文章编号: 1001-0920(2018)10-1757-10 DOI: 10.13195/j.kzyjc.2017.0564

求解N-车探险问题的Memetic烟花算法

刘 翱[1,2,3], 刘凡熙[4], 冯骁毅[5], 邓旭东[1,2], 刘 波[5], 任 亮[1,2,†]

(1. 武汉科技大学 管理学院，武汉 430065；2. 武汉科技大学 服务科学与工程研究中心，武汉 430065；
3. 智能信息处理与实时工业系统湖北省重点实验室，武汉 430065；4. 英国伦敦大学国王学院 刘鸣炜中国研究院，伦敦 WC2R 2LS；5. 中国科学院 数学与系统科学研究院，北京 100190)

摘 要: N-车探险问题是一类在燃油约束下安排N辆车的行驶顺序以使车辆行驶最远的NP-hard问题. 针对该问题,提出一种融合局部搜索的Memetic烟花算法(MFWA). 根据该问题等价于置换排序的特性,设计基于ranked-order value(ROV)规则的编码方式,引入动态爆炸半径,使用烟花算法进行全局搜索;设计插入、交换和反转等邻域操作,增强算法的局部搜索能力;利用实验设计探讨了关键参数对算法性能的影响. 基于14个标准问题的测试结果表明: 所设计的局部搜索操作有助于增强烟花算法在N-车探险问题上的寻优精度;MFWA的寻优精度、稳定性等整体优于(至少不劣于)标准烟花算法(FWA)、已有的启发式算法(H1-H4)、粒子群优化(PSO)和水波优化(WWO);与MFWA相比,禁忌变邻域局部搜索(TBVLS)用至少55倍的计算时间得到了最大竞争比为1.126的寻优精度. 这些结果表明,MFWA能在较短时间内获得较满意的寻优精度.

关键词: N-车探险问题；烟花算法；局部搜索；Memetic算法；启发式算法

中图分类号: TP18 **文献标志码:** A

Memetic fireworks algorithm for solving N-vehicle exploration problem

LIU Ao[1,2,3], *LIU Fan-xi*[4], *FENG Xiao-yi*[5], *DENG Xu-dong*[1,2], *LIU Bo*[5], *REN Liang*[1,2†]

(1. School of Management，Wuhan University of Science and Technology，Wuhan 430065，China；2. Center for Service Science and Engineering，Wuhan University of Science and Technology，Wuhan 430065，China；3. Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System，Wuhan 430065，China；4. Lau China Institute，King's College London，London WC2R 2LS，UK；5. Academy of Mathematics and Systems Science，Chinese Academy of Sciences，Beijing 100190，China.)

Abstract: The N-vehicle exploration problem is a class of NP-hard problem, which aims to schedule a fleet of N vehicles' trip sequence under limited oil to ensure one of the vehicles visit the farthest distance. A Memetic fireworks algorithm(MFWA) with local search enhancement is proposed to solve this discrete optimization problem. According to the characteristic of equivalence to permutation, a ranked-order value(ROV) scheme is adopted to encode the candidate solution, and the fireworks algorithm with dynamic radius of explosion is employed to search the solution space globally. To enhance the local search ability, three neighbor operations by insertion, exchange and reverse are employed to carry out local refinement. The effect of key parameters with regard to the performance is investigated. Numerical results about 14 benchmark instances are provided, and the comparisons suggest that the MFWA is superior to(at least not inferior to) the standard fireworks algorithm(FWA), the existing heuristic algorithms(H1-H4), particle swarm optimization(PSO) and water wave optimization(WWO) in terms of solution accuracy and stability; compared with the MFWA, tabu-based variable neighborhood local search(TBVLS) obtains a maximal competitive ratio of 1.126 at the expense of nearly 55 times computational efforts. These results indicate that the MFWA can achieve satisfactory results in reasonable time.

Keywords: N-vehicle exploration problem; fireworks optimization; local search; Memetic algorithm; heuristic algorithm

收稿日期: 2017-05-08；**修回日期:** 2017-07-25.

基金项目: 国家自然科学基金项目(71701156, 71101139)；教育部人文社会科学研究青年基金项目(16YJCZH056)；湖北省自然科学基金项目(2017CFB427)；湖北省教育厅人文社会科学研究青年项目(17Q034)；湖北省教育厅科学技术研究项目(Q20171104)；武汉科技大学服务科学与工程研究中心开放基金项目(CSSE2017KA01)；智能信息处理与实时工业系统湖北省重点实验室开放基金项目(2016znss18B)；武汉科技大学青年科技骨干培育计划项目(2016xz0l7, 2017xz031).

责任编委: 刘德荣.

作者简介: 刘翱(1987−), 男, 讲师, 博士, 从事智能优化、优化调度等研究；刘凡熙(1990−), 女, 博士生，从事定量研究方法及其应用的研究.

†通讯作者. E-mail: renliang@wust.edu.cn

0 引 言

N-车探险问题来源于吉普探险[1],是一类资源分配问题,它广泛应用于吉普加油[2-3]、航天器排列[4]、沙漠穿行[5]等领域. 已有研究表明,该问题等价于最优置换排序,一般情形下是NP-hard问题[6].

Fine等[1]提出了吉普问题,主要解决单个吉普车在燃油约束下如何在沙漠中穿行使得往返距离最远[2]. 随后,Phipps[3]建立了该问题的动态规划模型,分析了最优解的结构,并给出了某些特殊情况下的最优解的单调性. Franklin等[4]证明了吉普探险问题与航天器排列问题之间的等价性. Gale[5]、Cao等[6]和Hausrath等[7]推广并研究了具有N辆车的吉普问题,利用归纳法研究了吉普问题的最优序列及其结构性质. Imada[8]则研究了2维吉普问题,并探讨了learning robot自动求解该问题的可行性. Kuwata等[9]在吉普问题基础上提出了一类新的探索问题,分析了该问题的计算复杂性和近似算法. 李晓亚等[10]得到了最优车辆行驶顺序的必要条件,并设计了$O(n^2)$的多项式时间算法.

鉴于直接精确求解的复杂性,对于一般情形的N-车探险问题,研究人员主要关注如何设计高效的近似算法和启发式算法求解该问题. 例如,Li等[11]研究了N-车探险问题的最优结构,并提出了两种实时贪婪启发式算法. 徐扬扬等[12]建立了N-车探险问题的非线性混合整数规划模型,在此基础上提出了ϵ-近似算法. 更进一步,Xu等[13-14]研究了一类多任务N-车探险问题的算法复杂度,结果表明该类问题为NPC问题. 李晓亚[15]将N-车探险问题转换为动态规划问题,提出了一种基于启发式解的rollout算法. Wang等[16]建立了混合整数规划模型与N-车探险问题的等价性. Liu等[17-18]分别利用贪婪算法、禁忌搜索等方法求解该问题,并取得了较好的效果. 然而,目前关于N-车探险问题的研究大都集中在复杂度分析、贪婪算法、近似算法等,尚未发现应用群智能优化算法求解N-车探险问题的研究工作.

