Adversarial Coevolutionary Illumination with Generational Adversarial MAP-Elites

Timothée Anne[1], Noah Syrkis[1], Meriem Elhosni[2], Florian Turati[2], Franck Legendre[2], Alain Jaquier[2], and Sebastian Risi[1]

[1]IT University of Copenhagen, Copenhagen, Denmark
[2]armasuisse Science+Technology, Thun, Switzerland
Corresponding author: sebr@itu.dk

Abstract

Unlike traditional optimization algorithms focusing on finding a single optimal solution, Quality-Diversity (QD) algorithms illuminate a search space by finding high-performing solutions that cover a specified behavior space. However, tackling adversarial problems is more challenging due to the behavioral interdependence between opposing sides. Most applications of QD algorithms to these problems evolve only one side, thus reducing illumination coverage. In this paper, we propose a new QD algorithm, Generational Adversarial MAP-Elites (GAME), which coevolves solutions by alternating sides through a sequence of generations. Combining GAME with vision embedding models enables the algorithm to directly work from videos of behaviors instead of handcrafted descriptors. Some key findings are that (1) emerging evolutionary dynamics sometimes resemble an arms race, (2) starting each generation from scratch increases open-endedness, and (3) keeping neutral mutations preserves stepping stones that seem necessary to reach the highest performance. In conclusion, the results demonstrate that GAME can successfully illuminate an adversarial multi-agent game, opening up interesting future directions in understanding the emergence of open-ended coevolution.

Code and videos available at: `https://github.com/Timothee-ANNE/GAME`

Introduction

Quality Diversity (QD) (Pugh et al., 2016) is a branch of evolutionary algorithms that seeks to illuminate a problem, i.e., discovering a diverse set of high-quality solutions that maximize a fitness function while covering a behavior space. It has demonstrated success across various domains, including robotics (Cully et al., 2015), procedural content generation (Gravina et al., 2019), chemical synthesis (Jiang et al., 2022), and aeronautics (Brevault and Balesdent, 2024).

One may wish to apply such methods to adversarial problems, which arise across many domains, for example, in military conflicts (Schelling, 1980), economic regulation (Baldwin et al., 2011), machine learning (Chakraborty et al., 2018), and cybersecurity (Li et al., 2021).

Several such efforts have already been undertaken. For instance, in the turn-based strategy card game, Hearthstone, Fontaine et al. (2019) illuminate the deck space, while Fontaine et al. (2020) evolve a set of neural network controllers with a fixed deck to find different strategies. More recently, Samvelyan et al. (2024a) investigate strategic weaknesses in a Deep Reinforcement Learning (DRL) agent during a simulated football game, while Samvelyan et al. (2024b) search for different safety-violating prompts for a Large Language Model (LLM). A limitation of these approaches is that they only optimize one side of the adversarial problem. This is problematic, as adversarial systems often lead to an arms race (Dawkins and Krebs, 1979), which requires improvements on both sides to emerge.

Optimizing both sides simultaneously to broaden the illumination aligns with the artificial life challenge of fostering open-ended evolution through adversarial coevolution (Bedau et al., 2000; Dorin and Stepney, 2024). Some works explore this avenue: Costa et al. (2020) coevolve Generative Adversarial Networks (GANs) using QD, leading to improved models; Wang et al. (2019, 2020) coevolve bipedal walkers and their environments to foster open-ended evolution, creating an automatic curriculum; and Dharna et al. (2024) introduce a QD algorithm that generates Python scripts to control both sides of a car chase game. However, these works rely on problem-specific methods.

This paper introduces a new QD algorithm for adversarial problems, which we call Generational Adversarial MAP-Elites (GAME). Combined with a Vision Embedding Model (VEM) such as CLIP (Radford et al., 2021), GAME can work from videos of behaviors instead of requiring handcrafted behavior descriptors. GAME handles the embedding space's high dimensionality by using growing unstructured archives (Vassiliades et al., 2017). To evaluate the approach, we apply it to a multi-agent adversarial game and show through ablation studies that it effectively illuminates the space of strategies. The paper's main contributions are:

- a new adversarial coevolution QD algorithm, GAME;
- the use of a VEM as the behavior space in a MAP-Elites algorithm with a growing archive;
- an empirical evaluation of GAME on a multi-agent adversarial game, including comparisons with ablations.

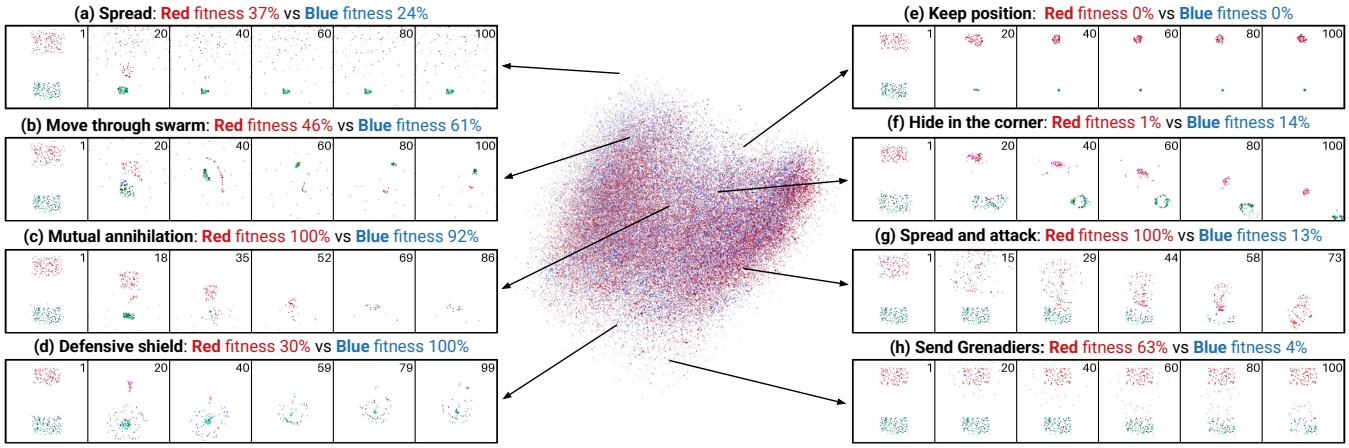

Figure 1: **GAME's illumination of a multi-agent adversarial game.** The point cloud is a 2D PCA projection (22% and 12% explained variance) of the intergenerational tournament between elites found for one run of GAME across 20 generations with 100 000 evaluations per generation. We display timed snapshots of eight duels exhibiting different behaviors. We also indicate the fitness of both sides, represented as the percentage of depleted health of the opposing side. Videos of these duels and supplementary data are available in the repository `https://github.com/Timothee-ANNE/GAME`.

