

CODE:

Editor - C:\Users\anugala\Documents\MATLAB\shiva.m

EDITOR PUBLISH VIEW

Save (Ctrl+S)

shiva.m

```

1 clear all;
2 %initiation
3 items = 10;
4 v = randi(8, [1, items]);
5 w = randi(9, [1, items]);
6 wmax = 10;
7 population_size = 10;
8 iteration_limit = 50;
9 population = round(rand([population_size, size(v,2)]));
10 %sampling chromosome
11 chromosome_before = population(1, :);
12 while iteration_limit > 0
13 %wmax if > population*w delete rand i from chromosome
14 for i = 1 : population_size
15 while sum(population(i, :) .* w) > wmax
16 population(i, randi(size(population, 2))) = 0;
17 end
18 end
19 %fitness
20 fitness = zeros(size(population, 1), 1);
21 for i = 1 : population_size
22 a = population(i, :) .* v;
23 fitness(i) = sum(a);
24 end
25 %choosing chromosome
26 [~, best_chr_idx] = max(fitness);
27 chr_idx_chosen = zeros(1, 2);
28 while chr_idx_chosen(1) == chr_idx_chosen(2) && ...
29 chr_idx_chosen(1) ~= best_chr_idx && chr_idx_chosen(2) ~= best_chr_idx
30 chr_idx_chosen = randi(size(population, 1), [1, 2]);

