International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-4 Number-1 Issue-14 March-2014

Implementing Amalgamation of Graphs on Set of Shorter Path Algorithm Using Genetic Algorithm

Tarak Nath Paul[1], Abhoy Chand Mondal[2]

Abstract

The aim of this paper is to verify the algorithm, "Search the Set of Shorter Paths" [23] which clip the graph into levels (Levelled Graph) and find the set of shorter paths from source to destination. When the algorithm operates on heuristically amalgamated graphs of three types is executing properly. The graph or network consists of the amalgamation of different types graph as Roadways, Railways and Airways. The amalgamation can be of any type incorporating packet network, pipeline network for liquid transportation and many more. The algorithm executes efficiently irrespective of the types of graph or network it has been applied to. The algorithm not only finds the shortest path but also find out the other shorter paths from the source to the destination node. The selection of graphs is done arbitrarily as to express the capability of the designed algorithm and to express the connection of nodes with other nodes. The proposed algorithm is compared with dijkstra's algorithm and the results are satisfactory. Simulated results are formulated using Matlab. The result assures the experimental potential of the algorithm.

Keywords

Shortest Path Algorithm, Network, Routing, Graph, Genetic Algorithm, Chromosome, Mutation and Fitness Criteria.

1. Introduction

The idea of this hybrid algorithm is implemented with the help of different types of graphs, leveling of graph and genetic algorithm. Checking and testing is based upon the ideas of genetic algorithm [1].The possible paths are represented by chromosomes and other possible paths are achieved by fussing them.

Manuscript received February 07, 2014.

Tarak Nath Paul, Computer Science Department, The University of Burdwan, Burdwan, West Bengal, India.
Abhoy Chand Mondal, Computer Science Department, The University of Burdwan, Burdwan, West Bengal, India.

Dijkstra's Algorithm finds the paths by weight precedence. It solves the single source shortest path issue [5][12][22] when the weights are non negative values. It is also a greedy algorithm and also similar to Prim's algorithm. The algorithm starts at the source node and a tree grows which reaches to all the nodes describing the shortest path. [6][8][10][13].

In the field of artificial intelligence genetic algorithm is a heuristic search algorithm. This focuses the idea of evolution of every individual in a specified population [7][9][11]. Genetic algorithm is an evolutionary algorithm that executes problem using natural selection, inheritance, crossover, mutation, selection and termination operations using fitness functions. The idea behind this is survival for the fittest (Law given by Charles Darwin). The primary monograph on the topic is Holland's (1975). A population undergoes selection including operators such as Selection, Cross-over and Mutation. A fitness function is designed to evaluate the fitness of the individuals. Based on the fitness function the best individuals are fussed together and the new populations is formed and applied to the algorithm until an optimal solution is obtained. [1][2][3][4].

2. Dijkstra's Algorithm

An algorithm for searching graph geodesics that searches the shortest path between two vertices in the particular graph. Its function is to construct a shortest path tree form source to destination vertex. The worst case running time for Dijkstra algorithm on raph with n nodes and m edges is $O(n^2)$ as because it allows for directed cycles. It even finds the shortest path form the source node to every node in the given graph. Dijkstra's algorithm uses the greedy approach to figure out the single source shortest path problem. It repeatedly takes from the unselected vertices v to nearest source s and declares the distance to be the actual shortest distance from s to v. The edges of v are then examined to see if their destination can be reached by v followed by the relevant outgoing edges. G = (V, E), Where V – is a set of vertices; E is a set of edges. [14][15][17].

Dijkstra's algorithm keeps two sets of vertices:

S → the set of vertices whose shortest paths form the source have already been determined.

V – S → the remaining vertices.

The other data structures needed are:

d – array of best estimates of shortest path to each vertex.

pi – an array of predecessors of each vertex.

The basic mode of operation is:

1. Initialize d and pi,
2. Set S to empty,
3. While there are still vertices in V-S,
    I. Sort the vertices in V-S according to the current best estimate of their distance from the source,
    II. Add u, the closest vertex in V-S, to S,
    III. Relax all the vertices still in V-S connected to u

Dijkstra (G, W, S)

```
1. INITIALIZE SINGLE-SOURCE (G, s)
2. S ← { }        // S will ultimately contains vertices of final shortest-path weights from s
3. Initialize priority queue Q i.e., Q ← V[G]
4. while priority queue Q is not empty do
5.    u ← EXTRACT_MIN(Q)     // Pull out new vertex
6.    S ← S È {u}  // Perform relaxation for each vertex v adjacent to u
7.    for each vertex v in Adj[u] do
8.       Relax (u, v, w) [18] [21]
```

3. Genetic Algorithm

Genetic Algorithm (GA) was introduced in United States in 1970s by John Holland at Michigan. GA must represent a solution to problem as a genome (or chromosome). The genetic algorithm then creates a population of solutions and apply genetic operator. Genetic Algorithms are the heuristic search and optimization procedures which takes the idea from natural evolution. The implementation of genetic algorithms is as follows:

1. Initialization: At the very beginning numerous individual solutions are randomly selected to form an initial population.
2. Selection: Individual solutions are picked out according to the fitness function.
3. Reproduction: Crossover, Mutation and Selection methods.
4. Termination: The process continues until a termination condition is identified. [16][19]

