Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 4** : 1–16.
ISSN-0037-8712
doi:10.5269/bspm.79950

Fibonacci-Weighted Lexicographic Product of Open Neighborhood Graphs for Smart IoT Optimization

Rudrapati Bhuvaneswara Prasad, A. Mallikarjuna Reddy (Rtd.) and G. Shobhalatha

ABSTRACT: The paper has determined that the optimizing networks in Smart City IoT systems by application of Fibonacci-weighted Lexicographic Product Graphs. The emphasis is put on the regular graphs and their associated Open Neighborhood Graphs. The research uses the Fibonacci edge weight on different graphs, regular, cycle, complete and others by application of lexicographic products graphs. It looks at major graph features like the degree of the vertices, the structure of the edges and the weights of the all the edges which are computed through the use of the Fibonacci sequences. These graph models are easy to generate and visualize with the help of the given MATLAB codes, as well as help to create more efficient communication structures. This technique used to monitor the environment between Smart Cities incidentally improves fault tolerance, reduces energy consumption, and improves the usefulness of communication in large-scale internet of things systems.

Keywords: Lexicographic product graph, open neighborhood graph, Fibonacci-Weighted graph, network optimization, Smart City IoT.

Contents

1. Introduction

Graph theory is now a critical method of mathematics used to model the complexity of real world systems, including communication networks, transportation systems, and Smart city infrastructures. Lexicographic product graphs are the best suited methods to describe interconnected subsystems as they offer a layer structure that is rich in local and global connectivity. The addition of Fibonacci-based weights to the edge weights in these graphs introduces new frontier in terms of structural performance analysis and resource optimization of IoT networks.

The Fibonacci sequences find special application in weighted graphs models as they are recursive and exhibit growth properties that are typically scaled and self-repeating, as many networks are. Because smart cities generate large amounts of data that environmental sensors generate, a balance between reliability, energy efficiency, and fault tolerance is a key factor.. It is found through the use of Fibonacci-weighted lexicographic product graphs providing a strong mathematical basis with practical use in the modeling and enhancement of such communication systems.

The main goal of this study is to determine the total Fibonacci edge weights of lexicographic product graphs that are built from various graphs, including regular, cyclic, complete graphs and etc. It also emphasizes how these models can be used to optimize IoT connectivity in Smart Cities, especially for environmental monitoring.

2020 *Mathematics Subject Classification:* 05C30, 05C90.
Submitted July 11, 2025. Published February 26, 2026

2. Literature Review

Lexicographic products are still being researched because they maintain local structure while increasing global connection, which is helpful for modeling multi-tier networks like the Internet of Things and sensing fabrics in smart cities. Geodetic numbers for lexicographic and related products are examples of recent work that refines structural parameters on products $G \circ H$. These parameters affect gateway placement and shortest-path coverage in hierarchical networks [1]. The placement of "mutually non-collinear" monitors is further guided by new positional/covering invariants for strong and lexicographic products (such as general-position numbers) in order to prevent redundant sensing paths [6]. When switching from device-level to neighborhood-level abstractions, the adjacency produced by $N(v)$ alters connectivity and degree spectra, which is formalized by these product-graph refinements, complementing previous foundations on open-neighborhood graphs and their iterations [2,10].

In order to clarify degree regularity, edge counts, and induced adjacency rules that are important for per-node load and inter-cluster bandwidth in big systems, new studies on the open-neighborhood and lexicographic interface examine edge attributes when $G \circ N(G)$ (or variants) is produced [9]. Weighted modeling, whether prices represent network dependability, latency, or energy, directly benefits from these structural insights.

Weighted Fibonacci generalizations and graph families with a Fibonacci flavor are gaining popularity, and they can support monotone, growth-aware weight schedules, even though "Fibonacci-weighted" graphs are a specialty. The "Fibonacci-run graphs" software explores graph families whose combinatorics inherit Fibonacci constraints, which are practical analogs for encoding diameter bounds and lightweight routing states [7]. Link-cost models in networks can be mapped to weighted extensions of Fibonacci sequences proposed by parallel number-theoretic work, which provide rational techniques to control weights (e.g., prioritizing high-degree relationships more softly than linear growth) [8]. A reference point for analyzing routing and connectivity topologies with Fibonacci regularities is still earlier findings on Fibonacci networks [3].

In the year 2024–2025 research focusing on IoT optimization shifts to topological and graph-theoretic control under practical restrictions. Graph structure and information-theory constraints co-design robust IoT layouts, as demonstrated by a 2024 topology-control study that incorporates local geometry information into global graph realizations to balance error probability and code rate [5]. Graph-powered techniques for data quality (such as anomaly/consistency graphs and dependency graphs) in monitoring platforms are listed in a 2025 survey published in the *Journal of Network and Computer Applications*. The survey emphasizes how weighted graphs direct cleaner, redundancy, and trust in sensor data streams [4].

In recent studies, the focus has been on layered IoT architectures, standardization, and performance metrics (latency, energy, and reliability) for smart cities. Graph models, such as product graphs, formalize communication between and within zones and facilitate scale optimization [7,8,11]. Cited domain-specific knowledge graphs are also used for semantic discovery, showing how diverse urban data is converted into graph backbones, which is important for control and analytics on top of physical connectivity graphs [12].

