Evolving Cooperative Communicative Behaviour

Tahmid Rahman and Shantanu Jain

Abstract

Communication is an essential skill for cooperative tasks, yet neuroevolved agents struggle to develop communicative behaviour. In fact, it has been shown that communicating agents are sometimes outperformed by noncommunicating ones on tasks that ostensibly require communication. In our paper, we explore mechanisms which could aid agents in developing communication through evolution. Specifically, we examine the idea of an opt-in mechanism whereby agents communicate only on request – the idea being that by eliminating communication when agents deem unnecessary, evolution is presented with less information to understand and organise. Spurious input would serve only to distract evolution and cause it to explore unconstructive lines of development, hence we expect to find opt-in controllers evolving better solutions faster. Additionally, we explore an extension of the opt-in mechanism wherein agents receive fitness penalties for each communication request made. The idea behind this is to incentivise agents to communicate efficiently, or more colorfully, to incentivise them to not distract evolution. We test the performance of agents with opt-in controllers on a task where communication and cooperative behaviour is required only half of the time; the rest of the time, agents are assigned fitness based on actions independent of each other. Their performance is compared to those of agents which communicate constantly and to noncommunicative agents. We do indeed find that opt-in controllers outperform the other kinds, although contrary to our expectations, they underperform the other controllers through the early stages of evolution.

1 Introduction

Much of what occupies humanity's attention can be described as cooperative multiagent problem solving, namely, politics and sport. Thus perhaps there is an argument to be made that developing artificial adaptive multiagent systems that are capable of solving such problems is of innate interest. Anyhow, the applications of cooperative multiagent systems are diverse and many [4].

The most robust solution that humans have found to solving such problems is communication. Politicians will discuss, debate and deliver their votes. Athletes will shout, signal and make suggestive eye contact. The importance of the role of communication in cooperative tasks has been highly studied in artificial life experiments [6]. Communication has been shown to be necessary or offer great advantages to solving specific tasks, for instance, tasks that involve agents assembling in specific configurations or requesting help from one another.

Yet we find that neuroevolved agents with communication mechanisms perform exceedingly poorly – so poorly, in fact, that they are outperformed by noncommunicating agents on tasks involving cooperation. Thus in the ensuing discussion, it will serve us well to understand why communication suits humans so well, yet trips up neuroevolved agents as they are currently implemented.

In the paper Coevolution of Role-Based Cooperation in Multiagent Systems, Yong and Mikkulainen [6] explore how to evolve cooperative controllers to solve a predator-prey capture task. Such

tasks involve a group of agents, called predators, attempting to hunt down a pre-programmed prey agent through cooperative and coordinated action. One of their chief and most surprising findings was that communicating agents were completely outperformed by noncommunicating agents on this task; as seen in Figure 1, the noncommunicating agents learned to solve the predator prey task in less than half the number of generations that the communicating agents took.

A reason that this is so surprising is that communicating agents were technically completely capable of mimicking the behaviour of noncommunicating agents in this case. If evolution had simply learnt to set the weights corresponding to the communicative inputs within the neural network architecture to zero, the communicating agents would be functionally equivalent to the noncommunicating ones.

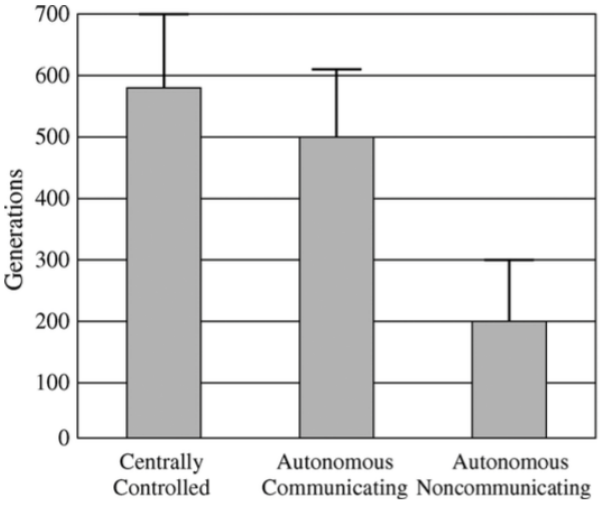

Figure 1: Noncommunicating agents evolve solutions in under half the time [6].

