Evolving Behavior using Genetic Algorithm: Natural Selection

Aridaman Singh Nandan
Department of Computer Science and Engg.
NIT-Jalandhar
Punjab, India
aridaman1995@gmail.com

Geeta Sikka
Department of Computer Science and Engg.
NIT-Jalandhar
Punjab, India
sikkag@nitj.ac.in

Abstract—The fitness function based method of artificial evolution is asserted to be inadequate for real world entities. Evolutionary computing refers to the paradigm of artificial intelligence without an explicit fitness function. Evolution of entities via natural selection is a way forward. This paper is an extension of Channon's and Damper's natural selection work and further focus on the principles of development of evolving neural network. To test the extended idea, a world(artificial) containing self evolving organisms is created and described. Rather than writing thoughtful fitness functions, we let a process found in nature-*evolution*-decide for us. Results show that such systems are well suited to long-term incremental development system.

Index Terms—Evolutionary computing, natural selection, neuroevolution

I. INTRODUCTION

Evolutionary Computing(EC) [5] is an algorithmic paradigm to solve search and optimization problems using core principles behind Darwinian evolutionary theory. There will be no discussion of protein synthesis, RNA and other topics related to actual biological process, rather we develop algorithms inspired by these principles. Can we think of DNA as object(brain) to an artificial organism? [2] Can objects generate new objects and pass on their DNA? The answer to the questions is yes. This paper look into biological principles of evolution and find way to apply those in code [3]. We simulate one of the most powerful process found in nature without going into depth of science of genetics. I encourage readers to explore possibilities for expanding the scenario provided and don't care so much on accurate simulation of evolution; rather, care upon applying evolutionary strategies in software.

II. TECHNIQUE TO EVOLVE BEHAVIOR

Artificial life presents us with problem which is difficult to specify to a machine. Therefore we should either increase our grasp on it, or create something which outperform the statements we give it [1]. The first possibility is not appropriate as it is based on traditional fitness function approach. It also includes manually increasing the understanding and modelling life by impressive systems within their environments.
The second option has no explicit evaluation function. Through organism interaction with environment(external stimulus) certain behaviors evolve from them without any long-term goal

[9]. Algorithm is implemented in specific way to solve organism behavior problem. As evolution is applied to inherently complex problems [3] finding adequate evaluation function becomes an obvious problem. At the same time natural selection is necessary for emergence of new generation of organisms but it doesn't imply sustained emergence. The changes in genotype must be taking place slowly and must be correlated with the fitness evaluation of an organism [10].

III. EVOLVING STRUCTURE

The organisms need to evolve over time so they need a human brain like structure. But how should the brain network be specified? The emergence of new generation requires new neural structures as well. There are various known examples of structures that can serve our purpose.
Gene theory is one of the solution. At any stage of development, in any one cell, only a small proportion of genes will be used. The gene lies in the cell and cell in the brain itself. The duplicate system could be obtained either by gene duplication or along with development process.
Direct duplication of gene as a source of neural structure duplication is not feasible(as it will only create a copy of parent). Therefore, for constructive evolutionary emergence of complex behavior, a mutation process is required. A newly created organism receive a duplicated copy of its parent, but the development process mutates genes in some constraints; based on some mutation rate. This modular development lead to emergence of new unexpected(may be good or bad) behavior. New combination of features occur and organisms evolve [2].

IV. NATURAL SELECTION PROCESS

Before we walk into the scenario and algorithm, let's take moment and describe Darwinian principles of evolution that will be required in our simulation. For the simulated natural selection process to occur as it does in nature, three of these elements must be present.

1) **Heredity.** It is the process by which parent properties or genes are transferred to children. The traits are passed down to their children in the next generation of organisms, if they live enough to reproduce. Hence the

newly created organism's brain will be inherited from parent.

2) **Variation.** Diversity of traits must be present in the population or a means to introduce variety must exist. Consider the example: A population of organisms in which all are exactly same: same color, same size, same everything. There is no variety in population; the children will always be identical to either parent or to each other. Nothing can evolve as new combination of traits will never happen.

