文章编号:1007－6735(2019)02－0167－07　DOI: 10.13255/j.cnki.jusst.2019.02.011

求解广义分配问题的拉格朗日蝙蝠算法

万晓琼[1]，　张惠珍[1]，　赵玉苹[2]
（1. 上海理工大学 管理学院，上海 200093；2. 国网上海市电力公司 物资公司，上海 200235）

摘要：基于广义分配问题(GAP)自身的特点，将拉格朗日松弛算法(LR)和蝙蝠算法(BA)相结合，提出了一种高效的拉格朗日蝙蝠算法(LR-DBA)。首先，基于 GAP 的数学模型，在 BA 算法的基本框架上，重新定义了蝙蝠速度、位置以及局部更新公式，得出全新的求解 GAP 的离散蝙蝠算法(DBA)。其次，将其与 LR 相结合，设计出求解 GAP 的 LR-DBA 算法。最后，经过大量算例测试表明，对比 DBA 算法，LR-DBA 混合算法在求解 GAP 时具有明显优势。

关键词：广义分配问题；蝙蝠算法；拉格朗日松弛算法
中图分类号：TP 301.6　　**文献标志码：**A

Lagrangian Bat Algorithm for Solving Generalized Assignment Problems

WAN Xiaoqiong[1], ZHANG Huizhen[1], ZHAO Yuping[2]
(1. *Business School, University of Shanghai for Science and Technology, Shanghai 200093, China*;
2. *Procurement Company, State Grid Shanghai, Shanghai 200235, China*)

Abstract: Generalized assignment problem (GAP) is a classic combinatorial optimization problem that has been proved to be NP-hard problem. For solving the GAP problem, combining the bat algorithm (BA) with the Lagrangian relaxation algorithm (LR), an efficient Lagrangian bat algorithm (LR-DBA) was proposed based on the characteristics of the GAP problem. The DBA takes BA as the basic framework and redefines the formulas of velocity, position and local updating. It was then combined with the LR algorithm to form the LR-DBA hybrid algorithm. A large number of examples show that compared with the DBA algorithm, the LR-DBA algorithm has obvious advantages in solving the GAP problem.

Keywords: *generalized assignment problem*; *bat algorithm*; *Lagrangian relaxation algorithm*

广义分配问题(generalized assignment problem，GAP)[1]是将给定的工作分配给有限定资源量的代理器来完成，以消耗最小的成本或取得最大利润为目标，是经典的组合优化问题。GAP 在实际生

收稿日期: 2018−01−24
基金项目: 国家自然科学基金资助项目(71401106)；教育部人文社科规划基金资助项目(16YJA630037)
第一作者: 万晓琼(1992−)，女，硕士研究生．研究方向：智能优化．E-mail：478205057@qq.com
通信作者: 张惠珍(1979−)，女，副教授．研究方向：运筹学、智能优化．E-mail：zhzzywz@163.com

活中具有广泛的应用，许多领域的容量约束问题都可以被抽象为GAP算例进行求解，如机器调度问题、有容量设施选址问题、供应链问题及车辆路径问题等。

GAP在理论和应用价值上的意义引起了国内外学者的极大关注，多种算法被用来求解此问题，这些算法大致可以归为3类：近似算法[2-3]、精确算法[4-6]和智能优化算法。由于GAP的复杂性，近似算法和精确算法对于求解大规模问题具有局限性，这促使了智能优化算法的广泛应用，如Bai等[7]采用粒子群算法(PSO)求解两阶段模糊GAP；Tapkan等[8]在蜂群算法中引入多种模糊排序方式用于解决模糊多目标GAP；Liu等[9]提出一种可扩展并行遗传算法(GA)求解GAP问题；Laguna等[10]在禁忌搜索算法(TS)中利用斥链定义邻域，并用其求解多级GAP。它们在解决组合优化问题时取得了一定的成果，同时也显现出单一智能优化算法的一系列缺点，如容易陷入局部最优解、稳定性差等。为了克服这些缺点，以及获得更精确的解，运用混合智能优化算法解决GAP问题得到长足的发展，如吴良杰等[11]将禁忌搜索算法作为蚁群算法的局部搜索策略，提出了混合算法，有效地解决GAP；Chen等[12]结合小生境遗传算法与蚁群算法，并建立蚂蚁搜索的禁忌规则，提出了小生境遗传蚂蚁算法来求解GAP；Osman[13]将禁忌搜索算法与模拟退火算法相结合，用于求解GAP。

