文章编号: 1001-0920(2019)07-1355-10 DOI: 10.13195/j.kzyjc.2017.1747

基于子种群拉伸操作的精英共生生物搜索算法

王艳娇†, 马 壮

(东北电力大学 电气工程学院，吉林省 吉林市 132000)

摘 要: 针对共生生物搜索算法存在易早熟、收敛速度慢等缺陷,提出一种基于子种群拉伸操作的精英共生生物搜索算法. 在“互利共生”阶段,根据适应度值将种群划分为两个子种群,设计有针对性的进化策略,使两个子种群分别负责开发和探索,有效地平衡算法的收敛速度与精度;在“偏利共生”阶段,利用最优个体的方向性引导信息,引入拉伸因子和差分扰动向量,并修正个体更新模式,从而在提高算法收敛速度的同时保证种群的多样性;模拟寄生体和宿主的生物关系,提出精英“寄生”机制,进一步平衡算法在整个迭代过程中的探索与开发能力. 对与标准共生生物算法、改进后的共生生物搜索算法以及其他4个群智能进化算法在17个函数上的测试结果进行比较分析,结果表明所提出的算法精度更佳,收敛速度优势明显.

关键词: 共生生物搜索；子种群策略；拉伸操作；精英机制；函数优化；自适应搜索

中图分类号: TP273 **文献标志码:** A

Elite symbiotic organisms search algorithm based on subpopulation stretching operation

WANG Yan-jiao†, MA Zhuang

(School of Electrical Engineering，Northeast Electric Power University，Jilin 132000，China)

Abstract: An elite symbiotic organisms search (SOS) algorithm based on subpopulation stretching operation is proposed to solve the problems of premature and slow convergence in SOS. In the mutualism phase, the population is divided into two subpopulations according to the fitness value: One is responsible for development and the other is aimed at exploration. The targeted evolutionary strategy is designed for each subpopulation, which makes the algorithm keep a good balance between convergence speed and accuracy. In the commensalism phase, the individual updating mode is modifred by using the directional information of the optimal individual and introducing the stretching factor and the difference perturbation vector, which improves the convergence speed of the algorithm and guarantees the diversity of the population. The “parasitism” mechanism of the elite, which is proposed by simulating the biological relationships between parasites and hosts, keeps a further balance between development and exploration. The comparison examinations of the standard SOS, the improved SOS and other four intelligent evolutionary algorithms on 17 functions indicate that the proposed algorithm has better accuracy and obvious advantage of convergence speed.

Keywords: SOS；subpopulation；stretching operation；elite mechanism；function optimization；adaptive search

0 引 言

2014年,Cheng等[1]模拟自然界中生物通过相互作用以提高自身生存能力的行为,提出一种新型群智能优化算法——共生生物搜索算法(Symbiotic organisms search, SOS). 文献[1]将SOS算法与遗传算法(GA)[2]、差分进化算法(DE)[3]、粒子群算法(PSO)[4]在26个标准测试函数上进行了大量实验比较,结果表明,SOS算法在收敛速度及收敛精度方面具有明显优势. 但与其他群智能优化算法类似[5-6],SOS算法也存在一定缺陷,如在搜索中后期收敛速度缓慢、处理高维多峰复杂优化问题时易陷入局部最优等. 为此,已有学者进行了初步改进,如文献[7]提出了多策略自适应共生生物搜索算法,根据适应度将种群分为3个群体,针对各群体分别设计不同的搜索策略以实现不同功能,并提出了一种基于实时信息反馈的混合搜索策略,使搜索具有一定的自适应性,虽然在种群多样性方面起到了一定的作用,但是后期收敛速度效果仍然有待提升;文献[8]提出了基于旋转学习策略的

收稿日期: 2017-12-22；**修回日期:** 2018-03-23.

基金项目: 国家自然科学基金项目(61501107, 61603073)；吉林省教育厅“十三五”科学技术研究项目(吉教科合字[2016]第95号)；吉林市科技创新发展计划项目(201750219).

责任编委: 夏元清.

†通讯作者. E-mail: wangyanjiao1028@126.com.

共生生物搜索算法,通过对所选取个体的每一维度都进行旋转学习,增强了算法跳出局部最优的能力,在收敛速度以及收敛精度方面都有所提高,但是算法实现较为复杂.

为进一步提升SOS算法的优化性能,本文提出一种基于子种群拉伸操作的精英共生生物搜索算法(Elite symbiotic organisms search algorithm based on subpopulation stretching operation, SPS-SOS). 主要创新点如下:首先,在“互利共生”阶段,依据个体适应度值将种群划分为优秀种群和劣等种群,并根据进化需求,分别设计有针对性的进化策略,各自负责开发或探索任务;其次,在“偏利共生”阶段,引入差分扰动,弥补算法开发能力不足的缺陷,对未得到改善的个体,利用拉伸因子将其直接牵引至最优个体附近搜索,进一步提升算法的收敛速度;最后,在寄生阶段,模拟生物界宿主与寄生体之间的作用关系,以精英个体作为“宿主”,设计精英“寄生”机制,以此平衡算法的开发和探索能力. 将本文算法与SOS算法、多策略自适应共生生物搜索算法(Multi strategy adaptive symbiotic organisms search, MSASOS)和其他4种应用较为广泛的群智能进化算法在17个标准测试函数上进行大量实验,结果表明本文所提算法的函数优化性能最优,特别是在收敛速度上具有明显优势.

1 标准共生生物搜索算法

在自然界中,生物之间的相互作用可能会使相互作用的两个生物都获利,单独某个个体在不损害其他生物的利益下使自身获利,或通过损害其他个体来使其自身获利. SOS算法模仿上述关系行为,最终形成“互利共生”、“偏利共生”和“寄生”3个进化阶段,引导个体逐渐进化. SOS算法的关键操作具体如下.

1.1 种群初始化

假设种群数目为N,个体的搜索范围上限和下限分别为U和L,按下式随机生成N个初始个体X_i:

$$X_i = L + \mathrm{rand}(1, D) \times (U - L). \tag{1}$$

其中:X_i表示种群中第$i(i = 1, 2, \cdots, N)$个生物个体,rand表示$[0, 1]$之间的随机数,D表示所优化问题的维数.

1.2 互利共生

对于种群中的每个个体X_i,随机选择一个其他个体$X_j(i, j \in \{1, 2, \cdots, N\}, j \neq i)$与其自身按照下式进行互利共生,生成2个新的个体:

$$\begin{cases} X_{i\mathrm{new}} = X_i + \mathrm{rand}(0, 1) \times (X_{\mathrm{best}} - \mathrm{MV} \times \mathrm{BF}_1), \\ X_{j\mathrm{new}} = X_j + \mathrm{rand}(0, 1) \times (X_{\mathrm{best}} - \mathrm{MV} \times \mathrm{BF}_2). \end{cases} \tag{2}$$

其中:$\mathrm{rand}(0, 1)$为[0,1]之间的随机数;X_{best}为当前迭代次数下的最优个体;$\mathrm{MV} = (X_i + X_j)/2$为“互利向量”,代表生物体之间的关系特征;$\mathrm{BF}_1$、$\mathrm{BF}_2$为获益因子,随机选择1或2.

