求解变量重叠型大尺度优化问题的相关性学习协同演化策略

王豫峰[1,2],董文永[1],董学士[1]
(1. 武汉大学计算机学院,湖北武汉 430072;2. 南阳理工学院软件学院,河南南阳 473000)

摘 要: 协同演化是解决大尺度连续优化问题的一种有效策略. 但是,对于决策变量重叠型(决策变量不可分且相互依赖)的高维问题,其分组方法可能会误导算法的搜索. 针对这一情况,本文提出一种全新的协同演化策略(Differential Evolution Cooperative Coevolution with Correlation Learning Between Variables,DECC-CLV),其思想是首先计算演化种群分布所包含的主特征轴,然后计算各维决策变量在主轴上的投影值并利用它们之间的正负相关性进行分组. 该算法在迭代过程中,利用期望最大化算法对种群进行概率主成分分析,并根据决策变量在当前种群主轴上的投影值大小关系对其进行动态分组. 通过和目前主流的演化算法在CEC2013的第三类函数上的仿真试验和分析,验证了算法的有效性和适用性.

关键词: 大尺度优化问题; 相关性决策变量; 协同演化; 大尺度优化问题分解
中图分类号: TP301 **文献标识码:** A **文章编号:** 0372-2112 (2018)03-0529-08
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.03.003

Cooperative Coevolution with Correlation Learning Between Variables for Large Scale Overlapping Problem

WANG Yu-feng[1,2], DONG Wen-yong[1], DONG Xue-shi[1]
(1. *Computer School, Wuhan University, Wuhan, Hubei* 430072, *China*;
2. *Software School, Nanyang Institute of Technology, Nanyang, Henan* 473000, *China*)

Abstract: Cooperative co-evolution (CC) is an effective strategy to solve large-scale continuous optimization problem. However, its grouping method may mislead the search direction when solving the large-scale overlapping problem (decision variables are non-separable and interact with each other). In order to overcome this issue, we propose a differential evolution cooperative coevolution with correlation learning between variables (DECC-CLV) to improve the performance of CC. DECC-CLV firstly detects the positive and negative correlations of variables based on the projected value of decision variables on the principal component of the population, and then groups variables into different groups. During the evolutionary process, DECC-CLV employs the expectation maximization algorithm for probabilistic principal component analysis on the population to deduce the complexity. Comparing with the state-of-the-art CCs on the large-scale overlapping benchmark functions on CEC2013, the experimental results verified the effectiveness and applicability of our proposed algorithm.

Key words: large-scale optimization problem; variables correlation; cooperative co-evolution; large-scale optimization problem decomposition

1 引言

现实世界中遇到的很多实际问题,都可以建模成大尺度优化问题(Large Scale Global Optimizations, LSGOs)[1]. 在这类优化问题中,决策变量重叠型大尺度优化问题是比较难解决的一类问题,由于互相重叠的决策变量之间有正相关和负相关影响,使得在求解的过程中并没有唯一最优的决策变量分组方法.

协同演化(Cooperative Coevolution,CC)[2]是目前解决大尺度优化问题比较流行的方法. 它通过分治策略

收稿日期:2016-09-29;修回日期:2017-01-05;责任编辑:梅志强
基金项目:国家自然科学基金(No. 61170305, No. 61672024);河南省高等学校重点科研项目计划(No. 17A520046)

来解决优化问题,也就是将一个优化问题分解为多个低维子问题,并分别对多个低维子问题进行求解[3].因此,决策变量的不同分组方法对于该协同演化算法的影响非常大.在1994年,Potter和De Jong首先提出采用固定分组方法的协同演化算法CCGAs[2].Yang等提出DECC-G[4],利用随机分组来对高维问题进行分解.MN Omidvar等提出通过对决策变量进行差分分组的算法DECC-DG[5].这些算法对于大尺度决策变量部分可分离类优化问题取得了比较好的效果.但是,对于决策变量重叠类优化问题效果不够理想,究其原因,主要因为它们只依据决策变量之间是否有相互作用进行分组,而没有考虑到决策变量之间可能会出现的正相关与负相关等因素.当把具有负相关的两个决策变量分到一组后,往往不能朝着最优解的方向进行演化,得不到最优解.

为了克服决策变量重叠类优化问题中决策变量之间正负相关性影响的问题,本文提出一种基于统计学习变量正负相关性,对决策变量进行自适应分组的协同演化算法(DECC-CLV).算法首先对每代种群进行概率主成分分析,统计出当前种群的主特征轴方向.然后,计算当前种群中最优个体每个维度的决策变量在主特征轴方向的投影值,并将投影值进行归一化.根据投影值归一化后的直方图分布情况,对决策变量进行动态自适应分组.由于分组时区分了变量间的正负相关性,可以有效避免将两个相关性冲突的决策变量分到一组,影响算法的收敛精度.在整个演化的过程中,算法循环地采用期望最大值估计(Expectation Maximization,EM)算法来学习当前种群的主轴,并修正分组的个数及每组中的决策变量个数,使得算法能够更加准确地捕捉问题的特性,引导算法快速、准确地搜索问题的最优解.

2 相关背景知识介绍

2.1 优化问题的类型检测

对于变量非重叠型的优化问题,从理论上来讲,在优化求解的过程中,都会有一个最优的决策变量分组方案.然而,对于变量重叠类的优化问题,由于重叠部分的变量之间有正相关和负相关的影响,该类问题并没有一个唯一的最优分组方案.因此,该类优化问题也是实际问题中最复杂的一类问题.

