论文翻译

《增强多智能体系统测试：基于多样性引导探索和自适应关键状态利用方法》

摘要

多智能体系统(MAS)在多机器人控制、智能交通、多玩家游戏等领域取得了显著成功。然而，为保证 MAS 在持续变化和意外场景下的鲁棒性，必须进行全面的测试。现有方法主要聚焦于单智能体系统的测试，无法直接应用于复杂的 MAS 环境；虽然已有部分研究针对 MAS 的对抗攻击，但这些方法主要关注失效场景(Failure Scenario)的发现，而忽略了场景的多样性。本论文提出一种名为MASTest的先进MAS测试框架，强调探索(exploration)（产生多样化行为）与利用(exploitation)（失效检测）之间的平衡。该框架结合个体多样性与团队多样性，设计了自适应扰动机制，对关键状态下的动作进行扰动，以触发更加丰富多样的系统失效场景。实验在Coop Navi与StarCraft II两个常见MAS仿真环境中进行，结果显示，与基准方法相比，MASTest生成的失效场景平均距离分别提升29.55%-103.57%与74.07%-370.00%，发现的失效模式覆盖率分别提高了71.44%-300.00%和50%-500.00%。

1. 引言

近年来，由于多智能体系统（MASs）能够解决涉及多个智能体之间交互的复杂问题[2]，例如多机器人控制[6, 8]、智能交通[32, 39]、智能电网[34]、无人机[5, 36]和多玩家游戏[4, 33]等，因此受到了广泛关注。然而，MAS 的鲁棒性是一个已知的挑战，这主要归因于训练复杂性，如稀疏奖励和信用分配。这些问题使得 MAS 在现实世界中遇到的动态和不可预见场景特别脆弱。因此，在将多智能体系统部署到实际环境之前，它必须经过严格的测试，以确认其鲁棒性，特别是在航空交通管制系统、军事系统和工业自动化等安全关键场景中。

多智能体系统测试的主要目标是尽可能多地发现潜在故障场景，在这些场景中，目标 MAS 做出不符合预期的决定，最终无法完成任务。识别这些不同的故障场景使开发者能够了解弱点以及不同的根本原因，从而进一步增强整体鲁棒性。然而，由于开放环境（如变化的天气条件、不同的地形和其他参与者的行为）的不确定性，可能存在无限多的场景。因此，多智能体系统测试的一个主要挑战是有效地检测这些不同的故障场景。

众多研究[7, 16, 29, 42, 44]已开发用于测试单智能体系统（SAS），如自动驾驶系统。然而，由于多个智能体之间复杂的交互和协作，这些方法难以应用于多智能体系统（MAS）。据我们所知，MAS 测试探索较少。**虽然一些研究已关注通过扰动团队内智能体的观察或行动来对 MAS 进行对抗性攻击[10, 19, 22, 45]测试，但它们主要针对从攻击角度的故障检测，往往忽略了故障的多样性和全面性**。从测试的角度来看，我们的目标是识别不同类型的故障，从而减少对类似故障的冗余分析，并揭示目标 MAS 中更广泛的鲁棒性问题。

本论文旨在开发一种有效的多智能体系统（MAS）测试方法，以生成多样化的失效场景，从而揭示各种鲁棒性问题。**具体而言，本研究面临两个主要挑战**：

1. **衡量场景的多样性，以确保生成足够丰富多样的失效情形**。一种直观的方法是比较两个场景的轨迹。然而，这种方法在复杂的MAS环境中面临巨大挑战。例如轨迹长度的差异、参与智能体数量的变化、智能体之间的相互作用，甚至环境中的细微变化，都可能显著影响轨迹的比较结果，进而影响到对场景多样性的测量。

2. **在生成失效场景时，有效地兼顾场景多样性与失效检测之间的平衡**。如果只注重覆盖多样化的行为，可能会产生大量并不真正引发失效的场景。这体现了测试中典型的"探索"（产生多样化行为）与"利用"（发

现失效）的平衡问题。而现有的许多工作主要是在系统运行之前扰动初始状态，这种方式可能无法有效实现上述平衡，因为在初始阶段进行修改很难精准控制其对MAS在运行期间内部状态的影响。

为了应对上述挑战，本文提出了一种名为 **MASTest** 的高级测试框架，用于多智能体系统（MAS），它通过**多样性引导的探索**与**自适应关键状态利用之间的平衡**，**在测试中触发多样化的失效场景**。