Definition 2.1 ***Open Neighborhood:***
Suppose that $G = (V, E)$ is a regular, finite, undirected, and simple graph with $|V| = p$ vertices and $|E| = q$ edges. The collection of all vertices that are directly connected to u by an edge is known as the open neighborhood of u for any vertex $u \in V$. This can be expressed symbolically as:

$$N(u) = \{v \in V : uv \in E\}.$$

The open neighborhood structure of the graph can be collectively represented as a family of sets:

$$S = \{N(u_1), N(u_2), \ldots, N(u_p)\},$$

which captures the local adjacency information of every vertex in G.

Definition 2.2 ***Lexicographic Product of Graphs:***
The construction of the lexicographic product $G[H]$ of two graphs G and H involves "expanding" each

vertex of G into a copy of H. The graph that is produced has a set $V(G[H]) = V(G) \times V(H)$ indicating that a complete duplicate of H is used to replace each vertex of G. In $G[H]$, if two vertices (u, v) and (u', v') are connected if either:

(i) if $u = u'$ and v is adjacent to v' in H, or

(ii) if u is adjacent to u' in G, then all of the vertices in the copy of H associated with u are connected to all of the vertices in the copy of H associated with u'.

Definition 2.3 ***Edge-Weighted Fibonacci Graph:***
A weighted edge A graph that uses Fibonacci numbers to determine the weights given to each edge is called a Fibonacci graph.

Let $G = (V, E)$ be a finite, undirected, and simple graph in formal terms. Denote the degree of each vertex $u \in V$ by using $\deg(u)$. Recursively, the Fibonacci sequence is defined as follows:

$$F_0 = 0, \quad F_1 = 1, \quad F_k = F_{k-1} + F_{k-2}, \quad k \geq 2$$

The Fibonacci weight of an edge $e = (u, v) \in E$ is determined as a function of the degrees of its incident vertices for every edge $e = (u, v) \in E$:

$$W(e) = F_{\deg(u)} + F_{\deg(v)}$$

Because of this, every edge has a numerical value that is based on the Fibonacci sequence and indicates how connected its endpoints.

The sum of all edge weights is the graph's overall Fibonacci weight:

$$W(G) = \sum_{e \in E} W(e)$$

3. Main Results

Theorem 1. Let G be a Lexicographic product of two Regular graphs G_1 and G_2, where G_2 is the Open Neighborhood Graph of G_1. Then the edge weight Fibonacci of a graph G is given by

$$\boldsymbol{W}(\boldsymbol{G}) = 2mn(2n+1)(m-1) \cdot \boldsymbol{F}_{(2n+1)(m-1)} \quad \textbf{for } m, n \geq 4.$$

Proof: Let G_1 be a regular graph of order m with the set of vertex $x_1, x_2, \ldots, x_m$ and let $G_2 = N(G_1)$ denote the Open Neighborhood Graph associated with G_1.

Assume G_2 has exactly $2n$ vertices. The vertex set is defined as

$$V(G_2) = \{y_1, y_2, \ldots, y_{2n}\} = \{y_j \mid 1 \leq j \leq 2n\} \tag{1}$$

The Lexicographic product graph G is a product of the regular graph G_1 with their corresponding Open Neighborhood Graph $N(G_2)$. Therefore, the number of vertices in the Lexicographic product Graph is $2mn$ and it is denoted by $G = G_1[G_2]$. Hence the vertex set is

$$V(G) = \{(x_i, y_j) \mid 1 \leq i \leq m,\ 1 \leq j \leq 2n\} \tag{2}$$

Now we have to compute the total number of edges in the Lexicographic product Graph G.

From the definition of Lexicographic product, if two vertices (x_i, y_i) and $(x_{i'}, y_{j'})$ are adjacent in G, if

(i) $x_i = x_{i'}$ and y_j is adjacent to $y_{j'}$ in $E(G_2)$, or

(ii) x_i and $x_{i'}$ are adjacent in $E(G_1)$.

Case (i): If $x_i = x_{i'}$ & $y_j y_{j'} \in E(G_2)$, for every vertex $(x_i, y_j) \in V(G)$ inside its own copy, the number of neighbors is equal to $\deg_{G_2}(y_j)$. Since $G_2 = N(G_1)$ is formed from Open Neighborhood of G_1 and each y_j has exactly $(m-1)$ neighbors.

$$\therefore \deg_{G_2}((x_i, y_j)) = m - 1 \tag{3}$$

Case (ii): If $x_i x_{i'} \in E(G_1)$, in addition, x_i has $(m-1)$ neighbors in G_1 and each neighbor contributes all $2n$ vertices from the corresponding copy in G_2. By giving $2n(m-1)$ neighbors.

$$\therefore \deg_{G_2}((x_i, y_j)) = 2n(m-1) \tag{4}$$

We conclude that for the above two cases, we have that the degree of each vertex in G is

$$\deg_G((x_i, y_j)) = (m-1) + 2n(m-1) = (m-1)(2n+1) \tag{5}$$

We know that the Lexicographic product Graph G is Regular Graph. Now we have to find the number of edges in the Lexicographic Product graph G. Since there are $2mn$ vertices in G.