本文运用蝙蝠算法(BA)[14]融合拉格朗日松弛算法(LR)[15-16]来解决基本GAP。该算法利用蝙蝠算法产生GAP最优解的上界，采用拉格朗日松弛算法求出GAP最优解的下界，两者结合，逐渐逼近GAP的最优解。

1 广义分配问题

GAP可以描述为：将n个相互独立的工作分配给m个有限定资源量的代理器，一个工作只能由一个代理器服务，一个代理器可以服务多个工作，且代理器完成工作所需的总资源量不能超过自身的限制。GAP的数学模型可表述为

$$\min F(x)=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij} \tag{1}$$

$$\text{s.t.}\sum_{j=1}^{n}r_{ij}x_{ij}\leqslant b_i,\quad \forall i\in I \tag{2}$$

$$\sum_{i=1}^{m}x_{ij}=1,\quad \forall j\in J \tag{3}$$

$$x_{ij}=\begin{cases}1, & \text{工作 } j \text{ 分配给代理器 } i\\ 0, & \text{否则}\end{cases} \tag{4}$$

式中：$I=\{i\mid i=1,2,\cdots,m\}$为代理器集；$J=\{j\mid j=1,2,\cdots,n\}$为工作集；$b_i$表示代理器$i$自身拥有的资源量；$r_{ij}$表示代理器$i$为工作$j$服务时消耗的资源量；$c_{ij}$表示代理器$i$为工作$j$服务时花费的成本；目标函数$F(x)$表示最小化分配成本；约束(2)保证分配给代理器$i$的工作消耗的总资源量不超出代理器自身拥有的资源量；约束(3)确保每个工作j只能分配给一个代理器来完成；约束(4)中的x_{ij}为决策变量。

2 蝙蝠算法

蝙蝠算法最早由剑桥大学的Yang于2010年提出，是一种基于微型蝙蝠群体回声定位行为的搜索全局近似解的智能优化算法。蝙蝠个体是蝙蝠算法的基本单元，它们通过口腔或者鼻腔将从喉部产生的超声波发射出去，然后依据发出超声波和接收到回音之间的时差、到达两耳之间的不同强度、回声定位波形的变化来判断猎物或障碍物的距离、方位以及性质，从而制定捕捉猎物和躲避障碍物的计划，同时也使得蝙蝠即使在黑暗中也能找到位于裂缝中的栖息地。蝙蝠算法最早被用来解决连续问题，近几年有学者利用它来解决离散问题，例如，旅行商问题[17]、车辆路径问题[18]及背包问题[19]等。

本文尝试运用蝙蝠算法来解决基本GAP问题。在求解过程中，假设蝙蝠群体随机散布于d维搜索空间下，一个蝙蝠对应GAP问题的一个解，GAP问题的目标函数决定蝙蝠的适应度值，蝙蝠群体的其他个体通过调整超声波的响度、脉冲发射率来追随当前群体中的最优蝙蝠在解空间中进行搜索，从而使蝙蝠群体在求解空间中的运动由无序演变为有序，直至达到满意解。

2.1 求解GAP问题的蝙蝠算法

由于GAP问题的特点，基础的蝙蝠算法已不适用。本文在不改变蝙蝠算法原有概念和结构的基础上，引入单点基因换位算子建立全新的速度、位置以及局部搜索更新规则，称之为离散蝙蝠算法(discrete bat algorithm，DBA)。现从蝙蝠位置初始编码、蝙蝠位置和速度更新、局部更新以及音量