3) **Selection.** This is the mechanism by which some of the population have the chance to pass down their genetic information and some do not. Such a model is called *survival of the fittest.* For example: A goat population is chased by lions every day. The faster once are likely to escape from the lions and can live longer and have a chance to reproduce to pass down their genes to their children. The term "fittest" however, can be misleading. It has no relation with organism being physically fit. In the case of our example, we define our own fitness criteria based on the environment.

V. THE NEURAL SYSTEM: DNA ENCODING

A. Scenario

We create an artificial life scenario to validate the above. The Asystem is a world which contain food, poison and organisms. The population of organisms will survive or die eating food or poison. Each creature has attraction and repulsion to food and poison. Along with this, they have an extent to sense food and poison; so they only eat those which are in their scope. The one with greater extent to sense food with higher repulsion to poison lives longer and get chance to reproduce, and the one with lesser attraction to food die early.

B. Genotype

The digital information-the actual genetic code is an organisms **genotype**. In simple words, it is the data that simulate the brain functionality. From generation to generation this information is passed down. What are entities in the world? How to design the genotype for the entities? What data structure would you use to store the properties? How to choose the encoding for the system? Attention to be paid to developing system in which children's network must resemble parent aspects-keeping in mind Darwinian principle of heredity.

Taking into account the proposed development environment system, each organism's DNA must have the following(Fig. 1):

1) **Attraction/Repulsion to food**
2) **Attraction/Repulsion to poison**
3) **Radius to sense food**
4) **Radius to sense poison**

The key part is that each member of population has a virtual *DNA*. These above mentioned properties become "genes" for organism. This set out how an organism will look or behave. Based on this DNA an organism will move in the system(left, right, up, down). The sensing proportion of the gene will sense
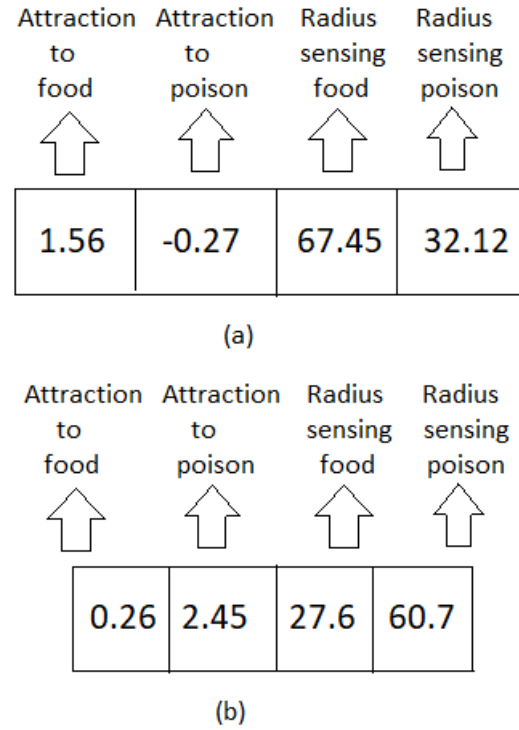

Fig. 1. Genotype: DNA encoding for an organism

the food in scope and remaining proportion decide whether it should be eaten or not(attraction or repulsion). The DNA drives the behavior of organism.

The DNA is encoded as an array of floating point numbers of size equal to the set of properties. It is randomly initialized at the time of construction. Encoded value is a non-zero lying within the constraints. We also define attraction/repulsion range between -3 and 3 and radius range as 5 to 100 pixels. Figure 1 shows DNA of two different organisms. The first one is more attracted to food with higher food sensing scope whereas the second one is much attracted to poison with greater poison sensing radius. Clearly, the later one would not be able to survive for long and get chance to reproduce. This defines the organisms character. The more the organism can sense food with higher attraction or the more the organism is repulsive to poison, the better it is for survival.

VI. DEVELOPMENT SYSTEM

For simulating natural system, let's look at a way in which we use genetic algorithm to build something that resembles the *ecosystem*. We create organisms(triangular shape) that moves about the screen searching food. Higher the sensing radius, the more it can locate and decide whether to eat or not.

```
1 class Organism {
2   this.dna = [random(-3,3), random(-3,3), random
       (5,100), random(5,100)];
3   this.health;
4   .....
```

```
5 }
```

Listing 1. DNA of Organism

We will store the population of organisms in a list(not in array) as we expect population to grow and shrink as how often organisms die or are born. We initialize the elements of list as organisms(Algorithm 1).