By handshaking Lemma implies that,

$$|E(G)| = \frac{V(G) \cdot \deg(G)}{2} = \frac{2mn(m-1)(2n+1)}{2} \tag{6}$$

$$\Rightarrow |E(G)| = mn(m-1)(2n+1)$$

Again, we prove that the edge weight Fibonacci of a Graph G. Now let $F_{t \geq 0}$ denote the Fibonacci series defined by $F_0 = 0, F_1 = 1$, then $F_t = F_{t-1} + F_{t-2},\ t \geq 2$.

$$\tag{7}$$

Therefore, the edge weight of a Fibonacci of a graph is

$$w(e) = F_{\deg_G((x_i, y_j))} + F_{\deg_G((x_{i'}, y_{j'}))}$$

$$(e) = 2 \cdot F_{(m-1)(2n+1)} \tag{8}$$

The total number of edges weighted in a Lexicographic Product Graph G is

$$W(G) = |E(G)| \times w(e) = mn(m-1)(2n+1) \times 2 \times F_{(m-1)(2n+1)}$$

$$W(G) = 2mn(m-1)(2n+1) \cdot F_{(m-1)(2n+1)} \tag{9}$$

Hence eq. (9) represents the total Fibonacci edge weight of G is

$$2mn((m-1)(2n+1)) \cdot F_{(m-1)(2n+1)}.$$

□

Example: $m = n = 4, 5, 10$

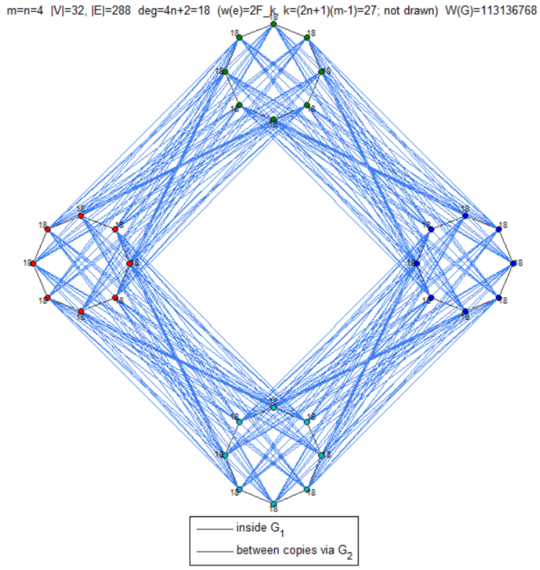

Figure 1: $m = n = 4$

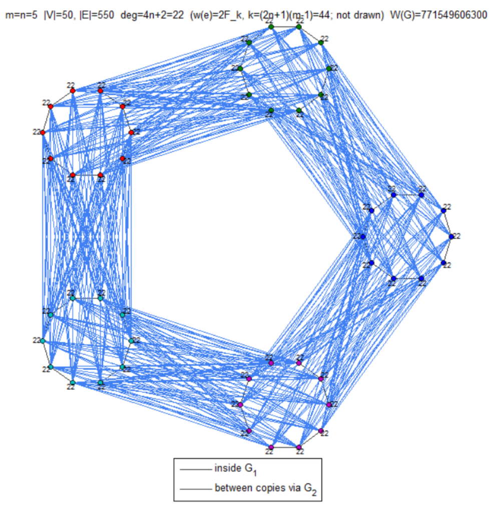

Figure 2: $m = n = 5$

Algorithm

Input: integer $n \geq 4$
Output: One plot of G with vertex degree labels (no per–edge labels); title summarizing counts and weights.

1. Read n and validate $n \geq 4$; set $m \leftarrow n$, $H \leftarrow 2n$, $N \leftarrow m \cdot H$.
2. Build edge sets:
   a) *Inside G_1 (per copy):* For each $u = 1, \ldots, m$ and $j = 1, \ldots, H$, let $j' = (j \bmod H) + 1$; add
   $$(\text{idx}(u, j), \text{idx}(u, j')).$$

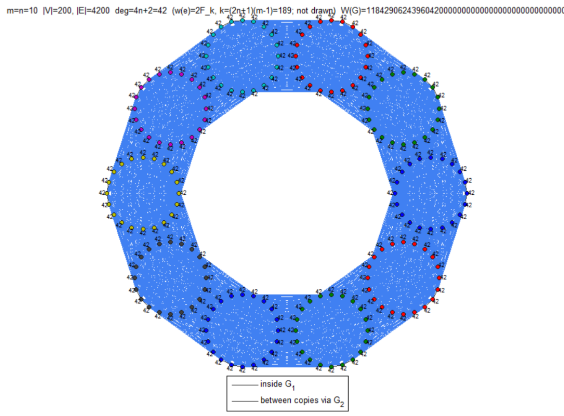

Figure 3: $m = n = 10$

b) *Between copies via G_2:* For each $u = 1, \dots, m$, let $v_{\text{next}} = (u \bmod m) + 1$; for all $j, k \in \{1, \dots, H\}$, add

$$(\text{idx}(u, j),\ \text{idx}(v_{\text{next}}, k)).$$

3. Compute counts (no need to materialize adjacency for counts):

$$|V| = N, \qquad |E| = m \cdot H \cdot (H+1) = 2n^2(2n+1), \qquad \deg = 4n+2 \text{ (regular)}.$$

4. Compute Fibonacci edge weight (for summary): $k = (2n+1)(m-1)$. Iteratively compute F_k via

$$(a, b) \leftarrow (0, 1); \quad \text{repeat } k \text{ times } (a, b) \leftarrow (b, a+b).$$

   Set $w(e) = 2 \cdot F_k$ and total weight $W(G) = w(e) \cdot |E|$.

