文章编号: 1001-0920(2019)03-0503-08 DOI: 10.13195/j.kzyjc.2017.1111

求解多维背包问题的二级协作果蝇优化算法

张清勇, 钱 浩†, 雷德明

(武汉理工大学 自动化学院，武汉 430070)

摘 要: 针对多维背包问题(MKP)约束性强和复杂度高的特点,提出一种新型二级协作果蝇优化算法(TCFOA). 提出一级果蝇和二级果蝇的产生机制,将二级果蝇划分为开发用果蝇和探索用果蝇两类以协调开发与探索之间的平衡;设计果蝇交流策略以及基于全局性价比的解的修复补偿机制,并利用二级结构扩大搜索范围、改善一级果蝇的质量,以提高求解质量. 基于MKP两个标准测试集的测试结果和算法性能对比,表明TCFOA在求解MKP方面具有较强的优势.

关键词: 多维背包问题；果蝇优化算法；二级结构；协作策略；全局性价比；全局搜索

中图分类号: TP273 **文献标志码:** A

Two-level cooperative fruit fly optimization algorithm for multidimensional knapsack problem

ZHANG Qing-yong, QIAN Hao†, LEI De-ming

(College of Automation，Wuhan University of Technology，Wuhan 430070，China)

Abstract: A novel two-level cooperative fruit fly optimization algorithm (TCFOA) is proposed for the multidimensional knapsack problem (MKP) with the characteristics such as strong constraints and high complexity. The TCFOA is constructed based on the generation mechanism of primary and secondary fruit flies, and the secondary fruit flies are aivided into two types for exploration and exploitation to implement the balance of exploration and exploration. A communication strategy of fruit flies and a compensation mechanism based on global cost/performance ratio are proposed. A two-level structure is used to expand the search range and improve the quality of the primary fruit flies to obtain high quality solutions. The test experiments are conducted on two sets of MKP instances, and the TCFOA is compared with the methods from literature. The results demonstrate the advantages of the TCFOA in solving the MKP.

Keywords: multidimensional knapsack problem；fruit fly optimization algorithm；two-level structure；cooperative strategy；global cost/performance ratio；global search

0 引 言

多维背包问题(Multidimensional knapsack problem, MKP)[1]是典型的NP-hard问题,其求解目标是在满足多个资源限制的条件下,从对象总体中选出能使总价值最大的一个对象子集. MKP具有较广泛的工程应用背景,可以用来描述货物装载问题[2]、下料问题[3]以及资源配置问题[4]. 早期求解MKP的方法主要为精确算法,如动态规划[5]和分支界定法[2]等,主要针对小规模问题,难以求解大规模MKP,但这类算法能够得到问题的最优解. 对于大规模MKP,通常采用智能算法求解. 目前,这类算法包括遗传算法[6]、粒子群算法[7]、蚁群算法[8]、果蝇优化算法(Fruit fly optimization algorithm, FOA)[9]、差分进化算法[10]、和声搜索算法[11]、狼群算法[12]和分布估计算法[13]等,这些智能算法在解决大规模MKP方面具有较好的表现.

FOA[14]是一种基于果蝇觅食行为的新型智能优化算法. 不同于其他智能算法(如遗传算法、粒子群算法以及蚁群算法等),FOA的优化过程简单、参数少、易于实现,同时也具备较强的搜索能力. 目前,FOA已成功应用于电力负荷预测[15-16]、集合覆盖问题[17]、生产调度问题[18-19]、PID参数寻优[20]、图像分割[21]、

收稿日期: 2017-08-22；**修回日期:** 2018-01-10.

基金项目: 国家自然科学基金项目(61573264)；国家级大学生创新创业训练计划项目(20171049711006).

责任编委: 刘德荣.

作者简介: 张清勇(1984−), 女, 高级实验师, 博士生, 从事智能系统优化与控制、故障诊断与容错控制的研究；雷德明(1968−), 男，教授，博士生导师，从事智能优化与调度、智能交通系统等研究.

†通讯作者. E-mail: m13961881504@163.com.

项目调度问题[22]、传感器网络[23]、负荷频率控制[24]、复杂函数优化问题[25-26]、多工位装配序列规划问题[27]、物流仓储[28]和背包问题[29]等的求解.

由于MKP具有NP-hard和约束性强等特点,无法确定最优解与当前解的差别. 单级果蝇优化算法仅能在群体中心位置附近产生果蝇,且解的质量很大程度上由步距决定[9,30],而二级FOA在一级果蝇的基础上再次进行搜索,并将二级果蝇划分为用于开发的果蝇和用于探索的果蝇,扩大了搜索范围,可避免算法陷入局部最优. 为此,提出一种二级协作果蝇优化算法(Two-level cooperative fruit fly optimization algorithm, TCFOA),设计一级果蝇和二级果蝇的产生策略,提出果蝇间的交流协作策略,设计一种基于全局性价比的修复机制. 大量实验表明,TCFOA在求解MKP方面具有较强的搜索优势和能力.

