Network Design Problem Using Genetic Algorithm: An Empirical Study On Selection Operator

Anand Kumar
Department of Master of Computer Applications
AMC Engineering College, Bangalore
kumaranandkumar@gmail.com

Dr. N. N. Jani
Faculty of Computer Studies
Kadi Sarva Vishwavidyalaya, Gandhinagar, Gujrat
drnnjanicsd@gmail.com

ABSTRACT
This paper presents an influence of selection function in genetic algorithm for network design problem. A network design problem for this paper falls under the network topology category which is a minimum spanning tree with various types of constraint which makes it NP-hard problem. Selection function plays an important key role in genetic algorithm approach. Since many researchers have tried to solve this problem for small to mid size, we have explored the use of genetic algorithm with various selection functions with modification but without changing the nature of genetic algorithm. Various selection functions have been developed here as per the requirement of the problem and applied with the various size of network. Selection is not made only on the principle of "Survival of the fittest" rather it is developed according to the nature of problem. In this paper we have tried to show that how selection functions affects the performance of genetic algorithm and also shown that GA is an alternative solution for this NP-hard problem.

KEY WORDS
Genetic Algorithm, Network design, Selection function Minimum spanning tree.

1 INTRODUCTION
In Genetic Algorithm selection is the process to select the chromosomes in the population for reproduction. The concept behind selection is **"Survival of the fittest".** The fitter the chromosome, the more times it is likely to be selected to reproduce. There are many selection schemes for genetic algorithms (Gas) each with different characteristics. Since the nature of genetic algorithm is very uncertain, various selection functions can be used to derive optimal result. This paper presents the influence of various types of selection process with various size of network and it is the extension of the research work Network design problem [6]. This problem is one of the hardest problems in NP-hard category. There are no traditional methods available to solve this problem. A genetic algorithm approach to design the network is one of the ultimate solutions because traditional heuristics has the limited success. Researchers in operation research have examined this problem under the broad category of 'minimum cost flow problem' [1]. A simple GA approach is applied by many researchers [2],[3],[4] but in this paper we have shown the influence of selection function in genetic algorithm. Genetic Algorithms are being used extensively in optimization problem as an alternative to traditional heuristics. It is an appealing idea that the natural concepts of evolution may be borrowed for use as a computational optimization technique, which is based on the principle "Survival of the fittest" given by "Darvin". We have tried to show that the influence of selection function and the little variation in genetic algorithm approach is very effective.

Published by Research Publications, Chikhli, India

1.1 Network Design

In this paper network design is considered as network topology which is a spanning tree consists of various nodes considered as vertex. A tree is a connected graph containing no cycles. A tree of a general undirected graph G = (V,E) with a node (or vertex) set V and edge set E is a connected subgraph T = (V',E') containing no cycles with (n-1) edges where n is total no of node.

In this study undirected networks are considered with the weight (distance) associated with each node. For a given connected, undirected graph G with n nodes, a minimum spanning tree T is a sub graph of a G that connects all of $G's$ nodes and contains no cycles [5]. When every edge (i, j) is associated with a distance c_{ij} , a minimum spanning tree is a spanning tree of the smallest possible total edge cost

$$C = \sum c_{ij}$$

Where (i, j). $\in$ T

1.2 Genetic Algorithm
Genetic algorithms (GA) is a powerful, robust search and optimization tool, which work on the natural concept of evolution, based on natural genetics and natural selection..

Work flow of GA
1. Initialisation of parent population.
2. Evaluation
   a) Self loop check
   b) Isolated node or edge check
   c) Cycle check
   d) Store the best result
3. Selection of child population
4. Apply Crossover/ Recombination
5. Evaluation
6. Replace the result if it is better than previously stored.
7. Apply Mutation
8. Evaluation
9. Replace the result if it is better than previously stored.
10. Go to step 3 until termination criteria satisfies

2 NETWORK DESIGN PROBLEM PRESENTATION AND ITS

SOLUTION USING GENETIC ALGORITHM APPROACH

The Network design problem is considered as a unidirectional graph and represented with the help of adjacency matrix. Parent population in the form of chromosome is generated randomly according to the size of network. Number of gene in a chromosome is equal to number of node in a network. The total number of chromosome may vary and it is based on user input. Here a chromosome is generated for a 10 node network. The association between nodes is considered between positions to position.

node	1	2	3	4	5	6	7	8	9	10

chromosome	2	10	4	9	6	7	5	9	8	3

The logic behind association is that, the node [1] is connected with node 2; node [2] is connected with 10 and so on.

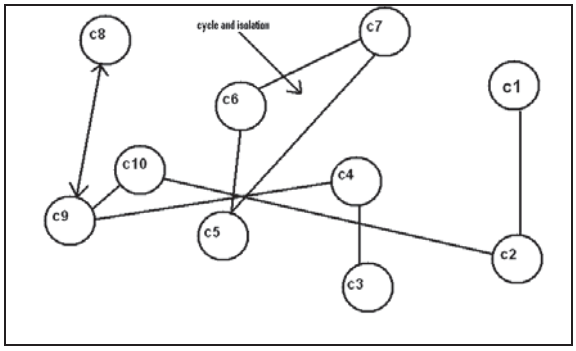

Figure 1

From fig-1 it is clear that this is not a spanning tree because of the isolated circle. Similarly with some other randomly generated chromosome, some other problems have been observed.