Algorithm 1 Making initial population of organisms

$world\_population \leftarrow 500$
$population \leftarrow list\ of\ organisms$
for $i \leftarrow 1$ to $world\_population$ **do**
  $organism \leftarrow create\ new\ Organism\ object$
  $population \xleftarrow{\text{add}} organism$
end for

So far we have an organism that moves across the window and code that manages variable number of organisms. To make it into a system that evolves, we add features to our world:

- **Organism die**
- **Organisms are born**

Organism dying acts as a replacement for fitness function. If organism dies, it no longer exits and cannot be selected to be parent! To ensure deaths in world we add health variable to Organism class(listing 1). An organism gets 100 health points when born. In each iteration of loop(animation) organism loses some health.

$$health = \begin{cases} +100, & \text{if } food & (1) \\ -100, & \text{if } poison & (2) \\ -1, & \text{otherwise} & (3) \end{cases}$$

If organism health drops down to 0, it dies.

Algorithm 2 Function to test if organism dead or not

if $health \leq 0.0$ **then**
  $return$ **true**
else
  $return$ **false**
end if

Perhaps you are thinking that all organisms start with 100 points and loses 1 point per iteration, then all will live for same amount of time and then die together. If this happens then there will be nothing to evolve. Hence we introduce food and poison. Organisms life is extended when it eats food as this increases its health points.

let's assume *food*(and *poison*) as list of vector co-ordinates of food(and poison). If food(or poison) lies in proximity, it eats food(or poison) and increase(or decrease) its health.

In this scenario organisms can eat food(Algorithm 3) to live longer and have likelihood to reproduce. The system will evolve organisms with ability to find and eat food.

VII. GENOTYPE AND PHENOTYPE

Its important to know distinction between concepts of *genotype* and *phenotype*. The actual genetic data Figure 1 is

---

Algorithm 3 Function to eat food

$i \leftarrow size\ of\ food\ list$
for $i \geq 0$ **do**
  $food\_location \leftarrow food[i]$
  $dist \leftarrow distance(food\_location,\ organism\_location)$
  if $dist \leq dna[2]$ **then**
    $health \leftarrow health\ +\ 100$
    $food\ list \xleftarrow{\text{remove}} food\_location$
  end if
end for

Algorithm 4 Function to eat poison

$i \leftarrow size\ of\ poison\ list$
for $i \geq 0$ **do**
  $poison\_location \leftarrow poison[i]$
  $dist \leftarrow distance(poison\_location,\ organism\_location)$
  if $dist \leq dna[3]$ **then**
    $health \leftarrow health\ -\ 100$
    $poison\ list \xleftarrow{\text{remove}} poison\_location$
  end if
end for

---

element's genotype. The **phenotype**, however, is the expression of that data. This decides how we use genetic algorithm in our work [10]. How will we outline genotype to visualize it as phenotype? What does genotype variables express? This is done by graphic programming. Consider probably simplest color example in Figure 2. As you see, genotype is digital data stored in integer variables and we choose to express that integer as color. But choosing expression for data is arbitrary. In other scenarios, you can use these integers to express length of line, weight of force etc. How can we express our organism? What could be the phenotype?

The organisms ability to locate food(or poison) is tied to two variables(genes)-

1) **Sensing genes.** To what extent organism can locate food(or poison)
2) **Attraction/Repulsion genes.** How quickly it moves toward the food(or poison)

Organisms with higher sensing power find food more often and one with higher attraction power cover more ground in shorter period of time.

Figure 3 shows organism is represented in a triangular

Genotype	Phenotype
Int c = 255;	⬜
Int c = 127;	⬛
Int c = 0;	⬛

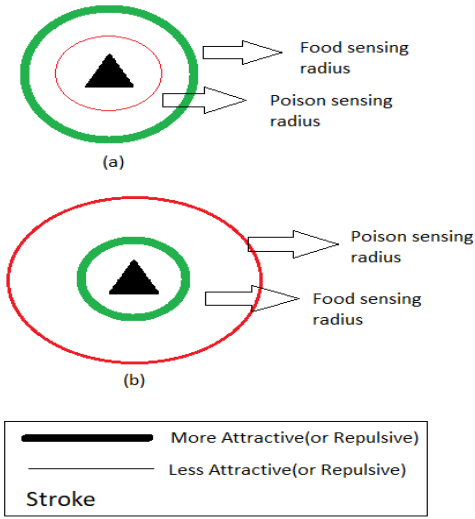Fig. 2. Phenotype visualizing corresponding to Genotype

