基于强化学习和质量多样性算法的多样策略生成

杨可芸 12012438 **曾宪清** 12012338 **陈驿来** 12013025
指导老师：刘佳琳

November 2022

目录

摘要

本篇报告主要通过结合强化学习和传统质量多样性搜索，复现 [3] 中的 PGA-Map-Elites 算法，并对其测试以验证其相对之前地传统算法更高效地产生多样性策略的能力。随后通过一系列不同的传统强化学习算法的思路的学习，对 PGA-Map-Elites 原有框架提出优化和改进方案。

关键词

强化学习, 质量多样性搜索, PGA-Map-Elites

1 问题背景

伴随计算机人工智能的发展，如今深度强化学习 (Deep Reinforcement Learning,DRL) 在游戏 [4]、蛋白质结构预测 [1]、无人驾驶等领域不断取得突破。同时，深度强化学习相关的研究也产生了新的细分，例如多目标强化学习、约束强化学习、质量多样性强化学习。其中，质量多样性强化学习，关注质量多样性优化（Quality Diversity Optimisation,QDO），训练具有多样行为（按照行为特征空间分布）的高质量（根据环境交互设置的奖励）策略集合，在游戏人工智能、机器人控制等领域有极大的应用潜力。本项目尝试融合质量多样性优化算法和传统深度学习算法，提出通用、高效、可拓展的质量多样性深度强化学习算法框架，最终实现具有行为多样性的策略集合生成。（参考 [6]）。

2 问题描述

2.1 强化学习问题描述

传统的强化学习（Reinforcement Learning,RL）的主要过程是构建最大化马尔可夫决策（Markov Decision Process, MDP）获得更高奖励的策略模型。进行学习，并做出决策的对象称为智能体。以如下图 1 为例，智能体（Agent）与环境（Environment）（所有与之相互作用的事物）中持续交互，智能体选择行为，环境做出对应的响应，并使得智能体呈现新的状态。环境在给定状态和动作后会转移到下一个状态并产生收益（reward）。这也是智能体在不断尝试动作的过程中想要最大化的目标。如图 1.

传统 RL 算法（DRL 算法）寻找最大的期望（折扣）奖励目标。为了能够不失一般性地考虑有限长度时间决策，我们让 R_t 表示智能体在环境中第 t 个时刻得到的

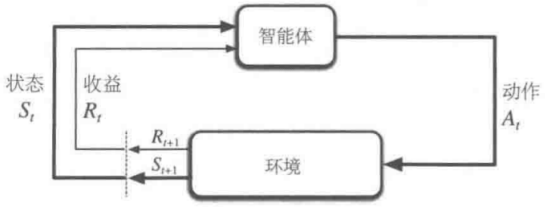

图 1: 图来自 *[5]Fig.3.1*。马尔可夫决策过程中的“智能体”与“环境”的交互

奖励，让 h 表示决策时长，两种目标（有无折扣）分别可以定义如下：

$$J = \frac{1}{h}\mathbb{E}\left[\sum_{t=0}^{h-1} R_t\right], \tag{1}$$

$$J = \mathbb{E}\left[\sum_{t=0}^{h-1} \gamma^t R_t\right], \gamma \in [0,1], \tag{2}$$

其中 γ 为折扣率，它决定了未来收益的现值。
本小节内容参考 [5][6]。

2.2 质量多样性优化问题描述

质量多样性优化最早在 [2] 中被提出，文章中的 MAP-Elites 算法旨在找到特征空间中每一个细胞所对应的质量函数值最大的解。令 $\mathcal{X}$ 为解空间，考虑连续的质量函数 $f(\cdot): \mathcal{X} \mapsto \mathbb{R}$，以及一系列特征函数 $b_i(\cdot): \mathcal{X} \mapsto \mathbb{B}_i$，其中 $\mathbb{B}_i$ 为第 i 个特征的特征空间，QDO 的目标是找到满足以下条件的解集 X^*：

$$X^* = \left\{ \arg\max_x [f(x), \text{ s.t. } b_i(x) = b_i] \,\middle|\, \forall b_i \in \mathbb{B}_i \right\} \tag{3}$$

