

文章编号:1671-6833(2023)05-0001-09

特约专栏:大规模全局优化算法

【主持人】梁静:国家自然科学基金优秀青年基金获得者

【按语】大规模全局优化(large scale global optimization, LSGO)问题是指待优化问题的决策变量个数很多,通常有上千个,甚至更多。大规模电路系统的设计问题、飞行器机翼的形状优化问题、多层次神经网络的权值训练问题、社区网络分析与挖掘问题、大规模无线传感器网络覆盖优化问题、城市交通管理与调度问题、车辆路径问题、大规模供水网络优化问题等,均可以抽象成 LSGO 问题进行求解。然而,随着决策变量维数的增加,搜索空间变得更加复杂,局部最优解的数量呈指数级增长,算法容易陷入局部最优,取得全局最优解的概率大大降低。搜索空间的属性也可能会随着决策变量规模的增加而产生变化,给优化求解带来困难。LSGO 问题的目标函数通常需要耗费较大的计算代价来评估,较大地增加了优化算法的运行时间。另外,决策变量之间的耦合关系使得 LSGO 问题的求解难度大大增加。因此,解决 LSGO 问题对于现代理论和应用科学都具有重要意义。LSGO 问题的解决可以帮助人们更好地理解 and 优化复杂的系统,改善工业设计和生产等应用,以及推动社会经济的发展。在论文《自适应两阶段大规模约束多目标进化算法》中,作者提出了一种自适应两阶段的大规模约束多目标进化算法,以解决大规模约束多目标优化问题难以收敛的问题;在论文《基于资源分配和动态分组的合作协同演化算法》中,作者提出了一种基于资源分配和动态分组的合作协同演化算法,更好地解决决策变量完全可分或者完全不可分的问题;在论文《基于深度强化学习的大规模敏捷软件项目调度》中,作者提出一种基于复合调度规则的优先经验回放双重深度 Q 网络算法,解决了所建立的大规模敏捷软件项目调度模型;在论文《基于联合分布适配的单向迁移差分进化算法》中,作者在传统的差分进化算法中引入迁移学习技术,提高了算法的求解质量与速度;在论文《惯性分组和重叠特征选择辅助的昂贵大规模优化算法》中,作者基于合作型协同演化策略,提出了一种惯性分组和重叠特征选择的方法来辅助求解昂贵大规模优化问题。

自适应两阶段大规模约束多目标进化算法

于坤杰, 杨振宇, 乔康加, 梁 静, 岳彩通

(郑州大学 电气与信息工程学院, 河南 郑州 450001)

摘 要:针对求解大规模约束多目标优化问题时遇到的收敛速度慢和可行解难以找到的困难,提出了一种自适应两阶段大规模约束多目标进化算法。首先,算法在第一阶段根据决策变量的性质,自适应地选择部分变量进行优化,且不考虑任何约束使种群快速跨过不可行区域,逼近无约束帕累托前沿。其次,算法在第二阶段考虑全部的约束,利用 ε 约束处理技术对变量进行整体优化;同时,利用存档将进化过程中获得的可行且非支配的解保存并更新,以不断地提高种群的收敛性与多样性。最后,将所提算法与其他 6 种算法在 37 个测试函数上进行实验对比,结果表明:所提算法在 25 个函数上取得了最佳结果,且分别至少在 31 个函数上优于对比算法;所提算法在 90% 以上函数中的可行率都能达到 100%,可以有效地解决大规模约束多目标优化问题。

关键词:大规模约束多目标优化; 算法; 自适应; 存档集; 帕累托前沿; 收敛速度; 测试函数

中图分类号: TP301

文献标志码: A

doi: 10.13705/j.issn.1671-6833.2023.05.006

收稿日期: 2023-02-10; **修订日期:** 2023-04-13

基金项目:国家自然科学基金资助项目(62176238, 61922072); 河南省高等学校青年骨干教师培养计划(2020GGJS006); 河南省高校科技创新人才(23HASTIT023); 河南省优秀青年基金资助项目(222300420088)

作者简介:于坤杰(1990—),男,河南周口人,郑州大学副教授,博士,博士生导师,主要从事进化计算理论及应用研究, E-mail: yunkunjie@zzu.edu.cn。

通信作者:乔康加(1996—),男,河南焦作人,博士研究生,主要从事约束多目标进化优化理论及应用研究, E-mail: qiaokangjia@yeah.net。

引用本文:于坤杰, 杨振宇, 乔康加, 等. 自适应两阶段大规模约束多目标进化算法[J]. 郑州大学学报(工学版), 2023, 44(5): 1-9. (YU K J, YANG Z Y, QIAO K J, et al. Adaptive two-stage large-scale constrained multi-objective evolutionary algorithm [J]. Journal of Zhengzhou University (Engineering Science), 2023, 44(5): 1-9.)

实际工程应用中会涉及许多约束多目标优化问题^[1] (constrained multi-objective optimization problem, CMOP), 这些问题不仅需要优化多个目标, 而且需要满足各种类型的约束。约束在搜索空间中会形成不可行区域, 这给算法的求解带来了巨大的挑战, 因此, 解决 CMOP 比解决无约束的多目标优化问题 (multi-objective optimization problem, MOP) 更困难^[2-3]。

虽然现有的约束多目标进化算法在处理具有小规模决策变量的 CMOP 方面表现出了良好的性能, 但随着决策变量的增加, 其有效性会因“维度灾难”而急剧退化^[4]。大规模约束多目标优化问题 (large-scale constrained multi-objective optimization problem, LCMOP) 也很常见, 例如飞行安全系统的优化问题^[5]、时变比误差估计问题^[6]、深度神经网络参数优化问题等^[7]。然而, 相关学者对 LCMOP 的研究较少。He 等^[8]将固定数量的邻域解与参考向量相关联, 以此来构建子种群。在每个子种群中, 利用竞争粒子群的策略来生成子代。Zhang 等^[9]针对供应链网络优化问题, 提出了一种双种群的进化算法: 第一个种群用于解决原始的复杂约束优化问题, 并且使用提出的线性修复算子来提高种群的可行性; 第二个种群用于解决抽象的无约束问题。Wang 等^[10]提出了一种基于动态种群的约束多目标进化算法, 种群的规模在进化过程中逐渐缩小, 这将有利于在进化的早期阶段对决策空间进行充分探索。