Fig. 3. Phenotype: Expression of virtual DNA

shape, and DNA phenotype is the circles around it. The two circles tend to locate food and poison and attraction/repulsion power is represented by stroke of circles. The more the stroke the more it tends to move towards the food or poison. Stroke is just mapping of steering force(attractive or repulsive) to circle outline width. Displaying phenotype is based on physics engine and consist of following steps:

1) Draw the triangle
2) Set stroke using mapping function on dna[0] for food
3) Set stroke using mapping function on dna[1] for poison
4) Draw food circle using dna[2]
5) Draw poison circle using dna[3]

VIII. SELECTION AND REPRODUCTION

A. Selection

Now having genotype and phenotype, we move to next step to devise a means for organisms to be selected as parents. As stated above longer living organisms have more chance to reproduce. The life length of organisms is its *fitness*.

One possibility could be to presume that whenever two organisms come in proximity of each other, they make a new organism [7]. The longer an organism lives, the more probable it is to come in scope with another organism. So in addition to eating food, they also need to find another organism for the likelihood of having child.

Other possibility would be to have *asexual* reproduction. An organism will not require a partner. At any moment, it can make a clone of itself. The cloned organism will have same genetic make-up. So selection algorithm becomes:

At any instant, organism has 1% chance of reproducing.

The longer an organism lives, the more chance is there that it make atleast one child. Selection algorithm can be implemented by writing function in Organism class that harvest a random number every iteration(animation frame). If picked number is lesser than 0.01(1%), a new organism is born.

B. Reproduction

How does an organism reproduce? The reproduction process will involve a function in Organism class that will make a new organism object from newly made DNA. Since single parent is making a child, we call a *copy* function instead. It copies the content of one array into the another.

Algorithm 5 Reproduce function returning new organism: the child

---

  if $random(1) \leq 0.005$ **then**
    $child\_dna \leftarrow Make\ a\ copy\ of\ dna$
    $child\_dna \leftarrow mutate(child\_dna,\ 0.01)$
    $child \leftarrow new\ Organism(location,\ child\_dna)$
    $return$ **child**
  else
    $return$ **null**
  end if

The child is born at same location as that of parent. Also note we've reduced reproduction probability from 1% to 0.5%. This makes a huge variation; with greater probability the ALife will rapidly move in direction of overpopulation. A lower probability, and everything will die out quickly. Once child DNA is created we put in the final process before adding it to the next generation-*mutation*. It exist because of principle of variation. It permits to introduce diversity through evolutionary process itself [2].

Mutation is expressed in terms of mutation rate. Genetic algorithm might have 1% or 0.5% etc. as mutation rate. A mutation rate of 1% means that there is 1% chance that it will mutate the gene. What does it mean to mutate? In this case, mutation means adding some random value to the gene. This random value could be positive or negative depending on the constraint described.

The process is shown in Figure 4. The behavior of system is greatly affected by mutation rate. Obviously, higher mutation rate say 80% would nullify the evolutionary process. If major parts of gene are generated randomly, we can't assure that "fit" genes occur with greater frequency. Algorithm 6 mutate child's genes based on mutation probability.

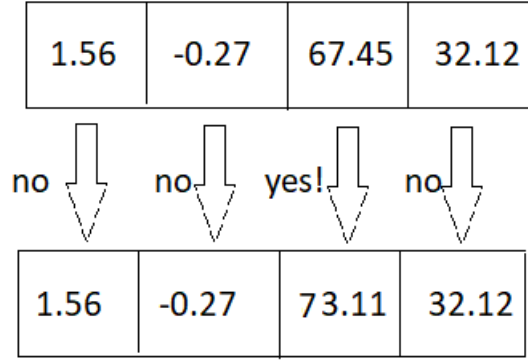**Algorithm 6** Mutate function mutating genes array based on mutation rate

