COMS30014 - AI

Genetic Algorithm Lab 2: Coevolution

0. Prep

=======

Before the lab session, please download the python GA code for this lab (ga-code-lab2.py) and get it running on your computer, and please read sections 1 through 4 of the Cartlidge and Bullock (2004) paper (see below; or see the Blackboard page for this lab).

- The code is written in Python 3.
- It does not use any packages apart from "random" and "statistics" (which are standard Python packages).
- I've tried to write code that is easy to read, rather than efficient, compact or even overly "Pythonic"
- (If you want to adapt it to run faster, etc., that's fine as long as you understand it first)
- If your code won't run, or something in the code doesn't make sense, raise a question on the forum

The code implements a very simple coevolutionary genetic algorithm operating on a very simple example problem (a version of 'one-max': where the fitness of a solution bit-string is determined by the number of '1's it contains), and allows us to explore the influence of "parasite virulence" on the performance of a coevolutionary GA that is suffering from coevolutionary disengagement.

- This model was first introduced in a paper: Cartlidge, J. & Bullock, S. (2004). Combating coevolutionary disengagement by reducing parasite virulence. Evolutionary Computation, 12(2), 193-222. It's available from the unit Blackboard page and also from: https://seis.bristol.ac.uk/~sb15704/pubs.html
- The paper explores a thought experiment on how coevolutionary search can be improved by introducing an idea from biology: moderate virulence parasites. Before the lab, please read sections 1 through 4 (15 pages).
- We are going to use the code as a way of introducing the basic form of a coevolutionary genetic algorithm .. and as a way of thinking about slightly more complicated coevolutionary genetic algorithm ideas

- if you complete this lab and the previous GA lab your ability to answer exam questions or complete the unit coursework will be improved

Once you have the code downloaded and running OK, feel free to read the following ahead of the lab. But please don't start the lab exercises until you're in the lab with the other students. :)

The code implements a very simple coevolutionary genetic algorithm hereafter referred to as 'the standard coevolutionary GA set up':

```
1. *Initialise* a population of N hosts and a population of N parasites
2. *Assess* each host vs n randomly chosen parasites (and vice versa)
3. Repeat: until we complete G generations of evolution
4.    *Breed* N new offspring hosts from the fitter hosts
5.    *Breed* N new offspring parasites from the fitter parasites
6.    *Mutate* the new offspring hosts and the new offspring parasites
7.    *Assess* each host vs n randomly chosen parasites (and vice versa)
8.    Every g generations: *Write out* data on the hosts and parasites
```

Every individual in each population is a fixed-length bit-string. The fitness of a host is related to the number of parasites that it defeats during n competitions with randomly chosen parasites. The fitness of a parasite is related to how many of these competitions it wins. When two individuals compete, the one with most "1"s in their bit-string wins (ties are broken in favour of hosts). In this toy problem, we are interested in seeing how well coevolution can generate hosts with as many "1"s as possible.

To complete the lab, you need to be able to run the code (or your own version of the code) and adapt it in some relatively simple ways to see what happens.

As with Lab #1, the ability to take data from your GA and quickly plot it nicely will be very useful.