Journal of Computer Applications
计算机应用,2017,37(11):3219-3225
ISSN 1001-9081
CODEN JYIIDU
2017-11-10
http://www.joca.cn

文章编号:1001-9081(2017)11-3219-07
DOI:10.11772/j.issn.1001-9081.2017.11.3219

求解大规模优化问题的新型协同差分进化算法

董小刚[1],邓长寿[1*],谭毓澄[2],彭　虎[1],吴志健[3]
(1.九江学院 信息科学与技术学院,江西 九江 332005;　2.九江学院 理学院,江西 九江 332005;
3.软件工程国家重点实验室(武汉大学),武汉 430072)
(*通信作者电子邮箱 dengtju@aliyun.com)

摘　要:基于分而治之的策略,研究求解大规模优化问题的新方法。首先,基于加性可分性原理提出一种改进的变量分组方法,该方法以随机取点的方式,成对检测所有变量之间的相关性;同时,充分利用相关性学习的信息,对可分变量组进行再次降维;其次,引入改进的差分进化算法作为新型子问题优化器,增强了子空间的寻优性能;最后,将两项改进引入到协同进化框架构建 DECC-NDG-CUDE 算法。在 10 个选定的大规模优化问题上进行分组和优化两组仿真实验,分组实验结果表明新的分组方法能有效识别变量的相关性,是有效的变量分组方法;优化实验表明,DECC-NDG-CUDE 算法对 10 个问题的求解相对于两种知名算法 DECC-DG、DECCG 在性能上具备整体优势。

关键词:大规模优化;变量分组;加性可分;优化器;协同进化
中图分类号:TP18　　**文献标志码**:A

Cooperative differential evolution algorithm for large-scale optimization problems

DONG Xiaogang[1], DENG Changshou[1*], TAN Yucheng[2], PENG Hu[1], WU Zhijian[3]
(1. *School of Computer Science and Technology, Jiujiang University, Jiujiang Jiangxi* 332005, *China*;
2. *College of Science, Jiujiang University, Jiujian Jiangxi* 332005, *China*;
3. *State Key Laboratory of Software Engineering (Wuhan University), Wuhan Hubei* 430072, *China*)

Abstract: A new method of large-scale optimization based on divide-and-conquer strategy was proposed. Firstly, based on the principle of additive separability, an improved variable grouping method was proposed. The randomly accessing point method was used to check the correlation between all variables in pairs. At the same time, by making full use of the interdependency information of learning, the large groups of separable variables were re-grouped. Secondly, a new subcomponent optimizer was designed based on an improved differential evolution algorithm to enhance the subspace optimization performance. Finally, this two kinds of improvements were introduced to co-evolutionary framework to construct a DECC-NDG-CUDE (Cooperative differential evolution with New Different Grouping and enhancing Differential Evolution with Commensal learning and Uniform local search) algorithm. Two experiments of grouping and optimization were made on 10 large-scale optimization problems. The experimental results show the interdependency between variables can be effectively identified by the new method of grouping, and the performance of DECC-NDG-CUDE is better than two state-of-the-art algorithms DECC-D (Differential Evolution with Cooperative Co-evolution and differential Grouping) and DECCG (Differential Evolution with Cooperative Co-evolution and Random Grouping).

Key words: large-scale optimization; variable grouping; additive separability; optimizer; cooperative co-evolution

0　引言

工业界和学术界各类数据规模的爆炸式增长,使得大规模数据处理技术成为各领域研究的热点,数据优化领域也进入了大规模时代。大规模黑盒优化问题高度复杂,无法采用传统的数学方法有效求解,其中决策变量的大规模性是构成求解难度的关键因素。进化计算是一类求解复杂优化问题的高效方法,过去二十多年的研究和应用表明,在小规模优化问题上进化算法取得了理想求解结果;然而,当优化问题的规模迅速增大时,进化计算面临严重的挑战,求解性能恶化严重。如何利用进化计算方法,求解大规模优化问题,是目前科学和工程应用领域的研究热点。

1　相关工作

当前,针对大规模优化问题的求解,学者们进行了多方面的研究工作,这些工作可归纳为以下两个方面。

一方面,基于先进的进化计算算法[1-3]进行改进,开发可靠性、精确性更高的进化计算算法(新的进化算子、混合进化模式、增强局部搜索、代理模型等),从而提高大规模优化问题的求解性能。如:文献[4]的研究证明将进化计算与其他

收稿日期:2017-05-11;**修回日期**:2017-06-07。
基金项目:国家自然科学基金资助项目(61364025);江西省教育厅科技项目(GJJ161072,GJJ161076)。
作者简介:董小刚(1979—),男,陕西宝鸡人,讲师,硕士,CCF 会员,主要研究方向:智能计算;　邓长寿(1972—),男,安徽合肥人,教授,博士,CCF 会员,主要研究方向:智能计算、大数据;　谭毓澄(1964—),男,江西九江人,副教授,硕士,主要研究方向:应用数学、智能计算;　彭虎(1981—),男,湖南长沙人,副教授,博士,CCF 会员,主要研究方法:智能计算、大数据;　吴志健(1963—),男,江西上饶人,教授,博士、CCF 会员,主要研究方向:智能计算、并行计算、智能信息处理。

技术进行混合,可以有效提高优化算法求解大规模问题的性能。文献[5]将高效的局部搜索技术与进化算法结合,提出一种文化基因算法(Memetic Algorithm, MA),该算法在一定程度上改善了复杂的组合优化问题的求解性能。文献[6]将正交交叉和反向学习结合,提出一种反向正交交叉的差分进化算法,实验结果表明,该算法对大规模优化问题的求解性能相对于其他几种知名的差分进化算法有明显的提升。文献[7]利用改进的精英学习进化算子和岛模型两种机制,对差分进化算法的优化性能进行改进,并将其部署到分布式平台 Hadoop 上,实验结果证明该方法可有效提高差分进化算法求解大规模优化问题的性能,同时具有较好的加速比。文献[8]提出一种代理模型辅助的群智能优化算法;该算法采用代理模型辅助的粒子群算法及基于粒子群的社会学习算法的协同优化来求解高维优化问题;两组不同维度的仿真实验结果表明,该方法所求得的解质量高于其他比较算法。

