Algorithms For Finding Liar Domination Number
For Fuzzy Path And Fuzzy Cycle

S. Roseline Mary S. Ruban Raj J. Maria Joseph

Abstract: We define the Liar’s Domination Set for fuzzy graphs using the membership value of strong arcs as follows. Let \( G = (\sigma, \mu) \) be a fuzzy graph on \( V \). Let the node \( t \) be dominated by the node \( s \) if \( t \in N[s] \), where \( N[s] = \{ s \} \cup \{ r \in V(G)/\mu(r, s) = \mu(t, s) \} \). \( D \subseteq V(G) \) is said to be Liar’s Domination Set if it satisfies the following two conditions. Every \( t \in V(G) \) is dominated by at least two nodes in \( D \). For Every two nodes \( t, s \in V, t + s \) is dominated by least three nodes in \( D \). The minimum fuzzy cardinality of Liar’s Domination Set is known as Liar’s Domination Number. In this paper we have given the algorithms for finding Liar’s Domination number of Fuzzy paths and Fuzzy Cycles.

Index Terms: Fuzzy Path, Fuzzy Cycle, Fuzzy Line Graph, Liar Domination Set, Liar Domination Number, Strong Fuzzy Graph, Weakest arc.

1 INTRODUCTION
Fuzzy graph theory plays vital role in various fields like clustering analysis, database theory, network analysis, Information theory, etc. [1] Fuzzy model can be used in problems handling uncertainty to get more accurate and precise solutions. [2] In 1975 Rosenfeld presented the notion of fuzzy graph and some fuzzy analogues of graph theoretical concepts such as Paths, Cycle and Connectedness. In 1977, Batacharya [3] and in 1989 Bhutani [4] explored the concept of Fuzzy Automorphism groups. In 1993, Moderson [5] and Nair [6] discussed cycles and co-cycles of fuzzy graphs. In 1998 and 2004, A. Somasundaram and S. Somasundaram [7] discussed the concepts of domination in fuzzy graphs. A. Nagoorgani and Ahamad [8] investigated strong and weak domination in Fuzzy Graphs. A. Nagoorgani and V. T. Chandrasekaran [9] introduced domination in fuzzy graphs using Strong Arc. P.J. Slater and M. L. Roden [10] and [11] introduced the notion of liar’s dominating set in graph theory. B. Balandra and B. Canoy [12] characterized the p-liar’s dominating sets in the composition of graphs. B.S. Banda and S. Paul [13] strengthened the complexity results of liar’s domination decision problem. B. Balandra and B. Canoy [14] studied the concept of liar’s domination in the join, corona and lexicographic product of graphs. Durgan and Altundag [15] have found liar domination number for middle graphs of some specific graphs. Let \( G \) be a fuzzy graph which can be used to model a system with each node \( t \) in \( V(G) \) representing an area of the facility such as a room, hallway, or ventilation duct. Likewise, fuzzy graph \( G \) can indicate a computer network where each node \( t \in V(G) \) indicates a processor. Edges of \( G \) can connect nodes indicating adjacent areas of the systems or processors with direct edges. A system or a processor will be recognized with the node that indicates it. Systems are subject to having an “intruder” such as a fire, saboteur, or thief that must be detected and have its location precisely recognized.

In this method an intruder must be located in a multiprocessor system. In general, it is assumed that the probable locations for one intruder are all of the nodes in \( V(G) \). \( N(t) = \{ s \in V | \mu(t, s) = \mu(t, s), where t \in V(G) \} \) is called the neighbourhood of \( t \) and \( N[t] = N(t) \cup \{ t \} \). It is assumed that a protection device placed at a node \( s \) can detect the existence of an intruder exactly when the intruder is in \( N[s] \). When a protection device at node \( s \) can identify the presence of an intruder at \( s \) or at a node in \( N(s) \), but which node in \( N(s) \) cannot be located, then one is involved in having a locating-dominating set. This set is called liar’s domination set. That is, liar’s domination set provides a single fault tolerant protection placement set where a location of fault device could be recognized. We have introduced liar’s domination for fuzzy graphs [16]. In this paper we have given algorithms for finding liar’s domination number for fuzzy paths and fuzzy cycles.

