University of Isfahan

Department of artificial intelligence

Evolutionary Computing

Professor: Dr.Hossein Karshenas

Investigating and analyzing the performance of evolutionary search in different conditions to solve some benchmark problems that can be displayed in binary and numerical form

One of the challenges of using evolutionary algorithms in solving problems is the correct determination of the components and parameters of these algorithms. Obtaining a comprehensive understanding of how different options for components and parameters affect algorithm performance can be useful in using these algorithms as well as possible. The purpose of this exercise is to investigate and analyze the performance of evolutionary search in different conditions to solve some benchmark problems that can be represented as binary and integer. The framework of the evolutionary algorithm used in this exercise is introduced in section 1, fitness functions. Benchmark problems with binary representation and the desired conditions for algorithm review are specified in section 2. The problem of password guessing which is displayed numerically and the various fitness functions related to it are introduced in section 3. The items that must be delivered are also in section 4.

1. The evolutionary algorithm framework used

The main steps of the evolutionary algorithm are shown in the pseudo code below:

Algorithm: EvolSearchExercise	
1	*pop* = **Generate** *popSize* initial candidate solutions of size *problemSize*
2	*popFit* = Evaluate *pop* using **f(x)**
3	While not **Teriminate()**
4	*parentsPool* = **Select** *popSize* solutions from *pop* using *popFit*
5	*parentPairs* = Shuffle *parentsPool* and randomly pair solutions
6	*offspr* = Perform **Recombination** on *parentPairs* with P_c
7	*offspr* = Perform **Mutation** on *offspr* with P_m
8	*offsprFit* = Evaluate *offspr* using **f(x)**
9	[*pop*, *popFit*] = Best *popSize* solutions from joint *pop + offspr using popFit and offsprFit*
10	Return best solution in *pop*

In the first step, a popSize population is randomly generated from solutions with a problemSize length. Different values for these two parameters should be considered, which will be explained in section 2 for binary representation and in section 3 for numerical representation. In the second step, population solutions are evaluated using the fitness function related to the desired benchmark problem. , and after that, in the loop specified in lines 3 to 9, the main process of evolution takes place. In this loop, in line 4, a sample is taken from the population using a selection method, and the parents are obtained. Different options are obtained. which should be considered for this component, which will be explained later in this section. After creating random pairs from the selected parents (line 5), the recombination operator is applied to each pair of parents with probability p and two children is produced. The different options that should be considered for the recombination operator are explained in the rest of this section. All offspring produced from all pairs, the parents, form the initial population of offspring. 6. Each offspring individually. Mutation is given to create the final population of children, line 7. For this, the value of each gene of each child changes with the probability Pm. Finally, after evaluating the new offspring produced in line 9, the best popSize individual from the common set of population individuals and new offspring are selected to be in the next generation population. are placed, then this list is sorted based on fitness values, and at the end popSize, the person at the top of the list, who has the highest fitness, will be selected. (1) Finding the optimal solution to the problem (2) The number of iterations of the loop

(number of generations) reaches 300. After the end of the evolution loop, the best solution obtained in line 10 is returned as the answer of the algorithm.

Question 1: Suppose two parents 1110001111 and 1000111100 are paired together in the binary representation in line 5 of the EvolSearchExercise algorithm, and in line 6 the combination is performed with a probability of 1.(Pc=1)

Answer following questions:

a. Assuming that Single-point Crossover is used in line 6, is the string 1010000111 can be one of the resulting children?

b. Assuming that Double-point Crossover is used in line 6, is the string 1110001111 can be one of the resulting children?

c. Assuming that Uniform Crossover is used in line 6, is the string 1010101100 can be one of the resulting children?

The main options considered for the selection method are as follows:

In the fitness proportional selection method, each member of the population can be selected with a probability, which is defined based on its fitness value as follows: ($\vec{x}_i$, the i-th person from the population):

$$P(\vec{x}_i) = \frac{f(\vec{x}_i)}{\sum_{j=1}^{popSize} f(\vec{x}_j)}$$

In the Binary Tournament selection method, two people from the population are randomly selected to participate in the competition and the fitter person is chosen as the winner of the competition. The number of competitions is equal to the number of people to be selected and the people who are Randomly selected to participate in one competition, they can also participate in other competitions.

