中国科学: 信息科学 2024 年 第 54 卷 第 12 期: 2811–2827
SCIENTIA SINICA Informationis
论文

深度嵌入适应度评估分配策略的约束多目标进化优化方法

左明成[1], 巩敦卫[2*]

1. 中国矿业大学人工智能研究院, 徐州 221008
2. 青岛科技大学自动化与电子工程学院, 青岛 266061
* 通信作者. E-mail: dwgong@vip.163.com

收稿日期: 2024–01–17; 修回日期: 2024–04–13; 接受日期: 2024–08–27; 网络出版日期: 2024–12–09

国家自然科学基金 (批准号: 62133015, 62303465) 资助项目

摘要 很多实际问题可以归结为约束多目标优化问题. 尽管已有多种求解约束多目标优化问题的方法, 但是, 在全局搜索空间中高效分配适应度评估资源, 实现解方案可行性、收敛性和多样性的平衡仍然是个挑战. 鉴于此, 本文提出深度嵌入适应度评估分配策略的约束多目标进化优化方法, 识别搜索空间中的重点区域, 引导种群高效进化. 该方法首先采用去噪自编码器设计进化种群的降维模型, 获取种群在低维空间的流形; 然后, 在低维空间中聚类种群数据, 获得每类种群约束违反度的方差, 辅助感知适合每一种群个体的低维全局和局部搜索范围; 最后, 基于去噪自编码器获得种群个体的原始空间搜索范围, 准确分配适应度评估资源. 该方法可嵌入已有的进化算法, 能不同程度地提升这些进化算法的性能. 将所提方法应用于 33 个基准测试问题和 15 个矿山综合能源系统运行优化问题, 实验结果表明了所提方法求解约束多目标优化问题的有效性.

关键词 约束多目标优化, 适应度评估分配, 深度嵌入, 进化优化, 矿山综合能源系统

1 引言

约束多目标优化问题 (constrained multi-objective optimization problems, CMOPs), 是指在满足一或多个等式/不等式约束的前提下, 同时优化多个相互冲突性能指标的求解问题. 很多实际问题可以归结为 CMOPs, 如综合能源系统运行优化 [1] 和航天器轨道设计优化 [2]. 不失一般性, 一个 CMOP 可定义为

$$\begin{aligned} \text{Min} \quad & F(x) = (f_1(x), f_2(x), \ldots, f_M(x)), \\ \text{s.t.} \quad & g_i(x) \leqslant 0, i = 1, 2, \ldots, l_1, \end{aligned} \tag{1}$$

引用格式: 左明成, 巩敦卫. 深度嵌入适应度评估分配策略的约束多目标进化优化方法. 中国科学: 信息科学, 2024, 54: 2811–2827, doi: 10.1360/SSI-2024-0025
Zuo M C, Gong D W. Constrained multi-objective evolutionary optimization method with a deep-embedded fitness evaluation allocation strategy (in Chinese). Sci Sin Inform, 2024, 54: 2811–2827, doi: 10.1360/SSI-2024-0025

© 2024《中国科学》杂志社

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

$$h_j(x)=0, j=l+1, l+2, \ldots, l_2,$$
$$x=(x_1, x_2, \ldots, x_D) \in \mathbb{R}^D,$$

其中, x 为 D 维决策变量, $f_k(x)$ 为第 k 个目标函数, $k=1,2,\ldots,M$. $g_i(x) \leqslant 0$ 为第 i 个不等式约束, $i=1,2,\ldots,l_1$. $h_j(x)=0$ 为第 j 个等式约束, $j=l+1,l+2,\ldots,l_2$.

与一般的多目标优化问题 (multi-objective optimization problems, MOPs) 相比, 求解 CMOPs 不仅需要同时优化多个相互冲突的性能指标, 还要求决策变量满足一或多个约束, 因此多目标优化问题的求解非常具有挑战性. 近年来, 多目标进化优化成为求解该问题的主流方法. 采用该方法求解 CMOPs 的关键在于约束处理, 以平衡优化解的可行性、在目标空间的收敛性和多样性. 常用的约束处理方法包括罚函数法、目标与约束分离法、目标增广法、拆分法、混合法和多进化策略法, 这些约束处理技术对于简单的约束多目标优化问题较为有效 [3]. 然而, 对于复杂约束多目标优化问题, 更强的约束条件与更大规模的自变量空间给求解算法准确找到可行解带来了新的挑战. 而实际应用中对求解算法适应度评估次数的限制, 要求进化算法能根据优化模型特点自主合理分配不同求解阶段的计算资源, 进一步提升 CMOPs 的求解质量和效率.

鉴于此, 本文提出深度嵌入适应度评估分配策略的约束多目标进化优化方法. 该方法首先采用去噪自编码器设计进化种群的降维模型, 获取种群在低维空间的流形; 然后, 在低维空间中聚类种群数据, 获得每类种群约束违反度的方差, 辅助感知适合每一种群个体的低维全局和局部搜索范围; 最后, 基于去噪自编码器获得种群个体的原始空间搜索范围, 准确分配适应度评估资源.

本文的主要贡献体现在如下 3 个方面.

• 由低维空间向高维空间迁移进化知识的方式新. 与现有知识引导进化优化的算法相比, 在低维空间挖掘的进化知识特征明显, 获取进化知识的效率高, 能有效降低计算资源消耗.

• 基于低维空间获取不同约束关联关系的方法新. 与现有在原始空间对约束分类的方法相比, 低维空间的变量能关联更多约束, 有利于识别不同约束间的耦合关系.

• 通过低维种群分布感知进化搜索范围的方法新. 与现有通过调整进化参数确定搜索范围的算法相比, 根据低维种群分布所感知到的搜索范围更加精确, 有助于算法的自主求解.

本文余下部分的安排如下: 第 2 节评述相关的研究工作, 并指出已有研究存在的问题; 本文提出的方法在第 3 节阐述, 包括基于去噪自编码器的种群降维、低维种群辅助感知的搜索范围和适应度评估分配嵌入的种群进化生成; 第 4 节是实验结果与分析; 第 5 节是所提方法在矿山综合能源系统运行优化问题中的应用; 最后, 第 6 节总结全文, 并指明后续的研究方向.

2 相关工作

本文研究约束多目标优化问题的进化求解方法, 通过挖掘低维空间的种群分布信息确定种群的搜索范围. 鉴于此, 本节主要讨论已有的约束多目标进化求解方法. 考虑到本文所提方法涉及低维种群信息挖掘, 因此, 还讨论了考虑降维种群的多目标进化优化算法成果.

2.1 约束多目标进化优化

目前的约束处理技术可以概括为转换法和分解法两类. 这两类处理方法的核心思想都是将复杂问题转换为简单问题, 并采用已有经验阶段性求解问题. 转换法具体包括罚函数法和目标增广法; 分解法具体包括目标与约束分离法、拆分法、混合法和多进化策略法.

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

转换法中, 罚函数法是最常用的一种方法, 该方法的思想是: 通过罚函数将约束转换为解性能的评价函数, 进而将约束优化问题转化为无约束优化问题, 以采用无约束优化方法求解转换后的问题. 惩罚因子是影响罚函数法性能的主要因素, 设置合理的惩罚因子对提高算法性能至关重要. 如果可行域是几个非连通区域, 那么, 较大的惩罚因子会让进化搜索过于贪婪, 把种群限制在搜索空间的某一局部区域中, 导致种群早熟收敛; 较小的惩罚因子会让种群的搜索区域过大, 难以快速找到问题的可行解. 鉴于此, 已有工作着重关注了惩罚因子的动态调整问题. Asif 等 [4] 将惩罚因子的取值与进化种群中可行解的比例相关联, 实现种群可行性引导的惩罚因子的自适应调整. 针对进化求解的阶段性需求, Maldonado 等 [5] 采用种群进化阶段引导的惩罚因子动态调整方法. Vaz 等 [6] 也将进化过程划分为三阶段用于惩罚因子的调整. 可以看出, 罚函数法的思想简单, 但需要设置合理的惩罚因子, 而该因子的设置往往依赖于人工经验. 与罚函数法相同的是, 目标增广法也将约束多目标优化问题转换为无约束多目标优化问题, 但该方法将约束视为一或多个优化目标, 在同时优化所有目标函数的过程中降低优化解的约束违反程度. 转换得到的优化目标数量对算法性能有重要影响. Long [7] 和 Zhou 等 [8] 将所有的约束转换为 1 个与约束违反程度相关的优化目标; Vieira 等 [9] 将约束转换为总的约束违反程度和约束违反个数 2 个优化目标. 不难看出, 该方法可灵活处理约束, 转换后的目标个数可根据需要确定. 但是, 目标函数的增多, 一定程度上加大了问题求解的难度.