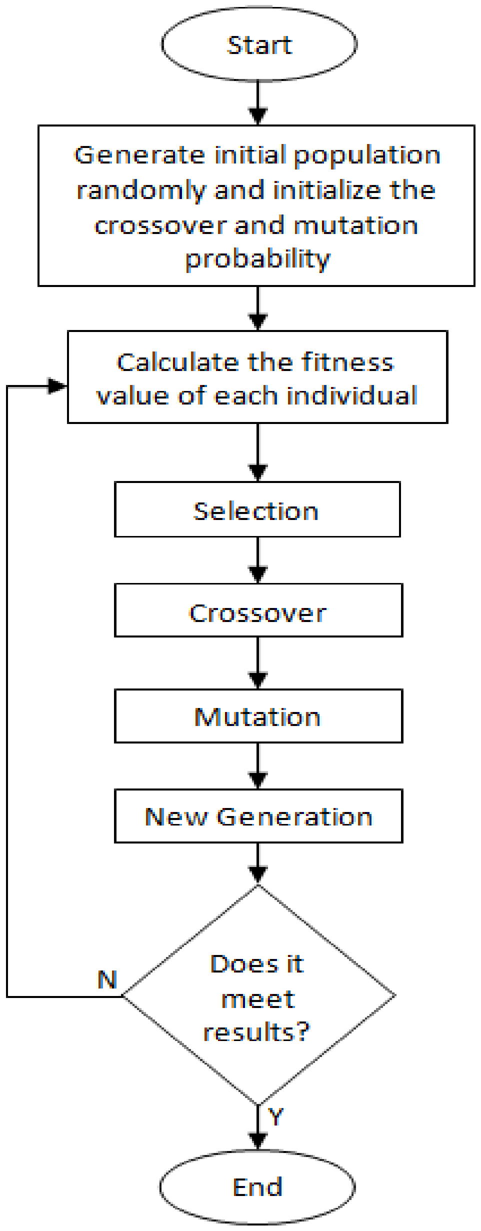
Start
Generate initial population randomly and initialize the crossover and mutation probability
Calculate the fitness value of each individual
Selection
Crossover
Mutation
New Generation
Does it meet results?
N
Y
End

Figure 1: Flow Chart for Genetic Algorithm

4. Proposed Algorithm

Let us consider an arbitrary graph of n (=27) number of nodes (Cities/Town etc.). Consideration of real time data about their location and connectivity between the nodes are discussed in [23] and it is also possible with arbitrary data. Connection and Distance between nodes of is shown in Figure 2 to 4 & 7 and tabular representation in Table 1. Let Graph G = {V, E} where V = {P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27} and E = {{P1, P6}, {P1, P7}, {P1, P25}, {P1, P4}, {P1, P2}, {P2, P4}, {P2, P5}, {P3, P2}, {P3, P5}, {P3, P9}, {P3, P27}, {P3, P10}, {P4, P7}, {P4, P8}, {P4, P5}, {P5, P9}, {P5, P8}, {P6, P18}, {P6, P13}, {P6, P7}, {P10, P9}, {P10, P15}, {P10, P22}, {P18, P13}, {P18, P19}, {P18, P25}, {P19, P23}, {P19, P25}, {P20, P23}, {P20, P24}, {P21, P24}, {P21, P27}, {P21, P22}, {P22, P15}, {P22, P27}, {P24, P27}, {P23, P25}, {P23, P24}, {P26, P25}, {P26, P23}, P26, P24}, {P26, P27}, {P4, P13}, {P4, P14}, {P5, P14}, {P5, P15}, {P23, P13}, {P23, P14}, {P24, P14}, {P24, P15},{P7, P11}, {P7, P13}, {P8, P11}, {P12, P11}, {P13, P11}, {P14, P11},{P8, P12}, {P9, P12},{P15, P12}, {P9, P15}, {P12, P14}, {P13, P16}, {P13, P19}, {P19, P16}, {P17, P16},{P14, P16}, {P14, P17}, {P15, P17}, {P15, P21}, {P21, P17}, {P20, P17}, {P19, P16}, {P20, P16}}.

If the count of the nodes reduces then the execution time will also fall. And the search space will also reduce for the shortest and shorter paths. Procedure to create the Reduced Graph - Select those nodes, whose x-values and y-values lies between the x-values and y-values of the source and destination respectively and whose magnitude is less than the magnitude of the source and the destination nodes, shown in Figure 6 followed by Algorithm to create Simplified - Graph from Sub - Graph. Now let us look at the nodes for a graph of various numbers of cities for the co ordinates. The spaces between two cities through these channels like rail, road or air are the weights of the connecting edge. Now let us cut down the nodes of the graph and then we can be able to create new and smaller graph or sub graph.

Traversing a smaller graph of less number of nodes will take less times. Less time means high performance. Creation of sub graph is possible only if we select those nodes which will provide the set of shorter paths. Nodes provided by “Algorithm for creation of Simplified Graph from the Sub Graph” are the responsible nodes for shorter paths. Consider the example using Rail, Road and Airways network.