The main options used in this exercise for recombination in binary representation are:

In Single-point Crossover, each parent is randomly divided into two parts (the division in both parents is completely the same) and then the parent parts are moved together so that the first child from the combination of parts The first part is created from the first parent and the second part from the second parent, and the second child is created vice versa.

In Uniform Crossover, each gene from the first parent is randomly given to one of the children, and the other child will receive the corresponding gene from the second parent, and this is repeated for all genes.

The mutation method in binary representation is Bit-flipping, in which the value of each bit of each child changes with probability Pm. The mutation method in numerical representation is Creep, in which each

gene of each child changes with probability Pm. A random value is added or subtracted. Consider the probability distribution of this random value to be uniform from -5 to 5.

2. Benchmark problems with binary representation

The evaluation function (f(x)) intended in this exercise is from the category of pseudo Boolean functions in the form of:

$$f: \{0,1\}^n \longmapsto \mathbb{R}$$

which is described in Table 1.

OneMax	$f(\vec{x}) = \sum_{i=1}^{problemSize} x_i$
Peak	$f(\vec{x}) = \prod_{i=1}^{problemSize} x_i$
FlipFlop	$f(\vec{x}) = \sum_{i=1}^{problemSize-1} (x_i \text{ XOR } x_{i+1})$
FourPeaks	$f(\vec{x}, T) = \max(tail(0,\vec{x}), head(1,\vec{x})) + R(\vec{x}, T)$ $tail(b,\vec{x}) = number\ of\ trailing\ b's\ in\ \vec{x}$ $head(b,\vec{x}) = number\ of\ leading\ b's\ in\ \vec{x}$ $R(\vec{x}, T) = \begin{cases} problemSize & if\ tail(0,\vec{x}) > T\ and\ head(1,\vec{x}) > T \\ 0 & otherwise \end{cases}$
SixPeaks	$f(\vec{x}, T) = \max(tail(0,\vec{x}), head(1,\vec{x})) + R(\vec{x}, T)$ $tail(b,\vec{x}) = number\ of\ trailing\ b's\ in\ \vec{x}$ $head(b,\vec{x}) = number\ of\ leading\ b's\ in\ \vec{x}$ $R(\vec{x}, T) = \begin{cases} problemSize & if\ (tail(0,\vec{x}) > T\ and\ head(1,\vec{x}) > T)\ or \\ & (tail(1,\vec{x}) > T\ and\ head(0,\vec{x}) > T) \\ 0 & otherwise \end{cases}$
Trap	$f(\vec{x}) = 3 \times problemSize \times \prod_{i=1}^{problemSize} x_i - \sum_{i=1}^{problemSize} x_i$

Table 1: The fitness functions of benchmark problems with binary representation

According to the domain considered for the functions, it is clear that the solutions of the problem are displayed in binary form (bit strings).

Question: What is the optimal solution(s) for each of the above fitness functions? What is the fitness value of it(s)? In the FourPeaks and SixPeaks functions, consider the T value equal to 2 and assume that the length of the string is at least 5. For example, in the One Max function, the optimal solution is the whole string 1 (11…1), because if even one bit is zero, Its sum is less than the entire string of 1, and as a result, according to the OneMax fitness function, its value will be less.

Question: In the FourPeaks and SixPeaks functions, if the value of the super-parameter T increases, how does the fitness of the optimal solution and the probability of reaching the optimal solution change? Specifically, changing the value of T from 1 to 4 in Check a problem with problemsize=10.

Solve each of the benchmark problems presented in Table 1 for each of the states in Table 2 using evolutionary algorithms. In each row of this table, one of the parameters is variable and the rest are fixed. Obtain and analyze the number of fitness function calls using appropriate graphs. The requirements are as follow: (In the FourPeaks and SixPeaks functions, the value of T is equal and Consider problemSize *0.1).

