CS480/580 Introduction to Artificial Intelligence (Fall, 2017)

## Assignment 2

1. Hill Climbing Search:

After running for 100 generations as per given constraints, the found global minimum is

**-22.6615367037** at **[0.004183209674385256, 0.016748788651023458].**

The given graph show below is the global minimum found in each generation. The x-axis represents the generations and the y-axis represents the global minimum.

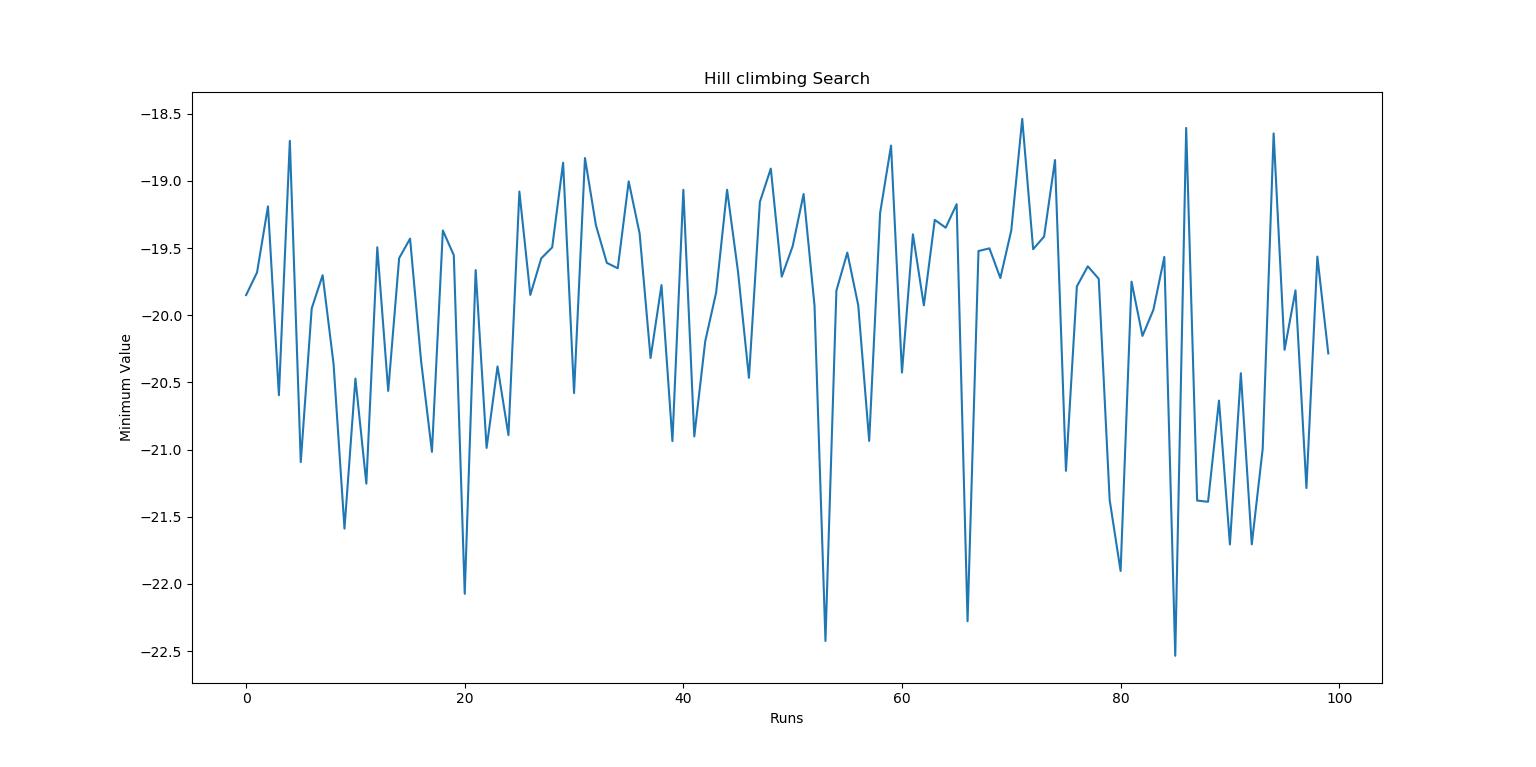


Fig: Distribution of global minimum for 100 runs

Here in hill climbing algorithm, the global minimum found in each generation is different i.e here the sometimes hill climbing algorithm stops after finding a local minimum in a region and the actual global minimum is found after a certain number of generations. This can be seen from the graph

1. Differential Evolution:

After running for 100 generation as per given constraints, the found global minimum is -**22.7182818285** at point **[-4.611566329315738e-17, -3.4834277233441134e-16]**.

