优策略以指导种群的搜索方向, 减少无效搜索, 为了尽可能使搜索范围覆盖全局, 策略初始定义为柔性策略.

潜力挖掘策略 m_2: 潜力挖掘策略是针对非精英种群设计的, 旨在将普通个体的转化为精英个体, 从而扩充精英种群的数量. 非精英种群和精英种群均是由整个种群中划分而来, 对于全局空间而言, 它们处于同一个或相邻的局部子空间, 基因信息具有一定的联系, 因此使用局部搜索算子可以充分挖掘普通个体之间的基因信息, 使其转化为精英个体. tanh-SBX 算子作为该策略的局部搜索算子.

加速收敛策略 m_3: 精英种群占比不断快速提升, 并占据主导地位, 非精英种群数量越来越少, tanh-SBX 算子逐渐失去效用. 精英种群通过 SARSA 算子逼近全局最优的效率较低, 这是因为 SARSA 算子的搜索范围和步长较大, 只能在全局最优附近振荡, 无法快速逼近全局最优, 因此通过 tanh-SBX 算子在全局最优区域进行局部搜索, 使种群快速逼近全局最优, 加快算法的收敛速度.

脱离局部最优策略 m_4: 加速收敛策略结束后, 种群有两种状态: 陷入局部最优和无限逼近全局最优. 针对第 1 种状态, 本文设计了脱离局部最优策略, 该策略是使非精英种群通过 SARSA 算子增加非精英种群的多样性, 让非精英种群尽可能地找到全局最优解空间, 最后通过 tanh-SBX 算子在全局最优区域内局部搜索, 带领种群无限逼近全局最优.

在算法的迭代前期, 初始解集随机分散于解空间的任意位置, 需要确定全局最优解的大致方向, 由于初始解空间过大, 需要消耗大量的计算资源和时间成本, 因此采用全局探索策略, 而 SARSA 算子是针对该问题被设计而来的. 采用全局探索策略之后, 精英个体会缓慢增加, 此时将会采用潜力挖掘策略, 是为了提高非精英个体转化为精英个体的效率, 精英种群占据主导地位之后, 此时将会出现两种情况, 第 1 种是处于全局最优解附近. 第 2 种是处于局部最优解附近. 针对这两种情况, 采用两种策略去应对, 第 1 种情况会采用加速收敛策略, 快速逼近全局最优解, 得到优化结果; 第 2 种情况会采用脱离局部最优策略, 让非精英种群尽可能地找到全局最优解空间, 脱离局部最优.

图 4 是 RLAP-NSGA-II 的流程图, 其中策略集 $M=\{m_i|i=1,2,3,4\}$, 上述 4 种策略均是以图 5 的流程作为基础框架, 不同之处在于交叉的对象不同和交叉算子不同, 其他设置均与文献[17]保持一致.

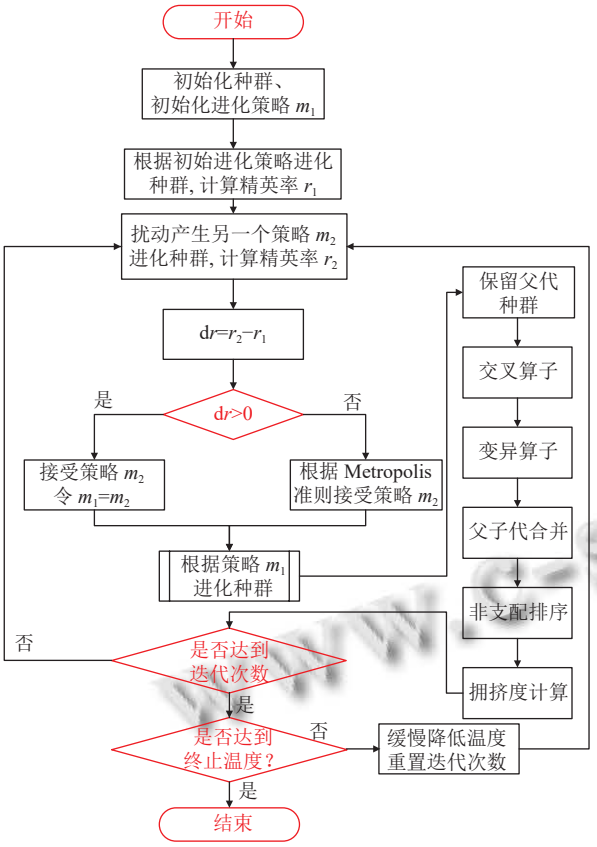


图 4 算法流程图

2.4 编码方式

根据第 2 节建立的数学模型可知, 一段序号可以表示一条路径, 例如, A 由 n 个序号 s 组成的一条路径, 即 $A=\{s^1,s^2,\cdots,s^n\}$. 由于该编码方式高效简洁, 在进行交叉和变异算子时更容易操作, 将地图环境与种群基因建立联系, SARSA 算子可直接通过种群基因与地图环境进行交互, 因此采用该编码方式表示路径.