1- Check the effect of increasing the problem size (problemSize) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the first row of Table 2.

2- Check the effect of increasing the problem size (problemSize) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the second row of Table 2.

3- Check the effect of increasing the population size (popSize) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the third row of Table 2.

4- Check the effect of increasing the population size (popSize) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the fourth row of Table 2.

5- Check the effect of increasing the probability of recombination (Pc) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are as the fifth row of Table 2.

6- Check the effect of increasing the probability of recombination (Pc) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the sixth row of Table 2.

7- Check the effect of increasing the probability of mutation (Pm) on the best fit obtained from each of the benchmark problems in the case that the components and parameters are as the seventh row of Table 2.

8- Check the effect of increasing the probability of mutation (Pm) on the best fit obtained from each of the benchmark problems in the case where the components and parameters are in the eighth row of Table 2.

9 Check the effect of the type of selection method on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the form of the ninth row of Table 2.

10- The effect of the type of selection method on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the tenth row of Table 2.

11- Check the effect of the type of recombination operator on the best fit obtained from each of the benchmark problems in the case where the components and parameters are in the eleventh row of Table 2.

12- Check the effect of the type of recombination operator on the best fit obtained from each of the benchmark problems in the case that the components and parameters are in the form of the twelfth row of Table 2.

BT = Binary Tournament

FP = Fitness Proportionate

SP = Single Point

UC = Uniform Crossover

Recombination method	Selection Method	Mutation Probability	Recombination Probability	Population Size	Problem Size	
SP	BT	0.5	0.7	200	10-30-50-100	1
UC	BT	0.1	1	100	10-30-50-100	2
UC	BT	0.05	0.7	50-100-200-300	50	3
UC	FP	0.3	0.5	50-100-200-300	30	4
SP	BT	0.3	0.5-0.7-0.9-1	50	10	5
SP	FP	0.1	0.5-0.7-0.9-1	300	50	6
UC	FP	0.05-0.1-0.3-0.5	0.9	200	100	7
SP	FP	0.05-0.1-0.3-0.5	0.5	100	30	8
UC	BT-FP	0.05	1	50	10	9
SP	BT-FP	0.5	0.7	300	50	10
SP-UC	BT	0.1	1	300	100	11
SP-UC	FP	0.3	0.9	50	30	12

Table 2 : Options or different values of each of the components and parameters for the binary representation problem

3. Password guessing problem with numerical representation

Suppose we are looking to guess a password problemSize of a digit, each of its digits can be one of the numbers 0 to 9, and repetition of the digit is also allowed. Consider the following three fitness functions for this problem:

Fitness function f1: if all the digits entered are correct, the fitness value is 1, otherwise, even if one digit is entered incorrectly, the fitness value is zero. For example, if the main password is 1111777799, the fitness value of the string 1111777799 is equal to 1 and the fitness value of any other string is 0.

Fitness function f2: This function examines each digit of the password separately. If a digit is the same as the corresponding digit in the main password, it gets a score of 1, otherwise it gets a score. Finally, the fitness value of the entire string is obtained from the sum of the points of all its digits. For example, if the main password is 1111777799, the fitness value of the string 0123456789 is equal to 3 The fitness value of 8858077792 is equal to 4.

Fitness function f3 : This function examines each digit of the password separately. If a digit is the same as the corresponding digit in the original password, it gets a point, otherwise it gets a point equal to the minus of the distance between the estimated digit and the original digit. Finally, the merit value of the whole field is obtained from the sum of the points of all its figures. For example, if the main password is 1111777799, the fitness value of the string 123456789 is equal to -11, and the fitness value of the string 8858077792 is equal to 39.

$$\text{Suppose the password is } 1111777799 \Rightarrow f_3(0123456789) = -(|1-0| + |1-1| + |1-2| + |1-3| + |7-4| + |7-5| + |7-6| + |7-7| + |9-8| + |9-9|) = -11$$

Obviously, the optimal solution for all three functions above is the original password with a fitness value of 1 problemSize and zero. Solve each of the evaluation functions introduced in the password guessing problem for each of the states in Table 3 using evolutionary algorithms. In each row of this table, one of the parameters is variable and the rest are fixed. Analyze the effect of changing the variable parameters on the best fit obtained and the number of calls of the fitness function using suitable graphs.

