**EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS**
2025, Vol. 18, Issue 1, Article Number 5614
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

# Graph-Based Approach to the Radio Assignment Problem in Wireless Communication Networks with Applications in Cryptography

Nawal M. NourEldeen[1,2], Elsayed Badr[3,4,5], Ismail M. Hagag[6], Hanan Shabana[7,*]

[1] *Department of Mathematics, College of Science, Taibah University, Madinah, Kingdom of Saudi Arabia*
[2] *Department of Mathematics, Women's College of Arts, Science and Education, Ain Shams University, Egypt*
[3] *Scientific Computing Department, Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt*
[4] *The Egyptian School of Data Science (ESDS), Benha, Egypt*
[5] *Department of Computer Science, College of Information Technology, Misr University for Science and Technology, Giza,Egypt*
[6] *Information Systems Department, El Madina Higher Institute of Administration and Technology, Egypt*
[7] *Physics and Engineering Mathematics Department, Faculty of Electronic Engineering, Menoufia University, Menouf, 32952, Egypt*

**Abstract.** The radio labeling problem has numerous applications including military communications, satellite communication, mobile networks, and internet technology. This issue is NP-hard. A rapid algorithm for determining the upper bound of the radio number of a given graph is proposed in this paper. The superior performance of the proposed algorithm is attributed to Saha and Panigrahi (2012). The superior performance of the proposed algorithm in terms of CPU time can be attributed to Badr and Mousa(2019). The mathematical model demonstrates that the proposed algorithm exhibits superior performance in terms of both the maximum number of radios and CPU time. The proposed algorithm is ultimately assessed using a collection of networks that possess a range of characteristics. Besides that, we present the application of such algorithm in cryptography.

**2020 Mathematics Subject Classifications**: 94A60, 94A62, 05B30, 0570

**Key Words and Phrases**: Cryptography, Graph labeling, Mathematical models, NP-hard problem, Radio labeling.

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v18i1.5614

*Email addresses:* neldeen@taibahu.edu.sa (N. M. NourEldeen), alsayed.badr@fci.bu.edu.eg (E. Badr), Drismailhagag@gmail.com (I. M. Hagag), hananshabana22@gmail.com (H. Shabana)

 

## 1. Introduction

Broadcasting of radio and television, satellite communication, mobile networks, internet technology, military communications and many other services are all examples of wireless communication networks applications. The radio frequency (channel) given to transmitting stations from the available frequency spectrum (bandwidth) determines how effectively and efficiently communications take place inside a network. The extensive use of wireless services implies a faster rate of channel saturation, which makes it impossible to keep up with the growing demand for bandwidth. *The frequency assignment problem* (FAP) is the process of allocating channels (radio frequencies) to transmitters situated at various places while avoiding interference and making the most effective use of the available bandwidth. Graph labeling technique proved as an effective technique for modeling and solving FAP[18]. In graph modeling of FAP, A network of $n$ transmitters (or stations) is formulated as a graph $G$ of $n$ nodes; each node for a transmitter. Any two nodes are adjacent if their corresponding transmitters are the geographically close. The task is to assign a channel to each transmitter and avoiding interference. Consequently, Almost, The graph $G$ is referred as an interference graph. Here, the interference is closely related to the geographical position of the transmitting stations. The interference is increasing whenever the stations are geographical close. To avoid such interference, then we have to assign frequencies to stations such that difference between the assigned channels must be large enough. Hence, in solving FAP as a graph labeling the objective is to find a labeling for all the vertices of the interference graph such that the range of the assigned channels is minimized.

Griggs and Yeh [17] presented the first graph labeling technique, also known as $L(2,1)$ labeling or distance two labeling to address the frequency assignment problem. The graph labeling $L(2,1)$ of a simple graph $G$ is a non-negative real-valued function $L : V\ (G) \to [0,\ \infty)$ where $V(G)$ is the vertex set of the graph $G$. Whenever $u$ and $v$ are two vertices in $G$ such that the distance between $u$ and $v$ is 1, then $\mid L(u) - L(v) \mid \geq\ 2$, and whenever the vertices $u$ and $v$ are of distance 2 from each other, then $\mid L(u) - L(v) \mid \geq\ 1$.

A novel graph labeling method called radio Labeling was introduced in 2001 by Chartrand et al [11] along with a modified definition of $L(2,1)$-labeling. In radio labeling the function $L : V\ (G) \to N$, where $N$ is the set of integer numbers, is used to give the radio labeling for the vertices of the graph $G$. The function $L$ is said to be a radio labeling of the graph $G$ if it satisfies the following condition: $|L(u) - L(v)| \geq diam(G) + 1 - d(u,v)$ for each $u,\ v \in V(G)$, with $diam(G)$ being the diameter of $G$ and $d(u,v)$ being the shortest distance between the vertices $u,\ v$. The difference between the highest and lowest labels employed by a radio labeling $L$ is known as the span of the labeling $L$. The main objective of the radio labeling problem is to determine the minimum span across all over such labeling $L$ on the graph $G$. Such minimum span is called the radio number of $G$ or as $rn(G)$. Such radio number is important in measuring the efficiency usage of the available bandwidth of radio frequencies. Determining the radio number of an arbitrary graph has been demonstrated to be an NP-complete problem [19]. Authors of [30] introduced an algorithm that determines the lower and upper bounds of the radio number of a graph.

Badr and Moussa [6] proposed the development of Sahas' algorithm and introduced a novel mathematical model for the radio labeling application. Radio labeling problem has been studied for different families of graphs [4, 5, 20–25, 27, 28, 31, 32, 39]. On the other hand, for more details about the mathematical models, the reader is referred to [1–3, 7–10, 13–16, 29, 37, 38]. Finally, for more details about the theory of cryptography, the reader may refer to [12, 26, 33–36].

Obtaining the radio number of generic graphs is computationally challenging and it is NP-hard to determine, the radio number of general graphs of diameter 2 and its complexity is unknown in general. In literature, there are some algorithms for finding an upper bound for radio number of a given graph. The main challenge of such algorithms is determining the first vertex of the graph $G$ that will labeled by zero. In [30] Laxman Saha select a random vertex to label by zero. The label zero is fixed at that vertex through all steps of Sahas' algorithm. Badr[6] proposed an algorithm moves the label zero from the vertex to another vertex in order to get the minimum radio number. Therefore, it gets the better upper bound of the radio number for a given graph $G$. However, if Badr algorithm moves all labels from the vertex to another vertex the algorithm will become an exponential algorithm. Here, we claim to introduce an algorithm that give better upper bound and minimize the time of finding such upper bound.

In this paper, a fast algorithm determining the upper bound of the radio number of a given graph is proposed. The proposed algorithm overcomes the other which is due to Saha and Panigrahi [30]. According to CPU time, the proposed algorithm outperforms the other which is due to Badr and Mousa [6]. The superiority of the proposed algorithm on the mathematical model is introduced for both the upper bound of the radio number and CPU time. Finally, the proposed algorithm is evaluated by a set of networks, which have variety properties.

The rest of the paper is structured as follows Section 2 present the fundamentals of the proposed algorithm for finding the upper bound of the radio labeling for a graph $G$. Besides that, the mathematical model for determining the upper bound of the radio number of a graph is presented. A computational study for verification of the validity and efficiency of the proposed algorithm is presented in Section 3. A cryptography application is presented in Section 4. The conclusion of the paper and future work are presented in Section 5.