烟花算法(FWA)是受烟花爆炸产生火花的启发而设计的一种群体协同优化算法,具有操作简单、易于实现、全局搜索能力强等优点[19]. 已有的实验结果表明,烟花算法在11个测试函数上的性能优于标准PSO和克隆PSO等[19-20],先后应用到作物施肥[21]、配电网重构[22]、桁架结构设计[23]、旅行商问题[24]、卫星调度[25]和生产调度[26]等领域. 具体而言,Tan等[24]研究了求解经典TSP问题的离散烟花算法,该算法充分结合TSP的问题特征,设计了2-opt、3-opt、2h-opt等邻域操作,增强其局部搜索能力. Liu等[25]针对卫星调度问题,设计了基于插入、交换、反转等邻域操作增强烟花算法的局部搜索能力. 曹磊等[26]利用最大位置法对置换流水车间调度问题的解进行编码,设计了锦标赛选择策略和混沌搜索策略增强算法跳出局部最优的能力. 从已有研究来看,应用烟花算法求解离散优化尤其是N-车探险问题的研究工作相对较少;进一步,为有效求解离散优化问题,需结合问题特征设计邻域操作以增强烟花算法的局部搜索能力.

鉴于烟花算法的优点及其离散优化的研究工作较少,本文提出一种融合局部搜索的Memetic烟花算法求解N-车探险问题. 该算法分为两个阶段:

1) 全局搜索阶段,该阶段对问题进行ROV编码,引入动态爆炸半径,使用烟花算法进行全局搜索;

2) 局部搜索阶段,该阶段结合插入、交换和反转等邻域操作进行局部搜索,以增强算法的局部搜索能力,从而获得更优解.

值得一提的是,本文首次尝试应用烟花算法求解N-车探险问题.

1 N-车探险问题

N-车探险问题是指: 假定车辆$1, 2, \cdots, n$携带燃油$a_1, a_2, \cdots, a_n$,单位耗油量为$b_1, b_2, \cdots, b_n$. 所有车辆从$S_0 = 0$出发同向直线行驶,在行驶途中无法从外界加油,但可以在途中停止等待,并将燃油转移给其他车辆. 问如何安排车辆行驶顺序,使得其中一辆行驶最远,且所有车都回到出发点S_0.

参见图1,定义行驶顺序$\pi_1 \Rightarrow \pi_2 \Rightarrow \cdots \Rightarrow \pi_n$,$\pi_i \Rightarrow \pi_j (i < j)$表示前面的车$\pi_i$给后面的车$\pi_j$供油.

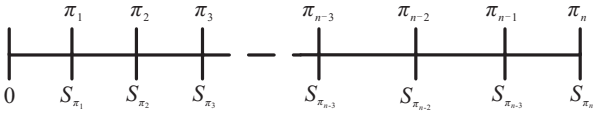

图1 车辆行驶序列$\pi_1 \Rightarrow \pi_2 \Rightarrow \cdots \Rightarrow \pi_n$[13]

记车辆π_i的行驶距离为S_{π_i},定义$S_{\pi_0} = 0, x_{\pi_i} = S_{\pi_i} - S_{\pi_{i-1}}, i = 1, 2, \cdots, n$,则$x_{\pi_i}$满足

$$\begin{cases} 2x_{\pi_n} b_{\pi_n} = a_{\pi_n}, \\ 2x_{\pi_{n-1}}(b_{\pi_{n-1}} + b_{\pi_n}) + 2x_{\pi_n} b_{\pi_n} = a_{\pi_{n-1}} + a_{\pi_n}, \\ \quad \vdots \\ \displaystyle\sum_{i=1}^{n} 2x_{\pi_i} \sum_{j=i}^{n} b_{\pi_j} = \sum_{i=1}^{n} a_{\pi_i}. \end{cases} \tag{1}$$

将式(1)化简,可知最远行驶距离$S_\pi = S_{\pi_n}$满足

$$S_\pi = \frac{1}{2}\Big(\frac{a_{\pi_1}}{\sum\limits_{i=1}^{n} b_{\pi_i}} + \frac{a_{\pi_2}}{\sum\limits_{i=2}^{n} b_{\pi_i}} + \cdots + \frac{a_{\pi_n}}{b_{\pi_n}}\Big). \quad (2)$$

因此,N-车探险问题等价于最优置换排列问题:给定数列$(a_1,b_1),(a_2,b_2),\cdots,(a_n,b_n)$,寻找$1,2,\cdots,n$的最优置换序列$\pi = (\pi_1,\pi_2,\cdots,\pi_n)$,使得式(2)中$S_\pi$的取值达到最大.

显然,N-车探险问题的搜索空间大小为$O(n!)$.对于小规模问题,可使用暴力穷举法得到最优解;而对于大规模问题,则需要设计高效的近似算法[17-18].

2　求解N-车探险问题的Memetic烟花算法

本节首先介绍烟花算法;然后给出Memetic烟花算法的编码策略、动态爆炸半径和局部搜索;最后介绍Memetic烟花算法流程.

2.1　烟花算法概述

烟花算法[19]是将每个烟花视为一个可行解,通过爆炸操作对每个烟花在不同的爆炸半径内产生不同数量的火花实现邻域内的局部搜索,并通过调整爆炸半径和火花数目实现全局探索和局部搜索的平衡. 烟花算法包括爆炸算子、变异算子和选择策略.

2.1.1　爆炸算子

记烟花种群为$X = [X_1, X_2, \cdots, X_N]$,$N$为种群规模,$X_i$为烟花$i$的位置,适应度值为$f(X_i)$. 烟花$X_i$通过爆炸算子在爆炸半径$A_i$内产生$S_i$个火花后的位置为

$$Y_i = X_i + A_i \times U \times \mathrm{rand}(-1,1), \quad (3)$$

其中U为取值0或1的D维随机矩阵,D为变量维数.

以最小化问题为例,根据烟花算法的基本思想,适应度好的烟花在较小范围内聚集较多的火花,适应度差的烟花在较大范围内分布较少的火花. 因而,烟花X_i的爆炸半径A_i和火花个数S_i满足

$$\begin{cases} A_i = A_{\max}\dfrac{f(X_i) - \min\limits_{1\leqslant k\leqslant N} f(X_k) + \epsilon}{\sum\limits_{i=1}^{N}(f(X_i) - \min\limits_{1\leqslant k\leqslant N} f(X_k)) + \epsilon}; \\ S_i = N_s\dfrac{\max\limits_{1\leqslant k\leqslant N} f(X_k) - f(X_i) + \epsilon}{\sum\limits_{i=1}^{N}(\max\limits_{1\leqslant k\leqslant N} f(X_k) - f(X_i)) + \epsilon}. \end{cases} \quad (4)$$

其中:$A_{\max}, N_s$分别为最大爆炸半径和火花总数,ϵ是为避免分母为0而设置的一个很小的常数.

