Avoiding Convergence in Cooperative Coevolution with Novelty Search

Jorge Gomes[1,2], Pedro Mariano[2], Anders Lyhne Christensen[1,3]
[1]Instituto de Telecomunicações, Lisboa, Portugal
[2]LabMAg – Faculdade de Ciências da Universidade de Lisboa, Portugal
[3]ISCTE-IUL, Lisboa, Portugal
jgomes@di.fc.ul.pt, plmariano@di.fc.ul.pt, anders.christensen@iscte.pt

ABSTRACT

Cooperative coevolution is an approach for evolving solutions composed of coadapted components. Previous research has shown, however, that cooperative coevolutionary algorithms are biased towards stability: they tend to converge prematurely to equilibrium states, instead of converging to optimal or near-optimal solutions. In single-population evolutionary algorithms, novelty search has been shown capable of avoiding premature convergence to local optima — a pathology similar to convergence to equilibrium states. In this study, we demonstrate how novelty search can be applied to cooperative coevolution by proposing two new algorithms. The first algorithm promotes behavioural novelty at the team level (*NS-T*), while the second promotes novelty at the individual agent level (*NS-I*). The proposed algorithms are evaluated in two popular multiagent tasks: predator-prey pursuit and keepaway soccer. An analysis of the explored collaboration space shows that (i) fitness-based evolution tends to quickly converge to poor equilibrium states, (ii) *NS-I* almost never reaches any equilibrium state due to constant change in the individual populations, while (iii) *NS-T* explores a variety of equilibrium states in each evolutionary run and thus significantly outperforms both fitness-based evolution and *NS-I*.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

Keywords

Cooperative coevolution, novelty search, convergence to stable states, behaviour space exploration

1. INTRODUCTION

Cooperative coevolution algorithms (CCEAs) are capable of evolving solutions that consist of interacting coadapted components [19]. Such approaches are promising because they potentially allow for very large problem spaces to be decomposed into smaller and more manageable sub-problems.

In a typical CCEA, each component of the solution is evolved in a separate population. Components are evaluated as part of a complete solution that consists of one component from each population. The individual components are thus scored based on the performance of the complete solution as a whole rather than their individual contribution.

A common application of CCEAs is the evolution of multiagent behaviours [20]. The natural decomposition of the problem into sub-components makes it a good fit for cooperative coevolution: each agent can be represented as a component of the solution, and the coevolutionary algorithm evolves a set of agent behaviours that solve the given task. In this way, coevolution allows for the evolution of heterogeneous multiagent systems, where each individual agent can evolve a specialised behaviour [20, 27, 12].

CCEAs are, however, plagued by a number of issues, among which premature convergence to equilibrium states stands out [25, 15]. CCEAs have been shown to gravitate towards equilibrium states, regardless of whether or not such states correspond to a (near-)optimal solution for a given problem. *Relative over-generalisation* is one of the main causes for this behaviour [18]. Relative over-generalisation occurs when coevolving populations are attracted to areas of the search space in which there are many strategies that perform well when combined with individuals from the other populations. This tends to generate individuals that are *jack of all trades, master of none* [26], which in turn can hinder the evolution of optimal or near-optimal solutions.

Relative over-generalisation is not the same as convergence to local optima found in single-population evolutionary algorithms [17]. The relative over-generalisation pathology means that even under favourable conditions, CCEAs are not necessarily attracted to the global optimum [18]. The consequence of relative over-generalisation and convergence to local optima [24] are, however, similar: the evolutionary process converges to sub-optimal regions of the solution space. The two pathologies are caused by a form of *deception*: in single-population evolutionary algorithms, evolution might be deceived by the fitness function [24], while in CCEAs, each population might be deceived by the other coevolving populations.

Lehman and Stanley [6] recently proposed an evolutionary approach aimed at avoiding deception in single-population evolutionary algorithms, called *novelty search*. In novelty search, candidate solutions are scored based on the novelty of their behaviour with respect to the behaviours of previously evaluated individuals, and not based on a traditional,

static fitness objective. Given the dynamic nature of the objective, novelty search has the potential to avoid deception and premature convergence. The approach has attained considerable success in many different domains, such as evolutionary robotics [9, 3], genetic programming [5] and reinforcement learning tasks [1, 8].

In this paper, we show how novelty search can be applied to cooperative coevolution. We propose two new algorithms: novelty search at the team level (*NS-T*) and at the individual level (*NS-I*). In *NS-T*, the novelty score assigned to an individual is based on the behaviour displayed by the team in which it is evaluated — regardless of its individual contribution. In *NS-I*, the novelty assigned to an individual is solely based on the behaviour of that individual agent when a team is evaluated. We apply the proposed algorithms in the evolution of controllers for embodied agents in two classic tasks that require a high degree of cooperation: predator-prey [11, 27, 21] and keep-away soccer [22, 23]. We show that by having a dynamic novelty objective, the populations avoid getting trapped in poor equilibrium states. Consequently, a coevolutionary process based on novelty search can find good solutions faster and more often than a traditional fitness-based cooperative coevolution algorithm.

