计 算 机 研 究 与 发 展
Journal of Computer Research and Development
https://crad.ict.ac.cn
62(2): 458−471, 2025

基于优先级先验的演化大规模多目标安全博弈框架

吴宇鹏[1] 钱 鸿[1,2] 王为业[1] 张杨文辉[1] 周爱民[1,2]
[1](华东师范大学计算机科学与技术学院 上海 200062)
[2](华东师范大学上海智能教育研究院 上海 200062)
(51215901031@stu.ecnu.edu.cn;)

A Framework for Evolutionary Large-Scale Multi-Objective Security Games Based on Priority Priors

Wu Yupeng[1], Qian Hong[1,2], Wang Weiye[1], Zhang Yangwenhui[1], and Zhou Aimin[1,2]
[1](*School of Computer Science and Technology*, *East China Normal University*, *Shanghai* 200062)
[2](*Shanghai Institute of AI for Education*, *East China Normal University*, *Shanghai* 200062)

Abstract Multi-objective security games (MOSGs) aim to simultaneously optimize the defender's payoff against multiple heterogeneous attackers, which is of great importance in practical applications. Recently, the space discretization based evolutionary search (SDES) framework has been proposed to transform the constrained high-dimensional step function optimization problem in MOSG into a low-dimensional combinatorial optimization problem and to solve the combinatorial optimization task using a greedy strategy. Although SDES can address large-scale MOSG tasks in time, it is difficult for SDES to converge to the optimal Pareto front, especially in the large-scale scenario. On the one hand, the convergence assumption of the greedy strategy of SDES becomes difficult to be satisfied with the scale of MOSG tasks. On the other hand, SDES uses multiple number of stage components, including spatial discretization, evolutionary optimization, evaluation, and refinement components. This poses a risk of stage coupling, where upstream components' optimization quality directly affects downstream components' performance. To address these issues, we exploit and utilize the priority prior of the protected targets in MOSG task to improve the quality of solutions and simplify the SDES framework, resulting in the SDES-P framework. SDES-P redesigns the evaluation component, which is the core component of SDES, and removes the refinement component. Specifically, SDES-P starts from the infeasible solution with the maximum resources. Then, SDES-P divides the protected targets into two groups based on the priority prior, and the higher-priority group gradually releases resources to find feasible solutions. Finally, SDES-P contains an evolutionary local search strategy combined with priority prior knowledge to enhance the quality of the final Pareto front. We reveal that SDES-P can maintain the advantages of low sample complexity and strong scalability of SDES, and the experimental results show that regardless of whether the convergence assumption is satisfied, SDES-P can find high-quality Pareto fronts with better convergence and diversity compared with SDES.

Key words large-scale evolutionary security games; Stackelberg games; multi-objective evolutionary optimization; priority prior; local search

收稿日期: 2023-06-06; **修回日期**: 2024-01-03

基金项目: 科技创新 2030 重大项目(2018AAA0100902); 上海市自然科学基金项目(21ZR1420300); 国家自然科学基金项目(62106076)
This work was supported by the Scientific and Technological Innovation 2030 Major Projects (2018AAA0100902), the Natural Science Foundation of Shanghai (21ZR1420300), and the National Natural Science Foundation of China (62106076).

通信作者: 钱鸿(hqian@cs.ecnu.edu.cn)

摘　要　多目标安全博弈（multi-objective security game, MOSG）旨在同时最优化防御者应对多个异质攻击者获得的收益，在实际应用中具有重要意义. 近期提出的基于空间离散化的演化搜索（space discretization based evolutionary search, SDES）框架将MOSG中的带约束的高维阶梯函数优化问题转换为低维组合优化问题，并使用贪心策略解决组合优化任务. 虽然SDES能够在有限时间内处理大规模MOSG任务，但是SDES难以收敛到大规模MOSG任务对应的最优Pareto前沿上. 一方面，SDES的贪心策略的收敛性假设随问题规模扩大而变得愈发难以满足；另一方面，SDES过多的阶段组件（空间离散化、演化优化、评估、解的精炼）存在阶段耦合的风险，即上游组件的优化质量直接影响下游组件的表现. 因此，挖掘并利用MOSG任务中被保护对象的优先级（priority）先验知识，旨在提高解的质量并简化SDES框架，从而提出了SDES-P框架. SDES-P重新设计了SDES的核心组件——评估组件，并移除解的精炼组件. 具体而言，SDES-P从具有最大资源的不可行解开始，根据被保护对象优先级先验将被保护对象分成2组，优先级较高的一组对象会逐渐释放资源以找到可行解. 最后，SDES-P包含了一种结合优先级先验的演化局部搜索策略，增强最终Pareto前沿的质量. 分析出SDES-P可保持SDES所具有的样本复杂度低、规模可扩展性强的优势，并且用实验结果表明，无论MOSG任务是否满足收敛假设，SDES-P可以找到相较于SDES收敛性、多样性更优的高质量Pareto前沿.

关键词　大规模演化安全博弈；Stackelberg博弈；多目标演化优化；优先级先验；局部搜索

中图法分类号　TP181

DOI: 10.7544/issn1000-1239.202330463　　**CSTR:** 32373.14.issn1000-1239.202330463

安全博弈[1](security game, SG)是一种特殊的Stackelberg博弈[2], Stackelberg博弈中的追随者和领导者分别对应于安全博弈中的攻击者和防御者. 安全博弈在自然资源保护、交通运输系统、网络安全等领域受到了广泛关注[3-8]. 多目标安全博弈[9-10](multi-objective security game, MOSG)是安全博弈论中的重要研究方向，旨在探索单个防御者如何在同时面对多个异质攻击者的情况下最大化获得的利益. 由于安全领域的特殊性，MOSG要求找到一个收敛性和多样性都良好的Pareto前沿[9]. 目前，MOSG已在洛杉矶警力资源分配(将有限警力资源分配到对洛杉矶不同潜在犯罪类型的防御上)等现实任务[9-10]中得以成功应用，展现了MOSG在安全博弈上的潜力. 然而，受限于算法的问题规模可扩展性，不少现有的MOSG算法难以在更大规模的问题(如自然资源保护、公共基础设施保护等)上得到广泛应用.

MOSG的潜力来自于其对异质攻击者场景进行多目标优化[11-12]建模的思想. MOSG将异质攻击者造成的每个损失视为单独的优化目标，在优化时同时追求每个目标的最优，而不是如传统安全博弈算法一样将异质的损失加权合并成单目标函数进行寻优. 多目标优化异质损失更符合现实任务的需求，例如，同时最优化盗窃造成的财产损失和恐怖袭击造成的人员伤亡这2类异质损失时，传统算法一般将异质损失转化为可加权合并的损失类型，如经济损失. 然而从安全角度，人员伤亡不应该仅用经济损失衡量，这表明以多目标方式同时优化多类异质损失是更合理且必要的. 因此，设计求解规模扩展性强且求解质量高的算法是MOSG的研究重点[9-10, 13]. 其中，具有代表性的一类ϵ-约束算法[9-10]能够解决中小规模(异质攻击者/类型3~5个，防御者几十个)MOSG问题，并通过误差上界限制Pareto近似集与Pareto参考集之间的差距在可控范围内. 另一类最新的基于演化算法的求解框架SDES[13]能高效求解大规模MOSG. 对于要在限定时间内完成的大规模MOSG问题，SDES框架能在规定时间内完成，而ϵ-约束算法往往会超时.

虽然现有的算法[10, 13-14]分别在MOSG任务的求解质量和规模可扩展性方面取得了一定的进展，但很少有算法能够在这2方面都取得令人满意的结果. 在求解质量方面，MOSG需要算法提供具有良好收敛性和多样性的Pareto前沿(PF)集合. 以ORIGAMI-M和ORIGAMI-A为代表的一类ϵ-约束框架[9-10]为Pareto前沿的收敛性提供了理论保证. 然而，为满足该收敛性保证，算法的样本复杂度会随攻击者数量呈指数级增长. 因此，在大规模攻击者的情况下，ϵ-约束框架在有限时间内难以提供具有良好收敛性和多样性的Pareto前沿. 在规模可扩展性方面，一类称为SDES的空间离散化框架[13]可以快速处理大规模MOSG任务. SDES框架包含离散化、优化、评估和解的精炼等4个组件. SDES使用离散化组件将MOSG

任务转化为组合优化问题,优化组件采用贪心策略解决组合优化问题. SDES 使用了评估组件以满足收敛假设的要求,然而,贪心策略收敛到最优 Pareto 前沿的假设条件随问题规模的扩大而愈发难以满足,这使得 SDES 在大规模场景中有时难以找到高质量的 Pareto 前沿. SDES 最后使用解的精炼组件在一定程度上补救收敛性假设不被满足的情形. 此外,SDES 是一个多阶段优化模型,阶段组件过多会引发阶段耦合的风险[15],从而影响算法优化的性能. 一方面,阶段耦合导致下游组件的优化效果依赖上游组件,上游组件的优化效果波动会对下游组件造成明显影响,降低了模型的稳定性. 另一方面,阶段耦合会累积上游组件的优化误差,愈加影响下游组件的效果. 综上所述,对于大规模多目标安全博弈任务,SDES 框架在提升求解质量和降低框架复杂度方面有较大提升空间,亟需强化.

