Evolutionary Computing and Its Application

As the name suggests, evolutionary computing is a type of artificial intelligence that solve problems using techniques from biological evolution. It simulates evolution on a computer. Kendall (2018) mentioned that evolutionary computing is derived from the Darwinian Principle, which is well-known as the survival of the fittest. Hemingway et al. (2005) postulated that the simulation will produce a series of optimisation algorithms, constructed from a set of attributes that resembles one of a genome.

According to Eiben and Smith (2015), the underlying metaphor of evolutionary computing associates with that of trial-and-error, which is a type of problem solving. In other words, evolution is analogous to problem solving. In evolution, there exists an environment that is populated by a group of individuals that procreates and fights for survival. Eiben and Smith (2015) also mentioned that the fitness of these individuals depends on how well they succeed in surviving the environment and procreate.

On the other hand, the trial-and-error problem solving process has a collection of possible solutions. Their quality, which is measured based on how well they solve the problem, governs the possibility of their future usage in making further candidate solutions.

Evolution		Problem Solving
Environment	⟷	Problem
Individual	⟷	Candidate Solution
Fitness	⟷	Quality

Table 1. The association between evolution and problem solving

The history of evolutionary computing can be said to date way back to the 1940s. In 1948, Alan Turing came up with genetical or evolutionary search. It was not until 1962 did Bremermann conducted computer experiments for evolution and recombination using optimization techniques.

In 1964, Rechenberg introduced evolution strategies. On the following year, Fogel, Owens and Walsh came up with evolutionary programming. In 1975, Holland invented genetic algorithms. During the early 1990s, Koza coined the idea of genetic programming. In Introduction to Evolutionary Computing, Eiben and Smith (2015) stated that the current nomenclature of evolutionary computing indicates its whole field that is comprised by a plethora of algorithms. The algorithms used are called evolutionary algorithms and it has many variants such as evolutionary programming, evolution strategies, genetic algorithms, and genetic programming.

As mentioned earlier, evolutionary computing is heavily inspired by the Darwin's theory of evolution. The first foundation of evolution progress is the competition-like selection. It was theorised that all environments have finite resources that can only sustain a limited number of individuals. Given all lifeforms come with basic instincts to procreate, it is then up to natural selection to decide who will survive. In order to increase their chances of procreation, the individuals must compete with each other for resources.

The second foundation of evolution progress is the phenotypic traits of the individuals in population. Eiben and Smith (2015) also asserted that phenotypic traits are "those behavioural and physical features of an individual that directly affect its response to the environment (including other individuals), thus determining its fitness". These traits are determined from both inheritance during birth and nurture during development. As a consequence of random changes, these traits are unique to each individual. If these traits result in higher chances of procreation and heritable, then they will tend to be passed down to future generations. In the Darwinian Principle, it was also theorised that "small, random mutations in phenotypic traits" will occur during procreation from one generation to another. The evolution will thrive by the survival and procreation of the best individuals. In short, population is made up of diverse set of individuals. Their success in procreation is dependent on how well they adapt to the environment, in comparison with other individuals. When the successful individuals procreate, it will result in variations through mutations that acts as a continual source of diversity. Essentially, this means that population is the measure of evolution.

In applications of predictive analytics such as stock market price prediction, recommender systems, and customer segmentation, feature selection is used. According to Gomez and Quesada (n.d.), feature selection can be described as:

The process of finding the most relevant variables for a predictive model. These techniques can be used to identify and remove unneeded, irrelevant and redundant features that do not contribute or decrease the accuracy of the predictive model.

One of the most used evolutionary algorithms in feature selection is genetic algorithms. This is due to the fact that it "optimizes the performance of a predictive model, by selecting the most relevant features" (Gomex & Quesada, n.d.).

For this paper, the application on focus is the use of genetic algorithm in a space invader game called CiTIUS Invaders. The goal of the game is to shoot the invaders to keep them in the range of 4 to 100 as they keep on breeding over time.