在处理实际问题中,如果已经知道该优化问题是变量重叠型优化问题,采用传统的先分组后协同演化的策略往往性能不够理想,需要提出新的动态分组方法,也即:分组和优化过程同时进行.在求解具体问题时,可以通过式(1)[6]是否成立来探测决策变量是否可分.

$$\begin{aligned}&\exists x,x'_i,x'_j:\\&(f(x_1\cdots,x_i,\cdots,x_j,\cdots,x_D)<f(x_1\cdots,x'_i,\cdots,x_j,\cdots,x_D))\wedge\\&(f(x_1\cdots,x_i,\cdots,x'_j,\cdots,x_D)<f(x_1\cdots,x'_i,\cdots,x'_j,\cdots,x_D))\end{aligned}\tag{1}$$

其中,如果相同位置的决策变量不同值x_i和x'_i,x_j和x'_j满足式(1),则第i个和第j个决策变量是相互有影响的.

一旦检测到所求解的问题属于不可分变量类型,就可以采用本文提出的基于相关性学习的动态分组方法来求解该问题.

2.2 概率主成分分析

概率主成分分析(Probabilistic Principal Component Analysis,PPCA)[7]是传统PCA的延伸,其主要思想和模型如下:基于从隐空间到数据空间的一个映射,也就是D维决策变量$\boldsymbol{x}$由M维隐变量$\boldsymbol{z}$的一个线性变换附加一个高斯"噪声"[8]组成的,即:

$$\boldsymbol{x}=\boldsymbol{W}\boldsymbol{z}+\boldsymbol{\mu}+\boldsymbol{\varepsilon}\tag{2}$$

其中,$\boldsymbol{z}$是一个M维隐变量,$\boldsymbol{W}$是$D\times M$的因子载荷,$\boldsymbol{\mu}$为D维样本均值,$\boldsymbol{\varepsilon}$为D维噪声变量.

$$p(\boldsymbol{z})=N(\boldsymbol{z}|\boldsymbol{0},\boldsymbol{I})\tag{3}$$

以隐变量$\boldsymbol{z}$为条件,决策变量$\boldsymbol{x}$的条件概率分布为:

$$p(\boldsymbol{x}|\boldsymbol{z})=N(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z}+\boldsymbol{\mu},\sigma^2\boldsymbol{I})\tag{4}$$

$$p(\boldsymbol{\varepsilon})=N(\boldsymbol{\varepsilon}|\boldsymbol{0},\sigma^2\boldsymbol{I})\tag{5}$$

根据概率的加和规则和乘积规则,边缘概率分布为:

$$p(\boldsymbol{x})=\int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})\mathrm{d}\boldsymbol{z}\tag{6}$$

由于对应一个线性高斯模型,因此边缘概率分布还是高斯分布.

$$p(\boldsymbol{x})=N(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{C})\tag{7}$$

$\boldsymbol{C}$为$D\times D$的协方差矩阵.

$$\boldsymbol{C}=\boldsymbol{W}\boldsymbol{W}^{\mathrm{T}}+\sigma^2\boldsymbol{I}\tag{8}$$

根据式(7),对应的对数似然函数为:

$$\begin{aligned}\ln p(\boldsymbol{X}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2)&=\sum_{n=1}^{N}\ln p(x_n|\boldsymbol{W},\boldsymbol{\mu},\sigma^2)\\&=\frac{ND}{2}\ln(2\pi)-\frac{N}{2}\ln|\boldsymbol{C}|\\&\quad-\frac{1}{2}\sum_{n=1}^{N}(x_n-\mu_n)^{\mathrm{T}}\boldsymbol{C}^{-1}(x_n-\mu_n)\end{aligned}\tag{9}$$

2.3 PPCA参数估计

PPCA模型采用EM算法[9]来估计其参数,具体做法如下:首先通过使用旧的参数值,计算的隐变量的后验概率分布求期望.然后,最大化完整数据对数似然函数的期望就会产生新的参数值,直到迭代终止.完整数据的对数似然函数的形式为:

$$\begin{aligned}&\ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2)\\&=\sum_{n=1}^{N}\{\ln p(x_n|z_n)+\ln p(z_n)\}\end{aligned}\tag{10}$$

分别使用式(3)和式(4),对隐变量上的后验概率分布求期望.

$$\begin{aligned}&\mathrm{E}[\ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{W},\sigma^2)]\\&=-\sum_{n=1}^{N}\{\frac{D}{2}\ln(2\pi\sigma^2)+\frac{1}{2}\mathrm{Tr}(\mathrm{E}[z_n z_n^{\mathrm{T}}])\\&+\frac{1}{2\sigma^2}\|x_n-\mu_n\|^2-\frac{1}{\sigma^2}\mathrm{E}[z_n]^{\mathrm{T}}\boldsymbol{W}^{\mathrm{T}}(x_n-\mu_n)\\&+\frac{1}{2\sigma^2}\mathrm{Tr}(\mathrm{E}[z_n z_n^{\mathrm{T}}]\boldsymbol{W}^{\mathrm{T}}\boldsymbol{W})+\frac{M}{2}\ln(2\pi)\}\end{aligned}\tag{11}$$

在 E 步,利用旧的参数去计算期望值.

