International Journal of All Research Education and Scientific Methods (IJARESM), ISSN: 2455-6211
Volume 9, Issue 9, September -2021, Impact Factor: 7.429, Available online at: www.ijaresm.com

Identifying Micro-Containment Zone In A Infected Community For Partial Lockdown An Emotionally Intelligent Combinatorial Algorithm

Saptarshi Naskar[1], Tanmoy Biswas[2], Arup Kr. Goswami[3]

[1]Assistant Professor, Department of Computer Science, Sarsuna College (Affiliated to University of Calcutta), West Bengal, India
[2]Assistant Professor, Department of Computer Science, Syamaprasad College (Affiliated to University of Calcutta), West Bengal, India
[3]State Aided College Teacher, Department of Computer Science, Vidyasagar College (Affiliated to University of Calcutta), West Bengal, India

----------**********----------

ABSTRACT

As per the research in the field of Neurosciences, our Subconscious mind actually controls our emergency motor neurons for smooth functioning our most vital muscles such as Heart, Lungs, Kidneys, Brain and other internal and external muscles to live us bypassing conscious logical decisions making memory, the conscious memory, and that is why our subconscious memory functions the most important role to live us and to deal with internal and external problems. This subconscious mind controls our emotions. That is why the emotionally intelligent human can take better decisions than a machine. The aim of this paper is to make man machine interaction more efficient by using emotionally intelligent combinatorial algorithms. A community of a country is consists of people. The whole community can be mapped into a graph network. Every distinct individual can be assumed as a node of the constructed graph for the community. To declare a pandemic situation for any country for that we first need to check whether the infection is spread throughout the Community or not. For this purpose we need to calculate the existence of cliques. Our aim is to find all these cliques, whose presence in the community graph network actually identifies the micro contentment zones in the network. Now putting the zones into lockdown instead of putting whole community into lockdown actually saves the community from economic disasters.

***Keywords*: Emotional Intelligence (EI); Clique; Combinatorial Algorithm; 3-SAT; Micro-contentment Zone.**

I. Introduction

Definition 1: *Emotional Intelligence (EI)*
From medical Science point of view, the capacity to be aware of, control and express one's emotions and to handle interpersonal relationships judiciously and empathically.

Definition 2: *Emotionally Intelligent Algorithm*
A Combinatorial Algorithm, that deals with intractable problems by mapping the original problems into suitable tractable problems as human intelligent brains do naturally.

Lemma: *Clique is an NP Complete problem*

Proof: It is easy to verify that a graph has a clique of size k if we guess the vertices forming the clique. We merely examine the edges. This can be done in polynomial time.

We shall now reduce 3-SAT to CLIQUE. We are given a set of k clauses and must build a graph which has a clique if and only if the clauses are satisfiable. The literals from the clauses become the graph's vertices. And collections of true literals shall make up the clique in the graph we build. Then a truth assignment which makes at least one literal true per clause will force a clique of size k to appear in the graph. And, if no truth assignment satisfies all of the clauses, there will not be a clique of size k in the graph.

To do this, let every literal in every clause be a vertex of the graph we are building. We wish to be able to connect true literals, but not two from the same clause. And two which are complements cannot both be true at once. So, connect all of the literals which are not in the same clause and are not complements of each other. We are building the graph G = (V, E) where:

V = {<x, i> | x is in the i^{th} clause} E = {(<x, i>,<y, j>) | x ¹ and i ¹ j}

Now we shall claim that if there were k clauses and there is some truth assignment to the variables which satisfies them, then there is a clique of size k in our graph. If the clauses are satisfiable then one literal from each clause is true. That is the clique - why? This is because a collection of literals (one from each clause) which are all true cannot contain a literal and its complement. And they are all connected by edges because we connected literals not in the same clause (except for complements).

