ISSN 2063-5346

Triple Connected Dominator Coloring Sets using TUS and Backtracking Algorithm for MANET

R. Jothiraj[1*] and B. Chandralekha[2]

[1]*Department of Mathematics, Tagore Engineering College, Rathinamangalam 600127, Chennai, Tamilnadu, India.*

ORCID ID:0000-0003-1121-2896

[2]*Department of Physics, Tagore Engineering College, Rathinamangalam 600127, Chennai, Tamilnadu, India.*

ORCID ID:0009-0003-9156-007X

[*]*Email:jothirajlvp@gmail.com*

Abstract

Let $G(V,E)$ be a graph. A triple connected dominator coloring of a graph G is a proper three coloring of G in which every vertex dominates every vertex of atleast one color class. The minimum number of colors required for a triple connected dominator coloring set of G is called the triple connected dominator chromatic number of G and is denoted by $\chi_{tcc}(G)$. The concept of triple umpiring system (TUS) by considering the existence of path containing any three vertices (node) of a graph G using triple connected dominator coloring sets. In our system each node's behavior from source to destination is closely monitored by a set of three umpires. If any misbehavior is noticed umpires flag off the guilty node from the circuit. We have proposed backtracking algorithm to exemplify the basic TUS for salvaging, the circuit during disruptions in route reply and data forwarding phases. In this paper, we obtain bounds for standard graphs and characterize the corresponding special graphs.

Keywords: Triple connected dominator chromatic number, Triple connected domination number, TUS, Backtracking algorithm.

1. Introduction

Graph theory is one of the most developing branches of mathematics with wide applications to computer science. Graph Theory is applied in diverse areas such as social sciences, linguistics, physical sciences, communication engineering and others. Wireless Ad Hoc Networks (WAN) are becoming increasingly attractive for a variety of application areas, including security, hurricanes and a broad range of military scenarios. A mathematical based performance comparison of TUS and triple connected domination sets for MANET [1-3]. Detection mechanism is based on promiscuous hearing. Promiscuous hearing means listening to communications that is not meant for oneself. This is made possible by the wireless nature of the medium. Thus when node B sends packets to C, A along with B's neighbors observes the event. The behavioral deviation, if any, on the part of B, in the retransmission of the messages it received from A, can be readily observed by all of the listening nodes. In the data forwarding B's behavior will be supervised by A, while during route reply process node C will be the supervisor.

In TUS, three umpires decide the fate of B collectively. Two of the umpires are neighbors specifically commissioned for the role; the third umpire will be immediate predecessor node A, in its umpiring role. In our system each node's behavior from source to destination is closely monitored by a set of three umpires. If any misbehavior is noticed umpires flag off the guilty node from the circuit [4-5]. We have proposed strong connected domination set to exemplify the basic TUS for salvaging, the circuit during disruptions in route reply and data forwarding phases.

The concept of triple connected graphs and triple connected dominator coloring was introduced [6-8]. Considering the path of the three vertices of G. They have read three attributes of attached drawings, which have made many decisions. In this paper, we use this idea to develop the concept of TUS with strong triple connected dominating set and strong triple connected domination number of a graph.

The rest of the paper is organized as follows: Section 2 discusses network model and mathematical assumptions. Section 3 discusses Triple connected dominator coloring using Backtracking algorithm and Section 4 gives the conclusions.

2. Network Model and Mathematical Assumptions

In this section, we formulate the wireless mobile ad hoc network and triple connected assumptions. A mathematical based performance comparison of TUS and triple connected domination sets for MANET and as shown in the Fig. 1.

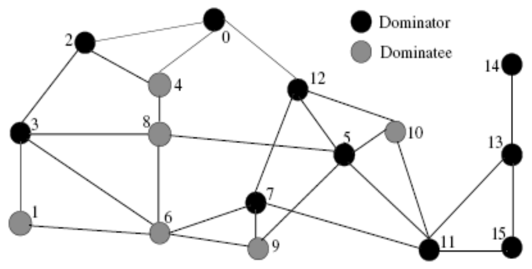


Fig. 1. Relation between dominator and dominate of network

3. Backtracking solution using Triple connected dominator coloring problem

Problems with coloring the vertices of a general graph are subject to the condition that no two adjacent vertices have the same color. This problem is called the graph coloring problem [9]. For the graph-coloring problem, we are interested in assigning colors to the vertices of an undirected graph, with the constraint that no two adjacent vertices are assigned the same color. The optimized version colors the map using minimal colors. A version of the decision, called k coloring, asks whether a graph is colorable using most k colors. The 3-color problem is a special case of the k-color problem $k = 3$. The 3-coloring problem is known to be $NP-$complete, meaning that there is no known polynomial time algorithm to solve it. Using backtracking does not provide an efficient algorithm either, but this is a good example to illustrate the technique.