有鉴于此,本文提出了一种基于优先级先验的演化大规模多目标安全博弈框架 SDES-P,该框架可以在平衡求解质量和可扩展性的同时简化 SDES 框架. 在 SDES-P 中,我们通过安全博弈的先验知识重新设计了 SDES 的核心组件——评估组件. 新的评估组件包含完全冷漠初始化策略、基于优先级的可行解搜索策略以及演化局部搜索策略. 具体而言,已有研究表明,SDES[13] 基于贪心策略的评估组件更倾向于选取资源数量更多的候选集. 因此,SDES-P 将解初始化为消耗最多资源但不一定可行的解. 此外,现有研究[16] 发现在安全博弈任务中,不同被保护对象收益的上下界决定了其获取保护资源的优先级. 为释放尽可能多的资源使解可行化,被保护对象可以分为释放资源优先级高和释放资源优先级低 2 组. 由于优先级高的被保护对象收益不可观,不容易被攻击者关注,因此释放其资源时对防御者收益的影响较小. 基于优先级先验,SDES-P 能在保持求解质量的同时将不可行解转化为可行解. 最后,从可行解开始,演化局部搜索策略借助优先级先验筛选非支配解集,构建收敛性和多样性良好的Pareto前沿解集. 相较于 SDES 框架,SDES-P 保留了离散化和优化组件,重新设计了评估组件,移除解的精炼组件,精简了框架的构成部分. 与此同时,SDES-P 可保持 SDES 所具有的样本复杂度低、规模可扩展性强的优势. 此外,各种规模下的 MOSG 对比实验表明,在 MOSG 任务不满足收敛假设时,SDES-P 可以找到相较于 SDES 收敛性、多样性更优的高质量 Pareto 前沿. 消融实验验证了 SDES-P 框架中各组件的必要性与贡献. 一言以蔽之,SDES-P 是一个同时考虑了求解质量和规模可扩展性的 MOSG 简化求解框架.

本文后续内容组织如下:介绍安全博弈相关工作,回顾基础知识,详细介绍本文提出的基于优先级先验的演化大规模多目标安全博弈框架,分析实验结果,最后总结全文. 附录部分是对正文细节的补充说明.

1 相关工作

安全博弈论研究如何在安全领域中高效地利用有限资源来保护目标,防御潜在攻击者. 按照目标的建模方式,安全博弈论可以分为基于加权合并的单目标安全博弈 SG[16] 以及基于多目标建模与优化的多目标安全博弈 MOSG[9]. 本节主要回顾现有安全博弈论算法的优势与不足,同时阐述本文所提出的 SDES-P 与现有算法之间的联系和区别.

在单目标安全博弈中,以贝叶斯安全博弈[16–18]为代表的标量决策方法因建模方式简单而在现实任务中得到广泛应用,包括用于机场保护的 ARMOR, IRIS, GUARDS[3–5],用于运输系统保护的 TRUSTS[6],用于海岸防护的 PROTECT[7] 以及用于网络安全的随机博弈模型[19]. 贝叶斯安全博弈使用贝叶斯方法依据概率将多目标函数值合并成单目标标量,在同类方法中具有明显的效率优势. DOBSS[15] 是第 1 个成功应用于 ARMOR[3] 系统的算法,它利用攻击者的独立性将几个线性规划问题转化为混合整数线性规划问题,并通过有效的编程方法来解决. 此外,ERASER[16] 直接在混合整数线性规划公式中解决 SG 问题,并使目标维度不再影响求解效率. 虽然标量决策方法可以高效地产生最优配置方案,但标量决策方法不适用于包含多个不可合并的异质攻击者的场景. 例如,财产和生命的损失并不具有相同的重要性[9],将异质目标加权合并会由于不合实际从而求解质量差. 本文设计的 SDES-P 框架能够独立建模异质的攻击者,同时最优化多个异质目标.

在多目标安全博弈中,以 ϵ-约束框架[9-10] 为代表的精确求解算法在中小规模(异质攻击者/类型 3~5 个,防御者几十个)MOSG 任务上的求解质量良好. 通过利用问题特性[16]、剪枝[9] 和其他方法[10],该框架能够在不破坏其他目标的前提下,逐一优化每个目标. 在不限制时间的情况下,ϵ-约束框架是一类可以保证Pareto前沿质量的精确算法. 然而,由于它维护Pareto前沿多样性的样本复杂度会随着攻击者数

量的增加呈指数增长，在大规模情况下，该算法难以在规定时间内找到满足要求的Pareto前沿. 本文提出的 SDES-P 求解框架能够在快速求解大规模 MOSG 任务的同时保证求解质量.

以空间离散化框架[13]为代表的近似求解算法在处理大规模 MOSG 问题方面表现良好. 在这种算法中，MOSG 任务被转化为一个组合优化问题. 其中，SDES[13]基于收敛假设，通过贪心决策解决组合优化问题. 当收敛假设成立时，SDES 可以在限定时间内处理大规模 MOSG 任务并保证求解质量. 然而，由于 SDES 会贪心地针对每个攻击目标优化最优解，这使得满足收敛假设的难度会随攻击目标数的增加快速增长. 因此，在保护对象为大规模的场景下，SDES 算法很难提供一个收敛性良好的解. 在 MOSG 任务不满足收敛假设的情况下，SDES 中解的精炼组件会对求解质量进行一定程度上的补救. 然而，上述收敛假设难以满足的问题是由评估组件采用贪心策略导致的，解的精炼组件并不能从根本上解决该问题. 本文提出的 SDES-P 框架尝试从根本上解决大规模场景下假设难以满足的问题，重点改进了评估组件，同时移除解的精炼组件以简化框架.

2 基础知识

本文所提算法主要涉及多目标安全博弈 MOSG 的建模和 MOSG 任务的求解框架 SDES，下面就相关概念和基本知识加以介绍.

2.1 多目标安全博弈

MOSG 是一种特殊的防御者-攻击者 Stackelberg 博弈，由 T 个被保护对象、1 个防御者 D 和 N 个异质的攻击者 A_i 组成，其中 $i\in\{1,2,\cdots,N\}$. 资源总量为 $r\times T$，由于资源受限，资源比例 $r\ll 1$. 防御者 D 的连续策略可以表示为覆盖向量 $\boldsymbol{c}=(c_1,c_2,\cdots,c_T)\in[0,1]^T$，其中 $\|\boldsymbol{c}\|_1\leqslant r\times T$，$c_t$ 表示给第 t 个被保护对象的资源量和成功覆盖保护对象 t 的概率[20]. 覆盖向量的解空间被定义为 $C=\left\{\boldsymbol{c}\in[0,1]^T\mid\|\boldsymbol{c}\|_1\leqslant r\times T\right\}$. 攻击者 A_i 的离散策略可以表示为 $\boldsymbol{a}_i=(a_{i1},a_{i2},\cdots,a_{iT})\in\{0,1\}^T$. 防御者在某个保护对象 t 上成功防御攻击者 A_i 的攻击会获得一定收益，MOSG 的目标即为找到防御者 D 最优的资源分配方案，以获得 D 的最优收益. 由于 MOSG 包含 N 个不可合并的异质攻击者 $A_1,A_2,\cdots,A_N$，因此 MOSG 任务需要在多个攻击者的攻击下尽可能最大化收益，其目标函数形式化为式(1)，其中 $U_i^{\mathrm{d}}(\boldsymbol{c})$ 为防御者 D 成功防御攻击者 A_i 获得的收益.

$$\max_{\boldsymbol{c}\in C}\boldsymbol{F}(\boldsymbol{c})=\left(U_1^{\mathrm{d}}(\boldsymbol{c}),U_2^{\mathrm{d}}(\boldsymbol{c}),\cdots,U_N^{\mathrm{d}}(\boldsymbol{c})\right). \tag{1}$$

在多目标优化中算法可能会找到多个解，但只有满足定义 1 中的帕累托支配(Pareto dominance)的解才满足要求. 定义 2 中的Pareto最优性和Pareto解的概念被广泛用于解的比较和选择.

定义 1. Pareto支配. 给定 $\boldsymbol{F}(\boldsymbol{c})$ 与 $\boldsymbol{F}(\boldsymbol{c}')$，若 $U_i^{\mathrm{d}}(\boldsymbol{c}')\leqslant U_i^{\mathrm{d}}(\boldsymbol{c})$ 对于所有 $i\in\{1,2,\cdots,N\}$ 都成立，则称 $\boldsymbol{c}$ 弱支配 $\boldsymbol{c}'$ 或 $\boldsymbol{F}(\boldsymbol{c})$ 弱支配 $\boldsymbol{F}(\boldsymbol{c}')$，表示为 $\boldsymbol{c}\succcurlyeq\boldsymbol{c}'$. 若 $\boldsymbol{c}\succcurlyeq\boldsymbol{c}'$，并且存在 $j\in\{1,2,\cdots,N\}$ 使得 $U_j^{\mathrm{d}}(\boldsymbol{c}')<U_j^{\mathrm{d}}(\boldsymbol{c})$，则称 $\boldsymbol{c}$ 支配 $\boldsymbol{c}'$ 或 $\boldsymbol{F}(\boldsymbol{c})$ 支配 $\boldsymbol{F}(\boldsymbol{c}')$，表示为 $\boldsymbol{c}\succ\boldsymbol{c}'$.

定义 2. Pareto最优性和Pareto解. 若不存在其他满足 $\boldsymbol{c}'\succ\boldsymbol{c}$ 的 $\boldsymbol{c}'\in C$，则称覆盖向量 $\boldsymbol{c}$ 为Pareto解. 非支配的覆盖向量集合称为Pareto集，相应的收益向量集合在目标空间中的分布称为Pareto前沿.