2.5 避障处理

首先, 对环境中的障碍物进行静态建模和识别, 然后, 将这些障碍物的信息存储在地图中, 最后, 在地图的可行区域随机生成若干条初始路径, 在遗传操作生成新路径时, 还需对路径进行碰撞检测, 判断路径是否与障碍物相交, 如果存在碰撞, 则对路径进行变异操作生成新路径.

2.6 适应度值

适应度值是衡量个体在适应度函数下的优劣程度的指标. 适应度函数的设计应该基于问题的特性和目标, 以便使得算法能够在合理的时间内找到最优解或次优解, 因此将适应度值与优化目标建立数学关系, 如式 (13) 所示:

$$F=\frac{1}{L}+\frac{1}{E}+\frac{1}{P} \tag{13}$$

其中, L、E 和 P 可根据式 (2) 计算.

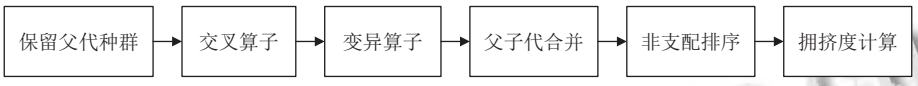


图 5 策略流程图

2.7 算法理论性分析

算法的理论性分析将从 3 个方面进行: 收敛速度、收敛精度和时间复杂度.

算法的收敛速度和收敛精度与每次迭代的更新步长有直接关系, 当算法处于探索状态时, 如果算法的更新步长过小, 那么需要更多的迭代次数才能接近最优解. 这一阶段采用具有全局搜索能力的 SARSA 算子, 通过学习率 α 更新策略的参数, 实际上就是通过学习率 α 逐渐加大了父代与子代的差距, 让每次迭代的更新步长逐渐增加, 算法可能会更快地接近最优解, 那么算法的收敛速度更快. 当算法处于利用状态时, 过大的更新步长会导致算法始终在最优解附近振荡, 此时全局探索策略无法完成优化任务, 通过模拟退火算法更新优化策略, 选择更新步长较小的优化策略达到逼近最优解的目的. 这一阶段采用具有局部搜索能力的 tanh-SBX 算子, 将迭代次数通过 tanh 函数引入到计算交叉分布指数 η 的公式中, 通过迭代次数的线性增长实现交叉分布指数 η 的非线性增长, η 值越大, 更新步长越小, 越接近最优解, 那么算法的收敛精度越高.

RLAP-NSGA-II 的时间复杂度由以下部分组成: 非支配排序和 SARSA 算子. 假如待优化目标为 M 个, 种群规模为 N, 在开始非支配排序时, 每个个体的 $n_p=0$, $S_p=\varnothing$. 在第 1 轮排序后, 每个个体 p 会记录 n_p 和 S_p, 故第 1 轮时间复杂度为 $O(MN^2)$, 在后续的排序中, 只需要对上一轮排序的非支配解集取一个个体 p, 在 p 的 S_p 中取出个体 q, 此时 $n_q=n_q-1$, 若 n_q 为 0, 则个体 q 进入下一轮排序的非支配解集. 若每轮都只有一个个体被选出, 每次需要花费 $O(N)$, 所以非支配排序的时间复杂度为 $O(MN^2)$. 假如 SARSA 算子需要训练 N 次收敛,

每次训练的时间步数量为M, 可根据算法 1 计算出 SARSA 算子的时间复杂度:

$$\begin{aligned}T(episode,t)=&t_1+(t_2+t_{2.1}+(t_{2.2}+t_{2.2.1}+t_{2.2.2}\\&+t_{2.2.3}+t_{2.2.4}+t_{2.2.5}+t_{2.2.6})\times M)\times N\\=&t_1+(t_{a1}+t_{a2}\times M)\times N\\=&t_1+t_{a1}\times N+t_{a2}\times M\times N\\=&t_{a2}\times M\times N\\=&M\times N\end{aligned}\tag{14}$$