本小节参考 [6]。

2.3 质量多样性强化学习

质量多样性强化学习是强化学习的重要子课题。在强化学习的设定下，智能体在与环境交互的过程中会得到一些奖励，而强化学习的目标则是最大化累积奖励，即智能体在环境中要得到最高的奖励分数。时至今日，在强化学习领域已经取得了一系列突破性的进展。例如，智能体可以在很多复杂的游戏中取得超越人类的表现，例如自适应控制，通过强化学习算法，使机械臂能够根据不同的材料和场景自适应地搭建积木桥。 所以质量多样性强化学习，允许程序自定义一系列行为特征函数（比如在场景中，AI 如何决定任意特定动作频率，）。我们此时将解集 $\mathcal{X}$ 替换为环境中所有可能的策略，质量指标 $f(\cdot)$ 替换为公式 (1) 或 (2)，即为质量多样性强化学习问题。

3 项目目标

质量多样性强化学习算法文章 [3] 提出了 PGA-Map-Elites 算法，它将基于 actor-critic 架构的面向连续动作空间的深度强化学习算法 TD3 与质量多样性算法 Map-Elites 相结合，成功地阐明了搜索空间中性能与关心的维度之间的关系，并在整个可能的行为空间内找到了高性能的解决方案。

本学期的最终目标为复现文章 [3]，并结合已有研究，对当前算法框架提出改进，更换其他 DRL 算法或根据实现细节改进现有算法中的一些不足 [6]。计划选择 Mujoco 机器人控制模拟环境作为测试所需的强化学习环境。

4 项目规划

4.1 初期规划

在项目初期阶段，我们计划首先阅读文献 [2][3]，了解质量多样性算法的产生背景及伪代码过程和相关应用。其次，通过学习分别基于价值的和基于策略的两种算法以入门强化学习，理解其核心思想。除此之外，搜索相关可用于强化学习算法测试的环境库并尝试配置文章 [3] 中的测试环境 QDgym。

4.2 中期规划

在项目中期阶段，我们计划继续沿着既定算法学习路线复现 DQN, DDPG，TD3 算法，并调研、配置项目内容中提及的强化学习环境以对算法进行调参测试。同时初步复现 PGA-MAP-Elites 算法，将运行结果与论文原代码结果进行比较，提出后期改进方向。

4.3 后期规划

在完成 PGA-MAP-Elites 算法的复现基础上，根据具体实现细节，进行优化，测试。可能尝试的优化方向：经验回放区的更新方式，优化神经网络的层次构建，添加噪声的方式，损失函数的计算等。同时尝试替换其他 DRL 算法或者测试环境，进行对比测试，优化，以期最终得出相对全面的结果，实现基于强化学习和质量多样性算法的多样策略生成目标。

5 项目分工

1) 曾宪清：学习并复现 DQN 算法，复现 PGA-MAP-Elites 算法中 TD3 算法部分。
2) 杨可芸：复现 PGA-MAP-Elites 算法中 MAP-Elites 算法部分。
3) 陈驿来：配置运行的服务器环境，测试并记录结果。

6 背景研究

6.1 MAP-Elites 算法 [2]

6.1.1 背景和概述

MAP-Elites 是一个高效的搜索算法。在该算法被提出之前，一些传统搜索算法具有如下缺点：只关注在搜索空间中找到一个或一组高质量的解决方案；无法解决一些需要“黑盒”优化的问题，因为对于这类问题我们难以获取决定性能的评估函数，因而不能使用需要计算函数的优化方法，例如梯度上升/下降；此外它们可能会因为依赖于随机启发式算法（即对好的解决方案进行随机更改，来尝试获得更好的解决方案）而导致局部最优，但却无法获得全局最佳的解决方案。相比之下，MAP-Elites 算法能够返回一组高性能且多样化的解决方案；阐明性能与解决方案中所关注的维度之间的关系；改进优化性能，因为它探索了更多的特征空间以避免陷入局部最优，从而找到不同的且更好的性能峰值。