以上算法虽然可以初步解决 LCMOP, 但算法的收敛速度较慢, 在很多 LCMOP 上依旧找不到可行解, 因此, 大规模多目标优化是一个具有挑战性的领域, 主要面临着以下难题。

(1) 随着搜索空间的增加, LCMOP 的可行域会变得更小, 使得种群难以满足约束条件, 从而无法找到这些可行区域。

(2) 变量间的耦合更为严重, 会给算法优化带来更大的困难。

(3) 当 LCMOP 具有许多离散可行域或者不可行区域, 种群易陷入局部最优。这不仅可以降低种群的收敛速度, 也会严重损失种群的多样性。

为了克服以上难题, 本文设计了一种自适应两阶段的约束多目标进化算法 (adaptive two-stage constrained multi-objective evolutionary algorithm, ATC-MEA), 在不同的阶段自适应地优化不同的变量。对于第一阶段, 算法在不考虑任何约束的前提下, 自适应地对部分变量进行优化, 以帮助种群更好

地逼近无约束帕累托前沿; 对于第二阶段, 算法将所有的变量整体进行优化, 采用 ε 约束处理方法引导种群向帕累托前沿收敛。此外, 存档种群作为最终输出种群, 用于保存可行解, 并不断更新以提高其收敛性和多样性。与其他 6 种算法相比, 在 37 个测试函数上的实验结果表明了所提算法的优越性。

1 相关背景及工作

1.1 相关背景

不失一般性, 一个大规模约束多目标优化问题可以描述为

$$\begin{aligned} \min \quad & \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x}))^T; \\ \text{s. t.} \quad & \begin{cases} \mathbf{x} = (x_1, x_2, \dots, x_D)^T \in S^D; \\ g_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, p; \\ h_i(\mathbf{x}) = 0, \quad i = p + 1, p + 2, \dots, n. \end{cases} \end{aligned} \quad (1)$$

式中: $\mathbf{F}(\mathbf{x})$ 为目标函数; M 为目标函数数量; \mathbf{x} 为 D 维决策空间 S^D 中的一个决策向量, 通常当 $D \geq 100$ 时, 被称为大规模优化问题; $g(\mathbf{x})$ 和 p 分别为不等式约束和它的数量; $h(\mathbf{x})$ 和 n 分别为等式约束和它的数量。

为了计算 \mathbf{x} 总的约束违反度, 等式约束一般通过松弛参数 δ 转换为不等式约束:

$$|h_i(\mathbf{x})| - \delta \leq 0, \quad i = p + 1, p + 2, \dots, n. \quad (2)$$

之后, 使用以下公式计算每个约束的违反度:

$$G_i(\mathbf{x}) = \begin{cases} \max(0, g_i(\mathbf{x})), & 1 \leq i \leq p; \\ \max(0, |h_i(\mathbf{x})| - \delta), & p + 1 \leq i \leq n. \end{cases} \quad (3)$$

最后, \mathbf{x} 总的约束违反度由以下公式计算得到:

$$G(\mathbf{x}) = \sum_{i=1}^n G_i(\mathbf{x}). \quad (4)$$

式中: 当 $G(\mathbf{x}) = 0$ 时, \mathbf{x} 为可行解, 否则为不可行解。给定两个可行解 \mathbf{x}_a 与 \mathbf{x}_b , 对任意 $j \in (1, 2, \dots, M)$, $f_j(\mathbf{x}_a) \leq f_j(\mathbf{x}_b)$, 并且存在 $k \in (1, 2, \dots, M)$, 使得 $f_k(\mathbf{x}_a) < f_k(\mathbf{x}_b)$, 则称解 \mathbf{x}_a 帕累托支配解 \mathbf{x}_b 。如果一个解不被任何解所支配, 那么该解被称为帕累托最优解。LCMOP 的目标是获得一组约束帕累托最优解集 (constrained Pareto set, CPS)。在目标空间中, CPS 映射为约束帕累托前沿 (constrained Pareto front, CPF)。此外, 将不考虑约束的帕累托前沿称为无约束帕累托前沿 (unconstrained Pareto front, UPF)。

1.2 相关工作

近年来, 各种新型的约束多目标算法被陆续提出, 主要包括基于双阶段的方法和基于多种群的

方法。

基于双阶段的方法将 CMOP 转化为两阶段优化问题,利用两个进化阶段来平衡目标和约束。Fan 等^[11]提出了一种推拉搜索的框架(push and pull search, PPS),在第一阶段,种群不考虑任何约束条件,仅优化目标,将种群推向无约束帕累托前沿;在第二阶段,种群考虑所有的约束和目标,并采用改进的 ε 约束处理方法将种群拉向约束帕累托前沿。Ma 等^[12]在此基础上提出了一种新的多阶段算法(multi-stage constrained multi-objective optimization, MSCMO)。与 PPS 不同的是,在第二阶段,MSCMO 逐步添加约束,以利用更多不可行解的信息来维持种群的多样性。这类方法在相应进化阶段有不同的侧重点,大大降低了问题的优化难度。其缺点是第一阶段探索到的有希望的个体无法在第二阶段得到充分利用。

基于多种群的方法将 CMOP 转化为协同优化问题,使用多个种群分别搜索可行解与不可行解,以协同解决问题。Tian 等^[13]提出了一种协同进化框架,一个种群用于解决原始的 CMOP,另一个种群解决无约束的辅助问题。两个种群在环境选择过程中通过共享后代种群来实现协同进化。Wang 等^[14]提出了一种多种群的协同差分进化框架,这样每个子种群仅优化一个单独的目标以及所有的约束。同时,存档种群不断保存并更新所得到的约束非支配解,逐步逼近 CPF。此外,进化多任务^[15](evolutionary multi-tasking, EMT)是最近提出的一种新的优化模式,其目的在于同时优化多个不同但相关的任务。通过任务间的知识迁移,以提高算法在每个任务上的求解能力^[16]。受此启发,Qiao 等^[17]通过创建简单的辅助任务,将 CMOP 转化为一个多任务优化问题。在实现的算法例子中,第一个任务是原始的 CMOP,另一个是从原始 CMOP 中提取出的 MOP。同时,定义了两类知识并提出了一种新的知识匹配方法,以提高知识迁移的有效性。这类方法利用多个种群,有效地对约束和目标进行了平衡,然而如何更有效地更新和利用辅助种群仍然需要深入研究。