分解法中, 将目标与约束分离是常见做法, 该方法分别计算优化解的目标函数值和约束违反程度, 在降低约束违反程度的同时, 最小化优化解的目标函数值. 代表性的方法有约束支配法 [10]、ϵ 约束法 [11] 和随机排序法 [12]. 在约束支配法中, 首先根据约束违反度判断两个解的支配关系; 当两个解均为可行解时, 根据优化目标对比解的支撑关系. Deb 等 [10] 在 NSGA-II 中首次采用这种约束处理方法. ϵ 约束法通过松弛约束并逐渐减小 ϵ 的值, 引导种群从不可行域到可行域的搜索. 当 $\epsilon = 0$ 时, ϵ 约束法等同于约束支配法. Jiao 等 [13] 使用 ϵ 约束法不断降低非可行解的约束违反程度, 得到了收敛到约束帕累托 (Pareto) 前沿的优化解集. 容易看出, 目标与约束分离法能很好地平衡优化解的可行性与目标满足程度, 但该方法涉及参数设置, 依赖于人工经验. 拆分法中的种群分解法是将种群划分为多个子种群, 通过不同子种群协同计算, 求解整个优化问题. Wang 等 [14] 提出一种协同进化优化方法, 将整个种群划分为与优化目标数量相等的子种群, 每一子种群求解一个约束单目标优化问题; 拆分法中的进化阶段分解法, 将整个进化过程分解为若干阶段, 不同阶段完成不同的优化任务. Fan 等 [15] 提出融合推进和回拉的搜索框架, 分别引导种群收敛到非约束帕累托前沿和约束帕累托前沿. 进化阶段和种群的划分也可以同时考虑, 结合多进化策略法实现不同粒度的阶段性进化求解. Liu 等 [16] 将整个进化种群分成多个子种群, 不同的子种群采用不同的交叉策略, 使得种群具有更好的全局探索能力. He 等 [17] 对可行解和较好的非可行解分别实施不同的交叉和变异操作, 以实现非可行解到可行解的转换. 可以看出, 拆分法通过分解优化任务, 降低了问题求解的难度, 但是, 如何将问题合理分解, 还依赖于人工经验. 进化算法与传统优化方法融合, 充分利用进化算法和数学规划的优势, 在保持种群收敛和多样性的同时, 关注优化解的可行性. 如 Schütze 等 [18] 基于种群进化得到的解, 利用传统优化得到的邻近解信息, 估计种群后续的进化方向, 引导种群朝可行域搜索. 可以看出, 混合法兼顾了全局和局部搜索, 并平衡了优化性能和约束满足程度. 然而, 进化算法和数学规划的融合时机, 是使用过程中需要解决的问题.

2.2 考虑降维种群的多目标进化优化算法

对于低本征维度 (intrinsic dimensions) 问题, 优化目标或约束只与部分变量有关, 通过挖掘降维种群特征, 可以辅助种群的高效进化. 目前, 采用降维种群特征引导种群进化的技术包括 3 类: 随机

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

嵌入、主成分分析和人工神经网络. 这些方法一般先将父代种群降维到低维空间, 然后在低维空间执行进化算子, 最后在高维空间评价种群优劣性. 不同的是, Qian 等 [19] 提出的随机嵌入技术通过一个随机转换矩阵将原始种群随机降维到一个低维空间, 而 Sun 等 [20] 采用的主成分分析法通过协方差矩阵提高了原始空间和低维空间的转换关系可解释性, 能使获得的低维空间种群更加稳定. 低维种群信息包含种群分布的主要信息, 因此低维种群的进化能够加速收敛. 然而, 如果将所有种群都映射到低维空间进化, 将对种群的多样性造成损害, 导致种群早熟收敛. 因此, Liu 等 [21] 将决策变量划分为多样性相关和收敛性相关两个组, 只将收敛性相关的决策变量降维到低维空间, 引导种群的双空间进化. 从降维效果上讲, 随机嵌入技术和主成分分析法均为线性降维方法, 在降维过程中容易忽略数据细节, 造成重要进化信息的丢失. 人工神经网络是典型的非线性数据降维模型, 能够最大化保留原始空间种群的重要信息. Tian 等 [22] 针对稀疏大规模多目标优化问题, 在每一代中训练去噪自编码器用于种群降维和进化引导. 可以看出, 降维模型、低维种群分析和信息应用方法是提高该类多目标进化优化算法性能的关键.

2.3 讨论

通过以上分析可知, 转换法和分解法分别通过不同的方式引导种群向可行域进化, 具有各自的问题求解特点与性能. 然而, 对于高维度和强约束的多目标优化问题, 不同的约束往往只与部分决策变量相关, 彼此间的关联度不高, 导致上述方法在处理约束时难以同时满足所有约束, 无法引导种群快速进化到可行域, 大大限制了 CMOPs 求解的质量和效率. 在低维空间中分析问题约束能有效提高约束间的关联度, 但已有方法尚未提供在低维空间中挖掘和使用约束相关性的方法. 鉴于此, 本文在第 3 节提出深度嵌入适应度评估分配策略的约束多目标进化优化方法. 该方法能够根据种群的分布预测模型, 识别约束关联度高的重点搜索空间, 为种群个体设计差别化的进化搜索范围, 进一步提升问题求解的性能.

3 本文方法

本节提出深度嵌入适应度评估分配策略的约束多目标进化优化方法, 思想是: 首先, 采用去噪自编码器设计进化种群的降维模型, 获取种群在低维空间的流形; 然后, 在低维空间中聚类种群数据, 获得每类种群约束违反度的方差分布, 辅助感知适合每一种群个体的低维全局和局部搜索范围; 最后, 基于去噪自编码器获得种群个体的原始空间搜索范围, 准确分配适应度评估资源. 容易看出, 本文方法通过深度挖掘低维空间的种群进化信息, 感知问题求解的关键决策变量, 平衡不同决策空间区域的计算资源分配. 该方法可灵活嵌入已有进化算法, 且不改变被嵌入算法的原有结构.

所提算法的整体框架如算法 1 所示. 该算法嵌入适应度评估分配策略只需要在已有进化算法的交叉、变异后增加深度嵌入优化模块 (deep embedding optimization, DEO) (第 6 行). 对于第 t 代种群 P_t, 交叉和变异后的种群分别记为 R_t 和 M_t (第 4 和 5 行), 执行 DEO 后的种群记为 Q_t. DEO 的执行流程如算法 2 所示, 共包含三部分: (1) 基于去噪自编码器的种群降维 (第 1~8 行); (2) 低维种群辅助感知的搜索范围 (第 9 行); (3) 适应度评估分配嵌入的种群进化生成 (第 10 行). 下面将详细介绍每一部分内容.

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

算法 1 所提算法的整体框架

输入: 进化算法参数: 最大进化代数 $T_{\max}$, 种群规模 NP; 去噪自编码器参数: 训练次数 g_1, 学习速率 lr, 训练数据噪声掺杂率 ϕ, 训练目标权值 α, 参数更新代数 T_{upd}; 聚类算法参数: 迭代次数 g_2; 策略嵌入率 γ.

$t \Leftarrow 1$;
初始化 P_t, 包含 NP 个种群个体;
while $t \leqslant T_{\max}$ **do**
  $R_t \Leftarrow \text{Recombination}(P_t)$;
  $M_t \Leftarrow \text{Mutation}(R_t)$;
  $Q_t \Leftarrow \text{DEO}(t, P_t, P_{t-1}, M_t, \gamma, g_1, g_2, \alpha, \text{lr}, \phi, T_{\text{upd}})$;
  $U_t \Leftarrow M_t \bigcup P_t$;
  根据进化算法规则从 U_t 中生成子代种群 P_{t+1};
  $t \Leftarrow t + 1$;
end while

输出: $P_{T_{\max}}$.

算法 2 DEO

输入: t, P_t, P_{t-1}, M_t, g_1, g_2, lr, γ, α, ϕ.

if $t = 1$ **then**
  根据 P_t, P_{t-1}, ϕ 和式 (3) 生成训练数据;
  根据 g_1, lr, α 和第 3.1 小节方法训练 DAE;
end if
if $t \mod T_{\text{upd}} = 0$ **then**
  根据 P_t, P_{t-1}, ϕ 和式 (3) 生成更新数据;
  根据 g_1, lr, α 和第 3.1 小节方法更新 DAE;
end if
根据 P_t, P_{t-1}, g_2 和第 3.2 小节方法计算 r_t;
根据 M_t, r_t, γ 和第 3.3 小节方法生成 Q_t;

输出: Q_t.