2. RELATED WORK

2.1 Cooperative Coevolution

Our work is based on the original cooperative coevolution architecture (CCEA) proposed by Potter and De Jong [19]. This architecture models an ecosystem consisting of two or more species, with each species represented in a separate population. This means that individuals only compete and reproduce with members of their own species. Each species is evaluated in turn. To evaluate an individual from one species, teams are formed with representatives the other species. The teams are then evaluated by a fitness function in the problem domain, and only the individual that is being evaluated receives the fitness score obtained by the team as a whole. Therefore, the fitness differential is strictly a function of the individual's contribution to the problem-solving effort within the context of the team members from the other species.

Popular extensions of the CCEA architecture include: Multiagent Enforced SubPopulations (MESP) [27], where neural networks are coevolved using the ESP neuroevolution algorithm; and COllective NeuroEvolution (CONE) [10], which extends MESP by allowing regulated reproduction between individuals of different populations.

The cooperative coevolution architecture has been used in several multiagent tasks to evolve cooperative agent behaviours. The most notable example is the predator-prey task [11, 27], where a number of agents (predators) need to cooperate in order to capture a preprogrammed fleeing prey. Other applications of cooperative coevolution include a herding task [20], where a group of robots have to cooperate to force another robot into a corral; collective construction tasks [13], which require that robots place building blocks in a specific sequence to build a predefined structure; and collective foraging [12], where a team of simulated autonomous vehicles need to find objects of interest with a maximal total value over the course of the team's lifetime.

2.2 Convergence to Equilibrium States

In a cooperative coevolution algorithm, the search space of each population is defined and limited by the behaviour of the team members. The search space is thus constantly changing, as the individuals from the other populations are evolving. It is consequently easy for a population to get mislead because of particular selections of representatives from the other populations. CCEAs therefore tend to evolve individuals that perform well with many different individuals in the other populations, regardless of whether or not these teams are globally optimal [26]. This pathological evolutionary dynamic is known as relative over-generalisation [18].

A number of strategies have been proposed to overcome the relative over-generalisation issue. The intent is to bias coevolution towards optimal solutions and escape sub-optimal Nash equilibria. In [17], it is shown that an optimistic reward scheme — the *maximum of N collaborations* — can be used to mitigate the problem of relative over-generalisation. The idea is to evaluate an individual not with one collaborator, but with N randomly chosen collaborators and use the maximum reward obtained. However, the results show that this scheme alone only avoids relative over-generalisation when the number of collaborators N is relatively high [14], which has a significant impact on computational complexity.

The maximum of N collaborations scheme was augmented by Panait et al. [17]: the fitness is based partly on the maximum of N collaborations with randomly chosen partners and partly on the reward obtained when partnering with the optimal collaborator. The results show that the augmented strategy using small values of N is superior to the maximum of N collaborations scheme using larger values for N. The assumption that the optimal collaborator is known is, however, unrealistic, and as such heuristic methods are necessary for estimating the optimal collaborator. To overcome this necessity, Panait et al. [16] proposed an archive-based algorithm called *iCCEA*, which attempts to reduce the number of evaluations by maintaining an archive of good collaborations for each of the populations. The idea is to identify which members of the collaborating population are likely to be good collaborators and only test individuals against that small set. The results show that *iCCEA* often performs better than the classical CCEA algorithm in which collaborators are chosen randomly.

The existing studies on over-generalisation and convergence to equilibrium states are, however, mostly focused on the function optimisation domain. Although the issue of convergence is not directly addressed in previous studies in the multiagent domain, the principles of the pathology also apply to other domains. Actually, in many studies that apply CCEAs to embodied multiagent systems, problem decomposition techniques [15] (in particular incremental evolution [4]) are used to achieve successful solutions in reasonable time (see for examples [10, 13, 27]). The described necessity of using problem decomposition supports that CCEAs tend to converge to sub-optimal equilibria.

It is not clear whether the aforementioned techniques for overcoming over-generalisation can be applied in the embodied multiagent domain. In particular, these techniques rely on using a relatively large number of collaborators to assess the fitness of each individual. This is typically not feasible in the multiagent domain, since evaluations tend to be computationally expensive.

2.3 Novelty Search

Novelty search [6] is a new approach that drives evolution towards behavioural novelty instead of a pre-defined goal. The distinctive aspect of novelty search is how the individuals of the population are scored. Instead of being scored according to how well they perform a given task, which is typically measured by a fitness function, the individuals are scored based on their behavioural novelty according to a *novelty metric*, which quantifies how different an individual is from other, previously evaluated individuals.

To measure how far an individual is from other individuals in behaviour space, the novelty metric relies on the average behaviour distance of that individual to the k-nearest neighbours. Potential neighbours include the other individuals of the current population and a sample of individuals from previous generations (stored in an archive). Candidates from sparse regions of the behaviour space therefore receive higher novelty scores, and in novelty search there is thus a constant evolutionary pressure towards behavioural innovation.

The behaviour distance between each two individuals is given by a function that should be provided by the experimenter. The behaviour of each individual is commonly characterised by a real-valued vector. The experimenter should design the characterisation vector so that it captures behaviour features that are considered relevant to the problem or task. The behaviour distance between two individuals is then given as the Euclidean distance between their characterisation vectors. The discussion of how to define behaviour characterisations is largely dependent on the application domain, and goes beyond the scope of this paper. In the past, behaviour characterisations have been specified for a number of different tasks in various domains, including single-robot systems [9, 6], multirobot systems [3, 2], reinforcement learning tasks [1, 8], among others.