Related-work

Artificial Life, Open-endedness, and Coevolution

GAME aims to create an open-ended adversarial coevolution of solutions that broaden illumination. Creating an open-ended artificial environment that continually presents new, interesting challenges, and in turn, fosters the evolution of novel and meaningful solutions to those challenges is one of the main quests in the artificial life community (Bedau et al., 2000; Dorin and Stepney, 2024).

A way to move toward open-endedness is through coevolution in an arms race dynamic (Dawkins and Krebs, 1979). A key difficulty lies in finding a balance that avoids the collapse of one side or the emergence of a stable equilibrium where both sides cease to innovate (Ficici and Pollack, 1998). Moran and Pollack (2019) demonstrate that a blend of cooperation and competition can lead to greater complexity growth. Similarly, Harrington and Pollack (2019) show that greater policy evolvability supports a longer-lasting arms race and increased complexity.

A form of non-strictly adversarial problems is the coevolution of an environment and the agent interacting with it. The objective is for the environment to evolve in a way that presents new challenges that are neither too easy nor too difficult, effectively creating a curriculum for the agent by offering a sequence of meaningful stepping stones. Brant and Stanley (2017) coevolve mazes and maze-solving agents using minimal criterion coevolution, i.e., any agent that solves a maze and any maze that is solved at least once can reproduce. Their approach demonstrates that no handcrafted fitness function or behavior descriptor is required.

Building on this idea, POET (Wang et al., 2019) generates an open-ended sequence of environments for training bipedal walkers. It maintains a population of active environment–controller pairs. New environments are generated from recently active ones that have shown sufficient progress where the task is neither easy nor difficult for the associated controller. Each controller is independently optimized using an evolutionary strategy. POET also facilitates transfer between pairs, attempting to apply controllers trained in one environment to others. A key insight is that some resulting controllers could not be obtained through direct optimization on the final environment, suggesting that POET enables the discovery of critical stepping stones.

Enhanced-POET (Wang et al., 2020) introduces a new measure of novelty for environments, called Performance of All Transferred Agents – Environment Comparison (PATA-EC). This metric requires only fitness scores, eliminating the need for handcrafted behavior descriptors. It is based on a round-robin tournament of agents, comparing their performance rankings across environments. The underlying idea is that a novel environment should yield a different ranking.

In contrast to those works, GAME is not designed for the particular problem of evolving controllers and environments, but applies to any adversarial problem.

Self-Play

Self-play is a closely related field, which creates a learning curriculum by making the agent play against itself or an earlier version of itself. Alpha-Star (Vinyals et al., 2019) achieves grandmaster level in the Real-Time Strategy (RTS) player-versus-player game StarCraft 2. It uses a league of agents that serve as training opponents for the main agents. Using a league prevents strategy collapse, improves robustness, and helps avoid local minima and forgetting.

Baker et al. (2019) propose an RL framework for adversarial self-play in a 3D multi-agent hide-and-seek game. They revealed how it allowed the emergence of six strategy

phases as both sides found new ways to counter the other's high-performing behaviors. The goal of self-play is, however, the learning of one robust and high-performing policy, not the illumination of all possible strategies.

Quality-Diversity for adversarial problems

Two popular QD algorithms are Novelty Search with Local Competition (NSLC) (Lehman and Stanley, 2011) and Multidimensional Archive of Phenotypic Elites (MAP-Elites) (Mouret and Clune, 2015). MAP-Elites works by constructing an archive of high-performing solutions, known as elites, which are organized into different cells that discretize the behavior space. MAP-Elites generates a new candidate solution at each iteration using evolutionary variation operators, evaluates it, and if it either exhibits a novel behavior (i.e., fills a previously empty behavior cell) or has a greater fitness than the current elite in its corresponding cell, it becomes the elite of that cell.

Multi-Task MAP-Elites (MT-ME) (Mouret and Maguire, 2020) is a variant of MAP-Elites that addresses multi-task problems, i.e., the simultaneous optimization of multiple fitness functions. The idea is that solving these tasks simultaneously can improve sampling efficiency, as related tasks may share similar solutions. Building on this, Multi-task Multi-behavior MAP-Elites (MTMB-ME) (Anne and Mouret, 2023) extends MT-ME by evolving diverse high-performing solutions for each task, rather than just a single solution. In this paper, GAME uses MTMB-ME at each generation to evolve solutions that compete against a set of solutions from previous generations (i.e., the tasks).

QD algorithms have been used to illuminate adversarial problems. For example, Fontaine et al. (2019) propose MAP-Elites with sliding boundaries to build different decks in Hearthstone, using as opponents first a fixed set of decks and then the best decks found in the first phase to explore the possible counters. Using the six best found decks as opponents, Fontaine et al. (2020) use Covariance Matrix Adaptation MAP-Elites to evolve a diverse set of neural network controllers and find different strategies when playing with one deck. Steckel and Schrum (2021) use MAP-Elites to illuminate the space for generated levels by different GANs in the Lone Runner game. Wan et al. (2024) apply QD principles to adversarial imitation learning to train agents on several behaviors simultaneously and show that it can potentially improve any inverse RL method. Samvelyan et al. (2024a) apply MAP-Elites to explore adversarial scenarios for a pre-trained DRL agent in a multi-agent game environment (Google Research Football), uncovering strategic weaknesses in the agent's behavior. Samvelyan et al. (2024b) use MT-ME to find a set of diverse adversarial safety attacks on an LLM, using an LLM for the variation operator and for evaluating the fitness of the attack. These methods only consider optimizing one side of the adversarial problem, significantly limiting the illumination.