对于最大化问题,将式(4)修正为

$$\begin{cases} A_i = A_{\max}\dfrac{\max\limits_{1\leqslant k\leqslant N} f(X_k) - f(X_i) + \epsilon}{\sum\limits_{i=1}^{N}(\max\limits_{1\leqslant k\leqslant N} f(X_k) - f(X_i)) + \epsilon}; \\ S_i = N_s\dfrac{f(X_i) - \min\limits_{1\leqslant k\leqslant N} f(X_k) + \epsilon}{\sum\limits_{i=1}^{N}(f(X_i) - \min\limits_{1\leqslant k\leqslant N} f(X_k)) + \epsilon}. \end{cases} \quad (5)$$

为控制爆炸产生过多或者过少的火花个数,引入如下机制对火花个数进行控制:

$$S_i = \begin{cases} \mathrm{round}(aN_s),\ S_i < aN_s; \\ \mathrm{round}(bN_s),\ S_i > bN_s; \\ \mathrm{round}(S_i),\ aN_s \leqslant S_i \leqslant bN_s. \end{cases} \quad (6)$$

其中:a、b为$[0,1]$的常数,round为四舍五入取整函数.

2.1.2　变异算子

烟花算法采取变异算子进行高斯变异,以增强烟花种群的多样性. 在烟花种群中随机选取N_g个烟花,按下式随机选择若干维数进行高斯变异:

$$Z_i = X_i U N(1,1). \quad (7)$$

其中:U为取值0或1的D维随机矩阵,D为变量维数;$N(1,1)$为均值和方差均为1的高斯随机数;Z_i为烟花i变异后的位置.

2.1.3　选择策略

由烟花、爆炸火花和高斯变异火花共同组成候选集合K,从中选取N个烟花个体组成下一代烟花种群. 最优烟花个体作为精英保留,其余$N-1$个个体采用轮盘赌的方式选择,其被选择的概率为

$$P(X_i) = \frac{\sum\limits_{j=1}^{K} \| X_i - X_j \|}{\sum\limits_{i=1}^{K}\sum\limits_{j=1}^{K} \| X_i - X_j \|}. \quad (8)$$

2.2　Memetic烟花算法

2.2.1　ROV编码

标准烟花算法用于求解连续优化问题,而N-车探险问题的解空间是全部置换序列. 鉴于此,借鉴置换流水车间调度问题的编码策略,采取基于随机键的ROV规则对N-车探险问题的解进行编码[27].

如表1所示,对于连续变量$X = (x_1, x_2, \cdots, x_n)$,其升序变量$Y$满足$y_1 \leqslant y_2 \leqslant \cdots \leqslant y_n$. 对于第$i$维变量$x_i$,找到$j$,满足$x_i = y_j$,并令$\pi_i = j$,$\pi = (\pi_1, \pi_2, \cdots, \pi_n)$即为置换序列.

在表1中,$X = (0.7, 0.2, 0.5)$的升序向量为$Y =$

表1 ROV编码的一个例子[27]

j	1	2	3
X	0.7	0.2	0.5
π	3	1	2

$(0.2,0.5,0.7)$,满足$x_1=y_3,x_2=y_1,x_3=y_2$,可得到置换序列$\pi_1=3,\pi_2=1,\pi_3=2$,即置换序列为$\pi=(3,1,2)$.

对于置换序列$\pi=(\pi_1,\pi_2,\cdots,\pi_n)$,采用如下方式将其转换为连续变量$X=(x_1,x_2,\cdots,x_n)$[27]:

$$x_i=X_{\min}+(X_{\max}-X_{\min})\frac{(\pi_i-1)}{n},\tag{9}$$

其中$X_{\min},X_{\max}$为变量X的下界和上界.

2.2.2 指数下降的动态爆炸半径

根据式(5),适应度高的烟花具有较小的爆炸半径和较大的火花个数. 对于最优烟花,其爆炸半径$\propto\epsilon$,即产生最多火花的最优烟花的爆炸半径几乎为0,产生大量相似的火花,浪费了搜索机会.

直观上看,爆炸半径$A_{\max}$过大,不利于算法的快速收敛;爆炸半径$A_{\max}$过小,则难以更好地执行全局探索. 合理的爆炸半径应满足:

1)进化早期,较大的爆炸半径能更好地执行全局探索;

2)进化后期,较小的爆炸半径促使算法在小范围内进行局部开发.

鉴于此,本文提出如下动态爆炸半径$A(t)$,以增强全局探索和局部开发的平衡能力:

$$A(t)=A_{\max}\left(\frac{A_{\min}}{A_{\max}}\right)^{t/T}.\tag{10}$$

烟花i的爆炸半径更新为

$$A_i=\begin{cases}A(t),\ f(X_i)=\min\limits_{1\leqslant k\leqslant N}f(X_k);\\ A(t)\dfrac{f(X_i)-\min\limits_{1\leqslant k\leqslant N}f(X_k)+\epsilon}{\sum\limits_{i=1}^{N}(f(X_i)-\min\limits_{1\leqslant k\leqslant N}f(X_k))+\epsilon},\ \text{else.}\end{cases}\tag{11}$$

其中t和T分别为当前迭代次数和最大迭代次数.

2.2.3 局部搜索

借鉴文献[28]中Memetic算法的全局搜索和局部搜索互补增强思想,本文引入图2中的3种邻域操作来增强烟花算法的局部搜索能力[26-27,29].

1) 交换:随机选取两个位置i和j,位置互换;

2) 插入:随机选取两个位置i和j,将j插入到i前面;

3) 反转:随机选取两个位置i和j,将序列$i,i+1,\cdots,j$反转.

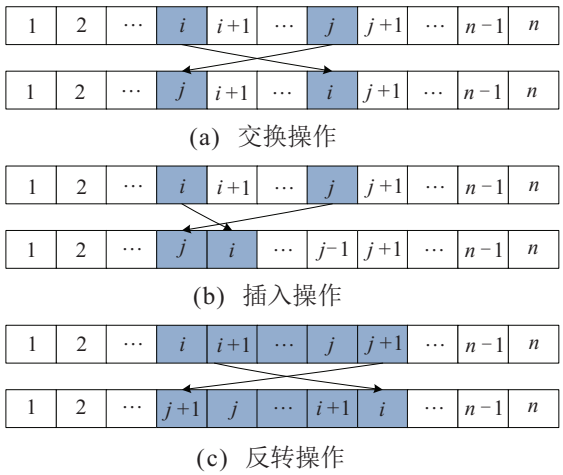

图2 3种邻域搜索操作

2.2.4 Memetic烟花算法流程

鉴于上述设计,本节给出求解N-车探险问题的Memetic烟花算法流程. 与标准烟花算法相比,结合N-车探险问题与最优置换排序等价的特性,该算法增加了ROV编码策略、动态爆炸半径和基于插入、交换、反转等邻域结构的局部搜索,以期增强算法的搜索能力.

