带时间窗的车辆路径问题的离散蝙蝠算法

戚远航[1],蔡延光[1],蔡 颢[2],黄何列[1]
(1.广东工业大学自动化学院,广东广州 510006;2.奥尔堡大学健康科学与工程系,丹麦奥尔堡 9220)

摘 要: 本文提出了一种离散蝙蝠算法求解带时间窗的车辆路径问题(vehicle routing problem with time window).该算法提出了蝙蝠位置的定义、速度的定义、位置更新操作、速度更新操作、频率更新操作,并采用惩罚机制与向量比较机制相结合的方法处理相关约束条件.该算法引入了随机插入策略、最少客户车辆插入搜索、普通插入搜索、交换搜索、带时间窗的2-Opt搜索等策略来扩大搜索空间、加强算法的收敛效率.实验结果表明:所提出算法具有较强的寻优能力、较高的鲁棒性、较少的时间耗费;本文所采用的关键参数值和策略能提高所提出算法的性能;通过假设检验证明了所提出算法与对比算法之间的算法性能均有显著性差异.

关键词: 离散蝙蝠算法; 车辆路径问题; 时间窗; 2-Opt
中图分类号: TP301 **文献标识码:** A **文章编号:** 0372-2112 (2018)03-0672-08
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.03.024

Discrete Bat Algorithm for Vehicle Routing Problem with Time Window

QI Yuan-hang[1], CAI Yan-guang[1], CAI Hao[2], HUANG He-lie[1]
(1. *School of Automation, Guangdong University of Technology, Guangzhou, Guangdong* 510006, *China*;
2. *Department of Health Science and Technology, Aalborg University, Aalborg* 9220, *Denmark*)

Abstract: This paper presents a discrete bat algorithm to solve the vehicle routing problem with time window (VRPTW). The proposed algorithm defines position, velocity, updated operation of the position, updated operation of the velocity and updated operation of the frequency, and uses a method which combines the penal function with vectorial comparison to deal with constrained conditions. The proposed algorithm adopts random inserted strategy, inserted research strategy for the vehicle with minimum customers, ordinary inserted research strategy, exchanged research strategy and 2-Opt strategy with time window to expand the search space and enhance the convergent rate. Experimental results show that, the proposed algorithm has a stronger optimization capability, higher robustness and less time consumption, key parameter values and strategies used in this paper can improve performances of the proposed algorithm, and according to the hypotheses testing, there exsits a significant difference between the proposed algorithm and comparative algorithms.

Key words: discrete bat algorithm; vehicle routing problem; time window; 2-Opt

1 引言

带时间窗的车辆路径问题(Vehicle Routing Problem with Time Window, VRPTW)是车辆路径问题(Vehicle Routing Problem, VRP)[1]的经典问题之一.经过多年的研究,学者们应用启发式算法求解VRPTW取得一定的效果.文献[2]提出一种求解VRPTW的离散粒子群算法;文献[3]提出一种混合混沌粒子群算法解决VRPTW,该算法使用混沌算法初始化和扰动粒子群算法,以提高粒子群的局部搜索能力;文献[4]提出一种基于智能体的合作学习算法求解VRPTW.但是,以上诸算法在不同类型的VRPTW算例中求解能力参差不齐,时间耗费较多,收敛效率也较低.

蝙蝠算法(Bat Algorithm, BA)是Yang X S在2010年提出的一种元启发式算法[5].近年来,学者们把蝙蝠算法应用到物流运输调度领域并取得了一定的研究成

收稿日期:2016-09-30; 修回日期:2016-12-21; 责任编辑:覃怀银
基金项目:国家自然科学基金(No. 61074147); 广东省自然科学基金(No. S2011010005059); 广东省教育部产学研结合项目(No. 2012B091000171, No. 2011B090400460); 广东省科技计划项目(No. 2012B050600028, No. 2014B010118004, No. 2016A050502060); 广州市花都区科技计划项目(No. HD14ZD001);广州市科技计划项目(No. 201604016055)

果. 文献[6]提出了改进的离散蝙蝠算法求解对称和非对称旅行商问题;文献[7]提出了基于路径重连的混合蝙蝠算法求解带容量约束的车辆路径问题等. 但是,目前还没有学者把蝙蝠算法应用到 VRPTW 中,而现有的这些离散蝙蝠算法也不能直接运用到 VRPTW 上. 因此,本文提出了一种离散蝙蝠算法(Discrete Bat Algorithm,DBA)求解 VRPTW,拓展了蝙蝠算法在物流运输调度领域的应用,具有一定的创新性、实际意义和较好的推广价值. 实验结果表明:DBA 具有较强的寻优能力、较高的鲁棒性、较少的时间耗费;所采用的关键参数值和策略能提高 DBA 的性能;通过假设检验证明了 DBA 与对比算法之间的算法性能均有显著性差异.