经式(2)产生新的个体$X_{i\mathrm{new}}$和$X_{j\mathrm{new}}$,通过与X_i和X_j进行适应度值的比较,最终保留$X_{i\mathrm{new}}$与X_i、$X_{j\mathrm{new}}$与X_j中的较优个体,替换X_i和X_j.

1.3 偏利共生

种群中的个体X_i按照下式进行“偏利共生”策略:

$$X_{i\mathrm{new}} = X_i + \mathrm{rand}(-1, 1) \times (X_{\mathrm{best}} - X_j). \tag{3}$$

其中:X_j为种群中与X_i不同的个体,即$i, j \in \{1, 2, \cdots, N\}, j \neq i$;$X_{\mathrm{best}}$为当前种群中的最优个体;$\mathrm{rand}(-1, 1)$为$[-1, 1]$之间的随机数. 通过式(3)新产生的个体$X_{i\mathrm{new}}$经适应度值比较之后决定是否要替换$X_i$.

由偏利共生的策略可见:与“互利共生”阶段类似,每个个体都从种群中随机选择一个其他个体与其自身相互作用;与之不同的是,二者相互作用仅使X_i受益,对于X_j自身的发展既无利益也无损害.

1.4 寄 生

在定义域内产生一个或者多个随机数(不多于基因位位数),修改个体X_i相应的基因位,形成新个体Parasite_Vector(简称P_V),与随机选择的个体$X_j(i, j \in \{1, 2, \cdots, N\}, j \neq i)$(称为“宿主”)进行适应度值比较. 若P_V的适应度值更优,则“宿主”X_j将被P_V取代,否则X_j将被视为免疫个体被保留.

2 基于子种群拉伸操作的精英共生生物搜索算法

大量实验证实,进化算法的不同进化策略适合于不同的求解领域,某一策略并不能适用于求解所有优化问题. 按此思想,SOS算法在“互利共生”、“偏利共生”以及“寄生”阶段采用不同的进化策略,以提升其鲁棒性. 但大量实验表明,在求解不同优化问题时,SOS算法仍然存在易陷入局部最优,收敛速度和收敛精度有待提高等不足. 本文通过深入研究,发现了存在上述不足的根本原因,并从3个方面进行改进,提出一种基于子种群拉伸操作的精英共生生物搜索算法. 下面进行详细介绍.

2.1 子种群分类的“互利共生”搜索过程

进化算法的核心是平衡探索能力和开发能力之间的矛盾. 其中,探索能力代表算法勘探整个搜索空间以寻找那些可能的最优区域;开发能力则是将搜索的重点放在具有较优适应度值的个体上. 在SOS

算法的“互利共生”阶段,所有个体都依靠式(2)进行自我更新,其作用实质是,最优个体提供方向性进化信息,引导各个体快速靠近最优个体. 显然,随着进化的演进,各个体均会聚集在最优个体所在区域,与最优个体极为相似. 若该最优个体并非全局最优,而是局部最优,则SOS算法不依靠其他机制很难跳出,即出现早熟收敛现象. 综上所述,SOS算法的探索能力较弱,这是导致算法容易出现早熟的根源所在. 因此,如何在探索与开发之间进行有效的权衡,是提高SOS算法性能的关键.

为了有效权衡算法的探索和开发能力,对于种群中适应度值较优的个体,应增强其与最优个体之间的学习和交流,增强局部开发能力,快速搜索出最优个体所在区域的局部极值,提升算法的收敛速度;而对于种群中适应度值较差的个体,则应尽量保留个体自身基因,保持种群多样性,增大其全局探索能力. 基于这一思想,在SOS算法“互利共生”阶段引入一种分类策略,根据个体适应度值的优劣,采用不同的进化策略,有针对性地控制个体的进化方向. 鉴于每一代中个体的相对优劣程度是动态变化的,分类策略将应用于每一代种群直至算法结束. 据此,本文提出“互利共生”阶段子种群分类策略,具体操作如下.

以最小化问题为例,若Fit(i)代表第i个个体的适应度值,Fitave代表种群中所有个体适应度值的平均值,则每一代种群中个体根据其适应度值可以大致分为两类.

1) 优秀个体. 若Fit$(i) \leqslant$ Fitave,则说明个体i的适应度水平优于平均水平,归为优秀个体. 对于这类个体,应加大与最优个体的交流,增强局部开发能力. 本文为其设计的进化策略如下所示:

$$X_{i\mathrm{new}} = \begin{cases} X_{\mathrm{best}} + \mathrm{rand}(0,1) \times (X_{\mathrm{best}} - \mathrm{MV} \times \mathrm{BF}_1),\ r < m; \\ X_{\mathrm{best}} + \mathrm{rand}(0,1) \times (X_i - \mathrm{MV} \times \mathrm{BF}_1),\ r \geqslant m. \end{cases} \tag{4}$$

$$X_{j\mathrm{new}} = \begin{cases} X_{\mathrm{best}} + \mathrm{rand}(0,1) \times (X_{\mathrm{best}} - \mathrm{MV} \times \mathrm{BF}_2),\ r < m; \\ X_{\mathrm{best}} + \mathrm{rand}(0,1) \times (X_j - \mathrm{MV} \times \mathrm{BF}_2),\ r \geqslant m. \end{cases} \tag{5}$$

其中: $m \in [0,1]$为人为设定的阈值,大量实验证实,$m = 0.5$即可取得较好效果,也可根据具体问题作出相应调整; r为$[0,1]$之间的随机数;其余各变量所代表的含义与原算法一致.

2) 较差个体. 若Fit$(i) >$ Fitave,则说明个体i的适应度水平劣于平均水平,归为较差个体. 对于这类个体,应注重自身信息的保留,保持种群多样性,增强其全局探索能力. 本文为“互利共生”阶段的较差个体设计的搜索策略如下所示:

$$X_{i\mathrm{new}} = X_i + \mathrm{rand}(0,1) \times (X_i - \mathrm{MV} \times \mathrm{BF}_1), \tag{6}$$

$$X_{j\mathrm{new}} = X_j + \mathrm{rand}(0,1) \times (X_j - \mathrm{MV} \times \mathrm{BF}_2). \tag{7}$$

与SOS算法“互利共生”阶段的原有进化策略相比,本文提出的上述子种群分类策略具有如下优势(以个体i为例):

1)对于优秀个体,如式(4)所示,基向量X_{best}可视为自我学习部分,$\mathrm{rand}(0,1) \times (X_{\mathrm{best}} - \mathrm{MV} \times \mathrm{BF}_1)$及$\mathrm{rand}(0,1) \times (X_i - \mathrm{MV} \times \mathrm{BF}_1)$可以分别看作社会部分和认知部分. 当$r < m$或$r \geqslant m$时,都以种群中的最优个体作为基向量直接参与进化,其自身携带更多的优秀进化信息. 当$r < m$时,可使最优个体在其遍历的较好区域内进行仔细搜索,必然可快速搜索出其所在区域的局部极值;而当$r \geqslant m$时,可促进最优个体与其他个体之间的信息交流,有利于扩大最优个体在搜索空间中的搜索范围,探求非最优个体所在区域内的局部极值,增大算法找到全局最优的概率. 即,$r < m$和$r \geqslant m$的搜索策略相互配合,可显著提升优秀个体的局部开发能力,增强算法获得全局最优的能力.

