# Off-the-shelf Genetic Algorithm Optimiser

Peter Seymour

February 9, 2020

## 1 Introduction

This papers details an algorithm to solve the optimisation problem of maximising:

$$f(x_1, \dots, x_n)$$

whithin a given hypercube:

$$[x_1^{min}, x_1^{max}] \times \cdots \times [x_n^{min}, x_n^{max}]$$

It uses the universal GA algorithm using real coding, roulette selction, one-point crossover and uniform mutation.

## 2 Coding

Each individual in any population is represented by a chromosone which in turn is an encoded vector from the problem space. The encoding maps real valued variables to discrete genotypes by:

$$encode(x) = \frac{x - x^{min}}{x^{max} - x^{min}} \cdot (2^{32} - 1)$$

The chromosone is then a $32n$-bit binary string concatenated from the genotypes:

$$encode(x_1)| \dots |encode(x_n)$$

To decode a chromsone split it into its $n$ genotypes and perform the reverse mapping:

$$decode(g) = x^{min} + \frac{g \cdot (x^{max} - x^{min})}{2^{32} - 1}$$

# 3 Main Algorithm

Let $(c_{1,t}, \ldots, c_{m,t})$ be the population of chromosomes at time (generation) $t$. The population size, $m$, is fixed over the entire algorithm. The algorithm inductively constructs the population at time $t+1$ from that at time $t$.

1 Create initial population of individuals for $t = 0$

2 If $t = t_{max}$ goto Step 7

3 Select individuals for reproduction

4 Perform crossover to create next generation

5 Mutate some genes

6 $t \to t+1$ goto Step 2

7 Select fittest from all generations

## 3.1 Initial Population Creation

Each $c_{i,0}$ is a uniformly distributed sample from the whole space.

## 3.2 Selection

Selection proceeds by evaluating the fitness of each chromosome in the population. This is done by decoding it and then evaluating under $f$ to form $f_i = f(c_i)$. Each $f_i$ will be in the range $(-\infty, \infty)$. A selection strategy is used to convert the fitness values to selection probabilities. The population is sampled with replacement $m$ times according to these probabilities.

### 3.2.1 Roulette Wheel Selection With Maximum Fitness Scaling

The selection strategy employed is a variant of standard roulette wheel selection to handle negative fitness values. First all the fitness values are translated by the mean to make the mean 0. A constant scaling factor, $\alpha$, is applied to each value such that all the values lie in the range $[-1, \lambda]$. Where $\lambda$ is a maximum scaling factor parameter and at least one of fitness values equals a bound. These are translated back to being non-negative by adding 1. The probability of selection, $p_i$, is then proportional to the total of the scaled fitness values. This occurs when:

$$\mu = \frac{\sum f_i}{n}$$

$$f_{min} = min(f_1, \ldots, f_n)$$

$$f_{max} = max(f_1, \ldots, f_n)$$

$$\alpha = min(\frac{-1}{f_{min} - \mu}, \frac{\lambda - 1}{f_{max} - \mu})$$
$$f'_i = \alpha \cdot (f_i - \mu) + 1$$
$$p_i = \frac{f'_i}{\sum f'_i}$$

# 4 Crossover

Crossover takes consecutive pairs of chromosomes, $(c_{2i}, c_{2i+1})$ called the parents, and produces two new chromosomes by randomly splicing pieces from the parents.

## 4.1 k-point Crossover

The simplest of all crossovers. The $k$ crossover points are taken by sampling with replacement from $\{1, \ldots, 31\}$ uniformly. Both parents are split at these points to yield $k+1$ substrings. The two children are formed by swapping every other substring in the parents.

# 5 Mutation

Each new chromosome under goes a mutation process which may result in no change.

## 5.1 Uniform Mutation With Intesity $\mu$

The simplest mutation for a binary string. Each gene (bit) is flipped with a probability $\mu$, the intensity.

# 6 Defaults

The following are a "reasonable" set of initial values that should work well in many instances.

- $m = 50$ for the population size.
- $t_{max} = 100$ for the number of generations.
- $\lambda = 10$ for selection scaling.
- $k = 1$ to achieve 1-point crossover.
- $\mu = \frac{1}{32n}$ for the mutation intesity.

The algorithmic complexity is roughly $O(m \cdot ln(m) \cdot t_{max})$. This arises since roulette selection requires the fitness values to be sorted and each generation is only dependent on the previous. Selection is the most time intensive calculation so increases in $t_max$ should be preferred over increases in $m$. The choice of $lamdba$ indicates a cap on the level of elitism (superior individuals dominating). The crossover is chosen to be the simplest.

It has been suggested in [1] that the mutation rate be $\frac{ln(\sigma)}{L}$. Where $L$ is the length of the chromosome. The selection pressure, $\sigma$, is a measure of the maximum fitness over the mean used in selection. Since the selection limits $\sigma$ to $\lambda$ a mutation rate of less that $\frac{ln(10)}{32n}$ should be used. This is roughly $\frac{2}{32n}$ but caution indicates it is better to be too low as populations can sink into chaos with too much mutation.

# References

[1] "Optimal Mutation Rates and Selection Pressure in Genetic Algorithms, Gabriela Ochoa, Inman Harveyand and Hilary Buxton.