2　数学模型

令 $K=\{1,2,\cdots,m\}$ 为配送中心车辆的集合,$N=\{2,\cdots,n\}$ 为配送中心的客户集合,$V=\{1,N\}$ 为配送中心(定义为"1")和所有客户的集合. V_k 为车辆 k 访问的包含客户集合,$c_{ij}(c_{ij}>0,c_{ii}=0,i,j\in V)$ 为各客户顶点间的距离,D 为车辆最大载重,$q_i(i(V)$ 为各客户的需求载重量,x_{ij}^k 表示车辆 k 服务完客户 i 以后是否服务客户 j($x_{ij}^k=1$ 为服务,$x_{ij}^k=0$ 为不服务),y_i^k 表示客户 i 是否被车辆 k 服务($y_i^k=1$ 为被服务,$y_i^k=0$ 为不被服务),$[a_i,_{bi}]$为客户 i 的时间窗限制,a_i 为客户 i 的允许开始服务时刻,b_i 为客户 i 的终止服务时刻,s_i 为客户 i 的开始服务时刻,则 VRPTW 的数学模型[8]如下:

$$\min\{F_1,F_2\}=\min\Big\{\sum_{j\in V_k}\sum_{k\in K}x_{0j}^k,\sum_{k\in K}\sum_{i\in V_k}\sum_{j\in V_k\backslash|i|}c_{ij}x_{ij}^k\Big\} \tag{1}$$

其中,

$$\sum_{j\in V_k\backslash|i|}x_{ij}^k=y_i^k,\quad \forall i\in V,\quad \forall k\in K \tag{2}$$

$$\sum_{i\in V_k\backslash|j|}x_{ij}^k=y_j^k,\quad \forall j\in V,\quad \forall k\in K \tag{3}$$

$$\sum_{i\in V_k}q_iy_i^k\leqslant D,\quad \forall k\in K \tag{4}$$

$$\sum_{j\in N}x_{0j}^k=\sum_{i\in N}x_{i0}^k=1,\quad \forall k\in K \tag{5}$$

$$\sum_{k\in K}y_i^k=1,\quad \forall i\in N \tag{6}$$

$$\sum_{i,j\in V_k}x_{ij}^k\leqslant|V_k|-1,\quad \forall k\in K \tag{7}$$

$$a_i\leqslant s_i\leqslant b_i,\quad \forall i\in N \tag{8}$$

$$x_{ij}^k\in\{0,1\},\quad \forall i,j\in V,\quad \forall k\in K \tag{9}$$

$$y_i^k\in\{0,1\},\quad \forall i\in N,\quad \forall k\in K \tag{10}$$

式(1)表示优化目标函数,F_1 为车辆数,F_2 为所有车辆的总行驶距离;式(2)表示车辆服务完客户 i 后直接服务客户 j;式(3)表示车辆在服务客户 j 之前只服务一个客户 i;式(4)表示每辆车不得超过最大载重量;式(5)表示车辆从配送中心出发后服务完客户后必须回到配送中心;式(6)表示每个客户只能由一辆车服务;式(7)表示避免服务过程中产生子回路;式(8)表示硬时间窗约束;式(9)和(10)为对决策变量的约束.

3　离散蝙蝠算法

3.1　参数定义与操作设计

3.1.1　蝙蝠位置

设 $Q\in N^+$ 为蝙蝠种群规模,顶点数为 n,车辆数为 m,维数 $w=n+m-2$. 定义第 i 个蝙蝠的位置为 $x_i=(x_{i1},x_{i2},\cdots,x_{iw})$,$i=1,2,\cdots,Q$,其中 x_i 是$(1,2,\cdots,w)$的一个置换.

蝙蝠位置按照规则与 VRPTW 路径形成一一对应关系:$x_{ij}=1$ 和 $x_{ij}>n$ 均认为是配送中心"1";蝙蝠位置的前后分别加上"1",构成一条完整的 VRPTW 路径;2 个"1"之间经过的客户点构成一台车辆的访问路径. 例如:$n=6,m=3,w=7,x_i=(2,3,5,4,7,1,6)$,其对应的路径为$(1,2,3,5,4,1,1,6,1)$. 因此,3 辆车的访问路径分别为$(1,2,3,5,4,1)$、$(1,1)$、$(1,6,1)$,车辆 2 为空车.

3.1.2　蝙蝠速度

定义第 i 个蝙蝠的速度为 $v_i=(v_{i1},v_{i2},\cdots,v_{iw})$. 其中,$1\leqslant v_{ij}\leqslant n,i=1,2,\cdots,Q,j=1,2,\cdots,w$.

