

Extensions of product cordial graph labeling using MATLAB

Varsha Rath[†]

Sweta Srivastav^{*}

Sangeeta Gupta[§]

Department of Mathematics & Data Science

Sharda School of Engineering & Science

Sharda University

Greater Noida 201310

Uttar Pradesh

India

Abstract

Graph labeling is a widely studied area in graph theory with applications in computer science, network analysis, and algorithmic modeling. A binary vertex labeling function $f: \alpha(G) \rightarrow \{0, 1\}$, causes the labeling of an edge $f: \beta(G) \rightarrow \{0, 1\}$ as determined by $f * (ty) = f(t) * f(y)$. If the number of vertices differs, this labeling is referred to as a product cordial labeling (respectively, edges) labeled with 0 and 1 does not exceed one. In this work, we extend the study of product cordial labeling using MATLAB programming. We construct and verify product cordial labelings for three classes of graphs: (i) number of copies of the shell graphs, (ii) the star structure of concentric n -sunlet graphs, and (iii) number of the copies of concentric n -sunlet graphs. The proposed MATLAB implementations provide an algorithmic approach to generate these graphs, assign labels, and validate the cordiality conditions. This computational perspective not only confirms theoretical results but also demonstrates the utility of MATLAB for investigating new extensions of product cordial graphs.

Subject Classification: 05C78.

Keywords: MATLAB programming, Graph labeling, Product cordial labeling, Concentric n -sunlet graph, Star of concentric n -sunlet graph, Shell graph.

1. Introduction

Graph labeling is an important area of graph theory where numbers—most often 0 and 1—are assigned to vertices, edges, or both according to certain rules. Although it is a theoretical concept, it has many real-world applications,

[†] E-mail: 2022401996.varsha@dr.sharda.ac.in

^{*} E-mail: sweta.srivastav@sharda.ac.in (Corresponding Author)

[§] E-mail: sangeeta.gupta@sharda.ac.in

including network design, communication systems, circuit construction, and data organization. Among the various labeling methods, *cordial labeling*, introduced by Cahit [1], [2], plays a significant role. In this type of labeling, the goal is to keep a balance so that the difference between the number of elements labeled 0 and those labeled 1 is at most one.

Over time, graph labeling has developed into a broad research area. A detailed overview of its growth and different techniques can be found in the survey by Gallian [3]. Researchers have applied cordial labeling to a variety of graph structures. For example, Meena, Renugha, and Sivasakthi [4] studied its application to shell graphs, while Prajapati and Raval [5], along with Rokad [6], explored its behavior in other graph families.

A further extension of this idea is *product cordial labeling*, introduced by Sundaram, Ponraj, and Somasundaram [7]. Vaidya and Vyas [8] studied how this labeling behaves under tensor products of graphs while Vaidya and Barasara [9] explored related graph structures such as friendship graphs.

In this approach, edge labels depend on the product of the labels assigned to their end vertices. This concept opened the door for many new studies. For instance, Vaidya and Dani [10] examined product cordial labeling in several common graph types such as stars and wheels. Similarly, Vaidya and Kanani [11] focused on cycle-based graphs. Recent work by Deen and Elmahdy [12] shows that interest in graph labeling continues to grow, with new variations and graph classes being explored. At the same time, research by Sivaranjani et al. [13] highlights how discrete mathematics also connects with algebraic structures, showing the wide scope of the field.

In this paper, we build on earlier work by studying product cordial labeling for specific graph families, including multiple copies of shell graphs, concentric nnn-sunlet graphs, and their corresponding star structures. A key feature of this study is the use of MATLAB as a computational tool. Instead of relying only on theoretical arguments, MATLAB is used to construct graphs, apply labeling methods, and check whether the required conditions are satisfied. This makes it easier to handle larger and more complex graphs and provides additional support for the results. By combining mathematical reasoning with computational techniques, this work offers a practical approach to further research in product cordial labeling.

2. Terminology and notation

Definition 2.1: In graph theory, labeling is the process of assigning numbers or symbols to the vertices, edges, or both, of a graph according to specific rules. Formally, A mapping of a labeling $f: V(G) \cup E(G) \rightarrow N$ (or to be another finite set). When the mapping is restricted to vertices, it is called vertex labeling, and when it is restricted to edges, it is called edge labeling. In MATLAB, such mappings can be implemented using arrays or mapping functions over the graph object.

