Many Hands Make Light Work: Group Evolution and the Emergent Division of Labour

Nicholas Tomko[1], Inman Harvey[1], Andrew Philippides[1] and Nathaniel Virgo[1]

[1]CCNR, Evolutionary and Adaptive Systems Group, University of Sussex, Brighton UK
nt79@sussex.ac.uk, inmanh@gmail.com, andrewop@sussex.ac.uk, nathanielvirgo@gmail.com

Abstract

Most standard genetic and evolutionary algorithms (GAs) are unable to evolve cooperative solutions to problems where there is a division of labour among genetically different component parts. This is because standard GAs evaluate and select all individuals on the same task which leads to genetic convergence within the population. The goal of evolutionary niching methods is to enforce diversity in the population so that this genetic convergence is avoided. One drawback with some of these niching methods is that they require *a priori* knowledge or assumptions about the specific fitness landscape in order to work. Another issue is that many of these niching methods are not set-up to work on cooperative tasks where fitness is only relevant at the group level. In this paper we present the Group GA which is a group based evolutionary algorithm that can evolve cooperative solutions to problems using emergent niching with minimal a priori assumptions. We demonstrate this novel GA on an immune system matching task and explain why we think this type of GA has the potential to effectively solve a wide range of problems that would benefit from being solved cooperatively.

Introduction

In biology, speciation and niching can be broadly described as the evolutionary process by which a single type of biological organism differentiates into multiple "specialised" organisms, that for instance, take advantage of different resources available in a given environment. In some cases niching produces competing species, but niching can also occur within a single species to produce different specialists that work together to solve a given task. An example of this are bacterial colonies, where within any single colony there are groups of different bacteria doing different jobs, all of which are contributing to the collective well being of the colony. In this case the fitness of the colony depends on the collective symbiotic functionality rather than the fitness of any individual bacteria.

Most standard artificial evolutionary and genetic algorithms (GAs) tend to take a very individual centric view of evolution, where the fittest individuals are selected to produce the next generation of individuals. These types of GAs work well on problems with a single global fitness peak, where each individual can solve the task on its own; but they are unable to find multiple solutions to multi-peaked problems or solve problems cooperatively, where there is a division of labour between population members which requires different genotypes. For a GA to be able to find cooperative solutions to problems, it must have the following characteristics: (1) It must be able to maintain diversity within the population so that niches can form and (2) it must allow for fitness to be evaluated at the group level.

Evolutionary niching methods solve problem (1) by enforcing diversity in standard GAs so that a single population can be split up into n different niches. One of the issues with some of the more common niching methods is that they require *a priori* knowledge about the specific fitness landscape to work; in particular whether n is 2 or 5 or some different number. Most of these evolutionary niching methods use either direct or indirect methods to determine the appropriate number of niches. Direct methods include cooperative coevolution where the number of species is set before evolution begins. Indirect methods include fitness sharing and crowding which rely on a pre-set niching (similarity) radius or some sort of similarity calculation in order to get the population to niche. The other problem with these niching methods is that they are tailored for tasks where each individual in the population can solve the task on its own, not for tasks that are best solved symbiotically where fitness can only be calculated at a group level.

In this paper we present a novel genetic algorithm, the Group GA, which niches based on the evaluation of groups of individuals and therefore can be used to solve tasks that require individuals working together doing different jobs. The Group GA has the added benefit of accomplishing this niching with minimal *a priori* knowledge of the fitness landscape and is able to niche without knowing the optimal number of niches or how the different jobs should be shared out. So unlike the more common niching methods it does not require the number of niches to be set ahead of time nor does it require setting any indirect niching parameter such as a similarity or niching radius.

We demonstrate the emergent niching ability of the Group

GA on an artificial immune system matching task. The goal of this task is to evolve a population of antibodies (protecting agents) to match a set of antigens (harmful invaders). Therefore to solve this task the population of antibodies needs to niche so that it contains different individuals that match different antigens. One reason this task was chosen is because the number of peaks in the fitness landscape can be changed by changing the number of antigens that the population of antibodies needs to match. The other reason for choosing this task is that it makes it very easy to determine when niching has occurred.