3.1. Triple connected dominator coloring using Backtracking

The graph coloring problem is solved by backtracking .Backtracking is defined below and how it solves this problem [10].

3.2. Backtracking

Suppose you have to make a series of decisions among various choices

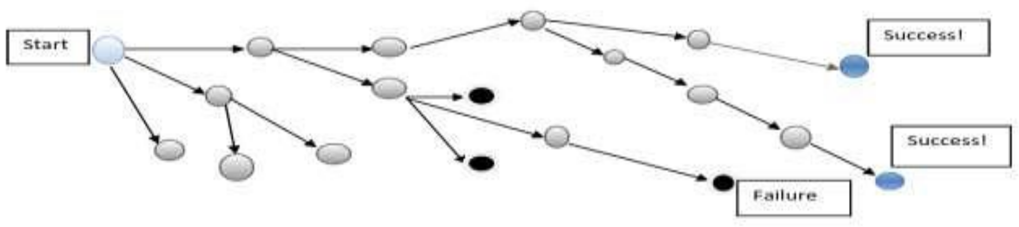
You don't have enough information to know which one to choose

Each decision leads to a new set of choices

A few choices (more than one) may be the solution to your problem

Backtracking is a systematic way of trying different sequence results until you find one

It “works”[11].



A problem space consists of states (nodes) and actions (paths leading to new states). A node can only see routes to connected nodes. If a node only fails, revert to its "parent" node. Try other alternatives. If all this leads to failure, further retreat may be required.

3.3. Backtracking is a state space search:

Regression is a systematic way of searching the solution space for a problem. In regression, we start by defining the solution space for the problem. At this point the problem must have at least one (optimal) solution. Generally, the solution is viewed as a sequence of stages with some decision to be made at each stage. In the maze traversal problem, we can define the solution space to include all paths from the entrance to the exit, which are then generated by a depth-first or breadth-first search). The resulting search tree is called a state-space (search) tree or backward search tree [12].

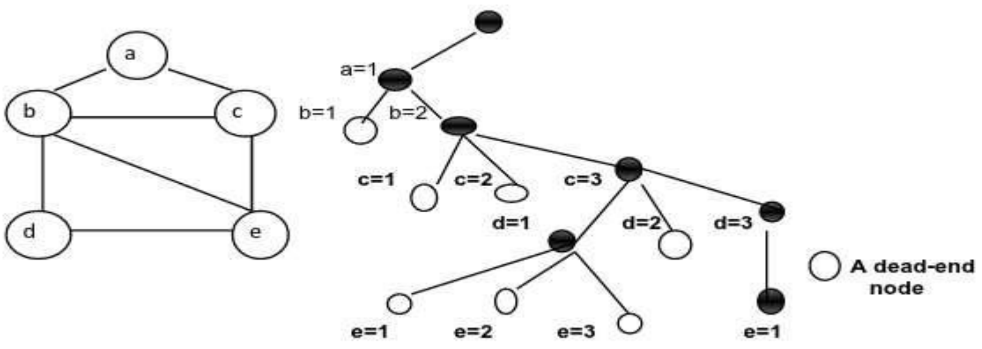


Fig. 2. Using backtracking to solve the 3-color problem. The state-space search tree is shown on the right.

A backtracking search tree starts at an unlabeled root. The immediate children of the root represent the available choices regarding the coloring of the first vertex (vertex a in this case) of the graph. However, tree nodes are created only if needed. The first (leftmost) child of the source represents the first choice of vertex color, which is to color the vertex with color 1as indicated by the label assigned to that vertex. So far, this partial coloring does not violate the constraint that adjacent vertices must be assigned different colors. So we grow the tree one more level where we try to assign a color to the second vertex of the graph. The left end represents the vertex color with color 1. However, such partial coloring $a=1$and $b=1$is not valid due to vertices $a$and b adjacencies. Also, there is no point in growing the tree further from this node, no matter what colors we assign to the remaining nodes. We will not get the correct color for all vertices. So we prune (drop) this path and consider the nearest alternative path [13]. We try the next choice for the color of vertex b, which is to paint it with color 2. The coloring $a=1$and $b=2$ is the correct partial color, so we continue to the next level of the search tree.