1 多维背包问题

MKP可描述为

$$\omega = \max \sum_{i=1}^{n} p_i x_i. \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{n} r_{ij} x_i \leqslant b_j,\ j = 1, 2, \cdots, m; \tag{2}$$

$$x_i \in \{0, 1\},\ i = 1, 2, \cdots, n.$$

其中:n为物品的数量;m为资源种类数;p_i为第i个物品的价值;r_{ij}为第i个物品消耗第j个资源的量;b_j为第j个资源的总量;ω为选中物品的总价值;x_i表示一个二进制变量,$x_i = 1$表示第i个物品被选中,$x_i = 0$表示第i个物品没有被选中. 多维背包问题的最优解即在多个有限资源的约束下,从n个物品中选择一组物品,使得物品的总价值ω最大.

2 二级协作果蝇优化算法

2.1 果蝇优化算法描述

FOA是一种模拟果蝇觅食行为的群智能优化算法,其主要依据果蝇敏锐的嗅觉系统对食物味道超强的感知能力. 它首先模拟果蝇的嗅觉搜索过程,使果蝇在解空间内搜索更优解,在所有果蝇完成嗅觉搜索后,再通过视觉搜索更新种群中心位置.

FOA的具体过程如下.

Step1: 初始化种群中心位置.

Step2: 嗅觉搜索,在种群中心位置附近随机产生N个解.

Step3: 计算每个解的适应度函数值.

Step4: 视觉搜索,寻找最优适应度函数值的解,更新种群中心位置.

Step5: 判断是否满足终止条件,若是,则输出最优解,否则跳转至Step2.

2.2 求解MKP的TCFOA

利用多种新策略构建TCFOA. 下面描述TCFOA的具体步骤.

2.2.1 初始群体中心位置

对于MKP,通常采用二进制编码,问题的解由n位的进制串$[x_1, x_2, \cdots, x_n]$表示,其中$x_i = 1$表示第i个物品被选中,$x_i = 0$表示第i个物品没有被选中.

对于物品i,定义全局性价比

$$t_i = \frac{\sum\limits_{i=1}^{n} \dfrac{r_{ij}}{b_j}}{p_i}. \tag{3}$$

其中:t_i表示第i个物品对于每种资源的占用比例之和与该物品价值之比,t_i越大,表明物品的全局性价比越高,单位价值对于整体资源的占用率越低;反之,t_i越小,表明物品单位价值对于整体资源的占用率越高.

初始群体中心位置的产生过程如下.

Step1: 对所有物品计算全局性价比,并按照其全局性价比降序排序.

Step2: 从第1个物品开始依次确定每个物品能否被选中,对于第i个物品,如果加入该物品后所有资源限制条件均能满足,则选择该物品;否则,忽略该物品.

Step3: 如果所有物品均检查完毕,则结束.

2.2.2 一级果蝇与二级果蝇

一级果蝇由离散化后的FOA嗅觉搜索过程产生[9],具体步骤如下.

Step1: 利用初始种群中心位置按文献[9]的嗅觉搜索过程产生fly_1个一级果蝇.

Step2: 对于每个一级果蝇,随机选择L个基因,将选中的基因由0变成1或由1变成0后,进行修复补偿.

现有文献[9, 31]大多采用单级果蝇,并未考虑二级果蝇和两级结构. 单级FOA只在一个群体中心位置附近进行搜索,解的质量较大程度上取决于步距[9]. 由于二级果蝇在一级果蝇的基础上再次进行了搜索,使得TCFOA能在群体中心位置及其周围多个不同位置附近搜索,这样扩大了搜索范围,提高了获得更高质量解的可能性; 同时为了平衡FOA的开发能力和探索能力,避免算法陷入局部最优,将二级果蝇划分为两类,一类果蝇负责开发,另一类果蝇实现探索.

针对每个一级果蝇,设其二级果蝇总数为fly_2,其二级果蝇的搜索过程如下.

Step1: 设置开发步距L_t,探索步距L_e,开发探索比ratio.

Step2: 确定开发果蝇数量$\mathrm{fly}_t = \mathrm{fly}_2 \times \mathrm{ratio}$,探索果蝇数量$\mathrm{fly}_e = \mathrm{fly}_2 \times (1 - \mathrm{ratio})$.

Step3: 以一级果蝇为中心位置,按照文献[9]的方式,产生fly_t个二级开发果蝇和fly_e个二级探索果蝇.

Step4: 对于每个二级开发果蝇和二级探索果蝇,分别随机选择L_t个和L_e个基因,将选中的基因由1变成0或者由0变成1.

Step5: 对所有二级果蝇进行修复补偿. 其中L_e, $L_t = 1, 2, \cdots, n, \mathrm{ratio} \in [0, 1]$.

2.2.3　交流策略

为了有效利用除最好果蝇外的其他果蝇的信息,TCFOA引入多种交流策略,包含一级果蝇交流和二级果蝇交流. 首先进行二级果蝇交流,挑选出总价值最高的二级果蝇作为新的一级果蝇后,再进行一级果蝇交流.

针对每个一级果蝇的所有二级果蝇,相应的交流过程如下.

Step1: 设定交流概率P_1, $P_1 \in [0, 1]$.

Step2: 对所有二级果蝇根据总价值降序排序,选择排在前$\theta\%$的二级果蝇.

Step3: 除最优二级果蝇外,针对每个排名前$\theta\%$的二级果蝇,比较该果蝇与最优二级果蝇的每个基因.若两个果蝇对应位置上的基因不相等,则产生一个区间[0, 1]上的随机数ζ;若$\zeta > P_1$,且该果蝇的基因用最优二级果蝇的同一位置上的基因替代后依然能够满足资源限制条件,则直接替代.