3.1.3　蝙蝠位置、速度和频率的更新操作

在连续域中,t 时刻第 i 个蝙蝠的位置和速度分别为 x_i^t、v_i^t,则 $t+1$ 时刻蝙蝠的位置 x_i^{t+1} 和速度 v_i^{t+1} 为[5]:

$$f_i=f_{\min}+(f_{\max}-f_{\min})\beta \tag{11}$$

$$v_i^{t+1}=v_i^t+(x_i^t-x_*)\times f_i \tag{12}$$

$$x_i^{t+1}=x_i^t+v_i^{t+1} \tag{13}$$

其中,f_i 、$f_{\max}$ 、$f_{\min}$分别表示第 i 只蝙蝠当前的频率、频率的最大值、频率的最小值;β 是一个随机变量且 $0\leqslant\beta\leqslant1$;$x_*$ 表示全局最优蝙蝠的位置.

根据式(11)~(13),离散域中蝙蝠位置、速度和频率的更新操作:

(1)$x_i-x_*=v_i^1$:设第 i 个蝙蝠的位置为 $x_i=(x_{i1},x_{i2},\cdots,x_{iw})$,全局最优的蝙蝠的位置为 $x_*=(x_{*1},x_{*2},\cdots,x_{*w})$,$v_i^1=(v_{i1}^1,v_{i2}^1,\cdots,v_{iw}^1)$. 如果 $x_{ij}=x_{*j}$,则 $v_{ij}^1=0$;如果 $x_{ij}\neq x_{*j}$,则 $v_{ij}^1=x_{*j}$. 其中,$j=1,2,\cdots,w$.

(2)$v_i^1\times f_i=v_i^2$:设第 i 个蝙蝠的频率为 f_i,随机生成 $f_r,v_i^2=(v_{i1}^2,v_{i2}^2,\cdots,v_{iw}^2)$. 如果 $f_r<f_i,v_{ij}^2=0$;如果 $f_r\geqslant f_i$,则 $f_i=f_i+(f_r-f_i)/\theta,f_i^{\text{new}}=f_i,v_{ij}^2=v_{ij}^1$. 其中,$f_{\min}<f_i<f_{\max}$,频率因子 $\theta>1,j=1,2,\cdots,w$.

(3)$v_i+v_i^2=v_i^{\text{new}}$:设 $v_{i1}^{\text{new}}=(v_{i1}^{\text{new}},v_{i2}^{\text{new}},\cdots,v_{iw}^{\text{new}})$. 如果 rand() <0.5,则 $v_{ij}^{\text{new}}=v_{ij}$;如果 rand()$\geqslant0.5$,则 $v_{ij}^{\text{new}}=v_{ij}^2$. 其中,$j=1,2,\cdots,w$.

(4)$x_i+v_i^{\text{new}}=x_i^{\text{new}}$:设 $x_{i1}^{\text{new}}=(x_{i1}^{\text{new}},x_{i2}^{\text{new}},\cdots,x_{iw}^{\text{new}})$. 令

$x_i^{new} = x_i$,如果 $v_{ij}^{new} \neq 0$,则把 x_i^{new} 中的第 x_{ij} 位置的分量和第 v_{ij}^{new} 位置的分量交换. 其中,$j = 1,2,\cdots,w$.

得到的 x_i^{new}、v_i^{new}、f_i^{new} 为第 i 个蝙蝠的新的位置、速度和频率,$i = 1,2,\cdots,Q$.

蝙蝠位置、速度和频率的更新操作从连续域到离散域的转换过程:

(1)通过对比 x_i 与 x_* 相同位置上的分量得到两者的差,以此对应连续域中的两个蝙蝠位置的距离差.

(2)通过 f_i 来改变$(x_i - x_*)$得到一个临时速度 v_i^2. 连续域中的 f_i 随机改变的,没有方向性. 在本算法中,f_i 是朝着增大方向变化,得到的 v_i^2 符合本算法的速度定义.

(3)通过随机选择来实现 v_i 和 v_i^2 的融合,以此对应连续域中两个蝙蝠速度的相加. 融合后得到的 v_i^{new} 符合本算法的速度定义.

(4)通过 v_i^{new} 不同位置的分量来改变 x_i,以此对应连续域中蝙蝠速度与位置的相加. 该步骤仅仅对 x_i 不同位置的分量进行交换,并没有增加额外的元素,因此,得到的 x_i^{new} 符合本算法的位置定义,x_i^{new} 仍然对应着一个 VRPTW 的路径.

3.1.4 蝙蝠响度和发射频度的更新操作

离散域的蝙蝠响度和发射频度的更新操作与连续域中的相似[5]. 设第 i 个蝙蝠的初始发射频度为 R_i^0,在 T 代其响度为 A_i^T,则在 $T+1$ 代蝙蝠的响度 A_i^{T+1} 和发射频度 R_i^{T+1} 为:

$$A_i^{T+1} = \alpha A_i^T \tag{14}$$

$$R_i^{T+1} = R_i^0 \times [1 - \exp(-\gamma \times T)] \tag{15}$$

其中,响度因子 α 和发射频度因子 γ 为常量且 $0 < \alpha < 1$、$\gamma > 0$.