以上两类新型的约束多目标算法具有不同的特点,并且已经较好地解决了小规模约束多目标优化问题。然而,当解决 LCMOP 时,这些算法往往收敛较慢并且难以找到可行域。为了有效解决 LCMOP,本文提出了一种新的自适应两阶段的大规模约束多目标算法。通过在两个阶段使用不同的决策变量优化方法,提高算法的收敛性。

2 算法描述

本文提出了一种自适应两阶段约束多目标算法来解决 LCMOP,进化阶段如图 1 所示。如图 1(a)所示,在第一阶段,所提算法在不考虑任何约束的前提下专注于搜索两个目标 f_1 与 f_2 。为了充分探索目标空间找到更精确的 UPF,提出了一种决策变量自适应优化的策略,即在满足相应条件下,自适应地优化收敛性与多样性变量;在第二阶段,所有的变量将同时被优化,并且 ε 约束处理技术被用来引导种群向 CPF 收敛,如图 1(b)所示。

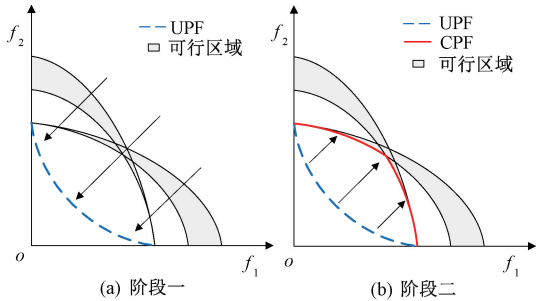


图 1 进化阶段示意图

Figure 1 Schematic diagram of evolution stage

2.1 变量自适应优化策略

在处理小规模优化问题时,之前的大多数进化算法通常将所有决策变量作为一个整体进行优化。然而,由于 LCMOP 变量维度很高,决策空间搜索范围更大,若仍然把所有的变量视为一个整体并进行优化,算法非常容易陷入局部最优并且难以找到可行域。因此,受文献[18]的启发,本文提出了一种自适应的进化策略来对变量进行优化。

Zhang 等^[19]利用聚类分析将决策变量分为两类:多样性变量和收敛性变量,并基于此提出了大规模多目标进化算法(large-scale many-objective evolutionary algorithm, LMEA)来迭代优化两种变量,以平衡收敛性和多样性。此方法变量分类精度高、算法效果好且优化时间短,因此本文采取 LMEA 来对变量进行分类。

该方法首先挑选几个候选解,对每个解中所有变量分别执行数次扰动。将变量值进行归一化后,计算每条拟合线与超平面法线之间的角度。之后根据角度来对决策变量进行聚类,将变量分为两个集:具有较小角度平均值的变量被归为收敛性变量集(CV);其余变量则被归为多样性变量集(DV)。此外,由于大多数变量会归为 CV, CV 中变量较多,因此需再对 CV 进行相关性分析,将其划分为 l 个子组来进一步降低优化难度^[18-19]。设 $\varphi(\mathbf{x})$ 为连续微分

函数,如果 $\exists \mathbf{x}, a_1, a_2, b_1, b_2$ 满足如下条件,则两个决策变量 x_i 与 x_j 相互作用^[19]。

$$\begin{cases} \varphi(\mathbf{x}) \mid_{x_i=a_2, x_j=b_1} < \varphi(\mathbf{x}) \mid_{x_i=a_1, x_j=b_1}; \\ \varphi(\mathbf{x}) \mid_{x_i=a_2, x_j=b_2} > \varphi(\mathbf{x}) \mid_{x_i=a_1, x_j=b_2} \end{cases} \quad (5)$$

需要注意的是 l 的大小由变量之间的交互作用决定。具体来说,每个变量与子组中的至少一个变量相互作用。如果变量之间没有交互作用,则每个变量都是一个子组;而如果所有变量都相互作用,则只有一个子组。所以, l 的取值为 $[1, |CV|]$ 。

将变量分组后, LMEA 交替对收敛性变量和多样性变量进行优化。但在每一次交替中,其都需要对收敛性变量优化 10 轮,这将耗费大量的计算资源。为了进一步节约资源,同时充分对目标空间进行搜索, ATCMEA 在相应的阶段对不同的变量自适应地进行优化。

图 2 给出了变量自适应的优化过程。在第一阶段,首先固定多样性变量,对 $(D-K)$ 个收敛性变量进行优化,满足条件后进行切换,只对 K 个多样性变量进行优化;第二阶段属于进化的中后期阶段,将着力寻找 CPF,而约束与绝大多数的变量都有关,因此将整体优化所有的变量。

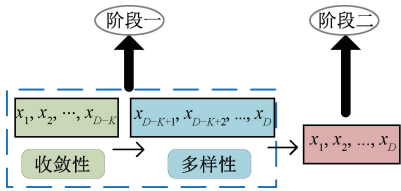


图 2 变量优化过程

Figure 2 Variable optimization process

2.2 ATCMEA 的第一阶段

在第一阶段,所有的约束将被忽略,专注于寻找 UPF。在将决策变量分为收敛性变量与多样性变量后,由于 2 类变量性质不同,对其分别进行优化。

在两目标 MW6 问题上^[20]对其中的两个决策变量分别执行数次扰动后得到的散点图如图 3 所示。其中,红色点为选取的初始解,蓝色点为固定其余变量只对一维变量进行扰动产生的解。此时,图 3(a)中的变量被视为收敛性变量,因为当对该变量扰动时,生成的解基本是被一一支配的,即优化这个变量时,对种群收敛性影响较大;而图 3(b)中的变量被视为多样性变量,对其扰动后生成的解基本是互相非支配的关系,即优化这部分变量时,对种群的多样性影响较大。