Step4: 选取具有最高总价值的二级果蝇作为对应的一级果蝇.

其中$100 > \theta > 0$,最优二级果蝇为总价值最高的果蝇.

由于排名靠前的二级果蝇质量相对较高,将这些果蝇与最优二级果蝇交流,有助于获得更高质量的二级果蝇,从而提高了一级果蝇质量,故$\theta \leqslant 50$,具体取值需通过实验确定. 通过大量实验,本文设置$\theta = 50$.

在所有二级果蝇执行完交流策略后,一级果蝇为相应二级果蝇中的最优解,一级果蝇再进行交流,以进一步提高解的质量. 一级果蝇的交流策略与二级果蝇交流策略基本步骤大致相同,不同的是,所有的一级果蝇都向最优一级果蝇学习,且交流概率为P_2.

2.2.4　修复补偿策略

FOA产生的解中,部分解违背MKP的资源约束,而有些解在增加部分物品的情况下依然能够满足所有资源约束条件. 为了保证所有解满足资源约束条件的同时,尽可能多地放入物品,提高解的质量,需对上述两类解进行修复补偿.

修复补偿的具体过程如下: 首先,针对违背资源约束条件的解,对解中所有被选中的物品,根据2.2.1节降序排序结果,从全局性价比最低的物品开始,逐个取出解中的物品,直至所有资源约束条件满足;其次,对那些可以添加物品的解,根据2.2.1节降序排序结果,对所有未包含在解中的物品,从全局性价比最高的物品开始,逐个检测每个物品,根据资源约束条件是否满足确定物品能否添加到解中.

上述修复策略与文献[32]的方法一样,也参照了MKP本身的结构特点,同时还不需要像伪效用比修复过程[9]那样,求解新的对偶问题,降低了算法的复杂度,避免面对大规模问题时计算量的显著增大; 同时,有些物品对于某个资源消耗量非常大,但对其他资源消耗量非常小,只考虑一个资源限制条件的修复方法无法全面地衡量物品的性价比,而上述策略利用衡量一个物品对于所有资源约束条件的全局性价比进行修复补偿,有效避免了上述情况的发生.

2.2.5　视觉搜索

TCFOA中视觉搜索分为两个过程: 1)对于每个一级果蝇,在其所有二级果蝇执行完嗅觉搜索和交流过程后,将二级果蝇中的最优解作为新的一级果蝇; 2)在所有的一级果蝇间的交流过程执行完毕后,寻找出总价值最高的一级果蝇作为下一次迭代的种群中心位置. TCFOA的视觉搜索过程如图1所示,其中一级果蝇表示其二级果蝇执行视觉搜索后产生的新的一级果蝇,种群中心位置代表本次迭代产生的最优解,同时作为下一次迭代的种群中心位置.

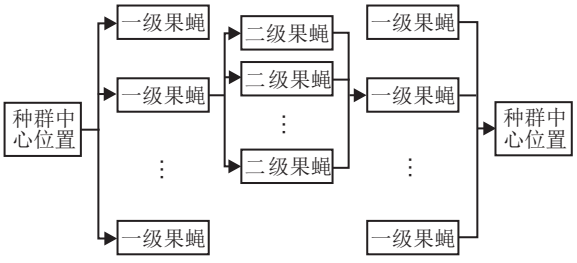

图1　TCFOA视觉搜索过程

2.2.6　算法描述

TCFOA的流程如图2所示. TCFOA利用一级和二级果蝇在群体中心位置和多个其他位置进行搜索,若以基因从0变成1或者从1变成0,以及修复补偿时

取物品或添加物品作为基本操作,算法的时间复杂性为 $O(G \times \mathrm{fly}_1 \times \mathrm{fly}_2 \times n)$,其中$G$为最大迭代次数.

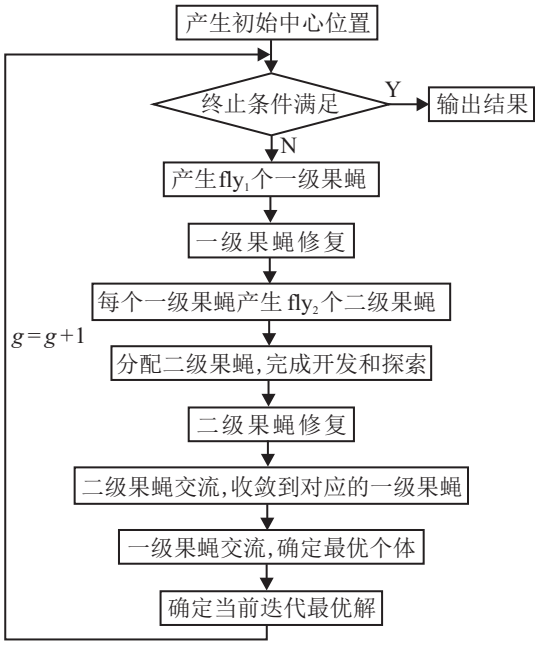

图 2 TCFOA流程图

3 仿真测试及比较

为了测试TCFOA的性能,采用两组标准测试集对算法进行测试,测试集1来自OR Library(http://people.brunel.ac.uk/~mastjjb/jeb/info.html),命名规则为$m-n-xx$,其中xx表示问题的编号. 测试集2(http://hces.bus.olemiss.edu/tools.html),包含了11组中大规模的问题,命名规则为Mk_gkxx. TCFOA算法在Visual C++6.0集成环境中编写,运行环境为Windows10操作系统,Intel Core i5-5200U处理器(2.20 GHz). 实验中每个问题均独立运行20次,如非特殊说明,每次测试允许的最大迭代次数为2 000.

