A Simple Environment for Research in Speciation

Gary B. Parker and Thomas B. Edwards

Department of Computer Science
Connecticut College
New London, Ct 06320 USA
parker@conncoll.edu and tedwards@conncoll.edu

Abstract

In this paper we discuss an environment that we've developed in order to investigate the replication of speciation in evolutionary computation (EC). There are many versions of EC that have some characteristics of speciation, but none that match natural processes. In an effort to develop such a form of EC, we've created a simple model that can be used to experiment with EC and environments that may result in natural speciation. Our goal for speciation is to have a single population eventually become two populations that are reproductively isolated even though they reside in the same environment. At this point we've developed the model and have completed experiments that show it is a viable environment for the exploration of speciation by replicating adaptation, survival of the fittest, and migration of a population.

Introduction

The goal for this research is to develop a form of evolutionary computation (EC) [De Jong 2006] where the system starting with a single population is capable of evolving into distinct populations. This would be a good step in the development of a general purpose EC and would help in understanding the principles of evolution. The form of EC that we use is based on the genetic algorithm (GA), which uses a population of potential solutions and sexual reproduction to produce better solutions [De Jong 2006]. In regards to this research, we consider a population to be distinct (and a separate species) if it is made up of individuals that are unable to produce viable offspring with individuals from the other population or if offspring are produced, they are sterile. The short term goal is to have individuals of differing species choose not to mate and if they do produce offspring, the offspring do not continue to reproduce. In this way, the gene pools for each of the species will be isolated.

Toward this goal, we've developed an environment that we plan to use for experimentation leading to an EC that has the ability for speciation. This environment needs to be simple so that many generations can be run in a reasonable amount of time and the results of experiments can be easily interpreted. With this in mind, we've completed a model and performed preliminary tests to determine if it's viable for our purposes. In these tests, we've shown that the building block components needed for evolution are present in the model.

In our simulated environment, we use a population of chromosomes that represent individuals. These individuals move around in the environment, eat food, gain fitness, reproduce to create new individuals, and die through aging and/or lack of life (fitness). The environment can replicate survival of the fittest where the chromosomes of the fittest individuals survive from generation to generation. In addition, individuals in the environment will adapt to match the available food sources and the populations will migrate across the environment as the availability of food sources change. We believe this is a good first step toward developing an environment to test speciation.

There has been research in the past that tends toward speciation with most of it focused on the use of niching [Goldberg and Richardson 1987] to improve the use of GAs for multimodal function optimization. These works use a similarity measure between chromosomes to form niches with subpopulations in the solution landscape. This has been shown to be an effective way to ensure distribution in solutions (represented as chromosomes) and further research has been completed to expand the idea to create new ways to maintain niches [Glibovets and Gulayeva 2013]. Although a good solution for multimodal function optimization, niching with its distinct subpopulations is not what we consider speciation. The individuals within the niches are determined by computing a similarity factor, which is used to create subpopulations that are reproductively isolated, as opposed to mating preferences as in natural systems.

One of the more popular systems that use niching for the evolution of agents is NeuroEvolution of Augmenting Topologies or NEAT [Stanley and Miikkulainen 2002]. NEAT uses a compatibility function to determine if two agents are part of the same species or not. It accesses the historical data for agents and uses this information to tell

which agents belong to which species (niche). In addition, NEAT uses explicit fitness sharing that makes similar agents share the pay-off of their fitness, which means that there is competition within a species, but not as much between two different species. This encourages new species by giving them an advantage. NEAT is similar to our research because it uses evolutionary processes to learn agent controls, the learning is in real time, and there is a concept of speciation, but in our system we want species to develop through natural reproductive isolation by choosing not to mate as opposed to being restricted from mating. Our system is intended to be more in line with the natural processes needed for biological speciation. We do not have a concept of explicit fitness sharing where agents within a species are required to share their fitness payoff since in biological systems there are also several cases where species compete against other species for resources.

Other research with similarities to our research is the Speciation Island Model (SIM) since it tries to evolve new subpopulations as well. It uses the biological concept of species as a model for parallel evolutionary algorithms [Gustafson and Burke 2006]. The species represent possible problem solutions and islands are processors that are running different solutions -- performing fitness calculations in their assigned sub-population. Outliers of an island's population are detected, removed from the population, and placed on another island. In this way, the populations can then evolve independently of each other. Some form of speciation is occurring due to the physical separation of the individuals, however the outliers are detected and actively removed from the population and added to new ones without natural migration. There is no notion of true speciation since the populations are never recombined to test for reproductive isolation.