---

  for $i \leftarrow 0\ to\ 3$ **do**
    if $random(1) \leq mutation\_rate$ **then**
      if $i\ <\ 2$ **then**
        $genes[i] \leftarrow genes[i]\ +\ random(-0.2, 0.2)$
      else
        $genes[i] \leftarrow genes[i]\ +\ random(-10, 10)$
      end if
    end if
  end for

IX. EXPERIMENTAL WORLD

Now we have all pieces in place to validate the above. A two dimensional world of artificial organisms is created and

Fig. 4. Mutation process on DNA genotype

Fig. 5. Ecosystem Simulation

each organism is controlled by network using the described development system. Evolution is governed by natural selection. No universal rule exist to delete organisms; this is under their own control. Each organism has a facing direction which is set randomly at birth. Sharp points indicate their direction. Figure 5 shows organisms with dominant radius are more fit(green) as compared to the ones with smaller search radius. Putting entire system together in Algorithm 7.

Algorithm 7 Genetic Algorithm: Evolving Ecosystem

Initialize: Generate population of N organisms with random DNA.
loop
  for *each alive organism* **do**
    Update organism position vectors.
    Call eatFood and eatPoison procedures.
    Call reproduce procedure for current organism.
    if *child is born* **then**
      Add born child to population.
    end if
  end for
end loop

X. RESULTS

- **Convergence and similarity:** When organism reproduce, the child's network almost resembles the parent's network. Examination of experimental world shows resemblance between many of them. Throughout the evolution, population remains converged in small number of species(choosing the reproduction and mutation rate carefully). The fitness criteria also met the developmental landscape, making it appropriate for long-term evolution.
- **Increasing Complexity:** For a short period, the organisms with basic genotype move randomly across the
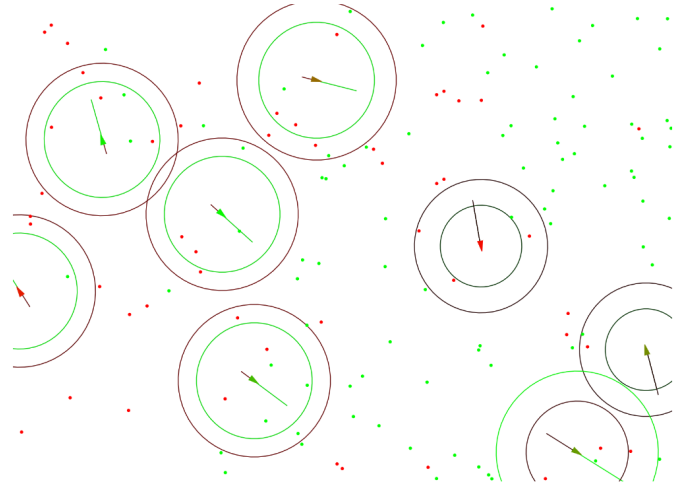
system. For next two to three thousand frames(iterations), we find organism's strategy of only eating food(or otherwise die) and having high repulsion to poison. At lateral stages when food is left in small proportion as compared to poison; organisms die but don't eat poison. Some networks do better than others but not so well to display a dominating effect. Success is understandable if we watch it in action.

Some organisms with lower food detection radius but higher poison detection radius do well(live longer) because of attraction/repulsion genes. Once the food is in range, high attraction to food steers the organism towards it abruptly(may be previously it was going to eat poison). If there are dominant organisms in ecosystem, then the world always have space for next generation to exist.

Taking count of number of organisms born and died, suggest that further generation emerge(Fig. 6). However, behavior level analysis is difficult. Observation reveals that current generation share characters with previous species but are still different. Population also start remaining steady after typical run.