$$\mathrm{E}[z_n]=M^{-1}\boldsymbol{W}^{\mathrm{T}}(x_n-\bar{x})\tag{12}$$

$$\mathrm{E}[z_n z_n^{\mathrm{T}}]=\sigma^2 M^{-1}+\mathrm{E}[z_n]\mathrm{E}[z_n]^{\mathrm{T}}\tag{13}$$

在 M 步,估计新的参数值.

$$\boldsymbol{W}_{\mathrm{New}}=[\sum_{n=1}^{N}(x_n-\bar{x})\mathrm{E}[z_n]^{\mathrm{T}}][\sum_{n=1}^{N}\mathrm{E}[z_n z_n^{\mathrm{T}}]]^{-1}\tag{14}$$

$$\begin{aligned}\sigma^2_{\mathrm{New}}&=\frac{1}{ND}\sum_{n=1}^{N}\{\|x_n-\bar{x}\|^2\\&-2\mathrm{E}[z_n]^{\mathrm{T}}\boldsymbol{W}_{\mathrm{New}}^{\mathrm{T}}(x_n-\bar{x})+\mathrm{Tr}(\mathrm{E}[z_n z_n^{\mathrm{T}}]\boldsymbol{W}_{\mathrm{New}}^{\mathrm{T}}\boldsymbol{W}_{\mathrm{New}})\}\end{aligned}\tag{15}$$

传统 PCA 对协方差矩阵的特征分解的计算复杂度为 $O(D^3)$,这里描述的 EM 算法没有显式地建立协方差矩阵,只对前 M 个特征向量和特征值感兴趣,计算复杂度为 $O(NDM)$. 如果问题维度 D 特别大的时候,$M \ll D$,这与 $O(D^3)$ 相比,计算量极大的降低了.

3　算法框架

本文的研究主要针对大尺度优化问题如何进行有效分组,并不侧重于具体的优化算法,因此可以把其视为大尺度优化问题的通用求解框架.

3.1　变量正负相关性学习

对于大尺度变量重叠型优化问题,在演化的过程中,通过不断地对决策变量正负相关性进行学习,对决策变量进行动态分组演化求解. 作为论文的分组基础,我们首先引入变量投影和变量相关的定义.

定义 1　投影值,假设 $\boldsymbol{X}$ 为 D 维空间中的可行解种群,$\boldsymbol{X}=(x_1,x_2,\cdots,x_N)$,$N$ 为种群大小. m 为 D 维空间的可行解种群的中心点.

$$m=\sum_{i=1}^{N}\omega_i x_{i:N}\tag{16}$$

$$\sum_{i=1}^{N}\omega_i=1,\omega_1\geqslant\omega_2\geqslant\cdots\geqslant\omega_N\geqslant 0\tag{17}$$

$\omega_{i=1\cdots N}\in\mathbb{R}^+$,是种群组合权重系数. 当 $\omega_{i=1\cdots N}=\frac{1}{N}$时,式(16)就是计算所有种群的平均值.

$\boldsymbol{v}$ 为 D 维空间所对应的最大主成分的特征向量,$\boldsymbol{v}=(v_1,v_2,\cdots,v_D)$. 每个可行解相对于种群中心点在空间主轴上的投影值可定义为:

$$\begin{aligned}\boldsymbol{p}&=|\boldsymbol{x}_{\mathrm{best}}-\boldsymbol{m}|\cos\theta\\&=\frac{|\boldsymbol{x}_{\mathrm{best}}-\boldsymbol{m}|\times|\boldsymbol{v}|\times\cos\theta}{|\boldsymbol{v}|}\\&=\frac{(\boldsymbol{x}_{\mathrm{best}}-\boldsymbol{m})\cdot\boldsymbol{v}}{|\boldsymbol{v}|}\end{aligned}\tag{18}$$

其中,θ 为 $\boldsymbol{x}_{\mathrm{best}}-\boldsymbol{m}$ 和 $\boldsymbol{v}$ 的夹角.

将投影值 $\boldsymbol{p}$ 按式(19),进行 Min-Max 归一化到 $[-1,1]$ 区间,并以区间间隔 $\tau=0.2$ 进行等分,变量投影值的直方分布图如图 1 所示.

$$p_i^r=\frac{2*(p_i-p_{\min})}{p_{\max}-p_{\min}}+1,i\in[0,D]\tag{19}$$

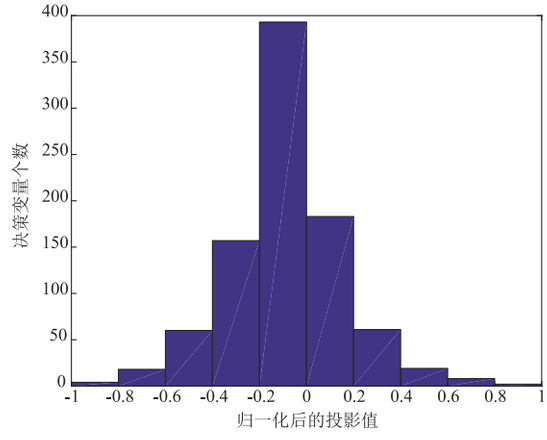


图1　投影值归一化后的直方分布图

定义 2　变量相关性,如果两个决策变量投影值归一化后的 $\boldsymbol{p}^r$ 分布在同一区间,则认为这两个决策变量具有正相关性.

$$\lceil\frac{p_i^r}{\tau}\rceil=\lceil\frac{p_j^r}{\tau}\rceil,i,j\in[0,D],\tau\in[0,1]\tag{20}$$

其中,归一化后的区间间隔 $\tau(0<\tau<1)$ 的大小将直接影响决策变量相关性的划分.