Potter and De Jong [1995] used something similar to islands and speciation in their Evolving Neural Networks with Collaborative Species research where cooperative speciation was applied. The research uses something similar to islands (processors) that each independently evolve their own populations. However, the islands communicate with each other because the islands select representatives to be combined with the other representatives into a single structure, which is then evaluated to see how well it solves the problem. The fitness of the solution returns to the islands via the representatives so island sub-populations can continue to evolve. This research used Cooperative Coevolutionary Algorithms [Potter and De Jong 1994], which are intended to solve large problems as opposed to replicate speciation. The "species" start out separate and remain that way, whereas in our research, we hope to have our agents start as one species and split into two. In addition, their species are intended to work together whereas our agents are competing for limited resources.

Although not the main point of our research, our EC uses variable population sizes. Other research in this area includes the Genetic Algorithm with Varying Population Size (GAVaPS), which is a method for varying the population by the use of agent aging where chromosomes die when they reach a certain age [Arabas, Michalewicz, and Mulawka 1994]. There are different processes that can be used to determine this age, which is specified for each chromosome. Our research also uses aging, however we do not pre-determine an age for death, use age to impact how much energy an agent receives each round. Once this energy level drops to zero, the agent dies.

Other variable population size EC systems include APGA, PRoFIGA, and the PSO-GA hybrid algorithm (PGHA) [Shi, Liang, Lee, and Wang 2005]. The PGHA has similarities to our research because agents that are parents continue to live, but the longer an agent lives, the higher the probability that it will die. The PGHA uses probability to determine if an agent is alive each round, allowing for a variable population size. In our work, the population size varies depending on resources with the agents gaining energy through food while they lose energy through activities with age affecting the amount of loss. In PGHA, selection is based on fitness and in our work, selection is primarily based on location (the proximity of two agents).

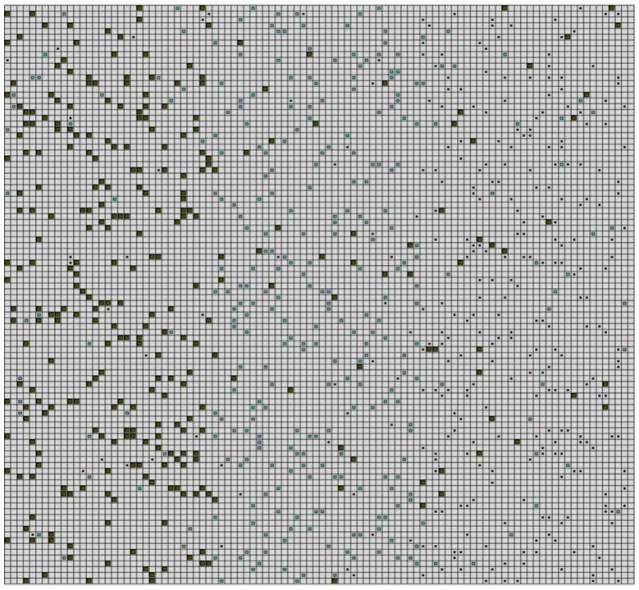

Figure 1: 100 x 100 grid environment populated with seeds but with no agents. The large seeds are mainly in the left third, medium seeds are mainly in the middle third, and small seeds are mainly in the right third.

Environment

The simulated environment was designed to be as simple as possible, yet complex enough to create situations where speciation could occur. The environment is a grid of discrete blocks (Figure 1) where each square represents a possible position for food and/or agents. The length and width of the grid can be set depending on the application. In the research reported in this paper we used a 100 by 100 grid environment. At each time cycle, the agents all move one block as specified by their controller and food pops up in the grid, but does not move, although it is removed if eaten by an agent. The agents are capable of moving, eating, and interacting with each other.

The food in the environment is made up of seeds with three different sizes: big, medium, and small. The density of each seed type can be specified, which dictates the numbers of each seed type added to the grid during each cycle of time. The grid is divided into divided into 3 sections: the left third, middle third, and right third; where differing distribution probabilities for each seed type can be specified. For example, a big seed could have an 80% chance of occurring in the left third of the grid, a 10% chance of occurring in the middle third of the grid, and a 10% chance of occurring in the right third of the grid. Having the grid environment split up into different sections simulates different distinct overlapping environments, which could be used to replicate different climates or soil types.

