Morphogenetic Engineering For Evolving Ant Colony Pheromone Communication

Neil Vaughan[1]

Abstract. This research investigates methods for evolving swarm communication in a simulated colony of ants using pheromone when foriaging for food. This research implemented neuroevolution and obtained the capability to learn pheromone communication autonomously. Building on previous literature on pheromone communication, this research applies evolution to adjust the topology and weights of an artificial neural network which controls the ant behaviour. Comparison of performance is made between a hard-coded benchmark algorithm, a fixed topology ANN and neuroevolution of the ANN topology and weights. The resulting neuroevolution produced a neural network which was successfully evolved to achieve the task objective, to collect food and return it to the nest.

1 INTRODUCTION

This research has developed a model of ant colony swarm intelligence behaviour. The novel aspect is that behaviour of pheromone navigation was not hard coded, as in most implementations, but has evolved using artificial neural networks (ANNs) and an implementation of neurovevolution. Compared to previous research which failed to evolve standard and fixed topology ANNs for ant behaviour (Collins & Jefferson, 1990a), this research produces successful evolution and applies a more comprehensive neuroevolution methodology including complexification and augmentation of ANN topology and weights, as described by NEAT (Stanley, 2004).

Inspired by biological ants, this research aims to provide insights to advance understanding of how pheromone communication evolved in biological organisms. Application of neuroevolutionary computational modelling provides a useful analogy to how brains may have evolved to produce biological organism behaviours.

There are many long standing open questions regarding the evolution of altruism, related to how any why the evolution of cooperation emerged among closely related individuals [Hamilton 1964]. Worker ants (Formicidae) are a perfect example of altruism, as they collect food for the good of the swarm but they get no individual rewards. This computer simulation method can provide new insights into altruism because each colony is only assessed by its fitness as a whole, not that of individual ants.

Pheromone trails can be seen as social memory or swarm memory used by all agents in the colony. The problem is called central place food foraging which is an optimisation problem. The aim of foriaging is to collect as much food as possible and return it to the nest. An ant's food collection consists of two phases: the search for food and retrieval of food back to the nest.

In this respect the problem relates to the new field of morphogenetic engineering (ME). In this task, the core challenge posed by ME is a reverse engineering one: How can the ants' micro-rules be inferred from the system's macro-objectives? (Doursat et al., 2013). In this case the macro objective is to optimise fitness of the swarm by using swarm communication, but the micro rules for each ant to achieve that were not provided in this system and needed to evolve autonomously with no prefabricated design.

2 RELATED WORK

Literature on pheromone communication is described by various key words: ant evolution simulation pheromone, central-place foraging algorithm (CPFA), pheromone recruitment (Letendre and Moses, 2013).

The core interest of this work is how ant pheromone communication can be evolved in a computational model. There have been some interesting works attempting to evolve ant pheromone communication, and others evolving swarm communication in general which is related closely enough.

A milestone early attempt to use a computer simulation to evolve ant foraging strategies using pheromones which resemble behaviours of biological ants was AntFarm (Collins & Jefferson, 1990a). AntFarm implemented an early form of neuroevolution, which was used to evolve the ANNs which learn behaviour for effective ant pheromone communication (Collins & Jefferson, 1990b). Neuroevolution methods in AntFarm evolved both the ANN connectivity pattern (topology) and weights of the ANN which were under genetic control in a genotype. Limitations were that: (1) AntFarm did not successful evolve any cooperative foraging which was the main objective. (2) A basic, conventional ANN was used, when compared to the wider range of operators, sigmoids and activation functions with complexification as used in more recent neuroevolution models such as NEAT (Stanley, 2004). (3) The number of neurons and connections were not under genetic control.

The first research to evolve Ant pheromone foraging was by Panait and Luke (2004).

More recently, Beem (2017) attempted to use NEAT to evolve the controller for individual agents in a swarm. However the methods failed to produce any ability for agents to find food, or communicate, or exhibit any swarm intelligence whatsoever. The most advanced behaviour that his agents ended by evolving was to walk in circles. Perhaps that was due to the coordinate system used, or a lack of random or sin wave inputs. The inputs to the NEAT ant controller included the ant's own position; the

[1] Dept. of Computer Science, Univ. of Chester, UK. Email: n.vaughan@chester.ac.uk.

intensity of pheromones at its location; whether or not the ant is carrying food at a given moment and the distance to the nearest food from two different points on the ant (for triangulation). The NEAT outputs are the agent's forward movement speed, its steering direction and the intensity of the pheromones it leaves behind. All agents within a swarm have the same neural network as controller.

Yong and Miikkulainen (2009) found that for cooperative tasks such as chase and evade, evolving several autonomous, cooperating neural networks to control a team of agents is more efficient and robust than evolving a single centralized controller. This potentially may apply in ants where two distinct roles are required – searching for food and returning to the nest.

Other attempts to evolve Swarm Intelligence using NEAT have failed, for example Chang & Worlanyo (2015) didn't see any communication being evolved. In other work, to some extent evolving swarm communication has succeeded (Floreano et al., 2007, Marocco and Nolfi., 2003, Yong and Miikkulainen., 2009). Rawal et al. (2012) successfully evolved cooperative communication between a group of predators who can only catch prey by communicating information codes to each other. A related work has evolved ants nest site localisation (Marshall, 2003).

