IMPROVING TRAFFIC MANAGEMENT USING GENETIC ALGORITHMS AND ANT COLONY OPTIMIZATION METHODS

Sankar. A[1], Vijay Sekar. E[2], M. Yamuna[3]

[1, 2], SCSE, VIT University (India)

[3]SAS, VIT University (India)

ABSTRACT

Traffic Congestion is a major issue in several cities of the world. Our paper proposes a solution to tackle this problem with the help of rerouting and finding best paths for faster travel time. The objective is to achieve a significant reduction in traffic congestion by plotting the best path between two points and also taking into account the traffic densities of the constituent edges at the same time. The proposed work is to achieve an optimal solution by using the concepts of Genetic Algorithms and Ant Colony Optimization techniques to develop a hybrid algorithm that produces significantly better results.

Keywords: Ant Colony Optimization, Artificial Intelligence, Genetic Algorithms, Graph Theory, Traffic Management

I. INTRODUCTION

Managing Traffic in a city has always been a problem despite trying various different technologies to cope up with it. There are various factors that influence the equation. Just knowing the route from the source to destination is not sufficient. Along with the best path from two places, we also need to consider the traffic density along the roads that make up the path. Also, the traffic management is dynamically changing issue, meaning the state of traffic in a particular area is rapidly changing with time. So if we have a best path from A-B, it might still not be the same after a time interval of about 10 minutes, for the traffic density along those roads would have significantly increased by the re-routing done.

II. GENETIC ALGORITHMS

Genetic algorithms are imitation of the natural process of selection from a population of abstract representations of candidate solutions to generate solutions to optimization problems. The candidate solutions are often represented as binary strings. A genetic algorithm requires a standard representation of solution in the form of an array of bits. The binary representation allows simple crossover functions to be used [1].

The major components in genetic algorithms is discussed in the following sections [2], [3], [4] since we use genetic algorithm in the proposed method.

2.1 Initialization

Initially, several solutions are generated using a random function. This set of solutions serves as the initial population. The population size is usually huge to allow simultaneous verification of large portions of the solution set.

2.2 Selection

During creation of each successive generation, a proportion of existing population is selected to breed the new generation. The fitness function (fitness based process) assigns a fitness value to each member of the population. The solutions with better fitness values are chosen for breeding purposes. The fitness function must be chosen such that less proportion of the total solution set fit the better values of the fitness function. A small portion of weak solutions are allowed to pass the fitness test so that premature convergence doesn't occur.

2.3 Crossover

The new generation is created using crossover. The solution string is divided at a certain point and one half of the solution is exchanged for another half from a different solution. And two new solutions are created. Crossover doesn't always occur; there is a 60-70% possibility of it happening.

2.4 Mutation

After selection and crossover, new solutions are created. But, it is possible that some unexplored feature, which currently doesn't exist in the current population, might be part of the actual result. In order to ensure inclusion of these results, we leave a very small probability of one of the bits of the solution being flipped forcibly.

2.5 Termination

This indicates to end of the genetic algorithm process. The final generation of a GA either contains the required solutions, or the maximum number of iterations has been reached. Expiry of budget/time limit might also lead to termination.

III. ANT COLONY OPTIMIZATION

Ant Colony optimization is a nature inspired algorithm that uses the behaviour of ants in modelling optimization algorithms. The algorithm is developed on the basis of working of the ant colony and how the food collection by the ant colony happens. The ants do not communicate to one another but instead leaves down a pheromone trail indicating a favourable path.

As time passes, the frequently traversed paths will keep accumulating the concentration of pheromone while the less frequently visited ones will not have such high concentrations. So the other ants could follow the best path by choosing that one which has a greater pheromone concentration which can easily sensed by the ant colony.

Also the pheromone deposit evaporates over time; this implies that only the frequently used paths will have the greater pheromone deposit while other less-frequented paths would be ignored over time with declining concentrations of the pheromone.

We can map the ant's behaviour to develop a system to replicate various different situations & scenarios and help in finding the best path in the road network of a given city.

We can also use the pheromone concentration levels to map out frequently travelled paths. This would help us in finding the busiest path, those that need to be avoided in a real-time system scenario [5].

IV. PROPOSED WORK

We propose to solve the above said traffic management issue by using the principles of the Genetic Algorithms and Ant Colony optimization. Initially we take a given state of traffic as input to the system and the source and destination the user would be going to. Using the given input, the best possible paths between them are computed. Then the traffic densities in the roads are computed and are checked if any of them exceed the threshold value. If they do, those paths are rejected. For basics of graph theory we refer to [6].

4.1 Representing the Road Network as a Graph

Step 1: Draw the Road System as a graph by taking the intersections as vertices and the roads as edges. Every edge has a length (length of the road), current traffic density and threshold traffic density. We assume the car has to travel from Start Vertex‘s’ to Destination Vertex‘d’.

Step 2: Label each of the edges from e_1 to e_n.

4.2 Selecting Short Paths using Modified Genetic Algorithm

Step 3: Create binary strings of length n (n = no. of edges) and if k^{th} digit is 1, then edge e_k is a part of the solution, else e_k is not present in the solution path.