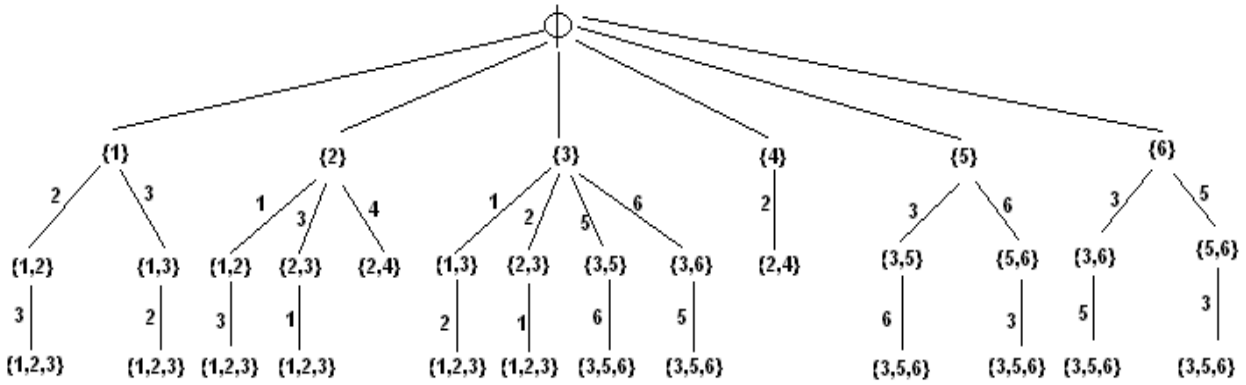

Fig. 1. Figure shows the possible cliques and the vertex sequences generated by the Algorithm proposed

On the other hand, suppose that there is a clique of size k in the graph. These k vertices must have come from different clauses since no two literals from the same clause are connected. And, no literal and its complement are in the clique, so setting the truth assignment to make the literals in the clique true provides satisfaction.

A small inspection reveals that the above transformation can indeed be carried out in polynomial time. (The degree will again be left as an exercise.) Thus the CLIQUE problem has been shown to be NP-hard just as we wished. One of the neat things about graph problems is that asking a question about a graph is often equivalent to asking quite a different one about the graph's complement. Such is the case for the clique problem. Consider the next problem which inquires as to how many vertices must be in any set which is connected to or *covers* all of the edges.

REVIEWING THE EXISTING ALGORITHMS:
We have compared our proposed algorithm with that of traditional backtracking algorithm. The inputs and outputs of the algorithm is as follows.

As a starting point, let us look at the unconstrained backtrack algorithm, a search in which no attempt is made to prune the search tree. Each node in the search tree corresponds to a vertex of the graph. A child of a given node C is obtained by adding to C a vertex x which does not belong to the set C that is adjacent to every vertex in C. The edge from C to child C ∪{x} corresponds to the vertex x. A sample graph is shown below. It is to be noted that each clique is generated many times. To eliminate the problem of duplicate cliques, the following two theorems were proposed.

THEOREM 1: *Let S be a node in the search tree T and let the first son of S in T to be explored be S ∪{x}. Suppose that all the sub-trees of the nodes S ∪{x} in T have been explored, so that all the cliques containing S ∪{x} have been generated. Then among the sons S ∪{v}, only those for which v don't belong to Adj(x) need be explored.*

THEOREM 2: *Let S be a node in the search tree T and let S′⊂S be a proper ancestor of S in T. If all sub-trees of the node S′∪{x} have been explored, so that all cliques containing S′∪{x} have been generated, then all unexplored sub-trees with roots S ∪can be ignored.*

International Journal of All Research Education and Scientific Methods (IJARESM), ISSN: 2455-6211
Volume 9, Issue 9, September -2021, Impact Factor: 7.429, Available online at: www.ijaresm.com

WORK OUT OF THE PROPOSED ALGORITHM:

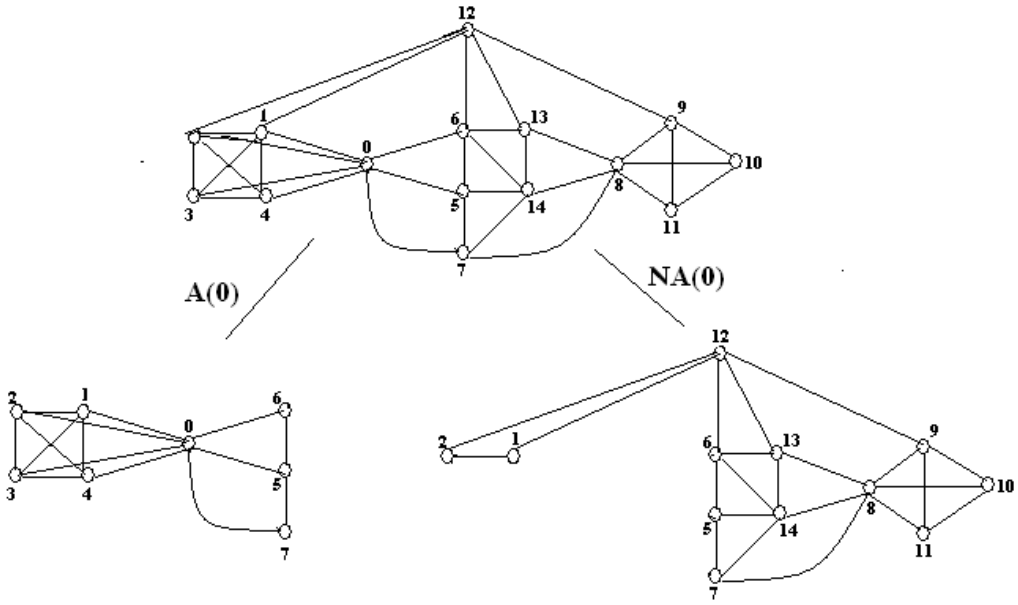

Fig. 2.worked out example of the Algorithm how the cliques are generating

```
S ← ϕ
CLIQUE (V,ϕ)
procedure CLIQUE (N,D)

if N ∪ D=ϕ then output S, which is a clique
else
{
    if N ≠ ϕ then
    {
        f ← vertex in N
        EXPLORE (f)
        while N ∩ (V – Adj(f) ≠ ϕ do
```

```
        {
            v ← vertex in N ∩ (V – Adj(f))
            EXPLORE (v)
        }
    }
}
Return

Procedure EXPLORE(u)
N ← N – {u}
S ← S ∪ {u}
CLIQUE (N ∩ Adj(u), D ∩ Adj(u))
S ← S – {u}
D ← D ∪ {u}
Return
```

International Journal of All Research Education and Scientific Methods (IJARESM), ISSN: 2455-6211
Volume 9, Issue 9, September -2021, Impact Factor: 7.429, Available online at: www.ijaresm.com

The Proposed Algorithm

INPUT DOMAIN FOR THE ALGORITHM: The proposed algorithm operates accurately for the input domain of all connected, undirected, simple, finite graphs. All definitions are given in the previous sections.

Given a graph or sub graph G as input. This algorithm partitions G, if possible, into independent sub graph. The algorithm when repeatedly applied for individual partition will generate cliques of G.

INPUT: *Adjacency structure of the given graph or sub graph G.*

OUTPUT: *Independent sub graphs of G which ultimately will generate cliques of the Graph.*

```
Procedure DIVIDE_GRAPH (A)
V= ϕ
while (true)
{
        if the vertex set contains n-1 degree for all elements,
                Output A Clique.
        Else find the maximum degree vertex and find the Adj and Nadj set of that vertex.
                DETECT_CLIQUE (Adj, Nadj)
}
Return

Procedure DETECT_CLIQUE (Adj', NAdj')
Find the number of vertices in the subgraph N(say).
If all the vertices has N-1 degree
{
        Then output the set as a Clique
        Return
}
else
{
        DIVIDE_GRAPH (NAdj') alongwith the vertex set that are Adjacent to all the elements of
NAdj'
}
```

- *A is used as the set of vertices for the source graph.*
- *Adj and Nadj are the set of vertices that are adjacent and non adjacent to the maximum degree node.*
- *The function DETECT_CLIQUE is used to output the cliques that are generated at the leaf of the search tree. The function DIVIDE_GRAPH is used to output the cliques in the intermediate branches of the search tree.*

Mechanism Of The Algorithm With An Example:

The graphs are divided into two sub graphs in each phase depending upon the adjacent and non adjacent vertex set of the highest degree vertex in the graph.