5. Layout vertices (*ring-of-rings*): place m copy centers on a circle of radius R; in each copy, place H vertices on a circle of radius r around its center.

6. Render (single plot): draw E_{G_1} and E_{G_2} with distinct colors; draw vertices colored by copy; label each vertex with $\deg = 4n+2$; add a title showing $m = n$, $|V|$, $|E|$, deg, k, F_k, and $W(G)$. Do not save to file and do not label per–edge weights.

Theorem 2. Let $G = C_n[N(C_n)]$, $n \geq 4$, the lexicographic product of the cycle graph C_n with its Open Neighborhood Graph $N(C_n)$. Then the edge weight (Fibonacci) of G is

$$W(G) = 4n^2(2n+1)F_{4n+2}.$$

Proof: Consider the graph $G = C_n[N(C_n)]$, which is defined as the lexicographic product of the cycle graph C_n with its Open Neighborhood Graph $N(C_n)$.

The lexicographic product of two graphs yields a new structure whose vertices are ordered pairs obtained from the lexicographic expansion of the constituent vertex sets, and edges are induced according to the adjacency rules of the original graphs. The resulting graph captures a higher–order connectivity structure.

Since in the lexicographic product the total number of vertices is $2n^2$ (each vertex of C_n contributes n vertices, and vertices of $N(C_n)$ are doubled in the product), each vertex in G exhibits a degree of

$$\deg_G = 4n + 2. \tag{10}$$

(To see this, combine the intra–copy neighbors from C_n and the inter–copy links contributed by $N(C_n)$ and the cross edges generated in G.)

From this, in the lexicographic product G every vertex is adjacent to precisely $2n+1$ other vertices in each relevant copy, and consequently the total number of edges is

$$|E(G)| = \frac{|V(G)| \cdot \deg_G}{2} = \frac{2n^2\,(4n+2)}{2} = 2n^2(2n+1). \tag{11}$$

Let $G = (V, E)$ be a Fibonacci–weighted graph. Define the Fibonacci sequence $F_0, F_1, \ldots$ by $F_0 = 0$, $F_1 = 1$, $F_k = F_{k-1} + F_{k-2}$ for $k \geq 2$. For an edge $uv \in E$, the Fibonacci weight is

$$w(e) = F_{\deg(u)} + F_{\deg(v)} = F_{4n+2} + F_{4n+2} = 2F_{4n+2}.$$

Hence, the total Fibonacci weight of G is

$$W(G) = \sum_{e \in E(G)} W(e) = |E(G)| \times w(e) = 2n^2(2n+1) \times 2F_{4n+2} = 4n^2(2n+1)F_{4n+2}. \tag{12}$$

Thus, (12) is the Fibonacci edge weight of the lexicographic product graph G.