Few QD methods currently try to coevolve both sides. Costa et al. (2020) use NSLC to coevolve GANs (i.e., the generators and descriptors), showing that it improves diversity and discovers better models. Closer to our work, Dharna et al. (2024) propose a self-play QD algorithm to coevolve adversarial controllers as Python code for an asymmetric adversarial game between pursuer and evader agents. They also use the embedding space of a foundation model for diversity, but use an LLM's embedding space from the Python controller script, which relates to genotypic diversity, not behavioral, as in this paper. Their work also uses an unstructured archive to handle the high-dimensional embedding space. However, they employ an NSLC-type unstructured archive to define novelty, which requires a more problem-specific tuning of the novelty threshold parameter compared to the growing archives used in GAME.

Using a VEM for behavior space

One limitation of MAP-Elites is the need to define a behavior space to determine what is different. Automated Search for Artificial Life (ASAL) (Kumar et al., 2024) have shown that a VEM, such as CLIP (Radford et al., 2021), can be used to explore the space of artificial life substrates. Inspired by them, we propose, to our knowledge, the first use of a VEM as a behavior space for a MAP-Elites algorithm.

One difficulty with such a space is its high dimensionality (e.g., $2\,560$ dimensions in this paper). The original MAP-Elites uses a grid to divide the behavior space, which is not scalable to high dimensions (as the number of cells grows exponentially with the dimensions). One solution is using Centroidal Voronoi Tessellations (CVT) (Vassiliades et al., 2016) to divide the behavior space uniformly. However, in very high-dimensional spaces, one cannot easily predict which proportion of the space will be covered by the studied system, making it difficult to tune the selection pressure. If too many cells are available to fill, the pressure will be too low, reducing quality, and if there are not enough cells, the pressure will be too high, reducing diversity. One solution is to learn a low-dimensional representation using an auto-encoder (Cully, 2019), which requires extra computation.

Our intuition is that we can grow the archive to cover the relevant part of the behavior space instead of defining it beforehand. This is similar to how NSLC defines novelty with a growing archive of solutions. This is called an unstructured archive (Vassiliades et al., 2017). Unlike NSLC, which uses a distance parameter threshold to determine that a new solution should be added to the archive, unstructured archives have a predefined maximal number of cells and only update the position and boundaries of those cells. This can be done by updating the centroids of CVT when the replacement of an old centroid by a new one would increase the minimal distance between all centroids, thus growing the archive.

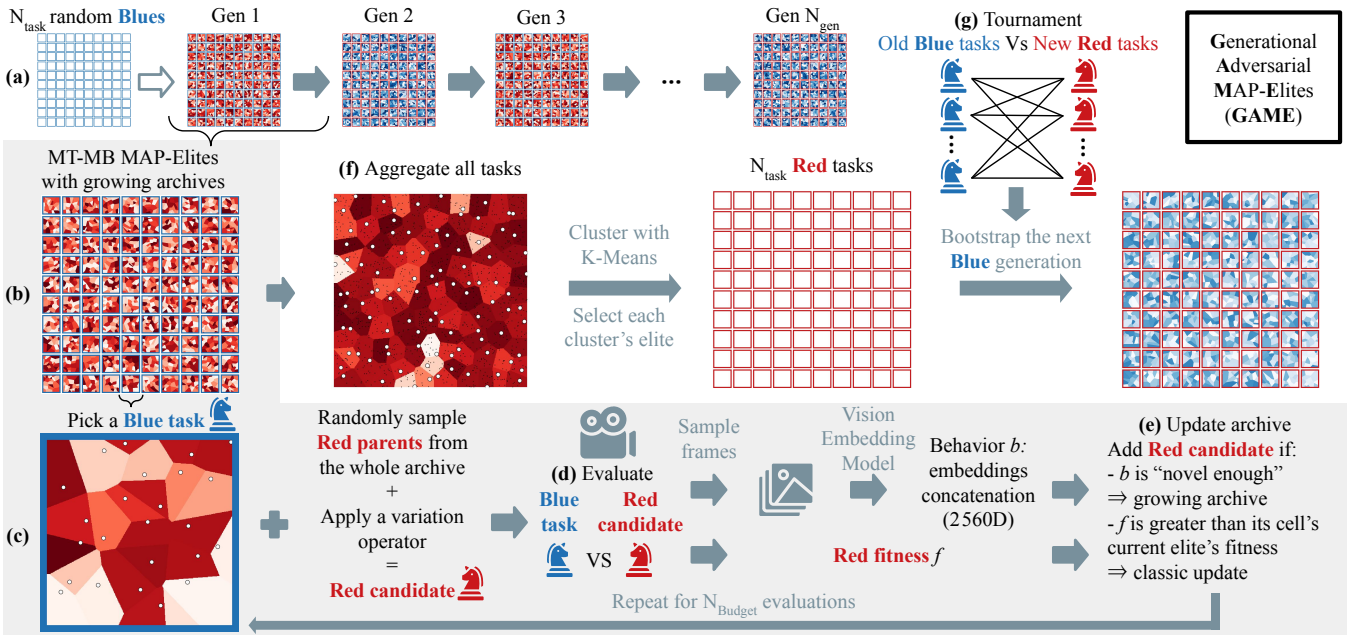

Figure 2: **GAME** is an adversarial coevolution QD algorithm that iterates the illumination of one side of an adversarial problem using MTMB-ME (Anne and Mouret, 2023), switching sides at each generation to promote an arms race dynamics that expands the illumination. One key feature of GAME is the ability to use a VEM as a behavior space.

Generational Adversarial MAP-Elites

GAME is a QD algorithm that aims to illuminate both sides (Blue and Red) of an adversarial problem using coevolution to nurture an open-ended discovery of solutions (Fig. 2). It is divided into two levels: (1) Intergenerational illumination of solutions, switching from one side to the other at each generation, Alg. 1 (**a**, **f**, and **g**); and (2) Intragenerational execution of MTMB-ME at each generation, using one side as tasks and the other as solutions, Alg. 2 (**b**, **c**, **d**, and **e**).