By observing these problems it has been concluded that there are three main reasons for illegal chromosome:

I. Self loop
II. Cycle and
III. Isolated node or edge.

Evaluation

By observing these problems, fitness functions have been developed [6] to evaluate these chromosomes. On the basis of these fitness functions, fitness points are given to the chromosomes and on the basis of these fitness points chromosomes are selected as a child population for next generation. Following fitness functions have been developed to evaluate chromosomes.

a) Self Loop
b) Isolated node or edge
c) Cycle

3 SELECTION

This is the main part of this paper. After the randomly generation of parent population and its evaluation, selection is the phase of genetic algorithm which decides the child population for the next generation from the current population. In genetic algorithm fit solution are likely to survive and bad solution are likely to die off. The purpose of selection is to emphasize the fitter individuals in the population in hopes that their offspring will in turn have even higher fitness. Following selection functions have been developed and tested for different size of network.

Following data structure have been used for all these developed functions.

Chromosomes: it is a matrix of size NXN to store N randomly generated chromosomes. After the fitness function, fitness point for each chromosome is stored in its last column. Subscript starts from 1.

S (1): it holds the no of row of matrix chromosome

S(2): it holds the no of column of matrix chromosome

Fitness: it is an array which holds the fitness value for each chromosome.

Random Selection: This function selects the chromosome on the basis of randomly generated number. If the randomly generated number is not within the range of no of chromosome, then it simply replaces with the fittest chromosome.

```
function [new_chromosomes] =
Random_selection(chromosomes)
a=size(chromosomes);
row = a(1);
col = a(2)-1;
k=1;
for i=1:row
    r=row*round(rand());
    if((r == 0) || (r > row))

    for l=1:row
     if(chromosomes(l,a(2)) == col+3)
      r=1;
      end

    end

 end
    if(r == 0)
        r=1;
    end
    for j=1:col
new_chromosomes(k,j)=chromosoms(r,j);
    end
    k=k+1;
end
```

Roulette wheel Selection : This function selects the chromosome on the basis of roulette wheel algorithm.

S = (fitness/Avg)
Avg = Sum of fitness / no. of chromosomes.
r →(0-S).

```
function [new_chromosomes] =
Roulette_ wheel_selection(chromosomes)
a=size(chromosomes);
```

```
disp(chromosomes);
row = a(1);
col = a(2)-1;

    r=row*round(rand());

s(row,1)=0;
temp=0;
for i=1:row
    temp=temp+chromosomes(i,a(2));

end
avg=temp/row;
disp('avg');
disp(avg);

disp('**temp**');

for i=1:row
    temp=(chromosomes(i,a(2))/avg);
    disp(temp);
end
temp=0;
for i=1:row

temp=temp+(chromosomes(i,a(2))/avg);
    s(i)=temp;
end

temp=s(i);
disp('s');
disp(s);
 k=1;
 for i=1:row
     r=rand*temp;
     disp('r');
     disp(r);
     for j=1:row
         if(r<s(j))
             for t=1:col

new_chromosomes(k,t)=chromosoms(j,t);
             end
             k=k+1;

             break;
         end
     end

 end
disp(new_chromosomes);
end
```

Best –Worst combination Selection: This selection function selects the chromosome as a pair of best-worst combination. First of all all the chromosomes are sorted on the basis of their fitness value and then selected as first-last, second-second last and so on.

```
function [new_chromosomes] =
Best_worst_selection(chromosomes)
a=size(chromosomes);
disp(chromosomes);

row = a(1);
col = a(2)-1;
k=1;
 t=2;
while(k<=row)
   for i=1:row

  if(chromosomes(i,a(2)) == (a(2)+t))
          for j=1:col

new_chromosomes(k,j)=chromosoms(i,j);
          end
          k=k+1;
       end

   end

   t=t-1;

end
disp(new_chromosomes);
return;
end
```

Fittest Selection: This function selects only fittest chromosomes up to a fixed fitness level. Here the fitness level is considered as

Fitness level = maximum fitness – 4.

```
function [new_chromosomes] =
Fittest_selection(chromosomes)
a=size(chromosomes);

disp(chromosomes);
row = a(1);
col = a(2)-1;
k=1;
 t=2;
while(k<=row)
   for i=1:row

       if(chromosomes(i,a(2)) ==
(a(2)+t))
          for j=1:col

new_chromosomes(k,j)=chromosoms(i,j);
          end
          k=k+1;
       end

   end
   if(k>row)
      break;
```

```
   end

   t=t-1;
   if(t< (-1))
       t=2;
   end

end
disp(new_chromosomes);
end
```