2 THEOREM 1
For all fuzzy paths of order \( n \), \( 4k + 3 \leq n \leq 4k + 6 \), where \( k = 1,2,3,\ldots \gamma(P_n) \leq \frac{3}{2}(n + 1) \)

2.1 Proof
Let \( v_1, v_2, \ldots, v_{(n-1)}, v_n \) be the vertices of \( P_n \). \( N[v_1] = \{ v_1, v_2 \} \). Since \( |N[v_1] \cap D| \geq 2 \), \( V_n \in P_n \). Similarly \( \gamma(P_n) \leq \frac{3}{2}(n + 1) \). In this method an intruder must be located in a multiprocessor system. In general, it is assumed that the probable locations for one intruder are all of the nodes in \( V(G) \). \( N(t) = \{ s \in V | \mu(t, s) = \mu(t, s), where t \in V(G) \} \) is called the neighbourhood of \( t \) and \( N[t] = N(t) \cup \{ t \} \). It is assumed that a protection device placed at a node \( s \) can detect the existence of an intruder exactly when the intruder is in \( N[s] \). When a protection device at node \( s \) can identify the presence of an intruder at \( s \) or at a node in \( N(s) \), but which node in \( N(s) \) cannot be located, then one is involved in having a locating-dominating set. This set is called liar’s domination set. That is, liar’s domination set provides a single fault tolerant protection placement set where a location of fault device could be recognized. We have introduced liar’s domination for fuzzy graphs [16]. In this paper we have given algorithms for finding liar’s domination number for fuzzy paths and fuzzy cycles.

By using the definition, if \( \gamma(P_{(k+1)}) \) \( v_{(k+2)} \) are members of \( D \), then \( v_{(k+3)} \) must be member of \( D \). Continuing this process one can identify that, When \( k = 1,4k + 3 \leq n \leq 4k + 6 \), maximum one vertex is non-member of \( D \). When \( k = 2,4k + 3 \leq n \leq 4k + 6 \), maximum two vertices are non-member of \( D \). When \( k = 3,4k + 3 \leq n \leq 4k + 6 \), maximum three vertices are non-member of \( D \) and so on. It follows that \( \gamma(P_n) \leq \frac{3}{2}(n + 1) \).
2.2 Example
Consider a Fuzzy Path having 23 nodes. It is shown in the Figure 3.

\[ L\text{D set} = \{v_1,v_2,v_3,v_4,v_5,v_6,v_7,v_8,v_9,v_{11},v_{12},v_{13}, \]
\[ v_{15},v_{16},v_{17},v_{19},v_{20},v_{21},v_{22},v_{23}\] \[ \gamma = 1 + 0.1 + 0.2 + 0.4 + 0.7 + 0.9 + 0.1 + 0.7 + 0.5 + 0.3 \]
\[ +0.2 + 0.7 + 0.5 + 0.4 + 0.7 + 0.2 + 0.1 + 0.7 + 0.9 = 9.3 \]

2.3 Corollary
For fuzzy line graphs of all fuzzy paths, \(4k + 3 \leq n \leq 4k + 6\) where \(k = 1,2,3,\ldots\), liar’s Domination number \(\gamma \leq \left\lfloor \frac{2}{5}n \right\rfloor \).

2.4 Proof
Fuzzy line graph of a fuzzy path \(P_n\) is the fuzzy path having \(n - 1\) vertices. From the previous theorem, \(\gamma(P_{n-1}) \leq \left\lfloor \frac{2}{5}n \right\rfloor\). Therefore, liar’s domination number of fuzzy line graphs of fuzzy path is \(\leq \left\lfloor \frac{2}{5}n \right\rfloor\).

2.5 Proposition:
For fuzzy Cycle \(C_n\), \(\gamma(C_n) \leq \left\lfloor \frac{3}{4}n \right\rfloor\)

2.6 Example

\[ \gamma = 13.4 \]

3. THEOREM 2
All fuzzy paths are strong.

3.1 Proof:
Fuzzy path is defined as the sequence of nodes \(u_1,u_2,\ldots,u_n\) such that \(\mu(u_i,u_{i+1}) > 0, \forall i = 1,2,\ldots,n - 1\). A fuzzy graph is said to be strong if all of its edges are strong edges. An arc is said to be strong, if \(\mu^v(u,v) = \mu(u,v)\), where \(u,v \in V\).
Since \(\mu(u_i,u_{i+1}) = \mu^v(u_i,u_{i+1})\) for all \(i = 1,2,\ldots,n - 1\) in a fuzzy path, all fuzzy paths are strong.