Definition 2.2: A graph $G = (V, E)$ is said to admit a product cordial labeling if its vertices are assigned values from the set $\{0, 1\}$ through a function $g: V(G) \rightarrow \{0, 1\}$. This vertex assignment naturally determines a binary labelling of the edges, given by $g*: E(G) \rightarrow \{0, 1\}$ described as $g*(uv) = g(u)g(v), \forall uv \in E(G)$. The labeling is a product cordial if the following balance conditions hold: $||V_0| - |V_1|| \leq 1$, and $||E_0| - |E_1|| \leq 1$, where V_0, V_1 denote the sets of vertices labeled 0 and 1, and E_0, E_1 denote the sets of edges labeled 0 and 1. In MATLAB, this condition can be checked algorithmically by counting the assigned labels for vertices and induced edges.

Definition 2.3: An the graph is n -sunlet graph, shows as S_n , is acquired by affixing one pendant edge to every vertex of a cycle C_n . Thus, S_n has $2n$ vertices and $2n$ edges. In MATLAB, it can be constructed by first generating the cycle graph C_n , then attaching pendant vertices systematically to each vertex of the cycle.

Definition 2.4: An n -sunlet graph that is 3- concentric, represented by $3 * S_n$, is the graph obtained by joining three n -sunlet graph copies in a concentric manner. The innermost copy consists of the cycle vertices $\alpha_1, \alpha_2, \dots, \alpha_n$ and its spokes are indicated as $\alpha'_1, \alpha'_2, \dots, \alpha'_n$, the inner most sunlet graph spokes are combined with the cycle vertices of the middle copy of the n -sunlet graph, whose vertices are again denoted by $\alpha'_1, \alpha'_2, \dots, \alpha'_n$ and its spokes as $x_1', x_2', \dots, x_3'$ and once more, its vertices of its spokes are merged with the outer sunlet graph. The outer most copy is shown by $x_1', x', \dots, x_3'$ and the spokes are represent as $w_1', w_2', \dots, w_n'$. Thus, $3*S_n$ is formed by arranging three layers of n -sunlet graphs concentrically, where the spokes of one sunlet serve as the cycle vertices of the next outer sunlet. In MATLAB programming, this graph can be generated by constructing the base n -sunlet and recursively fusing its pendant vertices with the cycle of the next layer until three concentric copies are obtained.

Definition 2.5: The $C_{(n,n-3)}$ is shell graph, also referred to as the graph which is fan graph f_{n-1} , is acquired by choosing $n - 3$ concurrent chords in a cycle C_n , where the common intersection point of the chords is called the apex vertex. Equivalently, is the shell S_n is isomorphic to the fan f_{n-1} . consequently, we may state this as $S_n = f_{n-1} = P_{n-1} + K_1$, where P_{n-1} is a path on $n - 1$ vertices and K_1 is a single isolated vertex that is joined to every vertex of the path. In MATLAB programming, this graph can be constructed by first generating a path graph P_{n-1} , then introducing a new vertex (the apex) and connecting it to all the vertices of the path, thereby forming the shell (or fan) structure.

3. Main Results

Theorem 3.1: For every positive integer n , the graph formed by joining two copies of the n -sunlet 3- concentric graph is the product cordial.

Proof: Examine the graph created by splitting 3-concentric n -sunlet graph into two copies. In the given graph, we denote the innermost a number of copy of the n -Sunlet graph as $\alpha_1, \alpha_2, \dots, \alpha_n$ and its radial edges indicated as $\alpha_1', \alpha_2', \dots, \alpha_n'$. The vertices of the middle sunlet graph are merged with each spoke of the inner most sunlet graph. The central copy of the n -sunlet graph is then indicated as $\alpha_1', \alpha_2', \dots, \alpha_n'$ and its spokes are fused with $x_1', x_2', \dots, x_n'$. Again, each spoke is combined with the graph of the outer vertices. The outermost copy is indicated by $x_1', x_2', \dots, x_n'$ and spokes are indicated as $w_1', w_2', \dots, w_n'$. The two graphs are connected together by fusing the outer vertices of both copies. To define the labelling, let $g : \alpha(3 * S_n) \rightarrow \{0, 1\}$, be given by the following cases:

$$G(t) = \begin{cases} 1; t = \alpha_i, i = 1, 2, \dots, n \\ 1; t = \alpha_i', i = 1, 2, \dots, n \\ 0; t = x_i', i = 1, 2, \dots, n \\ 0; t = w_i', i = 1, 2, \dots, n \end{cases}$$

With this labeling, the conditions $|\alpha_g(0) - \alpha_g(1)| \leq 1$ and $|\beta_g(0) - \beta_g(1)| \leq 1$ are satisfied. As a result, the graph G acknowledges a product cordial labeling. To illustrate this labeling scheme more clearly, let us consider an example and the labelled graph is shown in Figure 1.