To prove the analysis of the proposed algorithm is based on the graph of different types, such as Roadways, Railways and Airways are merged according to Genetic Algorithm and Graph Information System in an Intelligent Traversing Method of a Graph.

The detail analysis of my work is based on the following ingredients (Road, Rail and Airways) merged with Genetic Algorithm and Geospatial Information System in Intelligent Traversing Method of a Graph. The road map shown in the figure 2 is a two disjoint graph containing seven cities or vertexes each in a single graph. The information regarding the location, position, weight and connectivity of the city can be revealed by Geographic Information System or any arbitrary data shown in Figure 2.

Road ways (Road Map)

The road map given in the Figure- 2 is a disjoint map (graph). In other words we can say that it is the combination of two sub maps (or sub graph) containing seven cities each. And the information regarding any city can be retrieved form from the GIS or Arbitrary Data Management System. Information regarding height, magnitude, altitude, distance from the source, distance form temporary source to destination, traffic of a city (place / node / station), etc. This is the place where mutation can take place at the time of high traffic or any other type of problem. And we can modify the progress to destination from source. The data representation of the Road map is shown in the Table 1. The entries will describe the travelling distance from one node to another. Road ways can be recognized by red lines and blue blocks in data Table 1.

Railways (Rail Map)

The railway map given in the Figure-3 is a connected graph. Here every node is connected to at least one node. The graph is consists of 25 nodes (cities). The data representation of the Railways map is shown in the Table 1. The edges are travelling distance from one city to another. Railways are recognized by yellow lines and green blocks in the table.

Airways (Airways map)

The airways map given in Figure-4 is a connected graph. The graph is consists of 17 nodes. The matrix format of the Airways map is shown in the Table 1. The edge will discuss the travelling time from one city to another. Airways are recognized by blue lines.