根据决策变量的 $\boldsymbol{p}^r$ 值,按照以下策略对决策变量进行分组.

(1)按照区间间隔 τ 划分区间,在同一区间的决策变量分为一组.

(2)为了避免新划分组中的决策变量数量过多,影响演化速度. 当每个组决策变量个数超过阈值 ξ 时,可以将该组中所有决策变量按照 $\boldsymbol{p}^r$ 大小以每组 ξ 个决策变量再次进行划分;不足阈值 ξ 时,直接划分为一组.

3.2　基于相关性学习的动态分组算法

算法 1 描述了基于学习决策变量和种群特征轴相互关系的分组方法. 它与传统的协同演化分组算法主要有两个不同之处. 首先,在整个优化过程中,通过不断地学习决策变量对于演化种群前进方向的影响大小关系,来动态调整决策变量的分组. 其次,在第 3 步,在对

种群进行特征值分解的过程中,利用 EM 算法去估计概率主成分的参数,大大降低了算法复杂度.

算法 1 分组算法 grouping

Input: pop:种群;best:最优个体;
Output: groups:决策变量分组结果;
1. Initial ω,τ,ξ
2. $\boldsymbol{m} \leftarrow$ mean (pop ,ω), calculate the mean value from pop with the weight coefficients
3. $\boldsymbol{v} \leftarrow$ PPCA (pop), calculate the principal eigenvector by using the EM algorithm
4. $\boldsymbol{p} \leftarrow$ project (best ,$\boldsymbol{m},\boldsymbol{v}$)
5. $\boldsymbol{p}^r \leftarrow$ Min-Max ($\boldsymbol{p}$)
6. ($p_$order ,dims _ order) $\leftarrow$ sort ($\boldsymbol{p}^r$)
7. bincounts $\leftarrow$ histc ($p_$order ,τ)
8. count $\leftarrow 2/\tau$
9. for $i \leftarrow 1$ to count do
10. if (bincounts[i] < ξ) then
11. groups[k] $\leftarrow$ pop[:,bincounts[i]]
12. else
13. for $j \leftarrow 1$ to bincounts[i] /ξ do
14. groups[k] $\leftarrow$ pop[:,bincounts[i][$(j-1)*\xi, j*\xi$]]
15. $k = k+1$
16. end for
17. end if
18. $k = k+1$
19. end for

3.3 算法复杂度分析

设问题维度为 D,种群大小为 N,隐变量大小为 M. 算法 2 中第 2 步,计算种群均值算法复杂度为 $O(N)$;第 3 步,概率主成分分析算法复杂度为 $O(NDM)$;第 4 步,计算各决策变量在主特征向量上投影值的算法复杂度为 $O(D)$;第 5 ~ 7 步,对投影值进行归一化排序并统计分布频度的算法复杂度为 $O(D\log_2 D)$;第 8 ~ 19 步,根据投影值对决策变量进行分组的算法复杂度为 $O(D)$. 在本算法中,去除所有低阶项,算法 2 的时间复杂度为 $O(NDM)$.

4 仿真试验与分析

4.1 测试函数

为了验证 DECC-CLV 算法对于求解大尺度变量重叠问题的有效性,我们选择了 10 个大尺度变量重叠测试函数进行试验分析,所有测试函数维度 905 维. 其中,$F_1 \sim F_5$ 是变量正相关重叠函数,$F_6 \sim F_{10}$ 是变量负相关重叠函数. 变量重叠正负相关性的设置方法见文献[10],各个测试函数如表 1 所示.

4.2 对比测试算法

为了分析 DECC-CLV 算法的收敛速度和收敛精度,本文选取了六个知名演化算法进行对比试验(PSO[11]、ABC[12]、DECC-G[4]、DECC-D[13]、DECC-DG[5] 和 DECC-DML[13]). 其中,四个协同演化算法对于分解的子问题进行优化求解时,均采用 SaNSDE[14] 算法. 具体区别如表 2 所示.

表 1 试验中使用的测试函数

Type	Fun	Name	Range
Conforming Overlapping Subcomponents	F_1	Shifted Schwefels (f_{13} in 文献[10])	[-100,100]
	F_2	Shifted Sphere	[-100,100]
	F_3	Shifted Elliptic	[-100,100]
	F_4	Shifted Rastrigin	[-100,100]
	F_5	Shifted Ackley	[-100,100]
	F_6	Shifted Rosenbrock's	[-100,100]
Conflicting Overlapping Subcomponents	F_7	Shifted Schwefels (f_{14} in 文献[10])	[-100,100]
	F_8	Shifted Sphere	[-100,100]
	F_9	Shifted Elliptic	[-100,100]
	F_{10}	Shifted Rastrigin	[-100,100]
	F_{11}	Shifted Ackley	[-100,100]
	F_{12}	Shifted Rosenbrock's	[-100,100]

表 2 算法分类描述

算法	分组策略	分组次数	每组大小
PSO	无分组,群智能算法		
ABC	无分组,群智能算法		
DECC-G[4]	随机分组	一次	相同
DECC-D[13]	增量分组	多次	相同
DECC-DG[5]	差分分组	多次	相同
DECC-DML[13]	增量分组	多次	不同
DECC-CLV	基于学习策略分组	多次	不同

在对比试验中,测试问题维度 $D = 905$,函数的最大评估次数 $\mathrm{FEs} = 3 \times 10^6$,试验中每一个算法对于每一个测试函数均独立运行 25 次. 试验系统环境为 64 位的 Windows 7 系统,CPU 为 Intel(R) Core(TM) i3 3.60 GHz,内存为 4GB,代码运行环境为 Matlab R2015a.