Understanding how it is at all possible that noncommunicating agents solve a task requiring cooperation proves to be informative. The key idea is that an agent is capable of determining information about another agent's actions or position indirectly, through their effect on the environment. For instance, in Yong and Mikkulainen's predator-prey task, the direction in which the prey runs away implicitly reveals actionable knowledge about the other predators. This use of information implicit in the environment is termed stigmergy.

It is hard to believe that stigmergy is so powerful, yet this phenomenon proves quite robust. Stan Franklin gives multiple real life situations in which coordination of multiple agents is achieved without communication [2]. For example, Ghenghis, a six-legged insect-like robot, learned to walk with simply three sensors, two of which indicated pain and one of which indicated pleasure. There was no element of coordination, communication or signalling between the six legs.

In examining the stigmergic approach, the most minimal approach towards communication, one can see what pitfalls evolution steps into with explicit communication. Franklin points out that communication comes with built in computational costs, in that it needs more "architecture and intelligence" [2]. Yong and Mikkulainen similarly highlight that a network capable of explicit communication will require significantly more parameters (such as weights and hidden units). There are very many more moving parts that a network will have to learn to tune [6]. When humans apply communication to problem solving, this is not usually a consideration at all.

A related argument is that evolutionary algorithms (for instance NEAT [5]) have no knowledge of what the input really represents. Through evolution it simply guesses how to connect the inputs to the outputs with sensible weights. The overhead of additional inputs and outputs posed by explicit communication simply presents evolution with too much information to understand and organise through its methodical guesswork alone.

This leads to two lines of thought. One, some forms of representing information may prove significantly easier. It's not at all obvious exactly what an agent should be communicating and

what form this communication should take. With each additional dimension of communication, the task becomes that much harder to solve. Perhaps some form of principal component analysis should be applied first. This problem has been studied in the literature. For instance, Jim and Giles [3] investigate a version of the predator-prey task where agents are allowed to communicate arbitrary binary strings – in essence, the freest form of communication. Closer to home, Chang and Worlanyo [1] study a swarm foraging task where agents communicate through frequencies and attempt to learn to interpret these frequencies in meaningful ways.

Second, given that explicit communication necessarily involves overhead, how do we mitigate the effect of all this extra information. Yong and Miikkulainen describe how on their prey-predator task, communicating agents biased towards not communicating perform comparably to the non-communicating agents. Again, we notice that humans are very good at filtering through extraneous information, helping explain why communication is much more robust as a human strategy. The idea then is that we implement a mechanism to help agents filter information, and tune out unnecessary noise.

In our paper we explore the idea of allowing agents to opt-in to communication. Then we have given the agents the power to deem when it is necessary to communicate and when it is not. Then, by eliminating communication when deemed unnecessary, evolution is presented with less information to understand and organise, helping it focus on exploring constructive lines of development.

Thus when compared against controllers that receive a constant stream of communication on a task where communication is not always necessary and distracts from the objective, we hypothesise that controllers with an opt-in mechanism will achieve superior performance. As the opt-in mechanism reduces the amount of information evolution needs to process, we also predict that these controllers will evolve better solutions faster. Compared to noncommunicating controllers, opt-in controllers will perform better, while constantly communicating controllers will perform worse or at the same level, supporting the findings of Yong and Miikkulainen. Finally, we experiment with controllers that are penalised for requesting communication, and we conjecture that these controllers will learn not to communicate during the parts of the task communication is not required.

2 Experiments

2.1 World and Physics

Two agents are located in a circular world of diameter 100 units. Agents move around the world with a maximum velocity of 1 unit per timestep. Agents control themselves through their choice of acceleration, which is capped at a maximum of 0.1 units per timestep square. When an agent collides with the boundary of the world, its velocity is reset to zero. This results in a physics wherein agents have momentum, so their current velocity does constrain their actions.