- 针对挑战❶（如何提高场景多样性）：我们开发了一种基于抽象的方法来测量多个智能体的行为，**综合考虑了个体智能体和团队整体的动态特征**。与已有方法类似，我们通过智能体轨迹的抽象序列表示个体行为。而针对团队行为，我们的方法不仅抽象并整合了团队成员之间的状态信息，还通过分析成员间的交互强度来考察他们的协作关系。这种方式将个体和团队的行为同时作为探索测试的依据，即用来生成具有差异化行为的场景。

- 针对挑战❷（如何高效触发失效）：我们的框架设计了一种自适应机制，能够识别出可能引发不同失效的关键状态。然后，我们对这些关键状态进行针对性的动作扰动，以增加触发各种失效的可能性。为了实现这一目标，MASTest维护了一个**状态关键性表(state criticality table)**，表中记录每个状态的扰动潜力，这种潜力基于三个方面来评估：① **多样性增益、② 失效增益、③ 选择频率**。潜力分数高的状态被视为关键状态，将被优先用于扰动。每次测试运行结束后，这个表都会根据最新的多样性反馈和失效反馈动态更新，从而为后续测试提供准确的指导。

我们评估了 MASTest 在两个不同的多智能体环境中的有效性：Coop Navi [25]（一个合作任务）和 StarCraft II [27]（一个竞争任务）。对于每个环境，我们将 MASTest 与三种基线方法进行比较。结果表明，MASTest 在发现多样化故障方面显著优于基线。具体来说，在 Coop Navi 环境中，MASTest 将生成的故障场景的平均距离提高了 29.55%至 103.57%，在 StarCraft II 中，这一增长范围从 74.07%至 370.00%。此外，MASTest 在 Coop Navi 中提高了独特故障模式的覆盖率 71.44%至 300.00%，在 StarCraft II 中提高了 50.00%至 500.00%。此外，消融研究证实，故障反馈、个体多样性和团队多样性对多样化故障场景的发现都做出了显著贡献。我们还展示了这些多样化故障场景在增强系统鲁棒性方面的实际价值。在基于 MASTest 生成的故障场景应用修复后，我们观察到碰撞减少了 45.83%，任务完成率提高了 45.42%。

总之，本文的贡献包括以下几点：

- 据我们所知，本研究是**首个专门针对多智能体系统（MAS）测试的研究**，强调了这种新兴人工智能系统的重要性，以期引起学术界更多的关注。

- 我们**提出了名为MASTest的新颖测试框架**，该框架融合了多样性引导的探索和自适应关键状态利用方法。MASTest能够同时评估智能体的个体多样性和团队多样性，并且自适应地识别关键状态进行动作扰动，从而有效地触发更加丰富多样的失效场景。

- 我们**在两类不同的多智能体系统中进行了充分的实验验证**，并与六种现有方法进行了详细对比，实验结果证实了本文方法的有效性与优越性。

- 我们**开源**了MASTest框架的代码以及完整的实验数据，以便研究社区进行复现和进一步的研究。

2. 背景知识

2.1. 多智能体系统(Multi-Agent System)

多智能体系统由一组相互作用的智能体组成，每个智能体通过通信、合作完成大量单个智能体无法完成的复杂任务。在 MAS 中，复杂任务被划分为多个更小的任务，每个任务分配给不同的智能体。智能体的行为不仅影

响其自身状态，还影响其邻居的状态。这要求每个智能体在决定最佳目标导向行为时，都要考虑其他智能体的行为。
根据任务类型，多智能体系统（MAS）的应用可以分为两类：合作任务（例如，Coop Navi）和竞争任务（例如，StarCraft II）。合作任务是指共同完成给定的目标，如到达目的地，而竞争任务是指共同击败对手。

多智能体系统（MAS）的决策过程通常被建模为马尔可夫博弈。具体来说，一个 m 个智能体的马尔可夫博弈被定义为

$$ \left(\mathcal{N},\left{\mathcal{S}^{i}\right}*{i \in \mathcal{N}},\left{\mathcal{A}^{i}\right}*{i \in \mathcal{N}}, p, \gamma\right) $$