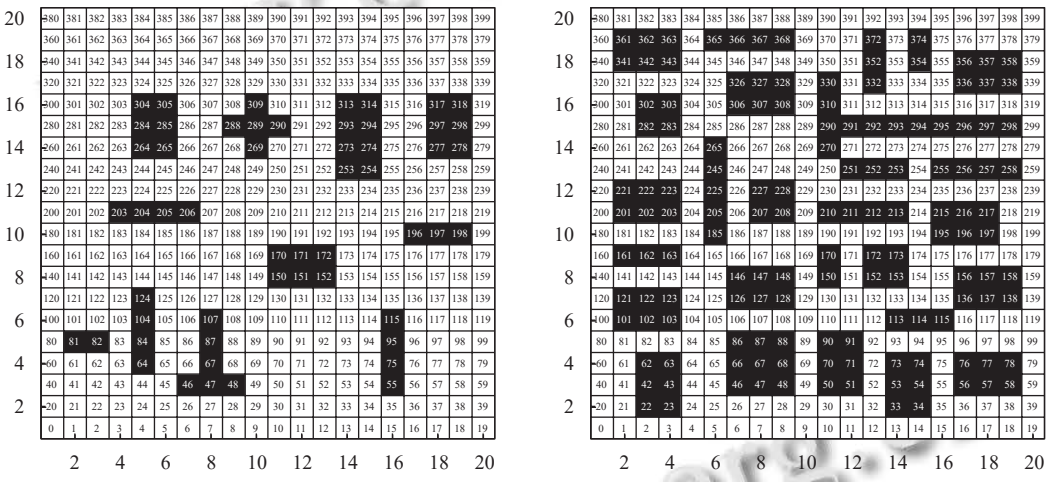
其中, t_1表示算法 1 步骤 (1) 所花费的时间, 当训练次数和时间步数量足够大时, t_1、t_{a1}和t_{a2}对M和N的影响很小, 因为$M\times N$的增长速度明显快于N, 所以函数T的主要影响因素是$M\times N$, SARSA 算子的时间复杂度为O(MN). 综上所述, RLAP-NSGA-II 的时间复杂度为O(MN^2).

3 实验结果和性能分析

实验平台的计算机操作系统是 Windows 10 (64 位), 仿真实验将在 AMD-RYZEN7-5800H CPU @ 3.2 GHz 的环境下进行. 本文从以下几个方面进行分析: 1) 不同环境下路径规划的仿真与对比分析. 2) RLAP-NSGA-II 的多样性分析.

3.1 不同环境下路径规划的对比分析

环境的复杂性主要由环境中障碍物的比例和密度来定义, 分为简单环境和复杂环境. 在简单环境中障碍物的比例和密度比在复杂环境中要小. 在本节中, 为了验证 RLAP-NSGA-II 的性能, 将采用 2 张不同障碍覆盖率的地图环境与 NSGA-II、Q-learning[18]和 IACO[19]进行对比实验, 地图分别为障碍覆盖率为 13.5% 的简单环境和障碍覆盖率为 34.5% 的复杂环境如图 6.



(a) 简单环境 (b) 复杂环境

图 6 地图环境

3.2 简单环境下的路径规划对比实验

在 20×20 的简单环境下, RLAP-NSGA-II、NSGA-II、Q-learning 和 IACO 的路径规划结果由图 7 可知, RLAP-NSGA-II 的拐点数为 5 个, 且路径角度变化较小, 均小于 90°; NSGA-II 的拐点数也为 5 个, 整体路径和前者十分相似; Q-learning 的拐点数有 8 个, 路径存在一些不必要的拐点; IACO 的拐点数有 6 个, 其中有两处直角拐弯. 为了更直观地比较算法之间的性能, 计算了路径长度和平滑度 (见表 1), 从表 1 数据可知, RLAP-NSGA-II 的路径长度和平滑度均优于 IACO 和 Q-learning, 与 NSGA-II 的性能持平. 同时为了避免实验结果的偶然性, 将 RLAP-NSGA-II、NSGA-II、Q-learning 和 IACO 这 4 种算法在简单环境下进行 15 次独立实验, 仿真结果见表 2 可知, RLAP-NSGA-II 和 NSGA-II 均能收敛至全局最优解, 而 Q-learning 和 IACO 容易陷入局部最优解, 且无法跳出, RLAP-NSGA-II 与 NSGA-II 相比, 前者收敛至最优解所需的迭代次数更小, 图 8 也更好的验证了这一结论.