## 2. Methodology

In this section, we propose an algorithm for finding the upper bound of the radio labeling for a graph $G$. We explain the mechanism of the proposed algorithm and its time complexity. On the other hand, Theorem 1 is introduced to prove the validation of this algorithm.

The basic idea of radio number upper bound optimization is how to choose the vertex that has the labeling zero. In 2012, Saha and Panigrahi [30] presented an algorithm to find the upper bound and lower bound of a radio number for a graph. They fixed the zero label

for the first vertex while Badr and Moussa [6] proposed an algorithm that give an upper bound better than the results are due to Saha [30]. In [6] the label zero is moved from one vertex to another vertex in order to obtain a better upper bound for the radio number.

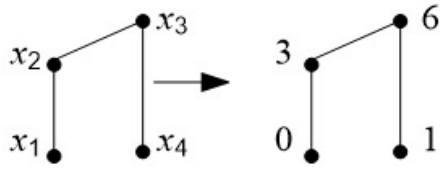


Figure 1: The label zero randomly selected and fixed at $x_1$ according to Laxman Saha [30].

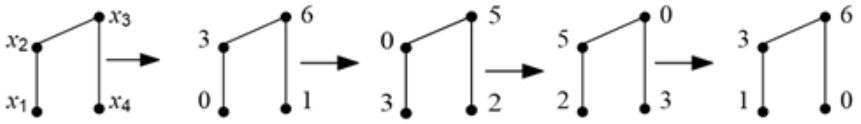


Figure 2: The label zero moves from $x_1$ to $x_4$ according to Badr [6].

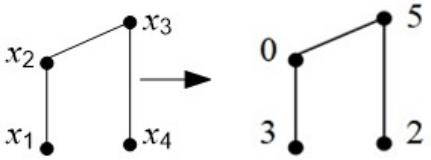


Figure 3: The label zero fixed at $x_2$ according to the proposed algorithm

Figure 1 shows that Saha and Panigrahi fixed the zero label for the first vertex while Figure 2 shows that Badr and Moussa move the zero label from one vertex to another vertex in order to obtain a better upper bound for the radio number. Finally, Figure 3 shows that label zero will assign to the center vertex of the given graph as we will describe in Section 2.1.

## 2.1. The description of the proposed algorithm

Let $G$ be a connected, simple, and undirected graph with $n$ vertices. Let $V(G)$ be the vertex set of $G$.

For a vertex $u \in V(G)$, let the sum of distances from the vertex $u$ to every other vertex in $G$ is $d^s(u)$. Thus, $d^s(u) = \sum_{v \in V(G)} d(u, v)$.

Let $X = \{d^s(u), \ \ \forall u \in V(G)\}$. A vertex $u$ is said to be the center of the graph $G$ if

$$d^s(u) = \min(X)$$

Hereafter, we use the center vertex of the graph $G$ to find the upper bound of the radio number of $G$. We aim to assign a radio labeling $L$ to these vertices such $L(v_0) < L(v_1) < L(v_2) < \ldots < L(v_{n-1})$. If $L$ is a radio labeling of $G$, then $|L(v_{j+1}) - L(v_j)| \geq diam(G) + 1 - d(v_{j+1}, v_j)$ for all $0 \leq j \leq n-2$. Consequently,

$$L(v_{j+1}) - L(v_j) = diam(G) + 1 - d(v_{j+1}, v_j) + \sigma_j$$

such that $\sigma_j$ is a positive integer. Moreover,

$$L(v_{j+1}) = l(v_j) + (diam(G) + 1) - d(v_{j+1}, v_j) + \sigma_j \text{ for all } 0 \leq j \leq n-2.$$

Therefore,

$L(v_{n-1}) = L(v_0) + (n-1)(diam(G)+1) - \sum_{j=0}^{n-2} d(v_{j+1}, v_j) + \sum_{j=0}^{n-2} \sigma_j$
As, $L(v_0) < L(v_1) < L(v_2) < \ldots < L(v_{n-1})$ then, $span(L) = L(v_{n-1})$.
Consequently, $rn(G) \leq L(v_n)$.
Our claim now is to minimize $\sum_{j=0}^{n-2} d(v_{j+1}, v_j) + \sum_{j=0}^{n-2} \sigma_j$ for all $0 \leq j \leq n-2$.

satisfying that

$|L(v_{j+1}) - L(v_j)| \geq diam(G) + 1 - d(v_{j+1}, v_j)$ ; $0 \leq j \leq n-2$.

The radio labeling of the vertex $v_1$ will be as follow

$$L(v_1) \geq L(v_0) + (diam(G) + 1) - d(v_1, v_0)$$

As we assume that $L(v_0) = 0$ then the upper bound of the value of $L(v_1)$ will depend only on the value of $d(v_1, v_0)$. The less the value of $d(v_1, v_0)$, the more value of $L(v_1)$. Hence, it is a good choice when we choose $v_0$ to be the center vertex of $G$. After that we work to label a vertex from $V(G)$ differ from $v_1$, $v_0$ and consider this vertex as $v_2$ satisfying that

$|L(v_2) - L(v_1)| \geq diam(G) + 1 - d(v_2, v_1)$ and

$$|L(v_2) - L(v_0)| \geq diam(G) + 1 - d(v_2, v_0)$$

Then we do the following computation, for each vertex $u \in V(G) - \{v_1, v_0\}$. Find

$$L(u) = \max\{L(v_0) + diam(G) + 1 - d(v_0, u), L(v_1) + diam(G) + 1 - d(v_1, u)\}$$

Consequently, the vertex $u$ will consider as $v_2$ and $L(v_0) < L(v_1) < L(v_2)$. To find the vertex $v_3$, we compute the below computations. For every vertex $u$ belongs to $V(G) - \{v_0, v_1, v_2\}$ compute

$$L(u) = \max\left\{\begin{array}{c} L(v_0) + diam(G) + 1 - d(v_0, u), \; L(v_1) + diam(G) + 1 - d(v_1, u), \\ L(v_2) + diam(G) + 1 - d(v_2, u) \end{array}\right\} \quad (1)$$

Then the vertex $v_3$ will be the vertex $u$ with minimum value of $L(u)$ from the computations proceed in Equation 1. The previous computations is proceed until the vertex $v_{n-1}$ is labeled.

Hereafter, we introduce an algorithm that output an upper bound of a radio labeling of a graph $G$.