DECC-CLV 算法种群大小 $N = 50$,区间间隔 $\tau = 0.2$,分组阈值 $\xi = 50$. 试验中的对比算法均使用其原文献中的参数设置,试验结果如表 3 所示,收敛曲线图如图 2 所示.

为了客观公正的评价 DECC-CLV 算法的性能,采用 Wilcoxon 秩和检验方法对试验结果进行统计分析,显著性水平为 0.05. 在试验结果表 3 底部的符号“ - ”、

“+”和“≈”分别表示劣于、优于和相当于 DECC-CLV 的结果.

4.3 试验结果比较分析

通过表 3 和图 2、图 3 分析可发现,DECC-CLV 算法在收敛速度、收敛精度和运行时间方面,与其他算法相比占有较大优势. 特别是在 F_1、F_3、F_4、F_6、F_7、F_9 和 F_{12} 测试函数上,DECC-CLV 算法可以快速收敛,并取得最优解.

表 3　七个算法的平均值、标准差和 Wilcoxon 秩和检验结果

Fun		PSO	ABC	DECC-G	DECC-D	DECC-DG	DECC-DML	DECC-CLV
F_1	Mean	8.81E+09 −	5.69E+10 −	1.62E+10 −	6.10E+10 −	1.82E+10 −	5.97E+09 ≈	2.64E+09
	Std	2.56E+09	1.35E+10	4.79E+09	1.83E+10	2.47E+10	2.24E+09	1.12E+09
F_2	Mean	4.64E+00 −	2.89E+00 −	3.74E−12 −	0.00E+00 −	1.05E−12 +	1.76E−15 +	3.75E−14
	Std	9.87E−01	8.13E−01	2.69E−12	0.00E+00	2.55E−12	3.90E−15	4.27E−14
F_3	Mean	1.32E+12 −	1.03E+13 −	6.71E+12 −	9.98E+11 ≈	2.02E+12 −	6.81E+11 ≈	6.60E+11
	Std	7.57E+11	8.83E+12	2.39E+12	2.82E+11	2.77E+12	3.00E+11	1.17E+11
F_4	Mean	7.17E+08 −	3.23E+09 −	5.24E+08 −	5.96E+09 −	1.42E+09 −	6.40E+08 −	1.57E+08
	Std	2.36E+08	9.41E+08	1.44E+08	1.65E+09	2.47E+09	2.59E+08	5.79E+07
F_5	Mean	7.68E+06 ≈	6.71E+06 ≈	3.90E+06 ≈	3.93E+06 ≈	3.47E+06 ≈	3.91E+06 ≈	3.90E+06
	Std	1.23E+06	1.87E+06	7.81E+05	7.86E+05	1.12E+06	7.83E+05	7.81E+05
F_6	Mean	6.88E+07 −	3.57E+08 −	1.32E+11 −	5.21E+07 −	4.90E+07 −	2.89E+07 −	5.21E+06
	Std	2.13E+07	8.98E+07	1.40E+11	1.66E+07	7.68E+07	4.25E+07	1.66E+06
F_7	Mean	3.21E+10 −	1.17E+12 −	2.67E+11 −	6.90E+11 −	2.32E+11 −	8.59E+10 −	8.89E+09
	Std	8.58E+09	6.34E+11	1.05E+11	3.05E+11	3.18E+11	6.72E+10	6.72E+09
F_8	Mean	6.20E+06 ≈	8.76E+06 ≈	5.42E+06 ≈	5.42E+06 ≈	4.80E+06 ≈	5.42E+06 ≈	5.42E+06
	Std	2.47E+06	2.19E+06	1.08E+06	1.08E+06	1.56E+06	1.08E+06	1.08E+06
F_9	Mean	9.86E+12 ≈	8.01E+13 −	7.98E+13 −	1.10E+13 −	5.49E+13 −	7.09E+12 ≈	6.10E+12
	Std	3.37E+12	4.25E+13	3.11E+13	3.02E+12	3.92E+13	2.70E+12	2.09E+12
F_{10}	Mean	8.21E+08 +	2.52E+09 ≈	7.50E+09 −	8.98E+08 +	2.68E+09 ≈	7.90E+08 +	2.09E+09
	Std	3.67E+08	7.83E+08	2.51E+09	1.86E+08	3.00E+09	7.90E+08	8.90E+08
F_{11}	Mean	6.93E+07 ≈	7.07E+07 ≈	6.94E+07 ≈	6.94E+07 ≈	6.99E+07 ≈	6.98E+07 ≈	6.92E+07
	Std	2.14E+07	3.25E+07	1.39E+07	1.39E+07	1.99E+07	1.93E+07	1.39E+07
F_{12}	Mean	6.33E+11 −	4.14E+11 −	6.29E+11 −	3.66E+11 −	3.70E+11 −	3.63E+11 −	4.51E+10
	Std	2.86E+11	1.03E+11	3.93E+11	1.10E+11	1.12E+11	1.24E+11	1.46E+10
−\+\≈		7\1\4	8\0\4	9\0\3	7\1\4	7\1\4	4\2\6	

在表 3 中,五种算法在 F_5、F_8 和 F_{11} 上都能取得最优解,表现基本上差不多. 在其他九个测试函数上,DECC-CLV 均优于 DECC-G. 由于 DECC-G 是在演化开始前,对所有决策变量进行一次随机分组. 而 DECC-CLV 是演化的过程中不断学习变量之间的相互关系,自适应的去动态调整决策变量的分组情况. 所以,在面对大尺度变量重叠类的优化问题上,DECC-CLV 全面优于 DECC-G.