3.3 复杂环境下的路径规划对比实验

在 20×20 的复杂环境下, RLAP-NSGA-II、NSGA-II、Q-learning 和 IACO 的路径规划结果由图 9 可知, RLAP-NSGA-II 有 8 个拐点, 均为小角度拐弯; NSGA-II 有 10 个拐点, 与前者相比路径更曲折; Q-learning 有 19 个拐点, 路径存在较多的小角度拐弯, 导致整体路径不平滑; IACO 有 8 个拐点, 且路径出现较多直角拐弯. 从表 3 的数据可知, 在路径长度方面, RLAP-NSGA-II 优于 Q-learning 和 IACO, 但略输于 NSGA-II; 在路径平滑度方面, RLAP-NSGA-II 均优于其余的算法. 同时

为了避免实验结果的偶然性, 将 RLAP-NSGA-II、NSGA-II、Q-learning 和 IACO 这 4 种算法在复杂环境下进行 15 次独立实验, 仿真结果见表 4 可知, RLAP-NSGA-II 的最佳适应度值均优于 NSGA-II、Q-learning 和 IACO, 与 NSGA-II 相比, RLAP-NSGA-II 完全收敛所需的平均迭代次数更小, 且更加稳定. 值得注意的是, 仔细观察可发现 RLAP-NSGA-II 与 NSGA-II 的路径较为相似, 区别在于躲避第 1 个障碍物 (左下 3×2 长方形) 的选择不同, 前者牺牲了路径长度, 选择道路更为宽阔的路径空间, 避免更多的拐弯, 在路径长度与平滑度之间做了更好的妥协, 后者则更趋向于路径长度, 但增加了两个拐点, 因此在实际应用中, 前者的路径更可能被决策者选择. 总体来说, RLAP-NSGA-II 在路径长度、平滑度和拐点个数上均优于 Q-learning 和 IACO, 在处理路径长度与平滑度这对矛盾上, RLAP-NSGA-II 能找到更好的妥协方案, 图 10 也正好验证了上述结论, 在 0–100 代期间, RLAP-NSGA-II 不断更新最优路径, 与 NSGA-II 相比, 在相同的迭代次数中, RLAP-NSGA-II 的适应度值大部分高于 NSGA-II. 在 100–300 代期间, NSGA-II 最优路径的适应度值变化不明显, 可能陷入局部最优, 而 RLAP-NSGA-II 在 200 代左右适应度值发生明显变化, 脱离局部最优, 且适应度值始终高于 NSGA-II. 在 300–1 000 代期间, NSGA-II 和 RLAP-NSGA-II 均在 400 代左右适应度值发生明显变化, 但 RLAP-NSGA-II 的适应度值更高, 在路径长度与平稳度之间做了更好的妥协. 与 RLAP-NSGA-II 相比, Q-learning 和 IACO 在迭代过程中均陷入局部最优且无法跳出, 无法较好地平衡路径长度和平稳度, 虽然 Q-learning 在迭代中后期, 适应度值发生略微提升, 但适应度值仍然较低. 综上所述, 在算法收敛后, RLAP-NSGA-II 的适应度值高于 NSGA-II 和 IACO, 因此 RLAP-NSGA-II 平衡路径长度和平稳度的能力更强.