如图4，MAP-Elites 算法在高维空间中搜索，以在低维特征空间中的每个点处找到性能最高的解决方案。我们将这种类型的算法称为“照明算法（Illumination algorithm）”，因为它照亮了特征空间中每个区域的性能潜力，包括性能变化和感兴趣特征之间的权衡。

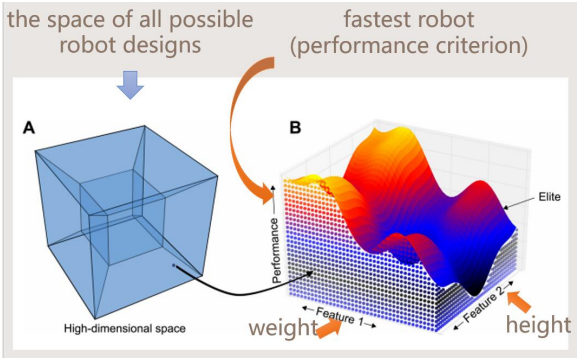

图 2: 修改自 *[2]Fig.1*。
照亮了特征空间每个区域的适应度潜力，包括性能和感兴趣特征之间的权衡

6.1.2 MAP-Elites 算法的细节

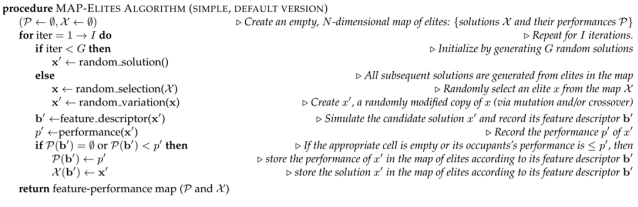

图 3: *[3]* 中的 *Fig.2*, 展示 *Map-Elites* 的 *Pseudo code*

Map-Elites 算法的伪代码如图3所示。首先，用户选择一个计算解决方案 x 的性能度量 $f(x)$。例如，如果寻找机器人的形态，性能指标可以是机器人的速度。其次，用户选择 N 个感兴趣的变化维度，组成一个名为 b 的向量，定义一个感兴趣的特征空间。比如说，我们对于机器人的形态，感兴趣的一个维度可能是机器人有多高，一个维度可能是它的重量，另一个可能是每米移动它的能量消耗，等等。给定一个特定的离散化程度划分特征空间，Map-Elites 将寻找 N 维特征空间中每个细胞的最高性能解（我们将特征空间中的每个单元格称为一个细胞）。例如，Map-Elites 将寻找以高、重、高效为特征的机器人中速度最快的机器人；以高，重，低效为特征的速度最快的机器人；以高，轻，高效为特征的速度最快的机器人等等。搜索在搜索空间中进行，搜索空间是 x 的所有可能值的空间，其中 x 是对候选解的描述。在我们的示例中，搜索空间包含所有可能的机器人形态的描述。我们称 x 描述符为基因组；函数 $f(x)$ 称为适应度函数，返回每个 x 的性能；行为函数 $b(x)$ 则是对于每个 x，确定 x 在每个 N 个特征维度中的值。换句话说，$b(x)$ 返回 b_x，它是一个描述 x 的特征的 n 维向量。在我们的例子中，$b(x)$ 的第一个维度是机器人的高度，第二个维度是它的重量，第三个维度是它的每米移动的能量消耗，等等。特征向量的一些元素可以直接在表型中进行测量（如身高、体重），但其他的元素（如能量消耗）需要在模拟环境中测量。