DECC-CLV 在 F_1、F_3、F_4、F_6、F_7、F_9 和 F_{12} 七个函数上要优于 DECC-D 和 DECC-DG,在 F_5、F_8 和 F_{11} 三个函数上表现相似. 虽然 DECC-D 和 DECC-DG 也都在演化过程中采取了多次分组,但是它们每次分组的大小都是固定的. DECC-CLV 通过学习变量之间的关系,在每次分组时,根据变量之间的关系动态调整决策变量分组的数量和每个组的大小. 通过试验可以看出,这种分组策略在处理变量重叠优化问题中,更加具有有效性和适用性.

DECC-CLV 在 F_2、F_4、F_6、F_7 和 F_{12} 五个函数上优于 DECC-DML. 由于两个算法都采取了多次分组和每次分组的大小不固定策略. 所以,对于部分函数,两个算法表现差不多. 但是,DECC-CLV 的收敛精度和收敛速度整体上要优于 DECC-DML.

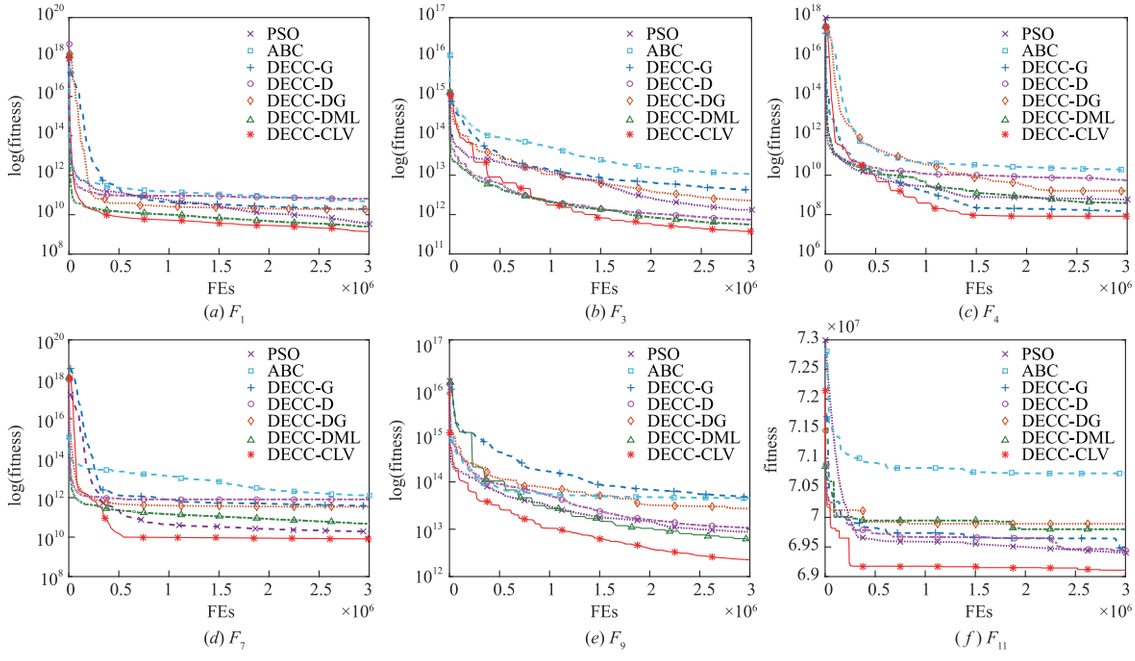


图2 各个算法在测试函数上的收敛曲线图

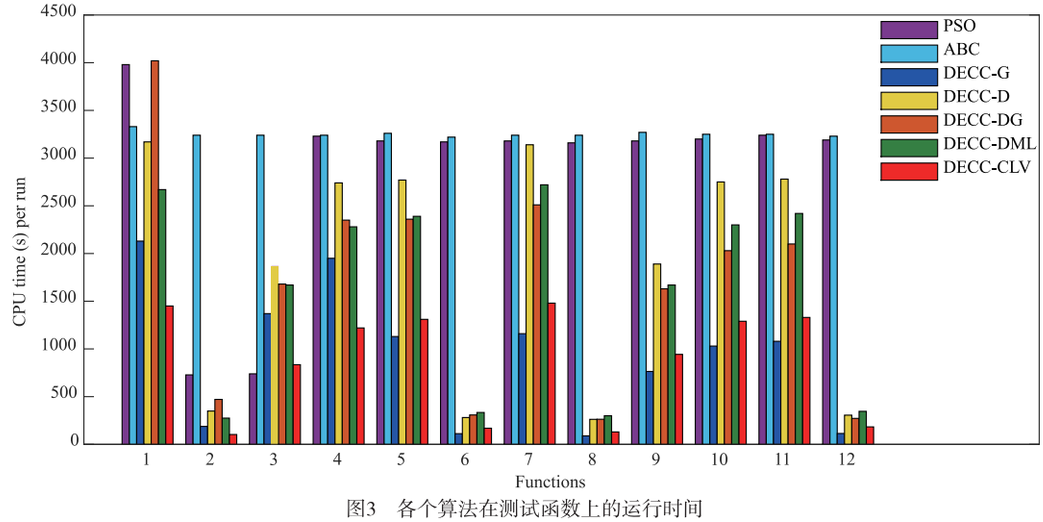


图3 各个算法在测试函数上的运行时间

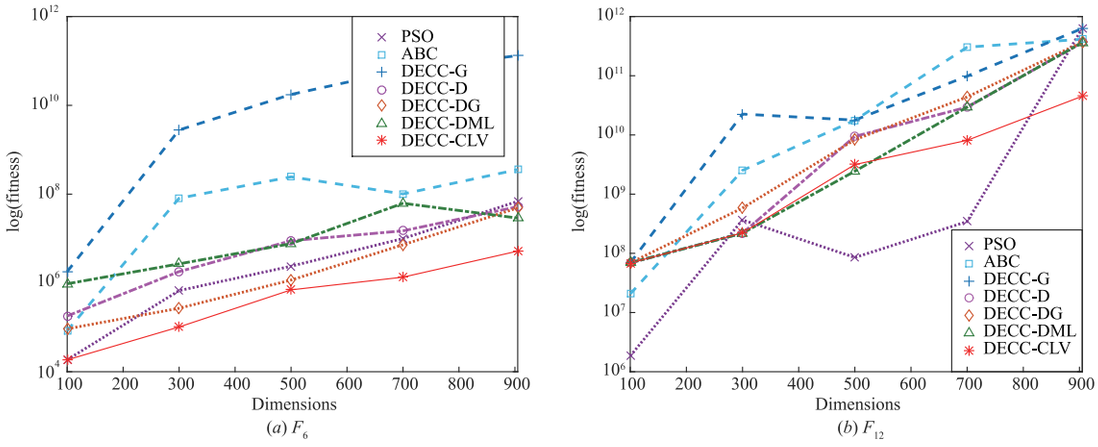
从表3底部的Wilcoxon秩和检验的统计结果可以看出,DECC-CLV算法相对于其他近几年出现的国际知名协同演化算法更具有明显的优势. 表4给出了DECC-CLV和其他算法的Friedman检验排名,DECC-CLV排在第一,从统计学上说明了DECC-CLV算法的优势.