It proceeds as follows (the paper's values in parentheses):

(a) randomly sample N_{task} (100) Blue solutions as tasks;
For N_{gen} (20) **generations:**
  (b) initialize a multi-task multi-behavior growing archive with N_{cells} (25) for each task;
  For N_{budget} (100 000) **evaluations:**
    (c) pick a $task$ at random and use a variation operator on randomly picked elites from the whole archive to generate a candidate solution s;
    (d) evaluate s against the $task$, collect the video and fitness f, subsample the video (five uniformly spaced frames) to get $frames$, get the embedding (512D) from each using a VEM (CLIP), and concatenate to get the behavior descriptor b (2 560D);
    (e) update $task$'s archive (Alg. 3): (1) if the new behavior b is farther from all the current cell's centroids than the closest pair of centroids, it is added to the archive and one of the two centroids from the pair is removed; (2) else if the new fitness f is greater than the corresponding cell's fitness then the new solution becomes the elite of its cell;
  (f) aggregate all the elites, cluster them into N_{tasks} clusters using K-Means, and select the elite of each cluster to form the next generation of tasks;
  (g) bootstrap the next generation with a tournament between the new and previous tasks.

Algorithm 1 GAME
Inputs: Red search space S_{Red}, Blue search space S_{Blue}
Parameters: N_{gen}, N_{task}

$Tasks \leftarrow$ Sample N_{task} random Blue solutions ▷ **(a)**
$Generations = \{0 : Tasks\}$
$\mathcal{B} = \emptyset$ ▷ For storing bootstrapping evaluations
for $gen\_id = 1$ in N_{gen} **do**
  S $\leftarrow S_{Red}$ if gen_id is odd else S_{Blue}
  $\mathcal{A} \leftarrow$ MTMB-ME$(Tasks, S, \mathcal{B})$ ▷ **(b-e) - Alg. 2**
  $Behaviors \leftarrow$ Aggregate the archive's elites ▷ **(f)**
  Clusters $\leftarrow$ K-means$(Behaviors, k = N_{task})$
  $Tasks \leftarrow \{\text{Elite}(cluster)\}_{cluster \in Clusters}$
  $Generations[gen\_id] = Tasks$
  $\mathcal{B} \leftarrow Tasks$ versus $Generations[gen\_id - 1]$ ▷ **(g)**
return $Generations$

Using a VEM for behavior space When using a VEM, GAME follows ASAL by using cosine similarity to compute the distance between two behaviors (which are 2 560D):

$$\text{dist}(b, b') = 1 - \frac{b \cdot b'}{\|b\|_2 \|b'\|_2}.$$

The reason is that Euclidean distance loses meaning as the number of dimensions increases. This doesn't impact the growing archive, as it uses the relative comparison of distances between behaviors and not their actual positions.

Algorithm 2 MTMB-ME with growing archive and a VEM

Inputs: $Tasks$, search space S, bootstrap set $\mathcal{B}$

Parameters: Number of evaluations N_{budget}, Number of initial random search N_{init}, evaluation function Evaluate, variation operator Variation, VEM

$\mathcal{A} \leftarrow$ Initialize N_{task} growing archives ▷ **(b)**
for $(task, s, f, b)$ `in` $\mathcal{B}$ **do** ▷ **(g)** - Bootstrapping
  $\mathcal{A} \leftarrow \text{Update}(\mathcal{A}[task], s, f, b)$ ▷ **Alg. 3**
for $i = 1$ `to` N_{budget} **do** ▷ Main loop
  $task \leftarrow$ Select a task at random from $Tasks$ ▷ **(c)**
  if $\mathcal{A}$ has less than N_{init} elites **then**
    $s \leftarrow$ Sample a random solution from S
  else
    $s \leftarrow \text{Variation}(\mathcal{A})$
  $f, video \leftarrow \text{Evaluate}(s, task)$ ▷ **(d)**
  $frames \leftarrow$ Subsample the $video$
  $b \leftarrow$ Apply the VEM on each frame
  $\mathcal{A} \leftarrow \text{Update}(\mathcal{A}[task], s, f, b)$ ▷ **(e) - Alg. 3**
return Archives

Growing an unstructured archive One point to consider when growing an archive is that when a new behavior takes over a cell, it can "steal" the elites of neighboring cells as it changes the whole CVT, which creates "holes". A simple solution to fill those "holes" is to reinstate the centroid as the elite of its cell, as it can never be "stolen." It only requires storing an additional solution per cell. More elaborate heuristics can be used, e.g., storing all the past elites and checking with the current CVT. However, we found that this doesn't significantly improve performance for most problems while increasing computation and memory space.

Case study: Multi-agent adversarial game

To evaluate GAME, we applied it to a competitive RTS game research environment, Parabellum (Anne et al., 2025). Competitive games are inherently adversarial, making them a direct target for illuminating adversarial problems. In addition, RTS games employ many agents in a top-down view of the map, and characterizing the multi-agent behavior is not trivial. Parabellum also shares similarities with ALife simulation, with many agents moving and interacting in a 2D map, allowing us to study using a VEM as a behavior space.

Parabellum Multi-Agent Game

Parabellum is a multi-agent game where two sides, Blue and Red, try to eliminate the other, i.e., the fitness of each side is the sum of the depleted health of the opposing side's units. Each unit can only see other units within its sight range (15 m with the map being 100 m wide). At each time step, each unit can either move, stand, attack an enemy in reach, or heal an ally in reach, given its local observation (position, type, side, and health of all the units in sight). It is implemented in JAX (Bradbury et al., 2018), a Python library for just-in-time compilation and vectorization. It permits fast parallelization of the units' behaviors using a dedicated GPU. Parabellum contains stochasticity, but we use the same seed for all encounters to make comparisons fair.