另一方面,引入协同进化思想,提高求解大规模优化问题的性能。协同进化思想的引入可分为两类。第一类,基于多个种群的协同进化。如文献[9]针对大规模数据优化问题,提出一种并行差分进化算法,该算法将协同进化思想引入到差分进化算法中,利用多个差分进化算法随机分解大规模优化问题,然后进行并行求解。该方法在大规模非线性函数优化问题的求解上取得了更高的精度和效率。文献[10]提出一种随机扩散搜索的协同差分进化算法,该算法利用随机扩散搜索策略将种群划分为成功和失败两个种群,利用不同进化策略并行协同进化,定期对两个种群进行信息交互。仿真实验表明该算法较基本差分进化和粒子群算法具有收敛速率和精度上的优势。第二类,基于分而治之的策略将大规模优化问题分解为多个子问题进行协同优化。如:文献[11]针对不可分问题提出一种决策变量随机分组(Random Grouping)的方法,该方法不考虑变量之间依赖关系,随机地划分变量分组。并通过多个轮次的进化,提高了相关变量进入同一分组的概率,一定程度上提高了大规模优化问题的求解效果。文献[12]针对固定大小的分组,深入分析了种群大小和分组大小对优化器求解性能的影响。在此基础上提出将部分评价资源用于对二者进行自适应调整的协同进化方法。实验表明该方法在分组大小固定的情况下,相对于其他 CC 框架具备一定的求解优势。文献[13]针对应用分而治之策略时子问题与原问题之间的维度匹配困难,提出一种自评价进化(Self-Evaluation Evolution, SEE)方法。SEE 采用元模型技术来解决由于维度匹配困难所造成的高计算消耗问题,实验结果表明,SEE 相对于四种代表性优化算法,在大规模优化问题的求解表现更为突出。文献[14]提出一种新的两阶段求解方法:第一阶段,检测搜索空间中变量之间的依赖关系,并据此形成变量的分组;第二阶段,依据经典的协同进化框架进行优化求解。文献[15]提出一种自动分组(Differential Grouping, DG)方法,能较为准确地发现决策变量之间的交互关系,并据此而形成相关变量的分组,一定程度上降低了各子分组之间的依赖关系,促进了大规模优化题求解性能的提升。

本文工作主要包括三方面:1)在 DG 算法的基础上,提出一种新的变量分组(New Different Grouping, NDG)方法,进一步改善了基于加性可分原理进行变量分组的准确性;2)利用前期研究提出的 CUDE(enhancing Differential Evolution with Commensal learning and Uniform local search)算法[16]设计子问题优化器,增强算法对子空间的寻优性能;3)将以上改进引入协同进化框架,构建求解大规模优化问题的新型协同差分进化算法 DECC-NDG-CUDE(Cooperative differential evolution with NDG and CUDE)算法,并在选定的测试集上进行仿真实验,实验结果表明,DECC-NDG-CUDE 能进一步提高大规模优化问题的求解性能,是一种有效的算法。

2 新型变量分组策略

2.1 函数的加性可分性原理

大规模黑盒优化问题由于其数学模型未知和不可获取,传统的数学方法,如求导法、单纯形法、梯度法等不再适用。针对这一情况,当前广泛采用的方法是通过对决策变量的分组,将问题分解成多个低维度的子问题来进行各个击破的独立求解。然而,当决策变量之间存在复杂的相互依赖关系时,盲目分组显然无法达到有效求解的目的。

基于加性可分性原理对变量之间的依赖关系进行先期学习,然后据此进行变量分组,可以最大限度地降低子问题之间的依赖关系,对提高求解性能有很大的促进作用。

定义 1 如果函数 $f(\boldsymbol{x})$ 满足如下条件:

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} f(x_i) \tag{1}$$

则函数 $f(\boldsymbol{x})$ 为加性可分解函数,其中 $\boldsymbol{x} = (x_1, x_2, \cdots, x_m)$,$x_i$ 为相互排斥的决策变量。

以二维函数 $f(x,y)$ 为例,根据加性可分函数的定义可知:

若 $f(x,y) = f_1(x) + f_2(y)$,显然可得出 $f(x + \Delta x, y) - f(x,y)$ 与 y 的无关。换句话说就是:$f(x_1,y) - f(x_2,y)$ 与 y 无关,即存在如下等式:

$$f(x_1,y_1) - f(x_2,y_1) = f(x_1,y_2) - f(x_2,y_2)$$

以此类推可得出如下推论:

n 维函数 $f(x_1, x_2, \cdots, x_n)$,x_i 与 x_j 之间加性可分,则存在如下等式:

$$f_{|x_i=p,x_j=t_1} - f_{|x_i=q,x_j=t_1} = f_{|x_i=p,x_j=t_2} - f_{|x_i=q,x_j=t_2} \tag{2}$$

其中 $i \neq j$, $p \neq q$, $t_1 \neq t_2$;反之,若:

$$f_{|x_i=p,x_j=t_1} - f_{|x_i=q,x_j=t_1} \neq f_{|x_i=p,x_j=t_2} - f_{|x_i=q,x_j=t_2} \tag{3}$$

则 x_i 与 x_j 相互依赖,不可分。

式(2)和式(3)由加性可分性函数的定义导出,是函数加性可分的性质。

文献[15]利用式(2)构建了大规模优化问题自动分组(DG)算法。DG 算法在可行域中选取 4 个测试点(边界和中心),计算式(2)左右两侧的两个差值 Δ_1 和 Δ_2,通过比较表达式 $|\Delta_1 - \Delta_2|$ 的值和设定阈值的关系来判定决策变量之间是否存在依赖关系,进而形成变量分组。DG 算法合理利用了函数加性可分的性质,能够较为准确地实现决策变量的分组。