Combining Novelty and Fitness

As novelty search is guided by behavioural innovation alone, its performance can be greatly affected by the size and shape of the behaviour space. In particular, behaviour spaces that are vast or contain dimensions not related with the task can cause novelty search to perform poorly [1], because most of the effort may be spent exploring behaviours that are irrelevant for the goal task. To address this issue, we use a linear scalarisation of novelty and fitness objectives [1]. This technique directs the exploration towards regions of the behaviour space associated with high fitness scores. An individual i is evaluated to measure both fitness, f_i, and novelty, ns_i, which after being normalised are combined according to the following equation:

$$score_i = (1 - \rho) \cdot \overline{f_i} + \rho \cdot \overline{ns_i} \ , \tag{1}$$

where ρ is a parameter set by the experimenter that controls the relative weight of fitness and novelty.

3. NOVELTY-DRIVEN COOPERATIVE COEVOLUTION

We propose two distinct approaches based on the novelty search algorithm to overcome convergence in cooperative coevolution. In the first approach, we use novelty search to promote diversity of team behaviours. In the second approach, novelty search is used to promote novel behaviours of the individual agents.

Team-level novelty

Following the classic cooperative coevolution architecture [19], each agent is evolved in a separate population. To evaluate each individual of each population, teams are formed with representatives from the other populations. The collective performance of the team is then obtained by testing it in the problem task. A straightforward way of introducing novelty search in this architecture is to characterise the behaviour of each team as a whole, and use this characterisation to compute the novelty score of the individual that is being evaluated. The team behaviour is measured using the behaviour characterisation design principles used in [3]: the characterisation should capture what the team as a whole achieves, rather than directly discriminating what each agent does for the team.

The rationale behind this algorithm is to reward individuals that facilitate novel collective behaviours. Individuals are not rewarded for being novel themselves — but rather for the novelty displayed by the team of which they are part. The evolutionary process is thus lead towards novel equilibrium states. As the notion of novelty is dynamic, the attractors keep changing throughout evolution, which in turn avoids premature convergence to a single point in the solution space. An overview of our team-level approach is described in Algorithm 1.

Algorithm 1 Novelty-driven cooperative coevolution based on team-level evaluations

---

1: Let A be an archive of behaviours.
2: **for** each generation **do**
3:     Choose the representatives r_n of each population p_n.
4:     **for** each population p_n **do**
5:         **for** each individual i in p_n **do**
6:             Form teams with individual i and representatives r from the other populations.
7:             Evaluate every team t, obtaining for each one a fitness score f_t and a characterisation b_t.
8:             $f_i \leftarrow f_{t^*}$, $b_i \leftarrow b_{t^*}$, where t^* is the team with the highest fitness score f_t.
9:         **for** each individual i in p_n **do**
10:             Compute the novelty score ns_i, using the characterisation b_i, and comparing to the other individuals in p_n and behaviours in A.
11:             $score_i \leftarrow$ combine f_i and ns_i.
12:         Update A, adding the behaviours b_i of R randomly picked individuals of p_n.
13:     **for** each population p_n **do**
14:         Run the selection, mutation, and crossover operators, based on the final scores $score_i$.

Individual-level novelty

It is typically not possible to assess the contribution of each agent to the success of the team (known as the *credit assignment problem* [15]). Nonetheless, it is possible to describe the behaviour of each individual agent, regardless of what the team achieved. We therefore study a novelty-based coevolutionary algorithm that uses individual agent characterisations, instead of the team-level behaviour characterisation. Individuals are rewarded for displaying novel agent behaviours, regardless of the team-level behaviour of the teams in which the individuals were evaluated. The goal is to maintain diversity inside each population, thus preventing convergence to any equilibrium state. The algorithm is similar to Algorithm 1, with the following modifications:

- There are N archives, one for each population.

- In the evaluation of a team, N behaviour characterisations are obtained, one for each representative of a population.

- To calculate the novelty score, the individual is compared only to the individuals of its own population, and the archive of that population.

- Archives only contain behaviours from the population to which they belong.

4. EXPERIMENTAL SETUP

Cooperative coevolution algorithms are used to evolve a group of heterogeneous agent controllers to solve the given task. The proposed methods are evaluated in two popular multiagent tasks: predator-prey and keepaway soccer. In both tasks, each agent of the group is controlled by a neural network. The neural networks are evolved with simple genetic algorithms that operate inside each of the coevolving populations. MASON [7] is the simulation platform used for the multiagent simulations in which the neural networks are evaluated. Each controller is evaluated in 10 independent simulations, and the final fitness score and behaviour characterisations are the average of the scores obtained in those simulations.

4.1 Predator-prey

The predator-prey pursuit is one of the most common tasks in multiagent coevolution (both cooperative and competitive) [11, 27, 21]. Pursuit games consist of a number of agents (predators) cooperatively chasing a prey. Individual predators are usually not faster than the prey, and the predators therefore need to cooperate in order to successfully capture the prey. In cooperative coevolution studies, typically only the team of predators is evolved, and the prey has a pre-specified fixed behaviour. The task is especially interesting because heterogeneity in the predator team is required to effectively catch the prey [27].

The predators are placed evenly spaced at one end of the 100x100 unit square arena. A single prey is randomly placed within a radius of 20 units around the centre. The arena is not bounded by walls. However, if the prey escapes the chase zone, the simulation ends. The initial conditions of the simulations are depicted in Figure 1 (left).