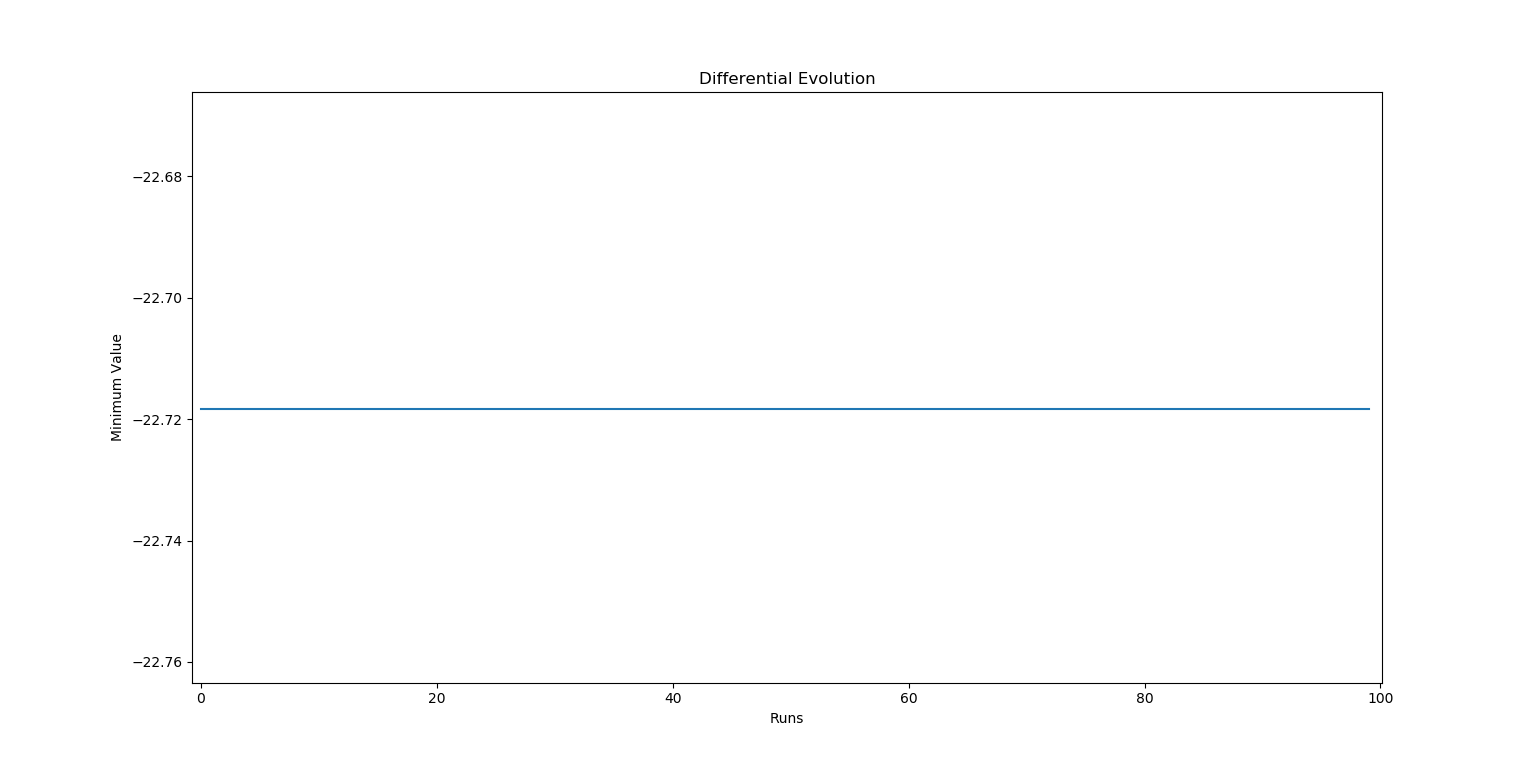


Fig: Distribution of global minimum for 100 runs

As you can observe form graph the global minimum found in each generation is almost same, so the graph consists of a straight line. This indicates that the differential evolution algorithm able to find the global minimum in the first run itself, where the hill climbing algorithm takes certain number of runs.

From this we can conclude that differential evolution algorithm is efficient and faster in find global minimum of an objective function.

And Differential evolution algorithm can find the global minimum, even many local minimum exists, whereas the hill climbing algorithms sometimes stops after finding a local minimum assuming it as a global minimum.

1. Travels man sales problem:

For solving the provided problem, the order-1 cross over and swap mutation is used.

And in this approach, the minimum path which obtained in the previous iteration and the global minimum path obtained till that iteration is included in the current population.

For the test case 1, the Least cost was nearly 23,505, as it generates randomly, for each run it may change, it will be almost between 23,000 and 39,000.

Similarly for the second test case the Least cost obtained is 20,058, and may change in between 20,000 to 35,000 every time we run.

For the third test case, the code ran for more than hour, and still running, till then the Least cost obtained was around 4,00,000. If we run for some more time. The least cost still can be reduced.

The solution for this test cases are almost success in finding the Lest cost.

For implementing the second cross over method, the method similar to uniform cross over is followed. In this method, by checking the each element in the parent list is checked whether it can be used for crossover not by generating a random probability for each element checking it with the crossover probability. And swaping those elements between two parent lists. If the element already exists in those lists, it was skipped. And the obtained two lists after swaping elements are returned as two childs.

Ex: [2,4,3,1,6]

[1,7,8,9,0]

Which may give the following lists after cross over:

[2,4,8,1,0]

[1,7,3,9,6]

Here the third and fifth element are swapped.