In the next section we will briefly review some of the common niching methods as well as a few related evolutionary algorithms that can solve problems symbiotically, where there is a division of labour required. Following our literature review we describe the artificial immune system task and the Group GA in detail. We will then show how the Group GA can be used to evolve a population of antibodies to match a set of four antigens, as well as how it can be used to evolve a population of antibodies that adapts to the addition and removal of antigens during evolution. Finally, we compare the Group GA to other evolutionary methods and discuss the types of tasks we feel the Group GA is best suited to solve.

Literature Review

We start by reviewing the most common niching methods in artificial evolution. The purpose of these niching methods is to stop the population from genetically converging during evolution as happens when using a conventional GA. All of these niching methods below can be classified as explicit niching methods because they either require the number of niches to be set *a priori* or require an indirect method of enforcing diversity in the population.

We will also briefly discuss SANE and the Binomics GA which are two GAs that are set-up to allow implicit niching to evolve symbiotic solutions to problems. Unlike the explicit niching methods, these algorithms attempt to evolve a diverse, niched population emergently using group evaluation. They also differ from the genetically based niching methods in that these GAs do not require that each individual in the population can solve the task on its own.

Genetically Based Niching Methods

In this section we briefly describe the common genetically based niching methods. These niching methods function based on the assumption that each individual in the population has its own fitness. For a more in depth summary see Dick (2005) and Mahfoud (1995).

Fitness Sharing and Clearing Fitness sharing (Goldberg and Richardson, 1987) is a niching method that relies on some distance metric or similarity measure (either genotypic or phenotypic) between individuals. By using suitable methods to adjust the fitness of any individual according to how many other similar individuals are within some predetermined niche (similarity) radius, there is a tendency for the population to spread out over multiple peaks or niches in the fitness landscape; thus diversity is maintained. Clearing (Petrowski, 1996) is very similar to fitness sharing but, instead of degrading the fitness of individuals within the same similarity radius or subpopulation, it removes the least-fit individuals within the similarity radius from the population. Horn et al. (1994) show that in Learning Classifier System models where fitness is shared amongst cooperating individuals implicit niching can occur.

Crowding Crowding was first introduced by De Jong (1975) as a method of removing similar individuals from a population, with the goal of trying to maintain diversity during evolution. Deterministic Crowding (Mahfoud, 1995) is a specific type of crowding that mates two in the population and then replaces the parent that is most similar to the offspring if the offspring is fitter. It is similar to fitness sharing because there needs to be some similarity calculation done between individual, but unlike fitness sharing there is no requirement to pre-specify a similarity radius.

Demes and Spatially Structured GAs

An alternative to genetically based niching methods are spatially structured GAs; for a good review see Dick (2005). In these types of GAs, the population is structured within some local geographical distribution (demes) that constrains which members of the population are allowed to be selected or be recombined with one another. By structuring the population into demes more genetic diversity can be maintained across sub-populations.

Cooperative Coevolution Cooperative coevolution was first introduced by Potter and De Jong (1994) as a method for function optimisation. In cooperative coevolution the population is pre-divided into different subpopulations, so it can be thought of as a type of spatially structured GA. Each subpopulation represents a subcomponent required to solve the overall task, which means that there needs to be some *a priori* knowledge of the problem so that the appropriate number of subpopulations is chosen. Each subpopulation is evolved separately using a standard GA, but the fitness of the individual members of each subpopulation is based on the performance of the cooperative solutions. In cooperative coevolution speciation is not emergent because the number of subpopulations needs to be determined before evolution begins. For this reason, this class of algorithms has been shown to work well on problems where there is an obvious way of dividing up the population, such as job shop planning and scheduling tasks (Husbands and Mill, 1991; Husbands, 1993; McIlhagga et al., 1996).

Symbiotic GAs

SANE (Moriarty and Miikkulainen, 1995, 1996), the Binomics GA (Harvey and Tomko, 2010) and simulated ecosystem evolution (Williams and Lenton, 2007) are three examples of GAs that cause implicit niching in the population and attempt to evolve symbiotic solutions to problems. In SANE and the Binomics GA, groups of individuals are evaluated together and then the individuals that are part of the fittest groups are selected to pass on their genes to the next generation. This differs from most standard GAs where individuals are evaluated and then the fittest individuals are selected. These algorithms are relevant to our discussion of speciation/niching because any time a problem is solved symbiotically then implicit niching must be occurring during evolution.