具体而言,Memetic烟花算法的步骤如下:

1) 在全局搜索阶段,利用ROV编码规则,引入动态爆炸半径,使用烟花算法进行全局搜索,同时得到该阶段的最优解X_{best};

2) 在局部搜索阶段,从插入、交换和反转中随机选取一个邻域操作对X_{best}进行局部搜索,得到X_{local};

3) 比较$f(X_{\text{best}})$和$f(X_{\text{local}})$,保留其中的最优解;

4) 重复1)~3),直到满足算法终止条件.

2.2.5 Memetic烟花算法的时间复杂度分析

假定种群规模为N,爆炸火花个数为N_s,变异火花个数为N_g,最大迭代次数为T.

Memetic烟花算法包括初始化、全局搜索、局部搜索和输出4个阶段.其中,在T次迭代过程中每次都需执行全局搜索和局部搜索.

1) 初始化阶段涉及N个烟花个体的生成和适应度值计算,其时间复杂度为$O(N)$;

2) 全局搜索阶段包括爆炸、变异和选择操作,时间复杂度分别为$O(N\times N_s)$、$O(N_g)$、$O((N+N_s+N_g)\lg(N+N_s+N_g))$. 因而,全局搜索阶段的时间复杂度为$O((N+N_s+N_g)\lg(N+N_s+N_g))$;

3) 局部搜索阶段的时间复杂度为$O(1)$;

4) 输出阶段涉及到N个烟花个体的排序,其时间复杂度为$O(N\lg N)$.

因此, Memetic烟花算法的时间复杂度为

$$O(N)+(O(N+N_s+N_g)\lg(N+N_s+N_g)+O(1))T+O(N\lg N)=O((N+N_s+N_g)\lg(N+N_s+N_g)T).$$

当N_s、N_g与N成线性关系时, Memetic烟花算法的时间复杂度为$O(N(\lg N)T)$.

3　仿真实验分析

下面利用文献[15,18]的标准测试集进行算法性能测试, 并与现有的启发式算法和优化算法进行比较. 该测试集包括14个标准问题, 车辆数目为9 ~ 14.

仿真环境设置为: Window 7, CPU奔腾T4400, 主频2.2 GHz, 内存2 G, 编程语言Matlab 2014b.

3.1　算法参数设置

烟花算法的主要参数包括种群规模N, 爆炸火花个数N_s, 变异火花个数N_g, 爆炸半径$A_{\max}$. 参照文献[26, 30], 针对车辆数目最多(19辆车)的标准问题13开展实验设计(DOE), 探讨参数对算法性能的影响. 根据正交表$L_9(3^4)$, 如表2所示, 每个参数设置3个水平, 对每组参数组合独立运行20次, 以平均性能作为响应变量(RV), 结果参见表3.

表 2　参数水平取值

参数	水平		
	1	2	3
N	10	15	20
N_s	10	20	30
N_g	5	10	15
$A_{\max}$	1	5	9

表 3　正交试验表及其RV值

编号	水平				RV
	N	N_s	N_g	$A_{\max}$	
1	1	1	1	1	43.619 25
2	1	2	2	2	43.537 29
3	1	3	3	3	42.836 07
4	2	1	2	3	42.777 59
5	2	2	3	1	42.791 28
6	2	3	1	2	42.350 56
7	3	1	3	2	43.972 04
8	3	2	1	3	42.981 24
9	3	3	2	1	43.495 84

表3和表4分别计算了各参数的RV值及其对算法性能影响的极差和等级. 由表4可见, 种群规模N对算法性能影响最大; 其次是爆炸火花个数和爆炸半径, 这二者影响了算法的全局搜索能力; 最后, 变异火花个数对性能影响最小, 这说明变异操作不能起到很好的局部搜索作用, 需引进更有效的邻域操作增强算法的局部搜索能力.

表 4　参数性能分析

水平	N	N_s	N_g	$A_{\max}$
1	43.330 87	**43.456 29**	42.983 69	**43.302 12**
2	42.639 81	43.103 27	**43.270 24**	43.286 63
3	**43.483 04**	42.894 16	43.199 80	42.864 97
极差	0.843 23	0.562 14	0.286 55	0.437 16
等级	1	2	4	3

基于上述分析, 在后续实验中算法参数设置为: $N=20, N_s=10, N_g=10, A_{\max}=1$; 其余参数依次设置为$a=0.1, b=0.9, A_{\min}=0.001$.

3.2　结果比较与分析

下面将从动态半径策略对比、局部搜索有效性分析、与启发式算法的平均性能对比、与其他优化算法的平均性能对比和统计对比分析等不同角度进行分析, 以验证应用Memetic算法求解N-车探险问题的可行性和有效性. 算法在每个问题上独立运行20次, 评价次数达到10 000时, 算法终止.

3.2.1　动态半径策略对比分析

为验证动态半径策略对算法搜索性能和搜索效率的影响, 本节对比分析FWA、edMFWA、MFWA、dynFWA[31]等4种算法的寻优精度和计算时间, 其结果汇总分别见表5和表6.

在表5中, FWA表示标准烟花算法, edFWA表示在FWA中采用公式(11)所示的指数下降的动态爆炸半径, dynFWA是文献[31]提出的另一种基于自适应动态半径的烟花算法, MFWA则是在FWA中同时加入指数下降的动态爆炸半径和局部搜索. 每个标准问题的最优结果用粗体标注, 次优值用下划线标注.

由表5可以看出:

1) 对于FWA而言, FWA在6个问题上的寻优精度比dynFWA要差, 在11个问题上的寻优精度比edFWA要差, 这说明加入自适应动态半径, dynFWA的寻优精度没有明显改善; 与FWA相比, 本文提出的基于指数下降的动态爆炸半径的edFWA有助于改进FWA的寻优精度, 增强FWA全局搜索能力.

2) 文献[31]提出的dynFWA在烟花算法中引入自适应动态半径策略, edFWA引入了指数下降的动态爆炸半径策略. dynFWA分别在9/1/4个问题上的寻优精度劣于/等于/优于edFWA. 这表明, 对于应用烟花算法求解N-车探险问题而言, 指数下降的动态爆炸半径策略比自适应动态半径策略更有效.