3.1 基于去噪自编码器的种群降维

3.1.1 降维维度的确定

对于约束多目标优化问题, 每一约束往往只跟部分决策变量相关, 导致不同约束之间的关联度低, 进化算法难以生成同时满足多个约束的可行解, 对进化搜索造成极大挑战. 然而, 低维空间的种群表示能显著提高约束之间的关联度, 有助于识别决策变量间的耦合关系. 自编码器 (autoencoder, AE) 是基于人工神经网络设计的非线性数据降维工具, 通过编码器和解码器两部分实现原始数据的降维和重构. 为了学习数据的稳健表示, 去噪自编码器 (denoising autoencoder, DAE) 通过在输入数据中添加噪声, 并尝试从噪声数据中重构出干净的原始数据, 提高了自编码器的特征提取能力. 因此本文采用去噪自编码器降维种群数据, 获取种群分布流形, 感知不同搜索空间的种群约束满足情况; 并根据肘点法计算特征值下降速度的拐点, 确定低维空间的维度. 肘点法确定低维空间维度的计算过程为: 首先, 使用主成分分析法获取种群数据的主成分特征向量和特征值; 然后, 对特征值按照由大到小排序, 计算特征值的下降速度; 最后, 根据特征值下降速度确定低维维度. 具体地, 对于种群 P_t, 成分分析所得的特征向量记为 $E^{\text{T}} = \{e_1, e_2, \ldots, e_d, \ldots, e_D\}$, 其中 e_d 表示第 d 主成分分量. 在每一特征向量上, 种群的分布方差为特征值, 记为 $C^{\text{T}} = \{c_1, c_2, \ldots, c_d, \ldots, c_D\}$, 其中 c_d 表示第 d 主成分分量的特征值. 从 c_2 到 c_{D-1}, 特征值的下降速度记为 $V^{\text{T}} = \{v_2, v_3, \ldots, v_d, \ldots, v_{D-1}\}$, 其中 $v_d = \frac{c_d - c_{d+1}}{c_{d-1} - c_d}$. 主成分分析法

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

通常选择特征值累积贡献率为 $q\%$ 的特征向量刻画低维数据. 假设在 C^{T} 中, $c_1, c_2, \ldots, c_{d_1-1}$ 的累积贡献率低于 $q\%$, 而 $c_1, c_2, \ldots, c_{d_1}$ 的累积贡献率高于 $q\%$, 那么降维的维度为

$$z = \arg\min_{d\in[d_1, D-1]} v_d. \tag{2}$$

可以看出, 该方法通过参数 $q\%$ 控制算法的计算效率和求解性能. $q\%$ 过大, 低维种群对信息的保留程度过高, 对数据分布流程的学习能力也较差; 而 $q\%$ 过小, 低维种群对信息的保留程度过低, 对数据分布流程的学习会存在较大误差. 因此在算法应用中, 可根据问题特点设置合适的 $q\%$ 参数. 当优化问题的非线性程度高, 搜索空间存在大量局部解时, 可设置较大的 $q\%$ 参数; 相反, 可设置较小的 $q\%$ 参数. 本文建议的 $q\%$ 取值范围为 $q\% \in [0.9, 0.975]$.

所提方法能较好处理降维带来的信息损失问题. 具体而言, 该方法只从低维空间学习分布流形, 将学习的信息嵌入到高维空间, 用于指导种群进化, 避免了在低维空间进化的局限性, 能够在全局搜索中兼顾学习的流形和随机搜索. 此外, 学习的流形只用于确定潜在搜索范围, 而该范围由一个搜索半径确定, 是一个相对宽泛的区域划定方法. 在这种模糊控制方法下, 使所提方法具有较高的容错性, 在利用流形信息的同时, 仍能以一定概率搜索到因种群信息丢失而忽略的搜索区域, 使算法具有全局搜索能力, 较好解决种群降维带来的信息丢失.

3.1.2 训练数据生成和 DAE 结构确定

本文采用 P_{t-1} 和 P_t 构造带噪声的输入数据 IP_t, 构造方法为

$$\mathrm{IP}_{j,i,t} = \begin{cases} P_{j,i,t-1}, & \text{if} \quad \mathrm{rand}(0,1) < \phi, \\ P_{j,i,t}, & \text{otherwise}, \end{cases} \tag{3}$$

其中, $P_{j,i,t}$ 表示种群 P_t 中第 i 个个体的第 j 个元素, ϕ 为噪声掺杂率, 控制噪声的掺入程度, rand(0, 1) 为 0∼1 之间的随机数. 增大 ϕ 会提高 IP_t 中的噪声比例, 难以准确预测数据的分布流形; 减小 ϕ 会降低 IP_t 中的噪声比例, 使去噪自编码器降解为普通自编码器, 同样难以准确预测数据的分布流形. 因此, 设定一个适中的 ϕ 值能更好地保持去噪自编码器的预测性能. 本文中, $\phi = 0.5$. 此外, 对于高维度问题, 当训练样本的数量低于问题维度时, 将导致自编码器训练的病态. 本文采用数据增广方法产生训练样本 $\mathrm{IP}'_{j,i,t}$, 具体方法为

$$\mathrm{IP}'_{j,i,t} = \mathrm{IP}_{j,i,t} \times \alpha, \tag{4}$$

其中, α 为服从正态分布的放缩系数, 即 $\alpha = \mathcal{N}(\beta, \sigma^2)$, $\beta = 1$, $\sigma = 0.01$. 当训练样本的数量低于问题维度时, 采用式 (4) 增广训练样本, 直到样本数量满足训练要求.

为了降低训练过程的时间消耗, 本文设置轻量化的全连接神经网络模型, 只包含 2 个隐含层, 隐含层之间使用 tanh 激活函数连接, 隐含层与输出层采用线性函数连接. 使用 tanh 激活函数可以加快收敛, 降低训练迭代次数. 2 个隐含层的神经元数量分别为 $n_1 = \lceil z + \frac{D-z}{3} \rceil$ 和 $n_2 = \lceil z + \frac{2}{3} \times (D-z) \rceil$, 可以看出, 隐含层节点数量与 DAE 的输出相关, 可以随着进化过程降低 DAE 的复杂度. 训练 DAE 的参数 θ, 即权重 w 和偏置项 b, 需要定义损失函数作为优化目标. 损失函数中需首先考虑重构损失 $\mathcal{L}_1 = \sum_{i=1}^{\mathrm{NP}} (\mathrm{OP}_{i,t} - P_{i,t})^2$, 通过使输出数据 OP_t 不断逼近 P_t, 从掺杂 P_{t-1} 信息的 IP_t 中提取 P_t 的分布特征, 实现根据父代种群分布预测子代种群分布的能力. 此外, 考虑正则化项 $\mathcal{L}_2 = \sum_{i=1}^{n} \theta_i^2$ 限制参数的大小和稀疏, 帮助 DAE 学习到更加简洁和稳健的特征表示, 其中, n 为神经网络节点数量. 综上, DAE 的损失函数表示为 $L(\mathrm{OP}_t, P_t, \theta) = \mathcal{L}_1 + \alpha\mathcal{L}_2$, α 为权重系数.

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

3.1.3 DAE 的训练和更新

在训练过程中, 设置最大训练次数 g_1、隐含层及其神经元数量和网络学习速率 lr, 第 l 个隐含层表示为 $h^{(l)}$, 其激活函数记为 f_l. 对于训练样本, 通过前向传播 $z^{(l)} = w^{(l)}c^{(l-1)} + b^{(l)}$ 获得训练样本的预测值, 计算预测误差 $L(\mathrm{IP}_t, \theta)$, 并沿预测误差负梯度方向调整网络的权重 w 和偏置项 b, 其中, $c^{(l-1)} = f_l(z^{(l-1)})$ 表示第 $l-1$ 层向第 l 层的输入, $c^{(0)} = \mathrm{IP}_{t-1}$, $w^{(l)}$ 和 $b^{(l)}$ 表示第 l 层节点的权重和偏置值. 当 $L(\mathrm{OP}_t, P_t, \theta)$ 满足要求或训练次数达到 g_1 时, 停止训练 DAE. 此时, 使用编码器可获取种群在低维空间的表示 $H_t = f(w_{1,t}\mathrm{IP}_t + b_{1,t})$, 同时 H_t 可以通过解码器 $\mathrm{OP}_t = g(w_{2,t}H_t + b_{2,t})$ 重构 P_t, 其中, w_1 与 b_1, w_2 与 b_2 分别为编码器和解码器的网络参数.