在双人 Stackelberg 博弈中，U_i 包含 A_i 的综合收益 $U_i^{\mathrm{a}}(\boldsymbol{c},\boldsymbol{a}_i)$ 和 D 的综合收益 $U_i^{\mathrm{d}}(\boldsymbol{c},\boldsymbol{a}_i)$，分别表示对给定的策略 $(\boldsymbol{c},\boldsymbol{a}_i)$，$A_i$ 或 D 在 T 个被保护对象上的收益.

$$U_i^{\mathrm{a}}(\boldsymbol{c},\boldsymbol{a}_i)=\sum_{t=1}^{T}a_{it}U_i^{\mathrm{a}}(c_t), \tag{2}$$

$$U_i^{\mathrm{d}}(\boldsymbol{c},\boldsymbol{a}_i)=\sum_{t=1}^{T}a_{it}U_i^{\mathrm{d}}(c_t), \tag{3}$$

其中，式(2)(3)中的 $U_i^{\mathrm{a}}(c_t)$ 和 $U_i^{\mathrm{d}}(c_t)$ 分别表示 A_i 和 D 在单个保护对象 t 上的收益. $U_i^{\mathrm{a}}(c_t)$ 或 $U_i^{\mathrm{d}}(c_t)$ 是关于覆盖概率 c_t 的线性函数[16]，如式(4)(5)所示. 如果被攻击对象 at_i 被完全覆盖(即 $c_t=1$)，A_i 或 D 的收益记为 $U_i^{\mathrm{c,a}}(t)$ 或 $U_i^{\mathrm{c,d}}(t)$. 如果 at_i 完全未被覆盖(即 $c_t=0$)，A_i 或 D 的收益记为 $U_i^{\mathrm{u,a}}(t)$ 或 $U_i^{\mathrm{u,d}}(t)$. 具体线性关系为

$$U_i^{\mathrm{a}}(c_t)=c_tU_i^{\mathrm{c,a}}(t)+(1-c_t)U_i^{\mathrm{u,a}}(t), \tag{4}$$

$$U_i^{\mathrm{d}}(c_t)=c_tU_i^{\mathrm{c,d}}(t)+(1-c_t)U_i^{\mathrm{u,d}}(t). \tag{5}$$

式(4)(5)中，由于 $U_i^{\mathrm{c,a}},U_i^{\mathrm{c,d}}$ 和 $U_i^{\mathrm{u,a}},U_i^{\mathrm{u,d}}$ 满足假设 $U_i^{\mathrm{c,a}}\leqslant U_i^{\mathrm{u,a}}$ 和 $U_i^{\mathrm{c,d}}\geqslant U_i^{\mathrm{u,d}}$，因此 A_i 不希望 at_i 被覆盖，而防御者则相反.

在完全理性的安全博弈中，给定 $\boldsymbol{c}$，A_i 和 D 的最佳响应(best response)定义为最大化它们的收益的策略集合. 具体而言，A_i 的最佳响应 $BR_i^{\mathrm{a}}(\boldsymbol{c})$ 被定义为 T 个被保护对象中最具吸引力的保护对象的集合，而 D 的最佳响应 $BR_i^{\mathrm{d}}(\boldsymbol{c})$ 被定义为 $BR_i^{\mathrm{a}}(\boldsymbol{c})$ 中对 D 最有利的目标的集合.

$$BR_i^{\mathrm{a}}(\boldsymbol{c})=\left\{t\mid U_i^{\mathrm{a}}(c_t)\geqslant U_i^{\mathrm{a}}(c_{t'})\right\}, \tag{6}$$

$$\forall t'\in\{1,2,\cdots,N\},$$

$$BR_i^{\mathrm{d}}(\boldsymbol{c})=\left\{t\middle|U_i^{\mathrm{d}}(c_t)\geqslant U_i^{\mathrm{d}}(c_{t'})\right\}, \tag{7}$$

$$\forall t' \in BR_i^{\mathrm{a}}(\boldsymbol{c}).$$

式(6)(7)分别对集合$\{1,2,\cdots,N\}$和集合$BR_i^{\mathrm{a}}(\boldsymbol{c})$做了取最大元素的max操作, 而max操作是不可微的. 同时, BR_i^{d}不止受c_t的影响, 还受到隐式变量BR_i^{a}的影响, 这使得$BR_i^{\mathrm{d}}(\boldsymbol{c})$的优化十分复杂, 大规模场景下尤甚. 式(6)刻画了对A_i最有吸引力的被保护对象的集合$BR_i^{\mathrm{a}}(\boldsymbol{c})$, 也被记为$\Gamma_i(\boldsymbol{c})$. $\Gamma_i(\boldsymbol{c})$在后文中简写为Γ_i, $|\Gamma_i| \subseteq \{1,2,\cdots,T\}$. SDES[13]指出, 只有分配资源给$\Gamma_i$中的目标才会增加$D$的收益$U_i^{\mathrm{d}}(\boldsymbol{c})$.

式(2)(3)描述了A_i和D策略与收益之间的关系. 式(6)(7)描述了A_i和D相互之间的最佳响应. 在完全理性的假设下, 基本的SG过程如下所述:

1) D在T个被保护对象上分配覆盖向量, 记作$\boldsymbol{c}$;

2) A_i攻击$BR_i^{\mathrm{a}}(\boldsymbol{c})$中的$t$, 返回$\max\limits_{t\in\{1,2,\cdots,T\}} U_i^{\mathrm{a}}(c_t)$, 其中$A_i$选择的保护对象称为被攻击对象$at_i$;

3) A_i获得收益$\max\limits_{t\in\{1,2,\cdots,T\}} U_i^{\mathrm{a}}(c_t)$;

4) 在A_i采取最佳响应后, D从Γ_i中选择t, 以最大化收益$U_i^{\mathrm{d}}(c_t)$;

5) D获得收益$\max\limits_{t\in\Gamma_i} U_i^{\mathrm{d}}(c_t)$.

2.2 多目标安全博弈

本文基于SDES展开了进一步深入研究. SDES包含离散化、优化、评估和解的精炼等4个组件. 具体而言, 离散化组件负责将MOSG任务从高维阶梯函数[13]优化问题转化为低维离散组合优化问题. 优化组件采用多目标演化算法NSGA-III[21]来求解低维组合优化问题. 评估组件采用贪心决策将低维解还原成高维解, 以在原始解空间中评估对应解的收益. 最后, 解的精炼组件尝试改进解集的收敛性.

算法 1. SDES框架.

输入: 随机初始化的T-长度的连续解$\boldsymbol{c}$;

输出: 高质量的T-长度的连续解$\boldsymbol{c}$.

① 使用离散化组件将T-长度高维连续解离散化为N-长度低维离散解, 其中$N \ll T$;

② while 未达到终止条件

③ 使用优化组件的NSGA-III算法优化低维离散解;

④ 通过贪心决策逐位优化将低维解还原为高维连续解;

⑤ 评估高维连续解并将解的适应度返回给优化组件;

⑥ end while

⑦ 使用解的精炼组件精炼最终输出的高维连续解的集合;

⑧ return 高质量的高维连续解的集合.

SDES提供了解大规模MOSG问题的可行思路:

1) 资源分配c_t的原搜索域是$[0,1]$, 但是SDES可以将搜索域从连续域转换到N-长度的离散集合π_{c_t}, 并保证最优解c_t^*存在于π_{c_t}中(具体的离散化方案参见SDES[13]). MOSG问题被转换为组合优化问题, 便于启发式方法求解.

2) MOSG任务扩展时, SDES的评估组件使用贪心策略规避了组合爆炸问题. 具体来说, SDES利用资源分配的独立性逐位优化c_t. 贪心策略使模型复杂度从$O\left(N^T\right)$降至$O(NT)$. 此外, 贪心策略还提供了找到最优解的收敛性理论假设, 称为分割假设, 参见假设1.

假设 1. 分割假设[13]. 贪心策略在第t轮优化c_t时, 假设其在攻击集并集$U=\bigcup\limits_{i=1}^{N}\Gamma_i$中选中的对应的候选集$\pi_{c_t}$是最小的候选集之一.

SDES想要最终找到Pareto前沿上的解要求贪心决策在T轮中都满足假设1. 但是随着攻击者数量N和保护对象数量T的增加, 贪心策略越来越难以保证找到最优解. 因此在大规模情况下, 贪心策略可能无法保证收敛. 虽然SDES框架额外设计了解的精炼组件来解决上述问题, 但这使SDES框架变得更加复杂.

3 基于优先级先验的演化大规模多目标安全博弈框架

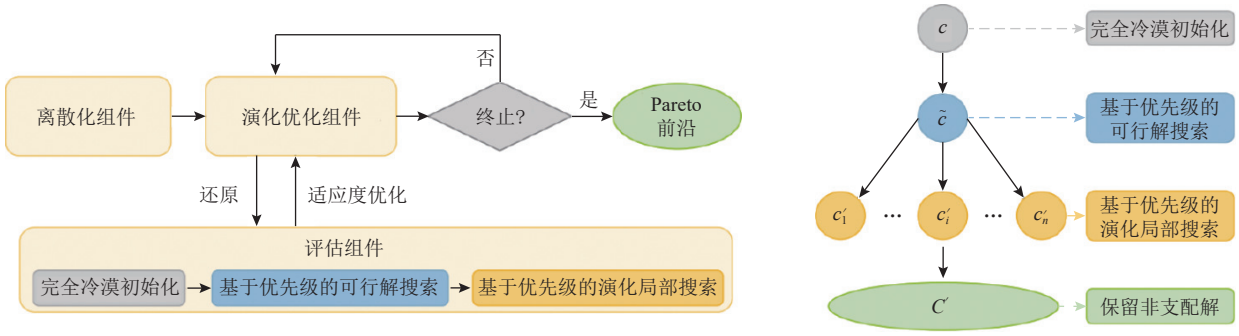
大规模多目标博弈任务是一个高维约束优化问题[9-10]. SDES[13]使用完全冷漠[16](complete indifference)将MOSG问题转化为组合优化问题[13], 其可以通过常规的启发式方法[21-22]来解决. 完全冷漠描述了攻击者关注被保护对象的偏好, 详见引理1.

引理 1. 完全冷漠[16]. 攻击者A_i对于Γ_i中的t表现出相同的兴趣, 但对于$\{1,2,\cdots,T\}\setminus\Gamma_i$中的$t'$表现出完全冷漠.