然而,DG 算法的实现仍然存在以下几个方面的不足:

1)采用可行域的边界作为测试点;然而,边界本身具有特殊性,不能反映一般性。例如式(4)所示的函数:

$$f(\boldsymbol{x}) = \sum_{i=2}^{n} (x_i * x_{i-1} * (x_{i-1}^2 - 25) * (i-1)) + x_1 * x_n * (x_n^2 - 25) * n \tag{4}$$

设式(4)所示函数的定义域为 $[-5,5]$。假设 $n = 4$,则 $f(x)$ 为 4 维函数。依据 DG 算法,用定义域的边界构造 4 个

点,检测第一维和第二维的相关性,具体过程如下:

设 $\boldsymbol{p}_1 = [-5,-5,-5,-5]$, $\boldsymbol{p}_2 = \boldsymbol{p}_1, p_2(1) = 5$。由此,求 Δ_1 为:

$$\Delta_1 = f(\boldsymbol{p}_1) - f(\boldsymbol{p}_2) \tag{5}$$

设 $\boldsymbol{p}_3 = \boldsymbol{p}_1, \boldsymbol{p}_4 = \boldsymbol{p}_2, p_3(2) = 0, p_4(2) = 0$,由此,求 Δ_2 为:

$$\Delta_2 = f(\boldsymbol{p}_3) - f(\boldsymbol{p}_4) \tag{6}$$

$$|\Delta_1 - \Delta_2| \tag{7}$$

则式(7)的值必定为 0。据此,DG 算法将第一维和第二维识别为不相关变量。此结论显然与函数本身的性质不符。

2)排除已经形成分组的所有相关变量,导致依赖关系检测不全面,影响了分组的成功率。

3)没有充分地利用分组学习阶段所获得的信息。比如将所有可分变量归为一个分组,当可分变量数量比较大时,所形成的分组规模较大,无法达到有效降维的目的,并且浪费了较多的评价资源。

2.2　新型分组算法

为了进一步提高变量分组的精度,提高大规模优化问题的求解性能。针对 DG 算法 3 个不足进行改进,提出一种新的变量分组(NDG)算法。

首先,NDG 算法避免采用可行域的边界作为测试点,而以边界附近的随机值为测试点。

其次,NDG 算法采用遍历形式对变量相关性进行检测,也就说并不因为形成分组而排除某个变量的相关性检测,从而避免遗漏。

最后,为对相关性学习的结果进行充分利用,尤其是针对学习后判定为可分的变量组,若其大小仍然较大,则对其进行二次分组,进一步降低维度。

NDG 算法的基本流程如下:

算法 1　NDG(*func*, *lbs*, *ubs*, *eps*)

1)　设置 *Dims* = {1,2,…,1 000},定义可分组变 *Seps*,相关分组变量 *Cgroups*,中间集合变量 *group*, *Agroups*, *Bgroups*;初始化上下界变量 *lb*, *ub*,评价次数变量 *FES* = 0;
2)　while (*Dims* 不为空)
3)　　group = [*Dims*(1)];
4)　　定义两个 1 000 维向量 $\boldsymbol{p}_1$ 和 $\boldsymbol{p}_2$,各维在下界附近随机取值;
5)　　将向量 $\boldsymbol{p}_2$ 的第 *Dims*(1) 维设置为上届附近的随机值;
6)　　$\Delta_1 = f(\boldsymbol{p}_1) - f(\boldsymbol{p}_2)$, *FES* = *FES* + 2
7)　　For i = 2:length(*Dims*)
8)　　　$\boldsymbol{p}_3 = \boldsymbol{p}_1$;
9)　　　$\boldsymbol{p}_4 = \boldsymbol{p}_2$;
10)　　　$p_3(i) = p_4(i) = (lb + ub)/2$
11)　　　$\Delta_2 = f(\boldsymbol{p}_3) - f(\boldsymbol{p}_4)$, *FES* = *FES* + 2;
12)　　　if abs($\Delta_1 - \Delta_2$) > *eps*
　　　　group = [*group*, *Dims*(i)]
13)　　　end if
14)　　end For 循环
15)　　if length(*group*) == 1
　　　Seps = [*Seps*, *group*];
16)　　Else
　　　Agroups = {*Agroups*{1:*end*}, *group*}
　　　group_u = *union*(*group_u*, *group*)
17)　　end if
18)　　if (length(*Dims*) > 0)
　　　设置 *Dims*(1) = [ ]
19)　　end if
20)　　end while 循环
21)　合并 *Agroup* 中包含相同元素的相关组,将结果赋值给 *Cgroup*
22)　返回 *Seps*, *Cgroups*;

其中,第 4)和第 5)步的随机值 NDG 采用上界或下界附近的随机值,有别于 DG 算法直接取上届或者下界。第 12)步 *eps* 为阈值,当小于该阈值时,即认为两个差值相等。大于该阈值时,认为两个差值不相等。并以此判断变量之间的相关性。第 22)步返回的可分组,在读取分组结果时进行判断,若仍然较大,则进行二次分组,按顺序每 50 个构成一个新的分组,最后一组为所有剩余变量。

3　基于改进差分进化算法的子问题优化器

高效的优化器是提高大规模优化问题求解性能的有力工具。2016 年文献[16]提出一种改进的差分进化算法(CUDE),实验表明 CUDE 寻优能力突出。为进一步提高求解大规模优化问题的性能,本文将 CUDE 作为子问题优化器引入到引入框架中。

3.1　改进的进化策略和参数调整方案

缩放因子 F 和交叉概率 CR 是影响 DE 算法性能的两个核心参数。关于 DE 参数的大量研究已经证明固定的参数设置并非提高算法性能的最好设置。一般来说,较大的 F 和 CR 适合用于进化初期提升算法的全局搜索性能。而较小的 F 和 CR 适合在进化后期加强算法的局部开采性能,并加速收敛;然而,简单的前大后小的设置对提高算法的性能意义不大。