SANE and the Binomics GA have been successfully applied to the evolution of artificial neural networks (ANNs). In both these algorithms the individuals in the population are partial networks that are combined to form fully specified ANNs which are then evaluated. The fitness score of each individual partial network is based on the fitness of the full ANNs that each individual partial network participated in. This means that over time, the individual partial networks that were part of the fittest ANNs will be selected for, while the partial networks that were part of the least fit ANNs will be modified using mutation and recombined with other partial networks. The goal of this method of evolution is to evolve a population of partial networks that symbiotically work together to form high fitness fully specified ANNs.

The Artificial Immune System Task

We have chosen an artificial immune system matching task to demonstrate the emergent niching abilities of the Group GA. In this section we will describe the details of this task and then in the next section we will describe the Group GA. This task which has previously been used by Forrest et al. (1993) and Potter and De Jong (2000) was chosen because it can be solved cooperatively and clearly illustrates how the Group GA can lead to emergent niching and how it can adapt to a changing fitness landscape, neither of which is possible with a conventional GA. Forrest et al. (1993) used the task to study adaptation in the immune system and Potter and De Jong (2000) solved different variations of this task using cooperative coevolution. We will compare the results of these two papers to the Group GA results later in the paper.

The goal of this task is to evolve a population of antibodies to protect the body from a set of antigens. Very simply speaking, antigens can be thought of as bacteria, viruses or other pathogens and the antibodies can be thought of as the body-guards who mark these antigens for removal. Antibodies in natural immune systems need to be adaptive so that they can combat new and different antigens that enter the body. Therefore this task tries to mimic this challenge of natural immune systems on a very basic level by attempting to evolve a population of artificial antibodies to match a variable set of antigens.

In this task both the antibodies and antigens are modeled as bit strings. How well an antibody combats a specific antigen is calculated as the number of bit matches between antibody and antigen. For example a [1 0 1 1] antibody matches a [0 0 1 0] antigen at location two and three and therefore the antibody's fitness is equal to two when matched to this antigen. For our purposes the higher the match (fitness) score the better.

Assuming that the length of the antibodies and antigens is the same, when there is more than one antigen in the antigen set the task can be thought of as symbiotic, because it is impossible for a single antibody to match an entire set of antigens on its own. In this case, the population of antibodies needs to evolve in such a way so that it contains specialists to combat each different antigen. Obviously the more antigens there are, the more difficult the task becomes, because the antibody population needs to evolve and maintain a larger number of specialists.

The Group GA

The Group GA is a novel evolutionary algorithm presented in this paper for the first time. It is based on the Microbial GA (Harvey, 2011) which is a steady-state GA that uses tournament based selection. The Microbial GA is similar to the more familiar GAs, but is minimalist in the sense that it strips away as much as possible, whilst still maintaining the essential components of natural selection which are heredity, variation and selection.

We will first describe the Group GA in general terms and then describe it in terms of the artificial immune system task we present in this paper. What differentiates the Group GA from more conventional GAs is that groups of population members, of some fixed size that is a parameter of the GA (rather than individual population members as in conventional GAs) are evaluated and then selected based on the overall fitness of the group. In other words, the driver of fitness based selection is the relative fitness of an entire group of population members that work together as a unit to solve some task. A single cycle (tournament) of the Group GA can be broken-up into the five following steps:

1. Randomly choose two possibly intersecting groups of population members from the population without regards to fitness.

2. Calculate and assign a fitness score to each group of population members based on the groups' performance on a given task. Fitness is assigned on the group level only; there need not be any way to define or calculate an individual's contribution to the group's fitness score.

3. All members of the group with the lower fitness score are removed from the population and replaced with mutated

copies of the members of the fitter group.

4. The members of the fitter group are put back in the population unchanged.

5. This process is repeated until some pre-defined stopping condition is met.

When we apply the Group GA to the immune system task, the fitness of a group of antibodies is calculated as the average of the best match scores achieved against all the antigens in the set. In other words, to evaluate a group of antibodies, all the antibodies in the group are matched against every antigen in the set and the average of the highest match scores against each antigen is the group fitness. This means that to get a perfect fitness score there has to be at least one antibody that matches each antigen perfectly in the group.

A single cycle (tournament) of the Group GA can be broken-up into the five following steps when applied to the immune system task described in the previous section (see figure 1).