1)动机. 虽然MOSG问题被转化为一个组合优化问题, 但随着问题规模的增大, 求解难度也急剧增加. 例如, 分量c_t的搜索空间大小为N, 因此SDES框架下的MOSG问题的样本复杂度是一个$O(T^N)$的组合优化问题. 为缓解维数灾难, SDES框架采用贪心策略来解决组合优化问题. 然而, 贪心策略要求假设1在T次迭代中都能够满足. 随着MOSG任务规模的扩大, 假设1变得难以满足. 为消除大规模问题中贪心策略的负面影响, SDES框架设计了一个额外的解

的精炼组件. 然而, 包含太多阶段性组件的框架面临阶段耦合的风险, 这增加了框架的复杂性和不稳定性. 因此, 在大规模 MOSG 任务中解决组合优化问题的贪心策略可能并不是一个好方案.

基于 SDES 框架[13], 本文提出了基于优先级的演化大规模多目标安全博弈框架 SDES-P. 相较于 SDES 框架, SDES-P 保留了离散化和优化组件, 重新设计了评估组件, 并且移除解的精炼组件, 详见图 1(a). 现有研究[13,16]表明, MOSG 先验在解决组合优化问题时具有潜在的优势. 具体包括: ①使用更多资源的解更有可能获得更优的收益[13]; ②释放不被攻击者关心的保护对象的资源对收益的影响较小[16]. 基于先验①, 本文首先设计了基于完全冷漠定理的解初始化策略. 基于先验②, 本文分别设计了基于优先级的可行解搜索策略和基于优先级的演化局部搜索策略. 评估组件的 3 个策略如图 1(b)所示. 通过这 3 个策略的组合, SDES-P 的样本复杂度在保持与 SDES 相近的前提下, 能找到一个多样性和收敛性更好的Pareto前沿. 本文提出的 SDES-P 框架比 SDES 更为精简, 降低了阶段耦合的风险.



(a) SDES-P 框架 (b) 评估组件细节

注: c 为初始解, $\tilde{c}$为可行解, c'_i为局部搜索解, C'为 Pareto 前沿.

Fig. 1 The process of SDES-P framework and the core evaluation component

图 1 SDES-P 框架和核心评估组件的流程

2)目标. 与 SDES 框架相比, 得益于优先级先验知识[13,16], SDES-P 具有更简单的框架和更强大的评估组件. SDES-P 能够实现以下目标:

①探索 MOSG 先验知识的影响. 即使没有贪心策略, 使用 MOSG 优先级先验的 SDES-P 也可以有效地解决组合优化问题, 并获得高质量解.

②简化 SDES 框架并降低阶段耦合风险. 借助新设计的评估组件, SDES-P 可以在不依赖解的精炼组件的情况下找到具有良好收敛性和多样性的Pareto前沿, 降低不同组件之间高度耦合带来的风险.

③提升求解质量的同时保持算法的可扩展性. 我们力求 SDES-P 在大规模 MOSG 任务中提高收敛性以及求解质量, 并且保持 SDES 框架的规模可扩展性优势.

3.1 完全冷漠初始化策略

本节首先利用定理 1 缩小解初始化的范围, 然后利用从 SDES 中获得的先验知识制定了一个启发式的解初始化策略.

定理 1[13]. 如果 $t\in\bar{U}$, 则 $c_t=0$, 否则 $c_t\in\pi_{c_t}$.

定理 1 提供了一种解的初始化思路. 在定理 1 中, 我们定义 $U=\underset{i\in\{1,2,\cdots,N\}}{\cup}\Gamma_i$ 和 $\bar{U}=\{1,2,\cdots,T\}\setminus U$. 由引理 1 可知, D的综合收益$(U_1^{\mathrm{d}}(\boldsymbol{c}),U_2^{\mathrm{d}}(\boldsymbol{c}),\cdots,U_N^{\mathrm{d}}(\boldsymbol{c}))$仅受到$U$中目标资源配置的影响. 因此, $\bar{U}$中的保护对象拥有的资源在初始化时分配为0. 对于U中的保护对象t, 最佳资源分配方案c_t^*的搜索域从连续空间$[0,1]$离散化为集合π_{c_t}, 其中$|\pi_{c_t}|\leqslant N$.

另一方面, SDES 框架的贪心策略发现在各种问题规模中, 高质量的解$\boldsymbol{c}$都倾向于选择备选集π_{c_t}中具有最大资源的备选方案c_t. 因此, 一个可行的想法是将每个组成部分c_t初始化为资源最多的备选方案.

$$c_t=\begin{cases}\max\left(\pi_{c_t}^1,\cdots,\pi_{c_t}^N\right), t\in U.\\ 0, \text{其他}.\end{cases} \tag{8}$$

式(8)展示了每个组成c_t的完全冷漠初始化策略. 对于所有$t\in U$, 其对应的c_t被初始化为π_{c_t}中资源数最大值. 根据式(3), 单目标函数$U_i^{\mathrm{d}}(c_t)$与资源c_t成正比. 因此, 以最大资源初始化的备选方案在多目标任务中更有潜力.

3.2 基于优先级的可行解搜索策略

3.1 节启发式策略将解初始化为具有最大资源数量的解, 通常会得到违反总资源限制的解, 即$\|\boldsymbol{c}\|_1>$

$r\times T$. 因此, 本文提出一种基于优先级的搜索策略来寻找可行解. 根据引理 1, 攻击者对$\bar{U}$中的保护对象表现出完全的冷漠, 因此U中的保护对象的优先级高于$\bar{U}$中的保护对象. 一个高优先级的保护对象具有以下特点: 它可以释放资源的同时保持多目标优化的质量, 因此释放高优先级保护对象拥有的资源可以使得初始解更接近可行解. 定义 3 定义了每次资源释放时可以释放的最小资源量Δ_{c_t}, 定理 2 从理论上保证了资源以最小资源释放量释放能最小化对单个防御者目标收益的影响. 进而该资源释放方式对防御者整体目标收益的影响有可能保持在可控范围内.

定义 3. 最小资源释放量. 假设当前$c_t=\pi^i_{c_t}$, c_t的最小资源释量记作$\Delta_{c_t}=\pi^i_{c_t}-\pi^{i+1}_{c_t}$.

定理 2. 释放资源影响最小化. 假设只有 1 个分量c_t在每次更新$\boldsymbol{c}$时发生变化, 且资源以最小资源释放量被释放, $\boldsymbol{c}$的这种更新方式对D的收益$U^{\mathrm{d}}_i(c_t)$的影响最小.

证明. 由式(5)得知, D的单个目标i的收益U^{d}_i与资源分配量c_t的关系为$U^{\mathrm{d}}_i(c_t)=c_tU^{\mathrm{c,d}}_i(t)+(1-c_t)\times U^{\mathrm{u,d}}_i(t)$. 经过简单的恒等变换得到$U^{\mathrm{d}}_i(c_t)=(U^{\mathrm{c,d}}_i(t)-U^{\mathrm{u,d}}_i(t))c_t+U^{\mathrm{u,d}}_i(t)$, 其中防御者保护成功的收益$U^{\mathrm{c,d}}_i(t)$一定不小于保护失败的收益$U^{\mathrm{u,d}}_i(t)$.

在定理 2 中, 假设分量$c_t=\pi^i_{c_t}$, 其中$\pi^i_{c_t}$表示当前第i大的候选集. 若c_t以定义 3 的最小资源释放量更新, 相比于其他更小的候选集$\{\pi^{i+1}_{c_t},\pi^{i+2}_{c_t},\cdots,\pi^N_{c_t}\}$, 最小资源释放量策略对$U^{\mathrm{d}}_i(c_t)$的影响是最小的. 证毕.

注意: 定义 3 中, 上标i表示$\pi^i_{c_t}$是π_{c_t}中第i大的元素. 定理 2 中更新方式的基本思想是, 只有被A_i选择的保护对象t对$U^{\mathrm{d}}_i(c_t)$的计算有贡献[13], 所有的$c_{t'}(t'\neq t)$对防御者的收益没有影响. 因此, 每次只更改 1 个分量不会对防御者的收益产生剧烈的影响. 值得注意的是, 为了防止c_t的资源减少过多而引起其余A_i的注意, c_t应以最小资源量Δ_{c_t}释放资源.

SDES-P 的基于优先级的可行解搜索策略见算法 2. SDES-P 采用了随机无重复采样的方式探索所有保护对象. 在选择更新的分量时, $t\in\bar{U}$具有更高的优先级(行③). 每次更新$\boldsymbol{c}$时, 只更新其中 1 个分量c_t, 更新c_t时以最小资源释放量Δ_{c_t}(定义 3)进行更新(行④). 更新过程直到找到一个可行解$\tilde{\boldsymbol{c}}$为止. $\tilde{\boldsymbol{c}}$是一个资源使用量大的高质量可行解.

算法 2. 基于优先级的可行解搜索算法.

输入: 完全冷漠初始化的解$\boldsymbol{c}$, 资源总量$r\times T$, $\boldsymbol{c}$所有分量的备选集$\{\pi_{c_1},\pi_{c_2},\cdots,\pi_{c_T}\}$;

输出: 高质量的可行解$\tilde{\boldsymbol{c}}$.

① while $\|\boldsymbol{c}\|_1>r\times T$

② 以$\boldsymbol{c}$更新U与$\bar{U}$, 见式(6);

③ 分别随机打乱U与$\bar{U}$, 生成有序集合$\Gamma_{\mathrm{pri}}\leftarrow\bar{U}\cup U$. 在生成的有序集合$\Gamma_{\mathrm{pri}}$中, 要求优先级高的$\bar{U}$的$t'$排列在优先级低的$U$的$t$之前, $\forall t'\in\bar{U},\forall t\in U$;

④ 寻找Γ_{pri}中第 1 个没被选中过的t, $c_t\leftarrow c_t-\Delta_{c_t}$;

⑤ end while

⑥ return 高质量的可行解$\tilde{\boldsymbol{c}}$.