其中：

- $\mathcal{N}$ 是 m 个智能体的集合
- $\mathcal{S}^{i}$ 和 $\mathcal{A}^{i}$ 分别是智能体i的状态(state)空间和动作(action)空间
- $\gamma \in [0, 1)$是折旧(discounting)因子
- $\mathcal{S}=\mathcal{S}^{1} \times \cdots \mathcal{S}^{m}$ 是联合状态(joint state)空间
- $\mathcal{A}=\mathcal{A}^{1} \times \cdots \mathcal{A}^{m}$ 是联合动作(joint action)空间
- 状态转移$p: S \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$由当前状态和联合动作控制，其中$Δ(\mathcal{S})$表示所有联合状态空间 $\mathcal{S}$ 上的概率分布的集合

对于智能体：

- 每个智能体i通过与状态和智能体动作有关的函数获得奖励值$r^{i}：\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$
- 在时间t，智能体i根据策略$\pi^{i}: \mathcal{S}^{i} \rightarrow \Delta\left(\mathcal{A}^{i}\right)$选择其动作$a_{t}^{i}$。
- 智能体的联合策略(joint policy)是$\pi=\prod_{i \in \mathcal{N}} \pi^{i}: \mathcal{S} \rightarrow \Delta(\mathcal{A})$
- 每个智能体i的目标是最大化其自身的期望累计回报$R_{i}=\sum_{t=0}^{T} \gamma^{t} r_{i}^{t}$。其中$\gamma$是折旧因子，$T$是时间范围。

2.2. 场景(Scenario)相关解释

(1) Scenario 场景

*场*景是一系列随时间变化（时序）的*场*景画面(Scene)组成的序列，其中每个*场*景画面是*环*境(Envrionmnet)在某一时刻的快照，包括静态和动态对象。

因此，一个场景可以通过一系列可配置的静态和动态属性来描述。

- **静态属性**设置场景中的静态对象，例如地图、目的地、障碍物等。
- **动态属性**定义动态对象的状态（例如位置、方向等）、轨迹和行为（例如移动、加速等），例如非玩家角色（NPC）的运动和行为。

为了**发现更加多样化且关键的场景，我们可以通过调整环境中这些静态和动态对象的可配置属性来实现**，即通过设置一系列轨迹点（waypoints）来控制动态对象的运动轨迹，从而构造出我们所需要的测试场景。

(2) Scenario observation 场景观察

*场景观察*是指的是在仿真环境（Simulation environment）中，每个时间步（time step）记录的所有对象的全局状态序列，包括目标智能体（Target agents）、非玩家角色（NPC）和其他环境要素。

此外，我们还可以单独获取目标多智能体系统（MAS）的团队轨迹（Team trajectory），从而更专注地分析目标系统自身的行为。
具体来说，目标 MAS 的团队轨迹τ由每个智能体的个人轨迹τ_i组成，其中i是智能体的 ID。τ_i包含目标智能体的局部状态，如其位置、速度和健康值(health value)（该指标是由任务特定的，即在不同 MAS 中都定义有所不同）。给定一个k维状态空间R^k，个人轨迹τ_i可以表示为$(s_0^i,\cdots,s_k^i)$，其中s_j^i指的是第j维状态。

(3) Failure Scenario 失效场景

失效场景（Failure Scenario） 指的是目标多智能体系统（MAS）做出不良决策，最终导致任务失效的场景。

在不同的测试环境中，**失效（Failure）** 有着不同的具体定义，例如：

- 在 **Coop Navi**（自动驾驶）环境中，失效的定义可能是目标多智能体系统是否与障碍物发生碰撞，以及系统内的所有智能体是否成功抵达它们各自的目的地。
- 而在 **StarCraft II**（对战游戏）环境中，失效可能定义为目标多智能体系统被对手击败，或者与对手战成平局。

此外，用户也可以根据需求，在MASTest框架中灵活配置其他形式的失效定义，只要这些不良的行为能导致明显较低的回报（Reward）即可。

其它部分的 GPT 理解

3. 方法

> 以下是按照原文结构，对该论文方法部分（**3 Approach**）各节的重点内容、要点进行系统性的讲解与总结：
>
> ---
>
> ## 3 方法（Approach）
>
> 本节重点描述本文所提出的MAS测试方法 **MASTest**，解释了如何通过**多样性引导的探索（Diversity-guided exploration）** 和**自适应关键状态利用（Adaptive critical state exploitation）** 达到测试目标。
>
> ---
>
> ### 3.1 概述（Overview）
>
> **图1（Figure 1）** 给出了本文方法MASTest的整体框架：
>
> - 测试时，MASTest会在运行过程中修改环境中非玩家角色（NPCs，其他动态对象）的动作，以测试目标MAS的鲁棒性。
>
> 具体流程为：
>
> - 在每个时间步（time step）记录一个具体状态（Concrete State (s)）。
> - 如果状态为结束状态（End State），则：