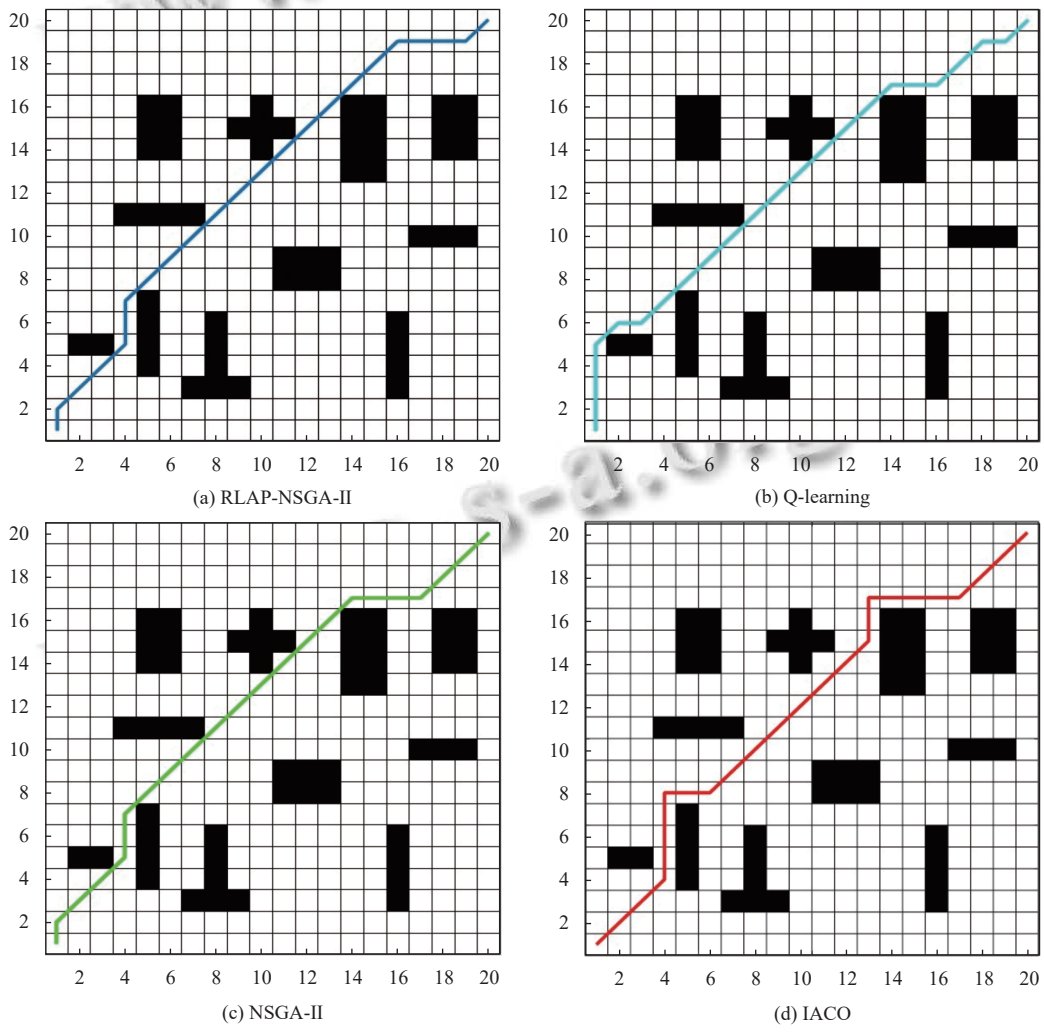


图 7 简单环境下的 RLAP-NSGA-II、Q-learning、NSGA-II、IACO 的对比结果

表 1　不同算法的各项指标值

算法	RLAP-NSGA-II	Q-learning	NSGA-II	IACO
路径长度 (m)	28.6274	29.2132	28.6274	30.2000
路径平滑度 (°)	15	21	15	42
拐点个数 (个)	5.0	8.0	5.0	6.0

表 2　4 种算法在简单环境中的仿真结果比较

算法	适应度值			迭代次数 (次)		
	最大值	平均值	标准差	最小值	平均值	标准差
RLAP-NSGA-II	0.102	0.0864	0.0117	25.00	219.0	142.4
Q-learning	0.0819	0.0708	0.0143	186.00	507.0	213.3
NSGA-II	0.101	0.0973	8.70E−3	39.00	226.0	179.8
IACO	0.0387	0.0368	8.00E−4	2.000	8.000	8.120