Algorithm 3 Growing unstructured archive update

Inputs: archive A, solution s, fitness f, behavior b

Parameters: max archive size N_{cells}, distance function dist

$(C, E, E_{backup}) = A$ ▷ Centroids, Elites and Backup Elites
if size$(C) < N_{cells}$ **then** ▷ Add a new cell
  $i = \text{size}(E)$
  $C_i = b$
  $E[i] \leftarrow (s, f, b)$
  $E_{backup}[i] \leftarrow (s, f, b)$
else ▷ Check behavior and fitness
  $distances = \{\text{dist}(C_i, C_j)\}_{0 \le i < j < N_{cells}}$
  $d_{min} = \min(distances)$
  $d = \min\{dist(b, C_i)\}_{0 \le i < N_{cells}}$
  $c_{id} = \text{find\_cell}(C, b)$ ▷ Closest centroid's index
  if $d > d_{min}$ **then** ▷ New enough behavior = growth
    $j, k \leftarrow \text{argmin}(distances)$
    $d_j \leftarrow \min\{dist(C_j, C_i)\}_{0 \le i \ne b < N_{cells}}$
    $d_k \leftarrow \min\{dist(C_k, C_i)\}_{0 \le i \ne a < N_{cells}}$
    $k \leftarrow j$ if $d_j < d_k$ else k
    $C_k \leftarrow b$
    $E[k] \leftarrow (s, f, b)$
    $E_{backup}[k] \leftarrow (s, f, b)$
    for $i = 0$ to N_{cells} **do** ▷ Check and repair holes
      if $\text{find\_cell}(C, E[i].b) \ne i$ **then**
        $E[i] \leftarrow E_{backup}[i]$
  else if f > E[c_{id}].f **then** ▷ Better fitness
    $E[c_{id}] \leftarrow (s, f, b)$
return (C, E, E_{backup})

We defined five types of units (each with a different color for visualization): **spearman**: slow and high-health close combat unit; **archer**: low-health long-range combat unit; **cavalry**: fast and medium health close combat unit; **healer**: low-health healing unit with close range; **grenadier**: low-health mid-range unit that inflicts area damage to all units.

Each side comprises 32 units of each type uniformly spread in each side's starting area, which are placed so that no unit can initially see an enemy unit (see Fig. 1 frames 1).

Behavior Tree as Controller

We pair Parabellum with behavior trees (BTs) (Colledanchise and Ögren, 2018) to determine the behavior of each agent. BTs are commonly used in robotics and video games, intuitive to design, and inherently interpretable.

More specifically, the units from each side share the same BT, meaning that the search space of GAME is one BT for the Red side and one BT for the Blue side. At each evaluation, each unit visits its BT starting from the root node, and undergoes a left-most depth-first visit of the tree until it finds a valid action. The BT comprises intermediate nodes: Sequence/Failwith nodes, which stop at the first invalid/valid node, and leaves nodes: Action nodes that return an action and its validity, and Condition nodes that return a boolean. To get a fast evaluation, we implemented a BT evaluation

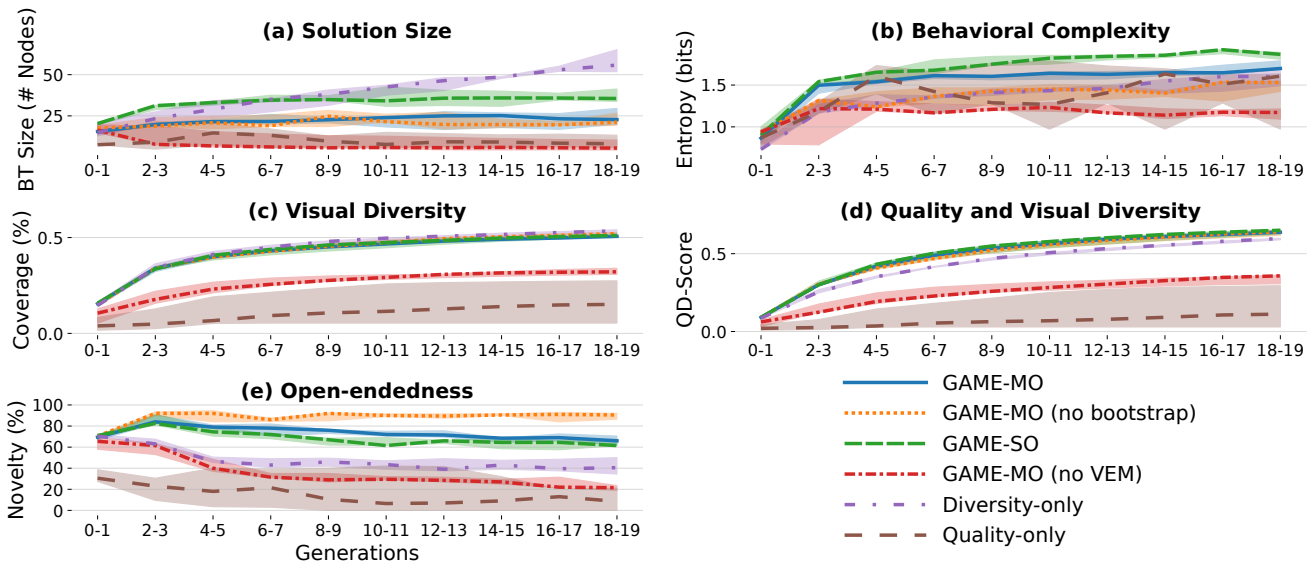

Figure 3: **GAME's variants and ablations comparisons.** The solid line represents the median, and the shaded area shows the min and max of 3 replications over 20 generations. (a) **Diversity-only** and **GAME-SO** show the largest increase in solution size, but (b) **GAME-SO** shows the largest increase in complexity. (c-d) **Quality-only** and **GAME-MO (no VEM)** lead to the worst performances, while the others are similar. (e) Removing bootstrapping leads to a constant discovery of novel solutions.

function in JAX that allows vectorization of the BT evaluation at the price of setting a maximal number of leaves (100).

Their tree nature allows the use of genetic variation operators for trees. For example, Iovino et al. (2021) evolve BTs for robot control, and Montague et al. (2023) use MAP-Elites with BTs to learn controllers for heterogeneous robot swarms. Taking inspiration from them, GAME randomly selects a variation operator: deleting a sub-tree (35%), adding a random node at a random location (21%), mutating a node by changing its parameters (7%), replacing a node with a random node (7%), or doing a crossover, i.e., copying a random sub-tree of another BT at a random location, (30%). Those probabilities could be tuned, but do not seem to change GAME's performances significantly.

Another advantage of using BT is that it is straightforward to increase its complexity by allowing it to grow, promoting an open-ended evolution (Harrington and Pollack, 2019). Such an increase in complexity is not trivial to obtain with neural networks and must rely on specific methods such as NEAT (Stanley and Miikkulainen, 2002).