表 5 FWA、dynFWA、edFWA 和 MFWA 的统计结果

编号	平均值				标准差			
	FWA	dynFWA	edFWA	MFWA	FWA	dynFWA	edFWA	MFWA
1-10	15.403 95	15.409 59	15.418 68	**15.426 67**	0.065 13	0.035 04	0.077 72	**0.033 49**
2-10	3.591 05	3.591 56	3.591 56	**3.595 73**	0.006 85	0.006 18	0.006 09	**0.003 49**
3-10	10.753 98	10.833 15	10.754 71	**10.875 43**	0.176 09	0.170 99	**0.098 64**	0.122 80
4-9	11.778 72	11.628 70	11.707 13	**11.791 20**	0.129 61	0.178 08	0.137 87	**0.116 53**
5-10	12.326 73	12.317 68	**12.503 25**	12.494 01	0.230 60	**0.141 79**	0.201 69	0.145 02
6-10	11.868 53	11.848 22	11.893 26	**11.938 55**	**0.091 32**	0.187 47	0.125 52	0.091 44
7-10	**12.504 29**	12.425 51	12.425 35	12.466 32	0.198 59	**0.149 16**	0.184 15	0.157 83
8-10	9.325 88	9.324 92	9.341 23	**9.369 82**	0.057 35	0.063 77	0.063 10	**0.04940**
9-10	17.981 07	17.905 74	17.997 95	**18.140 61**	0.222 25	0.200 43	0.289 72	**0.118 55**
10-10	15.447 84	15.484 00	15.484 16	**15.518 30**	0.127 72	0.108 72	**0.087 52**	0.101 06
11-10	41.190 65	41.203 02	41.109 54	**41.335 44**	0.116 18	0.204 56	0.190 10	**0.115 15**
12-14	15.007 63	14.970 24	15.061 25	**15.109 63**	0.116 38	**0.100 61**	0.119 91	0.103 50
13-19	41.314 30	40.684 41	41.672 51	**42.925 22**	1.317 92	**1.174 39**	1.697 17	1.760 14
14-10	2.392 58	2.394 18	2.393 65	**2.396 45**	0.004 39	0.004 30	**0.002 83**	0.003 52

表 6 FWA、dynFWA、edFWA 和 MFWA 的计算时间

编号	FWA	dynFWA [31]	edFWA	MFWA
1-10	0.223 58	0.776 18	0.224 68	0.275 44
2-10	0.216 07	0.754 66	0.216 38	0.279 41
3-10	0.216 80	0.765 87	0.217 48	0.275 54
4-9	0.216 89	0.747 90	0.214 81	0.266 20
5-10	0.221 91	0.773 55	0.226 75	0.270 67
6-10	0.217 43	0.775 33	0.216 52	0.278 56
7-10	0.225 51	0.772 74	0.236 49	0.261 51
8-10	0.228 37	0.816 79	0.216 96	0.273 11
9-10	0.226 29	0.831 33	0.224 86	0.274 40
10-10	0.230 75	0.803 46	0.224 52	0.263 64
11-10	0.229 74	0.785 12	0.232 87	0.262 89
12-14	0.247 86	0.844 77	0.253 14	0.314 18
13-19	0.259 93	0.902 76	0.267 78	0.298 20
14-10	0.222 27	0.780 70	0.221 54	0.295 62

3) 尽管 FWA、dynFWA、edFWA 在 14 个标准问题上的标准差各不相同, 但平均来看, 它们在同一标准问题上的标准差基本在同一个数量级.

表 6 汇总了 FWA、dynFWA、edFWA 和 MFWA 求解 14 个标准问题的计算时间, 该时间用同一实验环境下不同算法程序运行的 CPU 时间来衡量.

由表 6 可以看出, 与 FWA、edFWA、MFWA 相比, dynFWA 在 14 个标准问题上的平均计算时间是它们的 3 倍左右. 这表明, dynFWA 的寻优效率相对较低.

图 3 给出了 FWA、dynFWA、edFWA 和 MFWA 在问题 10 上 20 次运行的平均值迭代曲线.

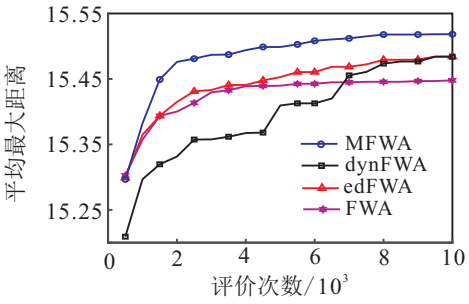

图 3 问题 10 的迭代曲线

由图 3 可以看出:

1) 随着迭代次数增加, FWA、dynFWA、edFWA 和 MFWA 的解越来越优;

2) MFWA 的寻优精度最优, dynFWA 和 edFWA 次之, edFWA 的精度略优于 dynFWA, FWA 的精度最弱;

3) 由于引入了插入、交换和反转等邻域操作, MFWA 在 FWA、dynFWA 和 edFWA 搜索停滞后仍能找到更优解, 表明 MFWA 具有更好的寻优能力.

3.2.2 局部搜索有效性分析

为验证交换、插入、反向等邻域操作对烟花算法的局部增强效果, 本节对比 FWA、edFWA(有动态半径, 无邻域操作) 和 MFWA(动态半径和邻域操作) 的平均值和标准差. 由表 5 和表 6 可知:

1) edFWA 在 11 个问题上的平均值优于 FWA, 在同一问题的标准差数量级相同, 表明动态半径策略有助于增强 FWA 的全局搜索能力.

2) MFWA 在 11 个问题上的平均值优于 edFWA,

且MFWA在10个问题上标准差更小. MFWA在edFWA的基础上引入了插入、变换、反转等3种邻域操作. 这说明: 邻域操作能够增强edFWA的局部搜索能力, 且具有更好的稳定性.

3) 由表6可以看出, MFWA在14个标准问题上的平均计算时间大于FWA和edFWA, 但基本都在同一数量级. 这表明, 与FWA和edFWA相比, 加入局部搜索策略会使得MFWA的计算时间略有增加, 但是并没有显著降低MFWA的搜索效率.

3.2.3 与启发式算法的平均性能对比

为进一步验证MFWA的寻优性能, 本节将对比分析MFWA与4个启发式算法H1～H4对N-车弹性问题的求解效果. 这4个启发式算法来源于文献[15], 分别为: H1-基于Greedy启发式的rollout算法, H2-基于Index启发式的rollout算法, H3-Greedy启发式算法, H4-Index启发式算法. 表7和表8分别汇总了MFWA、H1～H4在14个标准问题上的实验结果和计算时间. 其中, 每个问题的最优结果用粗体标注, 次优值用下划线标注.

表7　MFWA和H1～H4的结果汇总

编号	MFWA	H1[15]	H2[15]	H3[15]	H4[15]
1-10	15.426 67	15.316 62	**15.476 61**	15.131 96	15.133 28
2-10	**3.595 73**	3.511 29	3.564 98	3.027 15	3.027 52
3-10	10.875 43	**10.967 99**	10.806 81	10.806 81	10.806 81
4-9	**11.791 20**	11.753 22	11.564 32	10.062 64	10.062 64
5-10	**12.494 01**	12.303 55	12.392 84	10.516 17	10.516 17
6-10	**11.938 55**	11.389 57	11.421 51	10.206 27	10.208 14
7-10	12.466 32	**12.476 30**	12.369 68	10.688 93	10.688 93
8-10	9.369 82	9.353 99	**9.429 28**	7.666 73	7.656 13
9-10	18.140 61	**18.230 30**	17.752 12	17.752 12	17.773 83
10-10	**15.518 30**	15.021 76	15.021 76	14.639 87	14.672 11
11-10	**41.335 44**	39.614 05	39.614 05	39.614 05	39.614 05
12-14	15.109 63	14.899 75	**15.248 06**	14.217 07	14.217 07
13-19	42.925 22	45.179 68	**46.504 86**	41.616 26	41.925 80
14-10	**2.396 45**	2.340 86	2.376 66	2.018 10	2.018 35