3.1.5 随机插入策略

设第 i 个蝙蝠位置为 $x_i = (x_{i1}, x_{i2}, \cdots, x_{iw})$,$j$、$k$ 是随机产生的不相等的两个整数,$1 \leq j \leq w$,$1 \leq k \leq w$. 具体步骤:将 x_i 中第 j 个分量抽出来,再插入到第 k 个分量的位置,x_i 的其他分量做相应的移动.

3.2 目标函数

本文采用惩罚机制与向量比较机制相结合的方法改进式(1)后得到式(16):

$$\min\{F_3, F_1, F_2\} = \min\left\{P_{\max} \times \left(\sum_{k \in K}\left(\sum_{i \in V}\sum_{j \in J} x_{ij}^k q_i - D\right) + \left(\sum_{i \in V}\sum_{j \in J} x_{ij}^k (S_i^k - b_i)\right)\right), \sum_{j \in V_s}\sum_{k \in K} x_{0j}^k, \sum_{k \in K}\sum_{i \in V_s}\sum_{j \in V_s \setminus |i|} c_{ij} x_{ij}^k\right\} \tag{16}$$

其中,F_3 是容量和时间窗的惩罚之和,S_i^k 为车辆 k 实际到达客户 i 的时间,惩罚系数 $P_{\max}$ 是一个大的正整数.

适应度比较的具体步骤:设路径 1 的适应度为 $W_1 = \{F_3^1, F_1^1, F_2^1\}$,路径 2 的适应度为 $W_2 = \{F_3^2, F_1^2, F_2^2\}$,较优适应度 $W_{best} = \{F_3^{best}, F_1^{best}, F_2^{best}\}$. 如果$(F_3^1 < F_3^2) \| (F_3^1 == F_3^2 \&\& F_1^1 < F_1^2) \| (F_3^1 == F_3^2 \&\& F_1^1 == F_1^2 \&\& F_2^1 \leqslant F_2^2)$,则 $W_{best} = W_1$;否则 $W_{best} = W_2$.

3.3 局部搜索策略

根据 VRPTW 的特点,借鉴文献[9 ~ 11]的局部搜索策略,本文提出了普通插入搜索、最少客户车辆插入搜索、交换搜索、带时间窗的 2-Opt 搜索相结合的局部搜索策略. 具体步骤是:对于一个 VTPTW 路径,如果当前的迭代次数小于最大插入次数 M,则进行最少客户车辆插入搜索、普通插入搜索、交换搜索,否则只进行普通插入搜索、交换搜索.

(1)带时间窗的 2-Opt 搜索[11]:对每台车的访问路径分别进行带时间窗的 2-Opt 搜索,直到该车辆的访问路径不能再改进为止. 其示意图如图 1 所示,没有标号的顶点代表两个或者两个以上顶点间一系列的边. 如果 ab + cd > ac + bd && $W_{ab+cd} > W_{ac+bd}$,则删除边 ab 和 cd,同时增加边 ac 和 bd,并把顶点 b、c 之间的边反向. 其中,ab、cd、ac、bd 分别表示对应两点之间的距离,W_{ab+cd} 表示原路径的适应度,W_{ac+bd} 表示新路径的适应度.

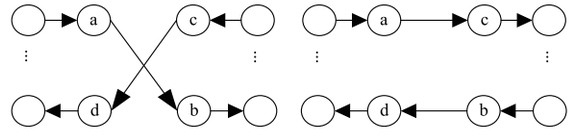

图1 2-Opt算法示意图

(2)普通插入搜索:随机选取 2 辆不同的车辆(记为车辆 1、车辆 2),再随机选取车辆 1 中的 1 个客户(记为客户 a);如果客户 a 插入到车辆 2 的访问路径后,车辆 2 仍然满足容量约束,对车辆 1、车辆 2 的新访问路径重新进行带时间窗的 2-Opt 搜索,如果 VRPTW 路径有改进,则认为是“插入成功”,停止插入搜索;如果“插入不成功”,重复上述操作,一直到尝试次数超过 L(最大尝试次数).

(3)最少客户车辆插入搜索:该搜索与普通插入搜索相似. 区别在于车辆 1 不再是随机选取的车辆,而是选取 1 辆访问客户数最少但不为 0 的车辆.

(4)交换搜索:随机选取 2 辆不同的车辆(记为车辆 1、车辆 2),再随机选取车辆 1 中的 1 个客户(记为客户 a),随机选取车辆 2 中的 1 个客户(记为客户 b);如果车辆 1 的客户 a 与车辆 2 的客户 b 交换后,车辆 1、车辆 2 仍然满足容量约束,对车辆 1、车辆 2 的新访问路径进行带时间窗的 2-Opt 搜索,如果 VRPTW 路径有改进,则认为是“交换成功”,停止交换搜索;如果“交换不成

功”,重复上述操作,一直到尝试次数超过 L.