2)对于较差个体,由式(6)和(7)可见,该策略以个体自身为基向量,且随机选择某一个体进行学习,所产生的新个体必然围绕在其自身附近,由于种群内的个体散布于整个搜索空间,该方式必然有利于维持对搜索空间进行遍历探索,从而保持种群多样性.

综上所述,通过以上分类,进化过程中每一代种群中个体的进化方式都得到了有针对性的调整,保留了丰富的种群进化信息,同时也加快了向全局最优靠近的速度,更适应个体自身的进化需求,即达到了开发能力和探索能力的有效平衡.

2.2　基于拉伸扰动的“偏利共生”搜索过程

SOS算法的“偏利共生”阶段通过式(3)产生新个体,并直接替换原个体,该搜索更新模式存在如下两方面缺陷:

1)新个体无论优劣都直接替换原个体,即使新个体劣于原个体,原个体也只能被舍弃,保留新个体. 由于原个体自身携带部分优异进化信息,新个体很可能偏离原进化方向,让其直接参与迭代搜索具有一定盲目性,必然会影响算法的收敛速度.

2)与“互利共生”阶段的搜索策略类似,“偏利共生”的搜索策略仍以其自身作为基向量,通过社会部分$\mathrm{rand}(-1,1) \times (X_{\mathrm{best}} - X_j)$与优秀个体之间进行信息交互,实现自我更新. 由于缺乏向其他个体学习的认知部分,即忽视了局部搜索信息对算法的影响,致

使算法探索能力不足,不辅以其他机制,必然容易陷入局部最优.

综上所述,经过“偏利共生”阶段未得到改善的个体,是影响算法收敛速度的一个关键因素,且原搜索策略过分注重开发能力,使探索能力不足.

鉴于个体本身就携带进化信息,尝试以此来修正原搜索策略,平衡算法的开发和勘探能力,并对未得到改善的新个体进行重新搜索,进一步提升算法的收敛速度,提出如下所示的基于拉伸扰动的“偏利共生”搜索策略.

Input: X_i, POP;

Output: $X_{i\mathrm{new}}$.

$$X_{i\mathrm{new}} = X_i + \mathrm{rand}\times(-1,1)\times(X_{\mathrm{best}} - X_j) + \mathrm{rand}(-1,1)\times(X_k - X_i) \tag{8}$$

//rand为$[-1,1]$之间的随机数,X_{best}为当前最优个体,X_j和$X_k(k\neq i\neq j)$为随机个体//

if Fit(i) > Fitnew(i)

$X(i) = X_{i\mathrm{new}}(i)$

else

$$X_{i\mathrm{new}} = X_{\mathrm{best}} + \mathrm{ls}_{(j)}\times w\times(X_j - X_i) + \mathrm{ls}_{(k)}\times w\times(X_k - X_i) \tag{9}$$

$$\mathrm{ls}_{(j/k)} = \frac{\mathrm{Fit}_{(j/k)} - \mathrm{Fitbest}}{\mathrm{Fitave} - \mathrm{Fitbest} - \alpha} \tag{10}$$

//在上述公式中,α为一个极小正数;Fit$_{(j/k)}$为个体j或者k的适应度函数值;Fitave为当前种群适应度函数的平均值;Fitbest和X_{best}为当前最优适应度值和最优个体;w为$[-1,1]$之间的随机数;ls为拉伸因子,衡量所选随机个体j和k与最优个体的差异程度,引导$(X_j - X_i)$和$(X_k - X_i)$向着最优个体靠近,ls的值越大,对整个更新公式所起到的作用越大//

与SOS算法的原“偏利共生”阶段相比,本文提出的基于拉伸扰动的“偏利共生”搜索模式对SOS算法产生如下作用:

1)在如式(3)所示的搜索策略中,增加了认知部分rand$(-1,1)\times(X_k - X_i)$,使种群中的非最优个体参与进化,与原搜索模式仅靠个体与优秀个体之间进行信息交互的方式相比,认知部分引入更多组合模式,必然会引入更多的局部搜索信息,使其不再单单围绕在最优个体附近搜索,从而提升算法的探索能力,降低陷入局部最优的风险. 此外,由于认知部分中的其他个体本身就携带了一定的进化信息,向其进行学习,不会过多地降低算法的收敛速度,即本文所提如式(8)所示的搜索策略可达到开发和探索能力的一定平衡.

2)若式(9)中的基向量为X_i,则其代表以自身为基础,按照个体X_j和X_k的优劣程度,自适应地确定搜索范围. 与此对应,本文所提式(9)则代表以最优个体为基准,在其附近以原搜索范围进行自适应搜索,即将个体由原个体所在搜索区域牵引至最优个体所在区域. 显然,由于最优个体携带最佳的进化信息,由式(9)产生的新个体比原个体优秀的概率很大,即该策略能够明显提升其开发能力.

3)对于较差个体,由于其自身劣于最优个体和其他个体,通过式(8)中社会部分和认知部分的学习,获得比自身优异的新个体的概率极高,与之相较,对于较为优秀的个体,新个体未得到改善的可能性会略高. 而式(9)仅对未得到改善的个体进行操作,即以大概率增强优秀个体的开发能力,这与2.1节的分析“应加强优秀个体的局部开发能力”相符,式(8)与(9)所示策略相互配合,可达到探索能力和开发能力的有效平衡.

2.3 精英“寄生”机制

SOS算法“寄生”阶段通过随机改变个体X_i的某个或几个维度,产生新的寄生个体P_V,并与随机个体X_j进行比较. 由于X_i本身具有一定的进化信息,在其基础上随机更改某些基因位,P_V仍继承X_i的大部分进化信息,仍与X_i较为相似. 若P_V优于X_j,则X_j将被P_V取代,种群中将不再含有X_j的局部进化信息,而是由与X_i极为相似的寄生个体P_V所代替. 显然,此种随机选择宿主进行比较的方式,对种群多样性的维护能力不佳,即算法的探索能力不足. 若P_V劣于X_j,则“寄生”阶段的搜索无效,浪费计算资源,降低了算法的收敛速度. 综上所述,SOS算法“寄生”阶段随机确定宿主个体的方式并不合理,无法有效平衡种群多样性和收敛速度.