3.1 参数设置

TCFOA的参数共有8个,分别为一级果蝇数量fly_1、二级果蝇数量fly_2、开发步距L_t、探索步距L_e、开发探索比ratio、一级果蝇的搜索步距L和两个交流概率. 相比前5个参数,后3个参数的设置相对容易,例如,一级果蝇的搜索步距可以单独通过实验测试确定,故本文仅针对前5个参数进行实验设计(Design of experiment, DOE)[33].

以Mk_gk06为例子,5个参数均取4个水平值,建立$L_{16}(4^5)$的正交实验表,每种参数组合算法均独立运行20次,将计算产生结果的平均价值(AVG)作为评价指标,如表1所示.

表1 正交表及AVG统计

参数组合	水平					AVG
	fly_1	fly_2	L_t	L_e	ratio	
1	80	900	2	7	0.4	7 652.56
2	80	1 000	3	8	0.5	7 654.32
3	80	1 100	4	9	0.6	7 654.38
4	80	1 200	5	10	0.7	7 651.60
5	100	900	3	9	0.7	7 651.29
6	100	1 000	2	10	0.6	7 651.40
7	100	1 100	5	7	0.5	7 654.50
8	100	1 200	4	8	0.4	7 654.85
9	120	900	4	10	0.5	7 652.30
10	120	1 000	5	9	0.4	7 653.25
11	120	1 100	2	8	0.7	7 651.00
12	120	1 200	3	7	0.6	7 652.20
13	140	900	5	8	0.6	7 655.10
14	140	1 000	4	7	0.7	7 654.85
15	140	1 100	3	10	0.4	7 651.95
16	140	1 200	2	9	0.5	7 651.25