1. Randomly choose two groups of antibodies from the population without regard for fitness

2. Calculate the match scores between all the antigens in the set and each of the antibodies in each group

3. Each group as a whole is assigned a fitness score which is calculated as described above.

4. The group with the lower fitness score is replaced with mutated copies of the antibodies of the more fit group

5. Both groups of antibodies are put back into the population and this process is repeated

We have set up this simulation in such a way that groups of antibodies are randomly chosen from the population and then assigned a fitness based on the ability of this group to match the different antigens in the antigen set. We understand that because an individual antibody can always be assigned its own fitness, some of the genetically based niching methods we reviewed earlier would be able to solve this task without any type of group evaluation. The reason we have used this task to demonstrate the Group GA is because as we will see in the next section it clearly shows how the Group GA causes emergent niching using group evaluation.

The Group GA can be applied more generally to tasks where individual fitness is meaningless because the Group GA randomly selects two groups of population members and uses them to construct two higher level entities that are evaluated and assigned a fitness score. The less fit group of population members is killed off and replaced with a mutated copy of the fitter group. These two groups are then put back into the population and this cycle is repeated. It is important to reiterate that in the Group GA it is the fitness of the group

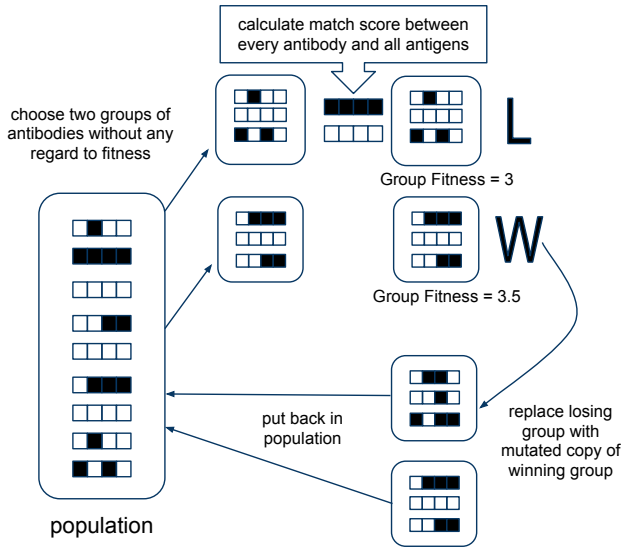

Figure 1: The Group GA as applied to a immune system task with 2, 4-bit antigens

of population members that drives evolution, which is different from most conventional GAs where it is the fitnesses of the individual population members that matters. How fitness is calculated depends on what type of problem is being solved, but regardless it is only the group fitness that matters when determining the tournament winner and loser.

Evolving Antibodies using the Group GA

In this section we will show how, using the Group GA, a randomly initialised population of antibodies can be evolved to match a set of antigens. In the first experiment we will evolve a population of antibodies to match a fixed set of four different antigens. This is equivalent to the Group GA solving a four-peaked fitness landscape. Then in the second experiment we will evolve a population of antibodies to match a variable set of antigens, where antigens are added and removed during evolution. This second experiment simulates a task where the number of fitness peaks changes during evolution.

In these experiments the antigen and antibodies were 64-bit binary strings. The antibody population size was 100 and the number of antibodies per group was 10. The mutation rate was set to 0.1/64, meaning that at each allele there was a probability of 1/640 of flipping that bit.

Figure 2 shows the antibody population after being evolved for 20,000 tournaments on a four antigen task. The four antigens used in this experiment were: [...0 0 0 0...], [...1 1 1 1...], [1 0 0 0...], and [...1 0 1 0 ...], where these 4-bit patterns are repeated 16 times to make the four full 64-bit antigens. These specific antigens were chosen to try to make the task as difficult as possible. The lower part of figure 2

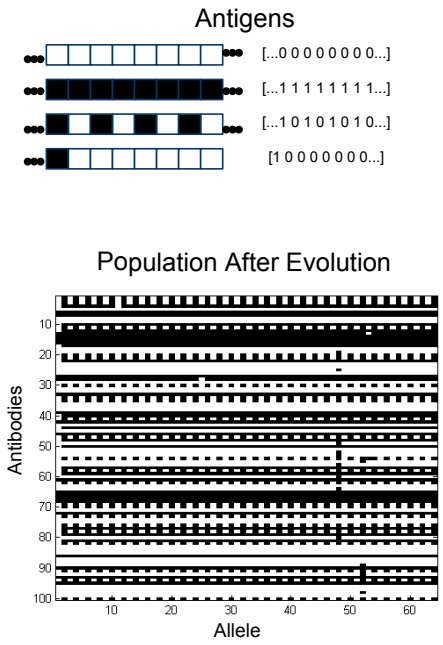

Figure 2: The antibody population after evolution on a 4 antigen task.