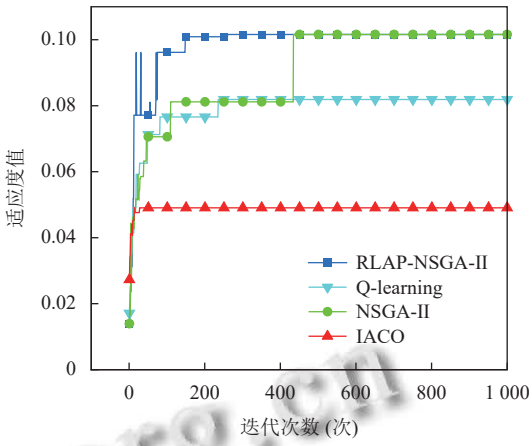


图 8　4 种算法在简单环境中的收敛曲线

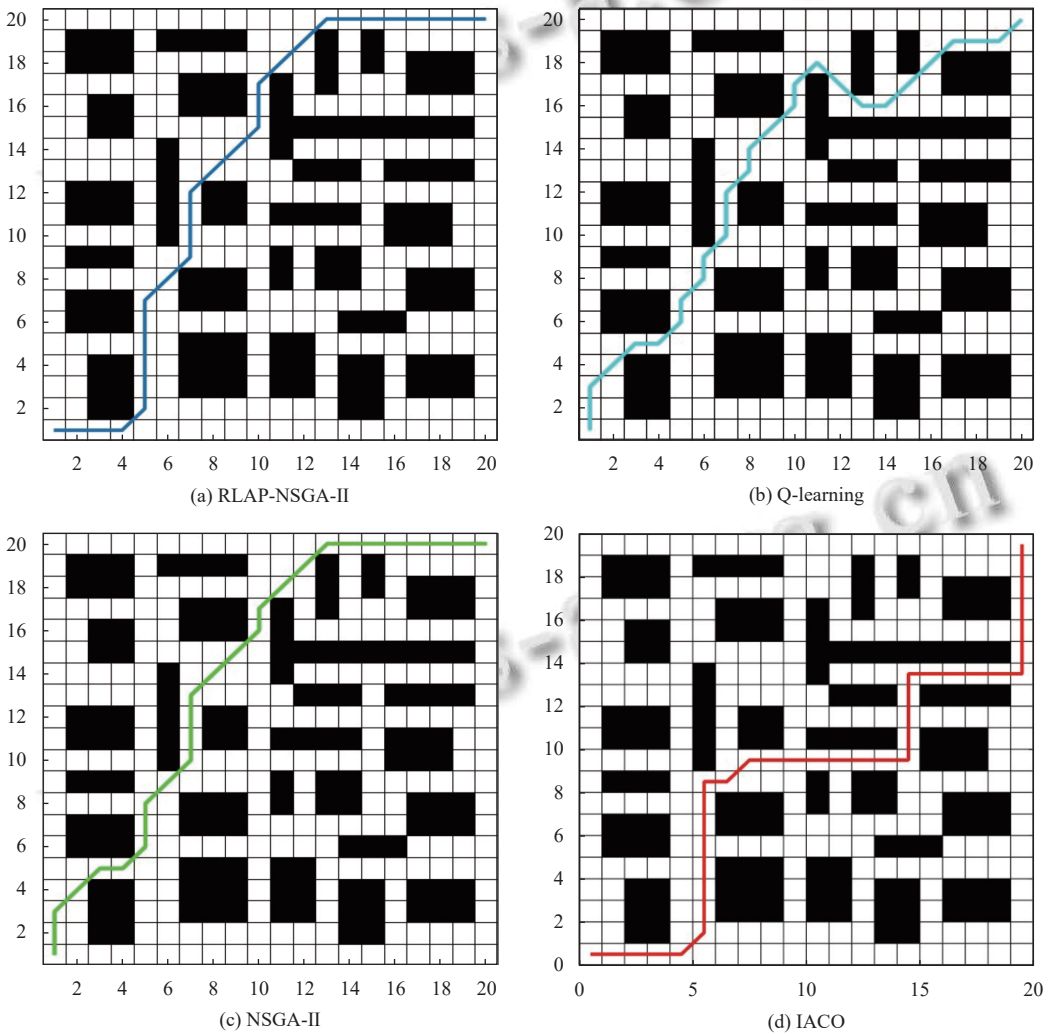


图 9　复杂环境下的 RLAP-NSGA-II、Q-learning、NSGA-II、IACO 的对比结果

3.4　不同算法的多样性分析

本节是比较不同算法之间解的多样性, 以路径规划为例, 由于复杂地图的可行解数量较少, 导致解的搜索空间较小和多样性较低, 很难体现出算法之间的差异, 因此选择简单地图进行实验, Q-learning 算法不适用多样性分析, 不参与该分析实验. 本文采用 Deb 等人[17]

设计的只衡量分布性的多样性指标Δ和 Miao 等人[20]提出的多样性指标 DIV, Δ值越小代表算法获得最优解集的分布性越好, 当 DIV 值处于波动状态时说明算法搜索最优解能力较强. 实验结果如图 11、图 12, 在迭代前期, 算法全局搜索可行解, 此时搜索空间较大, Δ值和 DIV 值均会大幅波动. 从图 11 可知, RLAP-NSGA-II 在 130 代左右Δ值趋近于 0, 且后续迭代过程中Δ值保持不变, 而 NSGA-II 和 IACO 在 400 代之后, Δ值才逐渐稳定趋近于 0.

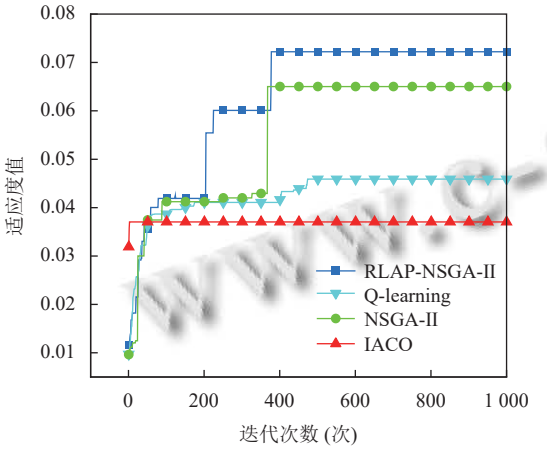


图 10 4 种算法在复杂环境中的收敛曲线

表 3 4 种算法的各项指标值

算法	RLAP-NSGA-II	Q-learning	NSGA-II	IACO
路径长度 (m)	32.7279	33.2132	31.5563	36.8000
路径平滑度 (°)	24	66	30	72
拐点个数 (个)	8.0	19	10	8.0