Ant algorithms are generally most widely known through the wide literature on optimisation problems with ant colony (ACO) by Dorigo et al., (2006). Differing from this research, ACO algorithms conventionally must be hard coded by a designer and not evolved automatically.

Letendre and Moses (2013) used genetic algorithms to show that ant foriaging is improved in random food distributions and using both pheromone and site fidelity foriaging strategies. However their GAs were used only to adjust a set of parameters affecting behaviour, not to learn the behaviours themselves, which were hard-coded and pre-existing.

3 STATE OF OUTSTANDING PROBLEMS

There are some state of art current outstanding problems specifically within the evolving pheromone communication, some of which are addressed in this work.

Collins & Jefferson (1990a) suggested future work should involve: (1) a systematic study of the effect of food distribution on the evolution of foraging strategies, testing the model of Johnson et al. (1987). (2) evolution of foraging strategies that are strongly affected by competition to see if competitor colonies sharing a single environment will interfere with each others strategies, disrupting communication by overwriting pheromone or misleading trails – which is related to Anti-pheromone which was later separately used by Panait and Luke (2004). (3) Investigate previous suggestions that pheromone evolution requires incremental changes to vary the environment, slowly making foraging more difficult over time.

Future work can also focus on the limitations of Panait and Luke (2004) which was suggested as future work. (1) When using multiple food sources which decay when eaten, this results in a dynamic changing environment and this makes pheromone evaporation more important. (2) How does pheromone navigation change with introduction of predators. (3) Future work can investigate ants which can produce more than 2 pheromones, so the ants can also learn complex tours with multiple way-points and self-intersecting paths.

4 SYSTEM COMPONENTS

There are five components in the system which occur when an ant makes a move.

1. Pre-computed Inputs (ant sensors).

Ants have 13 input sensors: (1, 2) the location within the 9 adjacent cells (Moore neighbourhood) of the highest pheromone, (3, 4) the location within Moore neighbourhood which is closest to the nest, given by a 'compass sensor'. (5, 6) location within Moore neighbourhood of food. (7, 8) the direction of the ant's previous move, (9, 10) a direction picked at random, (11) a Boolean indicating whether the ant is currently carrying food, (12) a random number, (13) a fixed value of 1 (Bias). These are referred to as the pre-computed inputs and they remain the same even when the controller is changed (BM1, ANNs, NEAT).

Having a compass avoids the requirement to use two different pheromones. Compass is calculated by Pythagoras theorem using the x and y differences between ant and nest. In a grid system following the compass does not produce a direct path, it results in diagonal motion followed by perpendicular motion.

All of the 5 direction pre-computed inputs are represented in a consistent manner using two variables for x and y. These represent the change required in the ant's current x and y coordinates. These variables can be positive, negative or neutral. If both are neutral the ant would stand still (which would never be beneficial when foraging). If both were negative, the ant would move diagonally towards the origin (NW). With this method, the two variables can encode any direction within the 9 squares of the ant's Moore neighbourhood. If the ant chooses to follow the compass, it would then ignore the pheromone and vice versa.

2. Controller.

The controller is a 'black box' brain which decides the animal behaviour at timestep t, based on the pre-computed inputs from the ant's sensors. The experiments were repeated using different controllers: a hard-coded benchmark (BM1), a fixed topology neural network and neuroevolution by adjusting the topology and weights of an ANN.

3. Outputs.

The resulting output of the controller determines the direction in which the ant moves.

4. Post move local updates.

After each ant has moved, a number of post-move local updates are automatically applied. (1) If the ant is now standing on food and isn't carrying any, it automatically picks food up. (2) If the ant is carrying food, pheromone is deposited with strength inversely proportional to the time since collection. (3) If an ant is already carrying food and is now standing on a nest, it automatically drops the food. This representation realistically assumes that biological ants already could pick up and drop food before they evolved pheromone communication. These tasks are regarded as automatic responses which we assume have been learnt previously.

5. Global updates

After a full iteration, when all ants have finished making a move, a global update is triggered in which all pheromone is evaporated (decremented). A number of different evaporation

rates including decrementing and various percentage reductions were tested to identify how evaporation rate affects the ability to evolve navigation controllers.

5 THE MACHINE LEARNING TASK

The given inputs and expected outputs were kept strictly equal for all tested controllers. Therefore here we can formally define the machine learning task based on the relationship between inputs and outputs of the controller. This is critical step because small changes to the representations of input and output can make big changes to the difficulty of the task for the controller to learn.

Inputs:

In total the task has 13 inputs: 10 (5 pairs of) input direction variables, 1 boolean input, 1 random number input and 1 fixed value (Bias). There are 2 outputs: x and y.

Of the 13 inputs 10 inputs are positional change inputs. These are in 5 pairs of x and y, relative to ants current position, to reach the optimal square within Moore neighbourhood for the 5 pre-computed inputs: food, pheromone, compass, same-move or random-move. These all have three possible values -1,0,1.

```
Closestnestx
Closestnesty
Foodherex
Foodherey
Highestpheromonex
Highestpheromoney
rand_x
rand_y
same_x
same_y
```

For food and pheromone, 0,0 only occurs when none is found, which means that there is no need for having separate Booleans indicating food and pheromone presence. For compass, 0,0 only occurs when standing on the nest (in which case compass would not be useful as the ant would not be carrying food because it would have been dropped automatically).