Selection sort: This selection function based on selection sort. It generates two random number for two random position. These two positioned chromosomes are compared and then selected. There are two selection function developed. First selection is based on greatest one and the second selection is made on smallest one. The procedure is repeated N times.

```
Function [new_chromosomes] =
selection_sort1(chromosomes)
a=size(chromosomes);
row = a(1);
col = a(2)-1;
k=1;
 t=2;
   for i=1:row

     p  = round(rand()* (a(2)-1));
     q =  round(rand()* (a(2)-1));
             if(p == 0)
                 p=1;
             end

             if(q == 0)
                 q=1;
             end

       if(chromosomes(p,a(2)) >
chromosomes(q,a(2)))
          for j=1:col

new_chromosomes(k,j)=chromosoms(p,j);
          end
          k=k+1;
       else
           for j=1:col

new_chromosomes(k,j)=chromosoms(q,j);
          end
          k=k+1;
       end

   end

return;
end

-------------------------
```

```
function [new_chromosomes] =
selection_sort2(chromosomes)
a=size(chromosomes);
row = a(1);
col = a(2)-1;
k=1;
 t=2;
   for i=1:row
       p  = round(rand()* (a(2)-1));
       q =  round(rand()* (a(2)-1));
             if(p == 0)
                 p=1;
             end

             if(q == 0)
                 q=1;
             end

       if(chromosomes(p,a(2)) <
chromosomes(q,a(2)))
          for j=1:col

new_chromosomes(k,j)=chromosoms(p,j);
          end
          k=k+1;
       else
           for j=1:col

new_chromosomes(k,j)=chromosoms(q,j);
          end
          k=k+1;
       end

   end
 return;
end
```

Crossover/Recombination

Chromosomes have been crossover on a single point. First chromosome on first column, with second chromosome. Third chromosome on second column , with fourth chromosome and so on.. If crossover point exceeds the maximum column no. then it starts from first column again.

Mutation

Only those chromosomes have been mutated which does not have maximum fitness point.

4 EXPERIMENTAL RESULT

Ten different network of size 10,20……100 have been used. Total number of chromosomes and total number of generations for each network is 100. The experiment is done in MATLAB R2008a version 7.6.0.324. All the selection function is experimented with various size of network and crossover and mutation function. Following table displays the result:

Table -1: Minimum Cost Of Network For Various Selection Functions

Size of Network ↓	Random	Roulette wheel	Best–Worst	Fittest	Selection sort	
					1	2
10	332	**246**	309	291	282	275
20	725	624	714	644	**613**	653
40	1475	**1324**	1468	1408	1306	**1271**
60	2365	2074	2028	2459	2083	**1993**
80	2854	**2708**	3047	2934	2756	2833
100	3855	**3541**	3728	3896	3545	3620

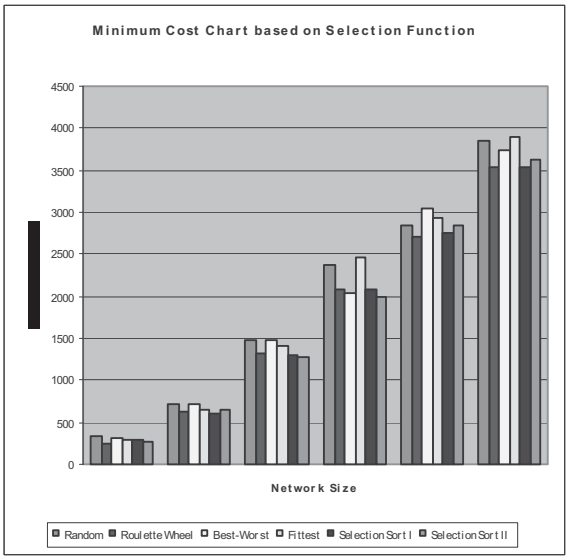

Figure 2

5 CONCLUSIONS

From the above experimental result of Table-1 and the chart shown in Figure-2, it is clear that, selection procedure is one of the main factor of genetic algorithm. From these six selection function it is observed that roulette wheel and selection sort function gives the better result.. This is the improved approach of evolutionary computing which gives the very positive result. We have described the importance of selection function.. The effectiveness of the methodology however can be increased by applying the various genetic operators with variations of network size as the densely connected locations.

6 ACKNOWLEDGEMENTS

Our thanks to the reviewers for their cogent and insightful comments.

7 REFERENCES

[1] Hamdy A. Taha. 2007. Operation Research An Introduction

[2] S.K. Basu, 2005. Design Methods and Analysis of algorithms (PHI) ISBN : 81-203-2637-7

[3] Melanie M. (1998). An Introduction to genetic Algorithm (PHI) ISBN 81-203-1385-5

[4] Michael D. Vose. 1999. The simple genetic algorithm : (PHI) ISBN 61-203-2459-5

[5] Narsingh Deo, 2000. Graph Theory with Applications to Engineering and Computer science: (PHI)

[6] Anand Kumar, Dr. N.N.Jani, Proceeding of the International Conference on Mathematics and Computer Science ICMCS FEB 2010 Chennai.