Ex: $n = 4, 5, 25$

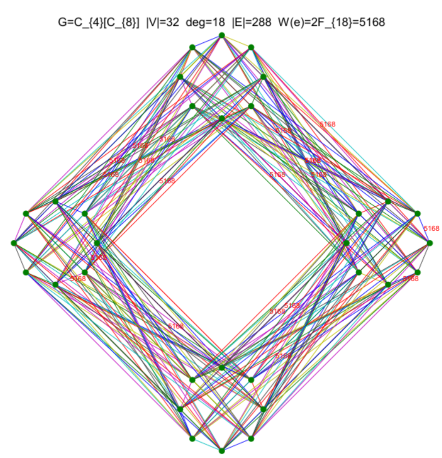

Figure 4: $n = 4$

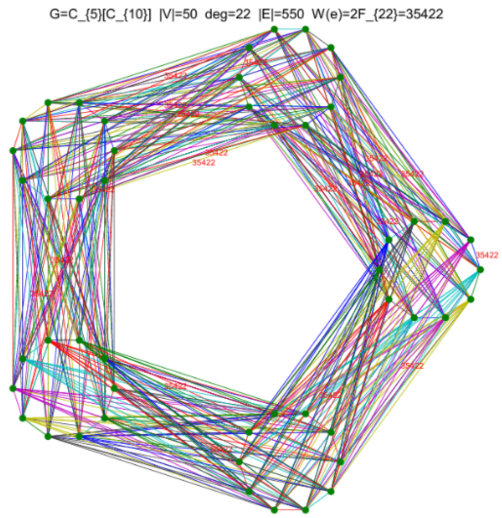

Figure 5: $n = 5$

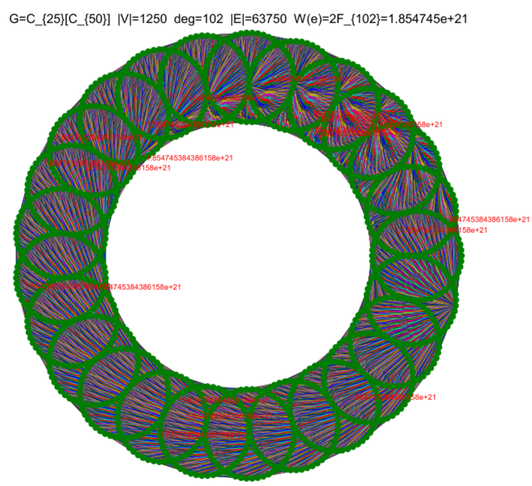

Figure 6: $n = 25$

Algorithm

1. **Input:** Accept values of n ($n \geq 4$) and define the output directory.
2. **For each** n**:** calculate $H = 2n$ and $N = nH$. Define an index mapping.
3. **Build edges:**
   (a) Within each C_{2n} cycle, connect consecutive vertices.
   (b) Between adjacent cycles in C_n, add complete bipartite edges of size $H \times H$.
4. Form the adjacency matrix, symmetrize, and compute degrees/edge count.
5. Compute Fibonacci weight: get F_{4n+2}; set $w = 2F_{4n+2}$.

6. Place vertices: n cycle centers on a large circle; each C_{2n} on a small circle.
7. **Visualization:** render edges in multiple colors, display nodes, add title, label up to 20 random edges with $w(e)$.
8. Save as `G_Cn_lex_C2n_weighted.pdf`.

Theorem 3: For $n \geq 3$, if $G = K_n[N(K_n)]$ denote the lexicographic product of the complete graph K_n with its Open Neighborhood Graph $N(K_n)$. Then the total Fibonacci edge weight of Graph G is given by

$$W(G) = 4n^2(2n+1)(n-1)F_{(2n+1)(n-1)}.$$

.

Proof: Let $G = (V, E)$ be a simple graph. For a vertex $v \in V$, the degree of v is denoted by $\deg(v)$. The Fibonacci number corresponding to the degree $\deg(v)$ is written as $F_{\deg(v)}$.

Now consider the Lexicographic Product $G = K_n[N(K_n)]$, by the reference of edge properties of the Lexicographic product of open neighborhood graphs of Complete Graph, which means $2n$ vertices in G.

Since every vertex in K_n is adjacent to all other $(n-1)$ vertices and in the Lexicographic product each adjacency expands across the $2n$ vertices of the Open Neighborhood Graph.

It follows that each vertex in G is adjacent to exactly $(2n+1)(n-1)$ other vertices.

Hence G is a regular graph with

$$\deg_G(v) = (2n+1)(n-1) \quad \forall v \in V(G). \tag{13}$$

Since the total number of edges in G are

$$|E(G)| = \frac{|V(G)| \cdot \deg_G}{2} = \frac{2n^2(2n+1)(n-1)}{2}. \tag{14}$$

For every edge $uv \in E(G)$, then the Fibonacci weight is

$$w(e) = F_{\deg(u)} + F_{\deg(v)} = F_{(2n+1)(n-1)} + F_{(2n+1)(n-1)} = 2F_{(2n+1)(n-1)}.$$

Therefore, the total Fibonacci edge weight of the graph G is

$$W(G) = |E(G)| \times w(e) = \frac{2n^2(2n+1)(n-1)}{2} \times 2F_{(2n+1)(n-1)}.$$

Hence

$$W(G) = 4n^2(2n+1)(n-1)F_{(2n+1)(n-1)}. \tag{15}$$

Therefore eq. (15) represents the total edge weight of Fibonacci Graph G.