When we reach the lowest level and try to color vertex e, we find that it is not possible to color it any of the three colors; therefore, we backtrack to the previous level and try the next choice for the coloring of vertex d. We see that the coloring of vertex d with color 2 leads to a violation (d and b are successively the same color). We continue $d=3$ and then go down to $e=1$, which gives the solution, $\{a=1, b=2, c=3, d=3, e=1\}$. A routine implementation of backtracking for graph k-coloring will generate all possible coloring solution vectors. For example, the output of our program in Fig. 2 for the graph above shows that there are 6 solutions.

```
solution found: 1 2 3 3 1
solution found: 1 3 2 2 1
solution found: 2 1 3 3 2
solution found: 2 3 1 1 2
solution found: 3 1 2 2 3
solution found: 3 2 1 1 3
```

Solution 1 corresponds to that shown in Fig. 2. Solutions 3 & 4 are derived from this Solutions 1 & 2 are obtained by interchanging colors 1 and 2. Solutions 5 & 6 are obtained from solutions 1 & 2 by interchanging colors 1 & 3. This type of output is expected because given a coloring solution vector, we can choose two colors and transform them into the solution vector; The result will be a different color

3.4. Running-time analysis of backtracking for the k-coloring problem:

In general, the solution to a $k-$coloring of an $n-$ vertex graph can be represented as an array $C[1,2,....n]$ where $C[i]$ is the color assigned to vertex $i.$. By a simple counting argument, there are a total of $k \times k \times .... \times k) = k_n$ vectors.

In the worst case, backtracking may end up generating a complete search tree. Because such a tree has k_n leaves, an ordinary backtracking algorithm for the k-coloring problem will be Ω(kn). In the worst case, regression will end up producing an exhaustive search tree. Since such a tree has k_n leaves, A typical backtracking algorithm for the $k-$color problem is $\Omega(k_n)$ [14]**.**

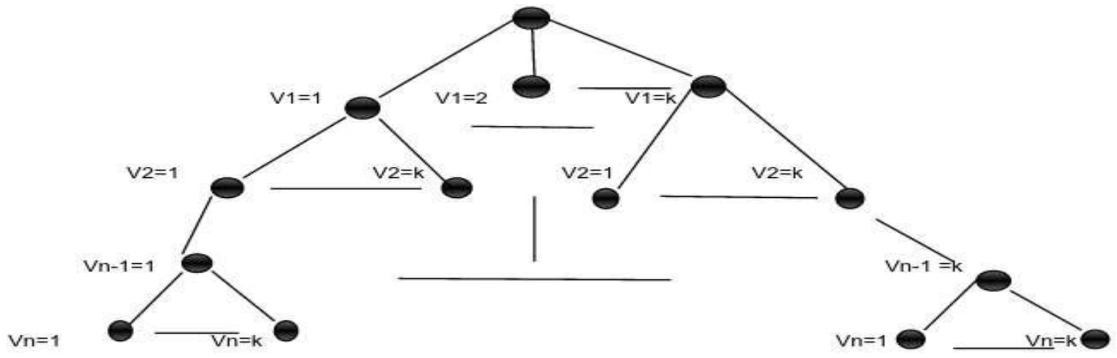


Fig .3. The complete backtracking search tree for k-coloring of an n-vertex graph has k_n leaves.