鉴于种群中的各个体都代表特定的局部进化信息,为维持良好的探索能力,可将各个体的寄生个体P_V与其自身或某一固定个体进行比较,即宿主应为个体本身或种群内的指定个体. 由于SOS算法“互利共生”和“偏利共生”阶段都十分注重最优个体的利用,其直接决定算法的进化方向和速度. 显然,与宿主为个体自身相比,将最优个体确定为各寄生体的宿主,更能充分提升算法的收敛速度. 此外,在生物学上,寄生个体对宿主作用,一般不会完全取代宿主,而是部分吸收彼此利益. 基于上述思想,本文提出如下的精英“寄生”机制:

$$X'_{\mathrm{best}} = X_{\mathrm{best}} + \mathrm{ls}_{(P\_V)}\times w\times(X_{\mathrm{best}} - P\_V), \tag{11}$$

$$\mathrm{ls}_{(P\_V)} = \frac{\mathrm{Fit}_{(P\_V)} - \mathrm{Fitbest}}{\mathrm{Fitave} - \mathrm{Fitbest} - \alpha}, \tag{12}$$

其中$X_{\mathrm{best}} - P_V$代表寄生个体对宿主的影响,其值越小,说明对宿主的利益影响越小. 最终,将变异个体X'_{best}与“宿主”X_{best}进行适应度值比较,若变异个体

优于“宿主”,则保留变异个体,取代“宿主”;否则,“宿主”发生免疫不变,淘汰变异个体.

由上述精英寄生机制可以发现,除最优个体以外的其余个体,利用与之相似的寄生体作用于最优个体,以提升最优个体的性能,使最优个体在下一次的循环中引导“互利共生”和“偏利共生”阶段中个体的更新,这势必可以提升算法整体的收敛速度,且在此过程中并不改变除最优个体以外的其余个体的进化信息,可维持良好的全局探索能力. 综上所述,本文所提精英寄生机制可有效平衡探索能力和开发能力.

2.4 SPS-SOS算法实现步骤

综上所述,本文提出一种基于子种群拉伸操作的精英共生生物搜索算法(Elite symbiotic organisms search algorithm based on subpopulation stretching operation, SPS-SOS),其具体步骤如下.

Step 1: 初始化种群规模N、求解问题维数D、搜索范围L和U、最大迭代次数T等;

Step 2: 按式(1)生成初始种群,并计算个体的适应值;

Step 3: 确定最优个体,并求出当前种群适应度值的平均值;

Step 4: 个体X_i进入2.1节的子种群分类“互利共生”搜索阶段,即随机选择个体$X_j(i \neq j)$,并根据X_i和X_j自身适应度函数值的大小来选择式(4)、(5)或者(6)、(7)进行更新,生成新个体$X_{i\text{new}}$和$X_{j\text{new}}$,并与X_i和X_j进行比较之后,择优保留;

Step 5: 个体X_i进入2.2节所示的基于拉伸扰动的偏利共生搜索过程;

Step 6: 个体X_i进入2.3节的精英“寄生”机制,即获得X_i的“寄生向量”*P_V*,通过式(11)对“宿主”进行操作,得到X'_{best},比较X'_{best}与X_{best}的适应度值,择优保留;

Step 7: 判断是否所有个体都完成Step 4 ~ Step 6,若完成,转至Step 8,否则对于下一个个体转至Step 4;

Step 8: 判断是否达到最大迭代次数,若达到,算法停止,否则转回Step 3.

3 实验仿真结果分析

为充分验证本文所提出的改进算法——SPS-SOS求解优化问题上的有效性和先进性,这里进行一系列实验,所有实验都在CPU: Intel(R) Core(TM) i5-3230 M、4 G内存、2.60 GHZ主频的计算机上实现,程序采用MatlabR 2010b语言实现.

表 1　标准测试函数表

	函数名	Function	D	Range	Optimal
f_1	Matyas	$f(x)=0.26(x_1^2+x_2^2)-0.48x_1x_2$	2	$[-10,10]$	0
f_2	Booth	$f(x)=(x_1+2x_2-7)^2+(2x_1+x_2-5)^2$	2	$[-10,10]$	0
f_3	Schaffer	$f(x)=0.5+\dfrac{\sin^2(\sqrt{x_1^2+x_2^2})-0.5}{(1+0.001(x_1^2+x_2^2))^2}$	2	$[-100,100]$	0
f_4	Boachevsky1	$f(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1)-0.4\cos(4\pi x_2)+0.7$	2	$[-100,100]$	0
f_5	Boachevsky3	$f(x)=x_1^2+2x_2^2-0.3\cos(3\pi x_1+4\pi x_2)+0.3$	2	$[-100,100]$	0
f_6	Easom	$f(x)=-\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2-(x_2-\pi)^2)$	2	$[-100,100]$	-1
f_7	Zakharov	$f(x)=\sum_{i=1}^{D}x_i^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^4$	10	$[-5,10]$	0
f_8	Michalewicz10	$f(x)=-\sum_{i=1}^{D}\sin(x_i)(\sin(ix^2/\pi))^{20}$	10	$[0,\ \pi]$	$-9.660\,2$
f_9	Griwank	$f(x)=\dfrac{1}{4\,000}\left(\sum_{i=1}^{D}(x_i-100)^2\right)-\left(\prod_{i=1}^{D}\cos\left(\dfrac{x_i-100}{\sqrt{i}}\right)\right)+1$	50	$[-600,600]$	0
f_{10}	Step	$f(x)=\sum_{i=1}^{D}(x_i+0.5)^2$	50	$[-5.12,5.12]$	0
f_{11}	Sphere	$f(x)=\sum_{i=1}^{D}x_i^2$	50	$[-100,100]$	0
f_{12}	SumSquares	$f(x)=\sum_{i=1}^{D}ix_i^2$	50	$[-10,10]$	0
f_{13}	Schwefel 1.2	$f(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{D}x_j\right)^2$	50	$[-100,100]$	0
f_{14}	Schwefel 2.22	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	50	$[-10,10]$	0
f_{15}	Dixon-Price	$f(x)=(x_1-1)^2+\sum_{i=2}^{D}i(2x_i^2-x_i-1)^2$	50	$[-10,10]$	0
f_{16}	Rastrigin	$f(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	50	$[-5.12,5.12]$	0
f_{17}	Elliptic	$f(x)=\sum_{i=1}^{D}(10^6)^{\frac{i-1}{D-1}}x^2$	50	$[-100,100]$	0

3.1 函数说明及性能指标

本文选取17个国际通用的标准测试函数进行测试,包括单峰函数和具有多个局部最优的多峰函数.其中:单峰函数包括f_1、f_6、f_7、$f_{10} \sim f_{14}$、f_{15}、f_{17};多峰函数包括$f_2 \sim f_5$、f_8、f_9、f_{16}. 具体如表1所示.

选用如下常用的性能指标来评价算法性能.

求解精度:算法在达到某一函数评价次数时所能获得的最优精度,越接近理论值,结果越好;

收敛速度:各算法在测试某一函数时,达到相同函数评价次数时所获得的最优解,或达到相同精度时所获得的函数评价次数;

鲁棒性:算法收敛到指定精度的成功率,成功率越高,其鲁棒性越好.

3.2 与SOS系列算法比较

为考察本文提出的SPS-SOS算法是否能够有效提升SOS算法的性能,将其与基本SOS算法以及目前对SOS算法性能改进较为优异的多策略自适应共生生物搜索算法——MSASOS(Multi strategy adaptive symbiotic organisms search)在收敛速度和收敛精度上进行比较.

