

COMSM0302: Evolutionary Computing

James Marshall

Welcome

- ▶ Welcome to COMSM0302: Evolutionary Computing!
- ▶ What is Evolutionary Computing?
- ▶ Evolutionary Computing (EC) is a collection of 'nature-inspired' heuristic optimisation techniques for 'hard' problems, e.g.
 - ▶ Genetic Algorithms (GA)
 - ▶ Genetic Programming (GP)
 - ▶ Evolutionary Strategies (ES)
 - ▶ ...
- ▶ Compare with traditional exact optimisation approaches for sufficiently 'simple' problems
 - ▶ Linear Programming
 - ▶ ...
- ▶ ...and other heuristic algorithms
 - ▶ Simulated Annealing
 - ▶ ...

Syllabus

- ▶ Implementation
 - ▶ Genetic Algorithms
 - ▶ Genetic Programming
 - ▶ Other EC algorithms
 - ▶ Applications in optimisation and modelling
- ▶ Theory
 - ▶ Descriptive models
 - ▶ Predicting behaviour
 - ▶ Fundamental limits on search and optimisation

Curriculum

- ▶ 20 hours of lectures, weeks 1-10
- ▶ Office hour
- ▶ Notes
 - ▶ Will be posted on the unit webpage before each lecture
 - ▶ Supplementary material, worked exercises, etc. will be presented during lectures... all is useful and examinable, *take notes!*
- ▶ 50% coursework, 50% exam
 - ▶ Genetic Algorithm assignment (30%, deadline early December)
 - ▶ Genetic Programming assignment (20%, deadline early December)
 - ▶ Theory exam (50%, January)
- ▶ Additional unassessed self-learning exercises (mixture of implementation and theory)
 - ▶ I am available to answer questions during my office hour
 - ▶ For routine programming enquiries, use the help-desk (MVB 3.19)
 - ▶ Answers provided after 1 week
 - ▶ Use these to prepare for the coursework and exam

Curriculum

► Lectures

<i>Week</i>	<i>Topic</i>
1	Introduction / Simple Genetic Algorithm
1	GA: Representations & Operators
2	GA: Populations & Selection
2	GA: Advanced Operators & Techniques
3	GA for Grouping Problems
3	Overview of Genetic Programming
4	GP: Case Studies
4	GP: Case Studies
5	GP: Case Studies
5	No Free Lunch Theorems

Curriculum

► Lectures

<i>Week</i>	<i>Topic</i>
6	Schema Theory
6	GAs as Markov Processes
7	Dynamical Systems GA Model
7	Statistical Mechanics Approximation of GA
8	Predicting GA Performance
8	Fitness Landscapes
9	Memetic Algorithms
9	Evolution Strategies
10	Estimation of Distribution Algorithms
10	Artificial Life & Modelling

Course Texts

- ▶ Recommended texts
 - ▶ Reeves, C. R. and Rowe, J. E. Genetic Algorithms - Principles and Perspectives: A Guide to GA Theory. 2003.
 - ▶ Falkenauer, E. Genetic Algorithms and Grouping Problems. 1998.
- ▶ Background texts
 - ▶ Holland, J. H. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. 1975 (2nd ed. 1992).
 - ▶ Goldberg, D. E. Genetic Algorithms in Search, Optimization and Machine Learning. 1989.
 - ▶ Mitchell, M. An Introduction to Genetic Algorithms. 1998.
- ▶ All texts are in the library
 - ▶ Errata sheet for Reeves and Rowe on unit homepage

What is Evolutionary Computing?

- ▶ Evolutionary Computing is a collection of 'nature-inspired' heuristic optimisation techniques, e.g.
 - ▶ Genetic Algorithms
 - ▶ Genetic Programming
 - ▶ Evolutionary Strategies
 - ▶ ...
- ▶ Mostly inspired by 'neo-Darwinian' evolutionary theory...