XI. CONCLUSION AND FUTURE WORK

The organisms natural selection in its environment is completely analogous to fitness function based algorithm; hence approach is viable. Selection and reproducing system results in similar looking genotypes and phenotypes with gradual variation over long-term evolution. This also makes clear that certain operations at low-level can obstruct and limit the evolution(birth & mutation rate). Once an organism has evolved, it can sustain itself for a period of time. We can implement crossover and mutation between two parent organisms when they are in certain proximity of each other. We can make organisms with gender. Can we create a scenario in which organism learn to cooperate with each other; can revive a dying organism.
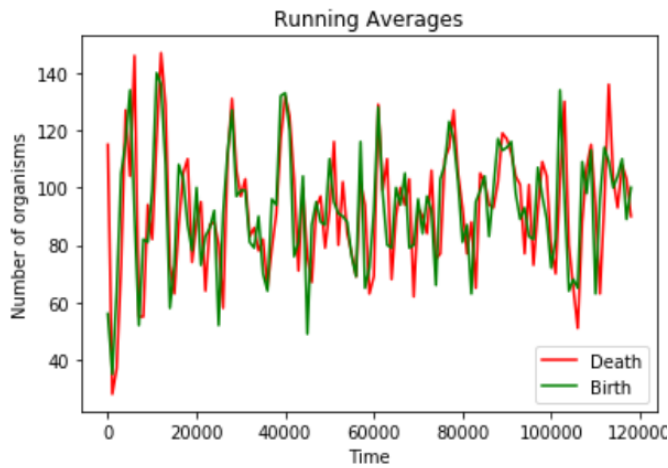
Fig. 6. Regular run: organisms birth and death count

REFERENCES

[1] Channon, Alastair D., and R. I. Damper. "Evolving novel behaviors via natural selection." (1998): 384-388.

[2] Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." Evolutionary computation 10.2 (2002): 99-127.

[3] Rodzin, Sergey, Olga Rodzina, and Lada Rodzina. "Neuroevolution: problems, algorithms, and experiments." 2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT). IEEE, 2016.

[4] Xue, Bing, et al. "A survey on evolutionary computation approaches to feature selection." IEEE Transactions on Evolutionary Computation 20.4 (2015): 606-626.

[5] Ryerkerk, Matt, et al. "A survey of evolutionary algorithms using metameric representations." Genetic Programming and Evolvable Machines (2019): 1-38.

[6] Darwish, Ashraf, Aboul Ella Hassanien, and Swagatam Das. "A survey of swarm and evolutionary computing approaches for deep learning." Artificial Intelligence Review (2019): 1-46.

[7] Maul, Tomás H. "Improving Neuroevolution with Complementarity-Based Selection Operators." Neural Processing Letters 44.3 (2016): 887-911.

[8] Agogino, Adrian, Kenneth Stanley, and Risto Miikkulainen. "Online interactive neuro-evolution." Neural Processing Letters 11.1 (2000): 29-38.

[9] Sugisaka, Masanori, Xiaoshu Wang, and Ju-Jang Lee. "Genetic algorithms (GAs) to evolve multiple-agent cooperative systems." Artificial Life and Robotics 3.3 (1999): 139-142.

[10] Ekárt, Anikó, and Peter R. Lewis. "Genotype–phenotype mapping implications for genetic programming representation: Commentary on "On the mapping of genotype to phenotype in evolutionary algorithms" by Peter A. Whigham, Grant Dick, and James Maclaurin." Genetic Programming and Evolvable Machines 18.3 (2017): 369-372.

[11] Mohamad, Mohd Saberi, et al. "Gene subset selection using an iterative approach based on genetic algorithms." Artificial Life and Robotics 14.1 (2009): 12-15.

[12] Goldberg, David E., and John H. Holland. "Genetic algorithms and machine learning." Machine learning 3.2 (1988): 95-99.

[13] Shiffman, Daniel, Shannon Fry, and Zannah Marsh. The nature of code. D. Shiffman, 2012.

[14] Cristea, Paul, Aurel Arsene, and Bogdan Nitulescu. "Evolutionary intelligent agents." Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512). Vol. 2. IEEE, 2000.

[15] Guo, Pengfei, Xuezhi Wang, and Yingshi Han. "The enhanced genetic algorithms for the optimization design." 2010 3rd International Conference on Biomedical Engineering and Informatics. Vol. 7. IEEE, 2010.

[16] Herrera, Francisco, Manuel Lozano, and Jose L. Verdegay. "Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis." Artificial intelligence review 12.4: 265-319.

[17] Lim, Ting Yee. "Structured population genetic algorithms: a literature survey." Artificial Intelligence Review 41.3 (2014): 385-399.

[18] Sekanina, Lukáš. "Evolvable computing by means of evolvable components." Natural Computing 3.3 (2004): 253-292.