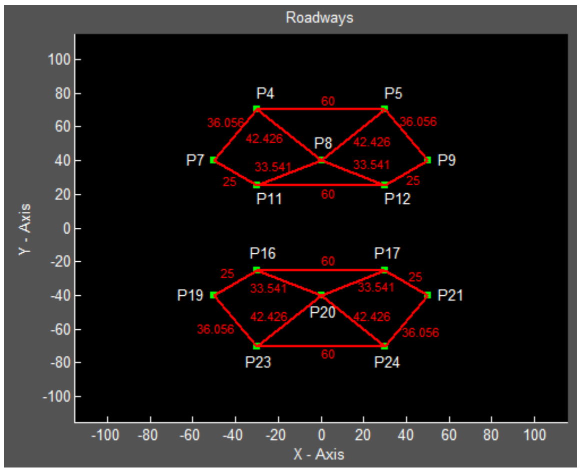

Figure 2: Graph of Roadways

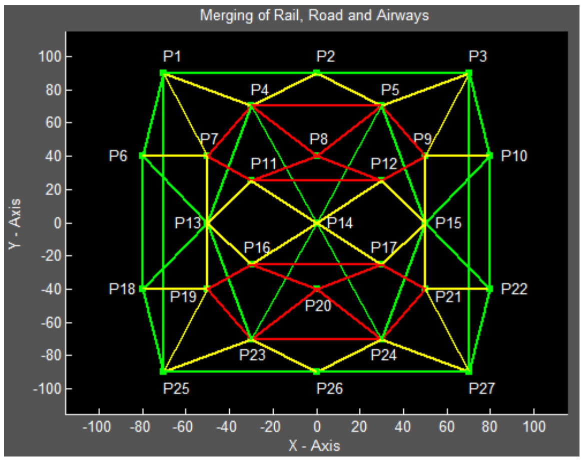

Figure 5: Combined Graph

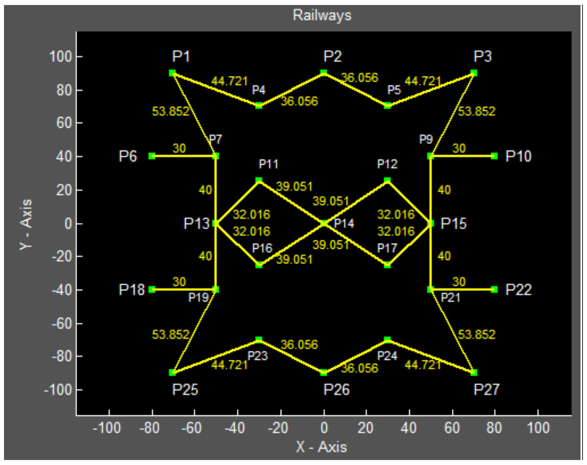

Figure 3: Graph of Railways

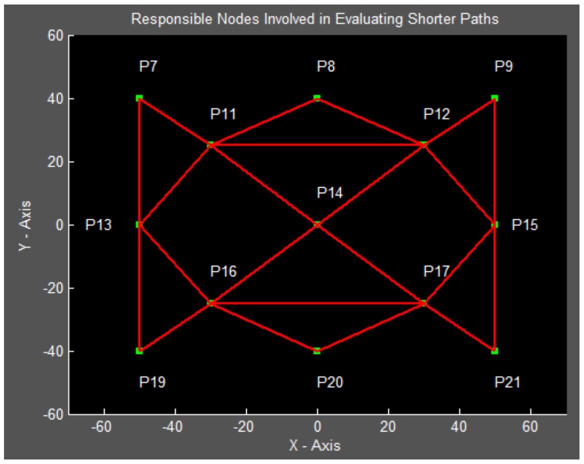

Figure 6: Require Graph with less nodes

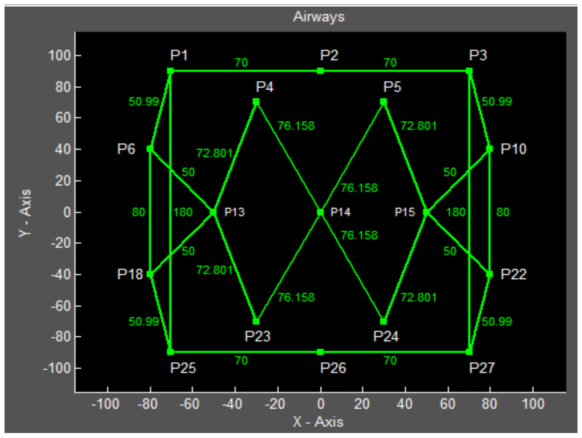

Figure 4: Grapy of Airways

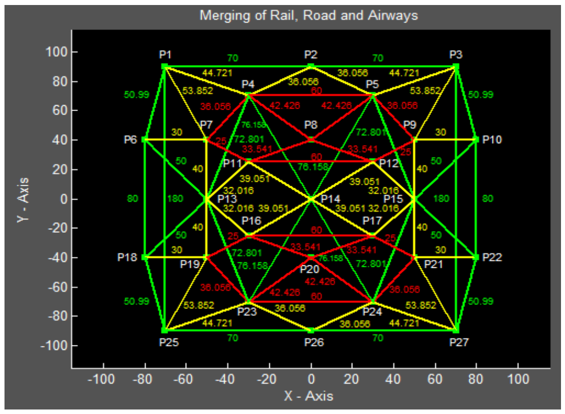

Figure 7: Merge of Rail, Road and Airways with it values

Table 1: Combination of Rail, Road and Airways

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27
P1	0	70	0	44.7214	0	50.9902	53.8516	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	180	0	0
P2	70	0	70	36.0555	36.0555	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P3	0	70	0	0	44.7214	0	0	0	53.8516	50.9902	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	180
P4	44.7214	36.0555	0	0	60	0	36.0555	42.4264	0	0	0	0	72.8011	76.1577	0	0	0	0	0	0	0	0	0	0	0	0	0
P5	0	36.0555	44.7214	60	0	0	0	42.4264	36.0555	0	0	0	0	76.1577	72.8011	0	0	0	0	0	0	0	0	0	0	0	0
P6	50.9902	0	0	0	0	0	30	0	0	0	0	0	50	0	0	0	0	80	0	0	0	0	0	0	0	0	0
P7	53.8516	0	0	36.0555	0	30	0	0	0	0	25	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P8	0	0	0	42.4264	42.4264	0	0	0	0	0	33.541	33.541	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P9	0	0	53.8516	0	36.0555	0	0	0	0	30	0	25	0	0	40	0	0	0	0	0	0	0	0	0	0	0	0
P10	0	0	50.9902	0	0	0	0	0	30	0	0	0	0	0	50	0	0	0	0	0	0	80	0	0	0	0	0
P11	0	0	0	0	0	0	25	33.541	0	0	0	60	32.0156	39.0512	0	0	0	0	0	0	0	0	0	0	0	0	0
P12	0	0	0	0	0	0	0	33.541	25	0	60	0	0	39.0512	32.0156	0	0	0	0	0	0	0	0	0	0	0	0
P13	0	0	0	72.8011	0	50	40	0	0	0	32.0156	0	0	0	0	32.0156	0	50	40	0	0	0	72.8011	0	0	0	0
P14	0	0	0	76.1577	76.1577	0	0	0	0	0	39.0512	39.0512	0	0	0	39.0512	39.0512	0	0	0	0	0	76.1577	76.1577	0	0	0
P15	0	0	0	0	72.8011	0	0	0	40	50	0	32.0156	0	0	0	0	32.0156	0	0	0	40	50	0	72.8011	0	0	0
P16	0	0	0	0	0	0	0	0	0	0	0	0	32.0156	39.0512	0	0	60	0	25	33.541	0	0	0	0	0	0	0
P17	0	0	0	0	0	0	0	0	0	0	0	0	0	39.0512	32.0156	60	0	0	0	33.541	25	0	0	0	0	0	0
P18	0	0	0	0	0	80	0	0	0	0	0	0	50	0	0	0	0	0	30	0	0	0	0	0	50.9902	0	0
P19	0	0	0	0	0	0	0	0	0	0	0	0	40	0	0	25	0	30	0	0	0	0	36.0555	0	53.8516	0	0
P20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33.541	33.541	0	0	0	0	0	42.4264	42.4264	0	0	0
P21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40	0	25	0	0	0	0	30	0	36.0555	0	0	53.8516
P22	0	0	0	0	0	0	0	0	0	80	0	0	0	0	50	0	0	0	0	0	30	0	0	0	0	0	50.9902
P23	0	0	0	0	0	0	0	0	0	0	0	0	72.8011	76.1577	0	0	0	0	36.0555	42.4264	0	0	0	60	44.7214	36.0555	0
P24	0	0	0	0	0	0	0	0	0	0	0	0	0	76.1577	72.8011	0	0	0	0	42.4264	36.0555	0	60	0	0	36.0555	44.7214
P25	180	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50.9902	53.8516	0	0	0	44.7214	0	0	70	0
P26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36.0555	36.0555	70	0	70
P27	0	0	180	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	53.8516	50.9902	0	44.7214	0	70	0