Predator-prey   **Keepaway soccer**
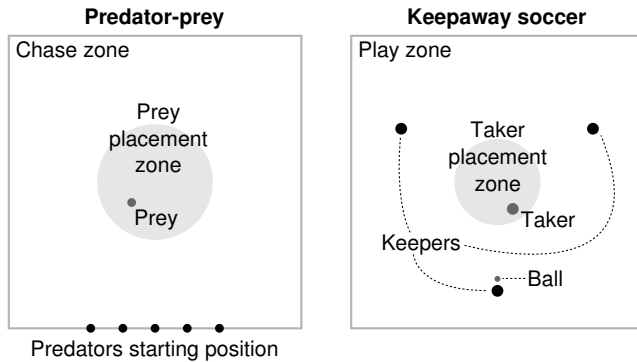
Figure 1: Initial simulation conditions, in the predator-prey task (left), and the keepaway soccer task (right).

We use a version of the task where the predators cannot communicate nor sense one another [27, 21]. Each predator receives only two inputs: (i) the distance to the prey, and (ii) the relative orientation of the agent with respect to the prey. These inputs are fed to the neural network, and two outputs control the speed and the rotation of the agent. The predators move at a maximum speed of 1 unit/step, and can only rotate a maximum of $90°$/step. The feed-forward neural network that governs each predator has 2 inputs, 5 hidden neurons, and 2 outputs, and is fully connected.

The behaviour of the prey consists of escaping from nearby predators. If there are no predators in a radius less than or equal to V, the prey does not move. Otherwise, the prey moves at a constant speed of 1 unit/step, with a direction opposite to the centre of mass of the nearby predators. Two variations of the task are used: (i) 3 predators, with a prey vision range V of 5 units, and (ii) 5 predators, with $V = 7$. The prey is captured if a predator collides with it. A simulation ends if the prey is captured, escapes the chase zone, or if 300 simulations steps elapse.

The fitness function F_{pp} is based on previous works [11, 27], and given by:

$$ F_{pp} = \begin{cases} 2 - \tau/T & \text{if prey captured} \\ (d_i - d_f)/size & \text{otherwise} \end{cases}, \qquad (2) $$

where τ is the simulation length, T is the maximum simulation length, d_i is the average initial distance from the predators to the prey, d_f is the average final distance, and $size$ is the side length of the chase zone.

The team-level behaviour characterisation $\mathbf{B}_{pp}(G)$ is a vector of length 4, with all elements normalised to [0,1]: (i) whether the prey was captured or not; (ii) the simulation length; (iii) the average final distance of the predators to the prey; and (iv) the average distance of the predators to their centre of mass (dispersion) throughout the simulation. The agent-level characterisation $\mathbf{B}_{pp}(a)$ of agent a is also a vector of length 4, with the elements normalised to [0,1]: (i) whether agent a captured the prey; (ii) the average distance of a to the prey; (iii) the average speed of a; (iv) the average distance of a to the other predators.

4.2 Keepaway soccer

Keepaway soccer is a simplified version of robot soccer in which there are usually three keepers and one or two takers. The keepers must learn to keep possession of the ball against a taker that actively tries to snatch it from the keepers. Keepaway soccer is a popular and challenging task in multiagent learning [22, 23]. Typically, the keepers are homogeneous and only one controller is evolved, which is cloned to each keeper. In our study, we use a heterogeneous version of the task: the keepers move at different speeds, and the controllers for the keepers are cooperatively coevolved.

The initial conditions of the simulation are depicted in Figure 1 (right). The arena is not bounded, but the ball must remain inside a 100x100 unit square. At the beginning of each simulation, three keepers are always placed in the same locations, and the single taker is randomly placed inside a circle of radius 15 units around the centre. The ball is placed in front of the bottom keeper. Each simulation runs for a maximum of 500 steps and ends if the ball leaves the play zone or is captured by the taker.

We use an abstraction of the task where the keepers do not actually have to learn in detail how to pass the ball.

Instead, the keepers only have to touch the ball, and two outputs of the controller dictate the power and direction of the pass. Each keeper is equipped with nine sensors: the distance and angle to each of the other keepers, to the taker, and to the ball; and the distance of the ball to the centre of the arena. The controller has four outputs: (i) the speed of the keeper, (ii) the rotation, (iii) the power of the pass, and (iv) the angle of the pass (relative to the agent's orientation). The last two outputs are only used when the agent touches the ball. The controllers are fully connected feed-forward networks with 9 inputs, 7 hidden neurons, and 4 outputs.

The three keepers are heterogeneous and can move at maximum speeds of 0.5, 0.75, and 1 unit/step, respectively. The keepers can pass the ball in any direction. After being shot, the maximum speed of the ball is 4 units/step, and it decelerates at a rate of 0.05 units/step^2. The taker is always chasing the ball at maximum speed S. Two variations of the task are used, with $S = 0.50$ and $S = 0.75$.

The fitness function F_{ks} is based on the number of successful passes accomplished throughout the simulation (as advocated in [23]). A pass occurs whenever the ball moves from one keeper to other, travelling a distance greater than 10 units. The distance criterion was included to prevent solutions were the agents achieve a very high number of passes while almost touching one another.