The BT evolution requires the design of atomic functions (the Actions and Conditions) that take the unit's local observation and return the corresponding output. To allow synchronization between units, each side can set a target position on the map and move toward it even if it's out of sight. Different qualifiers parametrize the atomics to improve interpretation and usage, e.g., a *target* qualifier corresponding to either closest, farthest, weakest, strongest, or random.

The available Actions are: `Stand` (do nothing); `Attack` (attack the *target* enemy in reach of a given type); `Heal` (heal the *target* ally in reach of a given type); `Move` (move toward/away from the *target* ally/enemy of a given type); `Go To` (got to the target position); `Set Target` (mark the position of the *target* ally/enemy as target position).

The available Conditions are: `In Sight` (Is there an ally/enemy of a given type in sight?) `In Reach` (Is there an ally/enemy of a given type in reach?) `Is Dying` (Is my health or an ally/enemy's health below a given threshold?) `Is Type` (Am I of a given type?) `Is Set Target` (Is the target position set on the map?)

GAME variants and ablations

To evaluate GAME, we evaluate six variants:

- **GAME-MO**: full variant with a multi-objective fitness that first maximizes the opposing side's depleted health and only secondly minimizes the size of the BT;
- **GAME-MO (no bootstrap)**: GAME-MO with an ablation of the bootstrapping between generations;
- **GAME-SO**: full variant that uses a single-objective fitness to maximize the opposing side's depleted health;
- **GAME-MO (no VEM)**: full variant with a 2D handcrafted behavior space equal to the evaluated side average remaining health and the time before completion (i.e., one side wins or 100 steps timeout) instead of a VEM;
- **Quality-only**: MT-ME instead of MT-MB ME, i.e., each task has only one solution;
- **Diversity-only**: do not use fitness, only rule (e.1).

We run three replications of 20 generations for each variant with 100 000 evaluations of 100 steps per generation, 100 tasks per generation, and 25 cells per task's archive.

Measures

After each run, we realize a tournament between each generation's tasks, 10 Blue generations against 10 Red generations (i.e., 1 000 Red elites against 1 000 Blue elites), resulting in 1 000 000 evaluations. This allows us to study whether there are intergenerational improvements.

Solution Size and Behavioral Complexity To measure the variations of solution size, we measure the number of nodes of the BT elites (Fig. 3.a), and for behavioral complexity, we measure the entropy of the distribution of actions performed over time and units of the same side (Fig. 3.b).

Visual Diversity and Quality Visual Diversity Coverage and QD-Score are classic measures of QD algorithms, but are not easily computed using the archive in a large behavior space. As a proxy, we compute a single 2D projection using PCA (the two components capturing 22% and 12% of the explained variance) for all the behaviors of the intergenerational tournaments for all replications (Fig. 4), discretize this space with a 100×100 grid, and compute the corresponding elites. The Coverage (Fig. 3.c) corresponds to the proportion of non-empty cells, and the QD-Score (Fig. 3.d) to the average fitness of the elites.

Quality One limitation with computing the quality directly from the fitness in the QD-Score is that the fitness can be high because the opposite side is bad, and not because the winning side is good. To measure a less ambiguous quality of the elites found by each method, we follow Dharna et al. (2024) and select the best 10 solutions from each side for each run of each variant, perform a round-robin tournament, and compute the ELO score (Elo, 1978) (Fig. 5).

Open-endedness The Coverage measures the volume of novelty discovered by GAME with the VEM. One limitation is that this behavior is dependent on two solutions. Another limitation is that visual behavior may not capture the full range of novelty. We use another measure similar to PATA-EC proposed in Enhanced-POET. The idea is that a BT is different from another if, in a tournament, the ranking it generates in the opposition is different. Using the intergenerational-tournament, we compute the ranking of all the BTs, and count the number of new rankings per generation compared to the previous generations (Fig. 3.e). The idea is that an open-ended system will continue to create new challenges that require new solutions.

Results

Solution Size (Fig. 3.a) As expected, **GAME-SO** and **Diversity-only** lead to the largest increases in BT size across generations. **Diversity-only** is the only variant showing a constant increase in size, suggesting that fitness maximization indirectly leads to pruning.

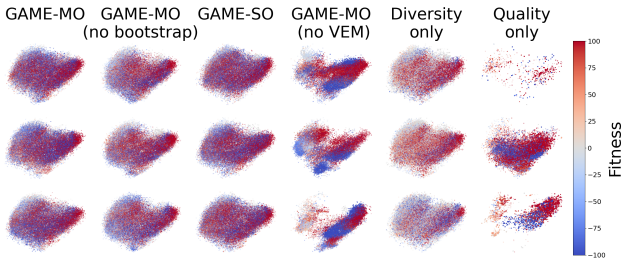

Figure 4: **PCA projections of the tournaments' behaviors.** GAME variants with a VEM show the most uniform coverage with less variance between replications.

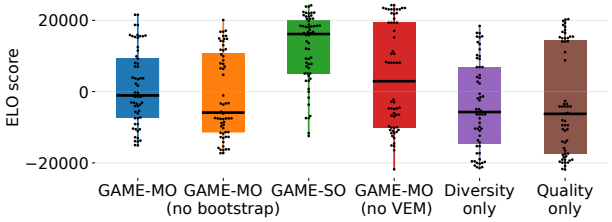

Figure 5: **Round robin tournament ELO score between the** 10 **best solutions of each replication. GAME-SO** is significantly better than all variants.

Behavioral Complexity (Fig. 3.b) **GAME-SO** also demonstrates the largest increase in behavioral complexity, suggesting that minimizing BT size prunes necessary stepping stones that lead to more complex behaviors. **Diversity-only** doesn't show an increase in behavioral complexity similar to its BT size growth, showing that increased size doesn't necessarily lead to greater behavioral complexity.

Visual Diversity (Fig. 3.c and Fig. 4) **Quality-only** and **GAME-MO (no VEM)** exhibit significantly less diversity than other variants, which show similar diversity levels, with **Diversity-only** being slightly superior. Additionally, using the VEM results in less variance in coverage between replications, further validating its use as a behavior space.

Quality and Visual Diversity (Fig. 3.d) **Quality-only** and **GAME-MO (no VEM)** show significantly worse QD-Scores (due to their poorer coverage). **Diversity-only** shows only a slightly worse QD-Score, suggesting that diversity alone is already a powerful optimizer, though the best QD-Score is achieved when also considering quality.