综合以上试验分析可以说明DECC-CLV算法具有较快的收敛速度、较强的收敛精度和较好的函数适用性,有效提高了对大尺度变量重叠复杂问题的收敛效率.

4.4 算法的稳定性分析

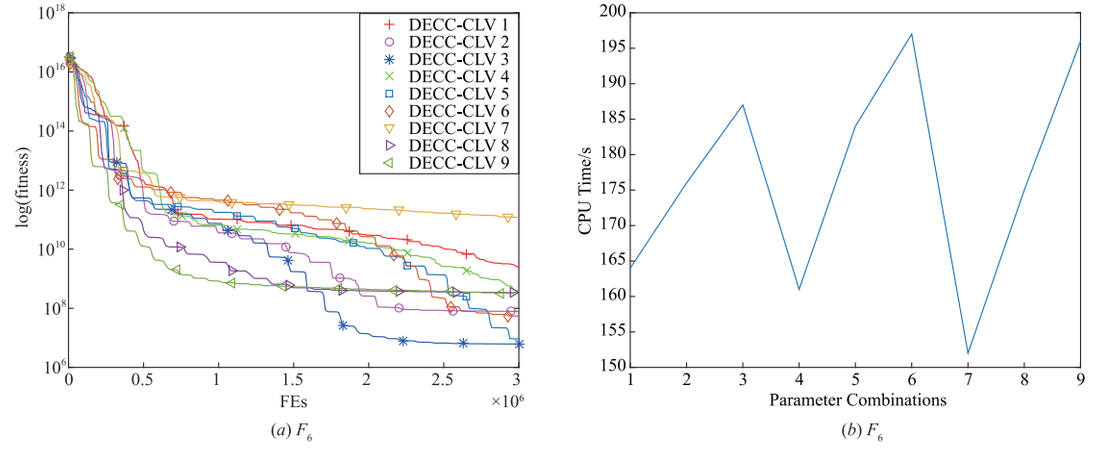
为了测试算法在不同维度下的表现情况,我们将所有对比算法在问题维度 $D=[100,300,500,700]$ 情况下分别进行对比试验. 由于篇幅有限,我们从决策变量正相关和负相关的测试函数各选一个 $F6$ 和 $F12$ 进行测试.

从图4看出,对于决策变量具有负相关性的问题时,当维度小于300的时候,PSO和ABC比基于协同演化策略的算法效果要好;但是当维度大于300的时候,协同演化策略表现出明显的优势. 基于变量正负相关性学习的DECC-CLV算法在不同维度下都要比其他的基于协同演化策略的算法表现要好,表现出了算法的稳定性.



(a) F_6　　(b) F_{12}

图4　各个算法在不同维度下收敛曲线图



(a) F_6　　(b) F_6

图5　不同参数组合在F_6上的收敛曲线和时间对比图

表 4　各算法的 Friedman 检验排名

算法	排名
PSO	4. 33
ABC	6. 25
DECC-G	4. 79
DECC-D	4. 17
DECC-DG	4. 08
DECC-DML	2. 63
DECC-CLV	1. 75

4.5　参数敏感性分析

本算法涉及的参数有两个:区间间隔 $\tau(0<\tau<1)$ 和分组阈值 ξ. 为了测试不同参数下,算法的稳定性,我们采用正交试验设计表在 $F6$ 测试函数上进行精度和时间对比试验.