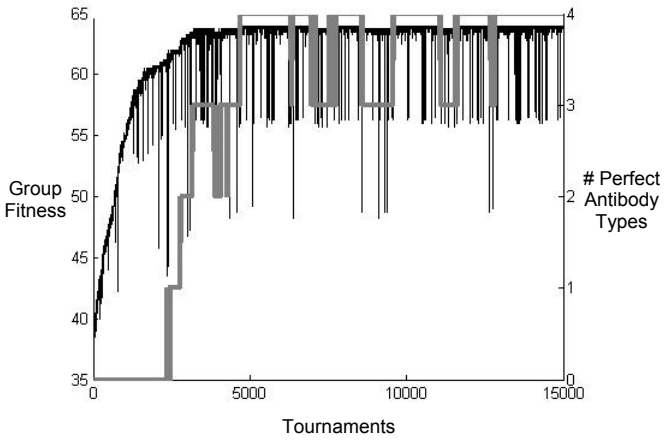

Figure 3: A plot of group fitness (black line) and number of antigens covered perfectly by at least one antibody (gray line) in the population over time for a single typical run of the 4 antigen task.

(as with the similar plots in later figures) displays each binary genotype in the population horizontally above the next genotype, with white and black representing 0 and 1 alleles respectively. Figure 2 clearly shows how the antibody population has niched during evolution to contains antibodies that perfectly match all four antigens in the set.

Figure 3 is a fitness versus time plot for a single typical run of the four antigen task. The black line shows the group fitness of the tournament winning group of antibodies at each tournament, calculated as described above and the gray line shows the number of antigens covered perfectly by at least one antibody at each tournament. The number of perfect antigens matched perfectly by at least one antibody can range from zero to the total number of antigens in the set. We believe that this is an important measure of performance for this task because if you think of the goal of the antibodies in terms of protecting a body from invasion, then it is important that the population contains at least one antibody to match each antigen. In this figure you can see that throughout evolution the group fitness drops significantly for a tournament or two without decreasing the fitness of the population (number of perfect antibody types). This is because antibody groups are randomly chosen from the population so there is always a chance that a very unfit group is chosen.

Figure 4 shows how the antibody population adapts when antigens are added and removed during evolution. In this experiment, the antigen set initially contained only two antigens [...0 0 0 0...] and [...1 1 1 1...]. At tournament 20K a third antigen [...1 0 1 0...] was added and evolution was resumed. At tournament 40K evolution was paused again and the [...1 1 1 1...] antigen was removed from the set before evolution was restarted. This figure clearly shows that when the antibody population is evolved using the Group GA the population can adapt to changes in the antigen set, adding and removing different types of antibodies as appropriate. Figure 5 shows the fitness versus time plot for this a single typical run of this task, where antigens are added and removed during evolution. As this figure shows, when an antigen is added, the fitness of the population drops before quickly recovering as the population adapts to match this new invader [1].

Comparison to Other Methods

To get a feel for how well the Group GA is able to solve on this task we compared it to both the Microbial GA (Harvey, 2011) and the Binomics GA (Harvey and Tomko, 2010) on the 4 antigen task described above. Using the Microbial GA to solve this task is equivalent to solving it using any standard GA where the fittest individual antibodies are selected. As expected, when we ran the Microbial GA for 100 runs, each run the antibody population converged to match a single antigen in the antigen set, failing to match the other three.

A more interesting comparison is between the Group GA and the Binomics GA. We chose to compare the Binomics GA as opposed to a genetic based niching method such as fitness sharing or crowding because like the Group GA, the

[1]There are potential similarities between the adaptive mechanism of the Group GA and clonal selection that need to be investigated further.