因此,在第一阶段,算法先对收敛性变量进行优化,旨在全力探索目标空间,使其快速逼近 UPF。之后固定收敛性变量,为了获得均匀分布的种群,对多

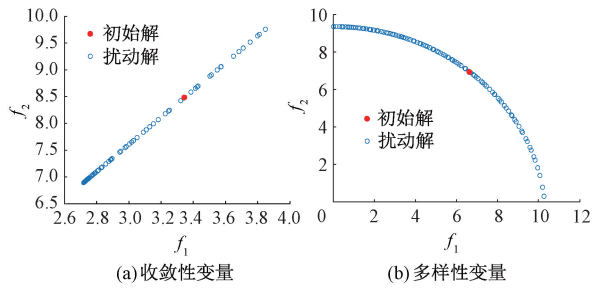


图 3 两种不同类型的变量

Figure 3 Two different types of variables

样性变量单独优化。第一阶段算法流程如下。

- Step 1 初始化种群 P 。
- Step 2 初始化外部存档 R , 令 $R = P$ 。
- Step 3 基于决策变量聚类的方法,将变量划分为收敛性变量集 CV 与多样性变量集 DV 。
- Step 4 对收敛性变量进行相关性分析,将其划分为 l 个子组。
- Step 5 固定 DV ,对 CV 的每一个子组依次执行如下操作:产生子种群 O ,将种群 P 与 O 合并为 S 后更新,输出下一代种群 P 。
- Step 6 所有 CV 子组优化一轮后,得到后代 O^c ,将 O^c 与 R 合并后更新存档。
- Step 7 如果满足相应的切换条件,则跳到 Step 8,否则跳转至 Step 3。
- Step 8 固定 CV ,只优化 DV ,生成子种群 O^D ,将 P 与 O^D 合并为 S 后更新,输出下代种群 P 。
- Step 9 将 O^D 与 R 合并后更新存档。
- Step 10 如果该种群满足切换条件,则第一阶段结束,否则跳至 Step 7。

其中,在 Step 5 的更新中,该阶段是对收敛性变量进行优化,所以当且仅当子代候选解在非支配排序中具有较小的层数或者具有相同的非支配层数,但到理想点的欧氏距离较小时,才认为其比父代候选解更优,即具有更好的收敛质量。

在 Step 8 的更新中,首先对 S 进行非支配排序,之后的环境选择包括以下两步:①选择前 $(z-1)$ 个前沿的候选解,其中 z 是满足 $|S_1 \cup S_2 \cup \dots \cup S_z| > |P|$ 的最小值 (S_1, S_2, \dots, S_z 分别代表第 1, 2, \dots, z 层的个体);②根据 S_z 中候选解在目标空间中的夹角,依次选择剩余的解,直到达到种群规模 $|P|$ 。

在 Step 6 与 Step 9 中,存档作为最终的输出种群,只对可行解进行保存,并利用 SPEA2^[21] (strength Pareto evolutionary algorithm II) 的环境选择来对存档进行更新,从而不断提高其收敛性与多样性。Step 7 与 Step 10 中的切换需要满足如下两个条件。

条件(1):当前种群进化代数满足 T 代;

条件(2): T 代后种群在每个目标上的平均值与上一代的变化值小于给定阈值 α 。

平均值的计算公式为

$$f'_i(\mathbf{x}) = \frac{f_i(\mathbf{x}) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}; \quad (6)$$

$$\bar{f}'_i = \frac{\sum_{\mathbf{x} \in P} f'_i(\mathbf{x})}{N}. \quad (7)$$

首先利用式(6)将目标值归一化,其中 \mathbf{x} 为种群 P 的一个解; $f_i(\mathbf{x})$ 为解 \mathbf{x} 在第 i 个目标上的原始值, $f'_i(\mathbf{x})$ 为解 \mathbf{x} 在第 i 个目标上归一化后的值; f_i^{\max} 与 f_i^{\min} 为种群在第 i 个目标上原始值的最大和最小值。之后利用式(7)计算均值,其中, N 为种群 P 中的个体数; \bar{f}'_i 即为种群 P 在第 i 个目标上的平均值。

条件(2)的计算公式为

$$|(\bar{f}'_i)^T - (\bar{f}'_i)^{T-1}| \leq \alpha. \quad (8)$$

式中: $(\bar{f}'_i)^T$ 与 $(\bar{f}'_i)^{T-1}$ 分别代表当前和上一代的平均值; α 为一个阈值,用于衡量种群的稳定程度。

条件(1)给出了种群最小的进化代数,为了保证种群能够充分进化,在此算法中将 T 设为50;满足条件(2)则说明当前种群波动较小,取值相对稳定,将阈值 α 设为0.01。

虽然优化收敛性变量与多样性变量的终止条件一致,但两个阶段用于判断的目的不同:优化收敛性变量是为了更充分地探索目标空间,使其快速逼近UPF,因此只计算位于非支配排序第一层的个体的均值;而第二阶段对多样性变量进行优化的目的是让种群能够在UPF上更均匀地分布,因此判断时需计算整个种群的均值。

2.3 ATCMEA的第二阶段

在第二阶段,考虑所有的约束对决策变量整体进行优化。为了更好地探索可行区域,采用 ε 约束处理技术将种群从UPF拉回CPF。之所以采用 ε 约束处理技术是因为其可以保留一些有希望的不可行解来探索未检测到的可行区域。其中约束放松参数 ε_i 计算公式^[22]为

$$\varepsilon_i = e_0 \left(1 - \frac{t}{t_{\max}} \right)^{\left(\frac{-\log e_0 - 6}{\log(1-\beta)} \right)}. \quad (9)$$

式中: t_{\max} 为第二阶段的代数; β 为控制 ε_i 下降比例的参数,设为0.5; e_0 为第二阶段初始种群的最大约束违反度值。

第二阶段算法流程如下。

Step 1 将第一阶段终止时的种群和存档设为初始种群 P 与外部存档 R 。

Step 2 根据种群 P 计算初始最大约束违反度 ε_0 ,令 $t = 0$ 。

Step 3 对变量整体优化,生成子代 O ,将 P 与 O 合并为 S 后应用 ε 约束处理技术来对其更新,得到下一代种群 P 。

Step 4 将 O 与 R 合并后更新存档。

Step 5 $t = t + 1$,根据式(9)对 ε 值进行更新。

Step 6 判断是否达到设定的最大评价次数,若达到,则算法终止并输出外部存档 R ,否则跳至Step 3。

其中Step 3中的种群更新方式与外部存档可行解更新方式保持一致。

3 实验设置和结果

为了验证所提算法ATCMEA在LCMOP上的性能,选择CCMO^[13]、PPS^[11]、MSCMO^[11]、DPSEA^[10]、LMOCSO^[23]和POCEA^[8]这6个具有代表性的约束多目标算法在LIRCOP^[24]、DASCOP^[25]和MW^[20]3个测试集上进行对比。