Darwin, Mendel and the Modern Synthesis

- ▶ 1859 - publication of the 'Origin of Species' by one Charles Darwin
 - ▶ Explained evolution as due to natural selection acting on heritable variation
 - ▶ Very similar ideas proposed by A. R. Wallace at about the same time
- ▶ 1865 - Gregor Mendel presents his work on 'Experiments in Plant Hybridisation'
 - ▶ Demonstrated the particulate nature of inheritance (Darwin had assumed it was blending)
- ▶ These two achievements together paved the way for the 'Modern Synthesis'...

The Modern Synthesis (aka neo-Darwinism)

- ▶ One problem that had worried Darwin was 'regression' of traits
 - ▶ Darwin assumed blending of heritable material
 - ▶ Hence the 'value' of any trait would tend to converge in a population as evolution progressed
 - ▶ The resulting lack of variation in the population would give natural selection nothing to act on, so evolution would stall
- ▶ Mendel's results went un-noticed by evolutionists for about 30 years

The Modern Synthesis (aka neo-Darwinism)

- ▶ Once rediscovered, Mendel's laws of particulate (i.e. genetic) inheritance were incorporated into Darwinian evolutionary theory
- ▶ The result was the 'modern synthetic theory of evolution', or 'neo-Darwinism'
 - ▶ Emphasises natural selection acting on genetic variation in populations
 - ▶ Primarily mathematical in nature, modelling gene spread in populations
 - ▶ Population genetics
 - ▶ Quantitative genetics
- ▶ Subsequently, the physical basis of particulate inheritance was found
 - ▶ Discovery of DNA by James Watson and Francis Crick, 1962 Nobel Laureates in Medicine

The Emergence of Evolutionary Computing

- ▶ Evolutionary ideas were being applied in optimisation as early as the mid 20th century, e.g.
 - ▶ Box (1957) Evolutionary operation: a method for increasing industrial productivity. Applied Statistics 6, 81-101
 - ▶ Bremermann (1962) Optimization through evolution and recombination. In: Self-Organizing Systems.
- ▶ In the 1970's Genetic Algorithms became a prominent research area
 - ▶ John Holland presented a mathematical definition of GAs, and theory explaining their performance
 - ▶ Holland (1975) Adaptation in Natural and Artificial Systems
 - ▶ Holland was mainly interested in adaptive systems, but his student Ken De Jong catalysed research on for GAs for optimisation, formulating a test suite of problems
 - ▶ De Jong (1975) An Analysis of the Behavior of a Class of Genetic Adaptive Systems. PhD thesis, University of Michigan

The Simple Genetic Algorithm

The Simple Genetic Algorithm

- ▶ What is the the 'Simple Genetic Algorithm'?
- ▶ Genetic algorithms have many variations on a central theme
 - ▶ Choice of selection operator
 - ▶ Choice of genetic operators
 - ▶ ...
- ▶ The SGA is the canonical description of what a GA is
 - ▶ Here we will describe the SGA according to Holland's original proposals
- ▶ The SGA is also used to refer to Michael Vose's mathematical description of a GA
 - ▶ Vose (1999) The Simple Genetic Algorithm: Foundation and Theory

Terminology

- ▶ Before we introduce the SGA, we must introduce some terminology
 - ▶ *Chromosome* - a solution to the problem the GA is solving, encoded as a fixed length string in some finite alphabet
 - ▶ *Gene* - a position on the chromosome which can take a particular value
 - ▶ *Locus* - see *gene*
 - ▶ *Allele* - a value that a gene can take (e.g. 0 or 1 in binary chromosomes)
 - ▶ *Population* - the set of chromosomes the GA is acting on
 - ▶ *Fitness* - the objective value of the solution a chromosome encodes for a problem
 - ▶ *Fitness/objective function* - the function that returns the objective value of a given solution
 - ▶ *Selection* - the mechanism for choosing chromosomes to reproduce, based on their fitness
 - ▶ *Parent* - a chromosome selected for reproduction
 - ▶ *Offspring* - the chromosome resulting from reproduction of one or more Parents
 - ▶ *Crossover* - the process of recombining alleles from multiple parents during reproduction to produce offspring
 - ▶ *Mutation* - the process of randomly replacing offsprings' alleles with randomly chosen alternatives during reproduction