3.3 基于优先级的演化局部搜索策略

3.2 节策略找到了一个资源使用量较大的可行解$\tilde{\boldsymbol{c}}$, 这与安全博弈的直觉相符, 因为更高的资源分配可能会获得更好的综合收益. 然而, 在异质攻击者的情况下, 过多的资源分配可能会对综合收益造成负面影响. 具体而言, 当分配大量资源给保护对象t, 期望通过防御A_i以追求更高收益$U^{\mathrm{d}}_i(c_t)$时, 由于资源有限, 其他t'会因为资源不足而遭到其他A_j的攻击, 从而降低综合收益. 这是因为分配的资源c_t可能会使得保护对象t对于A_j不再具有吸引力, 从而导致A_j攻击其他分配了较少资源的保护对象t', 最终D获得的$U^{\mathrm{d}}_j(c_{t'})$低于$U^{\mathrm{d}}_j(c_t)$. 这种性质使得贪心 SDES 框架很难满足假设 1, 并且假设达成难度随着问题规模的扩大而增加. 上述问题源于A_i的异质性, 即防御者D需要指定不同分配方案防御A_i, 但D最终只能采用其中一种方案. D的选择会由于A_i的异质性产生冲突, 并随问题规模扩大而愈发严重. 总之, 在异质攻击者的情况下, D不能仅通过增加资源分配来提高综合收益, 使用大量资源的可行解不一定是最优解.

一种最大化综合收益的可行方法是基于高质量可行解的局部搜索策略. 局部搜索策略能改善Pareto前沿的质量, 包括收敛性和多样性[23−25]. 本节介绍了基于优先级的演化局部搜索算法, 详见算法 3. 首先, 算法 3 从算法 2 找到的可行解出发探索使用更少资源的解. 在资源使用较少的方向上进行探索, 不仅可以确保新解是可行解, 还有可能找到更高质量的新解[13].

算法 3. 基于优先级的演化局部搜索算法.

输入: 可行解搜索算法找到的高质量可行解$\tilde{\boldsymbol{c}}$, 资源总量$r\times T$, $\boldsymbol{c}$分量的备选集$\{\pi_{c_1},\pi_{c_2},\cdots,\pi_{c_T}\}$;

输出: Pareto前沿C'.

① $C'\leftarrow$ Pareto前沿初始化$\{\tilde{\boldsymbol{c}}\}$;

② 以$\tilde{\boldsymbol{c}}$更新U与$\bar{U}$;

③ 随机打乱U与$\bar{U}$, 若$\bar{U}\neq\varnothing$, 则$\Gamma_{\mathrm{pri}}=\bar{U}$, 否则

$\Gamma_{\text{pri}} = U$;

④ for t in Γ_{pri}

⑤ $\boldsymbol{c}' \leftarrow$ 只更新 $\tilde{\boldsymbol{c}}$ 的一个分量 $\tilde{c}_t$, $\tilde{c}_t \leftarrow \tilde{c}_t - \Delta_{\tilde{c}_t}$;

⑥ $C' \leftarrow$ 保留 $C' \cup \{\boldsymbol{c}'\}$ 中的非支配解;

⑦ end for

⑧ return Pareto前沿 C'.

算法 3 的关键思想是一步减少 $\tilde{\boldsymbol{c}}$ 中的某个分量以得到一个新的可行解 $\boldsymbol{c}'$. 类似于基于优先级的可行解搜索策略, 优先级先验体现在 U 与 $\bar{U}$ 的分组(见定理 1)和最小资源释放量 $\Delta_{\tilde{c}_t}$(见定义 3)上, 能够引导算法选择要减少的分量以及量化减少的数量. 具体来说, 行③中, 演化局部搜索策略优先探索 $t \in \bar{U}$, 除非 $\bar{U} = \varnothing$. 此外, 行⑤资源 $\tilde{c}_t$ 的减少量遵循定义 3. 最后, $\boldsymbol{c}'$ 和 $\tilde{\boldsymbol{c}}$ 的非支配解被保留为Pareto前沿 C'.

3.4 SDES 框架与 SDES-P 框架的区别与联系

本节首先讨论了 SDES-P 和 SDES 之间在组件级别上的关系, 包括相似和差异之处. 然后给出了 SDES-P 中评估组件的伪代码, 并展示了 SDES-P 在解决组合优化问题上的优越性. 最后展示了 SDES-P 和 SDES 在 $O(NT)$ 采样复杂度方面的相似之处.

SDES-P 和 SDES 之间组件层面的关系为 SDES[13] 包括离散化、优化、评估和解的精炼等 4 个组件, 而 SDES-P 使用了除解的精炼组件外的同功能组件, 并对评估组件进行了深刻改进. 具体而言, SDES-P 和 SDES 都以低维优化、高维评估的思想处理大规模问题. 因此, 2 个框架都包含离散化、优化和评估组件. 此外, 2 个框架的评估组件都面临组合爆炸的问题, 但是它们的解决方式不同. 特别地, SDES-P 在优先级的指导下搜索高质量的解. 在大规模 MOSG 中, SDES-P 比 SDES 更不容易违反假设 1. 此外, SDES-P 不依赖于解的精炼组件, 比 SDES 更简单、耦合风险更小. 这些差异将在下面详细介绍.

SDES-P 的优越性在于解决组合优化问题时使用了 MOSG 的优先级先验. 对于 SDES, 贪心策略避免了组合优化的 $O(NT)$ 样本复杂度, 但随着问题规模的增加, 假设 1 愈加难以满足. 对于 SDES-P, MOSG 先验信息被用来解决组合优化问题. 具体而言, 定理 1 表明算法寻优中只需搜索 $t \in U$, 且 $t \in \bar{U}$ 上分配的资源具有更高的释放优先级. 定义 3 指定了需要释放的最小资源量. 此外, SDES-P 首先找到高质量的可行解, 然后基于可行解进行局部探索以获得更高质量的解. 因此 SDES-P 不像 SDES 一样面临假设 1 随着问题规模增加而变得难以满足的问题. 此外, SDES 中的贪心策略只返回 1 个非支配解, 而 SDES-P 中的演化局部搜索策略可以返回一个包含多个非支配解的非支配解集 C'.

算法样本复杂度的量级同阶. SDES-P 评估组件的伪代码展示在算法 4 中. 在算法 4 中, 评估组件的每次迭代都经过完整的完全冷漠初始化策略、基于优先级的可行解搜索策略和基于优先级的演化局部搜索策略. 其中复杂度最高的策略是可行解搜索策略, 参见算法 2. 具体而言, 更新长度为 T 的 $\boldsymbol{c}$ 时, 每次只能更改一个分量 c_t. 同时定理 2 表明, 每次算法 2 更新 c_t 时, 使用的 π_{c_t} 元素都比当前的 c_t 小, 并且由于所有 c_t 的初始值都设置为 π_{c_t} 中的最大值以及 $|\pi_{c_t}| \leqslant N$, 因此整个评估组件的最差时间复杂度为 $O(NT)$, 因此 SDES-P 的评估组件与 SDES 框架的评估组件具有相似的样本复杂度.

算法 4. SDES-P 的评估组件.

输入: $\boldsymbol{c}$ 所有分量的备选集 $\{\pi_{c_1}, \pi_{c_2}, \cdots, \pi_{c_T}\}$;

输出: Pareto前沿 C'.

① 以完全冷漠初始化策略初始化解 $\boldsymbol{c}$, 见式(8);

② $\boldsymbol{c} \leftarrow$ 以基于优先级的可行解搜索策略寻找资源使用量大的可行解, 见算法 2;

③ $C' \leftarrow$ 基于优先级的演化局部搜索策略, 见算法 3;

④ return Pareto前沿 C'.

4 实　　验

4.1 实验设置

4.1.1 实验环境与超参数设置

SDES-P 的实验环境与 SDES[13] 保持同阶. 所有实验都在随机拟真 MOSG 基准测试集[9-10,16] 上进行. 以下所有 MOSG 任务都为最小化任务. 主要的实验设置包括:

1) MOSG 问题规模主要受到目标数量 T 和攻击者数量 N 的影响, 分别满足约束条件: $3 \leqslant N \leqslant 20$, $T \in \{25, 50, 75, 100, 200, 400, 600, 800, 1\,000\}$.

2) 随机生成的 MOSG 问题的配置包括资源比 r、防御者 D 和攻击者 A_i 的成功和失败的收益分布. 资源比 r 控制资源总量 $r \times T$. 由于资源有限, 通常资源比 $r \ll 1$, 我们的实验中选择 $r = 0.2$. 此外, D 和 A_i 的成功收益遵循 [1, 10] 的均匀整数分布, 而失败收益遵循 $[-10, -1]$ 的均匀整数分布.

3) 其余实验设置包括终止条件和编程细节. 算法的运行时间以该算法从开始到收敛的时钟时间为准. 本文在具体实验中会设置最大等待时间, 若算法

收敛所需的时间超过最大等待时间, 则认为算法超时. 编程语言为 Python, 硬件条件为 255 AMD EPYC 7742 64-Core 处理器.

本文同时比较了 6 种算法, 包括基于ϵ-约束框架的 4 种精确算法和基于 SDES 框架的 2 种近似算法. ϵ-约束算法包括 ORIGAMI-M[9] 和 ORIGAMI-A[9]、ORIGAMI-M-BS[10] 和 DIRECT-MIN-COV[10]. 所有 ϵ-约束算法都使用步长$\epsilon=1$. ORIGAMI-A 使用阈值$\alpha=0.001$进行二分搜索决定是否终止. SDES 算法包括 SDES 基线算法[13] 和本文提出的 SDES-P.

4.1.2 多目标优化指标设置

定义 2 中的Pareto支配关系是单个解比较时的评估准则, 本节介绍的反转世代距离+(modified inverted generational distance, IGD^+)[26] 和超体积(hypervolume, HV)[27] 是 Pareto集合的评估准则, 用于算法评估. IGD^+衡量参考Pareto前沿到Pareto近似前沿的距离, 而HV衡量Pareto前沿的收敛性和多样性.