    - 记录完整的轨迹（Trajectory），判断任务是否成功完成；
    - 若未成功，则标记为失败场景（Failure Scenario）。
- 从该轨迹中收集反馈信息，包括：
    - 个体多样性（Individual Diversity）
    - 团队多样性（Team Diversity）
    - 失败反馈（Failure Feedback，包含失败程度与失败频率）
    - 状态选择频率（Selection Frequency）
- 反馈信息用于更新一个**状态关键性表（State Criticality Table）**：
    - 关键性表记录状态的重要程度，以指导后续动作的扰动（Perturbation）。
- 如果当前状态非结束状态：
    - 根据关键性表判断该状态是否关键（Critical）：
        - 若关键，则对该状态执行动作扰动；
        - 若非关键，继续原始动作，不作扰动。
- 以上过程持续进行，直到测试预算（如测试时长）用尽。

3.2 状态抽象（State Abstraction）

由于MAS环境中状态极其复杂且数量无限，因此很难直接比较具体状态。作者采用了**网格化抽象（Grid-based abstraction）**方法，简化状态空间，使得状态更容易被比较和处理。

具体过程为：

- 对于MAS中的一个智能体，其具体轨迹为：[\tau_{\alpha} : (s_{\alpha}^0, s_{\alpha}^1, \dots, s_{\alpha}^k)]
- 每个状态通常是高维向量，作者将每个维度（dimension）划分为 (N) 个相等的区间。
- 每个具体状态 (s) 被映射到一个抽象状态（abstract state）：[\hat{s} = g(s)] 其中 (g) 是网格化的抽象函数（grid-based abstraction function）。
- 因此，类似的具体状态被归为同一个抽象状态，从而显著降低了状态空间的复杂度。

图2（Figure 2）提供了状态抽象的直观解释：

- 二维的状态空间被划分成网格，每个状态位于某个网格中。
- 两条不同的轨迹在抽象后可表示为网格序列：
    - 如轨迹1表示为 `{..., 15, 11, 7, 13, 18, ...}`
    - 轨迹2表示为 `{..., 5, 1, 7, 18, 14, ...}`

3.3 多样性反馈（Diversity Feedback）

为了生成多样化的失败场景，需要对MAS行为进行度量。作者从**个体多样性**和**团队多样性**两个角度进行测量：

3.3.1 个体多样性（Individual Diversity）

- 个体多样性通过比较当前场景中每个智能体的抽象轨迹和历史抽象轨迹来衡量：[d_{\tau_{\alpha}} = \min_{\hat{\tau}'{\alpha} \in T{\alpha}} dist(\hat{\tau}{\alpha}, \hat{\tau}'{\alpha})]

- 轨迹距离 (dist) 用**归一化汉明距离（Normalized Hamming Distance）**计算： [dist(X, X') = \frac{Hamming(X, X') + |len(X)-len(X')|}{\max(len(X), len(X'))}]
- 最终，个体多样性定义为所有智能体的平均多样性。

3.3.2 团队多样性（Team Diversity）

团队多样性评估MAS整体行为差异，作者提出四步方法：

- **时间步采样（Time Step Sample）**
  - 为避免逐步分析每个状态，作者从轨迹中选取代表性的时间步进行分析；
  - 采样策略为选择奖励值（reward）最高的时刻附近的一些时间步。
- **基于图的抽象（Graph-based abstraction）**
  - 将MAS团队抽象为无向图，每个顶点代表一个智能体，顶点属性为智能体的状态信息（位置、速度等），边权值为智能体之间的互动强度（如距离）。
- **图嵌入（Graph embedding）**
  - 将抽象的图嵌入为向量，方便快速比较不同团队行为的差异。
  - 公式为： [h_v^{k+1} \leftarrow (Sum(h_v^k, Mean(\frac{1}{w(u,v)} h_u^k), u\in N(v)))]
- **团队多样性计算**
  - 使用图瓦瑟斯坦距离（Graph Wasserstein distance）计算当前图与历史图之间的差异。