表8　MFWA和H1～H4的计算时间

编号	MFWA	H1[15]	H2[15]	H3[15]	H4[15]
1-10	0.275 44	0.266 34	0.247 91	0.054 40	0.020 50
2-10	0.279 41	0.166 90	0.165 20	0.015 39	0.010 86
3-10	0.275 54	0.156 25	0.163 46	0.012 28	0.009 98
4-9	0.266 20	0.119 62	0.115 95	0.010 61	0.010 72
5-10	0.270 67	0.159 17	0.158 92	0.016 00	0.008 96
6-10	0.278 56	0.161 44	0.157 03	0.011 97	0.009 76
7-10	0.261 51	0.150 69	0.150 84	0.011 86	0.008 57
8-10	0.273 11	0.161 60	0.161 73	0.012 22	0.010 19
9-10	0.274 40	0.155 57	0.153 31	0.014 89	0.009 99
10-10	0.263 64	0.173 55	0.171 59	0.011 23	0.008 16
11-10	0.262 89	0.157 04	0.181 07	0.010 89	0.007 86
12-14	0.314 18	0.387 37	0.403 78	0.025 44	0.010 52
13-19	0.298 20	0.893 04	0.898 89	0.046 65	0.012 87
14-10	0.295 62	0.172 32	0.162 94	0.013 37	0.009 90

由表7和表8可以看出:

1) MFWA、H1、H2分别在7、3、4个问题上取得最优值, 并且, MFWA在5个问题上取得次优值. 整体上看, MFWA的寻优精度优于(至少不劣于)这些启发式算法, 初步验证了MFWA具有优良的寻优精度. 可以预期, 通过精细调整合适的算法参数或结合高效的搜索机制, MFWA的搜索性能将得到进一步改进.

2) 从计算时间看, MFWA在14个标准问题上的平均计算时间约为H1和H2的2倍, 约为H3和H4的10倍. 表明H1～H4具有更高的寻优效率, 后续可以考虑结合MFWA的寻优精度和H1～H4的寻优效率, 开发更优的高效优化算法.

3.2.4 与PSO、WWO和TBVLS的平均性能对比

本节对比分析MFWA与粒子群优化(PSO[27])、禁忌变邻域局部搜索(TBVLS[18])和水波优化(WWO[32])的寻优精度和寻优效率. 表9和表10汇总了4个优化算法在14个标准问题上的平均值和计算时间. 在表9中, 每个标准问题的最优结果用粗体标注, 次优值用下划线标注.

表9　MFWA、PSO、WWO和TBVLS的结果汇总

编号	MFWA	PSO[27]		WWO[32]		TBVLS[18]	
	平均值	平均值	竞争比	平均值	竞争比	平均值	竞争比
1-10	15.403 95	12.501 26	0.812	13.693 15	0.889	**15.476 61**	1.005
2-10	3.591 05	2.997 88	0.835	3.54922	0.988	**3.603 31**	1.003
3-10	10.753 98	3.471 94	0.323	5.700 12	0.530	**11.023 60**	1.025
4-9	11.778 72	5.894 52	0.500	10.075 10	0.855	**11.946 62**	1.014
5-10	12.326 73	6.070 89	0.492	10.258 88	0.832	**12.722 98**	1.032
6-10	11.868 53	6.024 76	0.508	10.220 60	0.861	**12.124 01**	1.022
7-10	12.504 29	6.111 21	0.489	7.040 61	0.563	**12.769 68**	1.021
8-10	9.325 88	5.643 17	0.605	9.077 32	0.973	**9.429 28**	1.011
9-10	17.981 07	6.372 34	0.354	9.523 91	0.530	**18.277 18**	1.016
10-10	15.447 84	8.632 86	0.559	14.509 15	0.939	**15.656 69**	1.014
11-10	41.190 65	30.095 05	0.731	40.298 98	0.978	**41.451 98**	1.006
12-14	15.007 63	9.960 66	0.664	14.193 26	0.946	**15.248 06**	1.016
13-19	41.314 30	8.172 38	0.198	11.973 09	0.290	**46.504 86**	1.126
14-10	2.392 58	2.004 93	0.838	2.366 145	0.989	**2.402 208**	1.004

表 10 MFWA、PSO、WWO和TBVLS的计算时间

编号	MFWA	PSO[27]		WWO[32]		TBVLS[18]	
	平均值	平均值	竞争比	平均值	竞争比	平均值	竞争比
1-10	0.275 44	0.468 78	1.702	2.258 13	8.198	19.010 99	69.020
2-10	0.279 41	0.459 84	1.646	2.210 10	7.910	18.870 58	67.537
3-10	0.275 54	0.468 88	1.702	2.271 55	8.244	18.772 26	68.129
4-9	0.266 20	0.455 68	1.712	2.205 02	8.283	14.722 38	55.306
5-10	0.270 67	0.450 52	1.664	2.176 38	8.041	19.055 25	70.399
6-10	0.278 56	0.468 27	1.681	2.207 03	7.923	18.450 17	66.234
7-10	0.261 51	0.456 14	1.744	2.163 61	8.274	18.232 40	69.720
8-10	0.273 11	0.469 06	1.717	2.220 87	8.132	18.692 35	68.443
9-10	0.274 40	0.454 32	1.656	2.219 19	8.088	18.639 46	67.929
10-10	0.263 64	0.449 71	1.706	2.204 14	8.361	18.685 92	70.878
11-10	0.262 89	0.452 62	1.722	2.182 39	8.302	19.119 53	72.728
12-14	0.314 18	0.540 23	1.720	2.410 54	7.673	40.027 85	127.406
13-19	0.298 20	0.459 32	1.540	2.229 03	7.475	186.487 06	625.373
14-10	0.295 62	0.458 53	1.551	2.195 50	7.427	18.623 23	62.997

表9和表10中的竞争比定义为该算法的平均值与MFWA的平均值之比. 例如,在问题1中,对PSO算法而言,平均距离的竞争比为12.501 26/15.403 95 = 0.812,平均计算时间的竞争比为0.468 78/0.275 44 = 1.702.

由表9和表10可以看出:

1) 在14个标准问题上,TBVLS具有最好的寻优精度,MFWA次之,WWO再次之,PSO最差. 与MFWA相比,在问题13上,TBVLS达到最大竞争比为1.126,表明TBVLS具有更好的寻优精度.

2) 从计算时间上看,PSO耗时最少,WWO次之,MFWA再次之,TBVLS耗时最多. 与MFWA相比,在问题13上,TBVLS的计算时间竞争比达到最大为625.373;在问题9上,TBVLS的竞争比最小为55.306;而MFWA在问题9和问题13上的计算时间分别为0.455 68和0.459 32. 与MFWA相比,TBVLS的寻优效率更低. 考虑到问题9和问题13的规模分别为9和19,这表明随着问题规模的增大,TBVLS的计算时间增加越大,寻优效率会降低;而MFWA具有较高且稳定的寻优效率.

3) MFWA能在较短时间内获得较满意的寻优精度,MFWA能较好地平衡寻优精度和寻优效率.