**Algorithm 1. The Upper bound of the radio number of a given graph** $G$

**Input**: The adjacency matrix of the graph $G$ and the diameter $D$ of $G$.
**Output**: The upper bound of radio number of $G$.
**Begin**
1: For all $u \in V(G)$ compute, $d^s(u) = \sum_{v \in V(G)} d(u,v)$.
2: Let $min1 = \min_{u \in V(G)} \{d^s(u)\}$
3: Choose a vertex $u \in V(G)$, such that $d^s(u) = min1$
4. Assign, $L(u) = 0$ .
5:$X = \{u\}$ .
6: For all $v \in V(G) - X$, compute, $temp(v) = \max_{t \in X} \{L(t) + D + 1 - d(v,t)\}$
7: Let $min2 = \min_{v \in V(G)-X} \{temp(v)\}$
8: Choose a vertex $v \in V(G) - X$, where
$$temp(v) = min2$$
9: Assign, $L(v) = min2$
10: $X = X \cup \{v\}$ .
11: Repeat Step 6 to Step 10 until all vertices are labeled.
**Stop.**
**End.**

**Theorem 1.** *Algorithm 1 results in a radio labeling and an upper bound of the radio number for a graph G*

*Proof.* Let Algorithm 1 be at intermediate step $r$, where the vertices of $G$ that belong to the set $X = \{u_1, u_2, \ldots, u_r\}$ are radio labeled such the radio label of a vertex $u_i = L(u_i)$ for all $1 \leq i \leq r$. In the next step, to label a new vertex $v$, the Algorithm 1 picks a vertex $v \in V - X$ such that $temp(v) = \max_{t \in X} \{L(t) + D + 1 - d(v,t)\}$ is the minimum. According to Algorithm 1, $L(v) = temp(v)$. Let us call the vertex $v$ as $u_{r+1}$, we find that $L(u_{r+1}) \geq L(t) + D + 1 - d(u_{r+1}, u_i)$ for all $u_i \in X$. Thus, $|L(u_{r+1}) - L(u_i)| \geq D + 1 - d(u_{r+1}, u_i)$ for all $u_i \in X$. Therefore, Algorithm 1 results in a radio labeling for the given graph $G$ and an upper bound for radio number of $G$.

**Complexity of Algorithm 1.** step 1 has a loop which has $O(n)$ time complexity. Step 2 has $O(n)$ time complexity and step 3 has $O(n)$ time complexity. Both of steps 4 and 5 has $O(1)$ time complexity. In step 6, there is a nested loop which has $O(n^2)$ time complexity. Each of steps 7 and 8 has $O(n)$ time complexity. For steps 9 and 10 each has one operation which has 2 $O(1)$ time complexity. At the end, step 11 has $O(n^3)$ time complexity. Therefore, the proposed algorithm has the following time complexity,

$5O(n) + 4O(1) + O\left(n^2\right) + O\left(n^3\right) = O(n^3)$

### 2.2. The integer programming mathematical model (ILPM) for the radio labeling problem

A new mathematical formulation for the radio labeling problem is proposed. We next describe the problem of finding the radio labeling problem for a graph in terms of an integer programming problem [19]. Let $G$ be a connected graph of order $n$ with $V(G) = \{u_1, u_2, \cdots, u_n\}$ and let $D = [d_{ij}]$ be the distance matrix of $G$, that is $d_{ij} = d(u_i, u_j)$ for $1 \leq i, \ j \leq n$. We suppose that $v_i$ are the labels of the vertices $u_i$ such that $1 \leq i \leq n$. Now, we can introduce the mathematical model for the radio labeling problem as integer programming model. We define the function $F$ by

$$\min F = v_1 + v_2 + \cdots + v_n$$

Subject to: $|v_i - v_j| \geq diam + 1 - d(u_i, u_j)$ for $1 \leq i \leq n-1$; $2 \leq j \leq n$ and $i < j$
where $v_1, v_2, \cdots, v_n \in \{0, 1, 2, \ldots\}$
The radio number of the graph $G = \max_{1 \leq i \leq n} \{v_i\}$.

## 3. Results and discussion

We describe our numerical experiments and present computational results, which demonstrate the efficiency of the proposed algorithm on a set of test graphs (path, cycle, friendship graphs ($F_{3,k}$, $F_{4,k}$, $F_{5,k}$ ), $\Delta_k$ *-snake* and $kC_4$*-snake* and *triangular path* $P_{3,k}$). Compiled using python 3.11, all runs were carried out under macos, on MacBook Air with Apple M1 chip with 8-core CPU and 7-core GPU, 8GB RAM

In Tables 1-8 the abbreviations *UB1*, *UB2* and *UB3* are used to denote upper bounds of radio number due to Saha and Panigrahi [30], Badr and Moussa [6] and the proposed algorithm (algorithm 1) respectively.

**Definition 1.** *Let $k$ be a positive integer. A triangular snake graph denoted as $\Delta_k$* -snake *is a connected and simple graph of $k$-blocks and each block is cycle $C_3$ and the block-cut-point graph is a path.*

Figure 4 illustrates the graph $\Delta_k$ *-snake.*

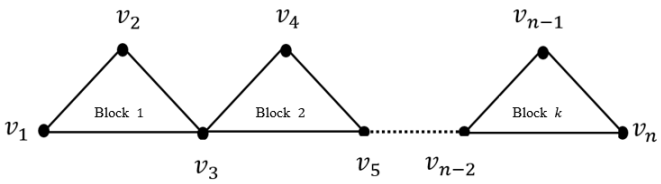


Figure 4: $\Delta_k$ *-snake*

**Definition 2.** *For given positive integers $k$, $m$, a friendship graph denoted as $F_{m,k}$ and represented as $k$ cycles (blocks) each of length $m$ and all have a one vertex common.*

For more illustration, Figure 5 shows the graph $F_{4,k}$ as an example of friendship graph.

**Definition 3.** *For a positive integer $k$, the graph denoted as $kC_4$-*snake *is a connected and simple graph of $k$-*blocks *and each block is cycle $C_4$ and the block-cut-point graph is a path.*

The graph *$kC_4$-snake* is shown in Figure 6.

**Definition 4.** *For a positive integer $k$, a triangular path $P_{3,k}$ is a connected and simple graph with $|V(P_{3,k})| = 3k$ and $|E(P_{3,k})| = 4k - 1$. It consists of a path $P_k$ of $k$ vertices such that every vertex from this path is a vertex in a cycle $C_3$.*

The illustration of $P_{3,k}$ is shown in Figure 7.

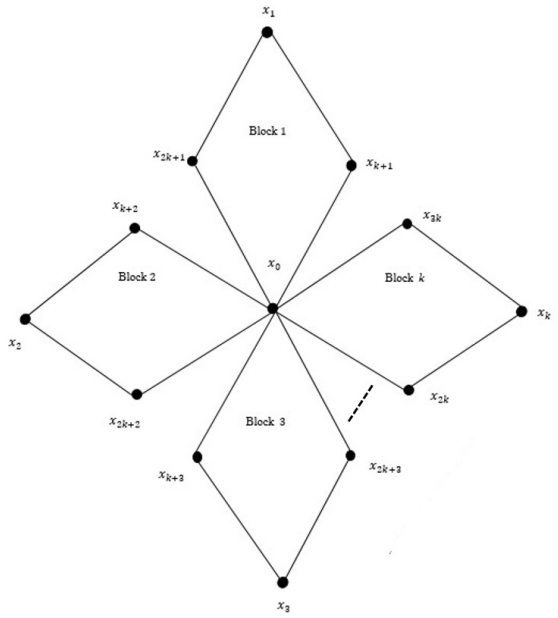


Figure 5: friendship $F_{4,k}$.