由于 DAE 的输出维度在动态变化, 因此需要每隔 T_{upd} 代更新一次 DAE. 鉴于神经网络的传统更新方法无法迁移异构 DAE 之间的参数, 本文采用上述方法重新训练 DAE, 更新 w_1, b_1, w_2 和 b_2.

3.2 低维种群辅助感知的搜索范围

本文采用 K-means 算法聚类低维种群数据, 获得种群的差异性分布, 并使用 Calinski Harabasz 方法自主估计聚类 H_t 的最佳 k 值, 更新聚类中心的迭代次数为 g_2. 聚类得到的第 e 个类记为 $\delta_{e,t}$, 类内所有种群的中心点为类中心 $\tau_{e,t}$. 为了分析所有约束在低维空间的耦合性, 计算类 $\delta_{e,t}$ 中种群的约束违反度方差 $\eta_{e,t}$. 假设第 $e1$ 个类的 $\eta_{e1,t}$ 值最高, 那么该类所在区域 $\Omega_{e1,t}$ 具有最强的约束耦合度, 是求解约束多目标优化问题的重点搜索区域. 因此, 当算法倾向于全局搜索时, 需要保证每个种群个体均能搜索到 $\Omega_{e1,t}$, 此时低维种群个体 $H_{i,t}$ 的最大进化搜索半径为 $r = ||H_{i,t} - \tau_{e1,t}||$, $\tau_{e1,t}$ 为第 $e1$ 个类的中心; 当算法倾向于局部搜索时, 限制每个种群个体只搜索其所在类的区域, 此时低维种群个体 $H_{i,t}$ 的最大进化搜索半径为 $r = ||H_{i,t} - \tau_{e2,t}||$, 其中 $\tau_{e2,t}$ 为 $H_{i,t}$ 所在类的中心. 本文根据非支配解的占比 μ 自主决定全局搜索和局部搜索的决策模式, 当种群不全为非支配解时, 倾向于全局搜索; 否则, 倾向于局部搜索. 综上, r 的计算方法为

$$r_{i,t} = \begin{cases} ||H_{i,t} - \tau_{e2,t}||, \text{ if } \mu = 1, \\ ||H_{i,t} - \tau_{e1,t}||, \text{ otherwise.} \end{cases} \tag{5}$$

3.3 适应度评估分配嵌入的种群进化生成

考虑到种群在原始空间进化, 需将低维空间学习到的搜索半径转换到高维空间. 为了操作方便, 可将 $H_{i,t}$, $\tau_{e1,t}$ 和 $\tau_{e2,t}$ 映射到原始空间, 记为 $H'_{i,t}$, $\tau'_{e1,t}$ 和 $\tau'_{e2,t}$. 那么, $r_{i,t}$ 可更新为

$$r_{i,t} = \begin{cases} ||H'_{i,t} - \tau'_{e2,t}||, \text{ if } \mu = 1, \\ ||H'_{i,t} - \tau'_{e1,t}||, \text{ otherwise.} \end{cases} \tag{6}$$

采用差分进化范式确定搜索范围内的进化方向, 将 P_t 在执行原算法交叉、变异后生成的子代种群个体记为 M_t, 那么, 执行 OperatorDEO 生成 $Q_{i,t}$ 的方式为

$$Q_{i,t} = M_{i,t} + F \times (M_{r1,t} - M_{r2,t}), \tag{7}$$

其中, $r1 \in [1, N]$, $r2 \in [1, N]$ 且 $r1 \neq r2 \neq i$. F 由 $||M_{r1,t} - M_{r2,t}||$ 和 $r_{i,t}$ 决定, 计算方法为 $F = \frac{r_{i,t}}{||M_{r1,t} - M_{r2,t}||}$.

DEO 通过学习低维空间种群分布信息影响种群的进化生成, 过度嵌入适应度评估分配策略, 会导致种群在原始搜索空间中的进化搜索陷入局部区域, 造成种群早熟收敛. 因此, 为了平衡种群在原始

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

空间中的进化搜索和适应度评估分配策略的深度嵌入, DEO 中通过参数 γ 控制适应度评估分配策略的深度嵌入概率. 那么, DEO 生成 Q_t 的方法为

$$Q_t = \begin{cases} M_{i,t} + F \times (M_{r1,t} - M_{r2,t}), & \text{if } \mathrm{rand}(0,1) < \gamma \wedge t \geqslant 2, \\ Q_t, & \text{otherwise}, \end{cases} \tag{8}$$

其中, $\mathrm{rand}(0,1)$ 为 $0 \sim 1$ 之间的随机数.

3.4 复杂度分析

3.4.1 本文方法的复杂度分析

第 3.1 和 3.2 小节的介绍可知, 所提方法的时间复杂度主要由嵌入率 γ、种群降维、低维种群聚类, 差分进化算子以及去噪自编码器训练与更新决定. 对于去噪自编码器, 假设编码器包含 h 个隐含层, 隐含层的平均节点数为 p, 输入与输出层的节点数分别为 D 和 L. 实施种群降维的时间复杂度与去噪自编码器结构、优化问题的决策空间 D、最大进化代数 $T_{\max}$、种群规模 N 相关; 低维种群聚类的时间复杂度与维度 L、类个数 k 和迭代次数 g_2 相关; 执行差分进化算子的时间复杂度与决策空间 D、最大进化代数 $T_{\max}$、种群规模 N 相关; 去噪自编码器训练或更新的时间复杂度与神经网络结构、训练或更新迭代次数 g_1、训练样本数量 b 和更新频率 T_{upd} 相关. 这四部分的时间复杂度分别为

- 实施种群降维的时间复杂度为 $O((D \times p + p^2 \times (h-1) + L \times p) \times T_{\max} \times N \times \gamma)$;
- 低维种群聚类的时间复杂度为 $O(T_{\max} \times L \times k \times g_2 \times N \times \gamma)$;
- 执行差分进化算子的时间复杂度为 $O(T_{\max} \times n \times D \times \gamma)$;
- 去噪自编码器训练或更新的时间复杂度为 $O(\frac{T_{\max}}{T_{\mathrm{upd}}} \times ((h-1) \times p^2 + D \times p) \times b \times g_1)$.

由此可知, 所提方法的时间复杂度主要由 γ, $T_{\max}$, N, D 和去噪自编码器的参数决定. 对于大规模约束多目标优化问题, 上述时间复杂度主要由种群降维和执行差分进化算子决定; 当优化问题的决策和目标空间维数较少时, 时间复杂度主要由去噪自编码器的训练和更新决定.

3.4.2 与经典方法时间复杂度的对比

POCEA 算法是求解高维度约束多目标优化问题的典型算法, 通过对比所提方法与 POCEA 的时间复杂度, 可评估该方法的时间消耗量级. POCEA 主要包含子代生成和环境选择两部分, 其中子代生成包括解方案关联、子种群构造和子种群选择. 由于 POCEA 和本文方法均包含环境选择, 所以这里只关注子代生成部分. 子代生成中解方案关联、子种群构造和子代种群选择的时间复杂度如下:

- 解方案关联的时间复杂度为 $O(T_{\max} \times V \times N \times D)$, 其中 V 为参考向量的长度;
- 子种群构造的时间复杂度为 $O(T_{\max} \times (K \times N \times D + V \times K \times y))$, 其中 K 为邻域尺寸, y 为约束数量;
- 子代种群选择的时间复杂度为 $O(T_{\max} \times K \times N \times M)$, 其中 M 为优化目标数量.

由此可知, POCEA 的时间复杂度同样主要由 $T_{\max}$, N, D, V 等参数决定. 由于本文 h, γ, g_1 和 g_2 的取值较小, 所以其时间复杂度与 POCEA 的时间复杂度处在同一数量级.

3.5 进一步说明

本文所提深度嵌入适应度评估分配策略的本质是探索适合每个种群个体的搜索范围和搜索方向, 深度嵌入是指采用深度去噪自编码器获取种群分布流形, 并跨越多个维度将低维空间学到的搜索范围应用于原始空间. 此外, 确定搜索范围需学习低维空间中的种群分布流形, 确定搜索方向需采用差分

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

进化范式. 适应度评估分配策略嵌入多目标优化算法的方式是修正原算法的进化算子, 引导种群对不同区域的差别化搜索. 该算法与其他已有理论技术的两点区别如下.

• 与随机嵌入和主成分分析等降维方法相比, DAE 操作简单且能保留非线性信息, 获取自变量间的耦合关系, 有利于分析变量与约束间、不同约束间的作用关系, 准确识别重要的局部搜索空间, 引导种群进化和搜索资源的分配. 此外, 本文降维原始空间种群的目的是, 预测进化算法的进化方向与范围等趋势. 融合连续两代种群构建带噪声的种群数据, 以带噪声和不带噪声的种群数据分别作为 DAE 的输入与输出, 可使 DAE 的模型参数具有估计子代分布流形的能力. 综上, 本文选择 DAE 降维种群数据具有优势.