3.1 实验参数设置

种群大小设为100,函数最大评价次数设为10 000× D ,实验中 D 的取值分别为100、150和200。为了公平比较,对比算法中的参数与原始文献中保持一致。所有的实验均是在多目标优化平台PlatEMO^[26]上实现。

3.2 评价指标

选取可行率 FR 和反向世代距离 IGD 作为评价指标。 FR 是算法至少获得一个可行解的运行次数与总运行次数的比率^[27], FR 越大,证明算法找到可行解的次数也越多,其解决大规模约束多目标问题的能力也越强。 IGD 指标衡量的是所求得最佳前沿近似解集到真实Pareto前沿中的点之间最小距离的平均值^[23], IGD 越小,表明测试算法的收敛性和多样性越好。

3.3 实验结果及分析

3.3.1 100维问题上的结果及分析

表1给出了决策变量100维的情况下,所有算法在3个基准问题测试集上的 IGD 与 FR 。在比较时,先比较可行率,可行率一致则比较 IGD 指标,并将最佳结果以灰色底色进行标记。“NaN”表示相应的算法无法在30次运行中每一次都能获得可行解,即 $FR \neq 100\%$,在这种情况下,呈现形式为NaN(FR);当 $FR = 100\%$ 时,呈现形式为 IGD 均值(方差)。使用Wilcoxon秩和检验在显著性水平为0.05下比较两种算法是否有显著性差异。

表 1 7 种算法在 100 维问题上的 IGD 及 FR
Table 1 IGD and FR of seven algorithms on 100 dimensional problems

问题	FR=100%:IGD 均值(IGD 标准差);FR≠100%:NaN(FR)						
	CCMO	PPS	MSCMO	DPSEA	LMOCSO	POCEA	ATCMEA
LIRCMP1	3.14e-1(2.40e-2)+	2.79e-1(1.25e-2)+	2.92e-1(1.92e-2)+	3.02e-1(1.80e-2)+	1.34e-2(1.14e-3)-	4.95e-2(1.83e-2)-	2.07e-1(1.28e-1)
LIRCMP2	2.37e-1(2.69e-2)+	2.36e-1(1.29e-2)+	2.46e-1(1.02e-2)+	2.62e-1(1.74e-2)+	1.52e-2(9.79e-4)-	5.75e-2(2.18e-2)-	9.27e-2(4.01e-2)
LIRCMP3	3.03e-1(2.83e-2)+	2.93e-1(3.29e-2)+	2.81e-1(1.97e-2)+	3.18e-1(2.31e-2)+	1.69e-1(1.22e-1)=	NaN(96.67%)+	1.27e-1(9.88e-2)
LIRCMP4	2.85e-1(3.19e-2)+	2.69e-1(2.97e-2)+	2.63e-1(1.85e-2)+	3.01e-1(1.93e-2)+	2.12e-1(8.42e-2)+	NaN(93.33%)+	1.05e-1(4.31e-2)
LIRCMP5	3.66e-1(3.35e-2)+	1.22e+0(2.79e-3)+	1.13e+0(2.71e-1)+	3.65e-1(2.42e-2)+	1.22e+0(2.53e-3)+	1.20e+0(1.61e-1)+	9.97e-2(2.67e-2)
LIRCMP6	4.22e-1(5.16e-2)+	1.30e+0(2.44e-1)+	1.10e+0(4.19e-1)+	4.12e-1(3.85e-2)+	1.35e+0(1.06e-3)+	1.35e+0(1.40e-3)+	1.03e-1(4.25e-2)
LIRCMP7	1.47e-1(1.82e-2)+	2.52e-1(3.88e-1)+	1.54e-1(1.94e-2)+	1.46e-1(1.25e-2)+	1.69e+0(1.41e-3)+	9.80e-1(7.70e-1)+	8.09e-2(3.42e-2)
LIRCMP8	2.53e-1(2.30e-2)+	5.22e-1(5.90e-1)+	2.56e-1(2.29e-2)+	2.44e-1(1.82e-2)+	1.69e+0(1.37e-3)+	1.41e+0(5.74e-1)+	7.29e-2(5.88e-2)
LIRCMP9	1.17e+0(1.35e-1)+	5.58e-1(1.19e-1)+	1.17e+0(1.52e-1)+	1.05e+0(2.53e-1)+	5.21e-1(9.36e-2)+	9.97e-1(3.10e-1)+	1.01e-1(2.86e-2)
LIRCMP10	2.82e-1(1.27e-2)+	4.11e-1(3.89e-4)+	5.24e-1(2.33e-1)+	4.19e-1(1.78e-1)+	6.38e-1(9.65e-2)+	8.12e-1(2.24e-1)+	2.31e-2(4.80e-3)
LIRCMP11	1.57e-1(7.78e-2)+	4.03e-1(1.56e-1)+	4.21e-1(2.95e-1)+	1.61e-1(1.06e-1)+	5.37e-1(1.22e-1)+	1.03e+0(1.06e-1)+	8.99e-3(4.41e-3)
LIRCMP12	8.58e-1(1.11e-1)+	3.13e-1(8.28e-2)+	8.86e-1(9.34e-2)+	8.54e-1(1.08e-1)+	2.65e-1(2.31e-2)+	7.46e-1(2.70e-1)+	1.69e-2(1.26e-2)
LIRCMP13	9.31e-2(9.70e-4)=	1.31e+0(1.64e-3)+	2.62e-1(2.93e-1)+	9.31e-2(1.20e-3)=	1.30e+0(1.02e-4)+	4.94e-1(5.68e-1)+	9.27e-2(9.82e-4)
LIRCMP14	9.58e-2(9.26e-4)+	1.27e+0(2.00e-3)+	3.11e-1(3.86e-1)+	9.63e-2(7.27e-4)+	1.26e+0(1.69e-4)+	5.25e-1(5.66e-1)+	9.54e-2(1.08e-3)
DASCMOP1	7.81e-1(1.87e-2)+	7.60e-1(3.29e-2)+	7.80e-1(1.51e-2)+	7.87e-1(1.62e-2)+	3.55e-2(6.45e-3)=	4.57e-2(1.65e-2)=	8.31e-2(7.12e-2)
DASCMOP2	2.92e-1(8.74e-3)+	2.93e-1(1.64e-2)+	3.05e-1(1.49e-2)+	2.94e-1(1.21e-2)+	4.26e-2(6.94e-3)+	4.01e-2(7.22e-3)+	2.34e-2(8.92e-3)
DASCMOP3	3.49e-1(3.78e-2)+	4.86e-1(1.97e-1)+	3.13e-1(7.91e-2)+	3.40e-1(1.10e-1)+	4.61e-2(1.40e-2)-	7.04e-2(4.30e-2)-	1.22e-1(5.72e-2)
DASCMOP4	1.14e-3(8.75e-6)-	1.82e-1(7.19e-5)+	1.14e-3(1.01e-5)-	1.14e-3(1.39e-5)-	NaN(6.67%)+	8.44e-2(7.03e-2)+	1.16e-3(2.21e-5)
DASCMOP5	2.68e-3(2.21e-5)-	NaN(53.33%)+	NaN(70%)+	2.69e-3(3.39e-5)-	NaN(13.33%)+	3.14e-2(1.66e-2)+	4.88e-3(1.96e-3)
DASCMOP6	2.23e-2(1.11e-2)=	NaN(80.00%)+	NaN(90%)+	1.65e-2(6.54e-3)-	NaN(6.67%)+	7.55e-2(9.77e-2)+	2.12e-2(5.39e-3)
DASCMOP7	3.08e-2(5.37e-4)-	1.01e-1(1.11e-1)+	3.07e-2(4.32e-4)-	3.16e-2(8.77e-4)-	NaN(20%)+	4.84e-2(5.47e-3)+	3.21e-2(6.77e-4)
DASCMOP8	4.08e-2(8.03e-4)=	6.67e-2(3.88e-3)+	3.99e-2(9.85e-4)-	4.10e-2(7.68e-4)=	NaN(13.33%)+	2.68e-1(1.25e-1)+	4.08e-2(1.00e-3)
DASCMOP9	4.60e-1(7.48e-2)+	5.51e-1(1.09e-1)+	4.25e-1(4.45e-2)+	4.55e-1(5.04e-2)+	5.44e-1(5.23e-2)+	NaN(3.33%)+	7.06e-2(7.95e-3)
MW1	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	1.62e-3(1.03e-5)
MW2	NaN(0%)+	NaN(0%)+	NaN(70%)+	1.80e-1(8.40e-2)+	NaN(0%)+	NaN(0%)+	3.82e-3(4.44e-4)
MW3	7.06e-3(9.58e-4)+	NaN(93.33%)+	3.79e-2(1.66e-1)+	6.22e-3(7.43e-4)+	1.47e-2(2.02e-3)+	1.85e-2(3.58e-3)+	5.06e-3(1.92e-4)
MW4	NaN(0%)+	NaN(93.33%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	4.06e-2(2.99e-4)
MW5	NaN(0%)+	3.36e-1(3.58e-1)-	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(93.33%)
MW6	NaN(93.33%)+	NaN(0%)+	NaN(93.33%)+	6.67e-1(2.48e-1)+	NaN(0%)+	NaN(0%)+	2.76e-3(3.08e-5)
MW7	5.40e-3(5.55e-4)+	1.27e-2(2.30e-3)+	6.91e-3(8.49e-4)+	5.10e-3(5.42e-4)+	7.84e-3(8.97e-4)+	1.56e-2(2.48e-3)+	4.30e-3(3.15e-4)
MW8	NaN(36.67%)+	NaN(0%)+	NaN(40%)+	NaN(73.33%)+	NaN(0%)+	NaN(0%)+	4.29e-2(5.76e-4)
MW9	NaN(0%)+	6.73e-2(1.67e-1)-	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(93.33%)
MW10	4.94e-1(1.56e-1)+	NaN(0%)+	NaN(6.67%)+	NaN(30%)+	NaN(0%)+	NaN(0%)+	3.43e-3(5.10e-5)
MW11	5.91e-3(1.06e-4)+	7.57e-3(4.57e-4)+	5.87e-3(7.43e-5)+	NaN(8.82e-5)+	3.35e-2(6.32e-3)+	4.62e-2(1.04e-2)+	5.56e-3(3.24e-4)
MW12	NaN(0%)+	1.02e-1(2.15e-1)-	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(0%)+	NaN(86.67%)
MW13	7.49e-1(3.32e-1)+	NaN(20.00%)+	8.80e-1(3.89e-1)+	3.74e-1(2.75e-1)+	3.22e+0(7.01e-1)+	2.99e+0(1.02e+0)+	1.18e-2(4.17e-3)
MW14	3.30e-1(1.17e-1)+	7.08e-1(1.30e-1)+	3.14e-1(1.18e-1)+	3.33e-1(1.37e-1)+	1.75e+0(3.61e-1)+	3.80e-1(2.01e-1)+	9.69e-2(1.59e-3)