和脉冲的更新这 4 个方面来介绍离散蝙蝠算法。

2.1.1 蝙蝠位置初始编码

蝙蝠算法中蝙蝠的位置编码代表 GAP 的可行解。设蝙蝠种群大小为 s，第 i 只蝙蝠的速度为 v_i，初始位置为 x_i，迭代次数 N。现介绍第 i 只蝙蝠个体初始位置编码方式。

a. 用 n 维向量 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{in})$ 表示第 i 只蝙蝠的初始位置，其中，x_{ij} 表示为第 j 个工作服务的代理器。

b. 将 n 个工作进行随机排列，记为 $\theta_i=(\theta_{i1},\theta_{i2},\cdots,\theta_{in})$，此排列为分配工作的顺序，即 θ_{iu} 为第 u 个要被分配的工作。

c. 从 θ_{i1} 开始进行分配，分配方式：将给定的代理器服务该工作时消耗的资源量按升序排列，从中随机选取前两个代理器中的任一个为其服务，若选中的代理器为 w，则记 $x_{i\theta_{i1}}=w$，依次进行，直至全部工作分配完毕。

d. 若此时有代理器消耗的总资源量超出自身资源限制，则通过以下方式进行调整：搜索超出资源限制的代理器服务的工作，并计算代理器完成该工作时消耗单位资源花费的成本 $P_{ij}=\dfrac{c_{ij}}{r_{ij}}$，将最大 P 值对应的工作重新分配给剩余资源量足够且花费成本最小的代理器。

2.1.2 蝙蝠速度和位置更新

第 i 只蝙蝠在定义的 d 维搜索空间中，第 t 时刻下速度 v_i^t 的更新公式为

$$v_i^t=H\left(x_i^{t-1},\ x_*\right) \tag{5}$$

式中：x_* 表示当前最优解，它是通过比较当前整个蝙蝠种群对应的适应度值而得到；$H\left(x_i^{t-1},x_*\right)$ 指 x_i^{t-1} 和 x_* 之间的汉明距离，具体为第 i 只蝙蝠在第 $t-1$ 时刻位置编码较最优蝙蝠位置编码对应位不同的个数。例如，$x_i^{t-1}=(1,3,2,2,1)$，$x_*=(1,2,3,2,1)$，可以看出，在 x_i^{t-1}，x_* 中工作 1，4，5 均被分配给 1，2，1 代理器，而工作 2，3 分配给不同的代理器，则此时 $v_i^t=2$。

蝙蝠的位置更新采用单点基因换位算子，即在蝙蝠原位置编码的基础上随机选取 2 个编码位置进行交换，换位后的位置编码对应的适应度值变优且符合约束条件时更新编码，直至共进行 v_i^t 次单点基因换位，位置更新结束。以 $m=3$，$n=7$ 为例，随机选取的 2 个编码位置为 2 和 5，则单点基因换位的方式如图 1 所示。

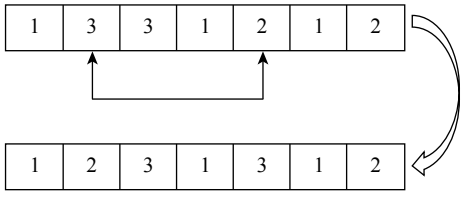

图 1　单点基因换位方式

Fig.1　Single point gene translocation

2.1.3 局部更新

类似于现存的其他智能优化算法，蝙蝠算法也可以进行局部搜索行为，且相对于其他算法，蝙蝠算法可以通过比较脉冲发生率 r_i 和随机数的大小，实现对全局搜索和局部搜索之间转换的动态控制。

当随机函数 $rand>r_i$ 时，在当前最优解集中选择一个最优蝙蝠，进行随机游走，产生局部新解。游走方式同样采用单点基因换位，具体是在当前最优解集中选择一个最优蝙蝠，在该蝙蝠原有的位置编码的基础上进行一次单点基因换位，更新后的位置编码对应的适应度值变优且在符合约束条件时更新编码。