The team-level behaviour characterisation $\mathbf{B}_{ks}(G)$ is a vector of four elements, each normalised to [0,1]: (i) the number of passes; (ii) length of the simulation; (iii) average dispersion of the keepers (average distance to centre of mass); and (iv) average pass distance. The individual-level characterisation $\mathbf{B}_{ks}(a)$ of a keeper a is also a vector of length 4: (i) the number of passes made by agent a; (ii) average pass distance of these passes; (iii) average distance to the other keepers; and (iv) average movement of a.

4.3 Evolutionary Setup

Our experiments use the classical CCEA architecture, where evolutionary algorithms operate to evolve the neural networks that control the agents. The weights of the networks are directly encoded in the chromosomes. To evaluate each individual, two teams are formed: one with the highest scoring individual of each other population; and one with a randomly picked individual from each other population. Novelty search is implemented as described in Section 3. The archive size is bounded: after reaching the size limit, random individuals are removed to allow space for new ones. All the evolutionary techniques are implemented over ECJ.[1] The parameters of the algorithm were tuned in preliminary experiments using the predator-prey task. For each parameter, a set of values was tested, and the value that yielded the highest fitness scores was chosen. The resulting parameter values are listed in Table 1.

Table 1: Parameters of the evolutionary algorithm.

Parameter	Value	Parameter	Value
Novelty nearest-k	15	Tournament size	5
Add archive prob.	0.01	Mutation prob.	0.05
Max archive size	1000	Crossover	none
Linear scal. ρ	0.50	Best collaborators	1
Population size	150	Random collabor.	1

[1]`http://cs.gmu.edu/~eclab/projects/ecj/`

5. RESULTS

As detailed in the previous section, we defined four task setups, summarised below:

P3 Predator-prey with three predators. The prey senses predators at a distance of $V = 5$.

P5 Predator-prey with five predators and $V = 7$.

KS Keepaway soccer with slow taker ($S = 0.50$).

KF Keepaway soccer with fast taker ($S = 0.75$).

Each task setup was tested with three evolutionary methods (see Section 3): Traditional fitness-based cooperative coevolution (*Fit*); novelty search with team-level evaluation (*NS-T*); and novelty search with individual-level evaluation (*NS-I*). Each experimental condition was repeated in 20 independent evolutionary runs.

5.1 Escaping equilibrium states

Figure 2 shows the performance of each method in each task, with respect to the highest fitness scores achieved at each generation. The first conclusion that can be drawn from the plots is that *NS-T* is clearly superior to fitness-based coevolution (*Fit*). *NS-T* achieves significantly higher fitness scores than *Fit* (*p*-value < 0.01, Mann-Whitney U test), at almost all stages of the evolutionary process in all task setups. The boxplots highlight that the evolutionary runs of *Fit* often fail to achieve solutions for the given task, reaching only low fitness scores. For instance, *Fit* was never able to find a solution for the predator-prey task.

We analysed the exploration of the collaboration space with each method, using the team-level behaviour characterisations defined for each task (see Section 4), to determine the cause of the poor performance of fitness-based coevolution. The four dimensions of the behaviour characterisations were reduced to two dimensions using Kohonen self-organising maps in order to obtain a visual representation of the collaboration space exploration. The Kohonen maps were trained with a sample of the behaviours found in each task, and the behaviours evolved in each evolutionary run were then mapped. The resulting plots can be seen in Figure 3.

Fitness-based coevolution tends to only explore narrow regions of the collaboration space, which suggests that evolution is strongly attracted to certain equilibrium states. However, these states do not necessarily correspond to optimal solutions — fitness-based coevolution often converged to sub-optimal equilibria, and appears to lack the ability to escape those equilibrium states. On the contrary, *NS-T* does not seem to be affected by premature convergence to sub-optimal equilibria. *NS-T* explores a much wider range of collaborations (i.e., solutions), and can reach collaboration regions associated with higher fitness scores. *NS-T* explored the behaviour regions that were explored by *Fit*, however, it did not get trapped in those regions. It is also possible to observe that *NS-T* is able to unveil a large diversity of solutions for a given task in a single evolutionary run.