The sub - graph has been created by special phenomenon. In the sub graph only those nodes are considered which are responsible for the shortest paths or responsible for other shorter paths. The connection of every node is represented in Figure-5. It shows the possible connection of every nodes of every graph. And the data representations of the connection between nodes are shown in Figujre-7. Minimum required nodes which are responsible for the shortest and group of shorter paths are shown in Figure-6.

Sub – Graph to Simplest – Graph Conversion Algorithm.

```
Step 1:  Choose Source Node. [TSource = Source, TSource - Temporary Source]
Step 2:  Create First Level.
Step 3:  Place TSource in the Level.
Step 4:  Find connecting nodes to TSource and Add it to Queue.
Step 5:  If all Nodes of the Same Level not Traversed.
                 TSource = un – Traversed node in the same Level.
                 Goto Step 6.
         else
                 Goto Step 3
Step 6:  Create the next Level (Level++).
Step 7:  Add the Queue to the Level.
Step 8:  If Destination Found && Every Nodes is Traversed in the same Level
                 Goto Step 9
         else
                 TSource = Next un – Traversed node in the Same level.
         Goto Step 4
Step 9:  End.
```

The algorithm that is discussed above (Figure 12 Flow Chart of same), TSource is considered as Temporary Source. Firstly our source from where we need to travel is selected and then the destination to where we need to reach is considered. Now creating a level is important. Then we place the source in the level and these nodes in the next level will be connected to the source node. Now we will move down to the next level and will do the same with every node in the level. The process will continue until the destination is reached, shown in Figure 8. [20][24]