2.1.4 蝙蝠音量和脉冲发生率的更新

在蝙蝠接近猎物的过程中，声波音量逐渐降低，同时脉冲发生率逐渐提高。音量 A_i^t 和脉冲发生率 r_i^t 的更新公式为

$$r_i^t=r_i^0\left[1-\exp(-\gamma(t-1))\right] \tag{6}$$

$$A_i^t=\alpha A_i^{t-1} \tag{7}$$

式中：α 为音量衰减系数；γ 为脉冲发生率增加系数。对任意 $0<\alpha<1$，$\gamma>0$，可以看出，$t\to\infty, A_i^t\to 0$，r_i^t 逐渐接近最大脉冲发生率 r_i^0。

为了简单起见，在实际运行中，通常使用 $\alpha=\gamma=0.9$。

2.2 求解 GAP 问题的蝙蝠算法设计

通过上文方式进行基础蝙蝠算法的改造，设计出的全新求解 GAP 问题的蝙蝠算法的具体步骤如下：

步骤 1　初始化蝙蝠种群。蝙蝠种群个数为 s，第 i 只蝙蝠初始位置为 x_i，速度为 v_i，音量为 A_i^0，脉冲发生率为 r_i^0，迭代次数 N。

步骤 2　运用式(5)以及单基因换位算子进行蝙蝠速度和位置的更新。

步骤 3　若 $rand>r_i$，进行局部搜索。

步骤 4　若 $rand<A_i$ 且 $F(x_i)<F(x_*)$，接受这个新解。

步骤 5　根据式(6)和式(7)进行蝙蝠脉冲发生

率和音量更新。

步骤 6 排列蝙蝠并找到当前最优蝙蝠 x_*。

步骤 7 $N'=N'+1$，若$N'>N$，则结束搜索并输出结果；否则，转步骤 2。N'为此刻的迭代次数。

3 拉格朗日蝙蝠算法

3.1 求解 GAP 的拉格朗日松弛算法

拉格朗日松弛算法（Lagrangian relaxation，LR）是一种分解协调算法，由 Herd 和 Karp 于 1970 年提出，最早用于解决旅行商问题。随后被用于各大领域，现已成为解决组合优化问题时产生解的下界的一种不可或缺的方法。拉格朗日松弛算法的基本原理是通过引入拉格朗日乘子 λ_j，将原问题中的某些约束作为惩罚项吸收到目标函数中，使原问题成为较易解决的拉格朗日松弛问题。通过拉格朗日乘子松弛约束式（3），松弛后的 GAP 可表示为

$$\min_x F'(x)=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij}+\sum_{j=1}^{n}\lambda_j\left(1-\sum_{i=1}^{m}x_{ij}\right) \quad (8)$$

$$\min_x F'(x)=\sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij}-\lambda_j)x_{ij}+\sum_{j=1}^{n}\lambda_j \quad (9)$$

s.t. 式（2）和式（4）

再通过标准次梯度优化算法修正拉格朗日乘子，求得拉格朗日对偶问题（Lagrangian dual，LD）的解。其中，拉格朗日对偶问题表示为

$$\max_\lambda \min_x F'(x,\lambda)=\sum_{i=1}^{m}\sum_{j=1}^{n}(c_{ij}-\lambda_j)x_{ij}+\sum_{j=1}^{n}\lambda_j \quad (10)$$

s.t. 式（2）和式（4）

利用标准次梯度优化算法，第 t 时刻对拉格朗日乘子的修正为

$$T^t=\frac{\pi(UB^t-LB^t)}{\sum_{i=1}^{m}G_j^2} \quad (11)$$

$$\mathrm{G}_j=1-\sum_{i=1}^{m}x_{ij}^t \quad (12)$$

$$\lambda_j^{t+1}=\max\left(0,\lambda_j^t+T^tG_j\right) \quad (13)$$

式中：π为常数，通常取π=2；G_j表示次梯度；UB^t指第 t 时刻 GAP 问题最优解的上界；LB^t表示第 t 时刻 GAP 问题最优解的下界。

式（11）和（12）为拉格朗日乘子步长的更新；式（13）表示拉格朗日乘子的更新。

通过以上方式求解拉格朗日对偶问题，解的目标函数值为原问题最优解的下界。但由于约束条件被松弛，扩大了 GAP 问题可行解的范围，导致松弛后的解不一定是原问题的可行解。