本算法的局部搜索策略按照一定的规则来交换客户的访问次序,得到的路径仍然对应着一个完整的 VTPTW 路径,同时也符合本算法蝙蝠位置的定义.

3.4 算法步骤

DBA 的步骤如算法 1 所示.

算法 1　离散蝙蝠算法

输入:种群 Q,频率的最大值 $f_{\max}$,频率的最小值 $f_{\min}$,响度的最大值 $A_{\max}$,响度的最小值 $A_{\min}$,发射频度的最大值 $R_{\max}$,发射频度的最小值 $R_{\min}$,频率因子 θ,响度因子 α,发射频度因子 γ,惩罚系数 $P_{\max}$,最大迭代次数 $N_{\max}$,顶点数 n,车辆数 m,最大插入次数 M,最大尝试次数 L;

输出:全局最优蝙蝠的位置 x_*;

1. 初始化蝙蝠种群并应用 3.2 节的目标函数计算每个 x_i 的 W_i,初始化 x_* 及其 W_*,$T=0$,$i=1,2,\cdots,Q$;
2. while $T < N_{\max}$
3.　　for $i=1$ to Q
4.　　应用 3.1.3 节操作生成 v_i^{new}、f_i^{new} 和 x_i^{new},再 $v_i = v_i^{\text{new}}$,$f_i = f_i^{\text{new}}$;
5.　　if rand( ) $> R_i$ then
6.　　应用 3.1 节的随机插入策略更新 x_i^{new};
7.　　end if
8.　　应用 3.3 节的局部搜索策略找出 x_i^{new} 邻域中的最优蝙蝠位置 x_i^{pbest} 及其 W_i^{pbest};
9.　　if $W_i^{\text{pbest}} < W_i$ && rand( ) $< A_i$ then
10.　　$x_i = x_i^{\text{pbest}}$,应用 3.1.4 节操作更新 A_i 和 R_i;
11.　　end if
12.　　if $W_i^{\text{pbest}} < W_*$ then
13.　　$x_* = x_i^{\text{pbest}}$,$W_* = W_i^{\text{pbest}}$;
14.　　end if
15.　end for
16.　$T = T + 1$;
17. end while

4 实验与分析

4.1 实验环境

本文算法的相关实验均在同一实验环境中进行,其中 CPU 主频为 2.30GHz,内存为 4GB,操作系统为 32 位 Windows 7,编程语言为 C + +.

4.2 实验结果与分析

实验 1　采用 Solomon 的 6 种类型 VRPTW 算例(共 56 个算例)[12] 测试 DBA,然后将 DBA 和遗传算法粒子群的混合算法(Combination of Genetic Algorithm and Particle Swarm Optimization, GA-PSO)[13]、基于 P 系统的混合进化算法(P-Based Hybrid Evolutionary Algorithm, PHEA)[14]、蚁群禁忌搜索的混合算法(Consists of Ant Colony Optimization and Tabu search, ACO-Tabu)[15] 进行比较实验,实验结果如表 1. DBA 的参数设置如下:$Q=100$;$L=20$;$f\in[0,1]$;$A\in[0,1]$;$R\in[0,0.9]$;$\theta=w$;$\alpha=0.999$;$\gamma=0.001$;$P_{\max}=99$. 在表 1 中,BKS 是已知最优解;BS 是算法独立运行 30 次后获得的最优解;AS 是算法独立运行 30 次后获得的平均解;NV 是车辆数;TD 是总行驶距离;Time 是算法独立运行 30 次的平均时间耗费(单位:秒).

从表 1 可以看出,DBA 具有较强的寻优能力、较高的鲁棒性、较少的时间耗费. 主要表现如下:

(1)在 C1 和 C2 类测试算例:DBA 均能找出已知最优解,略优于其他三种算法;在平均解和时间耗费方面,DBA 差于 GA-PSO 和 ACO-Tabu.

(2)在 R1 和 RC1 类测试算例:DBA 的最优解和平均解中远远优于其他三种算法,甚至优于已知最优解;时间耗费方面,DBA 略多于 GA-PSO,明显优于 ACO-Tabu.

(3)在 R2 和 RC2 类测试算例:DBA 的车辆数明显少于其他三种算法,但是总行驶距离却变长了;时间耗费方面,DBA 远远优于 GA-PSO、ACO-Tabu.

(4)从 6 类测试算例的总和来看:DBA 的最优解和平均解均优于其他三种算法,甚至 DBA 的最优解优于已知最优解;DBA 的总时间耗费略比 GA-PSO 多,但是远远少于 ACO-Tabu 的时间耗费. 其中,部分 VRPTW 算例的实验结果如表 2 所示.

实验 2　不同 θ、α 和 γ 取值下的 DBA(其他参数的设置与实验 1 相同)解决 VRPTW 算例的实验结果如表 3 所示.