There is one random number input called r. This is independent of random direction inputs. This is required so that ants can randomly determine when to move randomly.

```
r
```

A Boolean is included to represent whether food is currently being carried. This is an important flag because it defines one of two current states: (1) searching for food, or (2) bringing food back to nest. This information is not available in other inputs.

```
carryingfood
```

There are only two outputs. They represent the relative step the ant will take on this timestep. They can be a value from the set {-1,0,1}. Therefore the output of the controller purely determines the position of the ant's next move, which has 3^2 options, one for each square in the ant's Moore neighbourhood.

```
output_x
output_y
```

6 BENCHMARK ALGORITHM BM1

The developed system included designing a custom developed hard-coded benchmark algorithm (BM1) for pheromone based food foriaging behaviour, shown as pseudocode in Fig. 1. The BM1 algorithm was used in this research as a comparison or gold standard to assess the performance of the fixed topology and evolving ANN algorithms.

The benchmark BM1 does produce efficient foriaging behaviour and also demonstrates that the pre-computed inputs provide all required information to complete the foriaging task. The benchmark performance was measured and used to evaluate the performance of ANN driven behaviour controllers which later evolved. The pseudocode gives a description of what happens for each ant to decide which direction to move in at each timestep. This implements two modes: searching for food, and retrieving food based on the `carryingfood` Boolean flag.

```
if (carryingfood){
  //follow compass to nest
}else if (food in neighbourhood){
  //step onto the food
}else if (pheromone in neighbourhood){
  //step onto strongest pheromone
}else if (rand%100<90){
  //continue previous direction
}else{
  //use a completely random direction.
}
```

Fig. 1. Pseudocode for the Benchmark Algorithm (BM1).

In the BM1 benchmark, when a random direction has been chosen, on consecutive timesteps, the direction has 90% chance of remaining constant. This means that ants travel largely in straight lines, broken by abrupt changes on 10% of steps. This causes the ants to more effectively spread out and cover the whole grid more quickly. The main benefit is that ants then have a much higher chance of running into an existing pheromone trail. This outperforms total random movement, whereby ants often retrace their steps in consecutive turns which results in a lack of general directed movements. Also the random direction is chosen from an 8-square Moore neighbourhood – so that standing still is never chosen as it would have no benefit.

It can be seen that efficient pheromone communication (BM1) can be captured in this simple pseudocode which consists of only 5 IF statements, plus the defined actions to perform within each condition. The machine learning task is to replicate the behaviour of these 5 IF statements and associate the correct actions with each case, by using only the 12 given inputs. This summarises the difficulty of the learning task. If the machine learning fails, it must be because the IF statement structure was too complex to learn, or the actions were not associated with the

correct conditions. The BM1 already demonstrates that the given 12 inputs are satisfactorily informative to complete the foraging task.

In the event of machine learning failure, that could be investigated in terms of machine learning complexity, rather than anything specifically about the foriaging task, because it could be assumed that other machine learning tasks with the same level of complexity would equally fail to be learnt.

7 RESULTS OF BM1 IN FIXED LEVELS

For a comparison between controllers (BM1, ANN, Neuroevolution), one fixed level was used. The obstacles and food were located in the same places. That ensured that each controller was subject to the exact same challenge. The fixed level is shown in Fig. 2.

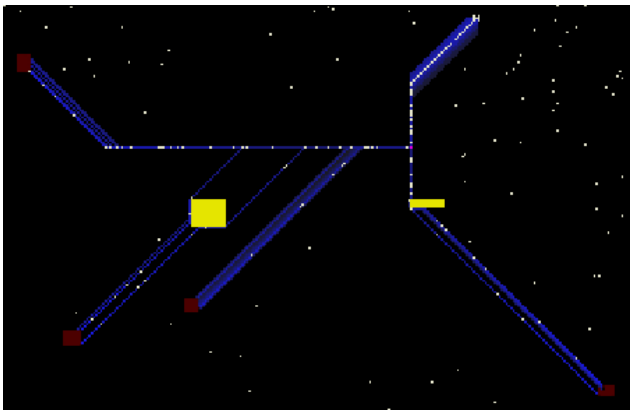

Fig. 2. This shows the layout of the fixed level used for evaluating and comparing a variety of controllers.

When the ant colony was controlled by the BM1 algorithm and foriaged within this fixed layout level (Fig. 2.), the ant colony makes very consistent progress every time it is run (Fig. 3.). The small variations are due to the random movement of ants, taking slightly different times to first discover food sources before they are subsequently attended by large recruited swarms.

In total the fixed level happens to have 6630 foods. On a typical run in this fixed level, as those shown on the graph Fig. 3, BM1 collected the first food after 106 timesteps. At 2500 timesteps, 3143 foods were collected. By the time the run was halted at the 5000th timestep, food was still being actively collected, in total 4852 foods had been collected, so 1778 foods remained uncollected.