3.2 Example:

\[ \mu^v(v_i,v_{i+1}) = \mu(v_i,v_{i+1}) \forall v_i \in V, i = 1,2,\ldots,7 \]

Here \(\mu^v(v_1,v_2) = \mu^v(v_3,v_4) = 0.2\)

This fuzzy cycle is strong. The following are the algorithms of finding liar’s domination numbers of fuzzy path and fuzzy cycle. This algorithm is verified using python programming language and the outputs are also shown below the algorithms.

**Algorithm 1: Liar Domination Number for a Fuzzy Path**

**Input**: Membership values for Nodes in a Fuzzy Path (FP)
**Output**: Fuzzy Liars Domination Numbers

begin
\(P\text{\_set} = \text{input from user}\)
\(L\text{D\_set} : \text{Liar Domination Set}\)
\(N\text{LD\_set} : \text{Non Member of LD set}\)
\(i = 0\)
\(\text{size} = \text{Size}(P\text{\_set})\)

procedure sum\_Vmember(num1, list1, n)
while num1 <= \text{size - 4} do
\(n = n + \text{list}[\text{num1}]\)
\(\text{num1} = \text{num1} + 4\)
return \(n\)

end procedure

end

**Fig. 1. Example for Fuzzy Path**

**Fig. 2. Example for Fuzzy Cycle**

**Fig. 3. Example for Strong Fuzzy Paths**

**Fig. 4. Example for Strong Fuzzy Cycle**

Therefore, this fuzzy path is strong.

4. THEOREM 3
All fuzzy cycles are strong.

4.1 Proof:
A fuzzy cycle is a path with \(v_0 = v_n, n \geq 3\) and having more than one weakest arc, weakest arc is the arc having least membership value. If there exist more than one weakest in a fuzzy cycle, then all of its edges are strong.

Therefore, all fuzzy cycles are strong.

4.2 Example:
procedure add_LD(num)
    while num <= Size - 4 do
        LD_set.append(P_set[num])
        num = num + 4
    procedure add_NLD(num)
        while num <= Size - 4 do
            NLD_set.append(P_set[num])
            num = num + 4

if Size >= 7 then
    LD_set.append(P_set[i])
    LD_set.append(P_set[i+1])
    LD_set.append(P_set[i+2])
    LD_set.append(P_set[Size - 1])
    LD_set.append(P_set[Size - 2])
    LD_set.append(P_set[Size - 3])
    ch1 = i + 3
    ch2 = i + 4
    ch3 = i + 5
    ch4 = i + 6

    Total1 = P_set[ch1]
    Total2 = P_set[ch2]
    Total3 = P_set[ch3]
    Total4 = P_set[ch4]

    if Total1 > Total2 and Total1 > Total3 and Total1 > Total4 then
        add_NLD(ch1)
        add_LD(ch2)
        add_LD(ch3)
        add_LD(ch4)
    elseif Total2 > Total1 and Total2 > Total3 and Total2 > Total4 then
        add_NLD(ch2)
        add_LD(ch1)
        add_LD(ch3)
        add_LD(ch4)
    elseif Total3 > Total1 and Total3 > Total2 and Total3 > Total4 then
        add_NLD(ch3)
        add_LD(ch1)
        add_LD(ch2)
        add_LD(ch4)
    elseif add_NLD(ch4)
        add_LD(ch1)
        add_LD(ch2)
        add_LD(ch3)
    else
        Size := 3 and Size <= 6 then
            for i in P_set do
                LD_set.append(i)
        else
            print("Path must have 3 or more than 3 Vertices")
            print("your LD set is:", LD_set)
            print("your NLD set is:", NLD_set)