注:符号“+”、“-”及“=”分别表示 ATCMEA 明显好于、明显差于和相似于对比算法,下同。

从表 1 中可以看出,在 IGD 方面,ATCMEA 在 37 个测试函数中有 25 个表现最好。对于其余算法,大都只在 0~4 个测试函数中取得最佳结果。经 Wilcoxon 秩和检验,对于所有的测试函数,ATCMEA 分别在第 31、34、34、31、32、33 个函数上的性能显著优于 CCMO、PPS、MSCMO、DPSEA、LMOCSO 和 POCEA。此外,ATCMEA 在 90% 以上函数中的 FR 都能达到 100%;相反,其余 6 个对比算法可行率则较差,尤其在 MW 问题集上,存在独立运行 30 次,每

次都未找到可行解的情况。原因在于 MW 问题集具有约束条件难,可行区域狭窄且收敛困难等特征。随着决策变量维度的扩增,MW 问题集变得更加困难,而本文提出的 ATCMEA 能很好地解决大规模 MW 问题。

3.3.2 150 维及 200 维问题上的结果及分析

为了进一步验证算法在解决大规模约束多目标问题上的性能,表 2 给出了 ATCMEA 在 100 维、150 维及 200 维问题上的 Wilcoxon 测试结果。显然,对

于每一种对比算法,在 *IGD* 指标上“+”的数量都远远大于“-”的数量,这充分证明了 ATCMEA 在不同维度测试函数上的显著优越性。

表 2 ATCMEA 与其他算法优劣性比较

Table 2 Comparison of advantages and disadvantages between ATCMEA and other algorithms