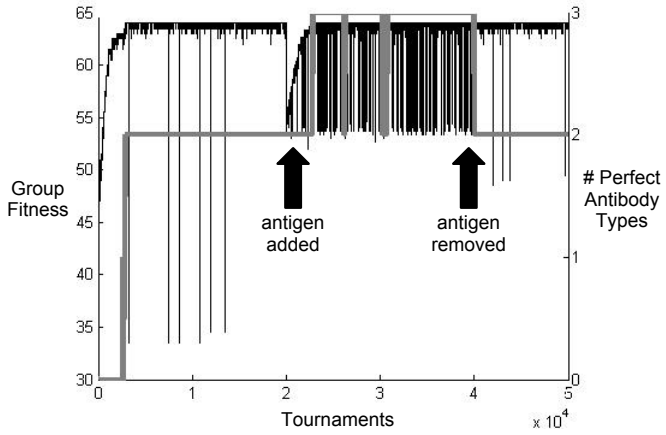

Figure 5: A plot of group fitness (black line) and number of antigens covered perfectly by at least one antibody (gray line) in the population over time for a single typical run of the task where antigens are added and removed during evolution.

Binomics GA was developed to solve cooperative tasks using emergent niching where group fitness is the driver for selection. As applied to this immune system task, the Binomics GA works as follows:

1. Randomly choose two antibodies from the population and compare their stored fitnesses.
2. The antibody with the lower fitness is genetically changed using mutation and recombination.
3. This modified antibody is combined with a group of randomly chosen antibodies from the population.
4. All the antigens are matched against all the antibodies in the group.
5. The fitness of this group of antibodies is equal to the mean maximum match score in the group.
6. All antibodies in the group have their current fitness updated using some sort of time smoothing that takes into account both their historical and newly calculated fitness.
7. All individuals are put back in the population and this cycle is repeated.

The difference between the Group GA and the Binomics GA is that in the Group GA, groups of antibodies are being both evaluated and selected, while in the Binomics GA groups of antibodies are being evaluated, but it is individual antibodies that are being selected based on this group fitness.

Using the same parameters as in the previous experiments, we compared the performance of the Group GA and the Binomics GA on the 4 antigen task over 10 runs. We decided to compare the performance of these two algorithms based on the number of evaluations it took to evolve a population that contained antibodies that perfectly matched all antigens in the set. Evolution was stopped at 1600 K evaluations if by that point the population did not contain 4 perfect antibodies. Over 10 runs the Group GA took a median number of 278 K evaluations, while the Binomics GA was unable to solve the task within the maximum number of evaluations allowed in any of the 10 runs. It should be mentioned that if the Binomics GA was allowed to run for more evaluations, it was able to niche to match the four different antigens, but nowhere near as efficiently as the Group GA. In the next section we will discuss why we think the Group GA outperforms the Binomics GA to this extent.

Discussion

In this paper we have presented a novel evolutionary algorithm that can cooperatively solve problems using emergent niching, where fitness is evaluated at the group level. We demonstrated this by using the Group GA to solve a multi-peaked artificial immune system matching task. Our results show that by evolving a population of antibodies using the Group GA, the population niches to match multiple antigens. We have also shown that when antigens are added and removed during evolution, the Group GA allows the antibody population to adapt to this change matching new antigens that are presented.

In the previous section we compared the performance of the Group GA to the Microbial GA and the Binomics GA. Unsurprisingly, the Microbial GA, where individual antibodies are evaluated and selected was unable to solve the multi-antigen task and ended up converging to match a single antigen every run. The Binomics GA, where groups of antibodies are evaluated and individual antibodies are selected, fared much better and was able to niche to match the different antigens, but took a lot longer as compared to the Group GA. We believe that the reason why the Group GA outperforms the Binomics GA methods on this task is related to the difference between what is being evaluated and what is being selected. Studying the subtle differences between evaluation and selection and how varying what is evaluated and selected affects artificial evolution is not part of the scope of this paper, but will be one of the focuses of our future research.

The two key characteristics of the Group GA that differentiate it from the niching methods described in the literature review are: (1) Niching is accomplished emergently without having to know the appropriate number of niches ahead of time or pre-setting any parameter such as a niche radius and (2) fitness is evaluated at a group level which means that the Group GA can be used to solve symbiotic task where fitness is meaningless at the individual level.