To make a statistical analysis of the behaviour space exploration, we devised measures of behaviour space coverage. The space was first divided in regions of equal size: each behaviour dimension was discretised into 5 levels, and each region corresponded to a unique combination of levels. We then calculated how many times each region was
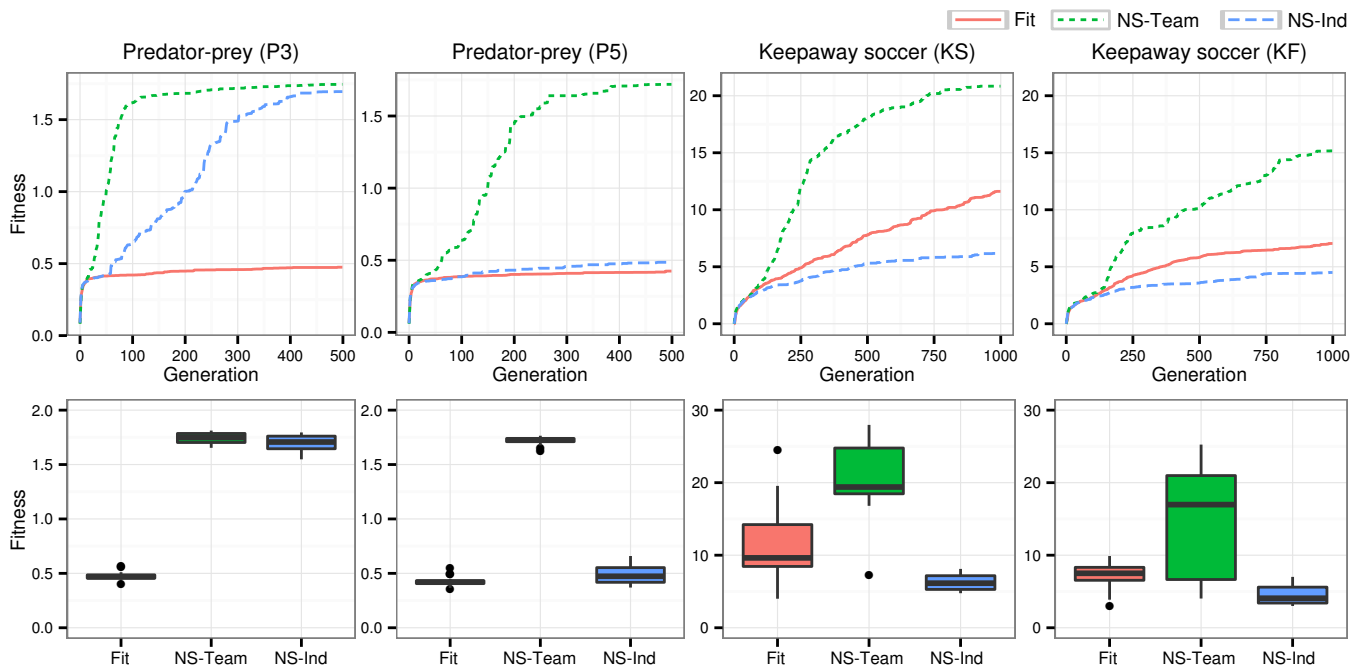
Figure 2: Top: best fitness score found at each generation with each method, averaged over 20 evolutionary runs. Bottom: boxplots of the highest fitness scores found in each evolutionary run.
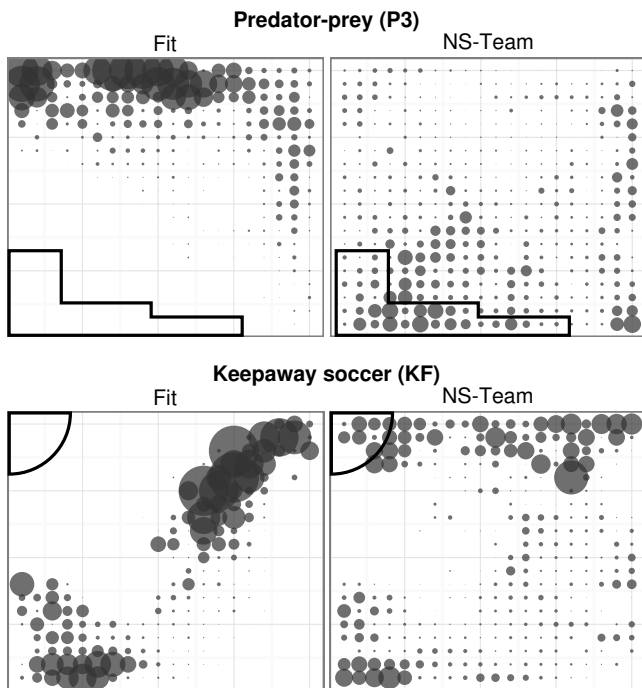

Figure 3: Exploration of the collaboration space in typical evolutionary runs of fitness-based coevolution and *NS-T*. The diameter of each circle is proportional to the number of teams found belonging to that behaviour region. The behaviour regions associated with the highest fitness scores are highlighted in the maps.

visited throughout the evolutionary process. To determine behaviour space coverage for an evolutionary run, we computed the behaviour distribution over the whole evolutionary run. The distribution was then compared with the uniform distribution using the Jensen-Shannon divergence. The complementary value of the divergence is the measure of behaviour space coverage. The non-visited regions of the behaviour space (the regions not reached by any method) were not used in the computation of the divergence. The results can be seen in Table 2 (top).

The results in Table 2 (top) confirm that *NS-T* covers the collaboration space significantly better than fitness-based coevolution. This result is consistent across all task setups. So, how does novelty search manages to evolve such a greater collaboration diversity? To shed some light on this question, we performed a distinct analysis of the coverage of the collaboration space, considering the composition of individual generations, instead of the whole evolutionary run. The results are shown in Table 2 (bottom).

In fitness-based coevolution, the average coverage in each generation is only slightly lower than the total coverage that occurs over the whole evolutionary run (see Table 2). Again, this supports the conclusion that fitness-based coevolution tends to quickly converge to certain regions of the collaboration space. The difference between the generation coverage and the total coverage is much more pronounced in novelty search. This result suggests that novelty search is able to shift the focus of convergence over the course of the evolutionary run — it manages to escape the strong attraction to stable states that plagues *Fit*. Furthermore, when compared to *Fit*, *NS-T* can maintain a much higher diversity of collaborations in the population at all times.