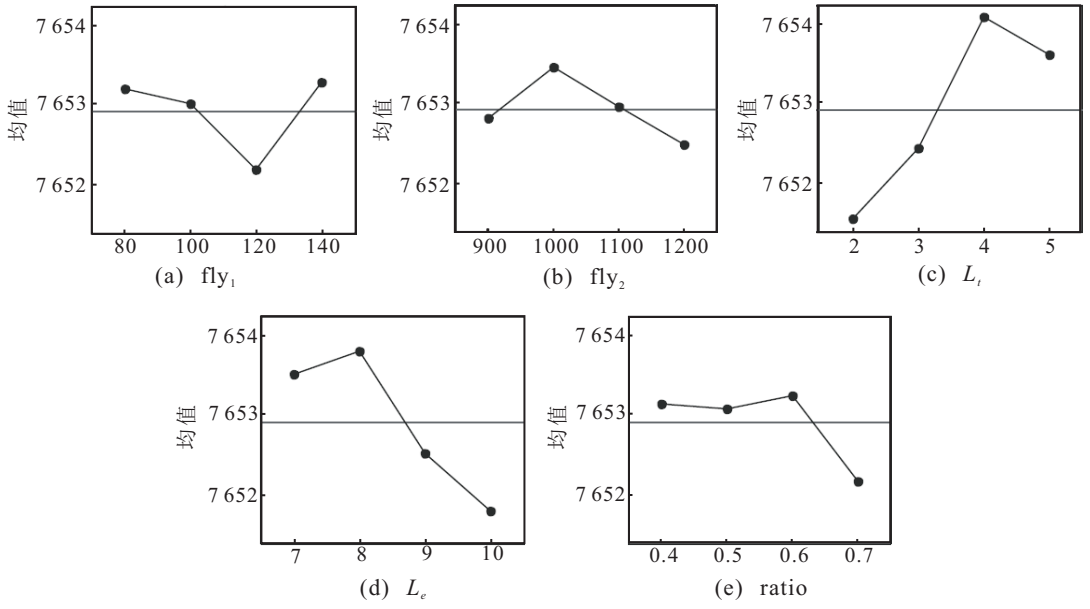

图 3 均值主效应

表2　TCFOA与bFOA1、bFOA2的结果对比

问题编号	维度	已知最优解	bFOA1			bFOA2			TCFOA		
			MIN.DEV	AVG.DEV	VAR. DEV	MIN. DEV	AVG. DEV	VAR. DEV	MIN. DEV	AVG. DEV	VAR. DEV
Mk_gk01	100.15	3 766	0.239 0	0.305 3	0.033 2	0.557 6	0.755 4	0.330 5	**0**	**0.078 3**	**0.001 4**
Mk_gk02	100.25	3 958	0	0.198 3	0.230 6	0.656 9	0.851 8	0.331 7	**0**	**0.036 6**	**0.002 5**
Mk_gk03	150.25	5 650	0.070 8	0.158 4	0.094 6	0.796 5	0.915 0	0.412 6	0	**0.053 9**	**0.000 2**
Mk_gk04	150.50	5 764	0.052 0	0.194 3	0.356 7	0.867 5	1.027 9	0.480 4	**0.017 3**	**0.086 7**	**0.000 6**
Mk_gk05	200.25	7 557	**0.052 9**	0.119 8	0.078 7	1.005 7	1.193 0	0.590 5	0.066 1	**0.111 1**	**0.000 5**
Mk_gk06	200.50	7 672	**0.117 3**	0.204 6	0.121 4	0.847 2	0.980 2	0.424 4	0.130 3	**0.164 2**	**0.000 3**
Mk_gk07	500.25	19 215	0.052 0	0.082 2	0.047 2	1.431 2	1.519 4	0.578 4	**0.036 4**	**0.075 7**	**0.000 1**
Mk_gk08	500.50	18 801	**0.085 1**	**0.140 2**	0.126 7	1.287 2	1.365 9	0.355 1	0.228 7	0.243 0	**0.000 1**
Mk_gk09	1 500.25	58 085				2.174 4	2.281 6	0.926 7	**0.072 3**	**0.102 0**	**0.000 3**
Mk_gk10	1 500.50	57 292				1.743 7	1.790 5	0.414 8	**0.205 9**	**0.247 1**	**0.000 1**
Mk_gk11	2 00.100	95 231				1.503 7	1.573 8	0.578 0	**0.243 6**	**0.255 6**	**0**

利用数据分析软件Minitab得到各个参数的排秩和主效应图,排秩由高到低依次为开发步距L_t、探索步距L_e、一级果蝇数量fly_1、开发探索比ratio以及二级果蝇数量fly_2的主效应如图3所示. 由分析可得,开发步距L_t为最重要的参数,一个合适的开发步距有助于增加算法的开发能力,增大获得最优解的成功率. 同时,探索步距L_t的合理设定也能使算法的搜索能力有效提高,减小陷入局部最优的概率. 对于fly_1, fly_2和ratio,设定一个相对适中的值即可.

根据DOE实验的结果,对测试集1以及测试集2中的MK_gk01～MK_gk06,设置一级果蝇数量为80,二级果蝇数量为1 100,开发步距为4,探索步距为8,开发探索比为0.6,交流概率$P_1 = P_2 = 0.5$,一级果蝇搜索步距为4. 对于MK_gk07～MK_gk11这几个规模较大的问题,重新调整一级果蝇和二级果蝇的数量,设置一级果蝇数量为50,二级果蝇数量为500,最大迭代次数调整为7 040,其余参数保持不变,这样所得到的结果优于利用上一组参数值所获得的解.

3.2　结果比较与分析

为了验证TCFOA算法二级结构的有效性,将其与文献[9]中的bFOA算法进行比较. bFOA为单级果蝇算法,该算法利用概率向量生成解,用差分进化公式更新概率向量,并采用两种不同的修复策略对不同规模的问题进行修复. bFOA1和bFOA2的区别仅为所使用的修复策略不同,bFOA1更适用于中小规模问题,而bFOA2在解决大规模问题时有更好的表现[9]. 表2给出了TCFOA与bFOA1和bFOA2的比较结果,分别统计了测试结果的最优偏差MIN.DEV,平均偏差AVG.DEV和偏差方差VAR.DEV.

由表2可以看出,在最优解上,TCFOA与bFOA1持平,但在平均值和方差上表现优于bFOA1,且TCFOA在最优解、平均值和方差上均远优于bFOA2. 相比于bFOA的单级结构,TCFOA的二级结构能够有效提升解的质量.

为了验证算法解决MKP的能力,将其与3种算法进行比较. 其中算法1是HEDA[13],该算法利用概率向量产生解,并采用伪效用比修复机制和基于问题本身结构特点的修复机制对解进行修复,是较为经典的解决MKP的算法之一,常用于算法的测试和比较[8-9]. 算法2为HHS[11],该算法结合了和声搜索和果蝇优化算法,同时引用了两种修复策略,是最新提出的一种能够有效解决MKP的算法. 算法3为ALMKP,该算法构建了低维MKP核问题[8],并利用ACO求解.

表3和表4分别给出了TCFOA与HEDA以及TCF-OA和HHS的对比结果,其中表3采用BEST、

表3　TCFOA与HEDA的结果对比

问题编号	已知最优解	HEDA			TCFOA		
		BEST	AVG	STD	BEST	AVG	STD
5.100.00	24 381	24 381	24 381	0	**24 381**	**24 381**	**0**
5.100.01	24 274	24 274	24 274	0	**24 274**	**24 274**	**0**
5.100.02	23 551	23 551	22 541	5.7	**23 551**	**23 551**	**0**
5.100.03	23 534	23 534	23 524	11	**23 534**	**23 534**	**0**
5.100.04	23 991	23 991	23 976	16.3	**23 991**	**23 991**	**0**
5.100.05	24 613	24 613	24 612	5.7	**24 613**	**24 613**	**0**
5.100.06	25 591	25 591	25 591	0	**25 591**	**25 591**	**0**
5.100.07	23 410	23 410	23 375	7.1	**23 410**	**23 410**	**0**
5.100.08	24 216	24 216	24 211	5.9	**24 216**	**24 216**	**0**
5.100.09	24 411	24 411	24 411	0	**24 411**	**24 411**	**0**
10.100.00	23 064	23 064	23 051	2.4	**23 064**	**23 064**	**0**
10.100.01	22 801	22 801	22 739	25.2	**22 801**	**22 801**	**0**
10.100.02	22 131	22 131	22 131	0	**22 131**	**22 131**	**0**
10.100.03	22 772	22 772	**22 772**	**0**	**22 772**	22 769.8	3.8
10.100.04	22 751	22 751	22 620	24.7	**22 751**	**22 746.2**	**21.1**
10.100.05	22 777	22 777	22 683	30	**22 777**	**22 759.1**	**27.9**
10.100.06	21 875	21 875	21 823	**9**	**21 875**	**21 871.6**	10.2
10.100.07	22 635	22 635	22 552	27.7	**22 635**	**22 630.8**	**18.3**
10.100.08	22 511	22 511	22 424	5	**22 511**	**22 511**	**0**
10.100.09	22 702	22 702	**22 702**	**0**	**22 702**	22 701.4	1.43