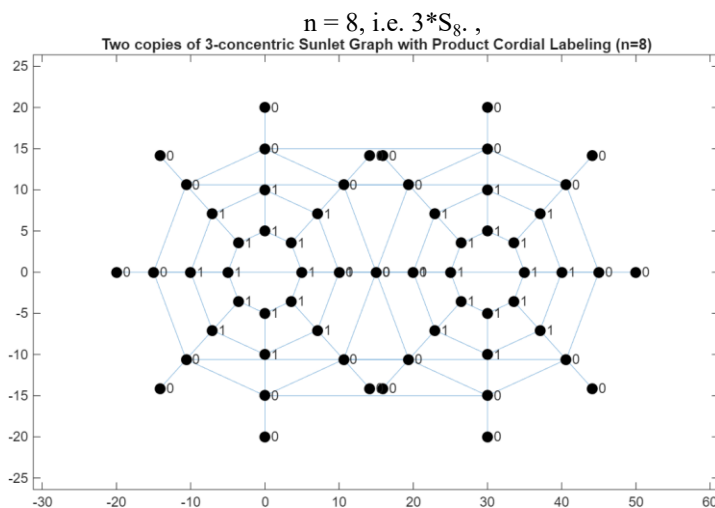


Figure 1
Two copies of $3 * S_8$

function theorem 3_1_productcordial(n)

% Theorem 3.1: Two copies of 3-concentric n -sunlet graph are product cordial

% Example: $n = 8$

% Create a graph object

```

G = graph();
% Build first copy centered at (0,0)
[G, coords1, labels1] = build_concentric_sunlet(G, n, [0,0], 5);
% Build second copy centered at (30,0)
[G, coords2, labels2] = build_concentric_sunlet(G, n, [30,0], 5);
% Fuse outer vertices of both copies (join copies by matching outer nodes)
for i = 1:n
    G = addedge(G, labels1.outer(i), labels2.outer(i));
end
% Combine coordinates
coords = [coords1; coords2];
% Assign product cordial labeling
nodeLabels = containers.Map;
% Labeling rule:
% Inner  $\alpha_i \rightarrow 1$ 
% Middle  $\alpha_i' \rightarrow 1$ 
% Outer  $x_i' \rightarrow 0$ 
% Pendant  $w_i' \rightarrow 0$ 
for i = 1:n
    nodeLabels(char(labels1.inner(i))) = 1;
    nodeLabels(char(labels1.middle(i))) = 1;
    nodeLabels(char(labels1.outer(i))) = 0;
    nodeLabels(char(labels1.pendant(i))) = 0;

    nodeLabels(char(labels2.inner(i))) = 1;
    nodeLabels(char(labels2.middle(i))) = 1;
    nodeLabels(char(labels2.outer(i))) = 0;
    nodeLabels(char(labels2.pendant(i))) = 0;
end
% Plot graph
figure;
p = plot(G, 'XData', coords(:,1), 'YData', coords(:,2), ...
    'MarkerSize', 7, 'NodeColor', 'k');
% Show labels (0/1) on vertices
lbls = cell(G.numnodes,1);
for i = 1:G.numnodes
    lbls{i} = num2str(nodeLabels(G.Nodes.Name{i}));
end
labelnode(p,1:G.numnodes,lbls);
title(sprintf('Two copies of 3-concentric Sunlet Graph with Product Cordial
Labeling (n=%d)',n));
% Verify product cordiality
verify_product_cordial(G, nodeLabels);
end
function [G, coords, labels] = build_concentric_sunlet(G, n, offset, rstep)