从表 3 可以看出:

(1)在 No. 1、2 和 3 的取值组合中,α 和 γ 相同,$\theta = 1\times w$ 时 DBA 的寻优能力和时间耗费的表现最均衡.

(2)在 No. 2、4 和 5 的取值组合中,θ 和 γ 相同,随着 α 取值的增大,DBA 的运行时间略为增加,但 DBA 的寻优能力却得到了质的飞跃.

(3)在 No. 2、6 和 7 的取值组合中,θ 和 α 相同,随着 γ 取值的变小,DBA 的时间耗费相差不大,但 DBA 的寻优能力却越来越强.

(4)在 7 种取值组合中,$\theta = 1\times w$、$\alpha = 0.999$、$\gamma = 0.001$ 时 DBA 的性能最好.

表 1 DBA 与其他算法的对比实验结果

算例	BKS		DBA		GA-PSO		PHEA		ACO-Tabu	
			BS	AS	BS	AS	BS	AS	BS	AS
C1	NV	10.00	10.00	10.13	10.00	10.06	9.89	——	10	——
	TD	828.38	828.38	897.18	828.38	839.28	921.86	——	841.92	843.55
	Time	——	——	264.84	——	62	——	——	——	210
C2	NV	3.00	3.00	3.36	3.00	3.05	3.00	——	3.3	——
	TD	589.86	589.73	643.45	589.86	601.29	589.86	——	612.75	611.12
	Time	——	——	143.25	——	135	——	——	——	142
R1	NV	11.83	8.83	9.13	13.08	12.78	12.25	——	13.1	——
	TD	1207.20	1096.67	1135.89	1182.04	1192.76	1194.54	——	1213.16	1241.24
	Time	——	——	77.47	——	60	——	——	——	698
R2	NV	2.73	2.82	3.51	5.36	4.72	2.91	——	4.6	——
	TD	946.74	970.63	991.07	899.98	916.62	955.74	——	952.3	961.11
	Time	——	——	84.11	——	182	——	——	——	655
RC1	NV	11.50	9.25	9.56	12.75	12.50	11.88	——	12.7	——
	TD	1400.89	1333.11	1328.22	1351.73	1362.02	1369.97	——	1415.62	1419.14
	Time	——	——	61.35	——	58	——	——	——	317
RC2	NV	3.25	3.25	3.80	6.38	5.82	3.38	——	5.6	——
	TD	1119.17	1161.00	1183.10	1034.28	1054.60	1139.53	——	1120.37	1119.24
	Time	——	——	56.84	——	149	——	——	——	407
ALL	NV	404	351	373.04	483	466.68	414	——	470	——
	TD	57235.33	55963.17	57845.14	55346.67	56092.75	57939.15	——	57799	58255.04
	Time	——	——	6329.82	——	6016	——	——	——	24399

表 2 DBA 解决 56 个 VRPTW 测试用例的部分实验结果

算例	BKS		N_{max}	M	BS		AS		
	NV	TD			NV	TD	NV	TD	Time
C101	10	828.94	10000	100	10	828.94	10.43	870.13	64.98
C104	10	824.78	60000	100	10	824.78	10.00	902.65	406.26
C201	3	591.56	2000	2000	3	591.56	3.80	634.16	52.33
C204	3	590.60	3500	2000	3	590.60	3.00	640.955	122.34
R101	18	1613.59	10000	100	12	1473.52	12.43	1499.25	86.98
R104	9	1007.24	10000	100	8	919.4	8.00	975.69	64.08
R201	4	1252.37	3000	3000	4	1312.09	5.10	1293.643	87.86
R204	2	825.52	1000	1000	2	814.03	2.77	837.92	27.78
RC101	14	1696.94	10000	100	10	1511.148	10.37	1585.39	87.86
RC108	10	1139.82	10000	100	9	1071.295	9.00	1152.99	42.34
RC201	4	1406.91	2000	2000	4	1497.651	4.93	1498.85	56.78
RC208	3	828.14	1000	500	3	821.795	3.00	909.97	18.77