1- Check the effect of increasing the size of the problem (problemSize) on the best fit obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are as in the first row of Table 3.
2- The effect of increasing the size of the problem (problemSize) on the best fitness obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are as the second row of Table 3.
3- Check the effect of increasing the population size (popSize) on the best fitness obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are as the third row of Table 3.
4- Check the effect of increasing the population size (popSize) on the best fit obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are as the fourth row of Table 3.
5- Check the effect of increasing the probability of recombination (Pc) on the best fit obtained from each of the fitness functions of the password guessing problem in the case where the components and parameters are in the form of the fifth row of Table 3.
6- Check the effect of increasing the probability of recombination (Pc) on the best fitness obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are as the sixth row of Table 3.
7- Check the effect of increasing the probability of mutation (Pm) on the best fit obtained from each of the fitness functions of the password guessing problem in the case where the components and parameters are as the seventh row of Table 3.
8- Check the estimate of the effect of increasing the probability of mutation (Pm) on the best fitness obtained from each of the fitness functions of the password problem in the case that the components and parameters are in the eighth row of Table 3.
9- Check the effect of the type of selection method on the best fit obtained from each of the fitness functions of the password guessing problem in the case where the components and parameters are as the ninth row of Table 3.

10- Check the effect of the type of selection method on the best fit obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are in the tenth row of Table 3.
11- Check the effect of the type of recombination operator on the best fit obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are in the eleventh line of Table 3.
12- Check the effect of the combination open operator type on the best fit obtained from each of the fitness functions of the password guessing problem in the case that the components and parameters are in the form of the twelfth line of Table 3.

BT = Binary Tournament

FP = Fitness Proportionate

SP = Single Point

UC = Uniform Crossover

Recombination method	Selection Method	Mutation Probability	Recombination Probability	Population Size	Problem Size	
SP	BT	0.5	0.7	100	5-7-10-15	1
UC	FP	0.1	1	200	5-7-10-15	2
UC	FP	0.05	0.5	50-100-200-300	5	3
SP	FP	0.3	1	50-100-200-300	15	4
UC	BT	0.5	0.5-0.7-0.9-1	200	10	5
UC	FP	0.3	0.5-0.7-0.9-1	50	5	6
UC	BT	0.05-0.1-0.3-0.5	0.9	200	7	7
SP	BT	0.05-0.1-0.3-0.5	0.7	300	15	8
SP	BT – FP	0.1	0.7	100	10	9
SP	BT - FP	0.05	1	50	7	10
SP-UC	FP	0.05	0.9	50	7	11
SP- UC	BT	0.1	0.5	300	5	12

Table 3 : Options or different values of each of the components and parameters for the password guessing problem

Question 4: Which of the fitness functions introduced in the password guessing problem of section (3) is similar to which of the fitness functions introduced in the binary representation of section (2) in terms of fitness perspective? Check this similarity closely.

Question 5: As you know, the evolutionary approach is one of the objective function optimization methods that goes through this process intelligently. What are the special features of each of the functions introduced in sections 2 and 3? Have the forces of selection and change in the evolutionary approach been successful in facing these special features in the optimization process?

Explain.

Attention: For each of the evaluation functions in sections 2 and 3 and each combination mentioned in tables 2 and 3, the algorithm must be tested at least 10 times independently and the average and standard deviation of the best fit are obtained, as well as the average and standard deviation to determine the number of fitness function calls, any analysis should be done with the average of at least 10 runs and not just with one run.