The behaviour of the BM1 can be further analysed by looking at Fig 4. which shows how often each direction was chosen. Standing still is the rarest move and diagonal bottom-right to top-right the most frequent. Also Fig. 5. shows how often each of the 5 IF statements from the pseudocode (Fig. 1) were triggered. Continuing in the same direction is the most frequent action and stepping onto food is the rarest action.

8 FIXED NEURAL NETWORK TRAINING

Using a typical run of BM1 in a fixed level, as in Fig. 1. a training set for an ANN was produced.

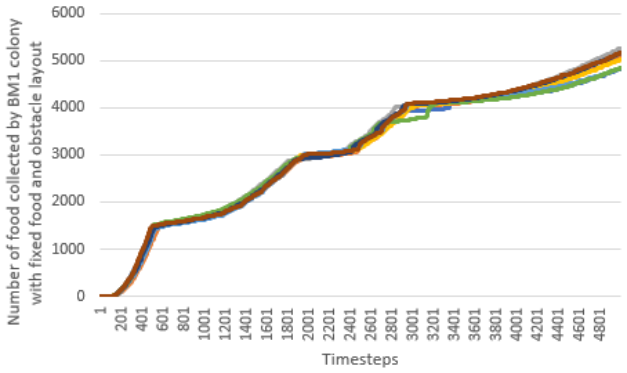

Fig. 3. Consistent results of BM1 run 8 times on a fixed level.

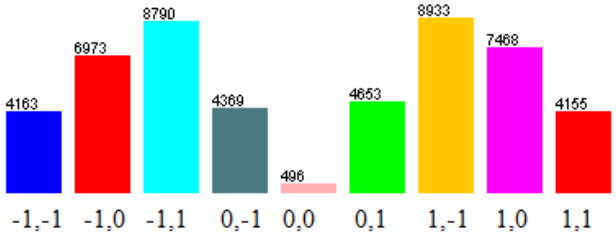

Fig. 4 How often each Moore neighbourhood direction was chosen.

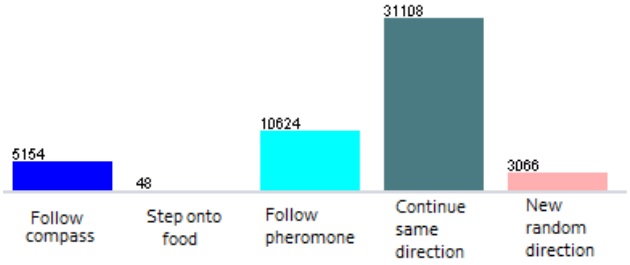

Fig. 5. How many times BM1 triggered each IF statement from the pseudocode.

This was achieved by writing to file all of the inputs and the resulting outputs for 10 ants over a 5000 timestep run. This data set contained 50,000 instances each containing the full set of 13 inputs and 2 outputs.

The training set was then used as the training set for a neural network. The aim was to identify whether an ANN could use backpropagation to learn the relationships between the inputs and the output produced by BM1. A 90% split was used to split into a 45,000 instance training set and an unseen 5,000 instance test data.

In order to use a single ANN to produce two outputs: x and y together, outpux_x and output_y were combined into a single output class with 9 values A-I (Fig 6). In total the ANN had 12 inputs and 9 output nodes, one for each class. With no hidden layers, the network classified 87.5% correctly. With one hidden layer of 10 nodes the MLP correctly classified 90.4% of test data. With two hidden layers of 10 nodes each (Fig. 6), accuracy improved to 92.3%. Training time increased with hidden layers.

It is not known if this trained ANN would in fact perform well as a controller for the ant simulation, or not. The misclassified instances could include important classes. It is hard to identify which situations the network failed in and if those would be critical or not to foriaging behaviour.

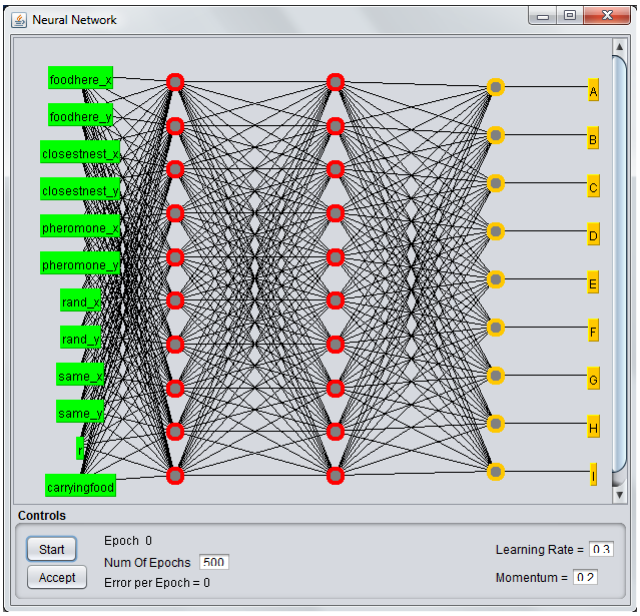

Fig. 6. The fixed topology ANN classified the optimal ant direction 92.3% of the time.