The Simple Genetic Algorithm

```
generate initial chromosome population;  
while termination criterion not met do  
  evaluate population fitness;  
  while insufficient offspring created do  
    select parents by fitness;  
    if crossover condition satisfied then  
      perform crossover;  
    end  
    if mutation condition satisfied then  
      select chromosome for mutation;  
      perform mutation;  
    end  
    add offspring to population;  
  end  
  select new population;  
end
```

Selection

- ▶ A key idea of the GA is that parents should be selected based on their fitness
- ▶ The simplest way of doing this is using *roulette wheel selection*
 - ▶ Each chromosome i in the population P is assigned a probability of selection p_i based on its fitness f_i as a proportion of total population fitness:

$$p_i = \frac{f_i}{\sum_{j \in P} f_j}$$

- ▶ So for a given population we construct a single-pointer 'roulette wheel' where each chromosome's proportion of the wheel is given by the equation above
- ▶ To select a parent for reproduction we uniformly sample from $[0, 1)$
 - ▶ *i.e.* we spin the roulette wheel once and see which chromosome its pointer indicates

Selection

- ▶ Having selected a parent we then reproduce that parent according to our operators, parameters, etc.
 - ▶ According to our parameters we may apply crossover, and/or mutation, and/or any other operators
 - ▶ Often crossover rate (probability of applying crossover during reproduction) is referred to as χ
 - ▶ Similarly mutation rate (per gene probability of mutation during reproduction) by μ
- ▶ Usually population size is held constant...
- ▶ ...we continue selecting and reproducing until we have enough offspring to replace the current generation
 - ▶ Hence such a GA is called a 'generational' GA

Representation

- ▶ The simplest representations use a binary alphabet
- ▶ In fact, as we'll see later, Holland argued for the *optimality* of binary encoding
- ▶ We shall assume binary chromosomes for the Simple GA

Single-Point Crossover

- ▶ Holland's initial proposal for genetic recombination in GAs was single-point crossover (1X)
- ▶ For a pair of parent chromosomes of length ℓ , we (uniformly) choose at random an integer from $\{1, 2, \dots, \ell - 1\}$
- ▶ We then exchange portions of each chromosome, combining the substring left of the cutpoint from one chromosome with the substring right of the cutpoint from the other chromosome, and vice versa
 - ▶ e.g. the parent chromosomes 11111 and 00000 with the randomly selected crossover point at position 3...
 - ▶ ...yield the offspring chromosomes 11100 and 00011
- ▶ We now have two offspring chromosomes, each containing alleles from both parent chromosomes
- ▶ Optionally we can discard one of the offspring if we only require one offspring

Mutation

- ▶ During reproduction mutation occurs at a (usually low) rate, in addition to crossover
- ▶ Normally mutation rate is expressed as a per-gene probability
- ▶ If a gene is mutated, its current allele is exchanged for a different randomly selected allele
 - ▶ If our chromosomes are binary strings, there is only one other possible allele and the gene is simply bit-flipped
- ▶ A typical mutation rate might be on the order of 0.01 probability of mutation per gene 'transcribed'

Recap - The Simple GA

- ▶ The Simple GA

generate initial population;

while *termination criterion unsatisfied* **do**

 evaluate population;

 select parents;

 reproduce;

 replace population with offspring;

end

- ▶ Genetic operators

- ▶ Single-point crossover (1X) - recombines alleles from two parents into offspring
- ▶ Mutation - randomly changes alleles in offspring at a very low rate

A Simple Example - Unitation

- ▶ Let us work through a simple example of the SGA solving a problem
 - ▶ Problem - *Onemax* (binary fixed-length chromosome, fitness is number of ones in chromosome; an example of a function of *unitation*)
 - ▶ N.B. *Onemax* is *not* a good problem to apply a GA to as the fitness function in *linearly separable* (e.g. hill-climbing will do much better), but it is often used for illustration
 - ▶ Chromosome length (ℓ) = 5
 - ▶ Population size (N) = 4
 - ▶ Crossover rate (1X) (χ) = 1.0
 - ▶ Mutation rate (μ) = 0.5
- ▶ *Exercise*: Implement the SGA on the *Onemax* function in a language of your choice