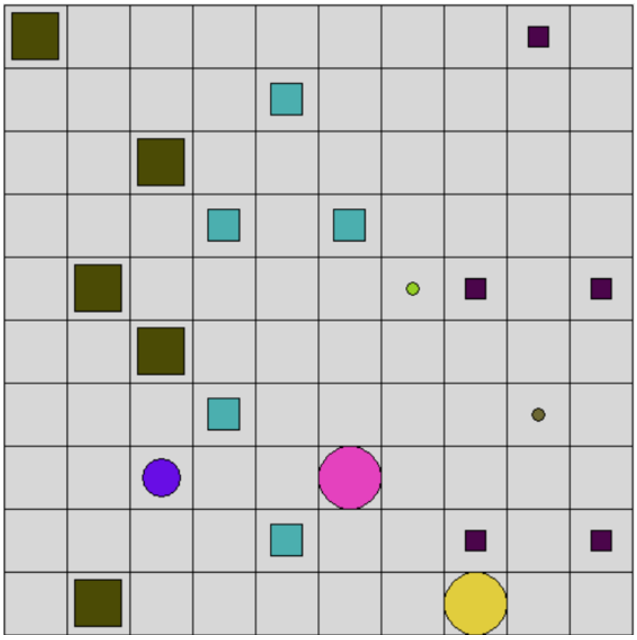

Figure 2: A 10 x 10 grid with agents (circles) and seeds (squares) of varying sizes

Agents

Agents move around within the environment, eat seeds, and have the potential to interact with each other. The amount of agents in the environment will increase and decrease depending on the environment and food sources. Each agent has its own chromosomes, which specify its characteristics, such as movement preferences, mating preferences, and physical attributes.

The physical characteristics of the agents are color and size. The initial population of agents receive random colors. An agent's color is stored as an RGB value, but as new agents are created, their color is inherited from the parents' colors which chance for mutation (adding/subtracting from a parents RGB value). The agents have varying sizes of big, medium, and small. The size of the agent corresponds to which seed that agent is most capable of eating. Figure 2 shows a close-up of the grid with a variety of different seed sizes (with distributions designated so large seeds are most likely in the left, medium in the middle, and small in the right) and agent sizes.

The agent's chromosomes dictate what activity it prefers depending on its surroundings, which will be discussed further in the agent controller section of the paper. An agent starts with 250 life and if it eats a seed it will get a certain amount of life depending on the agent's size and the seed being eaten as depicted in Table 1.

		Seed Size		
		Small	Medium	Large
Agent Size	Small	30	10	0
	Medium	10	30	10
	Large	5	10	30

Table 1: Energy from seeds

All the other actions have a cost and will decrease an agent's life, which is usually in proportion to the agent's age so older agents take more energy to survive each time cycle. An exception is when agents attempt to reproduce, which will cost both agents a standard 80 amount of life. An agent can only mate with another agent in its adjacent vicinity. If this happens and is successful, offspring agent is produced and placed into the environment within three blocks from the parents. Reproduction between agents of varying sizes will not always be successful. The probability that agents of differing sizes successfully produce an offspring can be seen in Table 2.

		Agent 1 Size		
		Small	Medium	Large
Agent 2 Size	Small	100	30	15
	Medium	30	100	30
	Large	15	30	100

Table 2: Percent probability agents of differing sizes successfully produce an offspring

Chromosomes

Each agent has two chromosomes: one that specifies its actions in different circumstances (this will be covered in the next section on control) and one that determines what it is looking for in a partner for reproduction. It dictates the desired partner size, maximum color difference between partners, and desired age range in the partner (Figure 3).

Agent 1 - **Size:** Medium, **R color:** 100, **G color:** 100, **B color:** 100, **Age:** 55

Chromosome: 1 1 0 0 1 1 0 1 0 0 1
Size Desires: 1 1 meaning any size
Max Color Dif: 0 0 1 meaning 189
Idea Age Range: 1 0 1 meaning 110 - 124
Actual Age Range: 0 0 1 meaning up and down 1 index spot, 95 - 139

Agent 2 - **Size:** Large, **R color:** 50, **G color:** 50, **B color:** 50, **Age:** 135

Chromosome: 0 0 1 1 0 0 0 0 1 0 0
Size Desires: 0 0 meaning medium
Max Color Dif: 1 1 0 meaning 664
Idea Age Range: 0 0 0 meaning 35 - 49
Actual Age Range: 1 0 0 meaning up and down 4 index spots, 35 – 109

Figure 3: Example mating preference chromosomes of two agents that would successfully reproduce with each other given their chromosomes and specific information.

There are four possibilities for desired partner size: medium, small, large, and any. The first 2 bits of the chromosome represent what an agent is looking for in partner size (00:medium, 01:small, 10:large, 11:any-size).