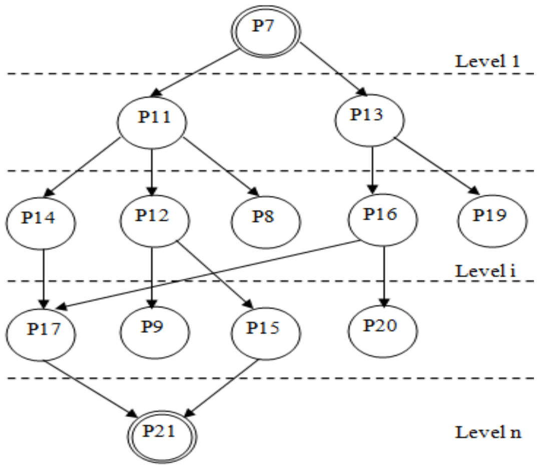

Figure 8:Leveled Graph

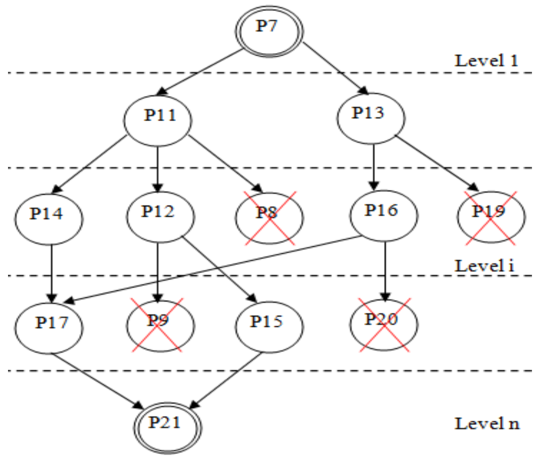

Figure 9: Delete Dead Node

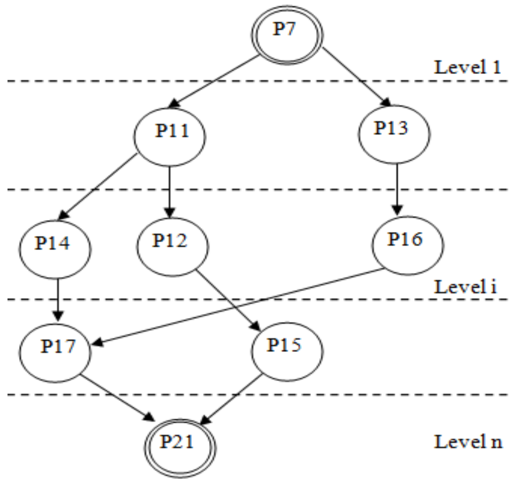

Figure 10: Simplest Graph

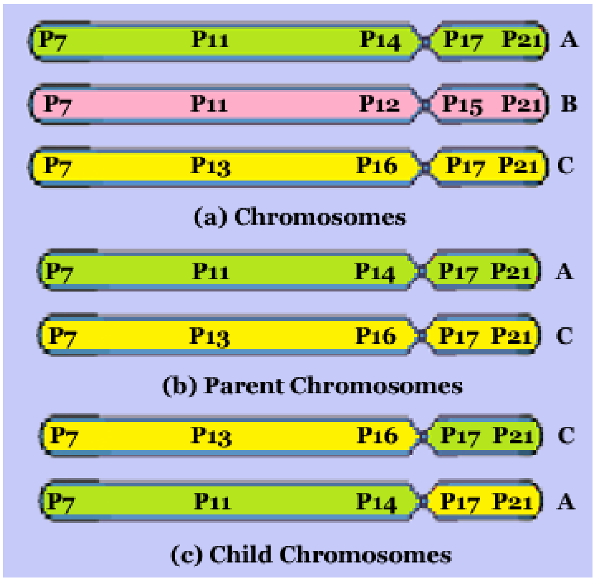

Figure 11: (a) Possible Shorter Paths (Chromosomes).
(b) Parent Chromosomes are Considered.
(c) Child Chromosomes are formed

The essence of the work is that it reduces the graph in levels (Figure 5 Leveled Graph) and the first level and the last level will contain only the source vertex and destination vertex respectively. As "Algorithm for creation of Simplified - Graph from Sub - Graph" gives birth to leveled Graph whose end leafs are destination node. Any traverse from first level (Source Node) to any of the leaf will be a path form source to destination, shown in Figure 10. The resultant is a reduced Leveled Graph which contains the shortest and the shorter paths in it. This reduces the number of vertex traversal increasing efficiency of the work done. The algorithm also produce dead nodes, those nodes need to be deleted. And condition for deletion of node is when a dead end is reached delete it, and also delete the sub node if it does not contain forwarding node in the next level shown in Figure-9. Every Traverse from First Level to last Level will produce the possible set of paths represented on the form of chromosomes shown in Figure 11 (a). Then parent Figure 11 (b) chromosomes are selected form the pool and mutate and similar child Figure 11 (c) are produced because Graph of 27 nodes has been reduced to 9 nodes in the simplified Graph.

5. Complexity

Considering the most worst case it can reach the destination vertex as many as n levels, and n is the maximum number of vertex in the graph, if and only if all vertexes lies in a straight line or in a series. The complexity runs up to O(n). If the solution space need only a set of m nodes then complexity runs up to O(m), where m<n. And in most favourable case we can reach to the destination in a single level distance (destination is the nearest node in a series).

Table 2: Comparison of the Proposed Algorithm with Dijkstra's Algorithm

Proposed Algorithm				**Dijksra'a Algorithm**	
Chromosome	**Path**	**Value**	**Discussion**	**Path**	**Value**
A	P7-P11-P14-P17-P21	128.1024	These are the possible feasible shorter paths from Source P7 to Destination P21 other than dijkstra's shortest path (Chromosome A)	P7-P11-P14-P17-P21	128.1024
B	P7-P11-P12-P15-P21	157.0156		Dijkstra's Algorithm only find the possible shortest path but do not find any other possible shorter path form source to destination so that it can be used in difficult or crisis situation.	
C	P7-P13-P16-P17-P21	157.0156			
Fusion of Chromosome shown in Figure : 11					
Parent1	P7-P11-P14-P17-P21	128.1024	Chromosome A		
Parent2	P7-P13-P16-P17-P21	157.0156	Chromosome C		
Child Chromosomes					
Child1	P7-P13-P16-P17-P21	157.0156	Chromosome C		
Child2	P7-P11-P14-P17-P21	128.1024	Chromosome A		