1) IGD^+. IGD^+[26] 是IGD[28] 的改进版. 在$IGD(C')=\frac{1}{|Z|}\left(\sum_{i=1}^{|Z|}\hat{d}_i^p\right)^{\frac{1}{p}}$中, IGD计算从参考Pareto前沿Z到Pareto近似前沿C'的欧几里得距离. 具体来说, IGD将Z中每个点分别与C'中的最近点组成点对, 并计算所有点对的欧几里得距离, 形式化为 $\hat{d}_i=\sqrt{(\boldsymbol{c}'_1-\boldsymbol{z}_1)^2+\cdots+(\boldsymbol{c}'_{\mathrm{T}}-\boldsymbol{z}_{\mathrm{T}})^2}$ 如式(9)所示, IGD^+规定只有正距离的分量才能做出贡献, 形式化为

$$d_i^+=\sqrt{(\max(\boldsymbol{c}'_1-\boldsymbol{z}_1,0))^2+\cdots+(\max(\boldsymbol{c}'_{\mathrm{T}}-\boldsymbol{z}_{\mathrm{T}},0))^2}.$$

该改进保证了IGD^+满足弱Pareto条件, 而IGD则不能保证满足这一条件[26]. 在本文指定 P=1.

$$IGD^+(C')=\frac{1}{|Z|}\sum_{i=1}^{|Z|}d_i. \tag{9}$$

2) HV. HV[27] 描述了被Pareto近似前沿支配的空间体积, 可以表示为

$$HV(C',\boldsymbol{r})=\mathcal{L}\left(\bigcup_{\boldsymbol{c}'\in C'}\{\boldsymbol{b}|\boldsymbol{c}'\succcurlyeq\boldsymbol{b}\succcurlyeq\boldsymbol{r}\}\right), \tag{10}$$

其中C'为待评估算法找到的Pareto近似前沿, $\boldsymbol{r}$为事先准备好的参考点, $\mathcal{L}(\cdot)$为集合的勒贝格测度$\mathcal{L}(B)=\int_{\boldsymbol{b}\in B}I_B(\boldsymbol{b})\mathrm{d}\boldsymbol{b}$. I_B为集合B的特征函数, 若$\boldsymbol{b}\in B$, 则$I_B(\boldsymbol{b})=1$, 否则$I_B(\boldsymbol{b})=0$, $\boldsymbol{c}'\succcurlyeq\boldsymbol{b}$表示$\boldsymbol{c}'$支配$\boldsymbol{b}$, 见定义 1.

HV优点是: 1) HV是唯一满足Pareto条件的单位指标[29]. 这表示HV是严格单调的[30], 即能确保最大化HV的解集是Pareto前沿. 较劣算法只会获得更小的HV指标, 但反之不成立[23]. 2)在计算HV时只需要提供一个参考点, 而IGD^+需要整个Pareto前沿. 3) HV可以同时评估算法的收敛性和多样性. 此外, 其缺点是: 1)虽然有许多HV的实现, 但其复杂度通常较高, 已知的复杂度上限为$O\left(|Y_N|^{m/3}\mathrm{ploylog}|Y_N|\right)$[31], 其中 ploy 表示多项式复杂度. 2)不同算法的HV相对关系受到参考点选择的影响[32], 当算法性能相似时尤为明显.

4.1.3 Pareto 前沿的构建

为了找到一个具有良好收敛性和多样性的近似Pareto前沿, 实验选择定量指标HV和IGD^+来评估算法的收敛性和多样性. 然而, IGD^+评估需要参考Pareto前沿, 而随机生成的 MOSG 任务无法提供参考Pareto前沿. 构建参考Pareto前沿的一种可行方法是收集所有求解器$s\in S$产生的一组 Pareto前沿, 即$PF_S^p=\bigcup_{s\in S}PF_s^p$, 并删除$PF_S^p$中的所有被支配解. 求解器$S$包括 2 种基于 SDES 的算法和 4 种$\epsilon$-约束算法.

4.2 性能分析实验

SDES 是解决大规模 MOSG 问题的有效框架[13]. 如 3.4 节所述, SDES-P 框架的样本复杂度与 SDES 框架同级. 因此, 本节讨论 SDES-P 框架在Pareto前沿质量方面相较于 SDES 框架的优越性. 性能实验包括: 1)比较所有算法在所有 MOSG 问题规模上的Pareto前沿质量与求解效率(见 4.2.1 节和 4.2.3 节); 2) 仅改变目标数量或保护对象数量, 随着 MOSG 问题规模的增加, 研究所有算法发现的Pareto前沿质量的变化(见 4.2.2 节). 需要说明的是, 实验 1)只展示所有算法都能完成的问题规模, 而实验 2)则没有这个限制, 它展示算法在各个维度上的极限规模.

4.2.1 在所有问题规模上的性能分析

本实验展示了 6 种算法在 21 种 MOSG 问题上的结果, 这些任务涵盖了各种问题规模. 21 种问题规模分别包括: 1) $N=3$时, $T\in\{200,400,600,800,1\,000\}$; 2) $N=4,5,6$时, $T\in\{25,50,75,100\}$. 其中规模 1) 考虑了大规模的被保护对象T的场景, 而规模 2) 在保持中等规模T的同时还考虑了中等规模的异质攻击者数量N. 算法在求解每个问题规模时, 只有在等待时间 30 min 内收敛才被记录为完成任务, 否则记为超时. 图 2 显示了 6 种算法在 21 个问题规模下的IGD^+和HV指标上的排序分布.

首先就求解质量而言, SDES-P 在大部分规模上能取得第一名, SDES 次之, 不过ϵ-约束算法也在少数几个问题上表现良好. 可以发现, ϵ-约束算法在中等规模$N=5$上的表现优于 SDES 系列的算法. 在中等规模问题中, 所有算法都能在规定时间内完成任务, 并且任务求解难度普遍大于小规模任务, 因此精确算法凭借良好的收敛性能获得较高的IGD^+和HV

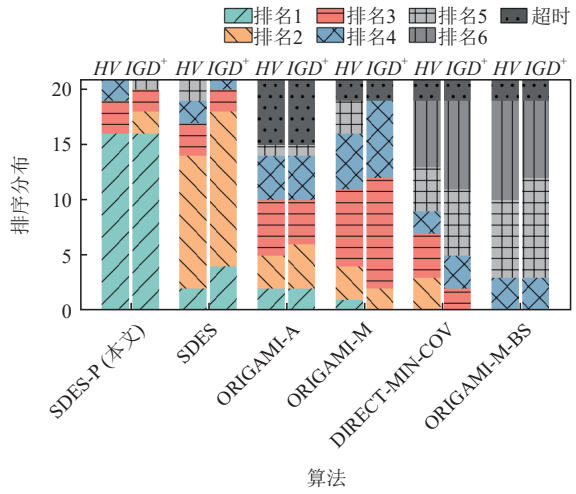


Fig. 2 Ranking distribution of IGD^+ and HV indicators of six algorithms on 21 MOSG tasks with different scales in 30 min

图 2 六种算法在 30 min 内在 21 种不同规模的 MOSG 任务上的IGD^+和HV指标的排序分布

指标评分. 同时, 在各种规模上的实现也表明了ϵ-约束算法的特性与局限, 即 ORIGAMI-M-BS 和 DIRECT-MIN-COV 算法虽然超时次数少但是求解质量低, 而 ORIGAMI-M 和 ORIGAMI-A 算法虽然在中等问题上表现良好但是难以向大规模问题扩展. 而 SDES 系列算法则同时达成了求解质量与规模可扩展性的目标. 其次就规模可扩展性而言, SDES-P 与 SDES 能按时完成所有任务的求解, SDES-P 在IGD^+和HV上的总和表现超过了 SDES. 这充分说明 MOSG 先验知识与优化技巧的重要贡献. 后文的消融性实验中还会详细分析本文核心工作的有效性.

4.2.2 算法精度随问题规模的变化

SDES-P 相比于 SDES, 使用了 MOSG 优先信息并降低了阶段耦合的风险. 在 4.2.1 节中, 我们展示了 SDES-P 相比于在 21 个各种规模的 MOSG 任务中的优势. 4.2.1 节讨论的任务很少涉及大规模场景. 事实上 SDES-P 在大规模 MOSG 任务中更具优势. 本实验分析了随着 MOSG 规模的增加(包括目标数和攻击者数量)的所有算法的性能表现. 由于ϵ-约束框架的算法难以在规定时间内完成大规模 MOSG 任务, 参与对比的算法只有 SDES, 其中IGD^+是最小化指标, HV是最大化指标. IGD^+和HV的相对大小受任务难度影响[27], 而任务难度不完全由问题规模决定, 所以IGD^+和HV指标与问题规模间不存在严格的单调性. 因此, 本节的实验结果采用表格呈现而非折线图. 此外由于HV指标的计算复杂度随目标数增加指数级变化, 因此结果展示中只分析了所有算法在IGD^+指标上的表现.

表 1 和表 2 显示 SDES-P 分别在 2 个任务中取得了 9/9 和 16/18 个最优. 在中小规模的场景下, SDES-P 和 SDES 各具优势, 但是随着T和N扩大, SDES 找到的Pareto前沿质量快速下降, 而 SDES-P 找到的Pareto前沿质量存在下界保证, 例如在表 2 的$N = 15 \sim 20$大规模实验中. 综上, T和N这 2 个维度单独变化的实验结果统计表明 SDES-P 相比 SDES 在求解质量上有很大的改进. 结合 3.4 节样本复杂度的分析, 我们能看出 SDES-P 具备 SDES 的样本复杂度低、规模可扩展性强的优势, 尤其在大规模场景下能找到相较于 SDES 收敛性、多样性更优的Pareto前沿.