表 4 4 种算法在复杂环境中的仿真结果比较

算法	适应度值			迭代次数 (次)		
	最大值	平均值	标准差	最小值	平均值	标准差
RLAP-NSGA-II	0.0722	0.0611	0.0112	100.0	400.4	202.1
Q-learning	0.0453	0.0376	0.0077	235.0	572.6	212.8
NSGA-II	0.0650	0.0612	6.50E−3	139.0	421.1	235.6
IACO	0.0377	0.0366	7.00E−4	2.000	7.933	5.347

从图 12 可知, 在 200 代之后, NSGA-II 的 DIV 值保持不变, 说明该算法处于停滞状态, 种群的多样性较差, 容易陷入局部最优, 而 RLAP-NSGA-II 的 DIV 值在区间内依然保持小幅波动且大于 0, 说明算法依然在搜索新解, 搜索最优解能力较强, 不易陷入局部最优. 值得注意的是, 在迭代过程中 IACO 的 DIV 值一直处于大幅波动状态, 即使是在迭代后期种群的多样性依然有较大的变化, 说明算法一直在搜索新解, 但搜索的效率过低.

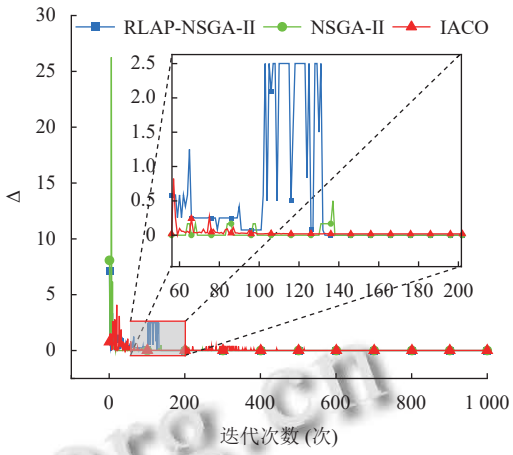


图 11 3 种算法在简单地图的分布性曲线

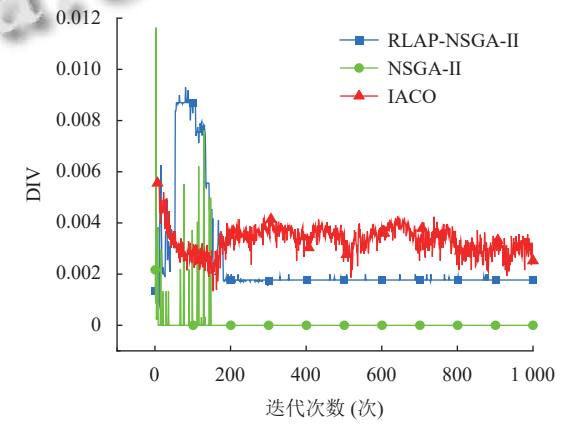


图 12 3 种算法在简单地图的多样性曲线

4 结 论

本文设计了一种融合强化学习的多目标路径规划算法 (RLAP-NSGA-II), 来弥补智能仿生算法在路径规划中的一些不足. 与其他算法不同的是该算法采用 SARSA 算子提高了算法在前期全局搜索的效率, 通过 tanh-SBX 算子加快了后期的收敛速度, 将种群划分为性质不同的子种群保持了种群多样性, 为了平衡算法探索与利用的效率问题, 设计了 4 种不同性质的进化策略, 通过模拟退火算法的 Metropolis 准则计算更新策略的概率, 使算法始终保持最适合的策略进行优化, 并有一定概率接受较差的策略, 随着时间的增加, 接受较差策略的概率趋于零, 算法的搜索过程逐渐稳定. 最后对 Q-learning、IACO、NSGA-II 和 RLAP-NSGA-II 进行不同环境地图的路径规划对比实验, 实验结果验证了, RLAP-NSGA-II 在多目标路径规划问题上的有效性. 实验研究表明, 在更为复杂的环境地图中, RLAP-NSGA-II 在处理路径长度与平稳度这对矛盾目标上, 能找到更好

的妥协方案, 且在多种环境地图下均能找到近似最优解. 相比传统智能仿生算法, 在更加复杂的环境中, 所提出的算法能有效平衡优化目标, 找到更优的安全路径.

随着后续研究的深入, 将 RLAP-NSGA-II 应用于更加复杂的动态环境下的多目标路径规划, 以验证所提出算法的性能, 或结合最新的强化学习理论范式提升算法性能, 也是值得深入研究的方向之一.