Step 4: Find fitness value of each string using the fitness function given as:

Fitness (string) = (Sum of edge lengths * No. of components +
(2 x Sum of edge lengths (if s is not included)) +
(2 x Sum of edge lengths (if d is not included)) +
$\sum$ [-1* (average edge weight * max length of path from s to d / 2) *
(Length of component) For all Components +
(Sum of edges of component) +
(Sum of edge lengths (If any vertex has degree > 3)*
(No. of vertices with degree > 3))]) (1)

Step 5: We select strings with low fitness values and perform crossover.

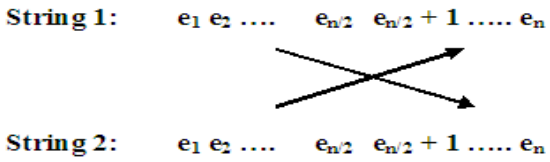


Fig.1 Crossover Method

Step 6: Then, perform mutation with a probability of 0.01%. Mutation is done by changing one of the digits of the solution from 1 to 0 or 0 to 1.

Step 7: Repeat steps 4 to 6 for several iterations till we get solutions with Minimum Fitness values.

4.3 Choosing the Efficient Path Using Ant Colony Optimization

Step 8: Select the path with least fitness value and check if the traffic density on this path is below threshold values. If they are, then route traffic through this path.

Step 9: As the traffic density changes with time, density at the path selected in step 8 keeps increasing. This implies that the pheromone deposit is constantly increasing along this path. If any of the edges exceed the threshold density value, another path is selected, generated from the Genetic Algorithm in **Step 7** and the pheromone deposit along the initial path starts decreasing.

Step 10: Now, the traffic congestion in the first path reduces after certain amount of time, and the pheromone deposit along that path reduces while the traffic is being routed on an alternate path. When the traffic density falls below the threshold value, switch to the initial path again, as it is shorter to travel and the pheromone level starts to increase again.

4.4 ACO Algorithm

```
Procedure ACO
while(not_termination)
{
        generateSolutions(GA);
        routeTraffic(solutions);
        pheromoneUpdate(edges);
}
End
```

V ILLUSTRATION

5.1 Draw the Road Network as a Graph (Vertices are Intersections and Edges are Roads)

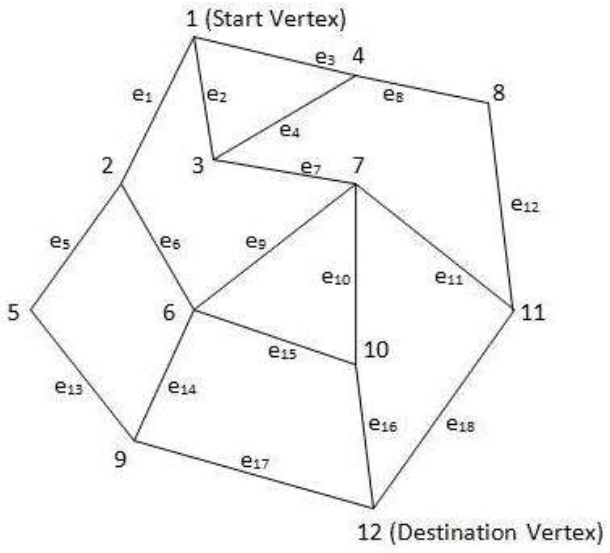


Fig 2: Graph Representation of Road Network

5.2 Creation of Edge Matrix E by Assigning Edge Numbers to All Possible Paths in the Given Network.

Table 1: Edge Matrix

Edge No.	Edge	Edge weight
1	1-2	8

2	1-3	6
3	1-4	5
4	3-4	7
5	2-5	7
6	2-6	5
7	3-7	4
8	4-8	5
9	6-7	6
10	7-10	7
11	7-11	8
12	8-11	5
13	5-9	8
14	6-9	4
15	6-10	6
16	10-12	9
17	9-12	6
18	11-12	7

5.3 Generating Initial Population

If the edge is part of the solution, the value is 1, else the value is 0. Use a random generator to create the solutions.

Initial Population (with fitness function F(x)):

Fitness = (113 * No. of components + (2 x 113 (if s is not included)) +
(2 x 113 (if d is not included)) +
$\sum$ [- (7 * 4) * (Length of component) For all Components +
(Sum of edges of component) + (113 *(No. of vertices with degree > 3))])

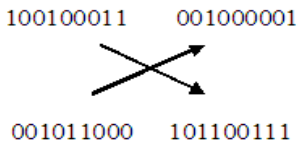
Select the strings with lower fitness values and discard those with higher values.

Table 2: Fitness Value of Initial Solutions

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**F(x)**
1	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	1	322
0	1	0	0	1	0	1	1	0	0	0	0	0	0	1	0	0	0	566
0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1	1	1	367
1	0	0	0	1	1	0	1	0	0	0	1	0	1	1	0	0	0	517
0	0	0	1	0	0	1	1	0	1	0	1	1	0	0	0	0	0	659
0	0	0	0	1	1	0	0	1	0	0	1	0	1	1	0	0	0	656

5.4 Cross Over

Swap the halves of two strings in the population to create two new string.