```

script Ln 12 Col 26

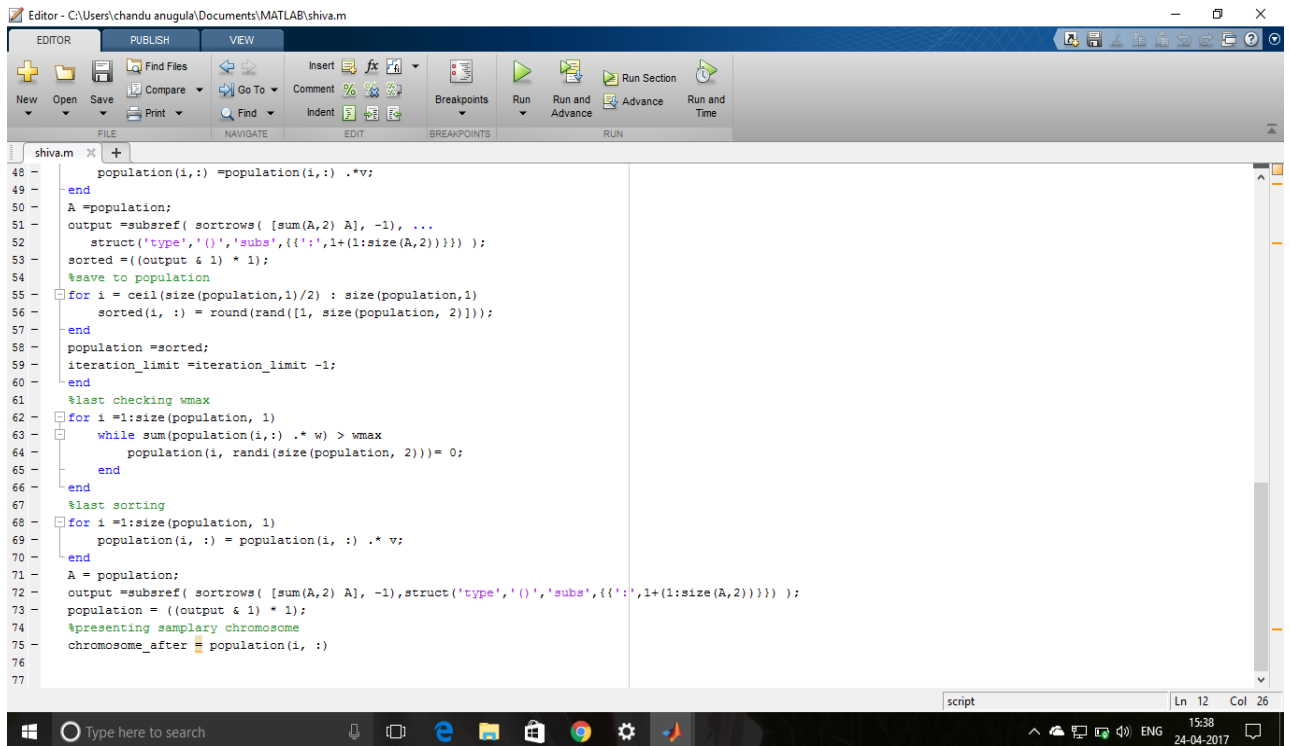
Type here to search

15:38 24-04-2017

The screenshot displays the MATLAB Editor interface. The main window shows a script named 'shiva.m' with the following code:

```
29 - chr_idx_chosen(1) ~= best_chr_idx && chr_idx_chosen(2) ~= best_chr_idx
30 - chr_idx_chosen = randi(size(population, 1), [1, 2]);
31 -
32 - %mask of values to exchange
33 - mask = zeros([1, size(population, 2)]);
34 - while sum(mask) < floor(round(size(population, 2)/2, 2))
35 -     ones_to_mask = zeros(1, size(population, 2));
36 -     ones_to_mask(randi(length(ones_to_mask))) = 1;
37 -     mask = (mask+ ones_to_mask) & 1 * 1;
38 - end
39 - %crossover
40 - a_part = mask .* population(chr_idx_chosen(1), :);
41 - b_part = mask .* population(chr_idx_chosen(2), :);
42 - population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) .* ~mask;
43 - population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) .* ~mask;
44 - population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) + b_part .* mask;
45 - population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) + a_part .* mask;
46 - %sorting
47 - for i = 1:size(population, 1)
48 -     population(i,:) = population(i,:) .* v;
49 - end
50 - A = population;
51 - output = subref( sortrows( [sum(A,2) A], -1), ...
52 -     struct('type','(',')','subs',{'{','}',1+(1:size(A,2))} ) );
53 - sorted = ((output & 1) * 1);
54 - %save to population
55 - for i = ceil(size(population,1)/2) : size(population,1)
56 -     sorted(i, :) = round(rand([1, size(population, 2)]));
57 - end
58 - population = sorted;
```