参考文献

1 Grimme C, Kerschke P, Aspar P, *et al*. Peeking beyond peaks: Challenges and research potentials of continuous multimodal multi-objective optimization. Computers & Operations Research, 2021, 136: 105489. [doi: 10.1016/j.cor.2021.105489]

2 Tan CS, Mohd-Mokhtar R, Arshad MR. A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms. IEEE Access, 2021, 9: 119310–119342. [doi: 10.1109/ACCESS.2021.3108177]

3 Oroko JA, Nyakoe GN. Obstacle avoidance and path planning schemes for autonomous navigation of a mobile robot: A review. Proceedings of the 2012 Sustainable Research and Innovation Conference. Kenya, 2012. 314–318.

4 Singh MK, Choudhary A, Gulia S, *et al*. Multi-objective NSGA-II optimization framework for UAV path planning in an UAV-assisted WSN. The Journal of Supercomputing, 2023, 79(1): 832–866. [doi: 10.1007/s11227-022-04701-2]

5 Ajeil FH, Ibraheem IK, Sahib MA, *et al*. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. Applied Soft Computing, 2020, 89: 106076. [doi: 10.1016/j.asoc.2020.106076]

6 Gul F, Rahiman W, Alhady SSN, *et al*. Meta-heuristic approach for solving multi-objective path planning for autonomous guided robot using PSO-GWO optimization algorithm with evolutionary programming. Journal of Ambient Intelligence and Humanized Computing, 2021, 12(7): 7873–7890. [doi: 10.1007/s12652-020-02514-w]

7 刘晓峰, 刘智斌, 董兆安. 基于记忆启发的强化学习方法研究. 计算机技术与发展, 2023, 33(6): 168–172, 180. [doi: 10.3969/j.issn.1673-629X.2023.06.025]

8 王楷文, 施文. 基于深度强化学习与状态预测的多智能体动态避碰路径规划方法研究. 2022 中国自动化大会论文集. 厦门: 中国自动化学会, 2022. 575–580. [doi: 10.26914/c.cnkihy.2022.053884]

9 Kiran M, Ozyildirim M. Hyperparameter tuning for deep reinforcement learning applications. arXiv:2201.11182, 2022.

10 Dong MH, Ying FK, Li XJ, *et al*. Efficient policy learning for general robotic tasks with adaptive dual-memory hindsight experience replay based on deep reinforcement learning. Proceedings of the 7th International Conference on Robotics, Control and Automation (ICRCA). Taizhou: IEEE, 2023. 62–66. [doi: 10.1109/ICRCA57894.2023.10087824]

11 Deb K, Agrawal RB. Simulated binary crossover for continuous search space. Complex Systems, 1995, 9(2): 115–148.

12 Deb K, Sindhya K, Okabe T. Self-adaptive simulated binary crossover for real-parameter optimization. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation. London: ACM, 2007. 1187–1194. [doi: 10.1145/1276958.1277190]

13 程元栋, 杨齐威, 闫俊. 基于混合自适应精英遗传算法的路径规划研究. 湖北民族大学学报 (自然科学版), 2023, 41(1): 51–57, 64. [doi: 10.13501/j.cnki.42-1908/n.2023.03.008]

14 孙波, 姜平, 周根荣, 等. 改进遗传算法在移动机器人路径规划中的应用. 计算机工程与应用, 2019, 55(17): 162–168. [doi: 10.3778/j.issn.1002-8331.1903-0387]

15 李开荣, 胡倩倩. 融合 Bezier 遗传算法的移动机器人路径规划. 扬州大学学报 (自然科学版), 2021, 24(5): 58–64. [doi: 10.19411/j.1007-824x.2021.05.011]

16 Wang ZJ, Zhan ZH, Kwong S, *et al*. Adaptive granularity learning distributed particle swarm optimization for large-scale optimization. IEEE Transactions on Cybernetics, 2021, 51(3): 1175–1188. [doi: 10.1109/TCYB.2020.2977956]

17 Deb K, Pratap A, Agarwal S, *et al*. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182–197. [doi: 10.1109/4235.996017]

18 Hu YM, Li DC, He YQ, *et al*. Incremental learning framework for autonomous robots based on Q-learning and the adaptive kernel linear model. IEEE Transactions on Cognitive and Developmental Systems, 2022, 14(1): 64–74. [doi: 10.1109/TCDS.2019.2962228]

19 李理, 李鸿, 单宁波. 多启发因素改进蚁群算法的路径规划. 计算机工程与应用, 2019, 55(5): 219–225, 250. [doi: 10.3778/j.issn.1002-8331.1805-0175]

20 Miao CW, Chen GZ, Yan CL, *et al*. Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm. Computers & Industrial Engineering, 2021, 156: 107230. [doi: 10.1016/j.cie.2021.107230]

(校对责编: 孙君艳)