CUDE 采用双边高斯分布进行 F 和 CR 的动态自适应更新。具体设置依据式(8)和(9)进行:

$$S_{\text{lower}} = \begin{cases} F_{\text{lower}} = N(0.5, 0.1) \\ CR_{\text{lower}} = N(0.1, 0.1) \end{cases} \tag{8}$$

$$S_{\text{upper}} = \begin{cases} F_{\text{upper}} = N(0.8, 0.1) \\ CR_{\text{lower}} = N(0.9, 0.1) \end{cases} \tag{9}$$

其中:N 代表高斯函数,S_{lower} 和 S_{upper} 代表双边参数设置的上、下界。

另外,为避免单一进化策略对问题的依赖性,CUDE 选择 DE/rand/1、DE/best/1、DE/target-to-rand/1 三种进化策略与双边高斯参数设置进行组合使用。

对双边高斯参数设置和三种进化策略,进行两两组合,共构造 6 种进化方案。然后,在每一代进化中 CUDE 采用轮盘赌的方式为每一个个体从 6 种方案中选择一个进行进化。在每一代进化之后,记录每个方案对种群中每个个体的成功更新次数 $S_{sch,i}$ 和尝试更新次数 $T_{sch,i}$,并进行相除得到更新成功率参数 $SR_{sch,i}$, $SR_{sch,i}$ 的计算公式如式(10)所示:

$$SR_{sch,i} = S_{sch,i}/T_{sch,i} \tag{10}$$

依据 $SR_{sch,i}$,构建学习字典二维表。在下一代中依据学习字典,采用轮盘赌的方式选择每个个体的进化方案。

3.2　基于均匀设计的局部搜索

采用基于均匀设计(Uniform Local Search, ULS)的局部搜索技术,CUDE 有效地平衡了 DE 算法的勘探和开采能力。均匀设计是一种高质量的实验设计方法,其基础理论为数论中的一致分布理论,最早由 Wang 等[17]和 Fang 等[18]提出。与同为实验设计方法的正交设计相比,均匀设计的优势是实验

次数更少,这在以个体适应度为评价体的进化算法中尤为重要。近年来已有不少学者将均匀设计引入到进化计算算法中,如文献[19]在利用均匀设计来进行算法种群的初始化。

均匀设计依据均匀设计表来安排实验方案。一个均匀设计表可表示为 $U_n(q^s)$,其中 n 为表的行数,代表要安排的实验次数; S 为列数,代表要考虑的因素数; q 代表每个因素要考虑的水平数。CUDE 所采用的 ULS 局部搜索方法,选择均匀设计表 $U_6(6^6)$ 进行实验方案安排。关于 $U_6(6^6)$ 的构造参考文献[16]。

表 1 均匀设计表 $U_6(6^6)$

Tab. 1 Uniform design table $U_6(6^6)$

No.	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

CUDE 采用以上两个方面的改进提高了算法的寻优能力,算法详细流程及参数见文献[16]。

ULS 在每一代进化中随机选择两个个体进行局部搜索。以二维空间为例,假设所选取的两个个体为 $A(x_1,y_2)$ 和 $B(x_2,y_2)$。则 ULS 首先会将 A 和 B 构成的二维子空间的每一维均匀地分解成 6 个部分,每个部分看作一个因素。然后,依据均匀设计表 $U_6(6^6)$ 组合每个水平上的各个因素,构造出 6 个实验体。最后,对 6 个实验个体进行评价,选择最优的替换 A 和 B 的一个完成局部搜索。ULS 的详细流程如算法 2。需说明的是 ULS 在进化的每一代中仅执行一次。

算法 2 ULS 算法。

1)输入:种群 P,评价次数计数器 FEs。

2)从 P 中随机选择两个个体 $x_{i,g}$ 和 $x_{j,g}$。

3)根据 $x_{i,g}$ 和 $x_{j,g}$,利用 $U_6(6^6)$ 进行均匀,产生 6 个实验个体 $y_1,y_2,\cdots,y_6$。

4)评价 6 个实验个体。

5)选取适应度最高的一个实验个体 O。

6)用 O 替换当前种群 P 中的个体 $x_{i,g}$,替换条件为 $f(x_{i,g})>f(o)$。

7) $FEs = FEs + 6$。

8)返回新种群 P 和参数 FEs。

3.3 协同进化框架

为了进一步提高求解大规模优化问题的性能,将 NDG 和 CUDE 引入到协同进化框架之下,构建 DECC-NDG-CUDE 算法(算法 3)。

算法 3 DECC-NDG-CUDE($func$,lbs,ubs,n)。

```
1)  groups = NDG(func,lbs,ubs)
2)  pop = rand(popsize,n)
3)  (best,best_val) = min(func(pop))
4)  for i = 1 to cycles do
5)    for j = 1 to size(groups) do
6)      indicies = groups[j];
7)      subpop = pop[:,indicies]
8)      subpop = CUDE(best,subpop,FEs)
9)      pop[:,indicies] = subpop
10)     (best,best_val) = min(func(pop))
11)   end for
12) end for
```

4 实验与分析

为了合理公平地评价 NDG 算法对变量进行分组的准确性和有效性,本文仿真分别进行分组和优化两个实验。分组实验验证 NDG 对变量分组的准确性,并将分组结果和文献[15]的 DG 方法进行比较,优化实验验证了 DECC-NDG-CUDE 求解大规模优化问题的性能,并将实验结果与 DECC-DG(Differential Evolution with Cooperative Co-evolution and differential Grouping)[15]、DECCG(Differential Evolution with Cooperative Co-evolution and Random Grouping)[11]两种算法进行比较。