2.2 Fitness

At each timestep agents receive fitness in the range $[0, 1]$. An agent's fitness score is calculated by averaging fitness over the lifetime of the agent. The world alternates between two modes, individual mode and team mode, at regular intervals of 100 time steps. The two modes are distinguished by the fitness functions applied to the agents whilst the world is in that mode, as specified by Table 1.

Individual mode:	$1 - d(\text{agent}, \text{goal})$
Team mode:	$0.75 \cdot \lvert v_a \cdot v_b \rvert + 0.25 \cdot d(\text{agent a}, \text{agent b})$

Table 1: Fitness functions for individual mode and team mode, where d is normalised distance.

In the individual mode, each agent has a goal point that it is rewarded for being close to. For instance, in Figure 1, the red agent is rewarded for being close to the orange point, while the blue agent is rewarded for being close to the light blue point. In the team mode, agents share fitness based off of the absolute value of the dot product of their velocities and their distance from each other.

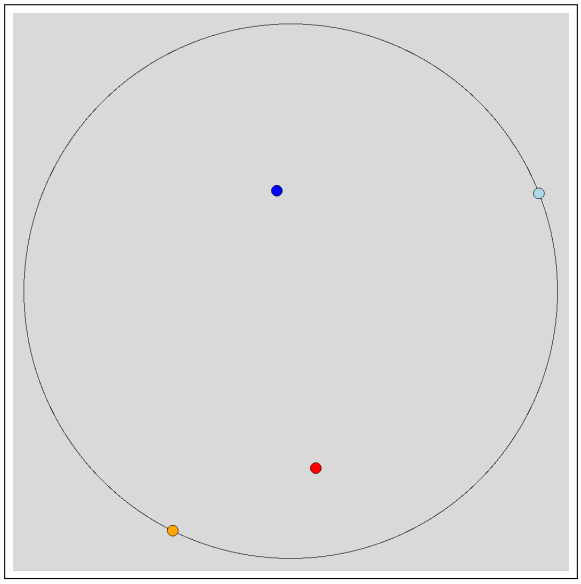

Figure 2: World in the individual mode

The goal point for each agent is chosen each time the world switches from team to individual mode. The point is randomly chosen on the boundary of the world, although to reduce the element of luck and the variability of fitness, we ensure that the goal point is between 40 and 65 units away from the agent at the moment it is chosen.

The best way to think of fitness in team mode is simply as a function that is dependent on both the agents' velocities relative to each other and their positions relative to each other. Going a little more in depth, we see agents are rewarded for moving in the same axis as each other (either parallel or antiparallel with the other), and for being as far apart as possible.

Additionally, penalty controllers, discussed in later sections, received a fixed penalty of 0.05 in fitness each timestep they chose to request communication.

2.3 Evolution

Autonomous heterogeneous neural controllers were evolved using NEAT [5]. NEAT, or NeuroEvolution of Augmenting Topologies, is a method for evolving artificial neural networks that acts differently than other genetic algorithms; instead of evolving with a fixed topology, NEAT determines the network topology in addition to the weights. It does so by slowly evolving increasingly complex neural network architectures. The evolution, as usual, prioritizes network architectures and weight set-ups that yield higher fitness scores.

The reason we evolved heterogeneous controllers as opposed to homogeneous ones is we hoped to develop pairs of controllers with strategies that could be asymmetrically dependent on the other's actions: for example, a pair of controllers could learn to let one be a leader, and the other a follower.

Coevolution of the heterogeneous controllers (which we'll call A and B) was implemented as follows. In each generation, an A controller is tested by pairing it with the ten best B controllers from the previous generation. Rather than using the average fitness across all pairings, the fittest pairing is used to determine the fitness of the A controller. B is then similarly evolved. The idea behind implementing coevolution in this way is that we want to evolve pairs of controllers that are attuned to their partner, as we wish for them to develop highly specific communication strategies. A controller with a strategy dependent on its partner responding a certain way will not perform well when its fitness is averaged over all pairings. A good analogy for this coevolutionary process is that of a dance: a controller's fitness is determined by its performance with the partner of its choice. We used a population size of 30, evolved for 40 generations in two different runs. The fitness of each pairing was determined by simulating the world for 10,000 timesteps.