In order to clarify further, a training set was created with all the same inputs, but with 5 possible output class values, representing the 5 IF statements used in the pseudocode (Fig. 1). There are 5 pairs of directional inputs and the purpose of the 5 IF statements is to choose which of those 5 directions to follow (see pseudocode in Fig. 1). This test can clearly identify whether the 5 IF statements were correctly learnt, without having to also learn the correct actions to take within each IF statement. With one hidden layer, a fixed ANN was created and trained (Fig. 7). The ANN reliably identified the correct one of five IF statements 99.98% correctly classified. Only 1 of the 5000 was incorrectly classified.

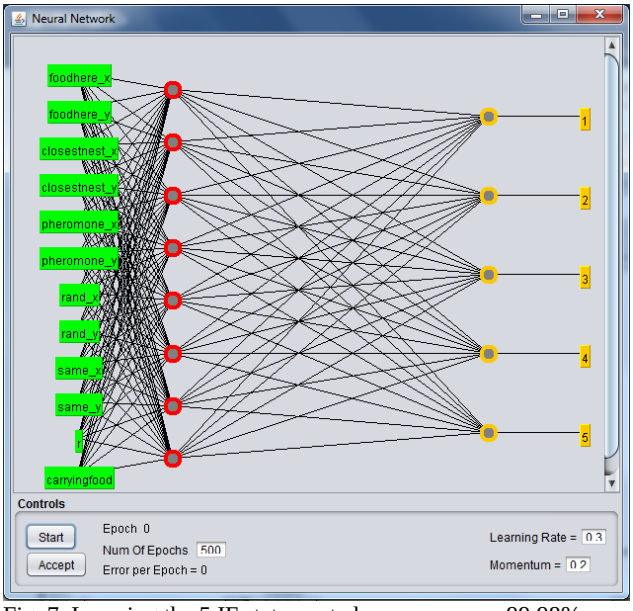

Fig. 7. Learning the 5 IF statement classes accuracy 99.98%.

To summarise this fixed ANN training section, in this approach BM1 output data was used to train an ANN by back propagation. This step was useful to demonstrate that an ANN is capable of performing this task when properly trained, so therefore the task should not be overly complex for an evolving neural network.

This result suggests that a trained fixed topology ANN can perfectly learn to recognise which of the 5 IF statements should be triggered given the inputs. A harder learning task is to also determine which actions to take when each of the 5 IF statements are triggered, the ANN achieved this with 92.3% accuracy.

It is recognised that this approach differs from biological ants with natural selection, which have no pre-existing data to train the ants. In evolution, skills must be evolved without training or guidance, not towards a particular aim or objective. Therefore, the next section focusses on unsupervised evolution.

9 EVOLVING ANN CONTROLLERS

Neuroevolution was applied to evolve neural networks which were then applied as the controller for ants. All ants in a colony had the same controller at each generation. But between generations, the controller was subject to genetic change, by modifying the ANN both in terms of the weights and the topological structure, including the number and location of connections.

Initially, the ANNs were set blank, with no hidden layer nodes. The additional nodes are added by evolution over time.

The fitness function was set to 1 point for each food picked up, 50 points for each food returned to the nest.

A comprehensive set of tests was done with 25 ants per colony, left to run for 900 timesteps. The ANN controllers were subject to neuroevolution in populations of size 15 organisms over 100 generations and this was repeated 10 times. Afterwards a further 5 repetitions of 100 generations was completed, this time with populations of 150 organisms which is a more conventional population size for neuroevolutionary algorithms.

In all runs food collection was learnt almost perfectly. By the 10th generation organisms often had peaked at a fitness of 25, meaning that every single ant successfully collected a food. In most experiments, the ANNs started to learn to return the food to the nest, which begun producing fitness of 62 in generation 13. At generation 42 the fitness was 2049, so the majority of ants were returning multiple foods to the nest. This cannot be explained by random movement alone which does not result in any food being returned. In the third run, the highest fitness reached 5059, meaning that 109 foods were collected, of which 99 foods were returned to the nest in only 900 timesteps, a highly efficient result, that means every ant on average collected food and returned it to the nest 4 times, outperforming BM1.

Visualisation of the evolved ANN structure (Fig. 8) shows that it had an additional 7 nodes had evolved, and 18 new connections, plus all of the weights throughout had evolved to optimal values.

It is hard to visualise why this ANN works so well because ANNs are a black box solution, yet some evolved nodes seem to make sense. The node in the bottom right was added by neuroevolution. It has connections to the Boolean flag pherom_here (input 14) and the pheromone x flag (input 5).