Following the size preference bits, the next three bits designate the maximum desired color difference in a partner agent. To find the difference in color we find the difference between the various RGB values of both agents, take the absolute value, and then add them up. This means that the smallest color difference between agents can be 0 and largest can be 765 (3 * 255). The range of color difference between two agents goes from 0 to 765. The range is broken up into chunks of 94 giving us the range: 0-94, 95-189, 190-284, 285-379, 380-474, 475-569, 570-664, and 665-765.

Agents grow older every time step in the simulation. The age of reproductive fertility was defined to be between the ages of 35 and 154. For desirability of a mate, this was broken up into eight sections: 35-49, 50-64, 65-79, 80-94, 95-109, 110-124, 125-139, 140-154. The next three bits of the chromosome designate an ideal lower bound and upper bound age of the partner. This represents the target age with the final 3 bits of the chromosome representing an added range for mating. The values represent a plus and minus index value from the ideal range.

All of these three mating characteristics can best be seen in the example in Figure 3. We can see that the two agents would reproduce. Agent 1 desires an agent of any size and Agent 2 desires an agent of medium size. In both cases the agents are compatible. The color difference between the two agents is 150 and both agents allow for a maximum color difference that is more than 150, so the color traits show the agents are compatible. The desired age range for Agent 1 is 95 - 139 and Agent 2 has an age of 135. The desired age range for Agent 2 is 35 - 109 and Agent 1 has an age of 55. The age of both agents fall into their partners desired range, so the age traits show that they are compatible. From this, the agents will choose to mate and attempt to create a child.

Controller

The second chromosome of each agent determines its actions while operating in the grid. This chromosome is 32 bits, with it having 8 genes of 4 bits each.

The genes are used to set the priorities of 8 possible action-rules (antecedent / consequent) that control the actions of the agent. If the antecedent of the rule is true, then the rule fires and the action in the consequent is taken. If more than one rule has the potential to fire, the priorities designated in the chromosome act as a means of conflict resolution. The agents are capable of performing the following 8 actions: move to a free space, move to a free space and attempt to reproduce, move to a big seed space and do not eat it, move to a medium seed space and do not eat it, move to a small seed space and do not eat it, move to a big seed space and eat it, move to a medium seed space and eat it, move to a small seed space and eat it. At each cycle of time the immediate surroundings of an agent (one square up, down, left, and right) are analyzed in conjunction with the antecedent of the rules to see if any rules fire. This is used to determine, depending on the rule priorities, the action from the rule base that the agent will perform.

Rule Base	Free Space	Reproduce	Big Seed No Eat	Med Seed No Eat	Small Seed No Eat	Big Seed Eat	Medium Seed Eat	Small Seed Eat
Chromosome	1 1 0 1	0 1 1 1	1 1 1 0	1 0 1 0	0 0 0 0	1 0 1 0	0 1 1 1	1 0 0 0
Decimal Value	13	7	14	10	0	10	7	8

Figure 5: An example action chromosome that determines priorities for the rule based system.

Figure 5 depicts a single agents' chromosome broken into its separate parts and what each part of the chromosome represents. The chromosome is broken into eight four bit sections. Whichever section has the highest decimal value is the action that is performed relative to the agent's surrounding. In this example, the agent would most want to move to a big seed area and not eat the food. If one of the adjacent spaces has a big seed, this is what it would do. If none of the adjacent spaces contains a big seed, this rule will not fire and the next highest priority will be considered. In the example, the agent would then move to a free space if one is available. This process of finding the highest rule to fire is continued until a rule is fired. In the case of a tie in rule priority, one of the tied rules is randomly picked.

Genetic Algorithm

The crossover and mutation operators for a genetic algorithm (GA) are used to create the chromosomes of the child of two mating agents, which allows the agents to pass their characteristics to their offspring. The general process of a GA is to have a population of individuals, where each individual has a chromosome, usually made up of 0's and 1's. Selection of two individuals is typically based on their fitness score where more fit individuals have a higher chance of reproducing. After two individuals have been selected, crossover of their respective chromosomes occurs. A random point in the chromosomes (for single point crossover) is picked and all the bits from one parent up until this point are concatenated with all the bits from this point until the end of a chromosome of the other parent. It is random which parent gets the first bits in the new chromosome and which gets the end bits in the new chromosome. After crossover, each bit in the new chromosome is subjected to mutation. This involves going bit by bit through the chromosome and with a very low probability flipping a bit to a 1 if it is a 0 or vice versa.