4.1 测试问题

为保证真实性、有效性,选取 CEC2010[20]标准测试集的 10 个 1 000 维的问题进行仿真实验。10 个问题的特征信息如表 2 所示。

$$f_1(x) = \sum_{i=1}^{D}[z_i^2 - 10\cos(2\pi z_i) + 10]$$

其中:可分变量数(Sep_Vars)为 1 000,不可分变量数($NonSep\_Vars$)为 0,不可分变量组数($NonSep\_Groups$)为 0。

$$f_2(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20 + e$$

其中:Sep_Vars 为 1 000,$NonSep\_Vars$ 为 0,$NonSep\_Groups$ 为 0。

$$f_3(x) = f_{\text{rot\_rastrigin}}[z(P_1:P_m)] * 10^6 + f_{\text{rastrigin}}[z(P_{m+1}:P_D)]$$

其中:Sep_Vars 为 950,$NonSep\_Vars$ 为 50,$NonSep\_Groups$ 为 1。

$$f_4(x) = \sum_{k=1}^{D/2m} f_{\text{rot\_rastrigin}}[z(P_{(k-1)*m+1}:P_{k*m})] + f_{\text{rastrigin}}[z(P_{D/2+1}:P_D)]$$

其中:Sep_Vars 为 500,$NonSep\_Vars$ 为 500,$NonSep\_Groups$ 为 10。

$$f_5(x) = \sum_{k=1}^{D/2m} f_{\text{rot\_ackley}}[z(P_{(k-1)*m+1}:P_{k*m})] + f_{\text{ackley}}[z(P_{D/2+1}:P_D)]$$

其中:Sep_Vars 为 500,$NonSep\_Vars$ 为 500,$NonSep\_Groups$ 为 10。

$$f_6(x) = \sum_{k=1}^{D/2m} f_{\text{schwefel}}[z(P_{(k-1)*m+1}:P_{k*m})] + f_{\text{sphere}}[z(P_{D/2+1}:P_D)]$$

其中:Sep_Vars 为 500,$NonSep\_Vars$ 为 500,$NonSep\_Groups$ 为 10。

$$f_7(x) = \sum_{k=1}^{D/m} f_{\text{rot\_rastrigin}}[z(P_{(k-1)*m+1}:P_{k*m})]$$

其中:Sep_Vars 为 0,$NonSep\_Vars$ 为 1 000,$NonSep\_Groups$ 为 20。

$$f_8(x) = \sum_{k=1}^{D/m} f_{\text{rot\_ackley}}[z(P_{(k-1)*m+1}:P_{k*m})]$$

其中:Sep_Vars 为 0,$NonSep\_Vars$ 为 1 000,$NonSep\_Groups$ 为 20。

$$f_9(x) = \sum_{k=1}^{D/m} f_{\text{rosenbrock}}[z(P_{(k-1)*m+1}:P_{k*m})]$$

其中:*Sep_Vars* 为0,*NonSep_Vars* 为1 000,*NonSep_Groups* 为20。

$$f_{10}(x) = \sum_{i=1}^{D-1}[100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2]$$

其中:*Sep_Vars* 为0,*NonSep_Vars* 为1 000,*NonSep_Groups* 为1。

$f_1(x) \sim f_{10}(x)$ 表达式中,$\boldsymbol{x} = (x_1, x_2, \cdots, x_D)$ 为候选解;$\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{o}$ 为偏移候选解,$\boldsymbol{o} = (o_1, o_2, \cdots, o_D)$ 全局最优解;P 是$\{1,2,\cdots,D\}$ 的一个随机排列,D 为问题维度。

4.2　分组实验结果与分析

分组实验中利用 NDG 算法对 10 问题进行变量分组,检验 NDG 算法对变量之间相关性的识别并依据其形成变量组的性能。为比较充分地验证 NDC 的性能,实验分别在两种阈值(10^{-1}和10^{-3})情况下进行实验,记录算法识别的可分变量数、相关变量数、相关变量分组数以及相关变量的识别成功率(所识别的相关变量除以相关变量总数),并将结果和文献[15]中知名的算法 DG 进行对比。实验结果见表 2 和表 3。

表 2　变量分组结果($\varepsilon = 10^{-1}$)

Tab. 2　Grouping result($\varepsilon = 10^{-1}$)

函数	*Sep_Vars*		*NonSep_Vars*		*NonSep_Groups*		成功率/%	
	DG	NDG	DG	NDG	DG	NDG	DG	NDG
f_1	1 000	1 000	0	0	0	0	100.0	100
f_2	1 000	1 000	0	0	0	0	100.0	100
f_3	950	950	50	50	1	1	100.0	100
f_4	500	500	500	500	10	10	100.0	100
f_5	512	500	291	500	36	10	58.2	100
f_6	500	500	500	500	10	10	100.0	100
f_7	1	0	999	1 000	20	20	99.0	100
f_8	20	0	640	1 000	72	20	64.0	100
f_9	79	0	60	1 000	359	202	6.0	100
f_{10}	0	0	1 000	1 000	500	1	100.0	100

表 3　变量分组结果($\varepsilon = 10^{-3}$)

Tab. 3　Grouping result($\varepsilon = 10^{-3}$)

函数	*Sep_Vars*		*NonSep_Vars*		*NonSep_Groups*		成功率/%	
	DG	NDG	DG	NDG	DG	NDG	DG	NDG
f_1	1 000	1 000	0	0	1	1	100.0	100
f_2	1 000	1 000	0	0	1	1	100.0	100
f_3	950	950	50	50	1	1	100.0	100
f_4	500	500	500	500	10	10	100.0	100
f_5	501	500	499	500	10	10	99.8	100
f_6	500	500	500	500	10	10	100.0	100
f_7	0	0	1 000	1 000	20	20	100.0	100
f_8	4	0	996	1 000	20	20	99.6	100
f_9	85	0	173	1 000	49	1	17.3	100
f_{10}	33	0	82	1 000	241	1	8.2	100