伪代码的过程：首先随机生成 G 个基因组，并确定每个基因组的性能和特征。这些基因组被放入它们的特征对应的细胞中（如果多个基因组映射到同一个细胞，则保留每个细胞中表现最高的一个基因组）。接下来的每次迭代中，特征空间中的一个细胞被随机选择，该细胞中的基因组通过突变和交叉产生一个后代。如果该基因组变异后对应的细胞是空的，或者后代的性能比当前的细胞对应的基因组更高，那么后代基因组将被放置在细胞中。

6.1.3 *Map-Elites* 算法的总结

与传统优化算法相比，Map-Elites 和其他照明算法映射整个特征空间，告诉用户特征和性能之间的各种权衡; 另外，Map-Elites 学习了自然界产生多样性的特点，因为它同时奖励了众多不同领域中表现最好的解决方案，最后，它在用户定义的特征空间的每个点上找到性能最好的解，是帮助我们了解复杂搜索空间的有价值的新工具。

6.2 强化学习算法

强化学习的最终目的是学习“做什么来使数值化之后的收益信号最大化”，学习体通过不断试错，尝试，去发现怎样可以产生更大的收益。“‘试错’和‘延迟收益’—-是强化学习两个最重要最显著的特征”(此句来自《强化学习》[5]) 强化学习最重要的四个要素为：策略、收益、价值函数、对环境建立的模型。

6.2.1 基于“价值”与基于“策略”

价值函数是所有强化学习算法不可缺少的要素，价值函数是指确定状态（或确定状态以及确定动作的二元组）的函数，目的是评估确定状态（和动作）下有多“好”，是否“好”则取决于收益的期望值。选取期望值最高的动作来作为下一步行为，则认为是基于“价值”(value-based) 的强化学习算法。但同时，学习体对未来的收益的期望取决于它具体选择了什么动作，所以有时价值函数与特定的行为方式（选择动作的概率）相关，此时我们称之为基于“策略”(policy-based)，“策略”可以认为是从状态到每个动作的选择概率之间的映射。[5]

6.2.2 *Q-learning* 算法

Q-learning 是强化学习算法中 value-based 的算法。Q 即为 $Q(s,a)$ 就是在某一时刻的 s 状态下 ($s \in S$)，采取 $a(a \in A)$ 动作的未来奖励期望（公式4，5），环境会根据 agent 的动作反馈相应的回报 reward r。算法的主要思想就是将 State 与 Action 构建成一张 Q-table 来存储 Q 值（初始化为 0），然后根据 Q 值来选取能够获得最大的收益的动作，获取 reward 后根据公式 (6) 来更新 Q-table 的值。对于动作的选取，还可结合 $\epsilon-greedy$ 策略，即 ϵ 概率选择 Q 值最大动作，$(1-\epsilon)$ 概率选择随机动作。引入 ϵ 可更好地权衡开发与探索。

$$Q(s_t,a_t) = E(G_t|s_t,a_t) \tag{4}$$

其中, G_t 为当前 t 时刻对未来的累计折扣奖励，

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \tag{5}$$

$$Q^{new}(s,a) = Q^{old}(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q^{old}(s,a)] \tag{6}$$

来自https://blog.csdn.net/qq_30615903/article/details/80739243?ops_request_misc

6.2.3 *Policy Gradient* 算法

当状态空间变得高维时，q-learning 方法则变得不适用，因为 Q-table 无法存储过多的状态，此时就需要

借助神经网络来处理。Policy Gradient 算法是 policy-based 的算法，它使用神经网络输入当前状态，输出在这个状态下采取每个动作的概率。网络使用反向传播算法，设置一个误差函数，通过梯度下降来使损失最小。但对于强化学习来说，我们不知道动作的正确与否，只能通过奖励值来判断这个动作的相对好坏。我们把基于上面的想法，产生非常简单的想法：如果一个动作得到的期望折扣奖励大，那么我们就使其出现的概率增加，如果一个动作得到的 reward 少，我们就使其出现的概率减小。我们定义 $J(\theta)$ 为采用 θ 为参数的神经网络得到的期望奖励的函数，我们希望的是通过改进参数 θ 来使得 $J(\theta)$ 值最大。改进 θ 的公式如7。