• 与其他学习型进化优化算法相比, 本文所提策略的嵌入方法操作简单, 数据挖掘的计算代价较低. 此前, Tian 等 [22] 针对大规模多目标优化问题, 提出了以去噪自编码器获取帕累托最优子空间的方法, 并在低维空间执行进化算子. 与本文不同的是, 该方法采用非支配解训练 DAE, 仅获取当前较好解的空间分布特征. 并且, 该方法仅针对大规模多目标优化问题, 不考虑问题约束且大多数帕累托最优解的决策变量为零, 在低维空间应用种群进化是可行的. 但对于约束多目标优化问题, 在低维空间实施种群进化容易导致多样性丢失, 因此本文在原始空间嵌入所提适应度评估分配策略.

4 实验

为了评价所提方法的性能, 本节进行如下 6 组实验. 第 1 组通过对比 4 种性能优越的约束多目标优化算法, 验证本文方法在基准测试问题的有效性; 第 2 组通过对比 13 种性能优越的约束多目标优化进化优化算法, 验证本文方法在基准测试问题的综合性能; 第 3 组通过对比最新发表的 PKAEO 算法, 验证本文方法对比其他深度学习进化优化算法的性能优势; 第 4∼6 组分别研究了 μ, γ 和 q 对算法性能的影响. 实验的运行环境为 Intel(R) Core(TM) i5-9500 CPU @ 3.00 GHz, 32 GB RAM, Windows 10 和 PlatEMO [23].

4.1 实验准备

4.1.1 基准测试问题

为了评价本文方法的性能, 选择如下 3 个基准测试问题: LIR-CMOP [24], MW [25], TREE [26]. 这 3 个测试问题涵盖了目前常见的 4 类约束多目标优化问题 [3], 即非约束帕累托前沿 (unconstrained Pareto front, UPF) 与约束帕累托前沿 (constrained Pareto front, CPF) 完全重合; UPF 的部分区域可行, 且 CPF 为 UPF 的一部分; UPF 的部分区域可行, 且 CPF 与 UPF 部分重合; UPF 位于不可行区域, 且 CPF 与 UPF 完全分离.

这说明, 选择的测试问题具有代表性, 有助于测试本文提出的方法.

4.1.2 对比算法与参数设置

本小节选择求解约束多目标优化问题的最新算法 C-MOEA-MS [27], AGEMOEA-II [28] 和 AR-MOEA [29], 以及典型算法 NSGA-II [10], 作为本文所提方法的嵌入对象, 并将算法变体表示为 "DEO-*". 通过对比有无引导策略嵌入时算法的性能, 验证本文方法的有效性. 选择求解约束多目标优化问题的流行算法 CMOEA\D [30], C-TAEA [31], IDBEA [32], LMOCSO [33], MOCell [34], MOEA\DD [35], MSCMO [36], MSOPS-II [37], MaOEAIT [20], POCEA [17], RVEAa [38], Top [39] 和 GDE3 [40] 作为 DEO-C-MOEA-MS 的对比算法, 验证 DEO 的综合性能. 上述算法的参数取值与原始文

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

献一致. 对于 “DEO-*”, 自编码器隐含层的节点数分别为 $\llcorner\frac{D}{2}\lrcorner$ 和 $\llcorner\frac{D}{4}\lrcorner$, 网络训练的学习率 $\mathrm{lr}=0.001$. 为了设置较高的问题求解难度, LIR-CMOP1∼LIR-CMOP4 和 LIR-CMOP9∼LIR-CMOP14 的决策变量维度为 $D=50$, 种群的规模设置为 $N=150$, 最大函数评价次数设置为 20000; LIR-CMOP5∼LIR-CMOP8 的求解难度较高, 设置决策变量维度为 $D=10$, 种群的规模设置为 $N=200$, 最大函数评价次数设置为 50000; LIR-CMOP1∼LIR-CMOP12 的优化目标个数为 $M=2$, LIR-CMOP13 和 LIR-CMOP14 的优化目标个数为 $M=3$. MW 的决策变量维度为 $D=30$, 种群的规模设置为 $N=100$, 最大函数评价次数设置为 20000, MW4, MW8 和 MW14 的优化目标个数为 $M=3$, 其余问题的优化目标个数为 $M=2$. TREE1 和 TREE2 的决策变量维度为 $D=60$, TREE3∼TREE5 的决策变量维度为 $D=120$, 种群的规模设置为 $N=100$, 最大函数评价次数设置为 10000, 所有问题的优化目标个数为 $M=2$.

4.1.3 性能指标

鉴于超体积 HV [41] 可以综合评价智能优化算法的性能, 本文选择它作为评价指标. 为了评价不同方法对性能指标差异的显著性, 采用 Wilcoxon rank-sum test 进行假设检验, 取显著水平为 0.05, 并采用符号 “+”, “−” 和 “=” 分别表示某个算法显著优于、劣于和无显著差别于对比算法. 此外, 采用显著水平为 0.05 的多问题 Wilcoxon signed-rank test, 评价两个算法性能差异的显著性.

4.2 所提方法的有效性

针对每一问题, 每一算法独立运行 30 次, 得到 HV 指标值. 实验结果见表 S-1, 其中高亮数据为最优值. 由实验结果可知, 对于 HV 指标, 最多有 25 个测试问题明显优于原始算法, 最少也有 20 个测试问题比原始算法明显优越. 表 1 中给出了算法对比的 Wilcoxon signed-rank test 结果, 用于检验成对样本的中位数是否有显著差异. R^+ 和 R^- 是检验统计量的两个部分, R^+ 是由正差值的秩和得到的统计量, 表示正差值的总和的秩和. R^- 是由负差值的秩和得到的统计量, 表示负差值的总和的秩和. 渐近 P 值 (asymptotic P-value) 是指在样本量足够大时, 根据渐近理论计算出的 P 值. P 值越小, 表明两组样本之间的差异越显著, 通常当 P 值小于 0.05 时, 可拒绝原假设, 即认为两组样本之间存在显著差异. 由 “DEO-*” 的 R^+ 值明显高于 R^- 值可知, 本文所提方法能显著提升原有算法的性能.

对于 LIR-CMOP 测试问题, 其可行域非常小, 甚至部分问题只包含一条曲线, 此外, 约束帕累托前沿被非常大的不可行域分割, 形成几个不相交的段或稀疏点. 由表 S-1 可知, DEO 能够提高所有原始算法在 LIR-CMOP 上的综合性能, 但在不同测试问题上的表现稍有不同. 对于 LIR-CMOP1∼LIR-CMOP6, 约束帕累托前沿分布在一条线或几个线段上, 问题求解相对容易, DEO 能显著提升大部分问题的求解性能, 但每种算法上的性能提升效果稍有不同, 如对 NSGA-II, AGEMOEA-II 和 AR-MOEA 的性能提升较为明显, 而不能显著提升 CMOEA-MS 的求解性能; 对于需要穿过不可行域才能定位约束帕累托前沿的 LIR-CMOP5∼LIR-CMOP12, DEO 的优势非常明显, 在 6 个问题上能提高至少 3 个原始算法的求解性能. 这得益于 DEO 对约束敏感搜索区域的识别和重点搜索, 有利于快速找到可行域和发现更多可行域. 对于 LIR-CMOP13 和 LIR-CMOP14, DEO 不能提高原算法的求解性能. 可能原因是问题的优化解较多但空间分布较分散, DEO 虽然能划定重点搜索区域, 但无法为搜索优化解提供足够的引导信息, 导致难以找到部分优化解.

对于 MW 测试问题, 约束具有多种不同的形式, 可行域非常小, 并由非常大的不可行域分割, 约束帕累托前沿包含多个孤立解. 除了 MW2 和 MW10, DEO 在大多数问题上具有有效性, 尤其对于 MW1, MW4, MW5, MW9 和 MW12, DEO 能帮助原始算法找到可行解. 原算法降低在约束违反度上

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

表 1 “DEO-*” 与对比算法的 Wilcoxon signed-rank test 结果

Table 1 Wilcoxon signed-rank test results between “DEO-*” and comparison algorithms

Algorithm	R^+	R^-	Asymptotic P-value
DEO-CMOEA-MS vs. CMOEA-MS	513.0	82.0	0.000183
DEO-NSGA-II vs. NSGA-II	586.5	8.5	0.000001
DEO-AGEMOEA-II vs. AGEMOEA-II	535.5	25.5	0.000005
DEO-AR-MOEA vs. AR-MOEA	549.5	45.5	0.000016

的贪婪性, 导致在部分问题的搜索过程中忽略了可行解所在的局部区域. 而 DEO 会在搜索过程中评估当前种群的约束违反度分布情况, 估计出约束对决策变量较为敏感的区域, 重点搜索容易被原始算法忽略的区域.