表4 TCFOA与HHS的计算结果

问题编号	已知最优解	HHS				TCFOA			
		AVG	MIN.DEV	AVG.DEV	VAR.DEV	AVG	MIN.DEV	AVG.DEV	VAR.DEV
5.100.00	24 381	24 378.4	0	0.010 7	0.046 5	**24 381**	**0**	**0**	**0**
5.100.01	24 274	24 264.4	0	0.039 5	0.032 3	**24 274**	**0**	**0**	**0**
5.100.02	23 551	23 532.75	0	0.077 5	0.030 4	**235 51**	**0**	**0**	**0**
5.100.03	23 534	23 489.9	0.029 7	0.187 4	0.068	**23 534**	**0**	**0**	**0**
5.100.04	23 991	23 964.85	0	0.109	0.020 9	**23 991**	**0**	**0**	**0**
5.100.05	24 613	24 603.05	0	0.040 4	0.068 8	**24 613**	**0**	**0**	**0**
5.100.06	25 591	25 538.8	0	0.205 2	0.118 4	**25 591**	**0**	**0**	**0**
5.100.07	23 410	23 368.8	0	0.176	0.031 8	**23 410**	**0**	**0**	**0**
5.100.08	24 216	24 216	0	0	0	**24 216**	**0**	**0**	**0**
5.100.09	24 411	24 379.8	0	0.127 8	0.099 9	**24 411**	**0**	**0**	**0**
10.100.00	23 064	23 041	0.030 4	0.099 7	0.097 4	**23 064**	**0**	**0**	**0**
10.100.01	22 801	22 739.55	0	0.269 5	0.116 1	**22 801**	**0**	**0**	**0**
10.100.02	22 131	22 096.25	0	0.157	0.143 5	**221 31**	**0**	**0**	**0**
10.100.03	22 772	22 753.85	0.039 5	0.079 7	0.092 8	**22 769.75**	**0**	**0.009 8**	**0.000 2**
10.100.04	22 751	22 657.05	0.237 3	0.412 9	0.194 1	**22 746.15**	**0**	**0.021 3**	**0.008 6**
10.100.05	22 777	22 717.42	0	0.261 6	0.110 7	**22 759.1**	**0**	**0.078 5**	**0.015 0**
10.100.06	21 875	21 814.9	0.185 3	0.274 7	0.094 1	**21 871.6**	**0**	**0.015 5**	**0.002 1**
10.100.07	22 635	22 518.7	0.371 1	0.513 8	0.032 7	**22 630.8**	**0**	**0.018 5**	**0.006 5**
10.100.08	22 511	22 416.75	0.324 3	0.418 7	0.055 7	**22 511**	**0**	**0**	**0**
10.100.09	22 702	22 645.78	0	0.247 6	0.078 9	**22 701.4**	**0**	**0.002 6**	**0.000 1**

表5 TCFOA与AL_MKP的结果对比

问题编号	维度	已知最优解	AL_MKP				TCFOA			
			MIN. DEV	AVG. DEV	STD. DEV	T/s	MIN. DEV	AVG. DEV	STD. DEV	T/s
Mk_gk01	100.15	3 766	0.000 0	**0.076 5**	0.044 9	**1.099 4**	**0**	0.078 3	**0.037 9**	8.019 2
Mk_gk02	100.25	3 958	0.000 0	**0.000 0**	**0.000 0**	**0.502 4**	**0**	0.036 6	0.050 1	8.937 4
Mk_gk03	150.25	5 650	**-0.017 7**	**0.029 0**	0.017 6	**1.810 0**	0	0.053 9	**0.015 3**	8.925 7
Mk_gk04	150.50	5 764	0.052 0	0.143 0	**0.023 6**	**2.590 5**	**0.017 3**	**0.086 7**	0.026 3	9.317 5
Mk_gk05	200.25	7 557	**0.026 5**	**0.070 9**	0.029 8	**3.096 4**	0.066 1	0.111 1	**0.023 8**	10.975 8
Mk_gk06	200.50	7 672	0.156 4	0.231 0	0.027 3	**5.075 8**	**0.130 3**	**0.164 2**	**0.018 1**	13.647 2
Mk_gk07	500.25	19 215	0.119 7	0.139 7	**0.007 9**	27.205 8	**0.036 4**	**0.075 7**	0.013 8	**26.917 5**
Mk_gk08	500.50	18 801	0.446 8	0.475 1	0.013 0	**24.861 8**	**0.228 7**	**0.243 0**	**0.010 7**	28.003 5
Mk_gk09	1 500.25	58 085	0.086 1	0.125 8	**0.009 3**	159.527 5	**0.072 3**	**0.102 0**	0.017 8	**100.371 2**
Mk_gk10	1 500.50	57 292	0.315 9	0.360 7	0.012 9	167.815 0	**0.205 9**	**0.247 1**	**0.011 7**	**107.684 7**
Mk_gk11	2 500.100	95 231	0.409 5	0.419 9	0.005 8	525.728 8	**0.243 6**	**0.255 6**	**0.004 9**	**303.010 7**