为体现对比的公平性,3个算法在测试相同的函数时参数设置相同,即种群数目N为50,函数评价次数为80 000(保证各算法都可收敛). 为避免算法单次运行的偶然性给算法评价带来的不良影响,各算法均独立运行30次. 在比较收敛精度时,测试项目包括多次运行中的最优值(Best)、最差值(Worst)、平均值(Mean)、标准差(SD);在比较收敛速度时,测试项目为在不同的评价次数时对应的解精度,由于函数$f_1 \sim f_6$为低维函数,分别测试它们在函数评价次数为4 000、6 000、8 000时获得的最优解精度,而高维函数$f_7 \sim f_{17}$则测试其在函数评价次数为40 000、50 000、60 000时所获的最优解精度. 具体实验结果如表2所示.

通过分析表2中的数据不难看出:

1)对于低维函数$f_1 \sim f_6$而言,在80 000次的评价次数之下,各算法基本都能收敛到最优值,然而对比在评价4 000、6 000以及8 000时的解精度可以发现,本文提出的SPS-SOS算法所获精度明显优于SOS算法及MSASOS算法,即对于低维函数优化,3种算法能同时收敛到最优,但本文算法SPS-SOS在收敛速度方面具有更明显优势.

2)对于高维函数$f_7 \sim f_{17}$而言,在80 000次评价次数内,SPS-SOS算法在f_7、f_8、$f_{11} \sim f_{14}$、f_{16}、f_{17}上均能收敛到理论最优值,MSASOS算法可在f_7、f_8、f_{11}、f_{12}、f_{16}上收敛到理论最优值,而SOS算法仅能在f_8和f_{16}上收敛到最优值. 对于f_9、f_{10}、f_{15}而言,3种算法都不能收敛到理论最优值,但是本文所提算法SPS-SOS所获结果已十分接近全局最优,说明本文算法在收敛精度上明显优于其他两种算法,且在所有测试函数上,本文算法所获标准差均远小于其他两种算法,说明本文算法更为稳定. 此外,在相同函数评价次数下,本文算法SPS-SOS均获得了比SOS算法和MSASOS算法更优的适应度值,说明本文算法在收敛速度上具有明显优势.

综上所述,与SOS算法和MSASOS算法相比,本文所提出算法SPS-SOS可获得更好的求解精度和收敛速度,即更好的函数优化性能,且随着函数维数的升高,优势愈加明显.

3.3 与非SOS系列算法比较

为充分验证SPS-SOS算法的先进性,将其与目前优化效果较好的4种算法,包括差分进化算法的改进算法MDE(Modified differential evolution with self-adaptive parameters method)[9]、万有引力搜索算法的改进算法IGSA/PSO(An improved gravitational search algorithm for green partner selection in virtual enterprises)[10]、人工蜂群算法的改进算法distABC(Artificial Bee Colony algorithm with distribution-based update rule)[11]和粒子群算法的改进算法ADN-RSN-PSO(All-dimension neighborhood based particle swarm optimization with randomly selected neighbors)[12],从求解精度、收敛速度方面进行全面比较.

为了对比的公平性,各算法的种群数目都为50,最大函数评价次数为80 000. 除本文算法外,其他算法的主要相关参数设置如下.

MDE:交叉概率$\mathrm{CR} = 0.4$,变异概率F在算法迭代过程中根据随机数Rand而定.

IGSA/PSO:引力常数$G_0 = 100, \alpha = 20$.

distABC: limit = (种群数目 × 维度) / 2.

ADN-RSN-PSO:权重因子$w = 0.729\,8$, $c_1 = c_2 = 2.05$.

3.3.1 求解精度比较

这一节考察SPS-SOS算法在求解精度上的能力,表3给出了各算法的评价次数同为80 000次时的实验结果. 其中:Best、Worst、Mean、SD分别代表30次独立实验中,所获结果的最优解、最差解、平均值、标准差;R代表算法在30次实验过程中最优值收敛至10^{-10}的次数;Sort代表各测试函数中算法所得平均值的排序结果.

表 2　SOS 系列收敛结果

Function	Methods	收敛精度比较				收敛速度比较		
		Best	Mean	Worst	SD	不同函数评价次数下的解精度		
f_1	SOS	7.681 0e-258	2.186 6e-252	1.122 7e-251	0	3.904 1e-16	6.340 9e-22	1.517 9e-27
	MSASOS	0	0	0	0	1.066 0e-18	6.428 5e-33	7.039 0e-48
	SPS-SOS	0	0	0	0	1.104 9e-55	4.252 4e-82	4.668 5e-109
f_2	SOS	0	0	0	0	8.984 1e-09	8.912 4e-12	5.422 2e-13
	MSASOS	0	0	0	0	1.802 3e-06	1.520 8e-12	8.520 7e-22
	SPS-SOS	0	0	0	0	0	0	0
f_3	SOS	0	0	0	0	0.009 7	0.002 3	1.776 2e-04
	MSASOS	0	0	0	0	0.001 3	4.270 6e-04	3.497 6e-04
	SPS-SOS	0	0	0	0	0	0	0
f_4	SOS	0	0	0	0	7.778 8e-10	1.128 0e-16	0
	MSASOS	0	0	0	0	0	0	0
	SPS-SOS	0	0	0	0	0	0	0
f_5	SOS	0	0	0	0	1.623 7e-09	5.551 1e-17	0
	MSASOS	0	0	0	0	0	0	0
	SPS-SOS	0	0	0	0	0	0	0
f_6	SOS	−1	−1	−14	0	−0.999 9	−0.999 9	−0.999 9
	MSASOS	−1	−1	−1	0	−0.999 9	−1	−1
	SPS-SOS	−1	−1	−1	0	−1	−1	−1
f_7	SOS	1.625 4e-168	1.552 0e-163	1.755 0e-162	0	9.057 0e-83	2.130 1e-103	3.072 1e-124
	MSASOS	0	0	0	0	1.174 8e-139	1.430 6e-189	5.380 6e-236
	SPS-SOS	0	0	0	0	0	0	0
f_8	SOS	−9.660 2	−9.657 4	−9.648 2	0.008 5	−9.624 6	−9.645 2	−9.655 1
	MSASOS	−9.660 2	−9.649 8	−9.589 2	0.023 4	−9.620 7	−9.639 0	−9.653 3
	SPS-SOS	−9.660 2	−9.659 4	−9.655 2	0.004 3	−9.660 2	−9.660 2	−9.660 2
f_9	SOS	0.030 8	0.093 7	0.118 2	0.045 7	0.402 3	0.229 1	0.047 8
	MSASOS	6.740 5e-07	2.095 5e-05	3.967 1e-04	3.493 8e-05	0.002 3	2.560 5e-05	1.762 1e-05
	SPS-SOS	1.477 9e-08	1.085 6e-07	5.783 1e-06	8.127 4e-07	0.001 7	1.597 1e-04	2.113 9e-05
f_{10}	SOS	8.930 5e-04	0.003 7	0.009 8	0.002 1	0.025 1	0.014 5	0.008 7
	MSASOS	0.001 5	0.005 2	0.010 1	0.003 0	0.026 9	0.017 9	0.009 6
	SPS-SOS	7.115 5e-32	1.543 5e-29	4.198 0e-28	7.641 8e-29	4.547 9e-20	6.970 9e-25	5.386 2e-29
f_{11}	SOS	7.831 2e-166	1.256 3e-161	2.980 1e-160	5.531 0e-161	2.694 0e-79	1.744 7e-99	6.693 6e-121
	MSASOS	0	4.455 5e-318	1.243 4e-316	0	1.142 9e-128	1.025 5e-188	2.654 4e-228
	SPS-SOS	0	0	0	0	0	0	0
f_{12}	SOS	1.565 5e-167	1.744 2e-163	1.782 5e-162	0	1.835 1e-81	9.042 4e-102	1.173 1e-122
	MSASOS	0	0	0	0	3.938 2e-138	3.296 5e-184	4.636 0e-234
	SPS-SOS	0	0	0	0	5.068 7e-308	0	0
f_{13}	SOS	1.286 3e-133	3.114 1e-131	1.964 8e-130	4.245 9e-131	8.551 3e-63	3.537 1e-80	4.595 4e-97
	MSASOS	2.788 2e-233	1.097 7e-227	2.037 1e-226	0	9.446 1e-104	6.916 7e-136	3.610 2e-167
	SPS-SOS	0	2.704 9e-318	4.237 4e-317	0	9.824 2e-219	6.423 2e-255	1.604 3e-282
f_{14}	SOS	1.204 8e-83	6.935 9e-81	6.409 9e-80	1.326 2e-80	7.236 6e-40	4.317 7e-61	9.019 5e-102
	MSASOS	7.403 7e-160	1.690 5e-150	2.230 0e-149	5.377 2e-150	3.390 7e-57	3.453 7e-81	2.830 8e-105
	SPS-SOS	0	2.228 7e-287	6.271 9e-286	0	1.151 1e-185	5.815 0e-206	1.506 6e-228
f_{15}	SOS	2.726 9	9.658 9	15.630 2	4.487 7	31.605	24.053 4	10.250 6
	MSASOS	7.915 8	17.629 8	25.770 6	10.344 9	60.234 7	41.037 8	30.570 6
	SPS-SOS	4.412 8e-25	4.406 4e-21	5.531 5e-20	1.425 7e-20	2.723 0e-12	2.044 1e-16	6.329 1e-18
f_{16}	SOS	0	0	0	0	0	0	0
	MSASOS	0	0	0	0	0	0	0
	SPS-SOS	0	0	0	0	0	0	0
f_{17}	SOS	1.556 3e-163	1.508 0e-157	4.346 0e-156	7.522 1e-157	6.861 5e-76	5.368 2e-97	2.152 2e-117
	MSASOS	1.755 0e-303	2.153 7e-289	2.100 0e-288	0	1.573 2e-91	2.249 3e-140	7.536 6e-191
	SPS-SOS	0	0	0	0	0	0	0