对于 TREE 测试问题, 问题规模大且约束数量多, 约束具有拓扑约束、时序约束、阶段约束等多种不同的形式; 非线性约束导致可行域非常小, 决策变量间的依赖性导致优化目标的不可分离性, 加大了求解难度. 容易看出, DEO 能显著提高原始算法的求解质量. 相较其他问题, TREE 测试函数的维度更高, 对求解算法在进化过程中利用种群信息, 引导种群高效进化的能力要求更高. DEO 中的种群降维机制, 可以准确识别高维度问题中的重要决策变量, 学习决策变量和约束间的作用关系, 为种群的进化搜索提供必要的引导信息.

综上所述, DEO 对提高进化算法的性能具有显著的效果. 但是, DEO 在原始空间进化搜索中产生的种群数据具有随机性, 影响了低维空间数据挖掘的稳定性, 造成较大的进化引导误差. 因此, 设计更加科学的原始空间进化采样和低维空间信息挖掘方法, 是提高 DEO 效果的关键因素.

4.3 所提方法的综合性能

4.3.1 对比其他约束多目标优化算法

以 DEO-CMOEA-MS 为例, 针对每一优化问题, 独立运行 13 个对比算法 30 次, 计算 HV 指标的均值和标准差, 如表 S-2 和表 S-3 所示. 可以看出, OPK-CMOEA-MS 最多在 33 个测试问题上明显优于对比算法, 最少也在 19 个测试问题上比对比算法明显优越. 此外, 使用 Friedman test 假设检验对所有算法的 HV 指标排名. 为了体现 DEO 的有效性, 对比算法中还考虑了 DEO-AGEMOEA-II, DEO-NSGA-II, DEO-AR-MOEA, AGEMOEA-II, NSGA-II 和 AR-MOEA, 结果如表 2 所示, 其中更小的排名值代表更好的排名. 从上述实验结果可得到如下结论.

• DEO-CMOEA-MS, DEO-AGEMOEA-II, DEO-NSGA-II 和 DEO-AR-MOEA 在所有算法中具有最好的排名, 且显著提高了 CMOEA-MS, AGEMOEA-II, NSGA-II 和 AR-MOEA 的算法性能, 如 NSGA-II 排名第 15, 而 DEO-NSGA-II 排名第 3. 在所有原始算法中, CMOEA-MS 具有最好的排名, 且相较 AGEMOEA-II, NSGA-II 和 AR-MOEA 具有显著优势. 而 AGEMOEA-II, NSGA-II 和 AR-MOEA 在嵌入 DEO 模块后, 算法性能均优于 CMOEA-MS. 这说明, DEO 是有效的, 且能大幅度提高原始算法的性能表现.

• “DEO-*” 的有效性受嵌入算法的性能影响较大. CMOEA-MS 在所有原始算法中的性能表现最好, DEO-CMOEA-MS 也在所有 “DEO-*” 中具有最佳性能, 即 DEO-CMOEA-MS 将 CMOEA-MS 排名值从 8.59 提高到了 4.75. 这说明, 在实际应用中需根据问题特点, 选择合适的原始算法作为 DEO 嵌入对象是非常必要的.

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

表 2　所有算法在基准测试问题的 Friedman 排名

Table 2　Friedman ranking of all algorithms on the benchmark problems

Algorithm	Ranking	Algorithm	Ranking
DEO-CMOEA-MS	4.75	LMOCSO	10.81
DEO-AGEMOEA-II	5.74	AGEMOEA-II	10.90
DEO-NSGA-II	6.59	AR-MOEA	12.31
DEO-AR-MOEA	8.25	NSGA-II	12.65
CMOEA-MS	8.59	MOCell	13.02
POCEA	8.89	MaOEAIT	13.37
C-TAEA	9.34	GDE3	15.06
MOEA\DD	9.63	IDBEA	16.05
RVEAa	9.91	ToP	16.72
CMOEA\D	10.40	MSOPS-II	17.44
MSCMO	10.59		

4.3.2　对比深度学习进化优化算法

以 PKAEO 架构[42] 为例, 本文采用 PK-CMOEA-MS 对比 DEO-CMOEA-MS 的性能优势. 独立运行 PK-CMOEA-MS 30 次, 计算 HV 指标的均值和标准差, 如表 S-4 所示. 容易看出, DEO-CMOEA-MS 相较 PK-CMOEA-MS 的性能显著, 整体具有 “13+/4−/16=” 的对比结果. 具体而言, DEO-CMOEA-MS 的优势主要集中在复杂优化问题, 如 LIR-CMOP13, LIR-CMOP14, MW13, MW14 和 TREE 等问题上. 从运行机制上, PKAEO 主要侧重于不同操作算子的动态选择, 以提高问题求解的自适应性. 相较于 DEO, PKAEO 无法在求解过程中准确识别重点搜索区域, 只能通过选择操作算子提高不同阶段的求解性能, 仍旧容易丢弃部分优化解, 对求解性能产生负面影响. 这说明, 文中所提方法能较好地利用深度学习提取种群数据特征, 引导种群高效进化, 对最新的深度学习进化优化算法具有显著优势.

4.4　参数的敏感性实验

4.4.1　平衡参数 μ 的性能影响

参数 μ 影响着全局搜索和局部搜索的平衡, 直接影响适应度评估分配策略的嵌入范围. 为了研究 μ 对算法性能的影响, 本文设置 $\mu = 0.2, 0.4, 0.6, 0.8, 1$, 并测试 DEO-CMOEA-MS 在不同 μ 参数取值下的性能表现. 每种版本的 DEO-CMOEA-MS 独立运行 30 次, 并统计 HV 指标的均值和标准差, 如表 S-5 所列. 容易看出, $\mu = 1$ 时, DEO-CMOEA-MS 具有最好的算法性能. 然而, 每种参数设置下的算法性能差距不大, 这说明本文划分全局搜索和局部开采的方法敏感度低且易于控制. 采用较为模糊的控制方法, 提高了全局/局部搜索的切换容错性. 具体而言, 所提全局搜索是以当前种群个体为中心, 以学习到的搜索步长为半径, 划定一个较大的搜索范围, 搜索范围内的搜索方向由差分进化算子决定. 而差分进化算子中随机选择交叉个体和随机生成缩放因子的机制, 能够确保全局搜索仍旧具有局部开采的能力. 因此, 当 $\mu = 1$ 时才将全局搜索切换为局部搜索, 但实际上算法求解过程中一直在考虑局部开采, 只是在 $\mu = 1$ 时, 会将全部的搜索集中在局部空间中.

4.4.2　嵌入率 γ 的性能影响

嵌入率 γ 影响着低维种群的流形学习和适应度评估分配策略的嵌入频率, 对平衡算法中的信息

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

学习和随机搜索具有重要意义. 为了研究嵌入率 γ 对算法性能的影响, 本文设置 $\gamma = 0.3, 0.4, 0.5, 0.6, 0.7$, 测试 DEO-CMOEA-MS 在不同 γ 参数取值下的性能表现, 所有测试问题的维度和算法的设置与前文一致. 每种版本的 DEO-CMOEA-MS 独立运行 30 次, 并统计 HV 指标的均值和标准差, 如表 S-6 所示. 容易看出, 多大和过小的 γ 取值对算法性能均有不利影响. 当 $\gamma = 0.5$ 时, 算法具有最好的性能表现; 当 $\gamma = 0.3$ 时, 对流形信息的嵌入不足, 无法找到 MW4 问题的可行解; 当 $\gamma = 0.7$ 时, 过多嵌入流形信息使种群早熟收敛, 导致算法性能最差. 综上, 设置 $\gamma = 0.5$ 是一个较好的选择.