AVG和STD三个指标分别描述最好解、平均解和标准差,表5给出了TCFOA与AL-MKP的测试比较结果,同时给出了最优解的首达时间T.

由表2和表3可知,对于测试集1中的被测问题,TCFOA均能够得到最优解,同时能够保证平均解非常接近最优解,TCFOA解的质量和稳定性远高于HEDA和HHS. 如表5所示,对于测试集2,在最优解方面,TCFOA仅关于MK_gk03和MK_gk05的结果不如AL_MKP,而关于其他问题的最好解均大于或等于AL_MKP的相应结果,尤其是关于大规模问题

MK_gk10和MK_gk11,TCFOA的优势更大.在平均值方面,TCFOA关于7个问题的结果占优,在标准偏差方面,TCFOA关于7个问题的结果也优于AL_MKP,这表明TCFOA有非常好的稳定性和鲁棒性;在最优解的首达时间上,对于中小规模问题,TCFOA不如AL_MKP,但是对于大规模问题,TCFOA远远优于AL_MKP,这表明TCFOA对于处理大规模问题具有较强的优势. 总之,TCFOA能够有效解决MKP,且具有较强的搜索优势.

4 结 论

为了解决MKP,本文提出了一种新型二级协作果蝇优化算法,设计了果蝇开发和探索平衡方法、一级果蝇和二级果蝇的产生机制、果蝇交流策略以及基于全局性价比的解的修复补偿机制,并利用两级结构改善搜索结果. 基于两个不同测试集的测试结果表明,TCFOA能够有效解决MKP,对于不同规模的问题均有较高的优化质量.

进一步的工作是针对MKP构造动态分配果蝇开发和探索数量的机制,进而得到更加有效的果蝇优化算法,并研究FOA对于其他组合优化问题的应用,如并行机调度问题等.

参考文献(References)

[1] Chu P C, Beasley J E. A genetic algorithm for the multidimensional knapsack problem[J]. J of Heuristics, 1998, 4(1): 63-86.

[2] Wei S. A branch and bound method for the multiconstraint zero-one knapsack problem[J]. J of the Operational Research Society, 1979, 30(4): 369-378.

[3] Gilmore P C, Gomory R E. The theory and computation of knapsack functions[J]. Operations Research, 1966, 14(6): 1045-1074.

[4] Vanderster D C, Dimopoulos N J, Parra-Hernandez R, et al. Resource allocation on computational grids using a utility model and the knapsack problem[J]. Future Generation Computer Systems, 2009, 25(1): 35-50.

[5] Toth P. Dynamic programming algorithms for the zero-one knapsack problem[J]. Computing, 1980, 25(1): 29-45.

[6] Ünal A N, Kayakutlu G. A partheno-genetic algorithm for dynamic 0-1 multidimensional knapsack problem[J]. RAIRO-Operations Research, 2015, 50(1): 47-66.

[7] Chih M, Lin C, Chern M, et al. Particle swarm optimization with time-varying acceleration coefficients for the multidimensional knapsack problem[J]. Applied Mathematical Modelling, 2014, 38(4): 1338-1350.

[8] 任志刚, 赵松云, 黄姗姗, 等. 求解多维背包问题的蚁群-拉格朗日松弛混合优化算法[J]. 控制与决策, 2016, 31(7): 1178-1184.
(Ren Z Z, Zhao S Y, Huang S S, et al. Ant colony-lagrangian relaxation optimization algorithm for solving multidimensional backpack problem[J]. Control and Decision, 2016, 31(7): 1178-1184.)

[9] Wang L, Zheng X, Wang S. A novel binary fruit fly optimization algorithm for solving the multidimensional knapsack problem[J]. Knowledge-Based Systems, 2013, 48(2): 17-23.

[10] Tasgetiren M F, Pan Q K, Kizilay D, et al. A differential evolution algorithm with variable neighborhood search for multidimensional knapsack problem[M]. Sendai: Institute of Electrical and Electronics Engineers Inc, 2015: 2797-2804.

[11] Zhang B, Pan Q, Zhang X, et al. An effective hybrid harmony search-based algorithm for solving multidimensional knapsack problems[J]. Applied Soft Computing, 2015, 29(C): 288-297.

[12] 吴虎胜, 张凤鸣, 战仁军, 等. 利用改进的二进制狼群算法求解多维背包问题[J]. 系统工程与电子技术, 2015, 37(5): 1084-1091.
(Wu H S, Zhang F M, Zhan R J, et al. Using the improved binary wolves algorithm to solve the multidimensional backpack problem[J]. Systems Engineering and Electronics, 2015, 37(5): 1084-1091.)