2.4 Controllers

We tested four types of controllers, detailed as follows. Silent controllers operated without communication. Constant controllers had constant communication. Opt-in controllers communicate through an opt-in mechanism that is implemented as follows. The controller has a dedicated output that signals a request for communication. When a controller makes a request, both controllers receive communicative inputs. Penalty controller are opt-in controllers but evolved with a fitness penalty for each request to communicate. They incur a penalty of 0.05 to their fitness for that timestep each time they decide to signal.

Table 2 details the inputs given to each kind of controller. All controllers receive as input their own positions and velocities, the current mode and the location of the goal during the individual mode. Silent controllers are restricted to just these inputs. Constant controllers always receive the positions and velocities of the other agent. Opt-in and Penalty controllers have additional inputs that correspond to the position and velocity of the other agent upon request. The decay input is a measure of time since the last communication request made. A signal resets its value to 1, after which it decays at a rate 0.9 with each time step.

Figure 2 is a visual representation of an opt-in controller that emphasises the idea that the world selectively filters input to the controller dependent on the signal output.

Inputs	What it Represents	Silent	Constant	Opt-In	Penalty
x, y	position of self	✓	✓	✓	✓
vx, vy	velocity of self	✓	✓	✓	✓
mode	current game mode	✓	✓	✓	✓
rx, ry	coordinates of individual goal	✓	✓	✓	✓
cx, cy	position of other agent	–	✓	o	o
cvx, cvy	velocity of other agent	–	✓	o	o
decay	measure of time since last communication	–	–	✓	✓

Table 2: Inputs for each controller. A checkmark indicates the controller had the input, while a dash indicates the controller did not have the input. An o indicates the controller opted into the input.

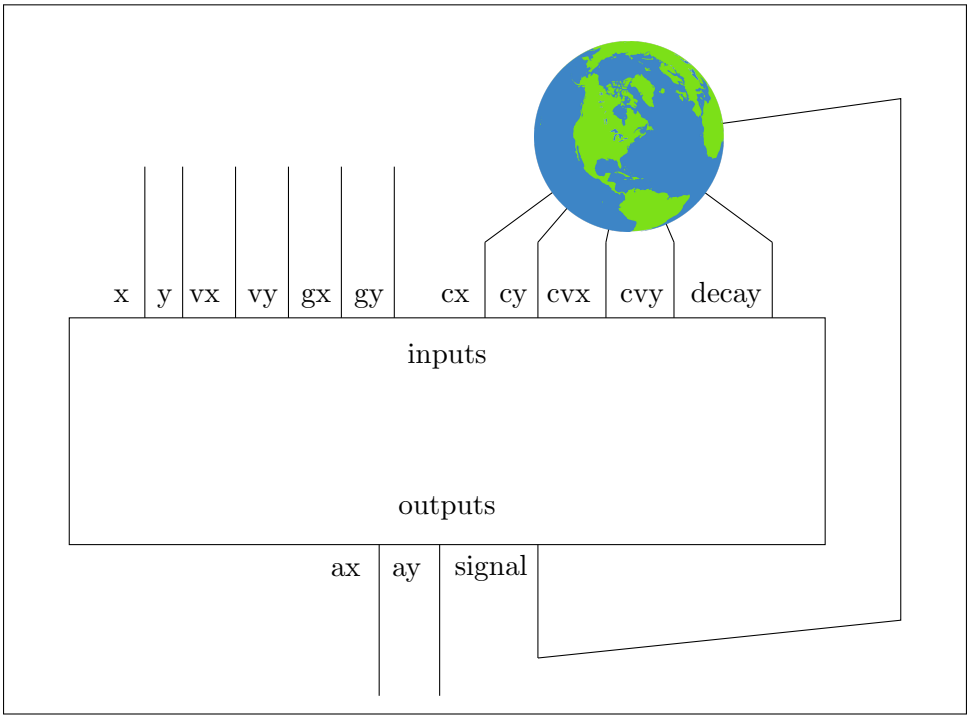

Figure 3: Inputs and outputs for an opt-in controller.