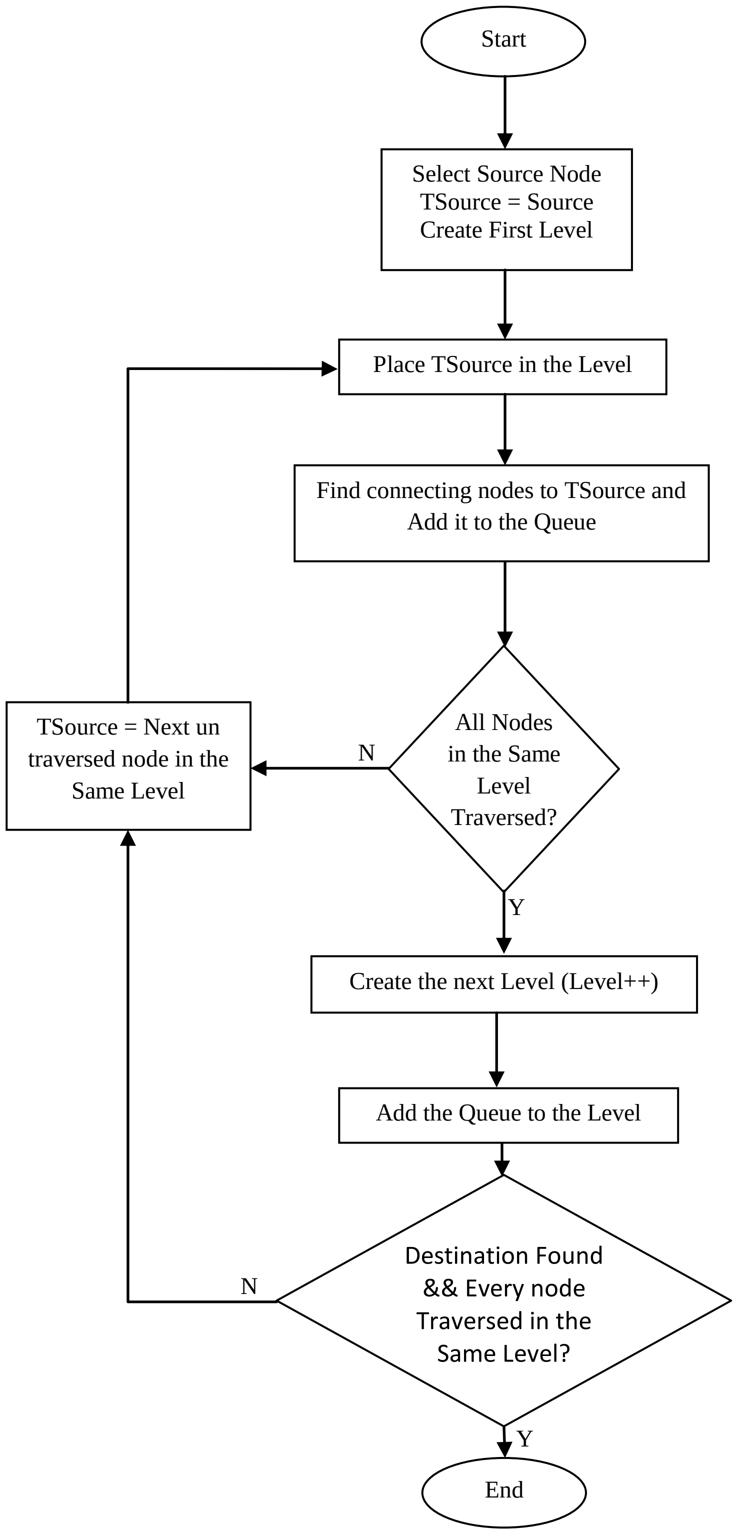
Start
Select Source Node
TSource = Source
Create First Level
Place TSource in the Level
Find connecting nodes to TSource and Add it to the Queue
All Nodes in the Same Level Traversed?
N
TSource = Next un traversed node in the Same Level
Y
Create the next Level (Level++)
Add the Queue to the Level
Destination Found && Every node Traversed in the Same Level?
N
Y
End

Figure 12: Flow Chart Representation of Creation of Simplest-Graph from Sub-Graph Algorithm.

6. Comparative Studies

Compression of the above example with Dijkstra's Algorithm is done in Table 2; we receive equivalent results with better prospects. The proposed algorithm produces the shortest path with other shorter paths, where as in Dijkstra's algorithm produces only the shortest path below, shown in Table 2.

[P7 – P11 – P14 – P17 – P21] → 128.1024

The above path is one of the elite chromosomes/individual provided by the proposed work. Dijkstra's algorithm provides only the shortest path, but do not provide any alternative shortest path to reach to the destination.

7. Conclusion and Future Work

The proposed work discusses the capability of "Search the Set of Shorter Path Algorithm" in finding the shortest and other possible alternative shorter path from source to destination. Every algorithm tries to work on the shortest path creating traffic on the nodes which will increase processing time on the nodes and the execution will be delayed and the time from source to destination will increase. As the algorithm also provides other possible shorter paths then the execution time through other shorter path will be less than the time taken by shortest path.

Decision on run time execution is not dealt with the proposed algorithm, can be treated as mutation on the path. In my forthcoming work I will be dealing this problem.