There are some ideas to reduce the size of the search tree, but these ideas are not sufficient to reduce the exponential size to polynomial size. For example, a careful examination of the $k-$coloring problem reveals that if there is no $k-$coloring with vertex 1, then no other color is assigned to vertex 1. This is because if vertex 1 has a k-coloring that uses some color c, then by replacing color 1 with color c, we get a $k-$coloring that uses vertex 1 color. Using this observation, it is enough to make a part of the complete tree; The descent area from $v=1.$ The size of the generated region is $1/k$ of the size of the complete tree. In this case, the regression search tree may still have $k_n - 1$ leaves, which is still exponential in n **[**15]**.**

3.5. Backtracking Algorithm:

Next, let's examine the pseudocode for implementing regression. A regression algorithm is best organized using two methods: one method of constructing a search tree and another method of testing the validity of a partial solution. In a general setting, these methods are, respectively, Gen Solutions () and Valid Solution (); or given special names appropriate to the problem. It can be seen that the nodes (part) of the search tree correspond to the solution vectors; Therefore, they can be created through the process of slot filling. Slot filling can be repetitive or iterative. In either case, it must satisfy three types of motions to construct and explore the search tree: [16]

• Move across a level to try different choices for coloring a particular vertex.
• Move down from the current tree level.
• Move up from the current tree level to the level above. This is regression.

The backtracking-based algorithm for building a search tree (Listing 1) can be encoded as a recursive method graph Color(), or its equivalent function method, graphColorId(), with the rest of the program code (Listing 2). Note that both methods do not terminate when the first solution is found; Instead, the search for all possible solutions continues. We should note that generating all possible solutions allows us to solve the optimization version of the problem. To end the search when the first solution is found, we use a boolean global variable sf-flag that is set when a solution is found. Lines in bold indicate the correct place to check the sf-flag and end the search once the first solution is found. Another important and integral method for the regression algorithm is the valid solution () method.

This method tests whether a partial solution satisfies the constraints of the problem. The significance of this method is two-fold. First, it implements a pruning strategy, and generally, it is better to prune early. Second, since this method is called for each node created in the search tree, its performance regression will have a direct impact on the overall performance of the algorithm. For the problem at hand, as shown in Listing 2, this pattern is given by the valid color (C, m).Because this method is already called with the reference vector $C[1,2...m-1]$. For a valid region color, it is sufficient to add the mth entry and its color $C[m]$ will lead to violation. Hence, it suffices to examine whether a vertex in $C[x] = C[m][1,2...m-1]$ has a vertex m near x. We can easily see that a valid color is $O(n)$; Therefore, our regression algorithm for the k-coloring problem on a graph with n vertices is $O(nk_n)$.

Observation 3.1 The backtracking-based algorithm (Listing 1) for generating the search tree can be encoded as a recursive method Graph Color (), or the equivalent iterative method, Graph Color IT(), with the rest of the program code (Listing 2).Note that both of these methods do not end when the first solution is found; rather, the search continues to find all possible solutions. We should note that generating all possible solutions allows for solving the optimization version of a problem. Every triple connected dominator coloring set is a TUS. During data forwarding phase if node N_i misbehaves i.e. it drops packets, it is observed by nodes N_{i-1} , U_{i-1}, and U_{i+1} and they send M-Flag messages, and convict the guilty node; further they unicast messages among themselves to establish an alternative path via N_{i-1} , U_i, U_{i+1} and N_{i+1}. In this segment Self_USS will operational as shown in the Fig. 4. On the other hand if node N_i goes out of communication link but the umpiring nodes do not receive the hello messages from N_i and simply switch over to alternative path, thus booking of innocent node is avoided. Thirdly umpire U_i may go out of communication link with U_{i-1}, N_{i-1}, N_i, and U_{i+1} .In such a case N_{i-1} monitors N_i and N_i monitors N_{i+1} under Self_USS.

Observation 3.2. Every triple connected dominator coloring set is a TUS. If node N_i misbehaves, and say announces a wrong sequence number. This is immediately observed by the node N_{i+1} which sets the status bit of N_i to red and starts RREP salvaging operation as shown in the Fig. 4. On the other hand if N_i simply loses communication link with N_{i+1}, N_i is not booked; only route reply salvaging is undertaken.