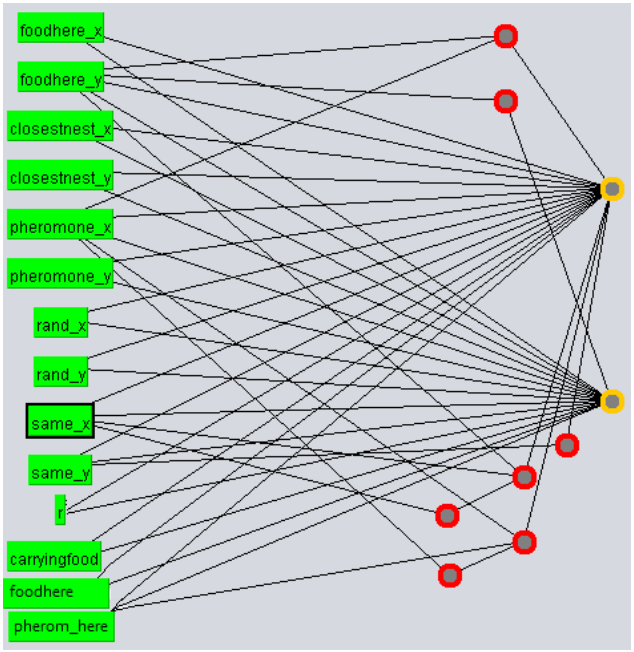

Fig. 8. The evolved ANN which has fitness 5059 and outperformed the benchmark algorithm. Red nodes were added by the neuroevolutionary algorithm, the two yellow nodes are outputs, one for the x movement, one for y movement. The green squares are the 14 ANN inputs.

That could make sense that the Boolean flag pherom_here could trigger the hidden neuron to send pheromone x information to the x output 1 only when present. Another observation is that most evolved nodes that connect to the x output do have connections coming from x inputs and the same is true for evolved nodes connected to the y output.

Subsequently the same experiment was repeated four times with a larger population size of 150. The runs produced fitnesses of 8533, 6178, 6083 and 2152. When controlled by the ANN with highest fitness 8533 the ants had collected 183 foods and returned 168 of those to the nest. Given the size of food clusters on the fixed map are over 1000 each, the score could be achieved by discovering a single food cluster.

For comparison, the BM1 was run 100 times for 900 time steps with 25 ants and the colonies had returned to the nest a range of food from 2 to 107, with an average of 60 foods. A possible reason why neuroevolution outperformed BM1 could be that it was overfitted to the test level layout.

Comparisons using purely random movement with 25 ants over 900 timesteps showed that a maximum of only 1 food was collected by ants, and no food was ever returned to the nest.

10 ALTERNATIVE SCENARIOS

A scenario was tested in which there was no compass and 1 pheromone. The compass input was experimentally removed, to identify whether a benchmark could be programmed without a 'nest compass' a sensor to nest direction, using only one pheromone. The algorithm would not correctly operate, because after finding the food there was no way for the ants to find a way back to the nest, so ants would move randomly, leaving pheromone all over the place attracting other ants in the wrong directions (fig. 9).

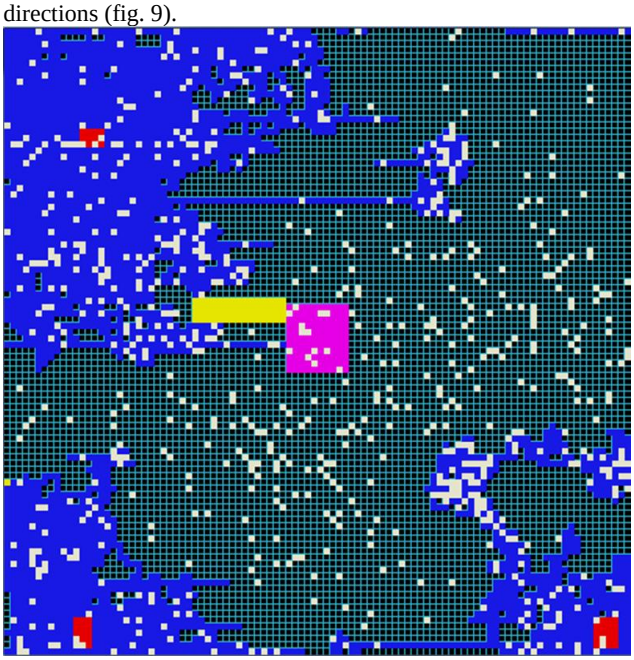

Fig. 9. The BM1 algorithm running without a compass sensor – ants have no way of finding the nest once food is discovered and pheromone is scattered randomly.

Food distance from nest has various effects. With closer food, the pheromone trail will be stronger and it takes less time to get back to the nest. But longer trails have greater chance of other ants walking into them by accident, so further food may attract more ants that way. This is shown in Fig. 10. Two foods were discovered: a small food in the upper right is favoured compared to a larger food in the bottom left, because it is closer, ther pheromone is stronger and all ants abandon the larger food until the pheromone evaporates and knowledge of it's location is lost to the swarm. In Fig. 11b, the same affect is shown. Ants recruited to the bottom two foods only collected food once and when they reached the nest, they chose to follow the top food because that pheromone was stronger and the top food quickly depleted.

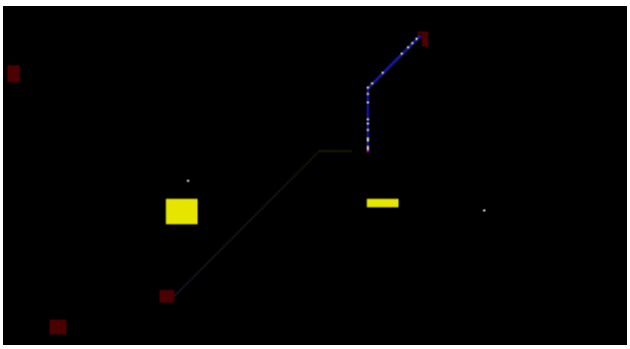

Fig. 10. A large food supply (lower left) is abandoned and forgotten in favour of a small food (upper right), because it is closer to the nest, causing a stronger pheromone trail.