[13] Wang L, Wang S, Xu Y. An effective hybrid EDA-based algorithm for solving multidimensional knapsack problem[J]. Expert Systems with Applications, 2012, 39(5): 5593-5599.

[14] Dai H, Zhao G, Lu J, et al. Comment and improvement on “A new fruit fly optimization algorithm: Taking the financial distress model as an example”[J]. Knowledge-Based Systems, 2014, 59: 159-160.

[15] Hu R, Wen S, Zeng Z, et al. A short-term power load forecasting model based on the generalized regression neural network with decreasing step fruit fly optimization algorithm[J]. Neurocomputing, 2017, 221(C): 24-31.

[16] Li H Z, Guo S, Li C J, et al. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm[J]. Knowledge-Based Systems, 2013, 37(2): 378-387.

[17] Crawford B, Soto R, Torres-Rojas C, et al. A binary fruit fly optimization algorithm to solve the set covering problem[C]. Int Conf on Computational Science and Its Applications. Bainf: Springer, 2015: 411-420.

[18] 郑晓龙, 王凌, 王圣尧. 求解置换流水线调度问题的混合离散果蝇算法[J]. 控制理论与应用, 2014, 31(2): 159-164.
(Zheng X L, Wang L, Wang S R. A hybrid discrete

drosophila algorithm for solving displacement pipeline scheduling problem[J]. Control Theory & Applications, 2014, 13(2): 159-164.)

[19] Li J Q, Pan Q K, Mao K. A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems[J]. IEEE Trans on Automation Science & Engineering, 2016, 13(2): 932-949.

[20] Song J, Pan H. PID control parameters optimize based on an immune fruit fly optimization algorithm[C]. Chinese Control and Decision Conf. Yinchuan, 2016: 6383-6388.

[21] 孙立新, 张栩之, 邓先瑞, 等. 自适应果蝇算法优化模糊均值聚类算法图像分割[J]. 控制工程, 2016, 23(4): 494-499.
(Sun L X, Zhang X Z, Deng X R, et al. Adaptive fruit fly algorithm optimization fuzzy mean clustering algorithm image segmentation[J]. Control Engineering, 2016, 23(4): 494-499.)

[22] 郑晓龙, 王凌. 随机资源约束项目调度问题基于序的果蝇算法[J]. 控制理论与应用, 2015, 32(4): 540-545.
(Zheng X L, Wang L. Stochastic resource constrained project scheduling problem based on the order of fruit fly algorithm[J]. Control Theory & Applications, 2015, 32(4): 540-545.)

[23] Dey A, Sarkar T, Ali S. Fruit fly algorithm based clustering protocol in wireless sensor networks[C]. Int Conf on Electrical and Computer Engineering. Dhaka: IEEE, 2017: 295-298.

[24] Huang C, Li Y. Linear active disturbance rejection control approach for load frequency control problem using diminishing step fruit fly algorithm[M]. Xiamen: Springer Singapore, 2016: 9-18.

[25] 王林, 吕盛祥, 曾宇容. 果蝇优化算法研究综述[J]. 控制与决策, 2017, 32(7): 1153-1162.
(Wang L, Lv S X, Zeng Y R. A review of fruit fly optimization algorithm[J]. Control and Decision, 2017, 32(7): 1153-1162.)

[26] Pan Q K, Sang H Y, Duan J H, et al. An improved fruit fly optimization algorithm for continuous function optimization problems[J]. Knowledge-Based Systems, 2014, 62(5): 69-83.

[27] 袁文兵, 常亮, 徐周波, 等. 基于果蝇优化算法的多工位装配序列规划[J]. 计算机科学, 2017, 44(4): 246-251.
(Yuan W B, Chang L, Xu Z B, et al. Multi-station assembly sequence planning based on drosophila optimization algorithm[J]. Computer Science, 2017, 44(4): 246-251.)

[28] Pan W T, Zhu W Z, Ma F X, et al. Modified fruit fly optimization algorithm of logistics storage selection[J]. Int J of Advanced Manufacturing Technology, 2017, 93(6): 1-12.

[29] Qian H, Zhang Q Y, Lei D M, et al. A cooperated fruit fly optimization algorithm for knapsack problem[C]. Chinese Automation Congress. Ji'nan: IEEE, 2018: 591-595.

[30] 王凌, 郑晓龙. 果蝇优化算法研究进展[J]. 控制理论与应用, 2017, 34(5): 557-563.
(Wang L, Zheng X L. Research progress of fruit fly optimization algorithm[J]. Control Theory & Applications, 2017, 34(5): 557-563.)

[31] Wang L, Liu R, Liu S. An effective and efficient fruit fly optimization algorithm with level probability policy and its applications[J]. Knowledge-Based Systems, 2016, 97(C): 158-174.

[32] 王凌, 王圣尧, 方晨. 一种求解多维背包问题的混合分布估计算法[J]. 控制与决策, 2011, 26(8): 1121-1125.
(Wang L, Wang S R, Fang C. Hybrid distribution estimation algorithm for solving multidimensional backpack problem[J]. Control and Decision, 2011, 26(8): 1121-1125.)

[33] Montgomery D C. Design and analysis of experiments[J]. J of the American Statistical Association, 2000, 16(2): 241-242.

(责任编辑：孙艺红)