After selection, crossover, and mutation have occurred, a new chromosome (individual) has been created. In a generational GA, this whole process is repeated until a new population is created which will then replace the older population. In a steady-state GA, the newly created individual is added to the population, usually replacing one of the current individuals in the population. In this way, the population is constantly changing and improving while maintaining the same number of individuals. The GA used in our research is similar to the steady-state GA described above, but the difference in this regard is that there is no replacement with our GA. New individuals are added to the population with no regard to the overall size of the population.

The major difference between standard generational / steady-state GAs and the one used in this research is in selection. In our research, selection is largely predicated on proximity to another agent rather than fitness. Agents will not try to reproduce unless they are in adjacent positions. The idea of selection weighted by agent fitness does not apply here. Instead, any agent can attempt to reproduce, however the fitter agents should produce more offspring because they will live longer and have more energy for reproduction.

Results

The first test that was run involved the establishment of a population of homogeneous individuals best adapted for the environment. As described previously, agents of medium size receive the most life from medium seeds. Agents of large and small size receive marginal amounts of life from medium seeds. The environment was set up such that only medium seeds existed in the grid in a 10/80/10 distribution. The initial random population of individuals was created, and as expected all the big and small agents died out while a population of medium agents established themselves in the middle third of the grid. This showed that the simulated environment that we developed will allow for individual populations to be established that are optimized for the environment.

The next test involved the migration of a population. A population of small agents was established in the right third of the grid (where the small seeds were most prevalent). After the population was established, the location of the small seeds changed. The small seeds changed to become most prevalent in the middle third of the grid and less prevalent in the right third of the grid. This meant that the population of small agents suddenly lacked enough food to sustain themselves because they were in the right third of the grid. Nevertheless, the population did not die out. Instead the agents migrated. The population left the right third of the grid and established themselves in the middle third of the grid due to food prevalence. This showed that individuals will migrate and follow the food source in order to survive.

The evolutionary capabilities of the agents and the adaptability of the population in this environment were also tested. A population of small agents was established in the right third of the grid. Originally the food source in this part of the grid was small seeds with no medium or large seeds in the environment. This was then changed such that the only food source in this area was medium seeds, with no other type of seed. As expected the population of small agents evolved and became a population of medium agents. This test gave evidence that the program was capable of replicating some form of survival of the fittest since when the agents with medium size became the fittest, they survived and passed their traits on to subsequent generations.

The next test was a combination of the previous two tests. A population of small agents in the right third of the grid was established. The production of small seeds in the right third of the grid ended and at the same time the production of medium seeds in the middle third of the grid began – medium seeds mainly existed in the middle third of the grid. The population of small agents survived by migrating to the medium seeds and then evolving to become medium sized. This test showed that agents were able to migrate and evolve in order to survive.

Conclusions

We believe our results show that our simple environment is a good testbed for further studies. We only have three different sizes of agents and three different sizes of seeds. The setup is not complicated and yet it managed to replicate key areas of biology, such as aspects of Darwinian Evolution, adaptation, survival of the fittest, migration, and a combination of migration and survival of the fittest. Our next step is to use this environment in an attempt to replicate speciation.

References

Arabas, J., Michalewicz, Z., and Mulawka, J. 1994. GAVaPS-a genetic algorithm with varying population size. *Proceedings of the First IEEE Conference on Evolutionary Computation.* IEEE World Congress on Computational Intelligence.

De Jong, K. 2006. *Evolutionary Computation A Unified Approach*, MIT Press, Cambridge, MA.

Glibovets, N. and Gulayeva, N. 2013. A review of niching genetic algorithms for multimodal function optimization. *Cybernetics and Systems Analysis* 49.6 (2013):

815-820.

Goldberg, D. and Richardson, J. 1987. Genetic algorithms with sharing for multimodal function optimization. *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms.*

Gustafson, S. and Burke, E. 2006. The speciating island model: An alternative parallel evolutionary algorithm. *Journal of Parallel and Distributed Computing* 66.8 (2006): 1025-1036.

Potter, M. and De Jong, K. 1994. A Cooperative Coevolutionary Approach to Function Optimization. *Proceeding from the International Conference on Parallel Problem Solving from Nature* (PPSN).

Potter, M. and De Jong, K. 1995. Evolving neural networks with collaborative species. *Summer Computer Simulation Conference*, Society for Computer Simulation.

Shi, X., Liang, Y., Lee, H., and Wang, L. 2005. An improved GA and a novel PSO-GA-based hybrid algorithm. *Information Processing Letters* 93.5 (2005): 255-261.

Stanley, K. and Miikkulainen, R. 2002. Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10.2 (2002): 99-127.