4.4.3 信息保留比例 $q\%$ 的性能影响

信息保留参数 $q\%$ 影响着降维过程中对原种群的信息保留比例, 对平衡种群多样性和种群收敛具有重要意义. 为了研究信息保留参数 $q\%$ 对算法性能的影响, 本文设置 $q\% = 93\%, 94\%, 95\%, 96\%, 97\%$, 并测试 DEO-CMOEA-MS 在不同 q 参数取值下的性能表现. 每种版本的 DEO-CMOEA-MS 独立运行 30 次, 并统计 HV 指标的均值和标准差, 如表 S-7 所示. 容易看出, $q\% = 96\%$ 时的算法性能最好; $q\% < 96\%$ 时的信息损失较高, 使学习到的流形分布偏差较大; $q\% > 96\%$ 时的信息冗余较大, 削弱了流形分布的学习能力. 综上, 对于所有测试问题, $q\% = 96\%$ 是一个较为合理的参数设置. 然而, 值得注意的是, $q\% = 96\%$ 的优势主要集中在 LIR-CMOP 问题上, 这说明对于其他测试问题, 还需要根据问题特点微调 $q\%$, 更好地平衡流形学习和信息保留.

5 DEO 在 ICMES 中的应用

本节将所提方法应用到矿山综合能源系统的运行优化, 证明其在实际工程问题中的有效性. 以文献 [1] 中提出的矿山综合能源系统运行优化模型为基础, 通过设置不同的冷、热、电负荷和光、风强度数据, 形成多种场景下的系统用能需求, 以降低用能成本和碳排放为优化目标, 设计 15 个强约束多目标优化问题 ICMES1∼ICMES15. 矿山综合能源系统主要包括能源供应、转换、存储和消费等四部分. 电力供应包括外部电网、风能、太阳能和天然气等, 热力供应包括乏风、矿井水、地热等伴生能源和空气热, 且供应的同时消耗电能; 能源转换主要实现冷负荷供应, 包含吸收式制冷机制冷和电制冷机制冷等两种方式; 该系统配备的储电和储热装置实现用能高峰期出力和低谷期储能; 本文研究矿山综合能源系统运行的日前调度问题, ICMES1∼ICMES15 均以 1 个小时为基本调度单元, 每个问题共包含 432 个决策变量. 系统的运行约束包括供能设备功率、设备爬坡率约束、供需功率平衡约束、蓄能设备的容量与功率约束等等式或不等式约束. 因此, 矿山综合能源系统的运行为高维度、强约束多目标优化问题.

与上述实验设置不同的是, CMOEA-MS 无法找到任何 ICMES1∼ ICMES15 问题的可行解, 因此本文同时选择 DEO-CMOEA-MS 和 DEO-NSGA-II 对比 CMOEA\D, C-TAEA, IDBEA, LMOCSO, MOCell, MOEA\DD, MSCMO, MSOPS-II, MaOEAIT, POCEA, RVEAa, Top 和 GDE3. 所有算法均设置函数评价次数为 50000, 种群规模为 100, 其余参数与上述实验中的设置相同. 针对每一问题, 每一算法独立运行 30 次, 得到 HV 指标的均值和标准差, 结果如表 S-8 所示, 其中, CMOEA-MS, C-TAEA, LMOCSO, MOCell, MSCMO, MSOPS-II, MaOEAIT, POCEA, RVEAa, Top 和 GDE3 均不能找到 ICMES 的可行解. 值得注意的是, 虽然 CMOEA-MS 无法找到任何 ICMES1∼ICMES15 问题的可行解, 但是 DEO-CMOEA-MS 能找到所有问题中 13 个问题的可行解, 且在所有算法中具有最好的综合性能表现. 根据 DEO-NSGA-II 与其余各算法的 HV 性能统计结果, 容易看出, DEO-NSGAII 至少在 11 个 ICMES 问题上比对比算法性能优越. 综上, 本文所提方法能显著提高在矿山综合能源系统

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

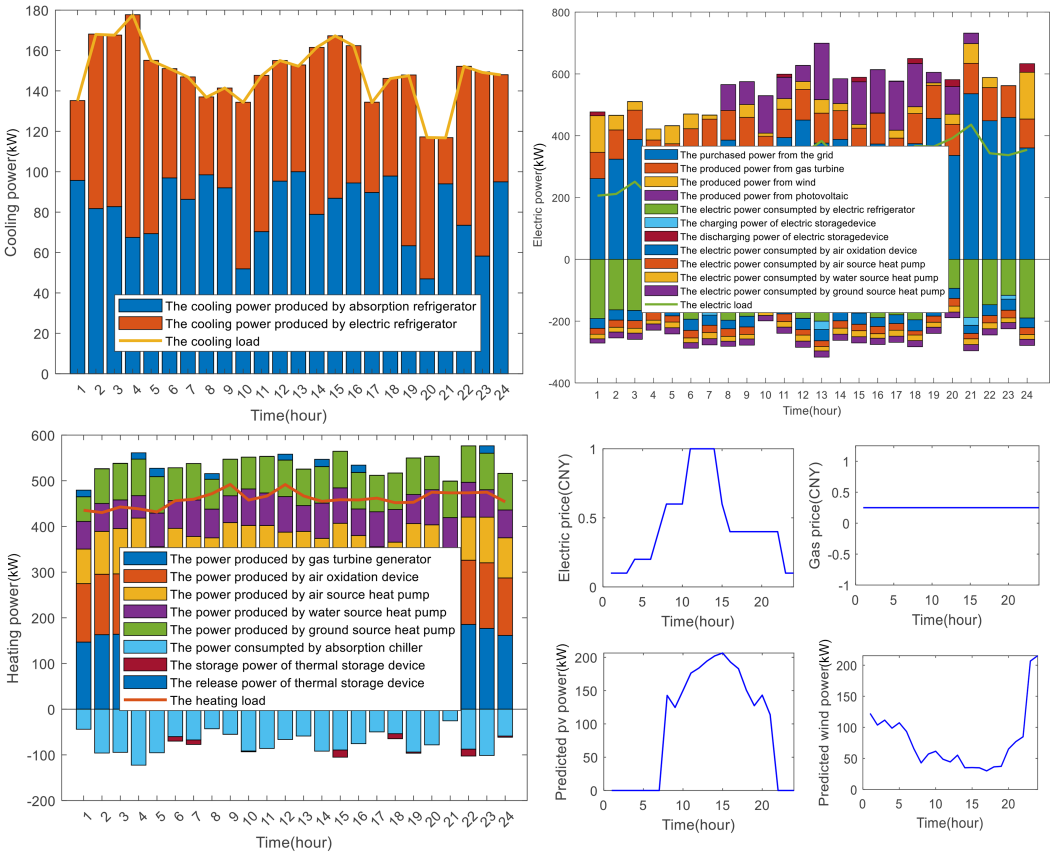


图 1 (网络版彩图) 解方案中对设备出力的调度情况以及优化模型中的关键参数

Figure 1 (Color online) Dispatch of device outputs in the solution and key parameters in the optimization model

运行优化的性能. 图 1 给出了 DEO-NSGA-II 找到的帕累托前沿和指定解的供能与用能情况, 在满足矿山生产用能的前约束下, DEO-NSGA-II 能优化用能成本和碳排放目标.

6 结论

针对约束多目标优化问题, 本文提出了一种深度嵌入适应度评估分配策略的约束多目标进化优化方法. 首先, 将 DEO 嵌入到 4 种性能优越的约束多目标优化算法, 验证其在 33 个基准测试问题的有效性, 实验结果表明: DEO 能不同程度提高所嵌入算法的优化性能. 然后, 在 33 个基准测试问题上对比其他 13 种性能优越的约束多目标优化算法, 验证 DEO 的综合性能, 实验结果表明: DEO 具有最好的综合性能, 尤其在定位小可行域和分散可行域等方面有显著效果. 此外, DEO 在矿山综合能源系统运行优化问题上的应用效果证明了其具有可扩展性, 能够提高所有算法在所有问题上的解方案可行率, 说明了其具有突出的约束多目标优化问题的求解能力. 需要注意的是, DEO 调整搜索范围只依靠低维种群分析, 可能导致部分优化问题的求解性能难以满足需求. 为了进一步提升 DEO 的性能, 我们今后将研究如下问题.

- 针对调整搜索范围只依靠低维种群分析, 导致部分复杂问题难以求解的问题, 拟采用强化学习评估低维种群分析的有效性, 并动态选择是否采纳低维种群的分析结果.

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

• 针对学习到的搜索半径可能存在误差的问题, 拟采用模糊方法设置带区间的搜索半径, 提高搜索范围设置的鲁棒性.

补充材料 表 S-1~S-8. 本文的补充材料见网络版 infocn.scichina.com. 补充材料为作者提供的原始数据, 作者对其学术质量和内容负责.