表 5　正交试验设计表

序号	1	2	3	4	5	6	7	8	9
τ	0. 1	0. 1	0. 1	0. 2	0. 2	0. 2	0. 4	0. 4	0. 4
ξ	50	100	200	50	100	200	50	100	200

通过图 5 可以看出,对于变量正相关性类问题,不同参数组合对算法影响较大;对于变量负相关性类问题,不同参数组合对算法影响较小. 区间间隔 τ 对于算法收敛精度影响较大,区间间隔越大,不相关的决策变量分到一个区间的可能性就越大,算法求解精度就越低. 分组阈值 ξ 对算法的运行时间影响较大,阈值越小,分的组越多,每组的决策变量就越少,算法求解速度就越快.

5　结论

本文提出了一种新的基于学习变量相关性进行分组的协同演化差分进化算法 DECC-CLV,通过对 12 个大尺度变量重叠类函数进行测试,试验结果表明与当前主流算法相比,本文算法在求解的精度与收敛速度上具有更优的性能.

进一步的研究将集中在如下几点:1)引入新的决策变量分类机制,使其能够自动检测求解问题所属的类型,增强分组算法的鲁棒性;2)尝试增加算法的自适

应控制机制,使其能够根据优化过程中的不同状态(停滞区、加速区等),采用不同的搜索机制,从而加速算法的收敛速度.

参考文献

[1] 巩敦卫,季新芳,孙晓燕. 基于集合的高维多目标优化问题的进化算法[J]. 电子学报,2014,42(1):77-83.
GONG Dun-wei,JI Xin-fang,SUN Xiao-yan. Solving many-objective optimization problems using set-based evolutionary algorithms[J]. Acta Electronica Sinica,2014,42(1):77-83. (in Chinese)

[2] POTTER M A,JONG K A D. A cooperative coevolutionary approach to function optimization[A]. Proceedings of The Parallel Problem Solving From Nature-PPSN III[C]. Berlin Heidelberg: Springer,1994. 249-257.

[3] CHUNYU H U,LIU H,ZHANG P. Cooperative co-evolutionary artificial bee colony algorithm based on hierarchical communication model[J]. Chinese Journal of Electronics,2016,25(3): 570-576.

[4] YANG Z,TANG K,YAO X. Large scale evolutionary optimization using cooperative coevolution[J]. Inform Sciences,2008,178(15): 2985-2999.

[5] OMIDVAR M N,LI X,MEI Y,et al. Cooperative co-evolution with differential grouping for large scale optimization[J]. Evolutionary Computation IEEE Transactions on,2014,18(3): 378-393.

[6] CHEN W,WEISE T,YANG Z,et al. Large-scale global optimization using cooperative coevolution with variable interaction learning[A]. Proceedings of the Parallel Problem Solving From Nature-PPSN Xi[C]. Kraków,Poland: Springer,2010. 300-309.

[7] TIPPING M E,BISHOP C M. Probabilistic principal component analysis[J]. Journal of the Royal Statistical Society,1999,61(3): 611 - 622.

[8] BISHOP C M. Pattern Recognition and Machine Learning[M]. Berlin:Springer,2006. 571-586.

[9] 周鑫. 基于EM算法的G0分布参数最大似然估计[J]. 电子学报,2013,41(1): 178-184.
ZHOU Xin. An EM algorithm based maximum likelihood parameter estimation method for the G0 distribution[J]. Acta Electronica Sinica,2013,41(1):178-184. (in Chinese)

[10] TANG K,LI X,SUGANTHAN P N,et al. Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization[J]. Nature Inspired Computation & Applications,2013,7(33):1-8.

[11] KENNEDY J. Particle swarm optimization[A]. Proceedings of the Neural Networks,1995[C]. Perth,Western Australia: IEEE,1995. 1942-1948

[12] KARABOGA D. An Idea Based on Honey Bee Swarm for Numerical Optimization[D]. Kayseri: Erciyes University,2005.

[13] OMIDVAR M N,LI X,YAO X. Cooperative co-evolution with delta grouping for large scale non-separable function optimization[A]. Proceedings of the Evolutionary Computation,2010 IEEE Congress on[C]. Barcelona,Spain: IEEE,2010. 1-8.

[14] YANG Z,TANG K,YAO X. Self-adaptive differential evolution with neighborhood search[A]. Proceedings of the Evolutionary Computation,2008[C]. Hong Kong,China:IEEE,2008. 1110-1116.

作者简介

王豫峰 男,1982 年生于山东平度,武汉大学计算机学院博士生,研究方向为智能计算、机器学习、数据降维等.
E-mail: shandian9876@163.com

董文永 男,1973 年生于河南南阳,计算机软件与理论博士,现为武汉大学计算机学院教授、博士生导师. 长期从事系统科学、仿真与控制、演化计算、并行计算、机器学习、数据挖掘等方面的工作.
E-mail: dwy@whu.edu.cn

董学士 男,1985 年出生于山东日照,武汉大学计算机学院博士生,研究方向为机器学习、智能计算等.
E-mail:dxs_cs@163.com.