3.4 失败反馈（Failure Feedback）

- 作者定义失败程度（Failure Degree）为任务完成度（Task Completion）与成本消耗（Cost Consumption）之和： [f = f_{goal} + f_{cost}]
- 例如：
  - Coop Navi中，任务完成度为智能体与目标地点的距离，成本为碰撞损失。
  - StarCraft II中，任务完成度为对手剩余生命值，成本为自身战斗力损失。

3.5 状态关键性表初始化与更新（State Criticality Table Initialization & Update）

- 初始时，每个状态的关键性统一设为1。
- 测试后动态更新每个扰动状态的关键性，基于三个因素：
  1. 多样性增益（Diverse Failures Gain）：新场景带来的多样性与失败程度提升。
  2. 失败频率（Failure Frequency）：触发失败的频率高低。
  3. 状态选择频率（Selection Frequency）：被选择扰动的次数。
- 计算关键性更新： [\Delta C_s = \lambda_1\cdot\Delta C_{div} + \lambda_2\cdot\Delta C_{ff} + \lambda_3\cdot\Delta C_{sf}]

- 根据关键性选择是否进行扰动，高关键性的状态被优先选择。

3.6 动作扰动（Action Perturbation）

- 对关键状态实施动作扰动，本文提出三种扰动方式（图4）：
  - 基因突变（Gene Mutation）
  - 两点交叉（Two-point Crossover）
  - 基因乱序（Shuffle）
- 根据动作维度不同选用合适的扰动方式，实现对环境中动态对象（NPC）的动作修改，从而诱发MAS失败。

方法整体总结：

本文提出的MASTest框架通过创新的状态抽象、多样性反馈与关键状态自适应利用机制，实现了高效、全面地发现多智能体系统的多样化失败场景，体现了"探索与利用"的平衡理念，具有显著的有效性与实用性。

4. 实验设计

以下是对论文中第4节（**Experiment Design 实验设计**）的详细总结，遵照原文结构与逻辑进行系统性讲解，尤其注意突出图表及公式的解释与关键内容：

---

4 实验设计（Experiment Design）

作者通过精心设计的实验，验证了提出的**MASTest方法**的有效性。实验设计主要围绕以下几个问题展开：

- **RQ1（有效性）**：与其他方法相比，MASTest能否有效地发现更多样化的失败场景？
- **RQ2（消融实验）**：MASTest中各部分（失败反馈、个体多样性、团队多样性）如何贡献于整体性能？
- **RQ3（实用性）**：通过MASTest发现的失败场景能否有效地帮助修复目标MAS？

4.1 研究问题（Research Questions）

实验设计围绕以下三个研究问题进行：

- **RQ1（Effectiveness 有效性）**
- **RQ2（Ablation Study 消融实验）**
- **RQ3（Usefulness 实用性）**

4.2 目标模型与环境（Target Models and Environments）

实验涵盖两种常见的MAS任务类型，分别是合作型（cooperative）和竞争型（competitive）任务，如图5所示。

图5说明（Figure 5）：

- 图5 (a)：**Coop Navi（合作型任务）**
  - 目标MAS（绿色小圆点）需协作达到目的地（大绿色圆圈），避免与障碍（黑色物体）或NPC（粉色物体）碰撞。
- 图5 (b)：**StarCraft II（竞争型任务）**
  - 目标MAS（蓝色方阵）需与对手智能体（红色方阵）对抗，目标是战胜对手。

具体实验环境与模型：

- **合作型任务**：采用开源的MPE环境（Coop Navi任务），使用OpenAI官方提供的MADDPG算法训练的MAS模型。
- **竞争型任务**：采用StarCraft II游戏环境，使用DeepMind官方提供的训练模型（off-policy actor-critic方法）。

4.3 基准方法与评价指标（Baselines and Evaluation Metrics）

4.3.1 基准方法（Baselines）

由于任务类型不同，作者选用两套不同的基准方法作为对照：

- **合作型任务**（Coop Navi）：
  - Random（随机扰动法）：随机选择NPC扰动其动作。
  - MDPFuzz：基于初始状态扰动的模糊测试框架。
  - GMT：当前SOTA测试框架，基于新颖性引导生成多样场景。
- **竞争型任务**（StarCraft II）：
  - Random（随机扰动法）：随机选择NPC扰动其动作。
  - QMIX：深度多智能体强化学习算法，用于训练对抗智能体。
  - Wuji：竞争型任务的SOTA测试框架，生成多样的对抗智能体。