Figure 1. CiTIUS Invaders game

Each space invader has genes or attributes which are speed, probability, size, and colour. After every 5 seconds, there is evolution time in which the invaders undergo mating to create new invaders that will inherit the genes of their parents. The higher the fitness of an invader, the higher the probability of getting selected for mating. The fitness of an invader is the number of evolutions that it has survived. Ultimately, this enables the invaders to improve themselves by learning which attributes have the best chances in beating the player. The essence of the game can be explained using a state diagram that depicts the training process using genetic algorithm.

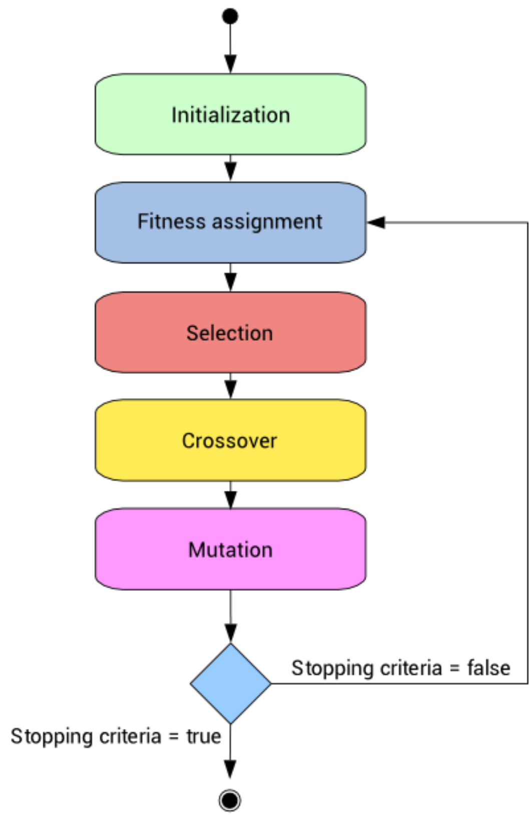

Figure 2. Training process using genetic algorithm

The training process begins with initialization of population which is the space invaders. For every space invader, there will be a predetermined fitness value. The training then progresses by selecting the best space invader by a measure of fitness. The best space invaders will then undergo crossover or mating. The new space invaders will then be mutated by multiplying with a Gaussian random distribution. As mentioned before, mutation of genes is required to produce a constant source of diversity. Even so, this does not mean that the inherited genes are the most optimal ones. The only thing known is the fact that the new invaders come from the fittest. By using mutation, it can open up the possibilities of discovering better genes that did not originate from the previously fittest invaders. The whole process will keep on repeating until a stopping criterion is met. In this case, it is when the number of invaders exceed 100.

References

Eiben, A., & Smith, J. (2015). *Introduction to Evolutionary Computing.* London: Springer.

Gomez, F., & Quesada, A. (n.d.). *Genetic algorithms for feature selection* . Retrieved from Neural Designer: https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection

Hemingway, B., Brunette, W., & Hamilton, T. (2005). *CSE 466 Software for Embedded Systems.* Retrieved from Paul G. Allen School of Computer Science & Engineering: https://courses.cs.washington.edu/courses/cse466/05sp/pdfs/lectures/10-EvolutionaryComputation.pdf

Hornby, G. S., Globus, A., Linden, D. S., & Lohn, J. D. (2006). *Automated Antenna Design with Evolutionary Algorithms.* American Institute of Aeronautics and Astronautics.

Kendall, G. (27 March, 2018). *Evolutionary computation has been promising self-programming machines for 60 years – so where are they?* . Retrieved from The Conversation: https://theconversation.com/evolutionary-computation-has-been-promising-self-programming-machines-for-60-years-so-where-are-they-91872

Raval, S. (16 August, 2017). Retrieved from https://www.youtube.com/watch?v=rGWBo0JGf50

Risto Miikkulainen, S. (17 May, 2018). *Evolutionary computation will drive the future of creative AI*. Retrieved from Venture beat: https://venturebeat.com/2018/05/17/evolutionary-computation-will-drive-the-future-of-creative-ai/