Each decision that an ant makes can be subject to random probability so that ants are always capable of doing something unpredictable at any time. The effect of introducing a probability of random decisions is shown between Figs 11a and 11b.

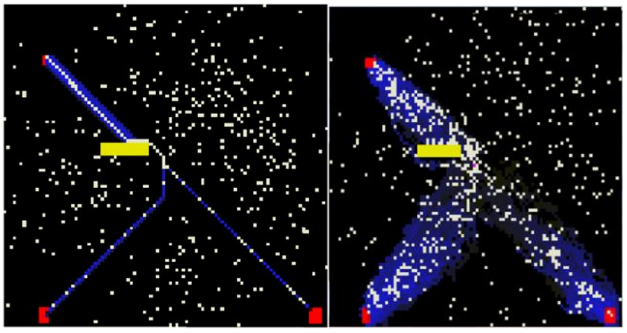

Fig. 11. (a) With 0% chance of random behaviour at each timestep. White: ants. Red: food source. Blue-black: pheromone strength. Yellow: obstacle. Central point: nest. (b) With 70% chance of random behaviour at each timestep.

11 COMMON PROBLEMS ENCOUNTERED

One problem with using a compass to return to the nest is that if obstacles block the route in a 'v' shape, the ant never get past. In Fig. 12 this has occurred and long after the bottom two food sources are completed, the blocked food source hasn't been exploited yet. If two pheromones were used this may be avoided.

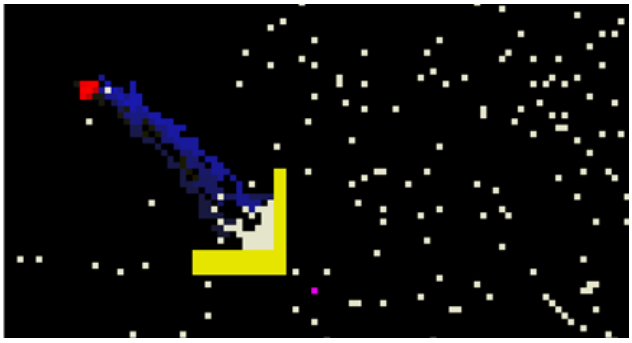

Fig. 12. Using one pheromone with compass, ants often get stuck behind a 'v' shaped yellow obstacle.

12 PHEROMONE STRENGTH DECREMENT

Hill climbing strategy has two main requirements. When an ant discovers a pheromone trail, there are usually two directions it could be followed. The ant should choose which direction using a hill climbing strategy to more easily find the food. That assumes that the strongest end of the trail will lead to the food. However, the whole trail evaporates over time, so the strongest end of each pheromone trail will naturally tend to be the end nearest the nest because the trail near the food has had a longer time to evaporate. Therefore, if hill climbing is to work, (1) the pheromone strength deposited by ants must decrease on consecutive squares as they get further from the food. Also (2) the evaporation rate must be slower than the reduction in strength left by ants on consecutive squares. This can be instructed either automatically, or it could be part of the ant's behaviour controller which is required to evolve. That would increase the search space for the ANN and would require two additional inputs (1) the number of steps taken since food and (2) the strength of pheromone already on this square from other ants and an output for the pheromone strength to deposit.

Problems can occur when ants are laying a pheromone trail and they walk across a separate pre-existing pheromone trail of a different strength (Fig. 13). In this case, ants should not be able to add to existing pheromone, up to a maximum limit on each square. They should also not be able to cause existing pheromone to reduce by overwriting pheromone left by other ants with a lower value. This causes problems when two paths from two food sources combine into one, and the ants from either food source will have taken a different number of steps and laying different strengths of pheromone. (Fig. 13.) The correct behaviour is that they should reset the square to their own calculated limit, unless a higher value is already present in which case they leave it as is.

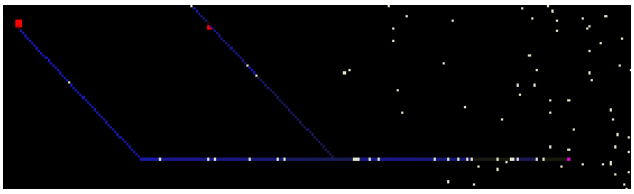

Fig. 13. When two paths combine, pheromone should not be overwritten by ants who have travelled on a longer path.

13 CONCLUSIONS & FUTURE WORK

This paper has investigated the benefits of neuroevolution (NEAT) compared to fixed topology ANN by testing how pheromone behaviour can evolve in both, in relation to a hard coded designed benchmark (BM1).

Future work could investigate simulating other colony or swarm intelligences with communication. Examples include smells in the air or environment such as territory marking, or sounds used for predator detection, warnings or communication.