```

```

% Build a 3-concentric n-sunlet graph with pendant on each outer vertex
theta = linspace(0,2*pi,n+1); theta(end)=[];
coords = [];
labels.inner = strings(1,n);
labels.middle = strings(1,n);
labels.outer = strings(1,n);
labels.pendant = strings(1,n);
% Inner cycle  $\alpha_i$ 
r = rstep;
for i=1:n
    x = offset(1)+r*cos(theta(i));
    y = offset(2)+r*sin(theta(i));
    name = sprintf('a%d_%d',i,offset(1));
    G = addnode(G, name);
    coords = [coords; x y];
    labels.inner(i) = name;
end
for i=1:n
    G = addedge(G, labels.inner(i), labels.inner(mod(i,n)+1));
end
% Middle cycle  $\alpha_i$ 
r = 2*rstep;
for i=1:n
    x = offset(1)+r*cos(theta(i));
    y = offset(2)+r*sin(theta(i));
    name = sprintf('b%d_%d',i,offset(1));
    G = addnode(G, name);
    coords = [coords; x y];
    labels.middle(i) = name;
    G = addedge(G, labels.inner(i), labels.middle(i));
end
for i=1:n
    G = addedge(G, labels.middle(i), labels.middle(mod(i,n)+1));
end
% Outer cycle  $\alpha_i$ 
r = 3*rstep;
for i=1:n
    x = offset(1)+r*cos(theta(i));
    y = offset(2)+r*sin(theta(i));
    name = sprintf('c%d_%d',i,offset(1));
    G = addnode(G, name);
    coords = [coords; x y];
    labels.outer(i) = name;
    G = addedge(G, labels.middle(i), labels.outer(i));
end

```

```

for i=1:n
    G = addedge (G, labels.outer(i), labels.outer (mod(i,n)+1));
end
% Pendants w_i' on each outer vertex
r = 4*rstep;
for i=1:n
    x = offset(1)+r*cos(theta(i));
    y = offset(2)+r*sin(theta(i));
    name = sprintf('w%d_%d',i,offset(1));
    G = addnode(G, name);
    coords = [coords; x y];
    labels.pendant(i) = name;
    G = addedge (G, labels.outer(i), labels.pendant(i));
end
end
function verify_product_cordial (G, nodeLabels)
    % Count vertex labeling
    v0 = 0; v1 = 0;
    for i = 1:G.numnodes
        if nodeLabels(G.Nodes.Name{i}) == 0
            v0 = v0+1;
        else
            v1 = v1+1;
        end
    end
    fprintf ('Vertex labels: 0=%d, 1=%d, |diff|=%d\n',v0, v1,abs (v0-v1));
    % Count edge labeling
    e0 = 0; e1 = 0;
    for i=1:G.numedges
        ends = G.Edges.EndNodes(i,:);
        lab = nodeLabels(ends{1}) * nodeLabels(ends{2});
        if lab==0
            e0 = e0+1;
        else
            e1 = e1+1;
        end
    end
    fprintf('Edge labels: 0=%d, 1=%d, |diff|=%d\n', e0,e1, abs(e0-e1));
end

```

2. $n=7, 3^*S_7$, Graph Output will come in Figure window as shown in Figure

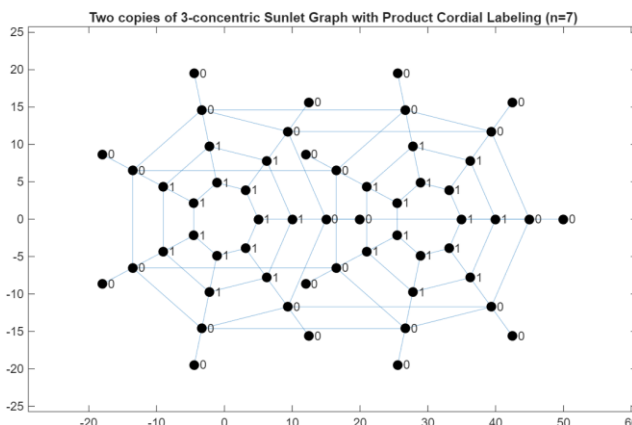


Figure 2
Two copies of $3 * S_7$

Theorem 3.2: A product cordial graph is created by combining two number of copies of the shell graph $2 * C_{(n,n-3)}$, $\forall$ odd n .

Proof: The labeling is constructed by joining two copies of the shell graph through a path. In this graph, we denote the vertices of the first duplicate of the shell graph as $\alpha_1, \alpha_2, \dots, \alpha_n$ and the 2nd copy of shell graph vertices are $x_1, x_2, \dots, x_n$. To specify the labeling, let $g : \alpha(C_{(n,n-3)}) \rightarrow \{0, 1\}$, be given by the below rule:

$$g(t) = \begin{cases} 1; t = \alpha_i, i = 1, 2, \dots, n \\ 0; t = x_i, i = 1, 2, \dots, n \end{cases}$$

With this labeling, the conditions $|\alpha_g(0) - \alpha_g(1)| \leq 1$ and $|\beta_g(0) - \beta_g(1)| \leq 1$ are satisfied. As a result, the graph G acknowledges a product cordial labeling.