参考文献

1 Hu H J, Sun X Y, Zeng B, et al. Enhanced evolutionary multi-objective optimization-based dispatch of coal mine integrated energy system with flexible load. Appl Energy, 2022, 307: 118130
2 Zuo M C, Dai G M, Peng L, et al. A case learning-based differential evolution algorithm for global optimization of interplanetary trajectory design. Appl Soft Computing, 2020, 94: 106451
3 Liang J, Ban X X, Yu K J, et al. A survey on evolutionary constrained multiobjective optimization. IEEE Trans Evol Computat, 2022, 27: 201–221
4 Asif J M, Zhang Q F. MOEA/D for constrained multiobjective optimization: some preliminary experimental results. In: Proceedings of IEEE Workshop on Computational Intelligence, 2010. 1–6
5 Maldonado H M, Saúl Z M. A dynamic penalty function within MOEA/D for constrained multi-objective optimization problems. In: Proceedings of IEEE Congress on Evolutionary Computation, Kraków, 2021. 1470–1477
6 Vaz F, Lavinas Y, Aranha C, et al. Exploring constraint handling techniques in real-world problems on MOEA/D with limited budget of evaluations. In: Proceedings of International Conference on Evolutionary Multi-Criterion Optimization, Shenzhen, 2021. 555–566
7 Long Q. A constraint handling technique for constrained multi-objective genetic algorithm. Swarm Evolary Computation, 2014, 15: 66–79
8 Zhou Y L, Zhu M, Wang J H, et al. Tri-goal evolution framework for constrained many-objective optimization. IEEE Trans Syst Man Cybern Syst, 2018, 50: 3086–3099
9 Vieira D A G, Adriano R L S, Vasconcelos J A, et al. Treating constraints as objectives in multiobjective optimization problems using niched Pareto genetic algorithm. IEEE Trans Magn, 2004, 40: 1188–1191
10 Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Computat, 2002, 6: 182–197
11 Takahama T, Sakai S. Constrained optimization by the ε constrained differential evolution with an archive and gradient-based mutation. In: Proceedings of IEEE Congress on Evolutionary Computation, Barcelona, 2010. 1–9
12 Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization. IEEE Trans Evol Computat, 2000, 4: 284–294
13 Jiao R W, Zeng S Y, Li C H, et al. Handling constrained many-objective optimization problems via problem transformation. IEEE Trans Cybern, 2020, 51: 4834–4847
14 Wang J H, Liang G X, Zhang J. Cooperative differential evolution framework for constrained multiobjective optimization. IEEE Trans Cybern, 2018, 49: 2060–2072
15 Fan Z, Li W J, Cai X Y, et al. Push and pull search for solving constrained multi-objective optimization problems. Swarm Evolary Computat, 2019, 44: 665–679
16 Liu Y J, Li X, Hao Q J. A new constrained multi-objective optimization problems algorithm based on group-sorting. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2019. 221–222
17 He C, Cheng R, Tian Y, et al. Paired offspring generation for constrained large-scale multiobjective optimization. IEEE Trans Evol Computat, 2020, 25: 448–462
18 Schütze O, Alvarado S, Segura C, et al. Gradient subspace approximation: a direct search method for memetic computing. Soft Comput, 2017, 21: 6331–6350
19 Qian H, Yu Y. Solving high-dimensional multi-objective optimization problems with low effective dimensions. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2017. 875–881
20 Sun Y N, Xue B, Zhang M J, et al. A new two-stage evolutionary algorithm for many-objective optimization. IEEE Trans Evol Computat, 2019, 23: 748–761

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

21 Liu R C, Ren R, Liu J, et al. A clustering and dimensionality reduction based evolutionary algorithm for large-scale multi-objective problems. Appl Soft Computing, 2020, 89: 106120

22 Tian Y, Lu C, Zhang X Y, et al. Solving large-scale multiobjective optimization problems with sparse optimal solutions via unsupervised neural networks. IEEE Trans Cybern, 2021, 51: 3115–3128

23 Tian Y, Cheng R, Zhang X Y, et al. PlatEMO: a MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput Intell Mag, 2017, 12: 73–87

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. Soft Comput, 2019, 23: 12491–12510

25 Ma Z W, Wang Y. Evolutionary constrained multiobjective optimization: test suite construction and performance comparisons. IEEE Trans Evol Computat, 2019, 23: 972–986

26 He C, Cheng R, Zhang C, et al. Evolutionary large-scale multiobjective optimization for ratio error estimation of voltage transformers. IEEE Trans Evol Computat, 2020, 24: 868–881

27 Tian Y, Zhang Y J, Su Y S, et al. Balancing objective optimization and constraint satisfaction in constrained evolutionary multiobjective optimization. IEEE Trans Cybern, 2021, 52: 9559–9572

28 Panichella A. An improved Pareto front modeling algorithm for large-scale many-objective optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, 2022. 565–573

29 Tian Y, Cheng R, Zhang X Y, et al. An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility. IEEE Trans Evol Computat, 2018, 22: 609–622

30 Jain H, Deb K. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, Part II: handling constraints and extending to an adaptive approach. IEEE Trans Evol Computat, 2013, 18: 602–622

31 Li K, Chen R Z, Fu G T, et al. Two-archive evolutionary algorithm for constrained multiobjective optimization. IEEE Trans Evol Computat, 2018, 23: 303–315

32 Asafuddoula M, Ray T, Sarker R. A decomposition-based evolutionary algorithm for many objective optimization. IEEE Trans Evol Computat, 2014, 19: 445–460

33 Tian Y, Zheng X T, Zhang X Y, et al. Efficient large-scale multiobjective optimization based on a competitive swarm optimizer. IEEE Trans Cybern, 2019, 50: 3696–3708

34 Nebro A J, Durillo J J, Luna F, et al. MOCell: a cellular genetic algorithm for multiobjective optimization. Int J Intell Syst, 2009, 24: 726–746

35 Li K, Deb K, Zhang Q F, et al. An evolutionary many-objective optimization algorithm based on dominance and decomposition. IEEE Trans Evol Computat, 2014, 19: 694–716

36 Ma H P, Wei H Y, Tian Y, et al. A multi-stage evolutionary algorithm for multi-objective optimization with complex constraints. Inf Sci, 2021, 560: 68–91

37 Hughes E J. MSOPS-II: a general-purpose many-objective optimiser. In: Proceedings of IEEE Congress on Evolutionary Computation, 2007. 3944–3951

38 Cheng R, Jin Y C, Olhofer M, et al. A reference vector guided evolutionary algorithm for many-objective optimization. IEEE Trans Evol Computat, 2016, 20: 773–791

39 Liu Z Z, Wang Y. Handling constrained multiobjective optimization problems with constraints in both the decision and objective spaces. IEEE Trans Evol Computat, 2019, 23: 870–884

40 Kukkonen S, Lampinen J. GDE3: the third evolution step of generalized differential evolution. In: Proceedings of IEEE Congress on Evolutionary Computation, 2005. 443–450

41 Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evol Computat, 1999, 3: 257–271

42 Zuo M, Gong D W, Wang Y, et al. Process knowledge-guided autonomous evolutionary optimization for constrained multiobjective problems. IEEE Trans Evol Computat, 2024, 28: 193–207

https://www.sciengine.com/doi/10.1360/SSI-2024-0025

Constrained multi-objective evolutionary optimization method with a deep-embedded fitness evaluation allocation strategy

Mingcheng ZUO[1] & Dunwei GONG[2*]

1. *Artificial Intelligence Research Institute, China University of Mining and Technology, Xuzhou* 221008*, China*;
2. *College of Automation and Electronic Engineering, Qingdao University of Science and Technology, Qingdao* 266061*, China*
* Corresponding author. E-mail: dwgong@vip.163.com

Abstract Many real-world problems can be framed as constrained multi-objective optimization problems. While various existing methods address these issues, efficiently allocating fitness evaluation resources across the global search space while balancing solution feasibility, convergence, and diversity remains a challenge. To tackle this, we propose a novel constrained multi-objective evolutionary optimization method that incorporates a deep-embedded fitness evaluation allocation strategy. It identifies key regions in the search space and guides population evolution efficiently. The method employs a denoising autoencoder to create a dimensionality reduction model for the evolutionary population, revealing its low-dimensional manifold. By clustering the reduced-dimension population and analyzing constraint violation variances within each cluster, the algorithm can sense global and local search ranges for individual population members. The denoising autoencoder then maps these insights back to the original search space, enabling the precise allocation of fitness evaluation resources. This versatile method can be integrated into existing evolutionary algorithms, enhancing their performance to varying degrees. We validated our approach on 33 benchmark test problems and 15 dispatch optimization scenarios of the integrated mine energy system. The results demonstrate the effectiveness of our proposed method in solving constrained multi-objective optimization problems across diverse applications.

Keywords constrained multi-objective optimization, fitness evaluation allocation, deep embedding, evolutionary optimization, integrated mine energy system

https://www.sciengine.com/doi/10.1360/SSI-2024-0025