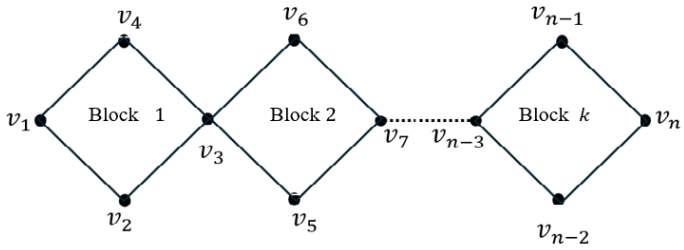


Figure 6: *$kC_4$-snake*

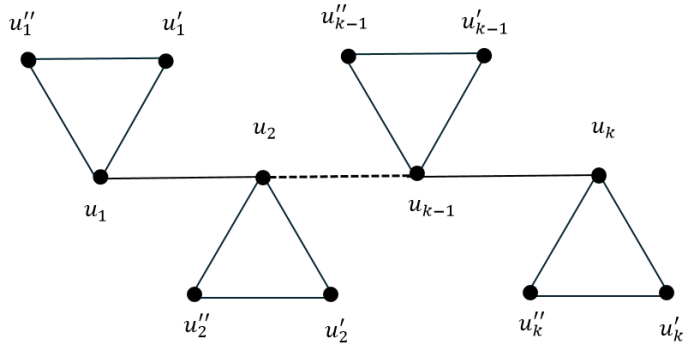


Figure 7: Triangular path $P_{3,k}$.

Tables 1-8 present the comparisons among four different algorithms, $UB1$, $UB2$, $ILPM$, and $UB3$, for path ($P_n$), cycle ($C_n$), friendships ($F_{3,k}$, $F_{4,k}$, $F_{5,k}$, ) triangular snakes $\Delta_k$-*snake* and $kC_4$-*snake* and triangular path ($P_{3,\mathrm{k}}$). The table provides various metrics, including the upper bound of the radio number $rn(G)$ and the $CPU$ time for each algorithm at different values of $n$. The first column (in Tables 1, 2 and second column in Tables 3-8) represents the value of $n$, which indicates the size (number of vertices) of the graph $G$. The second column (in Tables 1, 2 and third column in Tables 3-8) shows the value of $rn(G)$ obtained using the UB1 algorithm. Similarly, the further consecutive columns represent the values of $rn(G)$ obtained using the $UB2$ and $ILPM$ algorithms. The last column presents the values of $rn(G)$ obtained using the UB3 algorithm. The subsequent columns provide the $CPU$ time taken by each algorithm to compute the respective $rn(G)$ values for the given graph size.

Table 1: A comparison among $UB1$, $UB2$, $UB3$, and $ILPM$ for $G =$ path graph $P_n$

| $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|
| | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 0 | 0.000181 | 0 | 0.000353 | 0 | 0.000111 | 0 | 0.000033 |
| 2 | 1 | 0.000083 | 1 | 0.000454 | 1 | 0.000015 | 1 | 0.000041 |
| 3 | 3 | 0.000085 | 3 | 0.000733 | 4 | 0.000033 | 3 | 0.000058 |
| 4 | 6 | 0.000124 | 5 | 0.000014 | 9 | 0.00004 | 5 | 0.000077 |
| 5 | 11 | 0.000217 | 10 | 0.000267 | 16 | 0.000461 | 10 | 0.000126 |
| 6 | 15 | 0.000265 | 14 | 0.000531 | 25 | 0.000543 | 14 | 0.000209 |
| 7 | 22 | 0.000375 | 20 | 0.000705 | 36 | 0.000098 | 20 | 0.000308 |
| 8 | 28 | 0.000505 | 26 | 0.000934 | 49 | 0.000197 | 26 | 0.000372 |
| 9 | 37 | 0.000539 | 35 | 0.00123 | 64 | 0.000261 | 35 | 0.000504 |
| 10 | 45 | 0.000815 | 42 | 0.001666 | 81 | 0.000305 | 42 | 0.000674 |
| 11 | 56 | 0.000888 | 53 | 0.002232 | 100 | 0.000393 | 53 | 0.000828 |
| 12 | 66 | 0.001091 | 62 | 0.002816 | 121 | 0.000563 | 62 | 0.001044 |
| 13 | 80 | 0.001379 | 75 | 0.003461 | 144 | 0.000678 | 76 | 0.001316 |
| 14 | 91 | 0.001666 | 86 | 0.00451 | 169 | 0.000717 | 86 | 0.001645 |
| 50 | 1225 | 0.062379 | 1202 | 0.186789 | 2401 | 0.021464 | 1202 | 0.062123 |

Table 1 indicates that, for the path, the proposed algorithm is preferable to $UB1$. On the other hand, we discover that the proposed algorithm is preferable to $UB2$ in terms of execution time, while it is equivalent to $UB2$ until $n$ is less than or equal to 12. When $n$ is greater than or equal to 13, the algorithm $UB2$ oscillates between equality with the proposed algorithm and superiority. The proposed algorithm is preferable to $ILPM$ . As we can observe, for $n = 1$, all algorithms produce an $rn(G)$ value of 0. Each algorithm takes a different amount of CPU time to compute this result. $UB1$ takes the least time,

followed by $UB2$, $ILPM$, and $UB3$. As $n$ increases, the $rn(G)$ values also increase for all algorithms. This indicates that the complexity of the graph structure increases as the size of the path graph grows. The CPU time also increases as the graph size increases. This is expected since larger graphs require more computational effort to compute the $rn(G)$ values. For larger values of $n$, such as $n = 50$, the CPU times become significantly higher for all algorithms. This indicates that the computational complexity of these algorithms increases with the size of the input graph.

Table 2: A comparison among *UB*1, *UB*2,*UB*3 *and ILPM* for $G = cycle$

| $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|
| | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 0 | 0.000198 | 0 | 0.0000348 | - | 0.0000418 | 0 | 0.0000329 |
| 2 | 1 | 0.0000478 | 1 | 0.0000434 | - | 0.0000251 | 1 | 0.0000362 |
| 3 | 2 | 0.0000738 | 2 | 0.0000862 | 2 | 0.0000426 | 2 | 0.0000537 |
| 4 | 4 | 0.000116 | 4 | 0.000157 | 6 | 0.0000646 | 4 | 0.0000757 |
| 5 | 4 | 0.000127 | 4 | 0.00027 | 8 | 0.0001079 | 4 | 0.000121 |
| 6 | 7 | 0.000183 | 7 | 0.000441 | 15 | 0.0001039 | 7 | 0.000150 |
| 7 | 10 | 0.000265 | 9 | 0.000644 | 18 | 0.0001659 | 10 | 0.000245 |
| 8 | 13 | 0.000371 | 13 | 0.000977 | 28 | 0.0002579 | 13 | 0.000369 |
| 9 | 13 | 0.000512 | 12 | 0.001314 | 32 | 0.0004969 | 13 | 0.000508 |
| 10 | 17 | 0.000686 | 17 | 0.001701 | 45 | 0.0005569 | 17 | 0.000668 |
| 11 | 21 | 0.00088 | 20 | 0.002198 | 50 | 0.0008419 | 21 | 0.000853 |
| 12 | 26 | 0.001096 | 26 | 0.002863 | 66 | 0.0010169 | 26 | 0.001028 |
| 13 | 25 | 0.001412 | 24 | 0.003559 | 72 | 0.0011609 | 25 | 0.001272 |
| 14 | 31 | 0.001693 | 31 | 0.004457 | 91 | 0.0014649 | 31 | 0.001576 |
| 50 | 337 | 0.062019 | 337 | 0.186451 | 1225 | 0.0320259 | 337 | 0.062002 |

Table 2 indicates that, for the cycle, the proposed algorithm is preferable to $UB1$. On the other hand, we discover that the proposed algorithm is preferable to $UB2$ in terms of execution time, while it is equivalent to $UB2$ until $n$ is less than or equal to 12. When $n$ is greater than or equal to 13, the algorithm $UB2$ oscillates between equality with the proposed algorithm and superiority. The proposed algorithm is preferable to *ILPM*.