To illustrate this labeling scheme more clearly, let us consider an example and the labelled graph is shown in Figure 3.

$n = 11$

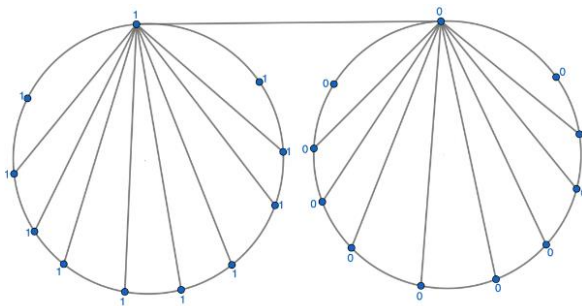


Figure 3
Two copies of $C_{(11,8)}$

```

% two_shells_product_cordial_no_axes.m
% Same as your function but hides the x/y axes and box so no axis lines appear.
function two_shells_product_cordial_no_axes(n)
    if nargin==0
        n = 11; % default
    end
    assert(mod(n,2)==1, 'Please choose odd n (as in theorem).');
    % Parameters for geometry
    R = 1.0; % radius of rim
    spacing = 3.0; % distance between the two copies
    center1 = [-spacing/2, 0];
    center2 = [ spacing/2, 0];
    hub_offset = [0, 1.1*R]; % hub placed above circle
    % Build first copy (nodes 1 .. n+1), hub is node 1 for copy1
    Nnodes = 2*(n+1);
    s = []; t = []; % edge lists
    % Build copy 1 (hub + rim)
    hub1 = 1;
    rim1 = 2:(n+1);
    for k = rim1
        s(end+1) = hub1; t(end+1) = k;
    end
    for k = 1:n
        a = rim1(k);
        b = rim1(mod(k,n)+1);
        s(end+1) = a; t(end+1) = b;
    end
    % Build copy 2 (hub + rim)
    hub2 = n+2;
    rim2 = (n+3):(2*n+2);
    for k = rim2
        s(end+1) = hub2; t(end+1) = k;
    end
    for k = 1:n
        a = rim2(k);
        b = rim2(mod(k,n)+1);
        s(end+1) = a; t(end+1) = b;
    end
    % join the two hubs by an edge (the path joining the copies)
    s(end+1) = hub1; t(end+1) = hub2;
    % Create graph
    G = graph(s, t);
    % Create coordinates for plotting
    XY = zeros(Nnodes,2);
    theta = linspace(0,2*pi,n+1);

```

```

theta(end) = []; % remove duplicate
for k = 1:n
    XY(rim1(k,:)) = center1 + [R*cos(theta(k)), R*sin(theta(k))];
end
XY(hub1,:) = center1 + hub_offset;
for k = 1:n
    XY(rim2(k,:)) = center2 + [R*cos(theta(k)), R*sin(theta(k))];
end
XY(hub2,:) = center2 + hub_offset;
% Labeling g: 1 for all vertices of copy1, 0 for all vertices of copy2
labels = zeros(Nnodes,1);
labels(1:(n+1)) = 1; % copy1 -> 1
labels((n+2):end) = 0; % copy2 -> 0
% Plot
figure('color','w','Position',[200 200 900 450]);
h = plot(G,'XData',XY(:,1),'YData',XY(:,2), ...
        'NodeLabel',[], 'LineWidth',1.5, 'MarkerSize',6);
axis equal
% Hide axes completely (no x/y lines, ticks, labels or box)
ax = gca;
ax.Visible = 'off'; % hides axis lines and ticks
ax.Box = 'off';
title(sprintf('Two copies of shell graph C_{(%d,%d)} joined by a path; n =
%d', n, n-3, n), 'FontSize',12);
% Color / annotate nodes by label
hold on;
for v = 1:Nnodes
    pos = XY(v,:);
    % small offset outward from center of that copy for text placement
    if v <= (n+1)
        center = center1;
    else
        center = center2;
    end
    dir = pos - center;
    if norm(dir) == 0
        offset = [0, 0.08];
    else
        offset = 0.08 * (dir / norm(dir));
    end
    text (pos(1)+offset(1), pos(2)+offset(2), num2str(labels (v)), ...
        'FontSize',10, 'Color',[0 0.4 0.8], 'FontWeight','bold', ...
        'HorizontalAlignment', 'center', 'VerticalAlignment','middle');
% draw node marker

```

```

    plot(pos(1), pos(2), 'o', 'MarkerFaceColor',[0.1 0.5 0.9],
'MarkerEdgeColor',[0.05 0.2 0.4], 'MarkerSize',6);
    end
    hold off;
    % save figure files for convenience (optional)
    saveas(gcf, sprintf('two_shells_n%d_noaxes.png', n));
    fprintf('Graph for n=%d created and saved as two_shells_n%d_noaxes.png\n',
n, n);
end
% Example:
% two_shells_product_cordial_no_axes (11);
% two_shells_product_cordial_no_axes (13);

```

2. $n=13$, Graph Output will come in Figure window as shown in Figure 4.