Table 1 IGD^+ Indicator Comparison of SDES and SDES-P about T in the N=3 MOSG Task in 30 min

表 1 SDES 与 SDES-P 在 30 min 内 N=3 的 MOSG 任务上针对 T 的 IGD^+指标对比

算法	$T = 25$	$T = 50$	$T = 75$	$T = 100$	$T = 200$
SDES	0.44±0.02	0.39±0.04	0.27±0.16	0.28±0.12	0.32±0.02
SDES-P	**0.31±0.02**	**0.22±0.03**	**0.18±0.01**	**0.17±0.04**	**0.15±0.00**
算法	$T = 400$	$T = 600$	$T = 800$	$T = 1\,000$	
SDES	0.32±0.03	0.42±0.04	0.42±0.08	0.44±0.01	
SDES-P	**0.20±0.01**	**0.25±0.03**	**0.12±0.01**	**0.20±0.03**	

注：黑体数值表示最优值.

表 1 和表 2 展现出的 SDES-P 在大规模 MOSG 任务求解精度上的优势主要来自于本文提出的优先级策略. 在 SDES-P 中, 优先级策略通过优先地释放$t \in \bar{U}$上的资源, 最小化异质攻击者A_i意图改变的数量. 上述过程中, 选择资源释放对象的搜索域为T, 而异质攻击者的数量为N. 虽然 SDES 也引入了攻击集$\{\Gamma_1, \Gamma_2, \cdots, \Gamma_N\}$的概念, 但是并没有明确提供一个显式的划分策略. 此外, SDES-P 还在优化过程中实时更新U. 具体来说, SDES-P 每次做演化局部搜索前都会按照当前资源分配情况重新计算U, 保证每次分组的正确性. 随着问题规模T和N的扩大, 采用静止的优先级策略的 SDES 的出错率将越来越大, 导致不能很好

Table 2 IGD^+ Indicator Comparison of SDES and SDES-P about N in the T=25 MOSG Task in 30 min

表 2 SDES 与 SDES-P 在 30 min 内 T=25 的 MOSG 任务针对 N 的 IGD^+指标对比

算法	$N=3$	$N=4$	$N=5$	$N=6$	$N=7$	$N=8$
SDES	0.44±0.02	0.32±0.02	**0.29±0.07**	**0.49±0.04**	0.58±0.03	0.83±0.03
SDES-P	**0.31±0.02**	**0.30±0.03**	0.65±0.15	0.53±0.17	**0.57±0.04**	**0.72±0.06**
算法	$N=9$	$N=10$	$N=11$	$N=12$	$N=13$	$N=14$
SDES	1.54±0.29	0.96±0.03	1.60±0.22	1.86±0.26	1.48±0.12	1.81±0.39
SDES-P	**0.76±0.17**	**0.64±0.03**	**1.29±0.09**	**1.03±0.17**	**1.08±0.35**	**1.21±0.29**
算法	$N=15$	$N=16$	$N=17$	$N=18$	$N=19$	$N=20$
SDES	2.18±0.30	2.06±0.41	2.36±0.38	2.27±0.32	2.24±0.27	2.37±0.26
SDES-P	**1.42±0.14**	**1.39±0.34**	**1.38±0.32**	**1.34±0.35**	**1.49±0.39**	**1.45±0.37**

注：黑体数值表示最优值.

地收敛到最优的Pareto前沿上.

4.2.3 算法效率随问题规模的变化

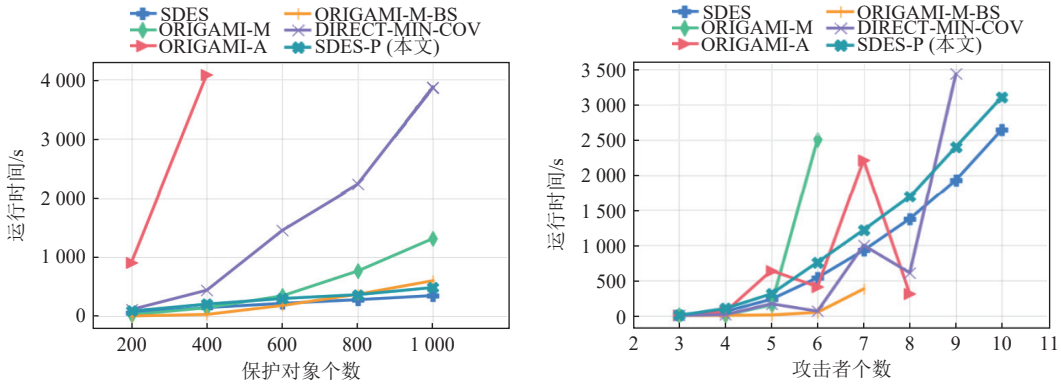
3.4 节指出 SDES-P 框架求解 MOSG 任务的样本复杂度和 SDES 框架是相同量级的. 本节在大规模 MOSG 任务中验证. 大规模 MOSG 任务可分为 2 类: 保护对象T大规模和攻击者N大规模, 分别展示在图 3(a)和图 3(b)中. 其中N只取到 10 是因为大部分对比算法都无法在最大等待时间 30 min 内收敛.

图 3(a)(b)表明 SDES-P 框架在大规模保护对象和大规模攻击者场景下都明显优于基于ϵ-约束框架的对比算法, 例如, ORIGAMI-M 和 ORIGAMI-A 等. 同时, SDES-P 的时耗变化率与 SDES 相似, 说明它们的样本复杂度在同一个量级上. 结合图 3(b)可以看出N=10 时 SDES-P 耗时距离最大等待时间 30 min 较远, 结合 SDES-P 关于N接近线性的变化率, 可以预估 SDES-P 在更大规模的问题上的可行性. 因此, SDES-P 的求解效率明显优于基于ϵ-约束框架的算法的同时, 保持了 SDES 框架中良好的规模可扩展性.

4.3 消融实验

在 4.2.2 节中, SDES-P 在大规模 MOSG 问题上取得了明显的优势. SDES-P 的主要贡献是重新设计的评估组件, 包括完全冷漠初始化策略、基于优先级的可行解搜索策略和基于优先级的演化局部搜索策略. SDES-P 的核心优势在于利用了 MOSG 优先级先验和使用了演化局部搜索策略. 其中, SDES-P 的核心算法不涉及特别的超参数. 因此, 本节专注于消融实验, 讨论 2 点: 1)优先级先验在 SDES-P 框架中的贡献; 2)演化局部搜索策略在 SDES-P 框架中的贡献.

表 3 中设计了 4 个场景: 1)SDES 作为基线模型; 2)SDES-P 移除优先级先验; 3)SDES-P 移除局部搜索; 4)完整的 SDES-P 框架. 具体而言, 没有优先级先验的 SDES-P 意味着可行解搜索策略和演化局部搜索策略不再受优先级先验的指导, 即目标$t\in\bar{U}$不再被优先探索. 没有演化局部搜索策略的 SDES-P 意味着由基于优先级的可行解搜索策略找到的非支配解将直接用于构建Pareto前沿. 由于不同场景的复杂度不同, 所以基于固定时间而非固定迭代次数来评估不



(a) 在大规模保护对象场景下（N=3, T=200~1 000）

(b) 在大规模攻击者场景下（N=3~10, T=25）

Fig. 3 Solving time of the six algorithms under large-scale protected objects and large-scale attackers

图 3 6 种算法在大规模保护对象和大规模攻击者下的求解时间

Table 3 Ablation Experiments on the Large-Scale MOSG Task with $N = 5$, $T = 200$

表 3 在$N = 5$, $T = 200$的大规模 MOSG 任务上的消融实验

算法	HV（排名）	IGD^+（排名）
SDES	6.80E5 ± 1.71E4(4)	0.90 ± 0.05(4)
SDES-P 移除优先级先验	6.83E5 ± 2.39E4(2)	0.66 ± 0.06(2)
SDES-P 移除局部搜索	6.81E5 ± 1.02E4(3)	0.89 ± 0.05(3)
SDES-P	6.85E5 ± 1.63E4(1)	0.61 ± 0.04(1)

同场景. 表 3 显示了 4 个场景下的Pareto前沿的HV和IGD^+指标.

消融实验选择$N = 5$, $T = 200$的 MOSG 问题规模, 这是因为此时所有的精确解法都超时, 该规模下的求解只能依赖基于 SDES 框架的算法. 表 3 展示了大规模问题下 SDES 基线和 SDES-P 的重要组件的表现. 具体地, 算法 1 的 SDES 基线在HV最大化指标以及IGD^+最小化指标上都表现得最差, 而算法 4 的 SDES-P 表现最好. 同时对比算法 1 和算法 3, 如果 SDES-P 不采用演化局部搜索策略, SDES-P 会退化成 SDES 算法. 算法 2 和算法 4 的比较也说明了优先级先验的作用, 算法 2 仅使用演化局部搜索但是没利用 MOSG 任务的优先级先验, 虽然效果优于算法 1 和算法 3, 但是仍然和完整的 SDES-P 框架有一定差距, 反映了 MOSG 优先级先验在进一步提升解的精度上的重要性. 同时这也是 SDES-P 框架能够去掉 SDES 框架中的解的精炼组件的重要原因.

5 结　论

本文提出了一种基于优先级先验的演化多目标安全博弈通用框架 SDES-P, 用于解决大规模多目标安全博弈任务. SDES-P 是一个同时考虑了求解质量和规模可扩展性的 MOSG 简化求解框架, 其通过优先级先验重新设计了评估组件并简化了 SDES 框架, 可以在降低阶段耦合风险的同时解决各种规模的 MOSG 任务. 本文分析发现 SDES-P 可保持 SDES 样本复杂度低、规模可扩展性强的优势, 并且各种规模下的 MOSG 对比实验表明, MOSG 任务不满足收敛假设时, SDES-P 可以找到相较于 SDES 收敛性、多样性更优的高质量 Pareto 前沿. 消融实验验证了 SDES-P 框架中各组件的必要性与贡献.