表 3 收敛精度比较

Function	Methods	Best	Mean	Worst	SD	R	Sort
f_1	MDE	2.159 3 e-121	2.255 1 e-115	5.276 4 e-114	9.588 2 e-115	30	2
	IGSA/PSO	1.628 1 e-20	3.059 4 e-19	7.920 5 e-18	3.257 8 e-19	30	4
	distABC	6.128 7 e-123	1.606 6 e-107	1.283 8 e-106	3.177 4 e-107	30	3
	ADN-RSN-PSO	3.741 9 e-27	2.779 3 e-09	1.425 6 e-08	6.992 1 e-09	22	5
	SPS-SOS	0	0	0	0	30	1
f_2	MDE	0	0	0	0	30	1
	IGSA/PSO	1.670 9 e-19	9.391 5 e-18	3.960 5 e-17	8.707 4 e-18	30	3
	distABC	2.375 8 e-04	0.028 2	0.052 0	0.017 6	0	5
	ADN-RSN-PSO	8.355 0 e-12	2.536 7 e-06	3.443 8 e-05	1.577 5 e-06	7	4
	SPS-SOS	0	0	0	0	30	1
f_3	MDE	0	9.715 9 e-04	0.009 7	0.003 0	27	2
	IGSA/PSO	3.646 1 e-04	0.006 0	0.009 7	0.003 3	0	4
	distABC	1.042 1 e-06	0.004 6	0.009 8	0.004 2	0	3
	ADN-RSN-PSO	3.053 1 e-04	0.036 1	0.094 2	0.024 6	0	5
	SPS-SOS	0	0	0	0	30	1
f_4	MDE	0	0	0	0	30	1
	IGSA/PSO	0	1.480 3 e-17	1.346 1 e-16	3.838 6 e-17	30	4
	distABC	0	0	0	0	30	1
	ADN-RSN-PSO	1.821 5 e-12	1.040 3 e-04	0.001 8	3.220 5 e-04	6	5
	SPS-SOS	0	0	0	0	30	1
f_5	MDE	0	0	0	0	30	1
	IGSA/PSO	0	1.110 2 e-17	1.415 2 e-16	2.688 0 e-17	30	4
	distABC	0	0	0	0	30	1
	ADN-RSN-PSO	1.913 5 e-15	5.259 9 e-05	5.751 8 e-04	1.434 0 e-04	9	5
	SPS-SOS	0	0	0	0	30	1
f_6	MDE	-1	-1	-1	0		1
	IGSA/PSO	-1	-1	-1	0		1
	distABC	$-0.996\,8$	$-0.938\,0$	$-0.808\,6$	0.048 3		3
	ADN-RSN-PSO	$-0.444\,4$	$-0.019\,1$	$-8.382\,5$ e-12	0.083 7		5
	SPS-SOS	-1	-1	-1	0		1
f_7	MDE	8.900 3 e-40	1.751 2 e-36	2.826 9 e-35	5.203 7 e-36	30	3
	IGSA/PSO	4.326 0 e-16	1.690 7 e-15	2.523 7 e-15	6.626 8 e-16	30	4
	distABC	3.712 3 e-56	1.569 1 e-52	1.909 7 e-51	3.730 2 e-52	2	2
	ADN-RSN-PSO	1.749 9 e-18	0.005 0	0.074 3	0.021 4	8	5
	SPS-SOS	0	0	0	0	30	1
f_8	MDE	$-9.660\,2$	$-9.437\,8$	$-9.130\,7$	0.106 5		2
	IGSA/PSO	$-8.868\,5$	$-6.671\,6$	$-5.567\,8$	0.9053		3
	distABC	$-4.332\,1$	$-3.366\,6$	$-3.020\,8$	0.264 6		5
	ADN-RSN-PSO	$-6.135\,3$	$-4.602\,3$	$-3.976\,2$	0.498 4		4
	SPS-SOS	$-9.660\,2$	$-9.659\,4$	$-9.655\,2$	0.004 3		1
f_9	MDE	1.064 7 e-13	1.273 7 e-12	1.113 4 e-11	2.037 1 e-12	30	1
	IGSA/PSO	1.089 6	3.174 3	6.159 2	1.502 5	0	3
	distABC	32.060 7	38.670 7	42.655 5	3.432 1	0	4
	ADN-RSN-PSO	27.085 9	57.834 4	66.744 8	10.934 3	0	5
	SPS-SOS	1.477 9 e-08	1.085 6 e-07	5.783 1 e-06	8.127 4 e-07	0	2
f_{10}	MDE	1.452 9 e-11	3.979 0 e-10	1.198 0 e-09	2.913 9 e-10	28	2
	IGSA/PSO	9.670 6 e-05	9.714 9 e-04	0.007 4	0.001 6	0	3
	distABC	7.693 1	10.026 2	11.041 2	0.912 0	0	4
	ADN-RSN-PSO	9.181 8	11.148 0	12.302 7	0.869 1	0	5
	SPS-SOS	7.115 5 e-32	1.543 5 e-29	4.198 0 e-28	7.641 8 e-29	30	1
f_{11}	MDE	3.144 8 e-08	2.677 2 e-07	2.596 3 e-06	4.758 9 e-07	0	2
	IGSA/PSO	6.306 5 e-05	8.831 1 e-04	0.005 3	0.001 2	0	3
	distABC	1.710 7	5.447 1	10.991 9	2.613 7	0	5
	ADN-RSN-PSO	8.6229 e-18	0.029 7	0.137 4	0.117 4	8	4
	SPS-SOS	0	0	0	0	30	1
f_{12}	MDE	3.458 6 e-09	3.865 7 e-08	1.516 9 e-07	2.920 8 e-08	0	2
	IGSA/PSO	0.002 9	0.257 9	1.454 2	0.327 0	0	4
	distABC	0.293 6	1.023 1	2.417 0	0.529 4	0	5
	ADN-RSN-PSO	5.039 1 e-25	0.108 5	2.379 9	0.449 6	6	3
	SPS-SOS	0	0	0	0	30	1
f_{13}	MDE	5.137 2 e+04	8.635 9 e+04	1.028 6 e+05	1.717 4 e+04	0	4
	IGSA/PSO	107.910 2	416.359 5	543.469 2	194.781 1	0	3
	distABC	1.771 8 e+05	2.342 4 e+05	2.715 4 e+05	3.077 0 e+04	0	5
	ADN-RSN-PSO	4.247 8 e-26	0.631 7	12.267 2	2.404 3	7	2
	SPS-SOS	0	2.704 9 e-318	4.237 4 e-317	0	30	1