为了保证解的可行性，本文对拉格朗日对偶问题的解进行可行化，具体方式为：搜索没有被分配的工作，选择完成该工作消耗量最小的代理器为其服务。同理，搜索有多个代理器服务的工作，选择完成该工作时消耗的资源量最小的为其服务，执行此操作直至所有工作都只有一个代理器为其服务。检验每个代理器消耗的总资源量，若超出自身占有，按照 d 的调整方式进行修正。可行化后的解为原问题的可行解，对应的目标函数值为原问题最优解的上界。

3.2 求解 GAP 问题的拉格朗日蝙蝠混合算法设计

将离散蝙蝠算法和拉格朗日松弛算法相结合，提出全新的解决 GAP 问题的拉格朗日蝙蝠算法（Lagrangian bat algorithm，LR-DBA）。两种算法取长补短，离散蝙蝠算法求得 GAP 问题最优解的上界，此解为标准次梯度提供优秀上界；拉格朗日松弛算法求得 GAP 问题最优解的下界，当可行化后的解优于蝙蝠算法的最差解时，替代最差解；求得的上界、下界相互逼近，逐渐接近 GAP 问题的最优解。该算法的具体步骤如下：

步骤 1 初始化蝙蝠种群。蝙蝠种群个数 s，初始音量 A_i^0，初始脉冲发生率 r_i^0，最大迭代次数 N，初始化蝙蝠个体位置 x_i。

步骤 2 蝙蝠速度和位置更新。计算得出最优蝙蝠 x_*。通过式（5）更新速度，利用单点遗传基因换位算子更新位置 x_{new}。

步骤 3 局部搜索。若 $rand>r_i$，从最优解集中选一个解，利用单点遗传基因换位算子产生一个局部解 x_{new}。

步骤 4 若 $rand<A_i$ 且 $F(x_{\mathrm{new}})<F(x_*)$，则更新当前满意解及其对应的目标函数值，并通过式（6）增大 r_i，式（7）减小 A_i。

步骤 5 排列蝙蝠并找到当前最优 x_*，计算 GAP 问题最优解的上界，记为 UB。

步骤 6 运用标准次梯度算法求解拉格朗日对偶问题，求得的解为 GAP 问题最优解的下界，记为 LB。

步骤 7 对拉格朗日对偶问题的解进行可行

化，求得 GAP 问题最优解的上界，记为UB'。若可行化后的解优于蝙蝠算法的最差解，则更新蝙蝠算法最差解。

步骤 8　$N'=N'+1$，若$N'<N$，则转步骤 2；否则，结束搜索并输出结果。

4　仿真实验与分析

为了验证算法的有效性，测试了 Beasely 的 OR-library 中的 16 组标准测试算例。其中，包括 12 组小规模算例，每组小规模算例中包含 5 个小算例；4 组较大规模算例，每组较大规模算例中包含 3 个小算例。在 Intel（R）Core（TM）i5-3320 CPU @2.60 GHz 2.60 GHz 内存，64 位 Windows 8 操作系统的环境下利用 Matlab R2015a 进行编程实现和数据处理。该混合算法的初始参数设置为：蝙蝠种群数量 s=20，迭代次数 N=500，初始音量$r_i^0=0.25$，初始脉冲发生率$r_i^0=0.5$，$\alpha=\beta=0.9$，$\pi=2$，拉格朗日乘子 λ_j 的初始值参照文献[20]的取值方式，即$\lambda_j=\max c_{ij},\ j\in J$。