For example, this same immune system task was solved

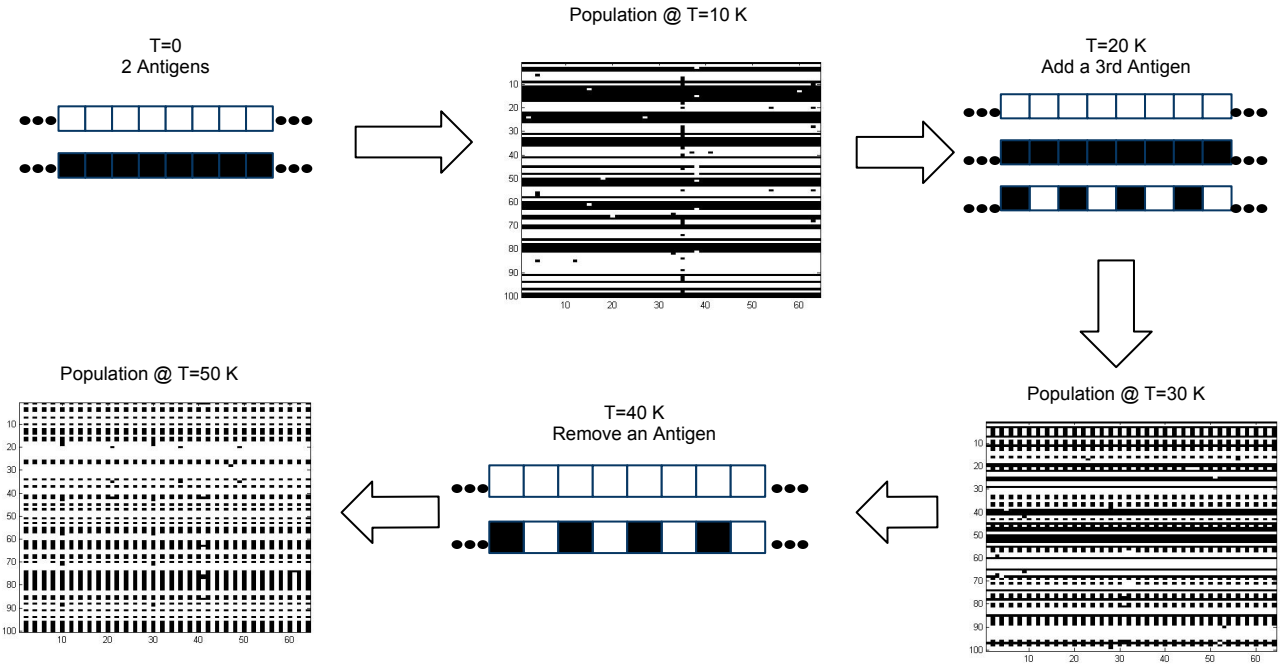

Figure 4: This figure shows how the antibody population adapts during evolution when 64 bit antigens are added and removed (T=10 K corresponds to tournament 10,000).

by Potter and De Jong (2000) using cooperative coevolution where the population was subdivided into n different species before evolution wass started. This method was successful at evolving a population of antibodies to match different antigens as long as the number of different antigens was known *a priori* and the number of antigens remained constant throughout evolution. To overcome these limitations of cooperative coevolution, Potter and De Jong (2000) applied an evolutionary stagnation measure to determine when a new sub-population should be added. This allows antibody species to be added and removed during evolution in response to new antigens, but as Potter and De Jong (2000) state, the level of stagnation at which species should be added or destroyed is task dependent.

This task was also solved by Forrest et al. (1993) using a GA with a best-match fitness scoring scheme. In their algorithm, an antigen is chosen at random and matched against a group of antibodies from the population. Only the antibody in the group with the highest match score gets its fitness increased by its match score, the fitness of all other antibodies remains unchanged. This fitness evaluation step is repeated many times and then the population is evolved using a standard GA. Like the Group GA, this method allows the antibody population to niche to match a set of antigens without needing to know *a priori* how many antigens are present. The major difference between this method and the Group GA is that this best match method requires that the fitness of individual population members can be evaluated on their own. This is possible for this task because each individual antigen can be evaluated on its own by matching it against a single antigen, but tasks where fitness can only be evaluated at the collective, group level will not be able to be solved using this best-match method. In general, the genetically based niching methods described earlier will struggle with this type of symbiotic task where individual fitness is meaningless. An example of this type of task is the evolution of artificial neural networks (ANN) task where the population is made up of partial sub-networks which have no fitness except when they are combined with other sub-networks to form a fully specified networks. Both SANE and the Binomics GA discussed earlier have been used to solve ANN tasks in this way.