表 3　不同频率因子、响度因子和发射频度因子取值下的 DBA 解决 VRPTW 算例的实验结果

No.	参数组合		C101			R104			R204			RC208		
			NV	TD	Time	NV	TD	Time	NV	TD	Time	NV	TD	Time
1	$\theta=0.5\times w$, $\alpha=0.999$, $\gamma=0.001$	BS	10	861.86	——	8	933.8	——	2	864.07	——	3	866.38	——
		AS	10.6	889.68	63.29	8	954.89	63.12	2.7	830.65	32.67	3	895.76	21.37
2	$\theta=1\times w$, $\alpha=0.999$, $\gamma=0.001$	BS	10	828.94	——	8	919.4	——	2	814.03	——	3	821.8	——
		AS	10.43	870.13	64.98	8	975.69	64.08	2.77	837.92	27.78	3	909.97	18.77
3	$\theta=2\times w$, $\alpha=0.999$, $\gamma=0.001$	BS	10	828.94	——	8	947.12	——	2	894.71	——	3	879.22	——
		AS	10.3	928.44	62.96	8	979.69	61.84	2.9	842.63	23.78	3.1	922.1	13.65
4	$\theta=1\times w$, $\alpha=0.9$, $\gamma=0.001$	BS	19	2421.19	——	9	1526.21	——	4	1153.98	——	4	1331.41	——
		AS	20.36	2816.79	43.57	9.71	1620.88	34.56	4.64	1233.37	9.86	4.21	1440.8	4.46
5	$\theta=1\times w$, $\alpha=0.99$, $\gamma=0.001$	BS	11	887.15	——	8	977.82	——	3	835.67	——	3	888.07	——
		AS	11	953.57	58.78	8	1021.25	62.53	2.93	866.98	19.96	3.07	1018.79	15.32
6	$\theta=1\times w$, $\alpha=0.999$, $\gamma=0.1$	BS	10	828.94	——	8	949.68	——	2	832.86	——	3	848.62	——
		AS	10.64	889.94	63.49	8	979.24	60.14	2.93	819.64	25.76	3	913.44	20.24
7	$\theta=1\times w$, $\alpha=0.999$, $\gamma=0.5$	BS	10	828.94	——	8	955.47	——	2	838.25	——	3	850.33	——
		AS	10.43	896.04	63.47	8	982.15	63.59	2.86	837.57	30.88	3	909.82	20.7

实验 3　DBA 不采用随机插入策略,其余操作和策略保持不变,得到的算法记为 DBA-ORI. DBA 不采用局部搜索策略,其余操作和策略保持不变,得到的算法记为 DBA-OLS. 因 DBA-OLS 不采用局部搜索策略,其迭代运行一次的时间将远远少于 DBA、DBA-ORI. 为了保证实验的公平性,增加 DBA-OLS 的 $N_{\max}$ 使 DBA-OLS 的时间耗费与 DBA 相同,再将 DBA-OLS 与 DBA、DBA-ORI 相比较(参数的设置与实验 1 相同),得到实验结果如表 4 所示.

从表 4 可以看出,DBA 的最优解和平均解均好于 DBA-ORI,DBA 的时间耗费也略少于 DBA-ORI;在相同时间耗费的情况下,DBA 相对于 DBA-OLS,最优解和平均解均有了质的飞跃,需使用车辆数也变的更少.

由此可见,随机插入策略和局部搜索策略能够显著地提高 DBA 的寻优能力、鲁棒性并降低其时间耗费.

表 4　DBA 与 DBA-ORI、DBA-OLS 对比的实验结果

算法		C101			R104			R204			RC208		
		NV	TD	Time	NV	TD	Time	NV	TD	Time	NV	TD	Time
DBA	BS	10	828.94	——	8	919.4	——	2	814.03	——	3	821.8	——
	AS	10.43	870.13	64.98	8	975.69	64.08	2.77	837.92	27.78	3	909.97	18.77
DBA-ORI	BS	10	828.94	——	8	937.85	——	2	838.73	——	3	827.32	——
	AS	10.71	878.3	68.95	8	989.17	69.78	2.86	839.22	36.76	3	911.23	28.64
DBA-OLS	BS	12	1023.7	——	9	1020.29	——	4	1000.37	——	5	1133.24	——
	AS	13.57	1189.57	——	9	1069.02	——	5.29	1033.5	——	5.29	1290.44	——

实验 4 使用 t-检验中的成对双样本均值分析方法来检验 DBA 与 GA-PSO、PHEA、ACO-Tabu 之间的算法性能是否显著性差异. 利用 4 个算法求解 56 个算例的最优解[13~15]来计算得到平均车辆路径(由总行驶距离除以车辆数得到)并使用平均车辆路径进行假设检验.

假设两个对比算法没有显著性差异 $H_0:\mu=\mu_0$,对立的备择假设为两个对比算法有显著性差异 $H_A:\mu\neq\mu_0$,该假设检验属于双尾测验,设显著性水平为 0.05,平均差为 0. 使用 EXCEL 的"数据分析"功能对相应算法的平均车辆路径进行成对双样本均值分析,实验结果如表 5 所示.

从表 5 可以看出,$|t|$ 分别为 6.192039、3.357368、5.710065,均大于 $t_{0.05}=2.004045$. 因此,均应否定 H_0,接收 H_A,说明 DBA 分别与 GA-PSO、PHEA、ACO-Tabu 的算法性能均存在显著性差异.