3.3 统计对比分析

本节对11个算法在14个问题上的寻优精度进行Wilcoxon Signed Rank检验[33],结果参见表11. w/t/l分别统计了MFWA优于/等于/劣于对比算法的次数,R^+, R^- 记录了其正/负符号秩的和.

表11的结果表明:

1) 在显著性水平$\alpha = 0.01$时,MFWA的寻优精度与H1、H2相当,除TBVLS外,MFWA的寻优精度显著优于其他7个算法;

表 11 Wilcoxon Signed Rank检验结果

对比	w/t/l	R^+	R^-	p value
MFWA vs. FWA	13/0/1	100	5	0.003
MFWA vs. dynFWA	14/0/0	105	0	0.001
MFWA vs. edFWA	13/0/1	101	4	0.002
MFWA vs. H1	10/0/4	77	28	0.012
MFWA vs. H2	10/0/4	76	29	0.014
MFWA vs. H3	14/0/0	105	0	0.001
MFWA vs. H4	14/0/0	105	0	0.001
MFWA vs. PSO	14/0/0	105	0	0.001
MFWA vs. WWO	14/0/0	105	0	0.001
MFWA vs. TBVLS	0/0/14	0	105	0.999

2) 在显著性水平$\alpha = 0.05$时,除TBVLS外,MFWA的寻优精度显著优于其他9个算法;

3) 尽管MFWA的寻优精度劣于TBVLS的寻优精度,但MFWA具有更高且稳定的寻优效率.

这些结果进一步验证了MFWA具有较好的寻优精度和较高的寻优效率,也证实了在FWA中引入邻域操作有助于增强算法的局部搜索能力.

4 结 论

N-车探险问题是一类离散优化问题,目前的求解算法以启发式算法为主. 鉴于此,本文提出了一种融合局部搜索的Memetic烟花算法求解该问题. 一方面,基于ROV规则对问题编码,使用带动态爆炸半径的烟花算法进行全局搜索以获得较优的初始解;另一方面,融合交换、插入、反向等邻域操作以增强算法的局部搜索能力. 实验结果揭示了关键参数对算法性能的影响,验证了应用Memetic烟花算法求解N-车探险问题的可行性和有效性.

尽管Memetic烟花算法的性能整体优于(至少不劣于)现有的启发式算法,但是Memetic烟花算法在部分问题上的性能无明显优势,需引入更优的机制来

增强其性能. 进一步的研究工作包括:

1) 设计自适应爆炸、变异和选择算子[20,24];

2) 考虑距离、低碳等多目标N-车探险问题[34-39];

3) 借鉴差分进化[40]、PSO[41]等算法思想, 融合更高效的算法策略, 开发混合Memetic算法[42]求解N-车探险问题;

4) 结合问题特征设计更高效的邻域操作, 开发更高效的离散烟花算法, 求解大规模多任务N-车探险问题[16-18]、航天器排列[43]、分布式车间调度[44-46]、云制造资源组合优化[47]等复杂工程优化问题.

参考文献(References)

[1] Fine N J. The jeep problem[J]. American Mathematical Monthly, 1947, 54(1): 24-31.

[2] Pollack M. The jeep problem: The maximum rate of delivery[J]. J of the Society for Industrial and Applied Mathematics, 1965, 13(4): 1079-1086.

[3] Phipps C G. The jeep problem: A more general solution[J]. American Mathematical Monthly, 1947, 54(8): 458-462.

[4] Franklin J N. The range of a fleet of aircraft[J]. J of the Society for Industrial and Applied Mathematics, 1960, 8(3): 541-548.

[5] Gale D. The jeep once more or jeeper by the dozen[J]. The American Mathematical Monthly, 1970, 77(5): 493-501.

[6] Cao W L, Ding Y M, Fan W T. Optimal logistics for multiple jeeps[J]. Acta Mathematica Scientia, 2010, 30(5): 1429-1439.

[7] Hausrath A, Jackson B, Mitchem J, et al. Gale's round-trip jeep problem[J]. The American Mathematical Monthly, 1995, 102(4): 299-309.

[8] Imada A. Can learning robot solve a 2-D jeep problem[C]. Proc of the Int Symposium on Autonomous Minirobots for Research and Edutainment. Argentina: AMiRE, 2007: 1078-1086.

[9] Kuwata T, Maehara H. Another exploration problem[J]. Discrete Mathematics, 2016, 339(5): 1543-1550.

[10] 李晓亚, 崔晋川. 关于一类N车探险问题的有效算法[J]. 系统工程学报, 2008, 23(4): 444-448.
(Li X Y, Cui J C. Efficient algorithm for kind of exploration problem with N vehicles[J]. J of Systems Engineering, 2008, 23(4): 444-448.)

[11] Li X Y, Cui J C. Real-time algorithm scheme for n-vehicle exploration problem[C]. Combinatorial optimization and Applications. Conf on The 3rd Int, Huangshan: Springer, 2009: 287-300.

[12] 徐扬扬, 崔晋川. N车探险问题的一种ϵ-近似度的近似算法[J]. 应用数学学报, 2009, 32(6): 1036-1043.
(Xu Y Y, Cui J C. A kind of ϵ-approximation algorithm for the exploration problem with N vehicles[J]. Acta Mathematicae Applicatae Sinica, 2009, 32(6): 1036–1043.)

[13] Xu Y Y, Cui J C. Multitask n-vehicle exploration problem: complexity and algorithm[J]. J of Systems Science and Complexity, 2012, 25(6): 1080-1092.

[14] Xu Y Y, Cui J C. A variant of multi-task n-vehicle exploration problem: Maximizing every processor's average profit[J]. Acta Mathematicae Applicatae Sinica, English Series, 2012, 28(3): 463-474.

[15] 李晓亚. N车探险问题的一种Rollout算法[J]. 应用数学学报, 2014, 37(1): 99-108.
(Li X Y. A kind of rollout algorithm for N vehicles exploration problem[J]. Acta Mathematicae Applicatae Sinica, 2014, 37(1): 99–108.)

[16] Wang L L, She B L, Liu J F. A linear mixed integer programming model for N-vehicle exploration problem[J]. J of the Operations Research Society of China, 2015, 3(4): 489-498.

[17] Liu R Y, Cui J C, Song Y. Forward greedy heuristic algorithm for n-vehicle exploration problem[C]. The 8th Int Symposium on Computational Intelligence and Design(ISCID). Hangzhou: IEEE, 2015: 243-246.

[18] Liu A, Deng X D, Tong Z P, et al. A tabu-based variable neighborhood local search for n-vehicles exploration problem[C]. The 12th IEEE Int Conf on Control and Automation. Kathmandu: IEEE, 2016: 951-955.

[19] Tan Y, Zhu Y. Fireworks algorithm for optimization[C]. Int Conf on Advances in Swarm Intelligence. Berlin: Springer, 2010: 355-364.