Tables 3-5 indicate that, for the friendships $F_{3,k}$, $F_{4,k,}$ *and* $F_{5,k}$ , the proposed algorithm is preferable to $UB1$. On the other hand, we discover that the proposed algorithm is preferable to $UB2$ in terms of execution time, while it is equivalent to $UB2$ until $n$ is less than or equal to 12. For $F_{3,k}$, Table 3 *shows that* the algorithms $UB2$ and the proposed algorithm are equivalent except at $k = 3, 5, 7$. For $F_{4,k,}$ *and* $F_{5,k}$, Tables 3-*4 show that* the algorithms $UB2$ and the proposed algorithm are equivalent except at $k = 2$.
The proposed algorithm is preferable to *ILPM*.

Table 3: A comparison among *UB*1, *UB*2, *UB*3 *and ILPM* for G = friendship graph $F_{3,k}$

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 3 | 2 | 0.000352 | 2 | 0.000117 | 4 | 0.000075 | 2 | 0.000116 |
| 2 | 5 | 5 | 0.000199 | 5 | 0.000277 | 7 | 0.000037 | 5 | 0.000148 |
| 3 | 7 | 8 | 0.000347 | 7 | 0.000675 | 10 | 0.000172 | 8 | 0.000283 |
| 4 | 9 | 9 | 0.000542 | 9 | 0.001369 | 13 | 0.000408 | 9 | 0.000519 |
| 5 | 11 | 12 | 0.000964 | 11 | 0.002313 | 16 | 0.000847 | 12 | 0.000958 |
| 6 | 13 | 13 | 0.001391 | 13 | 0.00376 | 19 | 0.000291 | 13 | 0.001402 |
| 7 | 15 | 16 | 0.002084 | 15 | 0.005837 | 22 | 0.000861 | 16 | 0.001972 |
| 8 | 17 | 17 | 0.002893 | 17 | 0.008048 | 25 | 0.001774 | 17 | 0.002885 |
| 9 | 19 | 20 | 0.004048 | 19 | 0.011799 | 28 | 0.002924 | 20 | 0.004035 |
| 10 | 21 | 21 | 0.00543 | 21 | 0.015302 | 31 | 0.004015 | 21 | 0.005126 |
| 11 | 23 | 24 | 0.007658 | 23 | 0.020273 | 34 | 0.005977 | 24 | 0.007088 |
| 12 | 25 | 25 | 0.009249 | 25 | 0.025301 | 37 | 0.007643 | 25 | 0.008754 |
| 13 | 27 | 28 | 0.011044 | 27 | 0.03207 | 40 | 0.000837 | 28 | 0.010948 |
| 14 | 29 | 29 | 0.014288 | 29 | 0.039837 | 43 | 0.003468 | 29 | 0.013579 |
| 50 | 101 | 101 | 0.675397 | 101 | 1.773485 | 151 | 0.263206 | 101 | 0.574317 |

Table 4: A comparison among *UB*1, *UB*2, *UB*3 *and ILPM* for G = friendship graph $F_{4,k}$

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 3 | 4 | 0.000115 | 4 | 0.000115 | 5 | 0.000063 | 4 | 0.000114 |
| 2 | 5 | 16 | 0.00024 | 15 | 0.00042 | 17 | 0.000128 | 16 | 0.000239 |
| 3 | 7 | 22 | 0.000676 | 22 | 0.001321 | 23 | 0.000276 | 22 | 0.000636 |
| 4 | 9 | 29 | 0.001434 | 29 | 0.003049 | 30 | 0.000525 | 29 | 0.001387 |
| 5 | 11 | 36 | 0.002458 | 36 | 0.006385 | 37 | 0.001315 | 36 | 0.002426 |
| 6 | 13 | 43 | 0.004054 | 43 | 0.011824 | 44 | 0.002745 | 43 | 0.003856 |
| 7 | 15 | 50 | 0.006445 | 50 | 0.019918 | 51 | 0.005039 | 50 | 0.00615 |
| 8 | 17 | 57 | 0.008756 | 57 | 0.031611 | 58 | 0.007677 | 57 | 0.008788 |
| 9 | 19 | 64 | 0.013099 | 64 | 0.047747 | 65 | 0.011479 | 64 | 0.012702 |
| 10 | 21 | 71 | 0.01666 | 71 | 0.069004 | 72 | 0.011591 | 71 | 0.016403 |
| 11 | 23 | 78 | 0.022505 | 78 | 0.09524 | 79 | 0.015292 | 78 | 0.02259 |
| 12 | 25 | 85 | 0.029908 | 85 | 0.130633 | 86 | 0.017542 | 85 | 0.028653 |
| 13 | 27 | 92 | 0.036358 | 92 | 0.174968 | 93 | 0.025987 | 92 | 0.037098 |
| 14 | 29 | 99 | 0.04407 | 99 | 0.228926 | 100 | 0.03328 | 99 | 0.044391 |
| 50 | 101 | 351 | 2.107236 | 351 | 27.88631 | 352 | 0.771716 | 351 | 1.882827 |

Table 5: A comparison among *UB*1, *UB*2, *UB*3 *and ILPM* for G = friendship graph $F_{5,k}$

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 5 | 4 | 0.00032 | 4 | 0.00031 | 14 | 0.000071 | 4 | 0.000182 |
| 2 | 9 | 18 | 0.000517 | 17 | 0.001497 | 24 | 0.000457 | 18 | 0.000108 |
| 3 | 13 | 25 | 0.001336 | 25 | 0.003841 | 35 | 0.000315 | 25 | 0.001426 |
| 4 | 17 | 33 | 0.002867 | 33 | 0.008418 | 46 | 0.001768 | 33 | 0.002879 |
| 5 | 21 | 41 | 0.005361 | 41 | 0.015528 | 57 | 0.004308 | 41 | 0.005419 |
| 6 | 25 | 49 | 0.008805 | 49 | 0.025917 | 68 | 0.007776 | 49 | 0.008887 |
| 7 | 29 | 57 | 0.013459 | 57 | 0.039922 | 79 | 0.002785 | 57 | 0.013896 |
| 8 | 33 | 65 | 0.01975 | 65 | 0.058754 | 90 | 0.008407 | 65 | 0.019518 |
| 9 | 37 | 73 | 0.027393 | 73 | 0.082683 | 101 | 0.016351 | 73 | 0.027462 |
| 10 | 41 | 81 | 0.038485 | 81 | 0.110506 | 112 | 0.025411 | 81 | 0.036522 |
| 11 | 45 | 89 | 0.048896 | 89 | 0.14308 | 123 | 0.038162 | 89 | 0.049273 |
| 12 | 49 | 97 | 0.060881 | 97 | 0.180638 | 134 | 0.050256 | 97 | 0.061367 |
| 13 | 53 | 105 | 0.078234 | 105 | 0.230682 | 145 | 0.067696 | 105 | 0.078807 |
| 14 | 57 | 113 | 0.097254 | 113 | 0.285646 | 156 | 0.085658 | 113 | 0.096769 |
| 50 | 201 | 401 | 4.265029 | 401 | 12.58799 | 552 | 3.06101 | 401 | 4.072021 |

