Nature Inspired Search and Optimisation

Exercise 1

1429527

October 30, 2016

The assignment is to design an algorithm to solve the equation

$$f(x) = y \tag{1}$$

1 Describe a suitable chromosome representation of an individual

I considered three options for representation: Binary, Gray coding and real-valued. The value for x in the given function 1 has not had boundaries defined so I am assuming that it will be on a continuous scale of floating point numbers.

Binary can up to double the real value with a single mutation which, though possibly advantageous during the initial search stages, may cause massive variations while fine tuning final stages of find local minima. Due to these Hamming Cliffs, binary coding would not be a suitable for this function.

Gray coding would overcome this problem as adjacent integers can be reached by a single bit-flip; This means that, using Gray coding, changing a single bit causes a small change in the value of the real number it represents.

Despite this, Gray coding is the most difficult for a human to understand as real numbers and binary are more common place. Both binary representations also have the flaw of string size as, for a floating point number, the string length required to maintain accuracy grows exponentially. For example:

Given domain [2,2] and precision of 6 decimal places

Divide domain [2,2] into 4 1000000 intervals

We need 22 bits (4 * 1000000 $< 2^{22}$) to represent this in binary

Whereas for real values, we would only need 7 digits.

For these reasons I have decided to use real-valued representation. This will cause the evolutionary operators to be more complex but will be easier for humans to follow while maintaining small mutations and conserve memory. [1] confirms my decision and explains that the genetic operators will consequently randomly choose a floating-point number in a range rather than flip a bit of the parent during mutation.

2 Design a suitable fitness function

A fitness function should maximise or minimise the fitness of the best solution. Because of this, the mathematical result of the fitness function should be best when f(x) is closest to y.

$$fitness = |y - f(x)| \tag{2}$$

2 will be my fitness function as the further from y f(x) is, the worse the fitness is. This will require my selection process to consist of a minimisation function where a fitness value of 0 is a correct solution for x.

y = 4				
f(x) =	4	7	1	-5
|y-f(x)|	0	3	3	9

Example of Fitness Function

There are other considerations to take into account, as described in[2], however we are not given any details of constrains or boundaries for the function, so they cannot be applied.

3 Describe what evolutionary operators you would use

The assignment states that we are to find all vectors x that satisfy equation 1; This means that my evolutionary algorithm can find multiple local minimum. For this I need an evolutionary operator that can both explore the search space and converge on local minimum during later iterations where exploitation is important.

As I would use real-valued representation, I require an operator that selects an offspring from a parent with a probability distribution. From the paper [3], their FEP is typically better than or equal to CEP depending on the function that it is assessing; Because of this, and the lack of knowledge about 1, I will produce two children per parent: one with Gaussian and the other with Cauchy mutation throughout my algorithm.

This will get the balance of explore and exploit as Cauchy mutation is superior for exploration with its infinite range but lower probability at the mean. Gaussian has a finite range and a higher probability at its mean so is better for probability. Producing a child in both mutations styles will give the benefits of both.

Paper [4] looks at this IFEP algorithm against their own quadratic approximate mutation algorithm. IFEP produces better results for multi-modal functions with many local minima which is what I am assuming 1 will be.

To create the two offspring (X_i,η_i) from a single parent (X_i,η_i):

For j = 1,...,n

$$\eta^{'}_{ij} = \eta_i j exp(\tau^{'} N(0,1) + \tau N_j(0,1)) \tag{3}$$

$$X^{'}_{ij} = X_{ij} + \eta_{ij} N_j(0,1) \tag{4}$$

$$X^{''}_{ij}0 = X_{ij} + \eta_{ij} C_j(1) \tag{5}$$

Where the factors are set to:

$$\tau = (\sqrt{2n})^{(} - 1) \tag{6}$$

$$\tau = (\sqrt{2\sqrt{n}})^{(} - 1) \tag{7}$$

4 Describe the selection scheme you would use

I would use tournament $(\mu + \lambda)$ - selection. This is where a population of μ parents, randomly selected by Linear Selection with a scaled fitness function of exponential scaling, produce λ children; The new populations of size μ is those with the best fitness function from the parents and the offspring.

λ is typically 2 [5] so I will use this in my algorithm; the best parent form the tournament will produce both offspring as described in **??**. The paper warns that this can cause monotonous evolution as there is a chance that no or little offspring are chosen to go to the next round. While this could increase the take-over time due to minimal change between generations, it will allow correct solutions to survive within the μ-elitism of the selection. As it is possible to find multiple correct solutions, this elitism is required to not lose solutions where the fitness = 0 i.e. the correct solution has been found.

I would also use Fitness Proportional Selection to assign a probability of selection to my parent population prior to selection for tournaments.

Fitness Proportional Selection:

$$\mathbf{Pr}[x] = \frac{(f(x)}{\sum_{y \in P} f(y)}. \tag{8}$$

This will be scaled by Exponential scaling as this includes a temperature that approaches 0 as time goes on. This means that, during exploration, parents with a lower fitness value have a reasonable chance of being selected compared to later in the search when exploitation will be more important.

Exponential Scaling:

$$f\ (x) := exp(f(x)/T) \tag{9}$$

where T > 0

5 Describe the experiments that you would carry out and the quantities that you would report to assess your solution. (You do not need to implement this but you need to describe your design.)

Based on articles [4] and [3], I have decided on the following attribute values for my function:

$$\text{n} = 30$$
$$\text{S} = [\text{-}600{,}600]$$
$$f_{min} = 0$$
$$Eval_{max} = 500000$$

Where n is the dimension of the function, f_{min} is the minimum value of the function and $S \subseteq R^n$ and the maximum number of iterations is the most required in a test containing Cauchy or Gaussian.

I have chosen these because; n is the most common for all the rest run in these papers; S is the largest range in the tests; the minimum value of my fitness function is 0. As no details are known about 1 going with this generalisation seems sensible.

I would then run the algorithm 50 times and calculate the mean, standard deviation and global hit rate across the evaluations. This is so that its effectiveness could be compared to other algorithms run on equation 1

References

[1] Traditional Techniques of Genetic Algorithms Applied to Floating-Point Chromosome Representations Leo Budin, Marin Golub, Andrea Budin

[2] V. Petridis, S. Kazarlis, and A. Bakirtzis,(1998) Varying fitness functions in genetic algorithm constrained optimization: The cutting stock and unit commitment problems, IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 28(5), pp. 629640. doi: 10.1109/3477.718514.

[3] X. Yao, Y. Liu and G. Lin Evolutionary programming made faster IEEE Transactions on Evolutionary Computation. 3(2):82-102, July 1999.

[4] K.-H. Liang, X. Yao and C. S. Newton Combining landscape approximation and local search in global optimization Proc. of the 1999 Congress on Evolutionary Computation. Vol. 2, IEEE Press, Piscataway, NJ, USA, pp.1514-1520, July 1999.

[5] Back, T. (1994). Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In Proceedings of the 1st IEEE Conf. on Evolutionary Computation, pages 5762. IEEE Press.