Search Based Software Engineering

Exercise 03 - Representation and Particle Swarm

Team:

1. Sagar Nagaraj Simha - sagar.nagaraj.simha[at]uni-weimar.de
   Matrikel Nummer - 120797, Degree - Master's Computer Science for Digital Media
2. Mohd Saif Khan - mohd.saif.khan[at]uni-weimar.de
   Matrikel Nummer - 120803, Degree - Master's Human Computer Interaction

Task 1. Representation

a) What is the difference between genotype and phenotype? Explain in your own words. Give an Example.

In Biology:

The genotype is the set of genes (a component of genotype) in our DNA which is responsible for a particular trait. The phenotype is the physical expression, or characteristics, of that trait (sometimes combined with the environmental influence) which are observable. Example, If a 'white mouse' had genes responsible for color to be inactive in its genotype. Then, that genotype would be responsible to express its phenotype (the white color) [3][4]

Adapting into the world of computation:

Genotype is the data itself, along with its structure, i.e the encoding used during breeding. The phenotype is the encoded data's expression or manifestation or behaviour during fitness assessment. A genotype of the type of a fixed length vector is called a 'chromosome'. And a particular slot position in the chromosome is called the gene. [2][5]

- In the case of Particle Swarm Optimization in Ex-3, the Genotype of a particle is the geometric point in the frame, i.e x and y coordinates encoded as an array. And, the phenotype is the visualization or how it moves/flows in the frame/space along with other particles towards the mouse pointer (global optimum).
- In the example of "Highly Configurable Software Systems" in Ex-2, the Genotype is a configuration array (binary string), [1,0,1,0,0,1,0,0,0,1,0,1,0,0,0,0,1] where each 'gene' (each slot/bit) in the genotype encoded whether a particular software feature is enabled or not. And the phenotype was the effect of that genotype (binary string) i.e the performance of the software system.

Other Examples:

Genotypes	Corresponding Phenotypes ("An Expression")
Array of vectors in a path choosing algorithm	Movement of a robot

A Matrix or array of matrices of choices	Movement/state of white & black pieces in Chess
A sequence of characters	Pronunciation of a word
Pixels in an image	Image visualization/rendering
Frequencies and amplitudes in audio	Sound of an audio
Trees / Graphs	Information flow in social networks

[5]

b) Explain the Hamming Cliff in your own words. How can it be prevented? Give an example.

One of the crucial beliefs in population-based methods is that individuals which are similar to each other tend to behave similarly. The choice of encoding (genotype) of individuals needs to accommodate this belief so that crossover/mutation of an individual picks the neighbouring individuals (phenotypes) only.

Example, **01111 and 10000** are representations (genotypes) of 15 and 16 (phenotypes), respectively. They have a Hamming distance of 5 (Hamming Distance is the minimum number of substitutions required to change one binary string into the other). For the mutation/crossover operation to change from 15 to 16, it must tweak all the 5 bits at one step, which is usually not possible. This presents as a Hamming Cliff and it is hard to overcome. 'Hamming Cliffs' are a major disadvantage in using binary strings as the genotype since the number of substitutions to change from one phenotype to another neighbouring phenotype may be large (i.e large Hamming Distance between neighbouring phenotypes). [6]

This can be addressed by using Gray Codes as genotypes instead of the binary strings where each successive string differs by only one bit. Hence, the above example with phenotypes 15 and 16 are now represented by the equivalent gray codes **01000 and 11000** respectively. Thus, if we are at 15, we can crossover to 16 with just 1 bit flip, overcoming the Hamming Cliff problem.

Task 2. Random Walk Mutation

a) Implement a Python function for the Random Walk Mutation. Test your function with a population of your choice.

→ random_walk.py is inside the folder

Task 3. Particle Swarm Optimization

a) Implement the Particle Swarm Optimization in Python for the given framework.

→ Present in the folder [9]

References:

[1] Lecture Slides

[2] Essentials of Metaheuristics - Sean Luke
[3] https://evolution.berkeley.edu/evolibrary/article/genovspheno_01
[4] https://en.wikipedia.org/wiki/Genotype%E2%80%93phenotype_distinction
[5] https://youtu.be/_of6UVV4HGo
[6] http://dilvanlab.github.io/old/dilvan/thesis.phd/genetic.html
[7] https://en.wikipedia.org/wiki/Hamming_distance
[8] https://en.wikipedia.org/wiki/Gray_code
[9] Jupyter Notebook - Lecture Resources