表 2 和表 3 的分组结果显示,在两种阈值下,对两个完全可分的问题f_1、f_2 上,NDG 和 DG 算法都能准确的识别出为完全可分问题,说明两种算法能准确的区分出可分解的决策变量;在两种阈值下,对f_3、f_4、f_6 三个部分可分解问题上,NDG 和 DG 算法的结果相同,能准确的识别出可分解变量和相关变量,说明两种算法对着三个问题的分解没有受到阈值的影响,成功识别了变量的相关性,并且相关变量的分组也准确无误。

在f_5、f_7、f_8、f_9、f_{10}部分可分的问题上,NDG 和 DG 两种算法表现出不同的性能。在阈值为 10^{-1}时,对问题f_{10}两种算法分组结果相同;在阈值为 10^{-3}时,DG 算法对相关变量的识别成功率较低,仅有 8.2%。而 NDG 算法仍能准确地识别出所有变量为相关变量,且相关变量分组也准确无误。这一结果表明,对问题f_{10},DG 算法受到了阈值设置的影响,在较小阈值的设置下,不能准确识别相关量,性能弱于 NDG 算法;对问题f_7,在阈值 10^{-3}时,两种算法表现相同,均能有效识别变量的相关性,并准确分组;在阈值为 10^{-1}时,算法 DG 未能完全识别相关变量,但只有 1 个相关变量未能准确识别,成功率为 99%。而 NDG 算法能准确识别所有变量的相关性,性能略优于 DG 算法。

在其他的三个问题f_5、f_8、f_9 上,DG 算法均对变量的相关性识别的成功率均未达到 100%,即没有准确识别所有相关变量,其中f_9在两种阈值下的相关变量识别均非常低。f_5、f_8 两个问题的相关变量的识别率,在阈值为 10^{-1}时分别为 58.2% 和 64%,在阈值为 10^{-3}时分别为 99.8% 和 99.6%。说明 DG 算法对这两个问题的分解受到阈值的影响较大。然而,NDG 算法在两种阈值下,对于三个问题均能准确识别变量的相关性,且分组未出现差错。

综上所述,对于选定的 10 个大规模问题,NDG 算法对变量相关性的识别性能整体上优于 DG 算法,能准确识别出相关变量并形成相关组。

图 1 和图 2 给出了存在差别的几个问题上相关变量识别成功率的柱状图,进一步比较了两种算法分组性能的区别。

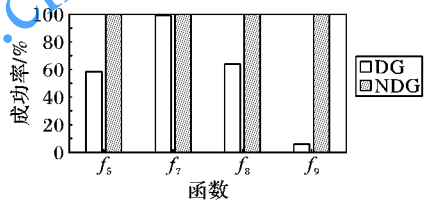

图 1　捕获不可分变量成功率($\varepsilon = 10^{-1}$)

Fig. 1　Success rate of capturing nonseparables variables($\varepsilon = 10^{-1}$)

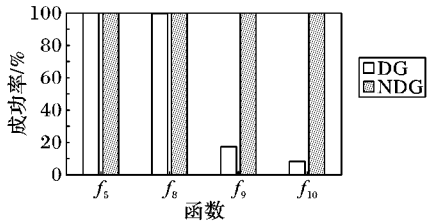

图 2　捕获不可分变量成功率($\varepsilon = 10^{-3}$)

Fig. 2　Success rate of capturing nonseparables variables($\varepsilon = 10^{-3}$)

4.3　优化实验结果与分析

优化实验验证 DECC-NDG-CUDE 算法对 10 个问题的优化求解性能。实验中各参数的设置为优化问题维度 D = 1 000,种群 *popsize* = 50,进化结束条件为适应度评价次数 *FEs* = 3E6。实验独立执行 25 次,记录最优解的平均值和方差。同时,为进行对比分析,实现了 DECCG[7]、DECC-DG[9] 算法,依据原文先相同设置进行仿真实验,并记录结果。实验结果见表 4。

从表 4 的求解结果均值来看,与 DECC-DG 算法相比,DECC-NDG-CUDE 在 7 个问题上优于算法 DECC-DG,3 个问题上差于算法 DECC-DG;与 DECCG 算法相比,DECC-NDG-CUDE 在 7 个问题上优于算法 DECCG,3 个问题上差于 DECCG。这说明 DECC-NDG-CUDE 算法求解 10 个大规模问题的数据结果整体上优于 DECC-DG 和 DECCG 两种算法结果。

为了保证以上数据结果的真实性，采用显著水平为 0.05 的秩和检验（Wilcoson test）对两种算法的实验数据进行检验，检验结果用三种符号“+”“-”“≈”标注在表 4 中。

表 4 优化实验结果

Tab. 4 Optimization results

函数	DECC-DG		DECCG		DECC-NDG-CUDE	
	mean	std	mean	std	mean	std
f_1	4.46E+03 -	1.39E+02	1.33E+03 -	4.23E+01	9.17E+02	2.69E+01
f_2	1.67E+01 +	2.82E-01	1.41E+00 +	9.38E-02	1.76E+01	2.68E-01
f_3	1.55E+08 ≈	2.90E+07	2.42E+08 -	7.90E+07	1.49E+08	6.30E+07
f_4	4.55E+03 -	1.55E+02	1.02E+04 -	3.49E+02	1.89E+03	1.99E+02
f_5	1.03E+01 +	8.40E-01	2.67E+01 -	1.47E+00	1.67E+01	9.86E-01
f_6	2.67E+03 -	5.17E+02	9.47E+04 -	9.36E+03	1.03E+03	1.34E+02
f_7	5.87E+03 -	7.99E+01	1.23E+04 -	7.53E+02	2.86E+03	2.81E+02
f_8	7.51E-13 +	6.64E-14	7.14E+01 -	5.62E+00	4.20E-01	6.95E-01
f_9	1.58E+10 -	3.09E+09	3.11E+04 +	1.61E+04	1.06E+08	4.19E+08
f_{10}	6.51E+10 -	8.87E+09	4.55E+03 +	1.04E+03	4.66E+07	9.21E+07
-	6		7			
+	3		3			
≈	1		0			