[20] 朱启兵, 王震宇, 黄敏. 带有引力搜索算子的烟花算法[J]. 控制与决策, 2016, 31(10): 1853-1859.
(Zhu Q B, Wang Z Y, Huang M. Fireworks algorithm with gravitational search operator[J]. Control and Decision, 2016, 31(10): 1853-1859.)

[21] Zheng Y J, Song Q, Chen S Y. Multiobjective fireworks optimization for variable-rate fertilization in oil crop production[J]. Applied Soft Computing, 2013, 13(11): 4253-4263.

[22] Imran A M, Kowsalya M. A new power system reconfiguration scheme for power loss minimization and voltage profile enhancement using fireworks algorithm[J]. Int J of Electrical Power & Energy Systems, 2014, 62(62): 312-322.

[23] Pholdee N, Bureerat S. Comparative performance of meta-heuristic algorithms for mass minimisation of trusses with dynamic constraints[J]. Advances in Engineering Software, 2014, 75(3): 1-13.

[24] Tan Y. Fireworks algorithms: Discrete firework algorithm for combinatorial optimization problem[M]. Berlin: Springer, 2015: 209-226.

[25] Liu Z, Feng Z, Ke L. Fireworks algorithm for the multi-satellite control resource scheduling problem[C]. 2015 IEEE Congress on Evolutionary Computation. Sendai: IEEE, 2015: 1280-1286.

[26] 曹磊, 叶春明, 黄霞. 应用混沌烟花算法求解置换流水车间问题[J]. 计算机应用与软件, 2016, 33(11): 188-192.
(Cao L, Ye C M, Huang X. Applied chaotic fireworks

algorithm in solving permutation flow shop problem[J]. Computer Software and Application, 2016, 33(11): 188-192.)

[27] Liu B, Wang L, Jin Y H. An effective PSO-based memetic algorithm for flow shop scheduling[J]. IEEE Trans on Systems, Man, and Cybernetics, Part B, 2007, 37(1): 18-27.

[28] Ong Y S, Keane A J. Meta-Lamarckian learning in memetic algorithms[J]. IEEE Trans on Evolutionary Computation, 2004, 8(2): 99-110.

[29] 汪恭书,唐立新. 并行机实时调度问题的LR&CG算法[J]. 控制与决策, 2013, 28(6): 829-836.
(Wang G S, Tang L X. LR&CG algorithm for parallel machine real-time scheduling problem[J]. Control and Decision, 2013, 28(6): 829-836.)

[30] 吴楚格, 王凌, 郑晓龙. 求解不相关并行机调度的一种自适应分布估计算法[J]. 控制与决策, 2016, 31(12): 2177-2182.
(Wu C G, Wang L, Zheng X L. An adaptive estimation of distribution algorithm for solving the unrelated parallel machine scheduling[J]. Control and Decision, 2016, 31(12): 2177-2182.)

[31] Zheng S, Janecek A, Li J, et al. Dynamic search in fireworks algorithm[C]. 2014 IEEE Congress on Evolutionary Computation. Beijing: IEEE, 2014: 3222-3229.

[32] Zheng Y J. Water wave optimization: A new nature-inspired metaheuristic[J]. Computers & Operations Research, 2015, 55(3): 1-11.

[33] Zhang H, Li B, Zhang J, et al. Parameter estimation of nonlinear chaotic system by improved TLBO strategy[J]. Soft Computing, 2016, 20(12): 4965-4980.

[34] 刘可, 巩敦卫. 手势分割问题的多目标优化模型及其进化求解方法[J]. 控制与决策, 2017, 32(1): 100-104.
(Liu K, Gong D W. Multi-objective optimization model and its evolution-based solution for gesture segmentation problems[J]. Control and Decision, 2017, 32(1): 100-104)

[35] Liu R, Li J, Fan J, et al. A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization[J]. European J of Operational Research, 2017, 261(3): 1028–1051.

[36] Yang P Y, Tang J F, Yu Y, et al. Minimizing carbon emissions for vehicle routing and scheduling in picking up and delivering customers to airport service[J]. Acta Automatica Sinica, 2013, 39(4): 424-432.

[37] 张瑞友, 张辉, 黄敏. 以低碳为目标的集装箱拖车运输问题及其时间窗离散化算法[J]. 控制与决策, 2016, 31(4): 717-722.
(Zhang R Y, Zhang H, Huang M. Container drayage transportation problem with objective of low carbons and its time window discretization based solution method[J]. Control and Decision, 2016, 31(4): 717-722.)

[38] Qian B, Wang L, Huang D X, et al. An effective hybrid DE-based algorithm for multi-objective flow shop scheduling with limited buffers[J]. Int J of Production Research, 2009, 36(1): 209-233.

[39] Zhu Z, Xiao J, He S, et al. A multi-objective memetic algorithm based on locality-sensitive hashing for one-to-many-to-one dynamic pickup-and-delivery problem[J]. Information Sciences, 2015, 329(2): 73-89.

[40] 刘波, 王凌, 金以慧. 差分进化算法研究进展[J]. 控制与决策, 2007, 22(7): 721-729.
(Liu B, Wang L, Jin Y H. Advances in differential evolution[J]. Control and Decision, 2007, 22(7): 721-729.)

[41] Liang J J, Qin A K, Suganthan P N, et al. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions[J]. IEEE Trans on Evolutionary Computation, 2006, 10(3): 281-295.

[42] Feng L, Ong Y S, Tan A H, et al. Memes as building blocks: A case study on evolutionary optimization + transfer learning for routing problems[J]. Memetic Computing, 2015, 7(3): 159-180.

[43] 李晓亚. 航空器排列问题的最优排序方法研究[J]. 运筹与管理, 2013, 22(5): 24-28.
(Li X Y. Research on optimal sequence of aircraft range problem[J]. Operations Research and Management Science, 2014, 22(5): 24-28.)

[44] 王凌, 王圣尧. 分布式车间调度优化算法研究综述[J]. 控制与决策, 2016, 31(1): 1-11.
(Wang L, Wang S Y. Survey on optimization algorithms for distributed shop scheduling[J]. Control and Decision, 2016, 31(1): 1-11.)

[45] Lu C, Gao L, Li X, et al. Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm[J]. J of Cleaner Production, 2017, 144(2): 228-238.

[46] 张嘉琦, 曹政才, 刘民. 融合代理模型和差分进化算法的并行机动态调度方法[J]. 计算机集成制造系统, 2017, 23(1): 75-81.
(Zhang J Q, Cao Z C, Liu M. Integrated differential evolution algorithm with surrogate model for dynamic parallel machine scheduling[J]. Computer Integrated Manufacturing Systems, 2017, 23(1): 75-81.)

[47] 朱李楠, 王万良, 沈国江. 基于改进差分进化算法的云制造资源优化组合方法[J]. 计算机集成制造系统, 2017, 23(1): 203-214.
(Zhu L N, Wang W L, Shen G J. Resource optimization combination method based on improved differential evolution algorithm for cloud manufacturing[J]. Computer Integrated Manufacturing Systems, 2017, 23(1): 203-214.)

(责任编辑：孙艺红)