表 5 t-检验:成对双样本均值分析

统计数值	DBA 与 GA-PSO 的对比		DBA 与 PHEA 的对比		DBA 与 ACO-Tabu 的对比	
	DBA	GA-PSO	DBA	PHEA	DBA	ACO-Tabu
平均	209.3191	136.203	209.3191	195.584	209.3191	157.602
方差	13461.25	2747.224	13461.25	12673	13461.25	4988.002
$\|t\|$	6.192039		3.357368		5.710065	
$t_{0.05}$	2.004045		2.004045		2.004045	

5 结论

本文提出了一种离散蝙蝠算法求解 VRPTW. 实验结果表明:DBA 具有较强的寻优能力、较高的鲁棒性、较少的时间耗费;所采用的关键参数值和策略能显著地提高 DBA 的性能;DBA 与对比算法之间的算法性能均有显著性差异.

参考文献

[1] Dantzig G B, Ramser J H. The truck dispatching problem [J]. Management Science, 1959, 6(1): 80 – 91.

[2] Gong Y J, Zhang J, Liu O, et al. Optimizing the vehicle routing problem with time windows: a discrete particle swarm optimization approach [J]. IEEE Transactions on Systems Man and Cybernetics Part C, 2012, 42(2): 254 – 267.

[3] Hu W, Liang H, Peng C, et al. A hybrid chaos-particle swarm optimization algorithm for the vehicle routing problem with time window [J]. Entropy, 2013, 4(4): 1247 – 1270.

[4] Barbucha D. A cooperative population learning algorithm for vehicle routing problem with time windows [J]. Neurocomputing, 2014, 146(146): 210 – 229.

[5] Yang X S. A New Metaheuristic Bat-Inspired Algorithm [M]. Heidelberg: Springer Berlin Heidelberg, 2010. 65 – 74.

[6] Osaba E, Yang X S, Diaz F, et al. An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems [J]. Engineering Applications of Artificial Intelligence, 2016, 48(C): 59 – 71.

[7] Zhou Y, Luo Q, Xie J, et al. A Hybrid Bat Algorithm with Path Relinking for the Capacitated Vehicle Routing Problem [M]. Cham: Springer International Publishing, 2016. 255 – 276.

[8] 孟祥虎,胡蓉,钱斌. 求解带时间窗车辆路径问题的有效混合 PBIL 算法 [J]. 系统工程理论与实践, 2014, 34(10): 2701 – 2709.
MENG Xiang-hu, HU Rong, QIAN Bin. Effective hybrid population-based incremental learning algorithm for vehicle routing problem with time windows [J]. Systems Engineering-Theory and Practice, 2014, 34(10): 2701 – 2709. (in Chinese)

[9] 葛斌,韩江洪,魏臻,等. 求解带时间窗车辆路径问题的动态混合蚁群优化算法 [J]. 模式识别与人工智能, 2015, 28(7): 641 – 650.
GE Bin, HAN Jiang-Hong, WEI Zhen, et al. Dynamic hybrid ant colony optimization algorithm for solving the vehicle routing problem with time windows [J]. Pattern Recognition and Artificial Intelligence, 2015, 28(7): 641 – 650. (in Chinese)

[10] Hiermann G, Puchinger J, Ropke S, et al. The electric fleet size and mix vehicle routing problem with time windows and recharging stations [J]. European Journal of Operational Research, 2016, 252(3): 995 – 1018.

[11] 周永权,黄正新,刘洪霞. 求解 TSP 问题的离散型萤火虫群优化算法 [J]. 电子学报, 2012, 40(6): 1164 – 1170.
ZHOU Yong-quan, HUANG Zheng-xin, LIU Hong-xia. Discrete glowworm swarm optimization algorithm for TSP problem [J]. Acta Electronica Sinica, 2012, 40(6): 1164 – 1170. (in Chinese)

[12] Qi Y, Hou Z, Li H, et al. A decomposition based memetic algorithm for multi-objective vehicle routing problem with

time windows[J]. Computers and Operations Research, 2015,62(C): 61 – 77.

[13] Xu S H, Liu J P, Zhang F H, et al. A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows[J]. Sensors, 2015,15(9): 21033 – 21053.

[14] Niu Y, He J, Wang Z, et al. A P-Based hybrid evolutionary algorithm for vehicle routing problem with time windows[J]. Mathematical Problems in Engineering, 2014, 2014(3): 1 – 11.

[15] Yu B, Yang Z Z, Yao B Z. A hybrid algorithm for vehicle routing problem with time windows[J]. Expert Systems with Applications, 2011,38(1): 435 – 441.

作者简介

戚远航　男,1993 年 6 月出生,广东湛江人. 现为广东工业大学博士生,从事供应链物流及智能算法的研究.

E-mail: qiyuanhang77@163.com

蔡延光(通信作者)　男,1963 年 2 月出生,湖北咸宁人. 1988 年和 1996 年分别在重庆大学和浙江大学获理学硕士和工学博士学位. 现为广东工业大学教授,博士生导师,从事复杂网络系统建模、控制与优化、物流控制与优化、智能交通系统、组合优化、智能优化、物联网信息处理与优化控制等方面的研究.

E-mail: caiyg99@163.com