4.3.2 评价指标（Evaluation Metrics）

实验评价使用以下指标：

- **状态覆盖率（%Coverage）**：
  - 覆盖的状态空间单元数与整体状态空间单元数的比值。
  - 状态空间每个维度划分为5份，形成多个单元。
- **失败场景平均距离（#Distance）**：
  - 公式（1）计算不同失败场景间的平均距离，评估多样性。
- **失败场景数量（#Failure）与比例（%Failure）**：
  - 测试中发现失败场景的数量及其占比。

- **失败模式覆盖率（%Pattern）**：
    - 人工分析确认的失败模式类型数量占所有可能失败模式类型的比例。

4.4 实验设置（Experimental Setup）

- 所有方法在相同的2小时测试预算内完成测试，避免随机误差，重复三次，取平均性能。
- 超参数选择（通过初步实验确定）：
    - 权重因子 (\lambda_1, \lambda_2, \lambda_3) 为 0.5, 0.5, 1。
    - 团队多样性窗口大小设为5，时间步采样间隔A设为2。
- 消融实验（RQ2）：
    - **w/o Diversity Feedback**: 去除多样性反馈，仅使用失败反馈。
    - **w/o Failure Feedback**: 去除失败反馈，仅使用多样性反馈。
    - **w/o Team Diversity**: 仅保留个体多样性与失败反馈。
    - **w/o Individual Diversity**: 仅保留团队多样性与失败反馈。
- 实用性实验（RQ3）：
    - 对Coop Navi的目标MAS进行模型修复实验，以评估发现的失败场景是否帮助系统提升性能。
    - StarCraft II由于目标模型为黑盒，无法直接修复，但可反馈给开发者。

图表、公式重点说明：

- **图5** 明确展示了两种任务的差异：
    - 图(a)展示了合作任务中智能体和NPC之间的关系；
    - 图(b)展示了竞争任务中双方智能体之间的关系。
- 评价指标部分使用到的距离公式（Normalized Hamming Distance）：[dist(X, X') = \frac{Hamming(X, X') + |len(X)-len(X')|}{\max(len(X), len(X'))}] 用于评估失败场景间的差异程度，这是实验评价多样性的关键公式。

总结实验设计的关键点：

- 设计了详尽且清晰的实验对比设置，包括了合作型与竞争型两种不同的MAS任务类型，以保证结论的普适性。
- 使用了充分、多元的评价指标，包括状态覆盖率、失败数量、失败比例、平均距离及人工模式分析，确保实验评估全面。
- 明确设定了消融实验来评估方法中各要素的重要性与贡献。
- 图表（如图5）清晰地展示了实验任务类型与环境特征的不同，为理解实验目标和方法提供了直观支撑。

通过上述实验设计，作者清晰地验证了MASTest方法在生成多样化失败场景方面的有效性、各部分设计的必要性，以及方法在实际场景修复中的实用性。

5. 实验结果与分析

以下是对论文第5节（**Results and Analysis 结果与分析**）的详细解读与总结，分别针对作者提出的三个研究问题（RQ1、RQ2、RQ3）进行具体讲解和分析，尤其是重点解释该部分的图表：

---

5 结果与分析（Results and Analysis）

这一部分旨在分析和验证前述实验设计中提出的三个研究问题（RQ1~RQ3）的实验结果。

5.1 RQ1：MASTest的有效性（Effectiveness of MASTest）

作者通过自动化评估与人工评估两种方式，验证MASTest生成失败场景的有效性和多样性。

5.1.1 自动化评估（Automated Evaluation）

表1（Table 1） 给出了自动化评估的实验结果，指标包括：

- 状态覆盖率 (%Coverage)
- 失败场景平均距离 (#Distance)
- 失败场景数量 (#Failure)
- 失败比例 (%Failure)
- 失败模式覆盖率 (%Pattern)

表1结果说明：

- **合作型任务**（Cooperative Task）：
  - MASTest的状态覆盖率、失败场景距离、失败数量、失败比例和模式覆盖率均明显高于Random、MDPFuzz和GMT三种基准方法。
  - 特别是模式覆盖率（%Pattern）达到100%，远超其他方法。
- **竞争型任务**（Competitive Task）：
  - MASTest在覆盖率和失败场景多样性指标（距离、模式覆盖）上表现最佳。
  - QMIX虽然有最高的失败场景数量，但它产生的失败场景同质性高，多样性指标较差。
  - MASTest在整体性能指标表现最优，体现了多样性和有效性之间的良好平衡。