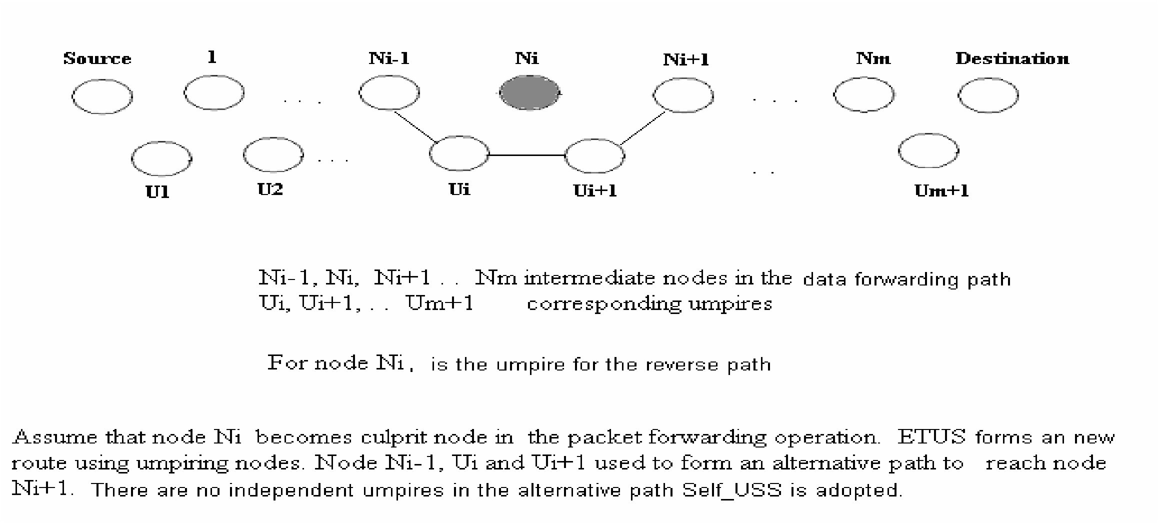

Source
1
Ni-1
Ni
Ni+1
Nm
Destination
U1
U2
Ui
Ui+1
Um+1
Ni-1, Ni, Ni+1 . . Nm intermediate nodes in the data forwarding path
Ui, Ui+1, . . Um+1 corresponding umpires
For node Ni , is the umpire for the reverse path
Assume that node Ni becomes culprit node in the packet forwarding operation. ETUS forms an new route using umpiring nodes. Node Ni-1, Ui and Ui+1 used to form an alternative path to reach node Ni+1. There are no independent umpires in the alternative path Self_USS is adopted.

Fig. 4: Self_USS.

Observation 3.3 Every triple connected dominator coloring set is a TUS and every triple connected dominating set is a dominating set. During data forwarding phase if node N_i misbehaves i.e. it drops packets, it is observed by nodes N_{i-1} , U_{i-1}, and U_{i+1} and they send M-Flag messages, and convict the guilty node; further they unicast messages among themselves to establish an alternative path via N_{i-1} , U_i, U_{i+1} and N_{i+1}. In this segment Self_USS will operational as shown in the Fig. 4. On the other hand if node N_i goes out of communication link but the umpiring nodes do not receive the hello messages from N_i and simply switch over to alternative path, thus booking of innocent node is avoided. Thirdly umpire U_i may go out of communication link with U_{i-1}, N_{i-1}, N_i, and U_{i+1} .In such a case N_{i-1} monitors N_i and N_i monitors N_{i+1} under Self_USS.

The ad-hoc networking environment have been proposed, but little performance information on each protocol and node tailed performance comparison between the protocols has previously been available.

Exact value for complete standard graphs

1) For the graph $K_n, n \geq 3, \gamma_{tc}(G) \geq 3$ and $\chi_{tcc}(G) = n.$

2) For the graph $K_n, n \geq 3,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) \geq 2n..$

3) For the graph $K_3, \gamma_{tc}(G) = 3$ and $\chi_{tcc}(G) = 3,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) = 6.$

4) For the graph $K_n,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) = 2n$ for $n = 3.$

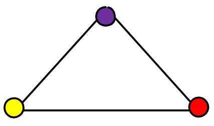
5) For the graph $K_n,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) = 2n - i$ for $i = n - 3$ and $n \geq 4.$

6) For the graph $K_4,$ $\gamma_{tc}(G) = 3$ and $\chi_{tcc}(G) = 4,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) = 7.$

Theorem 3.1 For the graph $K_n,$ We have $\gamma_{tc}(G) + \chi_{tcc}(G) \leq 2n - 1$ for $n \geq 4.$

Theorem 3.2 For any connected graph $G,$ $\gamma_{tc}(G) + \chi_{tcc}(G) = 2n$ iff G is isomorphic to $K_3.$