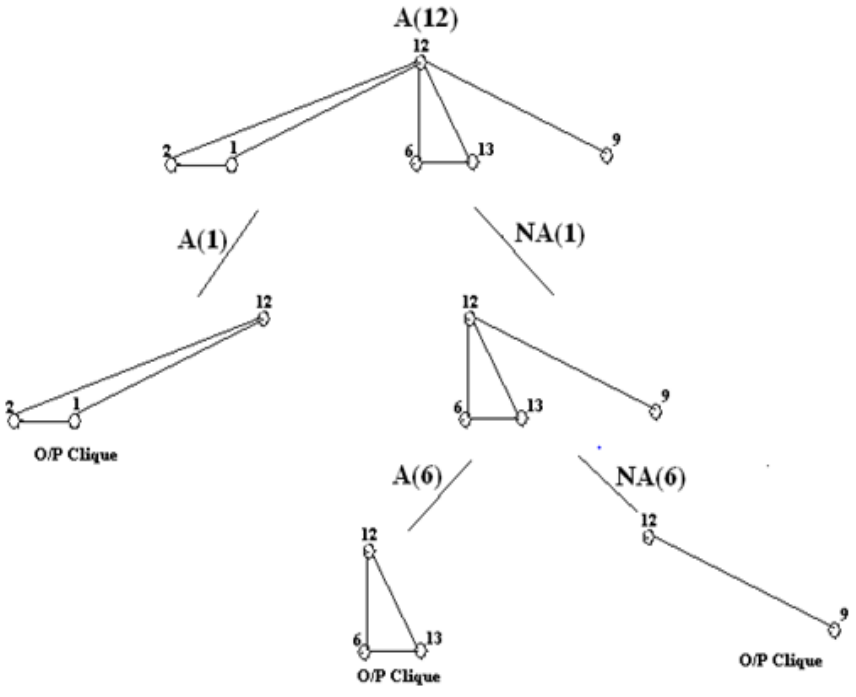

Fig. 3.worked out example of the Algorithm how the cliques are generating

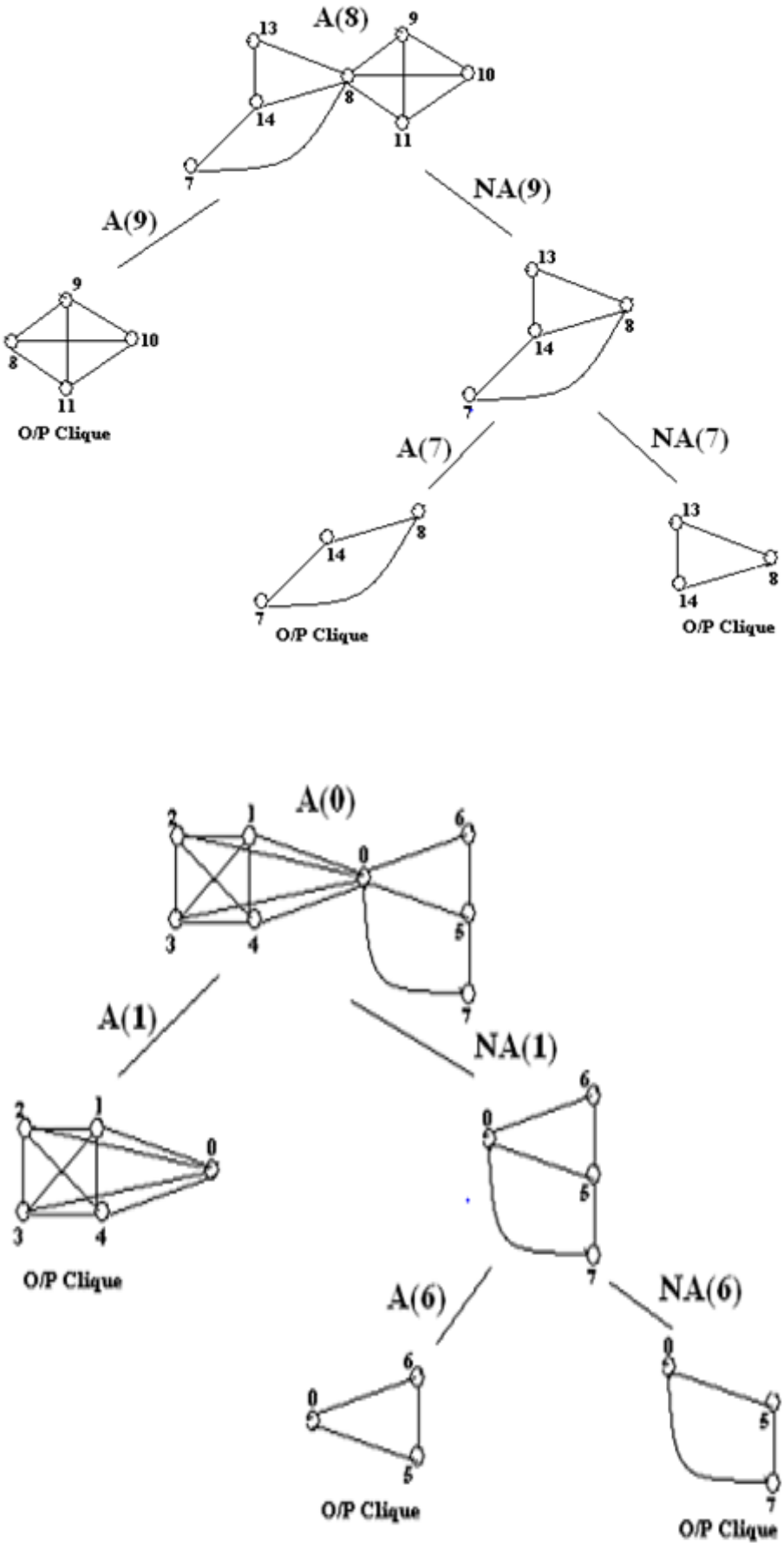

Fig. 4.worked out example of the Algorithm how the cliques are generating

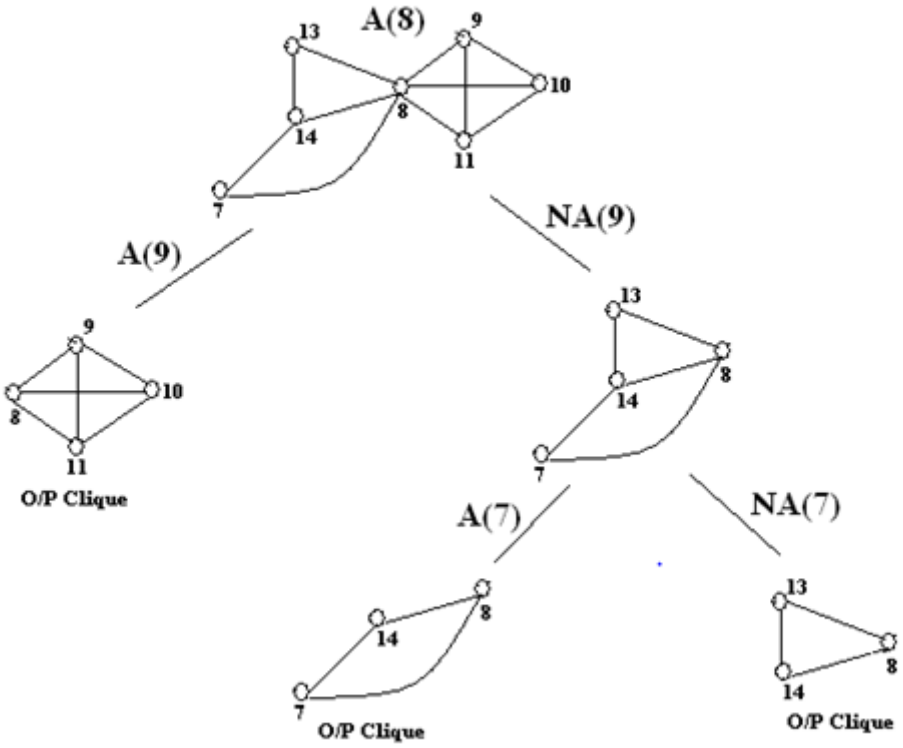