表 4 中，“+”“-”“≈”分别表示 DECC-NDG-CUDE 算法弱于、优于和相当于对比算法（DECC-DG 或 DECCG）。首先，从检验统计结果来看，DECC-NDG-CUDE 在 6 个问题（f_1、f_4、f_6、f_7、f_9、f_{10}）上是优于 DECC-DG、3 个问题（f_2、f_5、f_8）上弱于 DECC-DG、1 个问题（f_5）上与 DECC-DG 算法相当。这表明在 f_5 上，与 DECC-DG 相比，DECC-NDG-CUDE 数据结果的优势存在偶然性，实际两个算法的结果不存在差别。其次，与 DECCG 的检验结果与数据结果一致，这说明 DECC-NDG-CUDE 在这 7 个问题上的求解性能优于 DECCG 是真实的，与数据比较结果一致。

图 3 给出了其中 6 个问题的收敛曲线。需要说明的是，在 f_9 和 f_{10} 两个问题的收敛图中，DECC-NDG-CUDE 的收敛曲线差于 DECCG，其原因是这两个函数为完全不可分问题，DECC-NDG-CUDE 变量相关性识别损失了较多的评价资源，影响了后期的优化效果。但是，从 6 幅图的整体情况来看，收敛曲线仍然支持了以上数据分析的结果。

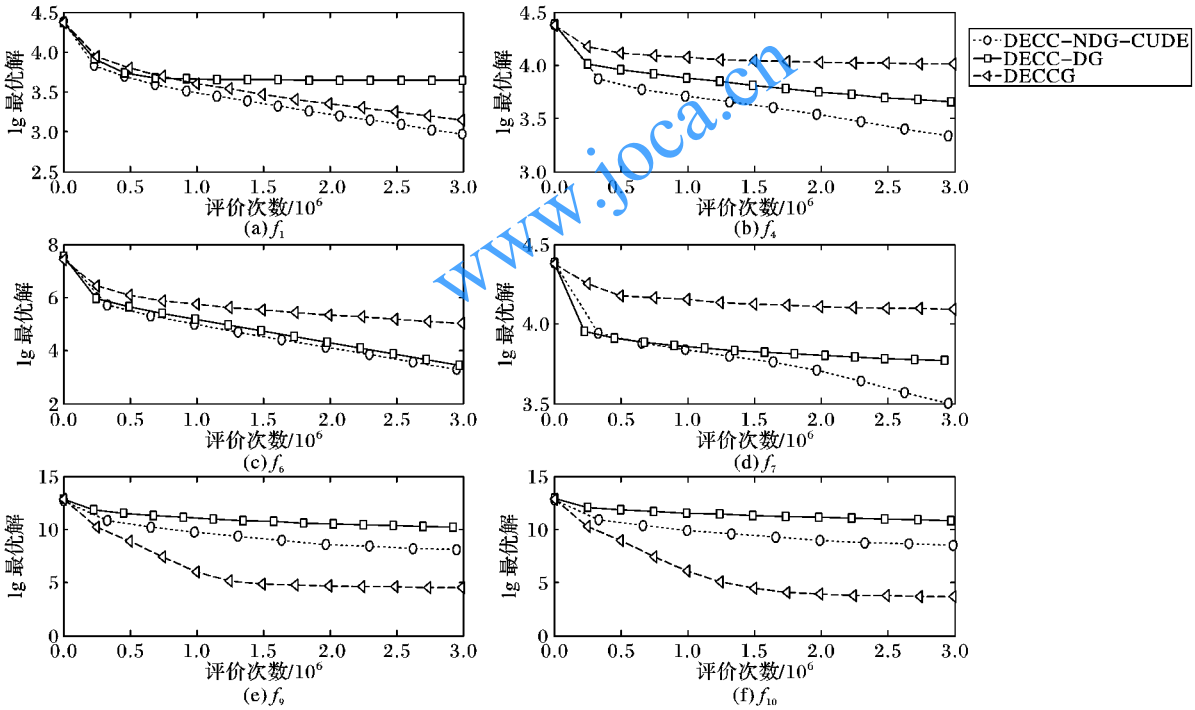

图 3 三种算法在 6 个问题上的收敛曲线

Fig. 3 Convergence curve of three algorithms on six problems

最后，为整体上比较三种算法求解 10 个大规模优化问题的性能，对三种算法优化结果进行 Friedman 排名检验，检验结果为，DECC-NDG-CUDE 的排名值为 1.60，DECC-DG 排名值为 2.10，DECCG 排名值为 2.30。因此，DECC-NDG-CUDE 的 Friedman 检验排名第一，整体性能最好。

综上所述，DECC-NDG-CUDE 算法对于 10 个优化问题的优化性能在整体上优于 DECC-DG 和 DECCG 两种算法。

5 结语

针对大规模优化问题，本文首先提出一种改进的分组方法 NDG，该算法采用随机选取测试点，逐一检测每对决策变量的相关性，并依据相关性对变量进行分组，从而将大规模优化问题划分为多个子问题，降低了子问题之间的依赖关系；同时，为了充分利用变量相关性的学习结果，将较大的可分变量组进行二次分组，在不影响相关组的情况下，降低了待优化子问题的维度。仿真实验表明 NDG 算法提高了变量分组的精确度，能够对协同优化求解起到促进作用；另一方面，为进一步提高优化问题求解进度，在改进的差分进化算法 CUDE 的基础之上，设计新的子问题优化器，并引入到 CC 框架之下，构建 DECC-NDG-CUDE 算法。优化仿真实验的结果表明，在 10 个大规模优化问题上，DECC-NDG-CUDE 能够进一步提高求解精度，求解性能优于 DECC-DG 和 DECCG 两种知名算

法,是求解大规模优化问题的一种有效的算法。在未来的研究工作中,将深入探索优化过程中降低函数评价次数的合理途径,从而减少因变量分组消耗评价次数对问题优化带来的影响。

参考文献(References)

[1] STORN R, PRICE K. Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of Global Optimization, 1997, 11(4):341 -359.