Proof: $\gamma_{tc}(G)+\chi_{tcc}(G)\le n+1+\Delta=n+1+\delta=n+1+n-1=2n$ if $\delta=\Delta,$ and we have $\gamma_{tc}(G)=n$ and $\chi_{tcc}(G)=n,$ then G is a complete graph on n vertices , Since $\gamma_{tc}(G)=3$ we have $n=3$ and $\chi_{tcc}(G)=3.$ Hence G is isomorphic to $K_3.$



Theorem 3.3 For any connected graph $G,$ $\gamma_{tc}(G)+\chi_{tcc}(G)=2n-1.$ If G is isomorphic to either K_4 or $C_4.$

Proof: Let $\gamma_{tc}(G)+\chi_{tcc}(G)=2n-1.$ Then there are two cases to consider $\gamma_{tc}(G)=n-1$ and $\chi_{tcc}(G)=n.$ Then G is a complete graph on n vertices, and since $\chi_{tcc}(K_n)=n,$ we have $n=4$ and $n=3.$ Hence G is isomorphic to $K_4.$

Exact value for some special graphs:

1) The Wagner graph is a 3-regular graph with 8 vertices and 12 edges given in Fig. 5.

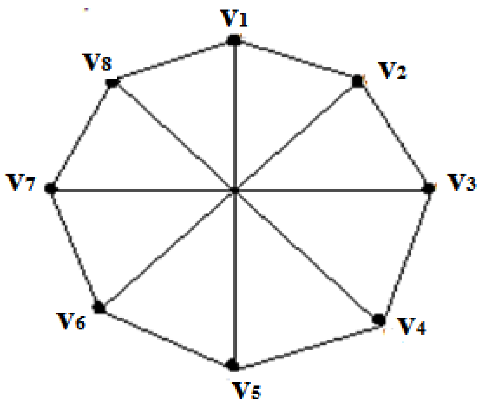


Fig. 5

For the Wagner graph $G,$ triple connected dominator chromatic number $\chi_{tcc}(G)=3,$ and also $\gamma_{tc}(G)=3.$ Here $S=\{v_6,v_7,v_8\}$ is a minimum dominating set.

2) The Tietze graph is a 3-regular graph with 12 vertices and 18 edges given in Fig. 6.

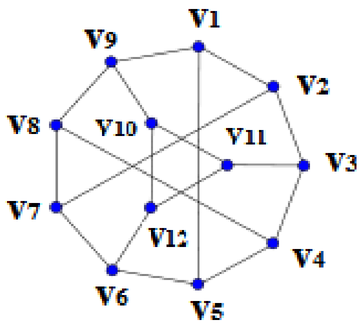


Fig. 6

For any Tietze graph G, triple connected dominator chromatic number $\chi_{tcc}(G) = 3$, and also $\gamma_{tc}(G) = 3$. Here $S = \{v_3, v_6, v_9\}$ is a minimum dominating set.

3) The Petersen graph is a 3-regular graph with 10 vertices and 14 edges given in Fig, 7.

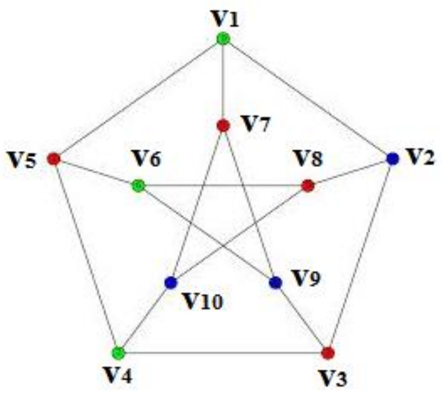


Fig. 7

For any Petersen graph G, triple connected dominator chromatic number $\chi_{tcc}(G) = 3$, and also $\gamma_{tc}(G) = 3$. Here $S = \{v_1, v_4, v_6\}$ is a minimum dominating set.

4) The Bidiakis cube graph is a 3-regular graph with 10 vertices and 14 edges given in Fig. 8.

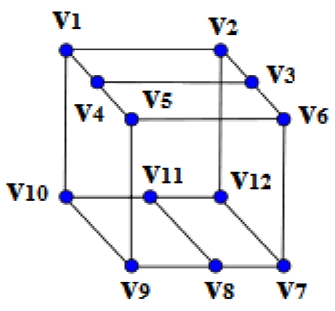


Fig. 8

For any Bidiakis cube graph G, triple connected dominator chromatic number $\chi_{tcc}(G) = 3$, and also $\gamma_{tc}(G) = 4$. Here $S = \{v_1, v_6, v_9, v_{12}\}$ is a minimum dominating set.