The interface includes a top menu bar with 'EDITOR', 'PUBLISH', and 'VIEW' tabs. Below the menu bar are various toolbars for file operations (New, Open, Save, Find Files, Compare, Go To, Find, Print), editing (Insert, Comment, Indent), breakpoints, and running the script (Run, Run and Advance, Run Section, Advance, Run and Time). The status bar at the bottom indicates the current line and column: 'Ln 12 Col 26'.



```

clear all;
%initiation
items = 10;
v = randi(8, [1, items]);
w = randi(9, [1, items]);
wmax = 10;
population_size = 10;
iteration_limit = 50;
population = round(rand([population_size, size(v,2)]));
%samplary chromosome
chromosome_before = population(1, :)
while iteration_limit > 0
    %wmax if > population*w delete rand 1 from chromosome
    for i = 1 : population_size
        while sum(population(i, :) .*w) > wmax
            population (i, randi(size(population, 2)))=0;
        end
    end
    %fitness
    fitness = zeros(size(population, 1), 1);
    for i = 1 : population_size
        a = population(i, :) .*v;
        fitness(i) = sum(a);
    end
    %choosing chromosome
    [~, best_chr_idx] = max(fitness);
    chr_idx_chosen = zeros(1, 2);
    while chr_idx_chosen(1) == chr_idx_chosen(2) && ...
        chr_idx_chosen(1) ~= best_chr_idx && chr_idx_chosen(2) ~=
        best_chr_idx
        chr_idx_chosen = randi(size(population, 1), [1, 2]);
    end
    %mask of values to exchange

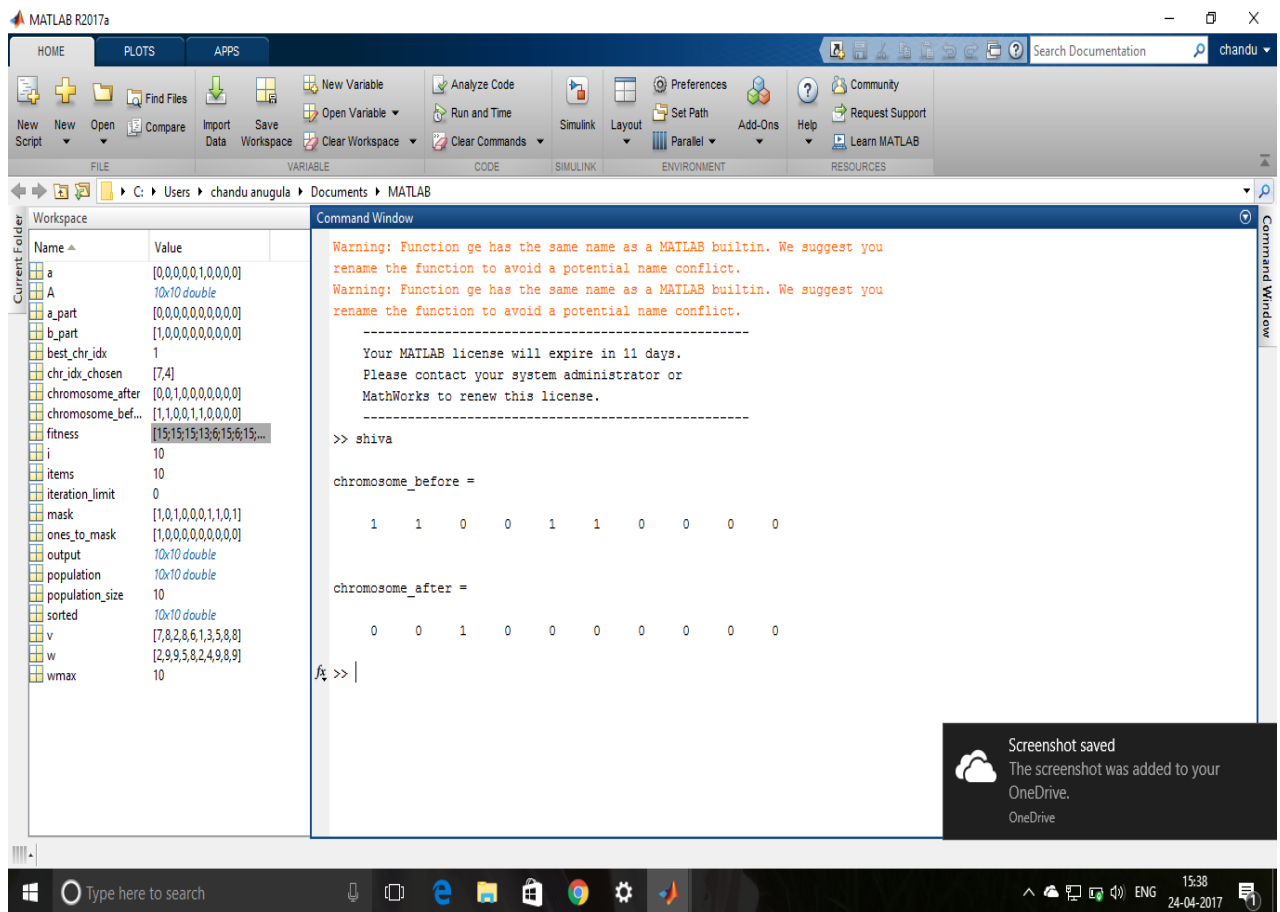
```

```

mask = zeros([1, size(population, 2)]);
while sum(mask) < floor(round(size(population, 2)/2, 2))
    ones_to_mask = zeros(1, size(population, 2));
    ones_to_mask(randi(length(ones_to_mask))) = 1;
    mask = (mask+ ones_to_mask) & 1) * 1;
end
%crossover
a_part = mask .* population(chr_idx_chosen(1), :);
b_part = mask .* population(chr_idx_chosen(2), :);
population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) .*
~mask;
population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) .*
~mask;
population(chr_idx_chosen(1), :) = population(chr_idx_chosen(1), :) +b_part
.*mask;
population(chr_idx_chosen(2), :) = population(chr_idx_chosen(2), :) +a_part
.*mask;
%sorting
for i =1:size(population, 1)
    population(i,:) =population(i,:) .*v;
end
A =population;
output =subsref( sortrows( [sum(A,2) A], -1), ...
    struct('type','()', 'subs',{':' ,1+(1:size(A,2))}) ) ;
sorted =((output & 1) * 1);
%save to population
for i = ceil(size(population,1)/2) : size(population,1)
    sorted(i, :) = round(rand([1, size(population, 2)]));
end
population =sorted;
iteration_limit =iteration_limit -1;
end
%last checking wmax
for i =1:size(population, 1)
    while sum(population(i,:) .* w) > wmax
        population(i, randi(size(population, 2)))= 0;
    end
end
%last sorting
for i =1:size(population, 1)
    population(i, :) = population(i, :) .* v;
end
A = population;
output =subsref( sortrows( [sum(A,2) A], -
1),struct('type','()', 'subs',{':' ,1+(1:size(A,2))}) ) ;
population = ((output & 1) * 1);
%presenting samplary chromosome
chromosome_after = population(i, :)

```

## RESULT:



chromosome\_before =

1 1 0 0 1 1 0 0 0 0

chromosome\_after =

0 0 1 0 0 0 0 0 0 0

## CONCLUSION:

The various functions which includes crossover operation , fitness scaling.

Genetic Algorithm plays a important role for obtaining an optimized solution and to find the best fitness values. As we increase the population size chromosome and fitness values with respect to population size.