Table 2: Average coverage of the collaboration space in a whole evolutionary run (top), and in each generation (bottom). Standard deviations are in parenthesis.

Average coverage per evolutionary run			
	P3	P5	KS	KF
Fit	0.20 (0.01)	0.20 (0.03)	0.22 (0.06)	0.22 (0.02)
NS-T	0.62 (0.06)	0.52 (0.03)	0.45 (0.05)	0.43 (0.10)
NS-I	0.44 (0.06)	0.32 (0.01)	0.24 (0.01)	0.28 (0.01)

Average coverage per generation			
	P3	P5	KS	KF
Fit	0.15 (0.03)	0.15 (0.03)	0.16 (0.05)	0.16 (0.02)
NS-T	0.37 (0.05)	0.32 (0.07)	0.25 (0.06)	0.26 (0.08)
NS-I	0.24 (0.08)	0.20 (0.04)	0.16 (0.03)	0.19 (0.04)

Table 3: Average coverage of the agent behaviour space with each method. Standard deviations are in parenthesis.

|	P3	P5	KS	KF
Fit	0.28 (0.01)	0.35 (0.03)	0.27 (0.04)	0.23 (0.04)
NS-T	0.42 (0.02)	0.44 (0.01)	0.32 (0.02)	0.34 (0.02)
NS-I	0.66 (0.03)	0.63 (0.01)	0.34 (0.01)	0.32 (0.01)

5.2 Team novelty / Individual novelty

The performance of individual-level novelty search (*NS-I*) was significantly inferior to *NS-T* (see Figure 2). The main issue is that cooperation is not directly encouraged in *NS-I*. As Table 3 shows, *NS-I* is effective in discovering a greater diversity of agent behaviours, when compared to *NS-T* and *Fit*. However, this does not translate in the discovery of novel collaborations (see Table 2).

Figure 4 shows the exploration patterns that occur inside each population with *NS-T* and *NS-I*, in typical evolutionary runs. This figure highlights that *NS-I* evolves a relatively high diversity of agent behaviours inside each population. However, the low fitness scores achieved by *NS-I* suggest that this high diversity can actually be counterproductive: since all populations are constantly changing, it is hard to form effective collaborations, as the individuals of a population do not have enough time and incentive to adapt to the individuals of the other popuations. This evolutionary dynamic is contrary to what occurs in *NS-T*, where each population tends to specialise in one area of the behaviour space at a time, thus allowing a better adaptation of the populations to each other.

Overall, our results showed that for the purpose of achieving effective solutions, novelty search with team-level characterisations is largely superior to novelty search with individual-level characterisations.

6. CONCLUSIONS

We proposed two methods for overcoming convergence to equilibrium states in cooperative coevolution, based on novelty search. The first method relies on team-level characterisations and encourages novel collaborations (*NS-T*), while the second method tries to promote diversity at an agent-
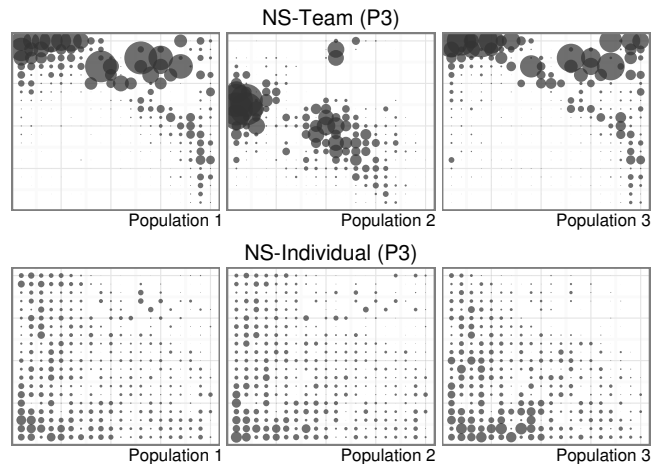
Figure 4: Exploration of the agent behaviour space inside each population, in typical evolutionary runs of *NS-T* and *NS-I* with the P3 task setup.

level (*NS-I*). Both methods were compared to traditional fitness-based coevolution (*Fit*) in two popular multiagent tasks: predator-prey pursuit and keepaway soccer.

Our results confirmed that *Fit* often converges to equilibrium states that correspond to narrow regions of the solution space. These regions often do not contain high quality solutions to the task, which results in a poor effectiveness of the evolutionary process. On the contrary, *NS-T* was able to overcome premature convergence to equilibrium states. By rewarding individuals that generate novel collaborations, an evolutionary pressure towards novel equilibrium states is created. The populations still converge, but the focus of convergence shifts throughout the evolutionary process.

As there is a more effective exploration of the solution space, *NS-T* can reach collaborations associated with higher fitness scores more often. The fitness scores achieved by *NS-T* were significantly superior to fitness-based coevolution in all tasks and at almost all stages of evolution. *NS-T* could also evolve a diverse set of solutions for a given task in a single evolutionary run.

Our results demonstrated that novelty search is significantly more effective when working at the team level, instead of at the individual level. *NS-I* could effectively maintain a higher behavioural diversity inside each population and prevented convergence. However, the outcome was not promising in terms of the performance of the resulting complete solutions. Since all populations are very diverse and are constantly changing, it is relatively hard to form effective collaborations, as individuals of a population do not adapt to the other constantly changing populations.