5.1.2 人工评估（Manual Evaluation）

作者人工分析了MASTest发现的失败场景，确定了具体失败模式（图6、表2）：

- **表2（Table 2）** 显示了人工分析总结出的具体失败模式：
  - **Coop Navi**：两类失败模式（碰撞F1、未抵达目的地F2）共计12种具体模式。
  - **StarCraft II**：两类失败模式（F3、F4）共计6种具体模式。
- **图6（Figure 6）** 展示了合作型任务的具体失败场景示例：
  - 例如F1-1代表多个智能体与NPC围绕目的地相互追逐，导致碰撞；
  - F2-1表示智能体错误决策而未到达目的地等。

- **表2结果说明**：
  - MASTest涵盖了所有人工总结的失败模式，展现了最佳的失败模式覆盖能力，体现出高效、多样的失败发现能力。
- **图7（Figure 7）** 提供了定性分析的具体场景示例：
  - (a) 显示MASTest能够在关键状态（Critical State）进行有效的动作扰动；
  - (b) 表明其他方法未能识别关键状态，导致扰动无效。

5.2 RQ2：消融实验（Ablation Study）

通过消融实验（表3），作者验证了MASTest各个关键模块对整体性能的贡献：

表3（Table 3） 显示消融实验的指标变化：

- **w/o Failure Feedback**（去掉失败反馈）：
  - 失败场景数量减少，多样性增加但状态覆盖率显著下降，说明失败反馈对识别有效失败场景至关重要。
- **w/o Diversity Feedback**（去掉多样性反馈）：
  - 失败数量增加但多样性大幅下降，覆盖率降低，表明多样性反馈对场景差异化生成至关重要。
- **w/o Team Diversity**（去掉团队多样性）与 **w/o Individual Diversity**（去掉个体多样性）：
  - 二者对整体性能都有影响，但去掉团队多样性的影响更大，说明团队多样性更能代表整体行为差异。
- 完整MASTest性能最佳，表明所有模块均对整体性能具有积极贡献，团队多样性的重要性尤其突出。

5.3 RQ3：MASTest的实用性（Usefulness of MASTest）

通过模型修复实验（表4），验证了MASTest产生的失败场景对实际MAS模型的提升作用：

表4（Table 4） 展示了模型修复前后的性能变化：

- 修复前，模型的碰撞次数较多，任务完成率较低。
- 采用MASTest发现的失败场景进行模型修复后：
  - 碰撞次数大幅下降（从386.4降低到209.3）；
  - 任务完成率显著提高（从51.3%提升到74.6%）。
- 对比其他基准方法：
  - MASTest的失败场景能提供更多有效信息，有效提高MAS模型性能。

对各表、图的重点说明解释：

- **表1（Table 1）**：
  直观比较了MASTest与基准方法在两个任务环境中的性能，多样性指标体现明显优势。
- **图6（Figure 6）**：
  展示具体失败模式的直观图示，帮助读者清晰理解MASTest发现的多样化失败场景。
- **表2（Table 2）**：
  人工分析提供的更深入理解，多样失败模式数量直观体现MASTest优势。
- **图7（Figure 7）**：
  通过具体实例（关键状态选择的有效性）直观地说明MASTest成功原因。
- **表3（Table 3）消融实验**：
  体现各模块对整体性能的贡献与作用，直观展示重要性排序（Failure Feedback和Team Diversity尤为重要）。
- **表4（Table 4）实用性实验**：
  明确地证实MASTest在实际应用中的价值，其产生的失败场景对系统修复具备明确优势。

总体小结：

- **RQ1的结果表明**：
  - MASTest在发现多样化失败场景方面明显优于现有方法。
- **RQ2的结果表明**：
  - MASTest方法中的所有模块（失败反馈、多样性反馈）都是重要的，团队多样性尤其关键。
- **RQ3的结果表明**：
  - 通过MASTest发现的失败场景，能有效提升目标MAS系统的实际性能，体现了方法的实用性与应用价值。

论文通过上述严谨的实验设计与详细的分析，充分验证了提出的MASTest方法在MAS系统鲁棒性测试中的有效性、必要性与实用性。这些详细的图表和定性分析帮助读者全面理解方法及其效果。