Tables 6 and 7 indicate that, for the $\Delta_k$- *snakes* and, $kC_4$*-snakes*, the proposed algorithm is preferable to. On the other hand, we discover that the proposed algorithm is preferable to $UB2$ in terms of execution time, while it is equivalent to $UB2$ until $n$ is less than or equal to 12. For $\Delta_k$- *snakes* Table 6 shows *that* the algorithms $UB2$ and the proposed algorithm are equivalent. For $kC_4$*-snakes,* Table 7 shows *that* the algorithms $UB2$ and the proposed algorithm are equivalent except at $k = 2, 5, 7, 13$. The proposed algorithm is preferable to *ILPM*.

Table 6: A comparison among *UB*1, *UB*2, *UB*3 *and ILPM* for triangular snakes $\Delta_k$

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 3 | 2 | 0.000297 | 2 | 0.000184 | 2 | 0.000093 | 2 | 0.000204 |
| 2 | 5 | 5 | 0.000138 | 5 | 0.000293 | 6 | 0.000035 | 5 | 0.000126 |
| 3 | 7 | 12 | 0.000283 | 11 | 0.000756 | 14 | 0.000082 | 11 | 0.000263 |
| 4 | 9 | 18 | 0.000531 | 17 | 0.001378 | 26 | 0.000032 | 17 | 0.000513 |
| 5 | 11 | 30 | 0.000871 | 28 | 0.002301 | 42 | 0.000037 | 28 | 0.000248 |
| 6 | 13 | 39 | 0.001379 | 37 | 0.003772 | 62 | 0.000038 | 37 | 0.001249 |
| 7 | 15 | 56 | 0.002113 | 52 | 0.005715 | 86 | 0.000097 | 52 | 0.002108 |
| 8 | 17 | 68 | 0.003721 | 65 | 0.008235 | 114 | 0.000314 | 65 | 0.003005 |
| 9 | 19 | 90 | 0.004074 | 85 | 0.011566 | 146 | 0.000321 | 85 | 0.004232 |
| 10 | 21 | 105 | 0.005461 | 101 | 0.015212 | 182 | 0.000494 | 101 | 0.005805 |
| 11 | 23 | 132 | 0.007225 | 125 | 0.020041 | 222 | 0.000584 | 125 | 0.007395 |
| 12 | 25 | 150 | 0.008868 | 145 | 0.025406 | 266 | 0.000696 | 145 | 0.008807 |
| 13 | 27 | 182 | 0.010933 | 174 | 0.031775 | 314 | 0.000751 | 174 | 0.010862 |
| 14 | 29 | 203 | 0.013203 | 197 | 0.040166 | 366 | 0.000277 | 197 | 0.013388 |
| 50 | 101 | 2525 | 0.661005 | 2501 | 1.575407 | 4902 | 0.000995 | 2501 | 0.563346 |

Table 7: A comparison among *UB*1, *UB*2, *UB*3 *and ILPM* for *G= kC₄-snakes* graph

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 4 | 4 | 0.000359 | 4 | 0.000182 | 4 | 0.000027 | 4 | 0.000138 |
| 2 | 7 | 17 | 0.00029 | 15 | 0.00069 | 17 | 0.000199 | 16 | 0.00021 |
| 3 | 10 | 33 | 0.000686 | 31 | 0.001912 | 40 | 0.000604 | 31 | 0.000615 |
| 4 | 13 | 58 | 0.001367 | 55 | 0.003751 | 75 | 0.000267 | 55 | 0.001338 |
| 5 | 16 | 86 | 0.002531 | 83 | 0.006906 | 122 | 0.001308 | 85 | 0.002419 |
| 6 | 19 | 122 | 0.004264 | 117 | 0.011311 | 181 | 0.002739 | 117 | 0.00385 |
| 7 | 22 | 162 | 0.006756 | 157 | 0.017645 | 252 | 0.00501 | 158 | 0.006121 |
| 8 | 25 | 212 | 0.009442 | 205 | 0.025285 | 335 | 0.007432 | 205 | 0.008543 |
| 9 | 28 | 266 | 0.012434 | 257 | 0.035349 | 469 | 0.001122 | 257 | 0.012233 |
| 10 | 31 | 325 | 0.016581 | 316 | 0.048055 | 579 | 0.004575 | 316 | 0.015686 |
| 11 | 34 | 389 | 0.022227 | 379 | 0.063325 | 656 | 0.009842 | 379 | 0.020953 |
| 12 | 37 | 462 | 0.028546 | 451 | 0.081115 | 787 | 0.016127 | 451 | 0.027238 |
| 13 | 40 | 538 | 0.034988 | 527 | 0.101463 | 873 | 0.022944 | 529 | 0.034055 |
| 14 | 43 | 622 | 0.043549 | 609 | 0.12424 | 981 | 0.030786 | 609 | 0.041897 |
| 50 | 151 | 7625 | 1.866284 | 7576 | 5.209234 | 14657 | 0.654571 | 7576 | 1.765682 |

Table 8 indicates that, for the path triangular $P_{3,k}$, the proposed algorithm is preferable to $UB1$. On the other hand, we discover that the proposed algorithm is preferable to $UB2$ in terms of execution time, while it is equivalent to $UB2$ until $n$ is less than or equal to 12. For $P_{3,k}$ Table 8 shows *that* the algorithms $UB2$ and the proposed algorithm are equivalent. The proposed algorithm is preferable to *ILPM*.

Table 8: A comparison among *UB1, UB2, UB3 and ILPM* for $G = triangular\ path\ graph$

| $K$ | $n$ | $UB1$ | | $UB2$ | | $ILPM$ | | $UB3$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time | $rn(G)$ | CPU Time |
| 1 | 3 | 2 | 0.000127 | 2 | 0.000123 | 2 | 0.000027 | 2 | 0.0000887 |
| 2 | 6 | 8 | 0.000198 | 8 | 0.000523 | 12 | 0.000234 | 8 | 0.000193 |
| 3 | 9 | 18 | 0.000543 | 17 | 0.001477 | 26 | 0.0003968 | 17 | 0.000519 |
| 4 | 12 | 32 | 0.001123 | 29 | 0.003144 | 46 | 0.0004982 | 29 | 0.001085 |
| 5 | 15 | 46 | 0.002093 | 45 | 0.005782 | 72 | 0.0005294 | 45 | 0.002035 |
| 6 | 18 | 65 | 0.003492 | 61 | 0.009949 | 104 | 0.0006289 | 61 | 0.003373 |
| 7 | 21 | 86 | 0.005367 | 82 | 0.016159 | 142 | 0.0007928 | 82 | 0.005314 |
| 8 | 24 | 110 | 0.007732 | 105 | 0.023089 | 186 | 0.0008923 | 105 | 0.007500 |
| 9 | 27 | 137 | 0.010508 | 134 | 0.032466 | 236 | 0.0019276 | 134 | 0.010086 |
| 10 | 30 | 167 | 0.015182 | 161 | 0.043037 | 292 | 0.0029187 | 161 | 0.014900 |
| 11 | 33 | 200 | 0.019532 | 198 | 0.059199 | 354 | 0.0030983 | 198 | 0.019228 |
| 12 | 36 | 236 | 0.025369 | 229 | 0.075771 | 422 | 0.0049876 | 229 | 0.025317 |
| 13 | 39 | 277 | 0.033481 | 274 | 0.094781 | 496 | 0.005691 | 274 | 0.031756 |
| 14 | 42 | 317 | 0.039851 | 309 | 0.115118 | 576 | 0.006892 | 309 | 0.039622 |
| 50 | 150 | 3827 | 1.840115 | 3801 | 5.127259 | 7452 | 0.26579 | 3801 | 1.702862 |