Table 3 details the outputs of each controller. Agents traverse the world by outputting acceleration along the x and y directions. We implemented an acceleration based movement mechanism as it imbues agents with momentum. Apart from being more realistic, this also makes the task harder for the agents as their current velocity constrains their ability to change position. Opt-in and penalty controllers have the additional signal output. When signal is greater than 0.9, the world interprets it as the agent requesting communication.

Outputs	What it Represents	Silent, Constant	Opt-In, Penalty
ax, ay	coordinate accelerations of agent	✓	✓
signal	a request for communication	–	✓

Table 3: Outputs for each controller.

3 Results

3.1 Overall Fitness Scores

For each run, we tested the fittest pairing from each generation over the course of 30 lifetimes, each of 10,000 timesteps. We compare and contrast the data gathered from each of the runs, rather than averaging the two, so that we have a better sense of how the controllers evolved relative to each other. Note that in testing, penalty controllers were not penalised for communication requests, so the fitnesses are in fact comparable.

We analyze the first run. As can be seen in Figure 4, the opt-in controller achieved the highest overall fitness. The constant controller fared roughly equally to the silent controller. The penalty controller scored similarly to the constant and silent controllers, albeit a bit worse. Furthermore, the silent controller was quickest to evolve an effective solution to our task.

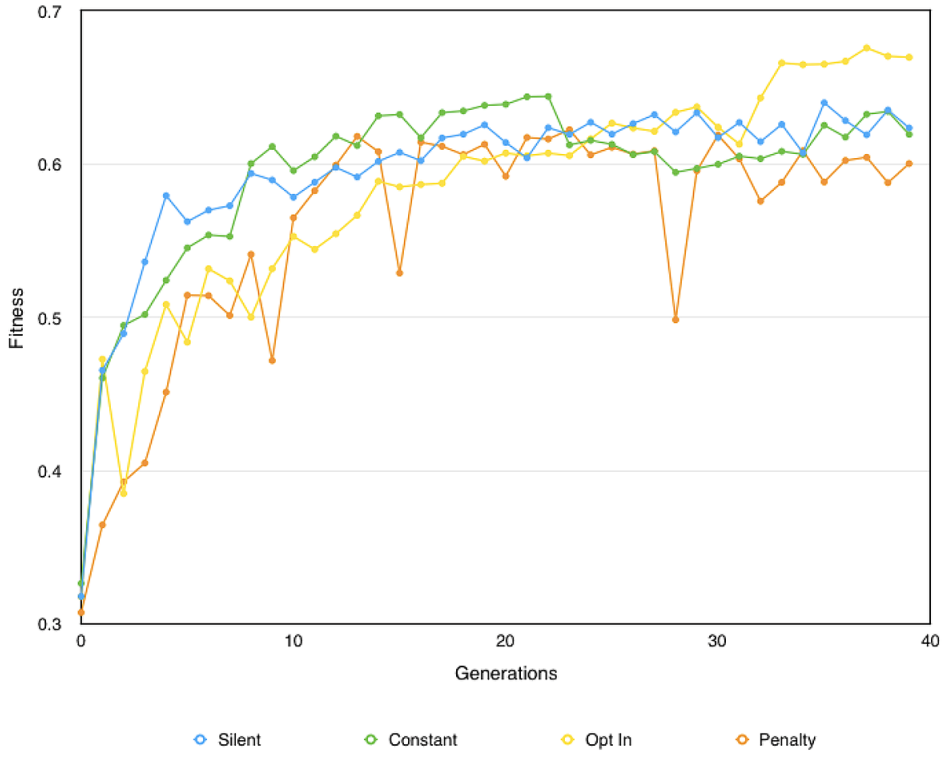

Figure 4: Results from the first run.

In the second run, as can be seen in Figure 5, the results are very similar. Once again the opt-in controller achieves the best performance and the silent controller evolves better solutions faster. The penalty controller once again performs the worst.