For the reasons given above we believe that the Group GA has the potential to be a useful algorithm that can use emergent niching to solve problems where the optimial division of labour is unknown. Going forward, we plan on testing the Group GA on a wide variety of tasks which may benefit from being solved cooperatively in order to find out when it performs well and under what circumstances it performs poorly. We also plan on studying the effect of varying the group size parameter on this immune system task as well as other tasks. Testing the Group GA on an ANN task may be a logical next step, as neural networks can be viewed as a group of neurons symbiotically working together to solve a

problem. We think that the Group GA could be the catalyst for the development of a new class of GAs that specialise in solving tasks cooperatively where there is limited *a priori* knowledge of the fitness landscape.

References

De Jong, K. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. Ph.d. thesis, University of Michigan, Ann Arbor.

Dick, G. (2005). A comparison of localised and global niching methods. In *17th Annual Colloquium of the Spatial Information Research Centre (SIRC 2005: A Spatio-temporal Workshop)*, pages 91–101, Dunedin, New Zealand. Citeseer.

Forrest, S., Javornik, B., Smith, R. E., and Perelson, A. S. (1993). Using Genetic Algorithms to Explore Pattern Recognition in the Immune System. *Evolutionary Computation*, 1(3):191–211.

Goldberg, D. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimisation. In Grefenestette, J., editor, *Proc. of the Second International Conference on Genetic Algorithms*, pages 41–49, Hillsdale, NJ. Lawrence Erlbaum Associates.

Harvey, I. (2011). The Microbial Genetic Algorithm. In Kampis, G., Karsai, E., and Szathmary, E., editors, *ECAL 2009, Part II. LNCS 5778*, pages 126–133, Heidelberg. Springer.

Harvey, I. and Tomko, N. (2010). Binomics : Where Metagenomics meets the Binary World. In *Proceedings of Artificial Life XII, 12th Intl. Conf. on the Synthesis and Simulation of Living Systems*, Odense, Denmark.

Horn, J., Goldberg, D., and Deb, K. (1994). Implicit niching in a learning classifier system: Nature's way. *Evolutionary Computation*, 2(1):37–66.

Husbands, P. (1993). An ecosystems model for integrated production planning. *International Journal of Computer Integrated Manufacturing*, 6(1):74–86.

Husbands, P. and Mill, F. (1991). Simulated co-evolution as the mechanism for emergent planning and scheduling. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 264–270. Morgan Kaufmann Publishers.

Mahfoud, S. W. (1995). *Niching Methods for Genetic Algorithms*. PhD thesis, University of Illinois at Urbana-Champaign.

McIlhagga, M., Husbands, P., and Ives, R. (1996). A comparison of optimization techniques for integrated manufacturing planning and scheduling. *Parallel Problem Solving from Nature IV*, pages 604–613.

Moriarty, D. and Miikkulainen, R. (1995). Learning Sequential Decision Tasks. In Honavar, V., Patel, M., and Balakrishnan, K., editors, *Advances in the Evolutionary Synthesis of Neural Systems*, Cambridgel, MA. MIT Press.

Moriarty, D. E. and Miikkulainen, R. (1996). Efficient reinforcement learning through symbiotic evolution. *Machine Learning*, 22:11–32.

Petrowski, A. (1996). A clearing procedure as a niching method for genetic algorithms. *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 798–803.

Potter, M. and De Jong, K. (1994). A cooperative coevolutionary approach to function optimization. In Davidor, Y. and Schwefel, H., editors, *Parallel Problem Solving from Nature (PPSN III)*, pages 249–257. Springer Verlag.

Potter, M. and De Jong, K. (2000). Cooperative coevolution: an architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29.

Williams, H. and Lenton, T. (2007). Artificial ecosystem selection for evolutionary optimisation. In Almeida E Costa, F. E. A., editor, *Advances in Artificial Life: Proceedings of the 9th European Conference on Artificial Life*, pages 93–102, Berlin. Springer Verlag.