Ex: $n = 3, 4, 15$

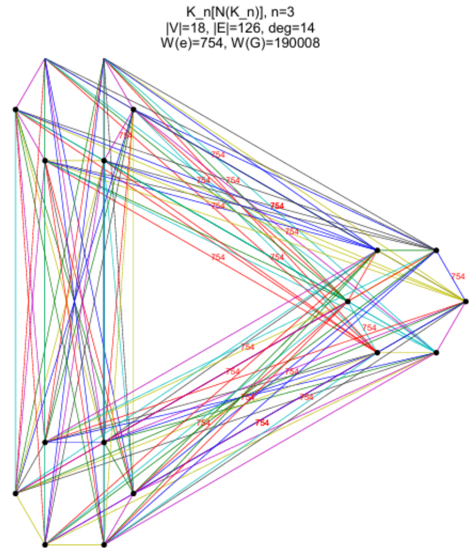

Figure 7: $n = 3$

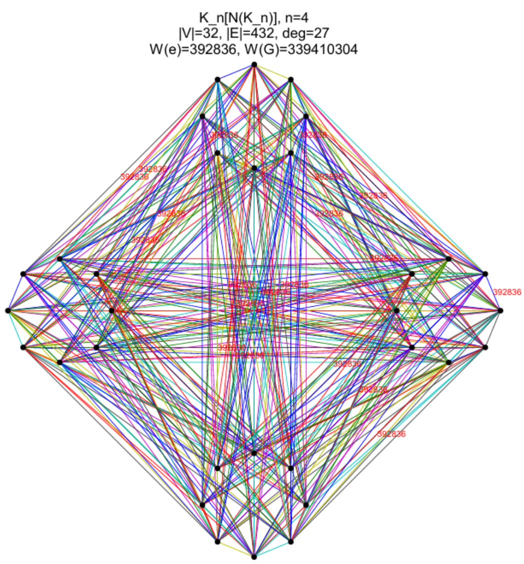

Figure 8: $n = 4$

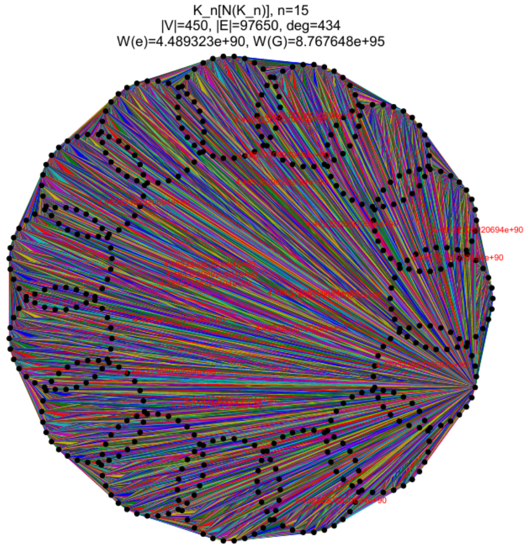

Figure 9: $n = 15$

Algorithm: Fibonacci-Weighted Graph $G = K_n[N(K_n)]$

1. **Input Check:** Take $n \geq 3$. If smaller, stop.
2. **Graph Parameters:** Vertices $= 2n^2$; Degree $= (2n+1)(n-1)$; Edges $= n^2(2n+1)(n-1)$.
3. **Fibonacci Step:** Compute the Fibonacci number at index = degree.
4. **Weights:** Edge weight $w(e) = 2 \times F_{\deg(v)}$; Total weight $= 2 \times$ Edges $\times$ Edge weight.
5. **Graph Structure:** Each copy of $N(K_n)$ forms a cycle of $2n$ nodes. Each pair of copies is fully connected.
6. **Layout:** Place n copies evenly in a large circle; inside each copy, arrange the $2n$ nodes in a small circle.
7. **Visualization:** Draw edges, scatter nodes, and label some edges with their weight.
8. **Output:** Report vertices, edges, degree, Fibonacci term, edge weight, total weight.

Theorem 3.1 *Theorem 4: Let $G = K_{n,n}[N(K_{n,n})]$ denote the Lexicographic product of the Complete Bipartite Graph $K_{n,n}$ with its Open Neighborhood Graph $N(K_{n,n})$ where $n \geq 2$. Then the total edge weight Fibonacci index of G is given by*

$$W(G) = 8n^2(4n+1) \cdot F_{4n+1}.$$

Proof: Consider G to be a graph of the lexicographic product of $K_{n,n}$ and $N(K_{n,n})$ with $8n^2$ vertices.

Since the Complete Bipartite Graph $K_{n,n}$ has $2n$ vertices and twice the vertices, that is, $2(2n) = 4n$ are in the open neighborhood of the complete bipartite graph $N(K_{n,n})$. Therefore G is a Regular Graph.

Now we show that the total number of edges in the Lexicographic Product Graph G. Using the definition of Lexicographic Product, every vertex in G has degree is $4n^2 + n = n(4n + 1)$.

$$\deg_G(v) = n(4n + 1). \tag{16}$$

Since in the graph, the total number of vertices is $(2n)^2(2n) = 8n^2$.

$$|V(G)| = 8n^2. \tag{17}$$

Therefore, the total number of edges are $4n^2(4n + 1)$.

$$|E(G)| = 4n^2(4n + 1). \tag{18}$$

To prove that the total edge weight Fibonacci of a Graph G. For each edge $e \in E(G)$, the total Fibonacci edge weight is