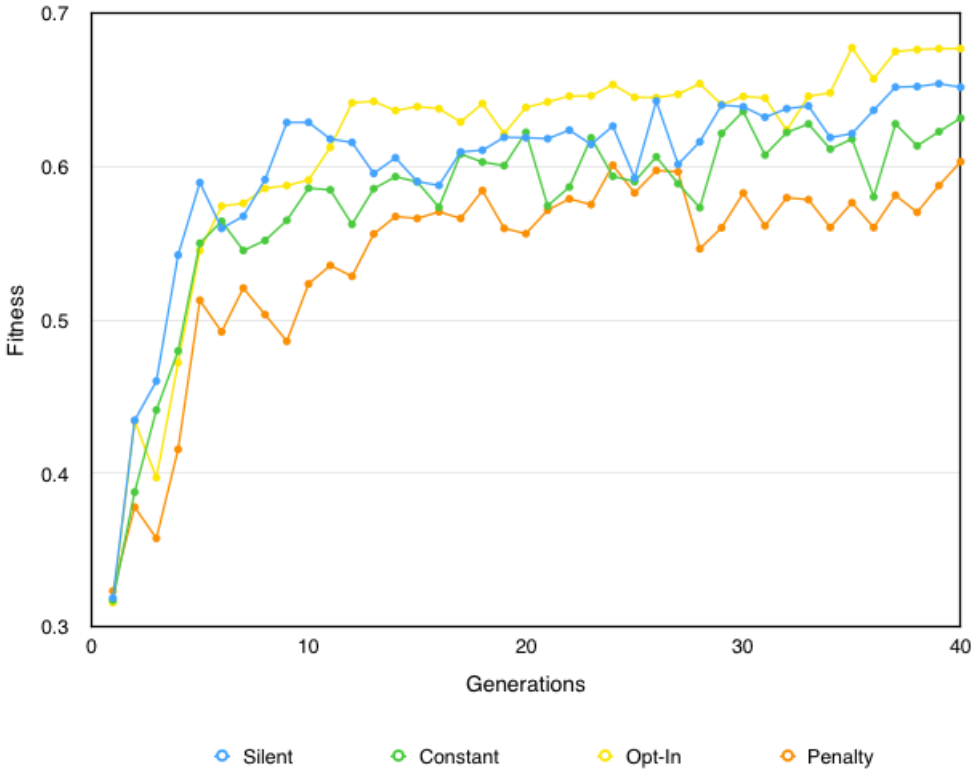

Figure 5: Results from the second run.

3.2 Analysis of Behavior

In the first run, the controllers across the board were able to navigate close to the goal points during the individual mode. However, during the team mode, we observed varying behavior. The silent controllers consistently settled into a clockwise loop during the team mode. This tacit agreement often performed alright on the component of the fitness related to velocities. However, the agents with these silent controllers tended to get very close to each other. The opt-in controllers generally circled each other but kept a large radius, and signalled almost constantly, about 90% of the time in both modes. The constant controllers displayed similar behavior to the opt-in controllers. The penalty controllers often acted chaotically, without any signs of higher purpose. Interestingly, the penalty controllers gradually learnt to never signal over the course of the run, functionally performing as silent controllers. This is likely because the penalty they received was around the margin of difference between their fitness and that of the silent controllers.

In the second run, controllers once again exhibited similar behavior during the individual mode, being able to navigate close to but not always exactly towards the goal points. Once more, we observed diverse behaviour in the team mode. The silent controllers would hug the perimeter,

often ensuring separation between the two. Opt-in controllers circled each other while keeping a large radius, with one of the agents consistently staying closer to the center. Constant controllers often settled into circling motions, but performed poorly in terms of the distance between each other. The penalty controllers performed very badly comparatively, being outperformed at every point in evolution, though this time the controllers from the final generation signalled most of the time in both game modes.

There was no clear evidence that the signalling corresponded to the game modes, which is what we were testing for. In fact, with the penalty controllers in the second run, in earlier generations sometimes the opposite of the expected signalling behaviour occurred, with more signalling in the individual mode than the team mode. The signalling behaviour changed pretty drastically between generations, even at the end of the run. This may explain why the penalty controllers performed poorly through the run.

4 Discussion

We find that all agents perform respectably on the individual mode, always heading towards their goal, and usually staying within the vicinity of the goal once reached. In the team mode, on the other hand, we find none of the controllers achieve clearly dominant behaviour. However, the results do support some of our hypotheses. Opt-in controllers did in fact outperform both silent and constant controllers. As predicted, and in support of previous results, we found that silent and constant controllers performed comparably, with silent controllers having a small upper hand.