Although the performance of novelty search with agent-level characterisations was low, we contend that this type of characterisations has considerable potential. For instance, they can be used to promote heterogeneity in the collaborations, by rewarding individuals that are different from the individuals of the other populations. They can also be used to identify populations that are evolving similar agents, and potentially merge such populations, similar to emergent behaviour specialisation [12]. In ongoing work, we are studying if and how agent-level characterisations can be used to achieve these objectives.

One limitation of using novelty search is the necessity of providing behaviour characterisations and a behaviour distance function, which are typically domain and task specific. Previous works have shown that novelty-based approaches can be used in a multitude of domains, and that it is often relatively easy to define behaviour distance functions. There are also some efforts regarding the definition of generic behaviour characterisations [2]. As such, we believe that our results will generalise to other domains – even outside the multiagent systems domain. The issues of relative over-generalisation and premature convergence are transversal to a wide range of coevolution problems, and the proposed novelty-based approach can potentially offer a solution to mitigate those issues.

7. REFERENCES

[1] G. Cuccu and F. J. Gomez. When Novelty Is Not Enough. In *European Conference on the Applications of Evolutionary Computation (EvoApplications)*, volume 6624 of *LNCS*, pages 234–243. Springer-Verlag, 2011.

[2] J. Gomes and A. L. Christensen. Generic behaviour similarity measures for evolutionary swarm robotics. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 199–206. ACM Press, 2013.

[3] J. Gomes, P. Urbano, and A. Christensen. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7(2–3):115–144, 2013.

[4] F. Gomez and R. Miikkulainen. Incremental evolution of complex general behavior. *Adaptive Behavior*, 5(3–4):317–342, 1997.

[5] J. Lehman and K. O. Stanley. Efficiently evolving programs through the search for novelty. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 837–844. ACM Press, 2010.

[6] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223, 2011.

[7] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan. Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527, 2005.

[8] H. Moriguchi and S. Honiden. Sustaining behavioral diversity in neat. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 611–618. ACM Press, 2010.

[9] J.-B. Mouret and S. Doncieux. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evolutionary Computation*, 20(1):91–133, 2012.

[10] G. S. Nitschke. *Neuro-evolution for emergent specialization in collective behavior systems*. PhD thesis, Vrije Universiteit Amsterdam, 2008.

[11] G. S. Nitschke, A. E. Eiben, and M. C. Schut. Evolving team behaviors with specialization. *Genetic Programming and Evolvable Machines*, 13(4):493–536, 2012.

[12] G. S. Nitschke, M. C. Schut, and A. E. Eiben. Collective neuro-evolution for evolving specialized sensor resolutions in a multi-rover task. *Evolutionary Intelligence*, 3(1):13–29, 2009.

[13] G. S. Nitschke, M. C. Schut, and A. E. Eiben. Evolving behavioral specialization in robot teams to solve a collective construction task. *Swarm and Evolutionary Computation*, 2:25–38, 2012.

[14] L. Panait. Theoretical convergence guarantees for cooperative coevolutionary algorithms. *Evolutionary Computation*, 18(4):581–615, 2010.

[15] L. Panait and S. Luke. Cooperative multi-agent learning: The state of the art. *Autonomous Agents & Multi-Agent Systems*, 11(3):387–434, 2005.

[16] L. Panait, S. Luke, and J. F. Harrison. Archive-based cooperative coevolutionary algorithms. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 345–352. ACM Press, 2006.

[17] L. Panait, S. Luke, and R. P. Wiegand. Biasing coevolutionary search for optimal multiagent behaviors. *IEEE Transactions on Evolutionary Computation*, 10(6):629–645, 2006.

[18] L. Panait, R. P. Wiegand, and S. Luke. A visual demonstration of convergence properties of cooperative coevolution. In *Parallel Problem Solving from Nature (PPSN)*, volume 3242 of *LNCS*, pages 892–901. Springer-Verlag, 2004.

[19] M. A. Potter and K. A. D. Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[20] M. A. Potter, L. A. Meeden, and A. C. Schultz. Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1337–1343. Morgan Kaufmann, 2001.

[21] A. Rawal, P. Rajagopalan, and R. Miikkulainen. Constructing competitive and cooperative agent behavior using coevolution. In *IEEE Conference on Computational Intelligence and Games (CIG)*, pages 107–114. IEEE Press, 2010.

[22] P. Stone, G. Kuhlmann, M. E. Taylor, and Y. Liu. Keepaway soccer: From machine learning testbed to benchmark. In *Robot Soccer World Cup IX*, volume 4020 of *LNCS*, pages 93–105. Springer-Verlag, 2005.

[23] A. Subramoney. *Evaluating modular neuroevolution in robotic keepaway soccer*. MSc thesis, University of Texas at Austin, 2012.

[24] L. D. Whitley. Fundamental principles of deception in genetic search. In *Foundations of Genetic Algorithms*, pages 221–241. Morgan Kaufmann, 1991.

[25] R. P. Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, George Mason University, 2003.

[26] R. P. Wiegand and M. A. Potter. Robustness in cooperative coevolution. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 369–376. ACM Press, 2006.

[27] C. H. Yong and R. Miikkulainen. Coevolution of role-based cooperation in multiagent systems. *IEEE Transactions on Autonomous Mental Development*, 1(3):170–186, 2009.