$$\theta_{t+1} = \theta_t + \alpha\widehat{\nabla J(\theta_t)} \tag{7}$$

来自https://blog.csdn.net/ygp12345/article/details/109009311

6.2.4 Actor-Critic

Actor 和 Critic 是两个不同的神经网络。Actor 用于预测行为的概率，Critic 是预测在这个状态下的价值。其中 Actor 使用策略函数，负责生成动作 (Action) 并和环境交互。而 Critic 可以预测每一个状态的价值，评估 Actor 的表现，并指导 Actor 下一阶段的动作。Actor-Critic 结合了 Policy Gradient (Actor) 和 Function Approximation (Critic) 的方法，Actor 基于概率选行为，Critic 估计每一个状态的价值，Actor 选取行为达到下一个状态时，Critic 用这个状态的价值减去下个状态的价值，得到 TD-error，TD-error 如果是正的，下个动作就要加大更新，如果是负的，就减小 actor 的更新幅度，Actor 的目标是最小化 TD-error。PGA-Map-Elite 算法将 Actor-Critic 算法 TD3 与 Map-Elite 相结合。实现了搜索空间的强大照明，在整个可能的行为范围内找到高性能的解决方案。

7 复现 PGA-MAP-Elites[3]

7.1 算法概述

PGA-MAP-Elites 算法是 MAP-Elites 算法的改进版本。他将策略梯度 (Policy Gradient) 方法的搜索能力和数据效率与遗传算法 (Genetic Algorithms) 的探索能力相结合，并加入了深度强化学习算法 TD3，实现更快寻找最优解的目标。PGA-MAP-Elites 的伪代码如下。π 代表 actor，是一个输入为当前状态 (state)，输出为每种动作 (action) 对应概率的神经网络。我们的目标为在每一个特征空间的单元格内找到性能最优的 π。首先我们初始化 TD3 算法需要的神经网络并将特征空间划分单元格。While 循环中，我们每次迭代生成 b 个 actor，一共需要生成 I 个。其中，前 G 个 actor 由随机函数生成。对于之后的每 b 个 actor，其中 1 个由 *traincritic* 函数 (见 6.2) 生成，旨在找到表现较好的 actor 并将其保留；另外 b-1 个 actor 由 variation 函数生成 (见 6.3)，旨在通过杂交 (crossover) 和梯度优化方法将当前表现较好的 actor 结合或稍加改动，以获得表现更好的 actor。每次生成 b 个 actor 后查找每个 actor 对应的单元格，若单元格中没有 actor，则将当前 actor 加入该单元格；若已有的 actor 的性能劣于当前，则将当前 actor 替换已有 actor。经过多次迭代，大多数单元格中都会存在一个表现较好的 actor，达到质量多样性的效果。