This paper has demonstrated neuroevolution applied to evolve pheromonome communication in simulated ant colonies. The core intelligence required to perform pheromone communication was summarised in form of the hard coded benchmark BM1, comprised of an IF block with 5 conditions. The 5 conditions were learnt by a fixed ANN and the actions to take within each IF statement were learnt with 92% accuracy.

This paper was organised into several stages: (1) Developed a benchmark algorithm which produces swarm food foriaging behaviour. (2) Used the benchmark to produce a training dataset linking the ant sensor inputs to the desired output movement direction. (3) The training set was used to train a fixed topology neural network which produced the desired output in 92% of cases. (4) Implemented neuroevolution to evolve an ANN with augmented topology and weights to produce foriaging behaviour. This was successful and the evolved ANN controller resulted in high number of foods being collected by the swarm and returned to the nest. The evolved controller outperformed the benchmark algorithm which presumably was due to overfitting to a fixed level layout.

REFERENCES

[1] Gaggioli. Optimal Experience in Ambient Intelligence. In: *Ambient Intelligence*. G. Riva, F. Vatalaro, F. Davide, M. Alcañiz (Eds.). IOS Press (2005).

[2] Jin, A., & Austin, M. (2014). Comparative Evolutions of Swarm Communication.

[3] Marocco D, Nolfi S. "The Emergence of Communication in Evolutionary Robots". In: Philosophical Transactions of the Royal Society London A.361 (2003), pp. 2397–2421.

[4] Baddeley, B., Graham, P., Husbands, P., & Philippides, A. (2012). A model of ant route navigation driven by scene familiarity. PLoS computational biology, 8(1), e1002336.

[5] University Of Sussex, 2006, How ants find their way, Science Daily, https://www.sciencedaily.com/releases/2006/10/061018094651.htm

[6] Collins, R. J., & Jefferson, D. (1990a). Antfarm: Towards simulated evolution. Computer Science Department, University of California.

[7] Marshall, J., Kovacs, T., Dornhaus, A., & Franks, N. (2003). Simulating the evolution of ant behaviour in evaluating nest sites. Advances in Artificial Life, 643-650.

[8] Hamilton, W. D. (1964). The genetical evolution of social behaviour. Ii. Journal of theoretical biology, 7(1), 17-52.

[9] Wolfram S, A new kind of science.

[10] Adler, F. R., & Gordon, d. M. (2003). Optimization, conflict, and nonoverlapping foraging ranges in ants. The American Naturalist, 162(5), 529–543.

[11] Doursat, R., Sayama, H., & Michel, O. (2013). A review of morphogenetic engineering. Natural Computing, 12(4), 517-535.

[12] Wohlgemuth, S., Ronacher, B., & Wehner, R. (2001). Ant odometry in the third dimension. Nature, 411(6839), 795–798.

[13] Hölldobler, B. (1976). Recruitment behavior, home range orientation and territoriality in harvester ants, Pogonomyrmex. Behavioral Ecology and Sociobiology, 1(1), 3–44.

[14] Thiélin-Bescond, M., & Beugnon, G. (2005). Vision-independent odometry in the ant Cataglyphis cursor. Naturwissenschaften, 92(4), 193–197.

[15] Beem FV, 2017, Analysis of NEAT and application in swarm intelligence, Amsterdam University, Undergraduate Dissertation, Supervisor: Rein van den Boomgaard.

[16] Conforth and Meng, Toward Evolving Neural Networks using Bio-Inspired Algorithms

[17] Chang R, Worlanyo S, 2015, Evolving Swarm Communication with NEAT.

[18] Rawal, A., Rajagopalan, P., Miikkulainen, R., & Holekamp, K. (2012). Evolution of a communication code in cooperative tasks. Artificial Life, 13, 243-250.

[19] Collins, R. J., & Jefferson, D. R. (1990b). An artificial neural network representation for artificial organisms. In International Conference on Parallel Problem Solving from Nature (pp. 259-263). Springer, Berlin, Heidelberg.

[20] Panait, L., & Luke, S. (2004). Learning ant foraging behaviors. In proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9) (pp. 575-580).

[21] K. Letendre and M. E. Moses. Synergy in ant foraging strategies: Memory and communication alone and in combination. In Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '13 Companion), pages 41–48, New York, NY, 2013. ACM.

[22] T. Paz Flanagan, K. Letendre, W. Burnside, G. M. Fricke, and M. Moses. How ants turn information into food. In 2011 IEEE Symposium on Artificial Life (ALIFE 2011), pages 178–185, Piscataway, NJ, 2011. IEEE Press.

[23] Hecker, J. P. (2015). Evolving Efficient Foraging Behavior in Biologically-Inspired Robot Swarms (Doctoral dissertation, The University of New Mexico).

[24] Stanley, K. O. (2004). Efficient evolution of neural networks through complexification (Doctoral dissertation).

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

[26] Floreano, D., Mitri, S., Magnenat, S., & Keller, L. (2007). Evolutionary conditions for the emergence of communication in robots. Current biology, 17(6), 514-519.

[27] Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. IEEE computational intelligence magazine, 1(4), 28-39.

[28] Johnson, L. K., Hubbell, S. P., & Feener Jr, D. H. (1987). Defense of food supply by eusocial colonies. American Zoologist, 27(2), 347-358.