References

[1] C W Ahn, R S Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations", IEEE Transactions on Evolutionary Computation, vol. 6, no 6, pp. 566-579, Dec. 2002.
[2] Ismail Rakip Karas and Umit Atila , "A genetic algorithms approach for finding the shortest driving time on mobile devices", Scientific Research and Essays Vol.6(2), pp. 394-405, 18 Jan, 2011.
[3] Jean Berger, Mohamed Barkaoui, "A Parallel hybrid genetic algorithm for the vehicle routing problem with time windows", Defence Research and Development Canada Valcartier, Decision Support System Section, 2459 Pie – Blvd. Val – Belair, PQ, Canada G3J 1X5, Volume 31, Issue 12, Pages 2037 – 2053, October 2004.
[4] J. Holland, "Adaptation in Natural and Artificial System", The University of Michigan Press, Ann Arbor, 1975.
[5] Praveen Kumar, Varun Singh, "Advanced Traveler Information System for Hyderabad City", IEEE Transaction on Intelligent Transportation System, Vol. 6, no. 1, March 2005.
[6] Xu MH, Liu YQ, Huang QL, Zhang YX, Luan, "An improved Dijkstra's shortest path algorithm for sparse network", Appl. Math Computing 185: 247-254, GF -2007.
[7] David Beasley, David R Bull, Ralph R Martin "An overview of Genetic Algorithms: Part 2", Citeseerx, pp. 170 181, 1993, 15(4).
[8] Naghibi, "Application GIS in petroleum Industry", India Geospatial Forum, 7-9 February, 2012.
[9] V. Purushotham reddy, G. Michael, M. Umamaheshwari, "Coarse-Grained Parallel Genetic Algorithm to solve the Shortest Path Routing Problem using Genetic operators", Indian Journal of Computer Science and Engineering, vol.2 No. 1, pp 39-42, 2011.
[10] K. Krishnakumar and D. E. Goldberg, "Control System Optimization Using Genetic Algorithms", Journal of Guidance, Control and Dynamics, vol. 15, No. 3, pp. 735 – 740, 1992.
[11] Jose Maria A. Pangilinan and Gerrit K. Janssens, "Evolutionary Algorithms for the Multi-objective Shortest Path Problem", World Academy of science, Engineering and Technology International Journal of Mechanical, Vol. 1, No 1, pp. 13 – 18, 2007.
[12] Yinzhen Li , Ruichun He , Yaohuang Guo, "Faster Genetic Algorithm for Network Paths", The Sixth International Symposium on Operations Research and Its Application, Xinjiang, China, pp 380-389, Aug 8 – 12, 2006.
[13] Hayder A Mukhef, Ekhlas M Farhan and Mohammed R. Jassim, "Generalized Shortest Path Problem in Uncertain Environment Based on PSO", Journal of Computer Science, Vol. 4, April, 2008.
[14] Bilal Gonen, "Genetic Algorithm finding the shortest path in Networks", Proceedings of The 2011 World Congress in Computer Science, Computer Engineering and Applied Computing, 2011.
[15] Hui Cheng, Shenxiang Yang, "Genetic Algorithm with Elitism-based Immigrants for Dynamic Shortest Path Problem in Mobile Ad Hoc Networks", Evolutionary Computation, pp. 3135 – 3140, 18-21 May 2009.
[16] A. Varsek, T. Urbacic and B. Filipic, "Genetic Algorithms in Controller Design and Tuning",

IEEE Trans. Sys. Man and Cyber, Vol. 23. No. 5, pp 1330 – 1339, 1990.

[17] B. Porter and S. S. Mohamed, "Genetic Design of Multivariable Flight – Control System Using Eign-Structure Assignment", Proc. IEEE Conf. Aerospace Control System, 1993.

[18] Peter Folger, "Geospatial Information and Geographic Information System (GIS)", Congressional Research Service, June 8, 2009.

[19] "Improved Selection Operator for GA", Omar Al Jadaan, Lakishmi Rajamani, C. R. Rao, Journal of Theoretical & Applied Information Technology, Vol. 4 Issue 4, p269-277. 9p, 2008.

[20] Tarak Nath Paul, Abhoy Chand Mondal, Constitute a Sub-Graph with n-levels to Search a Set of Shorter Paths Using Genetic Algorithm, IEEE International Conference on Intelligent Computing and Intelligent System, 3-0096-10943, Volume 3, 452-456, November 2011.

[21] Gihan Nagib and Wahied G. Ali, "Network Routing Protocol using Genetic Algorithms", International Journal of Electrical & Computer Sciences, IJECS-IJENS Vol:10 No:02.

[22] Kruskal, "On the Shortest Spanning Sub – tree of a Graph and the Travelling Salesman Problem", jr. J.B. Proc. Amer. Math. Soc. 7, 48 – 50, 1956.

[23] Tarak Nath Paul, Abhoy Chand Mondal, "Search the Set of Shorter Paths Using Graph Reduction Technique", International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online):2277-7970) Volume-3 Number-4 Issue-13, page 184 December-2013.

[24] F. Benjamin Zhan, "Three Fastest Shortest Path Algorithms on Real Road Network", Journal of Geographic Information and Decision Analysis, vol.1, no.1, pp. 69-82, 1997.

Tarak Nath Paul born in 1984 Adra, Purulia, West Bengal, India pursuing his PhD. from The University of Burdwan, Burdwan, West Bengal. He has received his MCA (West Bengal University of Technology) and M. Tech in IT (Karnataka State Open University) with star marks and currently as Assistant Professor in Maharana Pratap Group of Institution (MPGI), CSE Department. He is a reviewer of IEEE Journals. His research area in Shortest Path with Genetic Algorithm, on which he has presented a paper in 2011 IEEE International Conference on Intelligent Computing and Intelligent System (ICIS-2011) at Guandong University, Guangzhou, China.

Abhoy Chand Mondal. He is the Head and Associate Professor of Dept. of Computer Science, The University of Burdwan, Burdwan, West Bengal. He was born in 27-2-1964. He received his B.Sc and M. Sc. Mathematics in 1987 and 1989 respectively. Then he completed his MCA from Jadavpur University in 1992. He received his PhD. Degree from Burdwan University in 2004. His research areas are Soft Computing, Document Processing, Web Mining etc. He has a year of Industry and 18 years of teaching experience. He published 30 Journal and 20 Conferences papers. Two students are awarded PhD. Degree and eight students are under his supervision.