Quality (Fig. 5) Comparing the best solutions' quality, we observe that **GAME-SO** produces significantly better BTs. This suggests that increased behavioral complexity leads to significant performance improvements and that directly attempting to reduce solution size with a secondary objective prevents the discovery of the highest-performing solutions. **GAME-MO (no VEM)** also produces high-performing solutions. One explanation is that the handcrafted behavior space correlates more with the fitness, thus focusing the search toward better-performing solutions.

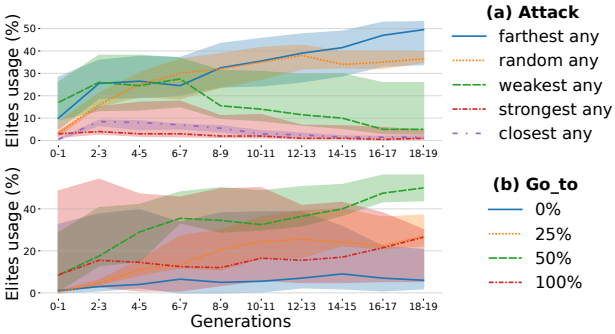

Figure 6: **Global trends**: Percentage of elites using (a) the `Attack` atomic with different *targets* and (b) the `Go_to` atomic for different *thresholds* through the generations. The solid line represents the median, and the shaded area between the min and max of **GAME-MO**'s three replications.

Open-endedness (Fig. 3.e) The **GAME-SO** and **GAME-MO** show similar open-endedness, which slightly decreases with each iteration, finding 80% new tasks in the 2nd and 3rd generations and declining to 60% by the final generations. **GAME-MO (no bootstrap)** consistently generates 90% novel BTs, suggesting this variant is the only one showing signs of true open-endedness throughout all 20 generations. The other variants demonstrate less open-endedness, with **Quality-Only** converging to 0% novelty after eight generations in one replication. This further demonstrates that diversity prevents getting trapped in local minima.

Two illumination takeaways GAME's goal is to illuminate the solution space. We looked at the elites of each generation across **GAME-MO**'s three replications and examined the atomics they used to determine global trends. We discovered two interesting findings.

Regarding the `Attack` atomic (with any unit type), all replications converged to favor attacking either the farthest unit or a unit at random (Fig. 6.a). Attacking at random spreads damage and avoids wasting resources on overkills. Attacking the farthest unit is also beneficial because grenadiers deal area damage that can be avoided if they target a unit sufficiently far away.

Examining the `Go_to` atomics, all replications favor using a threshold distance equal to 50% of the sight range while avoiding exact positioning, i.e., 0% (Fig. 6.b). This likely represents a balance between spending time moving precisely to the target position (i.e., 0%), which prevents units from performing other actions, and merely moving within sight of the target (i.e., 100%), which limits visibility of what can be observed from the target position itself.

An example of arms race We found an "arms race" in one of **GAME-SO**'s replications (Fig 7). The Red elites increased their focus on the archers, leading to the Blue elites' counter-strategy to focus their healing on their archers.

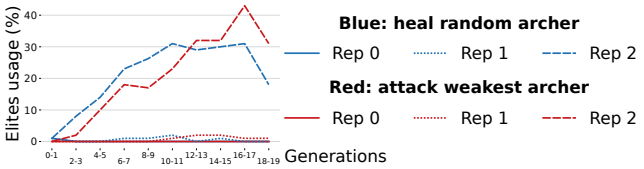

Figure 7: **Arms race example**: Percentage of Red elites using the `Attack` atomic on weakest enemy archer and Blue elites using the `Heal` atomic to random ally archers through the generations for **GAME-SO**'s three replications. For replication 2, as more Red elites targeted the Blue archers, Blue elites evolved to heal their archers.

Discussion

We hypothesized that minimizing the BT size as a secondary objective wouldn't impact quality and lead to smaller, more interpretable BTs. However, the results indicate it prunes the neutral mutations that seem essential stepping stones for high-performing solutions. This supports the neutral theory of molecular evolution (Kimura, 1979), which states that most variation at the molecular level is neutral yet leads to the evolution of complex organisms in nature.

We also hypothesized that bootstrapping each generation using solutions from previous ones would accelerate the search compared to starting from scratch. The results confirm this hypothesis. Nonetheless, **GAME (no bootstrap)** is the only variant demonstrating constant generation of new solutions throughout all 20 generations. This phenomenon could be related to extinction events (Lehman and Miikkulainen, 2015) and warrants further investigation, as it currently doesn't yield the best diversity or quality.

One limitation of our current evaluation of GAME is that Parabellum is a symmetrical game that may not present imbalance issues between sides. For example, POET (Wang et al., 2019), which coevolves controllers and environments, must rely on a selection of environments that are neither too easy nor too difficult for the current population of controllers. Future work should examine the application of GAME in asymmetric adversarial problems.

Conclusion

We present a new algorithm, GAME, that illuminates adversarial problems using coevolution. In combination with a vision embedding model, GAME implements a behavior space requiring only videos of the behavior instead of handcrafted behavior descriptors. Results on an adversarial multi-agent game confirm GAME's effectiveness in illuminating the adversarial space. The study also reveals interesting insights: bootstrapping reduces the reachable search space, thereby limiting open-endedness; and not pruning neutral mutations allows for preserving necessary stepping stones that lead to discovering the best-performing solutions. Future work will aim to demonstrate that GAME is a generalist algorithm applicable to various adversarial problems, not limited to symmetrical problems or behavior trees.

Acknowledgments

Funded by the armasuisse S+T project F00-007.

References

Anne, T. and Mouret, J.-B. (2023). Multi-task multi-behavior map-elites. In *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, pages 111–114.

Anne, T., Syrkis, N., Elhosni, M., Turati, F., Legendre, F., Jaquier, A., and Risi, S. (2025). Harnessing language for coordination: A framework and benchmark for llm-driven multi-agent control. *IEEE Transactions on Games*.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2019). Emergent tool use from multi-agent autocurricula. In *International conference on learning representations*.

Baldwin, R., Cave, M., and Lodge, M. (2011). *Understanding regulation: theory, strategy, and practice*. Oxford university press.