Algorithm 2 PGA-MAP-Elites algorithm

function PGA-MAP-ELITES
  create empty archive (X, P)
  Initialize critic networks $Q_{\theta 1}, Q_{\theta 2}, \pi_{\phi c}$
  Initialize target networks $Q'_{\theta 1}, Q'_{\theta 2}, \leftarrow Q_{\theta 1}, Q_{\theta 2},$
  Initialize target network $\pi'_{\phi c} \leftarrow \pi_{\phi c}$
  Initialize replay buffer $\mathcal{B}$
  $i = 0$
  while $i < I$ **do**
    if $i < G$ **then**
      $\pi_{\widehat{\phi 1}}, ..., \pi_{\widehat{\phi b}} = random - solutions(b)$
    else
      $\pi_{\widehat{\phi_1}} = TRAIN - CRITIC(Q_{\theta_i}, Q_{\theta'_i}, \pi_{\phi_c}, \pi_{\phi'_c}, \mathcal{B})$
      $\pi_{\widehat{\phi_2}}, ... = VARIATION(b-1, \mathcal{X}, Q_{\theta_1}, \mathcal{B})$
    end if
    $ADD - TO - ARCHIVE(\pi_{\widehat{\phi 1}}, ..., \pi_{\widehat{\phi b}}, \mathcal{X}, \mathcal{P}, \mathcal{B})$
    $i+ = b$
  end while
  $Return\ archive(\mathcal{X}, \mathcal{P})$
end function
function $ADD - TO - ARCHIVE(Controller - List, \mathcal{X}, \mathcal{P}, \mathcal{B})$
  for π_θ in Controller-List **do**
    $(p, b, transitions) \leftarrow$evaluate$(\pi_\theta)$
    $add\_to\_replay\_buffer(transition, B)$
    $c \leftarrow get\_cell\_index(b)$
    if $P(c) = empty$ or $P(c) < p$ **then**
      $P(c) \leftarrow p, X(c) \leftarrow \pi_\theta$
    end if
  end for
end function

7.2 训练过程

PGA-MAP-Elites 中应用的 TD3 是一种 Actor-Critic 算法，它结合了 Policy Gradient(Actor)（见 5.2.3）和 Function Approximation(Critic) 的方法。Actor 基于概率选择行为，Critic 基于 Actor 的行为评判行为的得分，即适应度，Actor 再根据 Critic 的评分修改选择行为的概率分布。因此，PGA-MAP-Elites 需要训练两个 Critic 神经网络 $Q_{\theta_1}, Q_{\theta_2}$ 来近似地推导出动作-价值函数, 算法详见 Algorithm 2。

此处选用以最大动作值预测动作的 greedy actor 与 critic 一起训练，以计算 critic 更新目标值，更新方法与 TD3 中使用的 actor 相同。这里的 actor 表示为 π_{θ_c}，是一个参数为 θ_c 的神经网络，同时也是 MAP-Elite 循环中演化的个体。然而，与 TD3 不同的是，更新后的 actor 并不直接与环境交互。在 PGA-MAP-Elite 的每次迭代中，都会从存档中采样一批 actor，以两种方式（遗传算法和策略梯度）进行变异，并在重新评估后再次选择性添加到存档中。在评估过程中与环境交互产生的经验被收集到经验回放区 B，该缓存区有最大容量限制，并且当达到上限后采取先入先出的方式更新。在 n_crit 次训练 critic 的每次循环中，随机从经验回放区抽取 N 份交互数据，计算 loss $L(\theta_1, \theta_2) = (y - Q_{\theta_1}(s_t, a_t))^2 + (y - Q_{\theta_2}(s_t, a_t))^2$ 的平均值，两个 critic 的神经网络参数都将朝着使 loss 下降的方向更新以减小动作价值函数的预估与真实目标值的差距。目标值 y 的计算与 TD3 中相同 (公式8)，按照贝尔曼方程（公式4）的形式并取两个 critic 输出的最小值作为预测，下一步采取的动作由 greedy actor 决定并加上一定范围内的噪声。

$$y = r(s_t, a_t) + \gamma \min_{i=1,2} Q_{\theta'_i}(s_{t+1}, \pi_{\phi'_c}(s_{t+1}) + \epsilon) \quad (8)$$

在随机环境中，PG 变异可能倾向于收敛到窄峰上的局部最优解，而噪声的添加恰能增加探索的机会以达到加强行为鲁棒性的目的。并且算法中还采取了 TD3 中延迟更新的方式，在每 d 次训练 critic 神经网络后才进行网络参数的更新，actor 将朝着选择最大化 critic 预估值的动作的方向更新，而目标网络将根据相应新旧网络按照一定比例更新。