Algorithm 2: Liar Domination Number for a Fuzzy Cycle

Input : Membership values for Nodes in a Fuzzy Cycle (FC)
Output : Fuzzy Liars Domination Numbers

Begin
    FC_Set = input from user
    LD_Set : Liar Domination Set
    NLD_Set : Non Member of LD Set
    i = 0
    size = SIZE(FC_set)
    procedure getCount(num1, num2)
        i = i + 1
        num1 = (num1 + 1) % size
        return(i)
    procedure check_max(list1, N)
        for x in list1 do
            if val >= x:
                return False
        return True
    if size > 3 then
        for i to size do
            Total_FC_set = FC_set[i] + FC_set[(i + 1) % size] + FC_set[(i + 2) % size]
            location = Total_FC_set.index(min(Total_FC_set))
            LD_set ← FC_set[location] , FC_set[(location + 1) % size], FC_set[(location + 2) % size]
            Front_loc = (location + 3) % size
            Back_loc = (location - 1) % size
            while getCount(front_loc, rev_loc) != 0, 1, 2, 3 or 4 do
                if FC_set[front_loc] >= FC_set[rev_loc] then
                    NLD_set ← FC_set[front_loc]
                    LD_set ← FC_set[(front_loc + 1) % size]
                    FC_set[(front_loc + 2) % size]
                    FC_set[(front_loc + 3) % size]
                    Front_loc = (front_loc + 4) % size
                else
                    NLD_set ← FC_set[rev_loc]
                    LD_set ← FC_set[(rev_loc + 1) % size]
                    FC_set[(rev_loc + 2) % size]
                    FC_set[(rev_loc + 3) % size]
                    Back_loc = (rev_loc - 4) % size
                    if check_max(temp_set, N) then
                        NLD_set ← max(temp_set)
                        LD_set ← remaining nodes of temp_set
                        Temp_set.remove(max(temp_set))
                        LD_set ← remaining nodes of temp_set
                    else
                        NLD_set ← FC_set[front_loc]
                        LD_set ← FC_set[rev_loc]
                        NLD_set ← remaining nodes of temp_set
                elseif getCount(front_loc, rev_loc) == 3 then
                    NLD_set ← FC_set[rev_loc]
                    LD_set ← FC_set[(rev_loc - 1) % size]
                    FC_set[(rev_loc - 2) % size]
                    FC_set[(rev_loc - 3) % size]
                    rev_loc = (rev_loc - 4) % size
                    if getCount(front_loc, rev_loc) == 2 then
                        N = FC_set[front_loc] + FC_set[rev_loc]
                        temp_set ← FC_set[front_loc + 1]
                        FC_set[(front_loc + 2) % size]
                        FC_set[(front_loc + 3) % size]
                        if check_max(temp_set, N) then
                            LD_set ← FC_set[front_loc]
                            NLD_set ← FC_set[rev_loc]
                            NLD_set ← remaining nodes of temp_set
                        else
                            NLD_set ← FC_set[front_loc]
                            NLD_set ← FC_set[rev_loc]
                            LD_set ← remaining nodes of temp_set
                    elseif getCount(front_loc, rev_loc) == 1 then
                        NLD_set ← FC_set[front_loc]
                        LD_set ← FC_set[rev_loc]
                        NLD_set ← remaining nodes of temp_set
                    elseif getCount(front_loc, rev_loc) == 0 then
                        NLD_set ← FC_set[front_loc]
                        LD_set ← remaining nodes of temp_set
                    end
                end
            end
        end
    end

Fig. 5. Output for Fuzzy Path Algorithm.
NLD_set ← max(temp_set)
LD_set ← remaining nodes of temp_set

elseif size == 3 then
LD_set ← FC_set
else
Set Must have minimum 3 vertices
End

4 EQUATIONS

5 CONCLUSION
Liar’s dominating set is the set which helps to identify and report intruder’s location when the intruder presents in a computer network. This is recent and very useful concept to control fault tolerance in a computer network. We have given the algorithms for finding liar’s domination number for fuzzy paths and fuzzy cycles. In our next paper we are going to find the upper bound for the liar’s domination number of some specific fuzzy graphs.

6 REFERENCES

Fig. 6. Output for Fuzzy Cycle Algorithm.