Bedau, M. A., McCaskill, J. S., Packard, N. H., Rasmussen, S., Adami, C., Green, D. G., Ikegami, T., Kaneko, K., and Ray, T. S. (2000). Open problems in artificial life. *Artificial life*, 6(4):363–376.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Brant, J. C. and Stanley, K. O. (2017). Minimal criterion coevolution: a new approach to open-ended search. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 67–74.

Brevault, L. and Balesdent, M. (2024). Bayesian quality-diversity approaches for constrained optimization problems with mixed continuous, discrete and categorical variables. *Engineering Applications of Artificial Intelligence*, 133:108118.

Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., and Mukhopadhyay, D. (2018). Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

Colledanchise, M. and Ögren, P. (2018). *Behavior trees in robotics and AI: An introduction*. CRC Press.

Costa, V., Lourenço, N., Correia, J., and Machado, P. (2020). Exploring the evolution of gans through quality diversity. In *Proceedings of the 2020 genetic and evolutionary computation conference*, pages 297–305.

Cully, A. (2019). Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 81–89.

Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. (2015). Robots that can adapt like animals. *Nature*, 521(7553):503–507.

Dawkins, R. and Krebs, J. R. (1979). Arms races between and within species. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 205(1161):489–511.

Dharna, A., Lu, C., and Clune, J. (2024). Quality-diversity self-play: Open-ended strategy innovation via foundation models. In *NeurIPS 2024 Workshop on Open-World Agents*.

Dorin, A. and Stepney, S. (2024). What is artificial life today, and where should it go?

Elo, A. E. (1978). *The Rating of Chessplayers, Past and Present*. Arco Publishing.

Ficici, S. G. and Pollack, J. B. (1998). Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In *Proceedings of the sixth international conference on Artificial life*, pages 238–247. MIT Press Cambridge, MA.

Fontaine, M. C., Lee, S., Soros, L. B., de Mesentier Silva, F., Togelius, J., and Hoover, A. K. (2019). Mapping hearthstone deck spaces through map-elites with sliding boundaries. In *Proceedings of The Genetic and Evolutionary Computation Conference*, pages 161–169.

Fontaine, M. C., Togelius, J., Nikolaidis, S., and Hoover, A. K. (2020). Covariance matrix adaptation for the rapid illumination of behavior space. In *Proceedings of the 2020 genetic and evolutionary computation conference*, pages 94–102.

Gravina, D., Khalifa, A., Liapis, A., Togelius, J., and Yannakakis, G. N. (2019). Procedural content generation through quality diversity. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE.

Harrington, K. and Pollack, J. (2019). Escalation of memory length in finite populations. *Artificial life*, 25(1):22–32.

Iovino, M., Styrud, J., Falco, P., and Smith, C. (2021). Learning behavior trees with genetic programming in unpredictable environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4591–4597. IEEE.

Jiang, Y., Salley, D., Sharma, A., Keenan, G., Mullin, M., and Cronin, L. (2022). An artificial intelligence enabled chemical synthesis robot for exploration and optimization of nanomaterials. *Science advances*, 8(40):eabo2626.

Kimura, M. (1979). The neutral theory of molecular evolution. *Scientific American*, 241(5):98–129.

Kumar, A., Lu, C., Kirsch, L., Tang, Y., Stanley, K. O., Isola, P., and Ha, D. (2024). Automating the search for artificial life with foundation models. *arXiv preprint arXiv:2412.17799*.

Lehman, J. and Miikkulainen, R. (2015). Enhancing divergent search through extinction events. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 951–958.

Lehman, J. and Stanley, K. O. (2011). Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218.

Li, D., Li, Q., Ye, Y., and Xu, S. (2021). Arms race in adversarial malware detection: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–35.

Montague, K., Hart, E., Nitschke, G., and Paechter, B. (2023). A quality-diversity approach to evolving a repertoire of diverse behaviour-trees in robot swarms. In *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*, pages 145–160. Springer.

Moran, N. and Pollack, J. (2019). Evolving complexity in prediction games. *Artificial Life*, 25(1):74–91.

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

Mouret, J.-B. and Maguire, G. (2020). Quality diversity for multi-task optimization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pages 121–129.

Pugh, J. K., Soros, L. B., and Stanley, K. O. (2016). Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Samvelyan, M., Paglieri, D., Jiang, M., Parker-Holder, J., and Rocktäschel, T. (2024a). Multi-agent diagnostics for robustness via illuminated diversity. *arXiv preprint arXiv:2401.13460*.

Samvelyan, M., Raparthy, S., Lupu, A., Hambro, E., Markosyan, A. H., Bhatt, M., Mao, Y., Jiang, M., Parker-Holder, J., Foerster, J., Rocktaschel, T., and Raileanu, R. (2024b). Rainbow teaming: Open-ended generation of diverse adversarial prompts. *ArXiv*, abs/2402.16822.

Schelling, T. C. (1980). *The Strategy of Conflict: with a new Preface by the Author*. Harvard university press.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127.

Steckel, K. and Schrum, J. (2021). Illuminating the space of beatable lode runner levels produced by various generative adversarial networks. In *Proceedings of the genetic and evolutionary computation conference companion*, pages 111–112.

Vassiliades, V., Chatzilygeroudis, K., and Mouret, J.-B. (2016). Scaling up map-elites using centroidal voronoi tessellations. *arXiv preprint arXiv:1610.05729*.

Vassiliades, V., Chatzilygeroudis, K., and Mouret, J.-B. (2017). A comparison of illumination algorithms in unbounded spaces. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1578–1581.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354.

Wan, Z., Yu, X., Bossens, D. M., Lyu, Y., Guo, Q., Fan, F. X., and Tsang, I. (2024). Quality diversity imitation learning. *arXiv preprint arXiv:2410.06151*.

Wang, R., Lehman, J., Clune, J., and Stanley, K. O. (2019). Poet: open-ended coevolution of environments and their optimized solutions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 142–151.

Wang, R., Lehman, J., Rawal, A., Zhi, J., Li, Y., Clune, J., and Stanley, K. (2020). Enhanced poet: Open-ended reinforcement learning through unbounded invention of learning challenges and their solutions. In *International conference on machine learning*, pages 9940–9951. PMLR.