续表　收敛精度比较

Function	Methods	Best	Mean	Worst	SD	R	Sort
f_{14}	MDE	1.370 5 e-05	8.939 7 e-05	1.966 3 e-04	4.623 3 e-05	0	2
	IGSA/PSO	0.082 1	0.419 0	2.038 2	0.497 4	0	4
	distABC	0.139 7	0.430 6	1.413 7	0.257 2	0	5
	ADN-RSN-PSO	2.355 1 e-04	0.246 4	3.511 2	1.524 3	0	3
	SPS-SOS	0	2.228 7 e-287	6.271 9 e-286	0	30	1
f_{15}	MDE	1.347 3 e-05	2.174 4	24.258 6	4.950 2	0	2
	IGSA/PSO	0.427 4	7.352 8	22.829 8	6.289 9	0	3
	distABC	1.475 6 e+03	2.291 1 e+04	1.398 5 e+05	2.900 9 e+04	0	5
	ADN-RSN-PSO	539.108 6	704.458 8	918.060 1	73.804 7	0	4
	SPS-SOS	4.412 8 e-25	4.406 4 e-21	5.531 5 e-20	1.425 7 e-20	30	1
f_{16}	MDE	92.736 7	121.033 9	197.127 7	23.168 9	0	4
	IGSA/PSO	66.741 1	103.595 9	157.230 4	21.540 5	0	3
	distABC	304.459 1	304.459 1	418.896 6	21.533 5	0	5
	ADN-RSN-PSO	5.032 5 e-06	35.973 1	392.019 9	104.603 8	0	2
	SPS-SOS	0	0	0	0	30	1
f_{17}	MDE	4.982 4 e-05	0.001 1	0.003 2	9.486 3 e-04	0	2
	IGSA/PSO	3.126 0 e+04	1.237 0 e+05	2.710 0 e+05	6.476 8 e+04	0	5
	distABC	4.830 6 e+03	1.687 1 e+04	4.481 7 e+04	8.628 3 e+03	0	4
	ADN-RSN-PSO	1.748 9 e-13	2.409 5 e+03	2.900 5 e+04	7.326 0 e+03	3	3
	SPS-SOS	0	0	0	0	30	1

通过分析表3中的各项数据值可知:

1) 对于低维函数 $f_1 \sim f_6$, 本文所提算法SPS-SOS可获得各函数的理论最优值; 而MDE在函数 $f_2 \sim f_6$ 上可以收敛到理论最优值, 但在 f_1 上表现并不十分稳定; IGSA/PSO在 f_6 上可以达到稳定收敛, 在 f_4 和 f_5 上有一定机会收敛到全局最优, 而在 $f_1 \sim f_3$ 上无法有效收敛; distABC仅在 f_4 及 f_5 上可以稳定收敛到理论最优; ADN-RSN-PSO则在任何函数上都无法有效收敛. 综上说明, 在处理低维函数优化时, IGSA/PSO和distABC表现相当, MDE较好, 本文算法SPS-SOS效果最优, 也最为稳定.

2) 对于高维函数 $f_7 \sim f_{17}$, 除了MDE算法在函数 f_9 上略胜于本文算法SPS-SOS, 这4种算法在其余函数上都与全局最优结果相差较远, 未成功收敛, 而本文算法SPS-SOS在函数 f_7、f_{11}、f_{12}、f_{16} 以及 f_{17} 上均可成功收敛, 在函数 f_{13} 和 f_{14} 上均有机会收敛至理论最优, 在函数 $f_8 \sim f_{10}$ 上获得的最优结果也十分接近理论最优结果, 且明显优于其他算法, 即与其他算法相比, 本文算法在高维函数上能够获得更优的函数优化性能.

综上说明, 本文算法SPS-SOS在求解精度方面明显优于其余4种算法, 且优化问题的维度越高, 优势越为明显.

3.3.2　收敛速度比较

这一节考察SPS-SOS算法在收敛速度上的能力, 各算法的参数设置保持不变, 均独立运行30次, 分别记录各算法在达到相同收敛精度(Value)时所需要的函数评价次数的平均值. 由于不同算法最终收敛精度差异较大, 为尽量保证在规定函数评价次数内各算法可达到预设精度(Value), 即保证对比的有效性, 各函数的预设精度设定有所差异, 具体预设精度值以及统计结果如表4所示(注: "/" 表示该算法在80 000次规定函数评价次数内未达到预设精度).

