CT5141 Lab Week 6

James McDermott

2D Cars

1. Suppose that a 2D car is defined by these parameters (notice types and ranges):

- Shape (8 floats in [0, 1], 1 per chassis vertex)
- Wheel size (2 floats in [0, 1], 1 per wheel)
- Wheel position (2 ints in [0, 7], 1 per wheel)
- Wheel density (2 floats in [0, 1], 1 per wheel)
- Chassis density (1 float in [0, 1])

How could we define these operators on 2D car genotypes? (No need to write code for this.)

- Initialisation
- Mutation
- Crossover

2. Is there linkage/epistasis in the 2D car problem?

Genetic Algorithm: Rastrigin

3. Get `ga.py` from Blackboard. Notice that all children are created by crossover, but only a fraction of them (`pmut`) are then mutated also. Run it with the built-in `rastrigin` function. Does it reach the global optimum? Do you get the same result every time? Try out different values of the tournament size (`tsize`) and mutation probability (`pmut`) hyperparameters.
4. Do you ever observe that the best individual *disimproves*? Add *elitism* to prevent this.
5. The GA has `uniform_crossover`. Try implementing a different crossover. Check that it works as expected before using it in the GA.
6. At each generation, output the following statistics on the objective function: min, mean, median, max, standard deviation. Plot the min (i.e. best) and standard deviation. How long does it take for the population to *converge*?

Feature Selection with LAHC and GA

Feature selection is the problem, in machine learning, of choosing which features we should use. Given a set of n features, this is a bitstring optimisation function.

7. What is the objective function? (No need to write code here.)

8. Get `feature_selection.py` from Blackboard. It contains an implementation of the objective for an example problem, the Boston Housing regression problem. Notice that we use `train_test_split(X, y, random_state=0)` so that we get the same train-test split every time. (This doesn't affect the randomness elsewhere in the algorithm.) There are 13 features in this problem. How large is the search space? Could we use enumerative search?

9. Run a standard bitstring LAHC using the objective given in 8, with fitness evaluation budget of 400 and $L = 40$. But first, make sure that you are not trying to maximise when you should be minimising. Check LAHC and the objective function.

10. Observe that the objective `C` never disimproves. But we know that LAHC allows disimproving moves. Explain this by looking at the LAHC code.

11. Try running a binary GA on this problem.