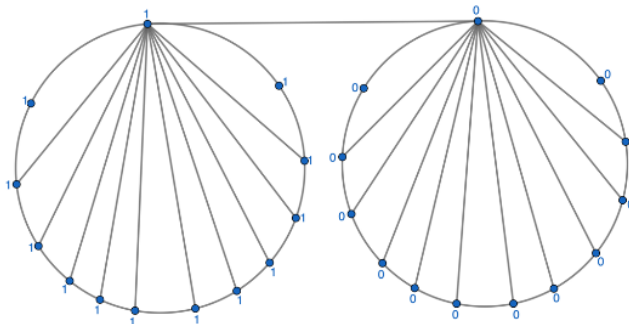


Figure 4
Two copies of $C(13, 10)$

4. Conclusion

Product cordial labeling in graph theory provides a systematic way of exploring balance conditions between vertices and edges under binary labeling. Through MATLAB programming, these concepts can be effectively implemented, allowing the construction, visualization, and verification of complex graph classes such as shell graphs, star of concentric sunlet graphs and concentric sunlet graphs. The computational approach not only confirms the theoretical results but also offers an algorithmic framework to extend product cordial labeling to larger graph families by means of duplication, fusion, and reconstruction techniques. Thus, MATLAB serves as a powerful tool for experimenting with new extensions of product cordial graphs and for supporting further investigations in graph labeling research.

References

- [1] I. Cahit, "A short proof of Groetzsch's three color theorem," (1958).
- [2] I. Cahit, "Cordial graphs—a weaker version of graceful and harmonious graphs," *Ars Combinatoria*, vol. 23, pp. 201–207 (1987).
- [3] J. A. Gallian, "Graph Labeling," *The Electronic Journal of Combinatorics*, Dynamic Surveys, DS6, (1998), doi: 10.37236/27.
- [4] S. Meena, M. Renugha, and M. Sivasakthi, "Cordial labeling for different types of shell graph," *International Journal of Scientific and Engineering Research*, vol. 6, no. 9, pp. 1282–1288 (2015).
- [5] U. M. Prajapati and K. K. Raval, "Some graphs in the context of product cordial labeling," *Mathematics Today*, vol. 33, pp. 58–66 (2017).
- [6] A. H. Rokad, "Product cordial labeling of double wheel and double fan related graphs," *Kragujevac Journal of Mathematics*, vol. 43, no. 1, pp. 7–13 (2019).
- [7] M. Sundaram, R. Ponraj, and S. Somasundaram, "Product cordial labeling of graphs," *Bulletin of Pure and Applied Sciences (Mathematics & Statistics)*, vol. 23E, pp. 155–163 (2004).
- [8] S. K. Vaidya and N. B. Vyas, "Product cordial labeling in the context of tensor product of graphs," *Journal of Mathematics Research*, vol. 3, no. 3, pp. 83–88 (2011).
- [9] S. K. Vaidya and C. M. Barasara, "Product cordial labeling for some new graphs," *Journal of Mathematics Research*, vol. 3, no. 2, pp. 206–211 (2011).
- [10] S. K. Vaidya and N. A. Dani, "Some new product cordial graphs," *Journal of Applied Computer Science & Mathematics*, vol. 8, no. 4, pp. 62–65 (2010).
- [11] S. K. Vaidya and K. K. Kanani, "Some cycle related product cordial graphs," *International Journal of Algorithms, Computing and Mathematics*, vol. 3, no. 1, pp. 109–116 (2010).
- [12] M. R. Zeen El Deen and G. M. Elmahdy, "Different types of odd harmonious labeling of super subdivision of various graphs," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 27, no. 8, pp. 2373–2407 (2024), doi: 10.47974/JDMSC-1945.
- [13] N. U. Sivaranjani, E. Nandakumar, G. Mittal, and R. K. Sharma, "The structure of the unit group of the group algebra $K_qSL(2, \mathbb{Z}_7)$," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 27, no. 8, pp. 2461–2472 (2024).

Received January, 2026