Fig. 5.worked out example of the Algorithm how the cliques are generating

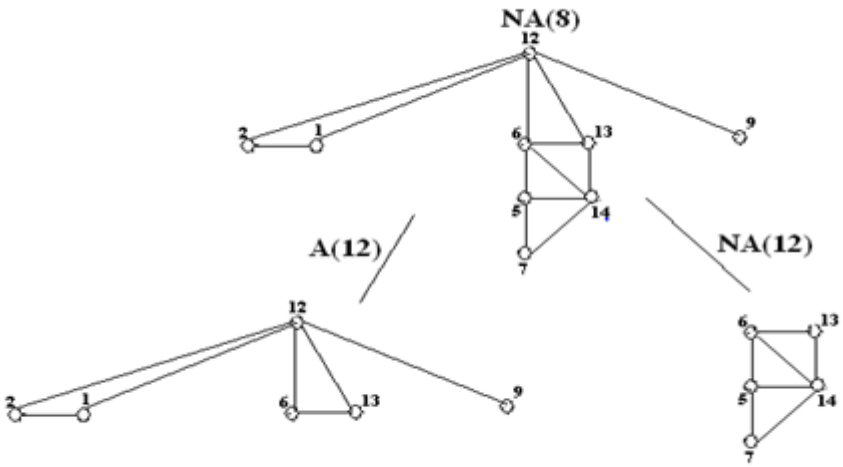

Fig. 6.worked out example of the Algorithm how the cliques are generating

Figures 2,3,4,5,6 are showing all the cliques being generated step by step using the proposed Algorithm