为了测试 LR-DBA 混合算法的求解性能，将其与 DBA 算法进行比较。表 1 和表 2 分别列出了小规模和较大规模算例的相关数据。表中，运行时间为 500 次迭代的总时间；算例的目标函数值为随机运行 5 次，取最好一次为满意解；G 为本文求得的满意解和文献[21]的最优目标函数值之间的差值；平均运行时间和平均 G 为一组算例中 5 个小算例的平均值；达到最优算例个数为一组算例中 5 个小算例中达到最优解的个数。C 为目标函数值，D 为最优目标函数值。G 的计算式为

$$G=\frac{|C-D|}{D}\times 100\%$$

由表 1 可以看出，DBA 算法求解的 60 个算例中只有 2 个达到最优解，而用 LR-DBA 混合算法求解时有 10 个达到最优，且 12 组算例的 G 均小于 1%，几乎达到最优解，表明 LR-DBA 混合算法在求解小规模算例中结果较好。表 2 中，对比 DBA 算法和 LR-DBA 混合算法的 G，很明显看出 LR-DBA 混合算法在求解较大规模 GAP 算例中也有很大的优势，LR-DBA 混合算法求解的 12 个算例中除 gapc3 以外，其他算例的 G 均在 5% 以内，而 DBA 算法求解所得 G 最大达到 15%，且 12 个算例中有 7 个 G 都大于 5%。分析全部算例，可以看出，LR-DBA 混合算法在求解 GAP 问题算例时具有更好的全局搜索能力，且收敛精度较好。

表 1　小规模 GAP 算例计算结果

Tab.1　Calculating results of a small-scale GAP

算例	规模		DBA			LR-DBA		
	m	n	平均运行时间/s	平均 G/%	达到最优算例个数/总算例个数	平均运行时间/s	平均 G/%	达到最优算例个数/总算例个数
gap1	5	15	3.44	0.66	1/5	17.52	0.12	3/5
gap2	5	20	3.70	0.38	1/5	21.01	0.18	4/5
gap3	5	25	4.00	1.30	0/5	27.50	0.59	1/5
gap4	5	30	4.29	2.16	0/5	23.65	0.45	2/5
gap5	8	24	4.75	1.88	0/5	40.26	0.78	0/5
gap6	8	32	5.62	2.89	0/5	48.93	0.77	0/5
gap7	8	40	6.33	2.29	0/5	45.23	0.80	0/5
gap8	8	48	6.81	1.68	0/5	58.04	0.93	0/5
gap9	10	30	6.00	2.05	0/5	40.90	0.73	0/5
gap10	10	40	6.52	1.95	0/5	64.10	0.63	0/5
gap11	10	50	8.29	3.72	0/5	62.45	0.60	0/5
gap12	10	60	9.30	2.42	0/5	82.97	0.76	0/5
平均值	7.67	34.50	5.75	1.95	0.17/5	44.38	0.61	0.83/5

表 2 较大规模 GAP 算例计算结果

Tab.2 Calculating results of a large-scale GAP

算例	规模		最优目标函数值	DBA			LR-DBA		
	m	*n*		运行时间/s	目标函数值	*G*/%	运行时间/s	目标函数值	*G*/%
gapa1	5	100	1 698	14.75	1 723	1.47	51.06	1 702	0.24
gapa2	5	200	3 235	52.41	3 304	2.13	48.10	3 238	0.09
gapa3	10	100	1 360	31.65	1 405	3.31	57.57	1 369	0.66
gapb1	5	100	1 843	7.22	1 978	7.22	72.24	1 921	4.23
gapb2	5	200	3 553	35.13	3 802	7.00	182.66	3 682	3.63
gapb3	10	100	1 407	21.27	1 587	12.79	112.34	1 454	3.34
gapc1	5	100	1 931	14.72	2 012	4.20	75.46	1 975	2.28
gapc2	5	200	3 458	30.84	3 736	8.04	167.96	3 576	3.41
gapc3	10	100	1 403	18.87	1 614	15.04	135.56	1 489	6.13
gapd1	5	100	6 373	11.62	6 666	4.60	166.16	6 572	3.12
gapd2	5	200	12 796	18.49	13 457	5.17	378.88	13 150	2.77
gapd3	10	100	6 379	12.28	6 820	6.91	226.33	6 681	4.73
平均值	6.67	133.33	3 786.33	22.44	4 008.67	6.49	139.53	3 900.75	2.89