By this we generate new strings:

100100011101100111

001011000001000001

5.5 Mutation

Forcibly change one digit of the string generated after crossover.

00101100001000001 → 00101100101000001

Table 3: Genetic Strings of the Solutions after 50 iterations

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	0	0	1	0	1	0	1	0	0	0	0	1	0	1	0	0
0	0	1	0	1	0	0	1	0	0	0	1	1	0	0	0	0	1
0	1	0	0	0	1	1	1	0	0	0	1	0	0	1	0	1	0
0	1	0	1	1	0	0	0	0	0	0	0	1	0	1	1	0	0
1	0	0	1	0	1	0	0	0	1	1	0	0	1	1	0	0	1

Table 4: Genetic Strings of the Solutions after 500 iterations

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0
1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0
0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1
0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1

Paths generated by the Genetic algorithms are

Path 1: $e_3\ e_8\ e_{12}\ e_{18}$ Length = 22

Path 2: $e_1\ e_6\ e_{14}\ e_{17}$ Length = 23

Path 3: $e_2\ e_7\ e_{11}\ e_{18}$ Length = 25

Path 4: $e_2\ e_7\ e_9\ e_{14}\ e_{17}$ Length = 26

Path 5: $e_2\ e_7\ e_{10}\ e_{17}$ Length = 27

Path 6: $e_1\ e_5\ e_{13}\ e_{17}$ Length = 29

5.6 Generating Solutions

We select the shortest 3 paths for applying Ant Colony Optimization based on traffic density

Path 1: $e_3\ e_8\ e_{12}\ e_{18}$ Length = 22

Path 2: $e_1\ e_6\ e_{14}\ e_{17}$ Length = 23

Path 3: e_2 e_7 e_{11} e_{18} Length = 25

The chosen paths have been highlighted in the graph (Fig 3)

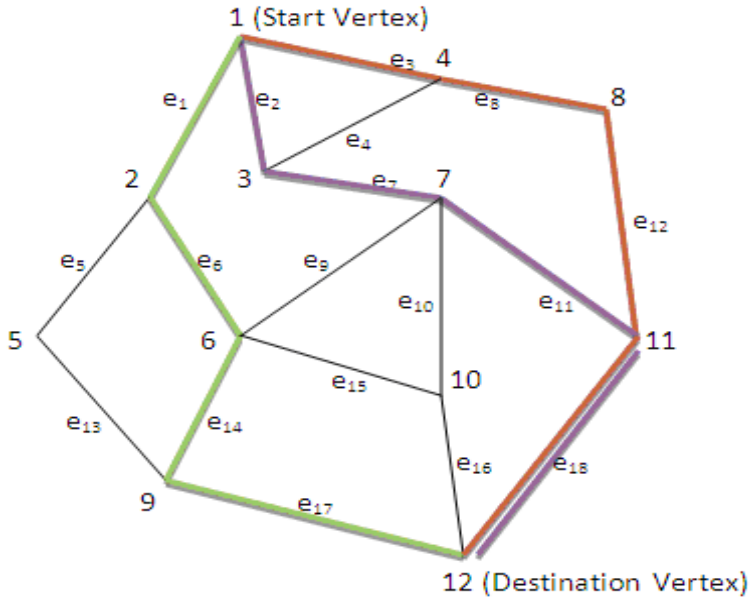


Fig 3: Top 3 Optimal Solutions

5.7 Route Traffic() Function

The traffic from 1 to 12 is routed through Path 1. The traffic densities of the edges on the path are increased.

5.8 Pheromone Update() Function

The pheromone value of Path 1 keeps on increasing till one of the edge densities reach threshold. When the threshold value is exceeded, then the pheromone value of Path 1 is decreased, and the pheromone value for the next best path (Path 2) is increased.

After a few iterations, the densities of the edges in Path 1 decreases again. Hence, the pheromone value can be increased again.

VI. CONCLUSION

The proposed solution can be used to solve traffic congestion between two locations. By applying it on every pair of vertices (intersections) iteratively, we can reduce congestion further. The Genetic Algorithm ensures that the path is of shorter length, and the Ant Colony Optimization ensures that traffic density along any edge[road] doesn't go too high. Although the genetic algorithm doesn't always guarantee optimal solution for all cases, the overall outcome of the modified algorithm with ant colony optimization seems to be promising as explained in the illustration.

REFERENCES

[1] http://en.wikipedia.org/wiki/Genetic_algorithm

[2] http://www.obitko.com/tutorials/genetic-algorithms/ga-basic-description.php

[3] H. Bhasin and N. Singla, (2012), Modified Genetic Algorithms Based Solution to Subset Sum Problem, IJARAI, Vol. 1 (1).

[4] H. Bhasin and S. Bhatia, (2011), Application of Genetic Algorithms in Machine learning, IJCSIT, Vol. 2(5).

[5] http://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms

[6] Narsingh Deo *Graph theory with applications to engineering and computer science* (Prentice Hall India 2010).