The fact that all controllers performed largely within the same fitness range indicates that our task may need revision, in order to better distinguish good solutions from average solutions. Additionally, the task was designed to necessitate communication half the time. However, the fact that noncommunicating controllers were able to stumble upon a tacit strategy that performed well may indicate that the task is largely solvable without communication, and so may need revision. If this were the case, then giving the controllers the ability to communicate at all would be burdensome. Derandomising the task (although we did limit the degree to which luck could affect fitness) may also prove more robust results, although preliminary experimentation indicated that these systems were much easier to game – that is, achieve good results without communication. For instance, we saw agents predicting the location of the goals.

The other important highlight of the results is that silent controllers evolved faster than opt-in controllers in both runs. This directly relates to the idea behind this experiment: the added infrastructure required by communication gives the neural network a lot more information to have to try to interpret. Opt-in controllers required even more infrastructure than constant controllers (in terms of inputs and outputs), and this fact most likely slowed their evolution. This in fact suggests that opt-in controllers could have achieved a bigger differential in performance if we had let them evolve for more generations.

We observed that penalty controllers performed consistently the worst out of the different types of controllers. A reason for this could be due to our choice of the fitness penalty incurred for communication. The difference in performance between penalty controllers and constant and silent controllers was on the order of the penalty incurred for communication per timestep. Therefore a penalty controller choosing not to communicate and becoming functionally equivalent to a silent controller would dominate a communicating penalty controller. We didn't lower the fitness penalty incurred as that would put the penalty within the fitness range produced by random variation

between lifetimes. This suggests that we need a more robust fitness function that produces greater difference. Alternatively, we could penalise communication more creatively. For instance, if a controller chooses to communicate, we compare its outputs with and without communication, and penalise it proportional to the difference in the outputs. This could teach the controller to request communication only when it is actionable. However, this could also just cause it to behave erratically every time it requests communication.

It is also possible that our choice of implementation of coevolution was not optimal. Rather than pick a partner at every timestep, it may have been better to evolve specific pairs of agents from the very start. This would also have been less computationally intensive, and would have allowed us to carry out more runs, for more generations. Consistent results over more runs would help clarify the significance of these results. However, we were constrained by the limitations of the NEAT library we used for evolution.

Finally, our choices for the communicative inputs may have been overbearing. It is possible for example that not providing velocities or providing relative positions could have improved performance. However, this relates back to the vein of thought of Jim and Giles, Chang and Worlanyo, as discussed in the introduction. It is hard to predict a priori the type and amount of information that will be most useful for communicating agents.

5 Conclusion

We find that the idea of allowing controllers to decide when they want to communicate is an interesting one. Humans excel at distinguishing signal from noise, and giving neuroevolved agents a similar ability to tune out helped them achieve superior performance on a task that only sometimes required communication.

6 Bibliography

1. R. Chang and S. Worlanyo, "Evolving Swarm Communication with NEAT" Dept. Comp. Sci., Swarthmore College, Swarthmore, PA., 2015.

2. S. Franklin, "Coordination Without Communication," Inst. Intell. Syst., Dept. Math. Sci., University of Memphis, Memphis, TN., 1996.

3. C. Giles and K. Jim, "Talking Helps: Evolving Communicating Agents for the Predator-Prey Pursuit Problem," Artificial Life, vol. 6, no. 3, pp. 237-254, Jun. 2000.

4. S. Luke and L. Panait, "Cooperative Multi-Agent Learning: The State of the Art," Autonomous Agents and Multi-Agent Systems, vol. 11, no. 3, pp. 387-434, Nov. 2005.

5. R. Miikkulainen and K. Stanley, "Competitive Coevolution through Evolutionary Complexification," Journal of Artificial Intelligence Research, vol. 21, no. 1, pp. 63-100, Jan. 2004.

6. R. Miikkulainen and C. Yong, "Coevolution of Role-Based Cooperation in Multiagent Systems," IEEE Transactions on Autonomous Mental Development, vol. 1, no. 3, Oct. 2009.