$$w(e) = F_{\deg(u)} + F_{\deg(v)} = 2F_{(4n+1)}. \tag{19}$$

Thus, the total edge weight Fibonacci of a Graph G is

$$W(G) = |E(G)| \times w(e) = 4n^2(4n + 1) \times 2F_{(4n+1)}.$$

Hence,

$$W(G) = 8n^2(4n + 1) \cdot F_{(4n+1)}. \tag{20}$$

□

Ex: $n = 2, 3, 20$

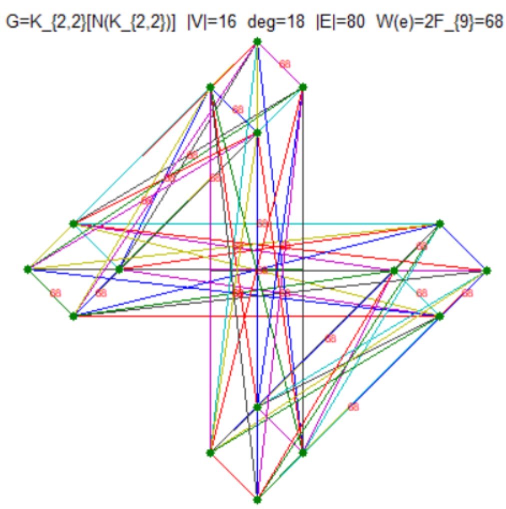

Figure 10: $n = 2$

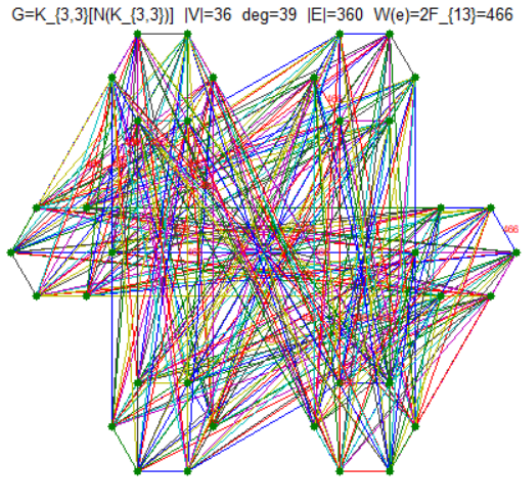

Figure 11: $n = 3$

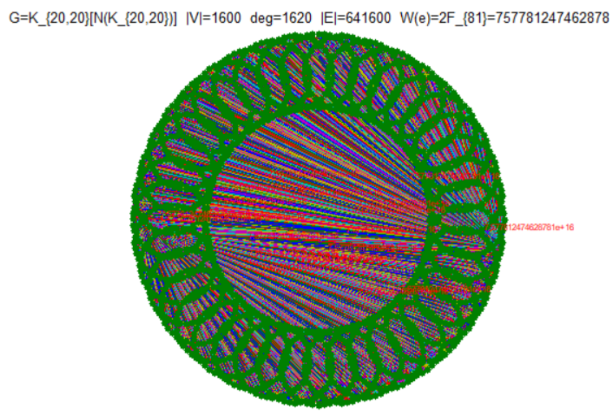

Figure 12: $n = 20$

Algorithm

1. Input $n \geq 2$.
2. Set $H = 2n$, total vertices $N = (2n) \times H$. Define index function $\mathrm{idx}(u, j) = (u - 1)H + j$.
3. Build edges inside each copy: connect vertices in a cycle of length H (K_{2n}).
4. Add edges between copies from $K_{n,n}$: for $u \in \{1, \dots, n\}$, $v \in \{n + 1, \dots, 2n\}$, connect all pairs $(\,\mathrm{idx}(u, j), \mathrm{idx}(v, k)\,)$.
5. Form adjacency matrix A, symmetrize it, and compute vertex degrees and edge count.
6. Compute Fibonacci number F_{2n+1} iteratively; set edge weight $w = 2F_{2n+1}$.
7. Embed vertices: place $2n$ copy centers on a large circle; each copy's H vertices on a small circle around its center.
8. Plot graph: draw colored edges, scatter vertices, add title, and label up to 20 random edges with weight W.

4. Application

In this section, Fibonacci-weighted lexicographic product graphs are applied to IoT networks in Smart Cities. The goal is to increase environmental monitoring's dependability, fault tolerance, and energy efficiency. The approach exhibits durability and scalability for more extensive IoT applications.

Edge Weighted Fibonacci of Lexicographic Product Graph in Smart City IoT Networks:

In the context of Smart City IoT networks, environmental monitoring applications – including air quality analysis, temperature and humidity tracking, and noise pollution assessment – are crucial for maintaining public health and urban safety. These applications rely on dense deployments of sensors across different city sectors, which creates the need for optimized communication to ensure reliable data transfer, minimal energy usage, and robust fault tolerance. A promising way to model and enhance such communication is through the Fibonacci-weighted lexicographic product graph

$$G = C_n[N(C_n)].$$

This graph-based approach provides an efficient representation of sensor groupings and the communication pathways both within and between city zones. By assigning Fibonacci-based weights to edges according to their connectivity level, the model captures essential parameters such as signal quality, transmission power, delay, and reliability. As a result, this method enables the design of IoT sensor networks that are not only more energy-efficient but also scalable and resilient, making them well-suited for Smart City environmental monitoring.

Scenario: Environmental Monitoring in a Smart City:

Think of a Smart City having 5 environmental areas and each area will have 10 sensors to check the air quality, temperature, humidity, and noise levels. These sensors must also be able to interrelate with one another within the zone and interconnect to give real time data of the environment. The issue is to ensure the optimization of communication paths to have the minimum energy usage, maximum reliability and minimum delays in data transmission.

Graph Representation

- **Clusters:** The sensors are placed in 10 clusters (each cluster has 10 sensors) in the 5 environmental zones. The sensors in a zone are linked in a cycle graph C_{10} to show local communication in the zone.
- **Edges Between Clusters:** Sensors in one zone are connected to all sensors in the other zone where they form a complete bipartite graph between two zones. This arrangement enables effective and easy flow of data in various parts of the city.