综合分析表 1 和表 2 的实验数据，LR-DBA 混合算法求解 GAP 问题性能优异，尤其针对小规模算例。但由于 LR-DBA 混合算法结合了拉格朗日松弛算法，在求解小规模 GAP 问题时，LR-DBA 混合算法的平均运行时间几乎是 DBA 算法运行时间的 8 倍，在较大规模的求解中，LR-DBA 的平均运行时间也是 DBA 算法的 6 倍，说明混合算法的收敛速度慢。

为了更直观地展现 LR-DBA 混合算法的求解效果，本文以 gapa3 为例，给出求解 GAP 问题的迭代过程图，如图 2 所示。

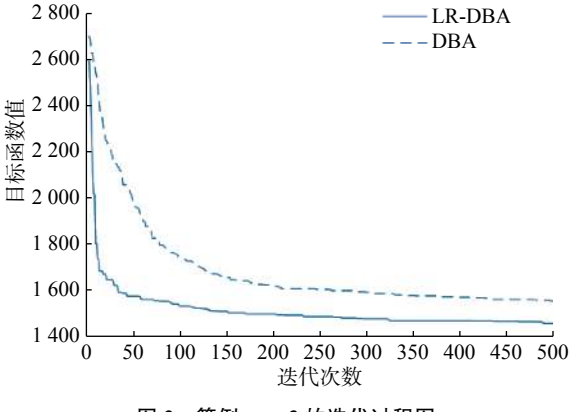

图 2 算例 gapa3 的迭代过程图

Fig.2 Iterative process of the numerical example gapa3

由图 2 可以看出，混合算法在求解过程中迭代的前期收敛速度快，能在较小的迭代次数得到一个较好解，但后期收敛速度较慢。综合比较迭代次数、目标函数值和运行时间，可以看出，拉格朗日松弛算法一方面扩展了蝙蝠算法的搜索范围，但另一方面也使得混合算法的运行时间变长。

5 结束语

应用 BA 算法来解决基本 GAP 问题，并针对 GAP 问题的特点，在基本 BA 算法框架基础上引入遗传算法单基因换位算子，提出 DBA 算法；在此基础上，将 DBA 算法与 LR 算法结合组成 LR-DBA 混合算法，具有较强的全局搜索能力和鲁棒性。通过大量的仿真实验结果，表明 LR-DBA 混合算法求解 GAP 问题的性能优于 DBA 算法。

本文不仅为广义分配问题提供了一种优异的求解方法，同时也拓宽了蝙蝠算法在离散问题上的进一步应用。但是，较现有的求解 GAP 问题的其他智能优化算法(如禁忌搜索和蚁群混合算法[11]、混合遗传算法[22]、禁忌搜索算法[23])，LR-DBA 混合算法的性能较差，并且如仿真实验部分所示，LR-DBA 混合算法在求解效率上也有待提高，这将是本文今后进一步研究的内容。

参考文献：

[1] ROSS G T, SOLAND R M. A branch and bound algorithm

for the generalized assignment problem[J]. Mathematical Programming, 1975, 8(1): 91–103.

[2] NUTOV Z, BENIAMINY I, YUSTER R. A (1−1/e)-approximation algorithm for the generalized assignment problem[J]. Operations Research Letters, 2006, 34(3): 283–288.

[3] HIRAYAMA K. An α-approximation protocol for the generalized mutual assignment problem[C]//Proceedings of the 22nd National Conference on Artificial Intelligence. Vancouver, British Columbia, Canada: ACM, 2007: 744−749.