## 4. Application of radio labeling in cryptography

In this section, we claim to use the upper bound of radio number of a graph obtained from Algorithm 1 for Encryption and decryption. We will consider the Triangular Path graph for the proposed cryptography approach.

### 4.1. Cryptography approach

Let's the sender and the receiver wish to communicate in secure manner. Assume that the letters A through Z in the English alphabet make up the message M, with spaces separating each word. Let the numerical counterparts of each letter in the English alphabet A, B,..., and Z be 1, 2,..., and 26 in this cryptosystem. Let 0 represent empty space and punctuation is disregarded. We introduce a matrix-based cryptographic technique. Thus

only both of sender and receiver are know the key, which is an invertible matrix. In this section, we suggest a method for creating such a matrix using upper bound of radio number of the triangular path graph $P_{3,k}$.

Suppose the length of plaintext $M$ is $k$. Then, we consider triangular path $P_{3,k}$. Let Algorithm 1 produce that the upper bound of radio number of $P_{3,k}$ that is $rn\,(P_{3,k}\,)$. After that, a $2x2$ matrix $A$ is constructed as follows

$$A = \begin{bmatrix} rn\,(P_{3,k}\,) - 2 & rn\,(P_{3,k}\,) - 3 \\ rn\,(P_{3,k}\,) & rn\,(P_{3,k}\,) - 1 \end{bmatrix} mod\ 27$$

As the determinant of $A$ does not equals 0. Then $A$ is an invertible matrix. As a result, the sender may decide to utilize the matrix $A$ for encryption while the receiver may use the matrix $A^{-1}$ for decryption, where

$$A^{-1} = \frac{1}{2} \begin{bmatrix} rn\,(P_{3,k}\,) - 1 & 3 - rn\,(P_{3,k}\,) \\ -rn\,(P_{3,k}\,) & rn\,(P_{3,k}\,) - 2 \end{bmatrix} mod\ 27$$

## 4.2. Cryptography algorithms

Hereafter, the encryption and decryption processes are described in details.

**Encryption Algorithm**

**Input**: A message M of length $k$
**Output**: A cipher text $C$ of the message M.
**Begin**.

**1.**Consider triangular path $P_{3,k}$.
**2.**Use Algorithm 1 to find the upper bound of radio number of $P_{3,k}$.
**3.** If Algorithm 1 results the upper bound of radio number of $P_{3,k}$ as $\alpha$ then construct an 2x2 matrix $A = \begin{bmatrix} \alpha - 2 & \alpha - 3 \\ \alpha & \alpha - 1 \end{bmatrix} mod\ 27$.
**4.** Convert each letter (including space) in the plaintext M to its equivalent number.
**5.** Make this string of numbers as a set of ordered pairs. We pair the last number of the string with 0 whenever, the string has odd length.
**6.** An ordered $(x,\ y)$ is encrypted to $\begin{bmatrix} x' \\ y' \end{bmatrix} = A \begin{bmatrix} x \\ y \end{bmatrix} mod\ 27$.
**7.** Represent numbers in each encrypted $\begin{bmatrix} x' \\ y' \end{bmatrix}$ by space or its equivalent character in the English alphabet.
**8.** Form the cipher text $C$ by appending 0 or 1 at the end of the sequence of characters produced in step 7. Appending 0 indicates that the message M is of length, even and 1 indicates that the message M has length is of length, odd.

**End**

**Decryption Algorithm.**

**Input:** A cipher text $C$ of length $w$
**Output:** A plaintext (Original message) M of $C$.
**Begin**

**1.** If 0 is found at the end of $C$, obtain the upper bound of radio number $\alpha$ of triangular path $P_{3,w}$. Using algorithm 1. If 1 is found at the end of $C$, use algorithm 1 to obtain the upper bound of radio number, $\alpha$ of triangular path $P_{3,w-1}$.

**2**. Build the key matrix for decryption that is $B = \frac{1}{2}\begin{bmatrix} \alpha - 1 & 3 - \alpha \\ -\alpha & \alpha - 2 \end{bmatrix} mod\ 27.$

**3**. Convert the characters and spaces in $C$ to ordered pairs of their equivalent numerical values

**4.** Each ordered pair $(x',\ y')$ is decrypted to $\begin{bmatrix} x \\ y \end{bmatrix} = B \begin{bmatrix} x' \\ y' \end{bmatrix} mod\ 27.$

**5.** Represent numbers in each matrix $\begin{bmatrix} x \\ y \end{bmatrix}$ by space or its equivalent character of $x$ and $y$ in the English alphabet to deduce the plaintext M of the received ciphertext $C$.

**End**

For more illustration of the cryptography approach, we pose the following example.

**Example 1.** *Let the sender wants to send the message "PAR'' to the receiver then encryption process is done as follow*

1. Since the length of the original message is 3, we use algorithm 1 to get the upper bound of radio number of $P_{3,3}$.
2. Following Table -8, k=3, UB3, the upper bound of radio number of $P_{3,3}$ is 17. hence, construct an 2x2 matrix $A = \begin{bmatrix} 15 & 14 \\ 17 & 16 \end{bmatrix}$.
3. Represent each character (including space) in the plaintext M by its numerical equivalent. That is 16, 1, 18.
4. Convert this string of numbers to a set of ordered pairs as (16,1), (18,0).
5. $(16,\ 1)$, is encrypted as $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 15 & 14 \\ 17 & 16 \end{bmatrix} \begin{bmatrix} 16 \\ 1 \end{bmatrix} mod\ 27. = \begin{bmatrix} 11 \\ 18 \end{bmatrix}$ and the ordered pair (18,0) will be $\begin{bmatrix} 15 & 14 \\ 17 & 16 \end{bmatrix} \begin{bmatrix} 18 \\ 0 \end{bmatrix} mod\ 27. = \begin{bmatrix} 0 \\ 9 \end{bmatrix}$
6. Finally, convert these ordered pairs to equivalent English letter that is $11 \equiv K,\ 18 \equiv R,\ 0 \equiv space,\ 9 \equiv I$.
7. Finally, the cipher text $C = KR \bigtriangledown I1$. Here**,** we append 1 at the end of the sequence of characters produced in step 6 to inform the receiver that the original message has an odd length. On the other hand, we use the symbol $\bigtriangledown$ as indication for the space.