Now finally the cliques thus generated are as follows: {14,15,7}, {6,14,5}, {13,6,14}, {12,9}, {1,12,2}, {13,12,6},{8,14,13}, {7,8,14}, {8,9,10,11}, {0,5,7}, {6,0,5}, {1,0,2,3,4}.

COMPLEXITY ANALYSIS OF THE ALGORITHM:

Metric Used: For complexity analysis, we have used the number of comparisons require for generating the adjacent and non adjacent set of the highest degree vertex in the graph.

ANALYSIS

Best Case:
The best case in the algorithm appears in the algorithm when the entire graph is a clique. In this case, we know that, all the nodes in the graph of n vertex (say) will have a degree of n-1. So, the number of comparisons required is:

$$n(n-1)=n^2-n$$
$$O(n^2)$$

Worst Case:
The worst case in the algorithm appears when the source graph contains all its edges as the cliques. In a graph of n nodes where all the edges are the cliques, the graph will have n-1 edges. Therefore the number of comparisons required for finding a **single clique** is as follows:

$$n(n-1)+(n-1)(n-2)+(n-2)(n-3)+\ldots\ldots+2$$
i.e, $O(n^2)$

Since, the number of cliques grows exponentially, there for there will be an exponential number of cliques and the time complexity for finding all of them will also be exponential.

Special Case:
In the special case if Moon Moser graph, the whole vertex set is divided into a number of triads. Now each time we divide a graph according to the policies of the algorithm we get a sub-graph containing n-3+1 number of vertices.

Therefore, the number of comparisons required is (n-3)(n-3+1). Again we divide the sub-graph thus obtained and are left with n-6+1 number of nodes in the currently obtained sub-graph. So, we can write

$$n(n-1)+(n-3+1)(n-3)+(n-6+1)(n-6)+\ldots\ldots+2$$

∗ The first term is used because at the first step, the entire graph of n nodes is to be checked by comparing each node with every other node for adjacency.

Here also if the number of triads is T, then the total number of cliques will be 3T, which is exponential. So, the time complexity for finding all the cliques will also be exponential.

The Algorithm Generates All Cliques Present In A Graph:

Procedural Details: The algorithm is based on dividing the graph into adjacent and non adjacent set of vertices of the node with maximum degree. And the sub graphs are recursively sub divided.

At each instance of the division, a check is done which checks to see whether the sub graphs having n number of nodes (say) have all nodes with degree n-1. If so, then the sub graph is declared to be a clique as a whole.

Decision Instances: In the best case, the source graph as a whole will be a clique which will be detected at the first instance before dividing the graph even for once as said in the procedural details.

In the worst case on the other hand, the graph may be a Moon Moser graph, where the cliques are only the single edges, the condition specified in the procedural details will be satisfied only at the leaf of the tree generated by dividing the graph into adjacent and non adjacent vertex set of maximum degree node of a sub graph.

In the average case, the cliques may be generated on the middle of the tree which will be satisfied by the condition specified in the procedural details and refrains from further division of the sub-graph or having further children of that node (the sub graph in concern).

Proof: Now, from the decision instances we can say that whether it is at the root node or middle of the tree or the leaf node, if there is a clique, it will emerge out of the tree. The reason behind is the check for each sub-graph of n nodes (say) having degree of each vertex as n-1, which declares a clique. In the worst case, we have to declare an edge as a clique, and since an edge consists of two vertices, therefore the degree of each node is (2-1) =1, hence the clique will be detected. Hence we can say that the algorithm will generate all the cliques present in the graph.

CONCLUSION

Graph algorithms have been often used to help understanding biology. Clique enumeration is a core component in many biological applications, such as gene expression networks analysis, and the quantitative study of high-throughput molecular phenotypes, and hence also can find out all the infected sub clusters of any infected community and the identified cliques can be labeled as micro containment zones. Several applications in bioinformatics and drug design, similarity of proteins or chemical formulae in general. The cliques of a graph are useful in cluster analysis in such fields as information retrieval and sociology. Complete Locked down for any community of any Country directly affects the countries' economy. Hence choosing micro containment zone in a infected community in polynomial time efficiently actually facilitates the countries' economy disaster in any pandemic situation. This work is nothing but a proposed model for dealing a massive blockage by clustering into sub parts, by solving small parts solves the whole problem in that situation in minimum time loss.