Fibonacci-Weighted Communication Links:

In this type of network, the weight of every communication connection is determined by the Fibonacci number of the degree of the sensor nodes. Communication cost is determined by the node degree which is the number of other sensors a node is connected to. The more the degree, the more the Fibonacci weight, and it provides a means of modeling and controlling the utilisation of energy or resources in the network:

$$W(e) = 2F_{\deg(G)} = 2 \times 610 = 1220$$

This weight may signify:

- Communication energy consumption (an increase in weight results in an increase in energy consumption).
- Transmission delay (greater weights indicate links of greater latency).
- Communication reliability (higher weight means that communication links are less reliable).

Total Network Weight:

The weight $W(G)$ of the network is the total of all the communication costs in the system:

$$W(G) = 4n^2(2n+1) \times F_{\deg(G)} = 4 \times 25 \times 11 \times 4 \times 610 = 154313721600.$$

This total weight measures the total cost of communication of the complete network, which can be optimized to be energy-efficient, to achieve data throughput and reliability.

Optimization Focus:

- **Energy Savings:** Use paths with a lower weight, and the result is a lower energy consumption.
- **Fault Resilience:** Due to the redundancy of the lexicographic structure, the data can be rerouted even in the event of a failure of a link or sensor.
- **Balanced Load:** Fibonacci weights aid to distribute communication loads and prevent congestion and facilitate the traffic flow.

5. Conclusion

The paper we study has provided a good foundation of Fibonacci-weighted lexicographic product graphs as the source of an Internet of Things network in Smart Cities. They make it possible for dispersed sensors to communicate effectively, consistently, and sustainably. The method increases robustness while lowering transmission costs. More broadly, this paradigm is flexible. It might cover areas like smart grids, healthcare monitoring, or urban mobility systems in addition to environmental monitoring. In each of these situations, improved communication and lower energy usage are still crucial.

6. Acknowledgments

The author, Rudrapati Bhuvaneswara Prasad, would like to express sincere gratitude to his research guide Prof. A. Mallikarjuna Reddy (Rtd.) for his constant support, valuable guidance, and encouragement throughout the course of this research.

The author also extends heartfelt thanks to Dr. G. Shobhalatha, for her insightful suggestions and continued support.

This work was carried out at the Department of Mathematics, Sri Krishnadevaraya University, Ananthapuramu, Andhra Pradesh.

Conflict of Interest

The author(s) declare that there is no conflict of interest regarding the publication of this paper. The author(s) confirm that there are no financial, personal, or professional relationships that could be construed as influencing the results or interpretations presented in this work.

References

1. K. R. Chandrasekar, *Geodetic numbers of tensor product and lexicographic product graphs*, *J. Discrete Math. Sci. Cryptogr.*, 2025. 10.1080/09728600.2024.2422535.
2. P. Schweitzer, *Iterated open neighborhood graphs and generalizations*, *Discrete Appl. Math.* 161 (2013), 1150–1162.
3. R. Govindaraju, M. S. Krishnamoorthy, N. Deo, *Fibonacci Networks*, *Fibonacci Q.* 32(4) (1994), 325–334.
4. P. Ferrer-Cid, J. García-Garreta, A. Cuzzocrea, *Graph-powered data quality in IoT monitoring platforms*, *J. Netw. Comput. Appl.*, 2025.
5. I. Dey, N. Marchetti, *Optimized Topology Control for IoT Networks*, arXiv:2410.03204, 2024.
6. K. Akrida *et al.*, *General position problems in strong and lexicographic products*, arXiv:2408.15951, 2024.
7. A. Ahadi, O. Giménez, M. Siggers, *Fibonacci-run graphs I: Basic properties*, *Discrete Appl. Math.* 305 (2021), 1–21.
8. G. Bhatnagar, *A weighted extension of Fibonacci numbers*, Preprint, 2023.

9. R. B. Prasad, A. M. Reddy, *Edge properties of lexicographic product graphs of open neighborhood graphs*, *Sci. Temper*, 2025.
10. S. Canoy Jr., J. Pabalinas, *The Neighborhood Graph of a Graph*, Preprint, 2015.
11. M. Zaman, N. Hossain, S. M. Liton, *A Review of IoT-Based Smart City Development and Management*, *Smart Cities* 7(3) (2024), 61.
12. Z. Wang, H. Chen, *A Survey on Knowledge-Graph Research in Smart Cities*, *ACM Comput. Surv.*, 2024.

Rudrapati Bhuvaneswara Prasad, Research Scholar
Department of Mathematics,
Sri Krishnadevaraya University, Ananthapuramu – 515 003,
INDIA.
E-mail address: `bhuvaneswaraprasad100@gmail.com`

and

A. Mallikarjuna Reddy (Rtd.), Research Supervisor
Department of Mathematics,
Sri Krishnadevaraya University, Ananthapuramu – 515 003,
INDIA.
E-mail address: `prof.malliavutala@gmail.com`

and

G. Shobhalatha,
Department of Mathematics,
Sri Krishnadevaraya University, Ananthapuramu – 515003,
INDIA.
E-mail address: `lathashobha91@gmail.com`