[4] MUNAPO E, LESAOANA M, NYAMUGURE P, et al. A transportation branch and bound algorithm for solving the generalized assignment problem[J]. International Journal of System Assurance Engineering and Management, 2015, 6(3): 217–223.

[5] CESELLI A, RIGHINI G. A branch-and-price algorithm for the multilevel generalized assignment problem[J]. Operations Research, 2006, 54(6): 1172–1184.

[6] AVELLA P, BOCCIA M, VASILYEV I. A branch-and-cut algorithm for the multilevel generalized assignment problem[J]. IEEE Access, 2013, 1: 475–479.

[7] BAI X J, ZHANG Y J, LIU F T. Particle swarm optimization for two-stage fuzzy generalized assignment problem[M]//HUANG D S, ZHAO Z M, BEVILACQUA V, et al. Advanced Intelligent Computing Theories and Applications. Berlin, Heidelberg: Springer, 2010: 158-165.

[8] TAPKAN P, ÖZBAKIR L, BAYKASOĞLU A. Solving fuzzy multiple objective generalized assignment problems directly via bees algorithm and fuzzy ranking[J]. Expert Systems with Applications, 2013, 40(3): 892–898.

[9] LIU Y Y, WANG S W. A scalable parallel genetic algorithm for the generalized assignment problem[J]. Parallel Computing, 2015, 46: 98–119.

[10] LAGUNA M, KELLY J P, GONZÁLEZ-VELARDE J, et al. Tabu search for the multilevel generalized assignment problem[J]. European Journal of Operational Research, 1995, 82(1): 176–189.

[11] 吴良杰, 魏东, 刘刚. 基于禁忌搜索和蚁群算法的广义分配问题研究[J]. 计算机工程与设计, 2009, 30(15): 3591–3593.

[12] CHEN Y F, LIU Y S, FAN J, et al. Niche-based genetic & ant colony optimization algorithm for generalized assignment problem[J]. Computer Applications, 2005, 25(1): 206–209.

[13] OSMAN I H. Heuristics for the generalised assignment problem: simulated annealing and tabu search approaches[J]. Operations-Research-Spektrum, 1995, 17(4): 211–225.

[14] YANG X S. A new metaheuristic bat-inspired algorithm[M]//GONZÁLEZ J R, PELTA D A, CRUZ C, et al. Nature Inspired Cooperative Strategies for Optimization (NICSO 2010). Berlin, Heidelberg: Springer, 2010: 65-74.

[15] HELD M, KARP R M. The traveling-salesman problem and minimum spanning trees[J]. Operations Research, 1970, 18(6): 1138–1162.

[16] HELD M, KARP R M. The traveling-salesman problem and minimum spanning trees: Part II[J]. Mathematical Programming, 1971, 1(1): 6–25.

[17] 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: 59–71.

[18] 马祥丽, 张惠珍, 马良. 带时间窗物流配送车辆路径问题的蝙蝠算法[J]. 计算机工程与应用, 2016, 52(11): 254–258.

[19] 李佩泽, 王姗姗, 樊岩. 基于改进蝙蝠算法的背包问题求解[J]. 计算机应用研究, 2015, 32(11): 3226–3229.

[20] 李倩, 张惠珍, BELTRANROYO C. 带投资约束 P-中位问题的混合蚁群算法[J]. 计算机应用研究, 2017, 34(6): 1704–1707.

[21] WILSON J M. A genetic algorithm for the generalised assignment problem[J]. Journal of the Operational Research Society, 1997, 48(8): 804–809.

[22] FELTL H, RAIDL G R. An improved hybrid genetic algorithm for the generalized assignment problem[C]// Proceedings of 2004 ACM Symposium on Applied Computing. 2004: 990-995.

[23] DÍAZ J A, FERNÁNDEZ E. A Tabu search heuristic for the generalized assignment problem[J]. European Journal of Operational Research, 2001, 132(1): 22–38.

（编辑：石　瑛）