7.3 变异算子

PGA-MAP-Elites 用到了两种针对 actor 的变异方法，分别是遗传算法（Genetic Algorithms）和策略梯度（Policy Gradient）。

遗传算法：通过随机算法选择两个 actor，将其神经网络参数如下式进行杂交（crossover），从而产生新的子代。

Algorithm 2 TRAIN_MAP-Elites Critic Training

function TRAIN_CRITIC($Q_{\theta_1}, Q_{\theta_2}, Q_{\theta'_1}, Q_{\theta'_2}, \pi_{\phi_c}, \pi_{\phi'_c}, B$)
  while $t = 1 < n\_crit$ **do** ▷ Training Loop: n_crit iterations
    Sample N transitions $(s_t, a_t, r(s_t, a_t), s_{t+1})$ from B
    ϵ clip$(N(0, \sigma_p), -c, c)$ ▷ Sample smoothing noise
    $y = r(s_t, a_t) + \gamma \min_{i=1,2} Q_{\theta'_i}(s_{t+1}, \pi_{\phi'_c}(s_{t+1}) + \epsilon)$
    $\theta_i \leftarrow \text{argmin}_{\theta_i} \frac{1}{N} \sum (y - Q_{\theta_i}(s, a))^2$ ▷ Update Critics
    if $t \mod d$ **then**
      Update greedy controller(actor) using gradient descent
      $\nabla_\phi J(\phi) = \frac{1}{N} \sum \nabla_\phi \pi_{\phi_c}(s_t) \nabla_a Q_{\theta_1}(s_t, a)\big|_{a=\pi_{\theta_c}(s_t)}$
      $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$ ▷ Update critic targets
      $\phi'_i \leftarrow \tau\phi_c + (1-\tau)\phi'_c$ ▷ Update actor targets
    end if
  end while
  return controller π_{θ_c}
end function

$$\phi_{child} = \phi_1 + \sigma_1 N(0,1) + \sigma_2(\phi_2 - \phi_1)N(0,1)$$

梯度下降：通过随机算法选择一个 actor，采用 Adam 梯度下降法更新 actor。

遗传变异使 actor 有机会遍布整个搜索空间，从而避免 actor 的适应度陷入局部最优的情况。而梯度下降法在性能较优的 actor 周围寻找性能更优的解，有效提升寻找最优解的速度。

7.4 训练环境 QDgym

7.4.1 评估任务

本次项目目前使用 QDgym 作为训练环境来测试 PGA-MAP-Elites, 测试环境设定为模拟一个机器人，目标是发现它可以行走的所有可能方式，同时最大限度地平衡速度和能量消耗。在使用 PyBullet 设定的任务中，通过在模拟寿命（1000 步）内累积的前进进度来定义适应度，并对模拟的每一时间步进行能耗惩罚和生存奖励。由于 PGA-MAP-Elites 算法在任一状态下，每一个

action 所取得的进展取决于训练 critic 的转变状态。我们将状态定义为当前重心高度、x、y 和 z 速度、横摇、俯仰和偏航角以及机器人关节的相对位置等。因此，状态会有很高的维度个数例如 15，26。动作 action 是施加到机器人关节以控制它的连续值扭矩，因此每次设置为 3 维数据。使用的 BD 是机器人的每只脚在一个行为中与地面接触的时间长度。同时，任务是随机的，因为初始关节位置是从高斯分布中采样的。

7.4.2 评估指标

1) QD-score: 存档中所有解决方案的适应度总和。由于适应度可能是负的，因此在计算 QD 分数时，所有行为的适应度都被给定任务的所有比较算法中发现的最低适应度行为所抵消。这避免了对发现其他解决方案的算法的惩罚。
2) Coverage: 存档中的解决方案总数。当存档中的每个单元找到解决方案时，即可实现全覆盖。
3) Max Fitness：存档中最适合的整体解决方案。
4) Fitness Difference：为了评估每个算法在随机环境中的鲁棒性，考虑了单个评估与当前最大适应度解决方案的 10 次评估的平均适应度之间的适应度差异。每个算法只允许对向存档添加解决方案进行一次评估，这需要算法在随机环境中有效的鲁棒性。我们使用具有最大适应度的解决方案来评估鲁棒性，因为它可能对噪声最敏感。