REFERENCES

[1] B. Rao and V. G. K. Murti. Enumeration of All Trees a Graph Computer Program, Electronics Letters, Vol. 6, No. 4, 1970.
[2] B. Rao and V. G. K. Murti. Enumeration of All Trees a Graph Computer Program, Electronics Letters, Vol. 6, No. 4, 1970.
[3] H. M. Trent, A Note on the Enumeration and Listing of All Possible Trees in a Connected Linier Graph, Proc. Nat. Acad. Sci. U.S.A., Vol. 40, p. 1004.
[4] Pak and A. Postnikov, Enumeration of Spanning Trees of Graphs, Harvard University, Massachusetts Institute of Technology, 1994.
[5] Pak and A. Postnikov, Enumeration of Spanning Trees of Graphs, Harvard University, Massachusetts Institute of Technology, 1994.
[6] M. Peikarski, Listing of All Possible Trees of a Linear Graph, lbid., CT-12, Correspondence, pp. 124-125, 1965.
[7] N. Deo, Graph Theory with Applications to Engineering and Computer Science. Prentice-Hall of India Private Limited, New Delhi, 2003.
[8] Reingold, Nievergelt and Deo, Combinatorial Algorithms, Prentice-Hall, Inc., 1977.
[9] S.L. Hakimi, On the Trees of a Graph and Their Generation. J. Franklin Inst. 270, pp. 347-359, 1961.
[10] S. Seshu and M. B. Reed, Linear Graphs and Electrical Networks, Rading, Mass, Addision- Wesley, 1961.
[11] S. Sen Sarma, A. Rakshit, R. K. Sen, A. K. Choudhury, An Efficient Tree Generation Algorithm, Journal of the Institution of Electronics and Telecommunications Engineers (IETE), Vol. 27, No. 3, pp. 105-109, 1981.
[12] W. Mayeda, Graph Theory, Wiley Inter-science, 1972.
[13] W. Mayeda & S. Seshu, Generation of Trees without Duplications, IEEE Trans, CT-12, pp. 181- 185, 1965.
[14] S.Arumugam and S.Ramachandran, *Invitation to Graph Theory*, 1st ed., Scitech Publication (India) Pvt. Ltd., Chennai, 2002.
[15] N.Deo, *Graph Theory with Applications to engineering and Computer Science*, Prentice Hall of India Pvt. Ltd., New Delhi, 2005.
[16] E.Horowitz, S.Sahani, S.Rajasekaran, *Fundamentals of Computer Algorithms*, 2nd ed., Universities Press (India) Pvt. Ltd (2008).
[17] Konneth Sorensen, Gerrit K. Janssens, "An Algorithm to Generate all Spanning Trees of aGraph in Order of Increasing Cost", University of Antwerp; Hasselt University Belgium(2005).
[18] A First Look at the Graph Theory:-Clark, Holton.
[19] Garey M. R. and Jhonson D.S, Computers and Intractability: A Guide to the Theory of NP- Completeness, Freeman, San Fracisco,1979.
[20] Chartrand G. and Lesniak L., Graphs & Digraphs, Third Edition, Chapman & Hall, 1996.
[21] Hartsfield N. and Ringle G., Pearls in Graph Theory A Comprehensive Introduction, Academic Press, INC, Harcourt Brace Jovanovich, publishers, 1997.
[22] Emotional Intelligence: Theory, Findings, and ImplicationsJohn D.Mayer, Peter Salovey ,David R. Caruso:Pages 197-215 | Published online: 19 Nov 2009 https://doi.org/10.1207/s15327965pli1503_02
[23] Emotionally Intelligent Algorithms dealing some practical combinatorial problems, Saptarshi Naskar, Tanmoy Biswas, Samar Sen Sarma, IJMA,VOL-11,ISSUE-9 SEPT-2020
[24] Finding Community infection spading factor's presence in a Community, Saptarshi Naskar, Arup Kr. Goswami, Samar Sen Sarma, IJEIT, VOL-10, ISSUE-12, June 2021pp. 8-9