SDES-P 框架的局限主要体现在样本复杂度和优先级预处理方面. 虽然在复杂度分析中 SDES-P 的评估组件和 SDES 的评估组件具有同级样本复杂度$O(NT)$, 但从大规模 MOSG 场景的性能实验中可以看出, SDES-P 在目标维度上的规模可扩展性稍逊于 SDES. 也即, SDES-P 的复杂度常数项比 SDES 大. 虽然从实验上来看 SDES-P 相比于 SDES 的复杂度的常数项更大, 但是相比于其他基于ϵ-约束框架的对比算法, 基于演化算法的 SDES-P 算法是易于并行化加速的. 此外, SDES-P 使用的优先级预处理的效果尚有进一步挖掘空间. 本文中的优先级预处理仅将目标分为 2 组: U组和$\bar{U}$组, 并指定保护对象$t \in \bar{U}$比$t \in U$组的目标具有更高的释放资源的优先级. 不过, 本文没有为属于同一组的 2 个被保护对象提供详细的分组方法. 然而, 性能实验结果表明即使随机选择同一组的被保护对象, SDES-P 也可以取得优越的结果, 尤其是在大规模 MOSG 中, 未来工作可以进一步完善 MOSG 优先级先验的分组策略.

作者贡献声明: 吴宇鹏主要负责算法实现、实验实现、后期校对以及部分写作; 钱鸿和周爱民负责论文的指导; 王为业和张杨文辉主要参与部分的实验以及后期校对.

参　考　文　献

[1] Wang Zhen, Yuan Yong, An Bo, et al. An overview of security games[J]. Journal of Command and Control, 2015, 1(2): 121−149 (in Chinese)
(王震, 袁勇, 安波, 等. 安全博弈论研究综述[J]. 指挥与控制学报, 2015, 1(2): 121−149)

[2] Cooney S, Wang Kai, Bondi E, et al. Learning to signal in the goldilocks zone: Improving adversary compliance in security games[C]//Proc of the 23rd Joint European Conf on Machine Learning and Knowledge Discovery in Databases. Berlin: Springer, 2019: 725−740

[3] Pita J, Jain M, Ordóñez F, et al. Using game theory for Los Angeles airport security[J]. AI Magazine, 2009, 30(1): 43−57

[4] Tsai J, Rathi S, Kiekintveld C, et al. IRIS—A tool for strategic security allocation in transportation networks[C]//Proc of the 8th Int Conf on Autonomous Agents and Multiagent Systems-Industry Track. Berlin: Springer, 2009: 37−44

[5] Pita J, Tambe M, Kiekintveld C, et al. GUARDS—Innovative application of game theory for national airport security[C]//Proc of the 22nd Int Joint Conf on Artificial Intelligence. San Francisco, CA: Morgan Kaufmann, 2011: 2710−2715

[6] Yin Zhengyu, Jiang A X, Tambe M, et al. TRUSTS: Scheduling randomized patrols for fare inspection in transit systems using game theory[J]. AI Magazine, 2012, 33(4): 59−72

[7] Shieh E A, An Bo, Yang Rong, et al. PROTECT: A deployed game theoretic system to protect the ports of the United States[C]//Proc of the 11th Int Joint Conf on Autonomous Agents and Multiagent

Systems. Berlin: Springer, 2012: 13−20

[8] Li He, Yin Ying, Li Yuan, et al. Large-scale dynamic network community detection by multi-objective evolutionary clustering[J]. Journal of Computer Research and Development, 2019, 56(2): 281−292 (in Chinese)
(李赫, 印莹, 李源, 等. 基于多目标演化聚类的大规模动态网络社区检测[J]. 计算机研究与发展, 2019, 56(2): 281−292)

[9] Brown M, An Bo, Kiekintveld C, et al. Multi-objective optimization for security games[C]//Proc of the 11th Int Joint Conf on Autonomous Agents and Multiagent Systems. Berlin: Springer, 2012: 863−870

[10] Brown M, An Bo, Kiekintveld C, et al. An extended study on multi-objective security games[J]. Autonomous Agents and Multi-Agent Systems, 2014, 28(1): 31−71

[11] Zhou Aimin, Qu Boyang, Li Hui, et al. Multiobjective evolutionary algorithms: A survey of the state of the art[J]. Swarm and Evolutionary Computation, 2011, 1(1): 32−49

[12] He Linjun, Ishibuchi H, Trivedi A, et al. A survey of normalization methods in multiobjective evolutionary algorithms[J]. IEEE Transactions on Evolutionary Computation, 2021, 25(6): 1028−1048

[13] Wu Yupeng, Qian Hong, Qin Rongjun, et al. Scaling multi-objective security games provably via space discretization based evolutionary search[J]. arXiv preprint, arXiv: 2303.15821, 2023

[14] Blank J, Deb K. Pymoo: Multi-objective optimization in Python[J]. IEEE Access, 2020, 8: 89497−89509

[15] Lin Xuan, Ahn M S, Hong D. Designing multi-stage coupled convex programming with data-driven McCormick envelope relaxations for motion planning[C]//Proc of the 38th IEEE Int Conf on Robotics and Automation. Piscataway, NJ: IEEE, 2021: 9957−9963

[16] Kiekintveld C, Jain M, Tsai J, et al. Computing optimal randomized resource allocations for massive security games[C]//Proc of the 8th Int Joint Conf on Autonomous Agents and Multiagent Systems. Berlin: Springer, 2009: 689−696

[17] Conitzer V, Sandholm T. Computing the optimal strategy to commit to[C]//Proc of the 7th ACM Conf on Electronic Commerce. New York: ACM, 2006: 82−90

[18] Paruchuri P, Pearce J P, Marecki J, et al. Playing games for security: An efficient exact algorithm for solving Bayesian Stackelberg games[C]//Proc of the 7th Int Joint Conf on Autonomous Agents and Multiagent Systems. Berlin: Springer, 2008: 895−902

[19] Jiang Wei, Fang Binxing, Tian Zhihong, et al. Research on defense strategies selection based on attack-defense stochastic game model[J]. Journal of Computer Research and Development, 2010, 47(10): 1714−1723 (in Chinese)
(姜伟, 方滨兴, 田志宏, 等. 基于攻防随机博弈模型的防御策略选取研究[J]. 计算机研究与发展, 2010, 47(10): 1714−1723)

[20] Mutzari D, Gan J, Kraus S. Coalition formation in multi-defender security games[C]//Proc of the 35th AAAI Conf on Artificial Intelligence. Menlo Park, CA: AAAI, 2021: 5603−5610

[21] Deb K, Jain H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting Approach, Part I: Solving problems with box constraints[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(4): 577−601

[22] Zhang Qingfu, Li Hui. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(6): 712−731

[23] Fleischer M. The measure of Pareto optima applications to multi-objective metaheuristics[C]//Proc of the 2nd Int Conf on Evolutionary Multi-Criterion Optimization. Berlin: Springer, 2003: 519−533

[24] Jaszkiewicz A. Many-objective Pareto local search[J]. European Journal of Operational Research, 2018, 271(3): 1001−1013

[25] Drugan M M, Thierens D. Stochastic Pareto local search: Pareto neighbourhood exploration and perturbation strategies[J]. Journal of Heuristics, 2012, 18(5): 727−766i

[26] Ishibuchi H, Masuda H, Tanigaki Y, et al. Modified distance calculation in generational distance and inverted generational distance[C]//Proc of the 8th Int Conf on Evolutionary Multi-Criterion Optimization. Berlin: Springer, 2015: 110−125

[27] Zitzler E. Evolutionary algorithms for multiobjective optimization: Methods and applications[D]. Zürich: University of Zürich, 1999

[28] Bosman P A N, Thierens D. The balance between proximity and diversity in multiobjective evolutionary algorithms[J]. IEEE Transactions on Evolutionary Computation, 2003, 7(2): 174−188

[29] Shang Ke, Ishibuchi H, He Linjun, et al. A survey on the hpervolume idicator in eolutionary mltiobjective otimization[J]. IEEE Transactions on Evolutionary Computation, 2021, 25(1): 1−20

[30] Audet C, Bigeon J, Cartier D, et al. Performance indicators in multiobjective optimization[J]. European Journal of Operational Research, 2021, 292(2): 397−422

[31] Chan T M. Klee's measure problem made easy[C]//Proc of the 54th Annual IEEE Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 2013: 410−419

[32] Knowles J D, Corne D. On metrics for comparing nondominated sets[C]//Proc of the 9th Congress on Evolutionary Computation. Piscataway, NJ: IEEE, 2002: 711−716



Wu Yupeng, born in 1999. Master candidate. Student member of CCF. His main research interests include evolutionary optimization and security games.

吴宇鹏, 1999 年生. 硕士研究生. CCF 学生会员. 主要研究方向为演化优化、安全博弈.



Qian Hong, born in 1991. PhD, associate professor. Member of CCF. His main research interests include derivative-free optimization, evolutionary learning, and intelligent education.

钱 鸿, 1991 年生. 博士, 副教授. CCF 会员. 主要研究方向为无梯度优化、演化学习、智能教育.



Wang Weiye, born in 2002. Undergraduate. His main research interest includes evolutionary optimization. (10215102506@stu.ecnu.edu.cn)

王为业, 2002 年生. 本科生. 主要研究方向为演化优化.



Zhang Yangwenhui, born in 1999. Master Candidate. Student member of CCF. His main research interests include evolutionary optimization and learning, and Bayesian optimization. (51215901086@stu.ecnu.edu.cn)

张杨文辉, 1999 年生. 硕士研究生. CCF 学生会员. 主要研究方向为演化优化与学习、贝叶斯优化.



Zhou Aimin, born in 1978. PhD, professor, PhD supervisor. Member of CCF. His main research interests include evolutionary optimization and learning, and intelligent education. (amzhou@cs.ecnu.edu.cn)

周爱民, 1978 年生. 博士, 教授, 博士生导师.CCF 会员. 主要研究方向为演化优化与学习、智能教育.