	ATCMEA (100 维)			ATCMEA (150 维)			ATCMEA (200 维)		
	+	-	=	+	-	=	+	-	=
CCMO	31	3	3	31	3	3	30	2	5
PPS	34	3	0	33	2	2	34	3	0
MSCMO	34	3	0	32	3	2	33	2	2
DPSEA	31	4	2	31	3	3	31	2	4
LMOCSO	32	3	2	32	4	1	33	4	0
POCEA	33	3	1	33	2	2	34	3	0

与此同时,图 4 给出了算法获得 100% 可行率的问题数量占总测试函数数量的比例。可以看出,提出的 ATCMEA 能在最大比例问题上获得 100% 可行率,说明其可以解决绝大多数的大规模约束多目标测试函数,验证了所提算法不仅可以获得收敛性和多样性均较好的解集,并且具有较好的鲁棒性。

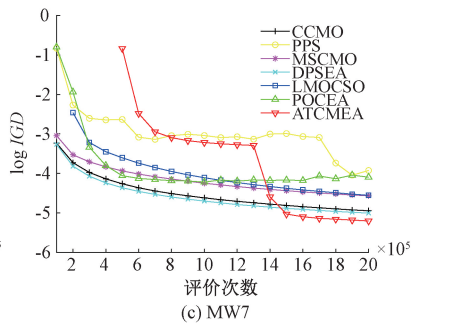
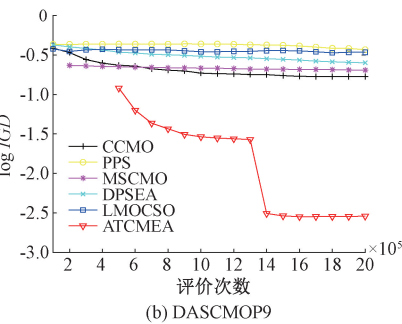
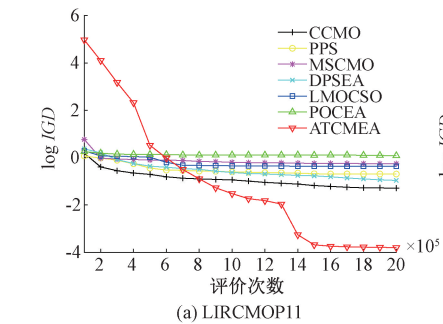


图 5 算法的收敛曲线

Figure 5 Convergence curves of the algorithms

区域的 LIRCMOP 时,这种优势尤为显著。

表 3 给出了 7 种算法在 37 个 100 维问题上的平均运行时间排名,ATCMEA 排名第 2,说明所提算法不仅运行时间较短,而且获得了更为突出的效果。

表 3 算法运行时间排名

Table 3 Running time ranking of algorithms

排名	算法	运行时间/s
1	LMOCSO	80.93
2	ATCMEA	154.19
3	MSCMO	165.53
4	POCEA	247.87
5	PPS	298.24
6	CCMO	307.42
7	DPSEA	842.66

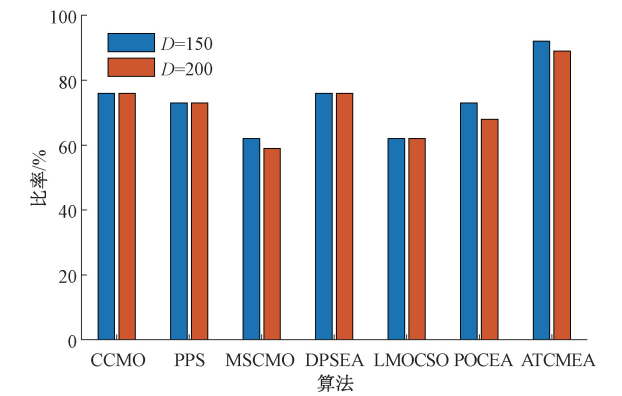


图 4 算法可行率为 100% 问题所占比例

Figure 4 Percentage of problems with 100% feasibility of the algorithms

3.3.3 算法收敛速度及时间

图 5 给出了 ATCMEA 和其他算法在 3 个选定问题(即 LIRCMOP11、DASCMP9 和 MW7)上 200 维时的 *IGD* 的收敛曲线,由于 POCEA 在 DASCMP9 上始终无法获得可行解,因此图 5(b)中没有展示其结果。由图 5 可以看出,与其他算法相比,ATCMEA 实现了更快的收敛,当处理具有复杂可行

4 结论

(1) 针对大规模约束多目标优化问题,提出了一种自适应两阶段的进化算法(ATCMEA)。算法采用两阶段的框架来平衡约束和目标,同时,为了降低高维变量带来的优化难度,结合变量聚类这一技术,对部分变量自适应优化。

(2) 在 3 个不同维度的测试问题上均进行了实验,并给出了算法的收敛曲线及运行时间。实验结果表明,相比于其他先进的约束多目标优化算法,提出的算法具有较好的获得可行解的能力,并且可以获得收敛性及多样性均较好的解集,有效地解决了大规模约束多目标优化问题。

(3) 大规模约束多目标优化问题目前仍是一个

较新且较难的领域,所提出的算法虽然在大部分测试集上取得了良好性能,但在部分 MW 问题上效果不佳,还有很大改进空间。此外随着维度的扩增,约束与目标之间的冲突将更加难以解决。因此,大规模约束多目标优化领域是未来一个重要的研究方向。

参考文献:

- [1] 李二超,李进. 两阶段三存档约束优化算法(TSDA)[J]. 郑州大学学报(工学版), 2018, 39(6): 23-29.
LI E C, LI J. Constraint optimization algorithm with two-stage and three-archive[J]. Journal of Zhengzhou University (Engineering Science), 2018, 39(6): 23-29.
- [2] 华一村,刘奇奇,郝矿荣,等. 非规则 Pareto 前沿面多目标进化优化算法研究综述[J]. 郑州大学学报(工学版), 2021, 42(1): 1-8.
HUA Y C, LIU Q Q, HAO K R, et al. A survey of evolutionary algorithms for multi-objective optimization problems with irregular Pareto fronts[J]. Journal of Zhengzhou University (Engineering Science), 2021, 42(1): 1-8.
- [3] 史非凡,史旭华. 基于参考向量的自适应约束多目标进化算法[J]. 计算机应用, 2022, 42(2): 542-549.
SHI F F, SHI X H. Adaptive reference vector based constrained multi-objective evolutionary algorithm[J]. Journal of Computer Applications, 2022, 42(2): 542-549.
- [4] 梁静,刘睿,瞿博阳,等. 进化算法在大规模优化问题中的应用综述[J]. 郑州大学学报(工学版), 2018, 39(3): 15-21.
LIANG J, LIU R, QU B Y, et al. A survey of evolutionary algorithms for large scale optimization problem[J]. Journal of Zhengzhou University (Engineering Science), 2018, 39(3): 15-21.
- [5] EVERSON R M, FIELDSEND J E. Multiobjective optimization of safety related systems; an application to short-term conflict alert[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(2): 187-198.
- [6] HE C, CHENG R, ZHANG C J, et al. Evolutionary large-scale multiobjective optimization for ratio error estimation of voltage transformers[J]. IEEE Transactions on Evolutionary Computation, 2020, 24(5): 868-881.
- [7] TIAN Y, ZHANG X Y, WANG C, et al. An evolutionary algorithm for large-scale sparse multiobjective optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2020, 24(2): 380-393.
- [8] HE C, CHENG R, TIAN Y, et al. Paired offspring generation for constrained large-scale multiobjective optimization[J]. IEEE Transactions on Evolutionary Computation, 2021, 25(3): 448-462.
- [9] ZHANG X, MA Z B, DING B W, et al. A coevolutionary algorithm based on the auxiliary population for constrained large-scale multi-objective supply chain network[J]. Mathematical Biosciences and Engineering, 2022, 19(1): 271-286.
- [10] WANG B C, SHUI Z Y, FENG Y, et al. Evolutionary algorithm with dynamic population size for constrained multiobjective optimization[J]. Swarm and Evolutionary Computation, 2022, 73: 101104.
- [11] FAN Z, LI W J, CAI X Y, et al. Push and pull search for solving constrained multi-objective optimization problems[J]. Swarm and Evolutionary Computation, 2019, 44: 665-679.
- [12] MA H P, WEI H Y, TIAN Y, et al. A multi-stage evolutionary algorithm for multi-objective optimization with complex constraints[J]. Information Sciences, 2021, 560: 68-91.
- [13] TIAN Y, ZHANG T, XIAO J H, et al. A coevolutionary framework for constrained multiobjective optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2021, 25(1): 102-116.
- [14] WANG J H, LIANG G X, ZHANG J. Cooperative differential evolution framework for constrained multiobjective optimization[J]. IEEE Transactions on Cybernetics, 2019, 49(6): 2060-2072.
- [15] GUPTA A, ONG Y S, FENG L. Multifactorial evolution: toward evolutionary multitasking[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(3): 343-357.
- [16] CHEN K, XUE B, ZHANG M J, et al. An evolutionary multitasking-based feature selection method for high-dimensional classification[J]. IEEE Transactions on Cybernetics, 2022, 52(7): 7172-7186.
- [17] QIAO K J, YU K J, QU B Y, et al. An evolutionary multitasking optimization framework for constrained multiobjective optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2022, 26(2): 263-277.
- [18] MA X L, LIU F, QI Y T, et al. A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables[J]. IEEE Transactions on Evolutionary Computation, 2016, 20(2): 275-298.
- [19] ZHANG X Y, TIAN Y, CHENG R, et al. A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization[J]. IEEE Transactions on Evolutionary Computation, 2018, 22(1): 97-112.
- [20] FAN Z, LI W J, CAI X Y, et al. Difficulty adjustable and scalable constrained multiobjective test problem toolkit[J]. Evolutionary Computation, 2020, 28(3): 339-378.
- [21] ZITZLER E, LAUMANN S, THIELE L. SPEA2: improving the strength Pareto evolutionary algorithm[EB/

OL]. (2001-05-10) [2022-06-13]. <https://doi.org/10.3929/ethz-a-004284029>.

[22] WANG B C, LI H X, LI J P, et al. Composite differential evolution for constrained evolutionary optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019, 49(7): 1482-1495.

[23] TIAN Y, ZHENG X T, ZHANG X Y, et al. Efficient large-scale multiobjective optimization based on a competitive swarm optimizer[J]. IEEE Transactions on Cybernetics, 2020, 50(8): 3696-3708.

[24] FAN Z, LI W J, CAI X Y, et al. An improved epsilon constraint-handling method in MOEA/D for CMOPs with large infeasible regions[J]. Soft Computing, 2019, 23(23): 12491-12510.

[25] MA Z W, WANG Y. Evolutionary constrained multiobjective optimization: test suite construction and performance comparisons[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(6): 972-986.

[26] TIAN Y, CHENG R, ZHANG X Y, et al. PlatEMO: a MATLAB platform for evolutionary multi-objective optimization educational forum[J]. IEEE Computational Intelligence Magazine, 2017, 12(4): 73-87.

[27] YU K J, LIANG J, QU B Y, et al. Dynamic selection preference-assisted constrained multiobjective differential evolution[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2022, 52(5): 2954-2965.

Adaptive Two-stage Large-scale Constrained Multi-objective Evolutionary Algorithm

YU Kunjie, YANG Zhenyu, QIAO Kangjia, LIANG Jing, YUE Caitong

(School of Electrical and Information Engineering, Zhengzhou University, Zhengzhou 450001, China)

Abstract: To address the difficulties of slow convergence and difficulty in finding feasible solutions when solving large-scale constrained multi-objective optimization problems, an adaptive two-stage large-scale constrained multi-objective evolutionary algorithm was proposed. In the first stage, the algorithm adaptively selected some variables for optimization according to the nature of the decision variables, without considering any constraint to make the population quickly cross the infeasible region and approach the unconstrained Pareto front. In the second stage, the algorithm considered all the constraints and optimizes the variables as a whole using the ε constraint-handling technique. At the same time, the feasible and non-dominated solutions obtained in the evolutionary process were saved and updated using archive to continuously improve the convergence and diversity of the population. Finally, the proposed algorithm was experimentally compared with the other six algorithms on 37 test functions, and the results showed that the proposed algorithm could achieved the best results on 25 functions and outperforms the comparison algorithm on at least 31 functions, respectively; meanwhile, the feasibility rate of the proposed algorithm in more than 90% of the functions could reach 100%, which could effectively solve large-scale constrained multi-objective optimization problems.

Keywords: large-scale constrained multi-objective optimization; algorithm; self-adaption; archive set; Pareto front; convergence rate; test function