4. Conclusion

The area of ad-hoc networking has attracted increasing attention among researchers in recent years, as existing wireless networking and mobile computing hardware platforms are capable of supporting the promise of this technology. Various new routing protocols targeting the ad-hoc networking environment have been proposed, but little performance information on each protocol and node tail performance comparison between protocols was previously available. We studied the triple connected dominator color set problem on special graphs with only bidirectional connections (CGB). The graphs can be used to model wireless ad hoc networks where nodes have different transmission ranges. TUS is to create a maximally independent set and then merge them. Graph color and dominance play an important role in many real-world applications and is still the subject of exciting research.

References

1. R. Jothiraj and B. Chandralekha, "Mathematical Based Performance Comparison of TUS and Triple Connected Dominator Coloring Sets for MANET", European Chemical Bulletin, Vol.12, Issue 4, 2023, pp 3191-3197.
2. R. Jothiraj, "A Strong Triple Connected Domination Set Based on Routing Scheme for MANET", Journal of Huazhong University of Science and Technology, Vol 50, Issue 5, 2021 pp.1-11.
3. R. Jothiraj and A. Kathirvel, "A Mathematical Based Performance Comparison of TUS and Strong Triple Connected Domination Sets for MANET", International Journal on Recent Researches in Science, Engineering & Technology, vol.4,Issue 2, Feb. 2016.
4. A.Kathirvel, and Dr. R. Srinivasan, "ETUS: An Enhanced Triple Umpiring System for Security and Performance Improvement of Mobile Ad Hoc Networks", International Journal of Network Management, John Wiley & Sons, Vol. 21, No. 5, September/October 2011, pp. 341 - 359.
5. A.Kathirvel, and Dr. R. Srinivasan, "ETUS: An Enhanced Triple Umpiring System for Security and Robustness of Mobile Ad Hoc Networks", International Journal of Communication Networks and Distributed Systems, Inderscience, Vol. 7, Nos.1 / 2, 2011, pp. 153 – 187.
6. Paulraj Joseph J., Angel Jebitha M.K., Chithra Devi P. and Sudhana G,Triple connected graphs, Indian Journal of Mathematics and Mathematical Sciences,Vol. 8, No.1, (2012), pp. 61-75.
7. Jothiraj R, Jayakumar S and Venugopal P ,"Triple Connected Dominator Coloring to Cyclic Open Shop Scheduling of Unit Circular- Arc Graph", International Journal of Research and Analytical Reviews,Vol. 6,Issue 1, (2019),pp. 579-583.
8. Jothiraj R, Jayakumar S and Venugopal P," Cyclic Open Shop Scheduling in a Paired Triple Connected Dominator Coloring of Circular- Arc Graph" Journal of Applied Science and Computations,Vol.7, Issue 1,(2018), pp. 675-680.
9. Bondy, 1. A., Bounds for the chromatic number of a graph, Journal of Combinatorial Theory, 7 (1969), pp. 96-8.
10. Broder, S., Final examination scheduling, Communications of the ACM, Vol. 7,

No.8 (1964), pp. 494--8.

11. Brooks, R. L. , On coloring the nodes of a network, Proceedings of the Cambridge Philosophical Society, 37 (1941), p. 194.
12. Brown,1. R. , Chromatic scheduling and the chromatic number problem, Management Science, Vol. 19, No.4 (1972), pp. 456- 463.
13. Sharmila S., Jeyanthi Rebecca L., Saduzzaman M., "Biodegradation of domestic effluent using different solvent extracts of Murrayakoenigii", Journal of Chemical and Pharmaceutical Research, ISSN : 0975 – 7384, 5(2) (2013) PP.279-282.
14. Christofides, N., Graph Theory -An Algorithmic Approach, (Academic Press, New York, 1975), pp. 58-78.
15. Garey, M. R., and Johnson, D. S., The Complexity of Near-optimal Graph Coloring, Journal of the ACM, Vol. 23, No.1 (Jan. 1976), pp. 43-9.
16. Hall, A. D., and Acton, F. S., Scheduling University Course Examinations by Computer Communications of the ACM, Vol. 10, No.4 (April 1967), pp. 235-8.