表 4　收敛速度比较

Function	Value	Methods				
		MDE	IGSA/PSO	distABC	ADN-RSN-PSO	SPS-SOS
f_1	10^{-5}	2 966	15 420	2 385	5 328	413
	10^{-10}	6 534	39 356	4 969	9 793	622
	10^{-15}	9 872	63 582	7 134	16 785	846
f_2	10^{-1}	1 245	4 752	3 625	3 361	105
	10^{-3}	1 968	9 836	21 317	10 309	260
	10^{-5}	3 373	18 982	/	26 526	408
f_3	10^{-1}	1 166	476	992	425	358
	10^{-3}	14 195	20 752	15 808	6 783	1 557
	10^{-5}	23 427	/	49 246	/	1 702
f_4	10^{-5}	4 647	26 992	3 248	11 585	612
	10^{-10}	6 538	50 823	4 612	65 120	805
	10^{-15}	9 241	72 548	6 657	/	1 014
f_5	10^{-5}	5 337	22 783	3 882	4 039	636
	10^{-10}	13 441	45 761	6 939	10 514	845
	10^{-15}	17 668	69 170	9 061	/	1 211
f_6	-0.2	733	859	1 896	39 247	103
	-0.4	927	1 270	3 305	58 134	192
	-0.6	1 098	1 541	4 512	/	308
f_7	10^{-5}	14 336	34 961	11 562	14 058	615
	10^{-10}	25 770	58 427	17 705	23 176	1 017
	10^{-15}	34 385	78 342	26 750	32 249	1 426
f_8	-2	174	257	196	182	122
	-4	703	14 842	57 719	6 729	324
	-6	3 378	27 234	/	16 325	605
f_9	10^{2}	3 605	423	9 547	826	168
	10^{-2}	32 286	/	/	/	17 496
	10^{-6}	51 762	/	/	/	67 826

续表 收敛速度比较

Function	Value	Methods				
		MDE	IGSA/PSO	distABC	ADN-RSN-PSO	SPS-SOS
f_{10}	10^2	6 858	2 378	13 541	2 075	109
	10^1	9 835	6 723	60 529	58 602	416
	10^0	14 642	12 453	/	/	2 566
f_{11}	10^2	19 863	4 971	58 567	1 928	117
	10^1	28 426	8 536	75 824	3 453	401
	10^0	38 260	13 504	/	5 209	812
f_{12}	10^2	17 184	9 312	42 377	2 273	396
	10^0	31 204	22 606	74 595	4 694	578
	10^{-2}	54 580	58 079	/	7 759	831
f_{13}	10^7	186	155	178	203	95
	10^6	3 496	478	573	366	216
	10^5	48 342	886	/	1 928	322
f_{14}	10^2	6 583	566	22 031	753	176
	10^0	29 647	9 764	75 389	2 287	428
	10^{-2}	55 034	/	/	6 805	608
f_{15}	10^5	8 321	183	68 507	458	122
	10^3	14 237	7 795	/	1 945	237
	10^1	30 231	56 089	/	/	676
f_{16}	10^3	173	146	258	153	84
	10^2	70 844	63 427	/	1 377	158
	10^1	/	/	/	2 804	237
f_{17}	10^7	11 738	4 374	33 629	2 184	227
	10^5	24 706	11 563	54 921	5 059	610
	10^3	33 987	/	/	7 632	837

通过分析表4中的数据可知,除了当精度达到10^{-6}后,MDE算法在f_9上所需的函数评价次数少于本文所提算法之外,对于其他各测试函数,在达到相同的收敛精度时,本文所提算法所需的函数评价次数均明显低于其他4种现在较为优异的优化算法. 由此表明,与其余4种算法相比,本文所提算法SPS-SOS在收敛速度方面具有明显优势.

4 结 论

针对共生生物搜索算法在优化单目标优化问题时存在着收敛速度和收敛精度不足等问题,本文提出了一种基于子种群拉伸操作的精英共生生物搜索算法SPS-SOS. 首先,为了保持种群的多样性,平衡算法的开发与探索能力,在“互利共生”阶段根据种群个体各自的适应度值将种群分为两个子种群,让不同子种群中的个体通过不同的搜索策略进行更新,进一步提高了算法的收敛速度与精度;其次,在“偏利共生”阶段,利用最优个体进行引导的同时,通过加入“扰动个体”以及“拉伸”因子提高了算法的收敛速度和收敛精度;在“寄生”阶段,采用精英“寄生”机制,进一步平衡了算法在整个搜索过程的探索与开发能力. 在17个标准测试函数集上的实验结果表明,相较于标准SOS算法、改进后的SOS算法以及目前被广泛应用的4个群智能进化算法,本文算法在收敛速度、求解精度、鲁棒性上均具有一定优势.

参考文献(References)

[1] Cheng M Y, Prayogo D. Symbiotic organisms search: A new metaheuristic optimization algorithm[J]. Computers & Structures, 2014, 139: 98-112.

[2] Harik G R, Lobo F G, Goldberg D E. The compact genetic algorithm[J]. IEEE Trans on Evolutionary Computation, 1999, 3(4): 287-297.

[3] Das S, Suganthan P N. Differential evolution: A survey of the state-of-the-art[J]. IEEE Trans on Evolutionary Computation, 2011, 15(1): 4-31.

[4] Bai Q. Analysis of particle swarm optimization algorithm[J]. Computer & Information Science, 2010, 3(1): 180-184.

[5] Ghosh A, Das S, Chowdhury A, et al. An improved differential evolution algorithm with fitness-based adaptation of the control parameters[J]. Information Sciences, 2011, 181(18): 3749-3765.

[6] Ishaque K, Salam Z, Amjad M, et al. An improved particle swarm optimization(PSO) — Based MPPT for PV with reduced steady-state oscillation[J]. IEEE Trans on Power Electronics, 2012, 27(8): 3627-3638.

[7] 周虎, 赵辉, 李牧东, 等. 多策略自适应共生生物搜索算法[J]. 空军工程大学学报: 自然科学版, 2016, 17(4): 101-106.
(Zhou H, Zhao H, Li M D, et al. Multi strategy adaptive symbiotic organisms search[J]. J of Air Force Engineering University: Natural Science Edition, 2016, 17(4): 101-106.)

[8] 王艳娇, 陶欢欢. 基于旋转学习策略的共生生物搜索算法[J]. 计算机应用研究, 2017, 34(9): 2614-2617.
(Wang Y J, Tao H H. Symbiotic organisms search algorithm based on rotating learning strategy[J]. Computer Application Research, 2017, 34(9): 2614-2617.)

[9] Li X, Yin M. Modified differential evolution with self-adaptive parameters method[J]. J of Combinatorial Optimization, 2016, 31(2): 546-576.

[10] Xiao J, Niu Y, Chen P, et al. An improved gravitational search algorithm for green partner selection in virtual enterprises[J]. Neurocomputing, 2016, 217: 103-109.

[11] Ismail Babaoglu. Artificial bee colony algorithm with distribution-based update rule[J]. Applied Soft Computing J, 2015, 34(C): 851-861.

[12] Sun W, Lin A, Yu H, et al. All-dimension neighborhood based particle swarm optimization with randomly selected neighbors[J]. Information Sciences, 2017, 405(C): 141-156.

作者简介

王艳娇(1985–), 女, 副教授, 博士, 从事进化计算、智能信息处理的研究, E-mail: wangyanjiao1028@126.com;

马壮(1994–), 男, 硕士生, 从事进化计算、智能信息处理的研究, E-mail: 867382014@qq.com.

(责任编辑: 齐 霁)