[2] KENNEDY J, EBERHART R. Particle swarm optimization[C]// Proceedings of the 1995 IEEE International Conference on Neural Networks. Piscataway, NJ: IEEE, 1995: 1942 -1948.

[3] KARABOGA D, BASTURK B. A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm[J]. Journal of Global Optimization, 2007, 39(3):459 -471.

[4] GOLDBERG D E, VOESSNER S. Optimizing global-local search hybrids[EB/OL]. [2016-11-20]. http://pdfs.semanticscholar.org/21b8/ae2a75de794a625df6737466483d93441f9b.pdf.

[5] FACHBEREICH V, INFORMATIK E. Memetic algorithms for combinatorial optimization problems: fitness landscapes and effective search strategies[EB/OL]. [2016-11-20]. http://dokumentix.ub.uni-siegen.de/opus/volltexte/2006/181/pdf/merz.pdf.

[6] DENG C, DONG X, YANG Y, et al. Differential evolution with novel local search operation for large scale optimization problems[C]// Proceedings of the 6th International Conference Advances in Swarm and Computational Intelligence. Berlin: Springer, 2015, 9140: 317 -325.

[7] 董小刚,邓长寿,袁斯昊,等. MapReduce 模型下的分布式差分进化算法[J]. 小型微型计算机系统,2016,37(12):2695 -2701. (DONG X G, DENG C S, YUAN S H, et al. Distributed differential evolution algorithm based on MapReduce model[J]. Journal of Chinese Computer Systems, 2016, 37(12):2695 -2701.)

[8] SUN C, JIN Y, CHENG R, et al. Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems[J]. IEEE Transactions on Evolutionary Computation, 2017, 21(4):644 -660.

[9] 刘剑英. 基于 GPU 的并行协同差分进化算法研究[J]. 计算机工程与应用,2012,48(7):48 -50. (LIU J Y. Research of parallel cooperation differential evolution algorithm based on GPU[J]. Computer Engineering and Applications, 2012, 48(7): 48 -50.)

[10] 张大斌,周志刚,叶佳,等. 基于随机扩散搜索的协同差分进化算法[J]. 计算机工程,2014,40(7):183 -188. (ZHANG D B, ZHOU Z G, YE J, et al. Cooperation differential evolution algorithm based on stochastic diffusion search[J]. Computer Engineering, 2014, 40(7):183 -188.)

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

[12] TRUNFIO G A, TOPA P, WAS J. A new algorithm for adapting the configuration of subcomponents in large-scale optimization with cooperative coevolution[J]. Information Sciences, 2016, 372: 773 -795.

[13] YANG P, TANG K, YAO X. Turning high-dimensional optimization into computationally expensive optimization[J]. IEEE Transactions on Evolutionary Computation, 2017, 21(9):1 -14.

[14] CHEN W, WEISE T, YANG Z, et al. Large-scale global optimization using cooperative coevolution with variable interaction learning[C]// Proceedings of the 11th International Conference on Parallel Problem Solving from Nature. Berlin: Springer-Verlag, 2010: 300 -309.

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

[16] PENG H, WU Z, DENG C. Enhancing differential evolution with commensal learning and uniform local search[J]. Chinese Journal of Electronics, 2017, 26(4): 725 -733.

[17] WANG Y, FANG K. A note on uniform distribution and experiment design[J]. Science Bulletin, 1981, 26(6):485 -489.

[18] FANG K, MA C, WINKER P, et al. Uniform design: theory and application[J]. Technometrics, 2000, 42(3):237 -248.

[19] PENG L, WANG Y. Differential evolution using uniform-quasi-opposition for initializing the population[J]. Information Technology Journal, 2010, 9(8): 1629 -1634.

[20] TANG K, LI X, SUGANTHAN P N, et al. Benchmark functions for the CEC' 2010 special session and competition on large-scale global optimization[EB/OL]. [2016-11-20]. http://goanna.cs.rmit.edu.au/~xiaodong/publications/lsgo-cec10.pdf.

This work is partially supported by the National Natural Science Foundation of China (61364025), the Science and Technology Project of Jiangxi Provincial Education Department (GJJ161072, GJJ161076).

DONG Xiaogang, born in 1979, M. S., lecturer. His research interests include intelligent computing.

DENG Changshou, born in 1972, Ph. D., professor. His research interests include intelligent computing, big data.

TAN Yucheng, born in 1964, M. S., associate professor. His research interests include applied mathematics, intelligent computing.

PENG Hu, born in 1981, Ph. D., associate professor. His research interests include intelligent computing, big data.

WU Zhijian, born in 1963, Ph. D., professor. His research interests include intelligent computing, parallel computing, intelligent information processing.

(上接第 3218 页)

[32] ZHANG Y, FANG S, ZHOU B, et al. Fingerprint match based on key minutiae and optimal statistical registration[C]// Proceedings of the 9th Chinese Conference on Biometric Recognition. Berlin: Springer, 2014: 208 -215.

This work is partially supported by the Natural Science Foundation of Zhejiang Province (Y1101304).

ZHANG Yongliang, born in 1977, Ph. D., associate professor. His research interests include biometric recognition, pattern recognition, artificial intelligence, machine learning.

ZHOU Bing, born in 1991, M. S. candidate. His research interests include biometric recognition, machine learning.

ZHAN Xiaosi, born in 1975, Ph. D., professor. His research interests include image processing, pattern recognition, biometric recognition, machine learning.

QIU Xiaoguang, born in 1976, M. S., senior engineer. His research interests include biometric recognition, smart fingerprint lock.

LU Tianpei, born in 1996. His research interests include biometric recognition, pattern recognition.