7.4.3 实验结果

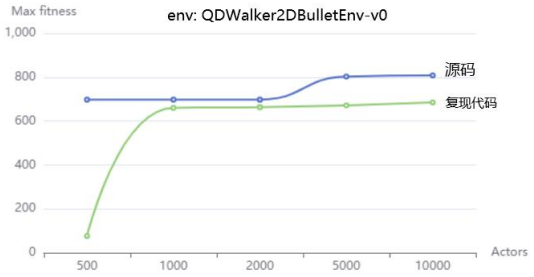

图 4:

以上是我们复现后的代码与源码在 QDWalker2DBulletEnv 环境下的表现对比，横坐标是迭代的总的 actor 数目，纵坐标是已迭代的 actor 中的最大适应度。可以看到，在 0-1000 代，源码比复现后的代码更快的达到了一个较高的适应度；在 2000-10000 代，源码的适应度以一定的速度稳步上升，而复现后的代码上升速度较为平缓。

7.5 PGA-MAP-Elites 相比于 MAP-Elites 的优势

1. Map-Elites 使用简单开环系统，即系统的输出取决于输入，而输入却独立于系统输出。相比之下，PGA-MAP-Elites 引入基于梯度的变异算子，结合反向传播，利用从批评性神经网络获得的梯度估计值来调整参数，实现快速找到性能更高的解。

2. Map-Elites 依赖于遗传算法，因而 MAP-Elites 只能适用于低维度问题（优化参数的数量通常保持在 100 个以下），相比之下，PGA-MAP-Elites 算法由于寻找解的效率更高，可用于解决更高维度的问题。同时，对于一些需求复杂且需要精确控制变量的问题，Deep-Neural-Network（DNN）需要加入巨量的参数来解决，而 PGA-MAP-Elites 算法可以相对轻松地完成相同工作，并且能够在随机的环境中，保持较好的学习鲁棒性。

8 小结

1、项目初期（1-7 周）：

1) 对于质量多样性优化，我们首先研读了提出 Map-ELites 算法的文章 [2]，对质量多样性算法的目的和内容有了初步认识。其次，我们阅读了项目目标需要复现的算法的文章 [3]，了解 PGA-Map-Elites 算法的大致框架，并尝试配置论文中提到的的开源环境 QDgym(https://github.com/ollenilsson19/QDgym)。
2) 对于强化学习，我们首先学习并复现了的 Q-learning、Sarsa 算法（代码见 Github 仓库https://github.com/tsukii7/Quality-diversity-with-reinforcement-learning），并调整参数进行测试。

2、项目中期（8-12 周）：

1) 沿着既定的算法学习路线，学习复现 DQN、DDPG，并完成 PGA-Map-Elites 算法中所需的 TD3 算法的复现，对算法进行调参测试。
2) 结合 TD3 与 MAP-Elites 最终初步复现 “Policy gradient assisted MAP-Elites”，将其应用于 QDGym 环境进行训练模型测试并与原论文 [3] 的实验结果进行比较。

参考文献

[1] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K.Tunyasuvu nakool, R. Bates, A. Žídek, and A. Potapenko et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[2] J.-B. Mouret and J. Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

[3] O. Nilsson and A. Cully. Policy gradient assisted map-elites. *Genetic and Evolutionary Computation Conference*, page 866–875, 2021.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrit twieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[5] R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction (2nd edition). *MIT press*, 2018.

[6] 王子祺. 基于强化学习和质量多样性算法的多样策略生成. 创新实践课题介绍, 2022.