**At the receiver, the decryption process performed as follows**

As the received message is of length $w = 4$ and has number 1 at the end, then the decryption process use the upper bound of radio number of $P_{3,w-1} = P_{3,3}$ that is 17 following Table-8, $k$=3, $UB3$.

Build the key matrix for decryption $B = \frac{1}{2}\begin{bmatrix} 16 & -14 \\ -17 & 15 \end{bmatrix} mod\, 27 = \begin{bmatrix} 8 & 20 \\ 5 & 21 \end{bmatrix}$. Converting the characters and spaces in C to their equivalent numerical values and making ordered pairs yields ordered pairs (11, 18) and (0, 9). Doing a decrypting for each ordered pair as follows

(11,18) will be $\begin{bmatrix} 8 & 20 \\ 5 & 21 \end{bmatrix}\begin{bmatrix} 11 \\ 18 \end{bmatrix} = \begin{bmatrix} 16 \\ 1 \end{bmatrix}$ and

(0,9) will be $\begin{bmatrix} 8 & 20 \\ 5 & 21 \end{bmatrix}\begin{bmatrix} 0 \\ 9 \end{bmatrix} = \begin{bmatrix} 18 \\ 0 \end{bmatrix}$.

Representing the entries of the matrices $\begin{bmatrix} 16 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 18 \\ 0 \end{bmatrix}$ to their equivalent character (or space) that is $\begin{bmatrix} 16 \\ 1 \end{bmatrix} \Rightarrow PA$ and $\begin{bmatrix} 18 \\ 0 \end{bmatrix} \Rightarrow R$. Hence, the plaintext is M=PAR**.**

## 5. Conclusion

This paper proposed a fast algorithm for determining the upper bound of the radio number of a given graph. The proposed algorithm was superior to that of Saha and Panigrahi (2012). According to CPU time, the proposed algorithm outperformed the other which is due to Badr and Mousa (2019). The superiority of the proposed algorithm on the mathematical model was introduced for both the upper bound of the radio number and CPU time. An application of radio labeling for cryptography is presented. In the future, we will seek a new algorithm based on selecting the vertex that accepts the label zero so that it is efficient and outperforms the existing algorithms. In addition to testing such an algorithm on graphs with varying specifications for number of vertices, diameter, symmetry, etc., the algorithm must also be examined on graphs with varying specifications for symmetry, diameter, etc. On other hand, wireless technologies have been recently used in a variety of services. Hence, it is more economical to try reusing the frequencies. In Future we claim to enhance our algorithm to work on this issue.

**Declaration of interests**: The authors declare no conflict of interest.

## References


[1] B. D. Acharya and M. K. Gill. On the index of gracefulness of a graph and the gracefulness of two-dimensional square lattice graphs. *Indian Journal of Mathematics*, 23:81–94, 1981.

[2] A. H. Alkasasbeh, E. Badr, H. Attiya, and H. M. Shabana. Radio number for friendship communication networks. *Mathematics*, 11:4232, 2023.

[3] E. Badr, M. Abdul Salam, S. Almotairi, and H. Ahmed. From linear programming approach to metaheuristic approach: Scaling techniques. *Complexity*, page Article ID 9384318, 2021.

[4] E. Badr, S. Almotairi, A. Elrokh, A. Abdel-Hady, and B. Almutairi. An integer linear programming model for solving radio mean labeling problem. *IEEE Access*, 8:162343–162349, 2020.

[5] E. Badr, S. Nada, M. Al-Shamiri, A. Abdel-Hay, and A. ELrokh. A novel mathematical model for radio mean square labeling problem. *Journal of Mathematics*, page Article ID 3303433, 2022.

[6] E. M. Badr and M. I. Moussa. An upper bound of radio k-coloring problem and its integer linear programming model. *Wireless Networks*, 26(7):4955–4964, 2020.

[7] Elsayed Badr, Mohamed EL-Hakeem, Enas E El-Sharawy, and Thowiba E Ahmed. An efficient algorithm for decomposition of partially ordered sets. *Journal of Mathematics*, 2023(1):9920700, 2023.

[8] Elsayed Badr, IM Selim, Hoda Mostafa, and Hala Attiya. An integer linear programming model for partially ordered sets. *Journal of Mathematics*, 2022(1):7660174, 2022.

[9] G. S. Bloom and S. W. Golomb. *Applications of Numbered Undirected Graphs*, volume 65. Proceedings of the Institute of Electrical and Electronics Engineers, 1977.

[10] J. Bosák. *Cyclic Decompositions, Vertex Labellings and Graceful Graphs*. Kluwer Academic Publishers, 1990.

[11] G. Chartrand, D. Erwin, P. Zhang, and F. Harary. Radio labeling of graphs. *Bulletin of the Institute of Combinatorics and its Applications*, 33:77–85, 2001.

[12] A. ELrokh, A. E. Badr, M. M. A. Al-Shamiri, and S. Ramadhan. Upper bounds of radio number for triangular snake and double triangular snake graphs. *Journal of Mathematics*, page Article ID 3635499, 2022.

[13] R. Frucht. Graceful numbering of wheels and other related graphs. *Annals New York Academy of Sciences*, 319:219–229, 1979.

[14] R. Frucht and J. A. Gallian. Labeling prisms. *Ars Combinatorica*, 26:69–82, 1988.

[15] J. A. Gallian. A dynamic survey of graph labeling. *The Electronic Journal of Combinatorics*, pages 1–95, 2023.

[16] R. L. Graham and N. J. A. Sloane. On additive bases and harmonious graphs. *SIAM Journal on Algebraic and Discrete Methods*, 1:382–404, 1980.

[17] J. Griggs and R. Yeh. Labelling graphs with a condition at distance 2. *SIAM Journal on Discrete Mathematics*, 5(4):586–595, 1992.

[18] W. K. Hale. Frequency assignment: Theory and applications. *Proc. IEEE*,

68(12):1497–1514, 1980.

[19] M. Kchikech, R. Khennoufa, and O. Togni. Linear and cyclic radio k-labelings of trees. *Discussiones Mathematicae Graph Theory*, 27(1):105–123, 2007.

[20] X. Li, V. Mak, and S. Zhou. Optimal radio labelings of complete m-array trees. *Discrete Appl. Math.*, 158:507–515, 2010.

[21] D. D.-F. Liu. Radio number for trees. *Discrete Math.*, 308:1153–1164, 2008.

[22] D. D.-F. Liu and M. Xie. Radio number for square of cycles. *Congr. Numer.*, 169:105–125, 2004.

[23] D. D.-F. Liu and M. Xie. Radio number for square paths. *Ars Combin.*, 90:307–319, 2009.

[24] D. D.-F. Liu and X. Zhu. Multi-level distance labelings for paths and cycles. *SIAM J. Discrete Math.*, 19:610–621, 2005.

[25] M. Morris-Rivera, M. Tomova, C. Wyels, and Y. Yeager. The radio number of $c_n \times c_n$. *Ars Combin.*, 103:81–96, 2012.

[26] M. I. Moussa and E. M. Badr. A data hiding algorithm based on dna and elliptic curve cryptosystems. *Journal of Information Hiding and Multimedia Signal Processing*, 10(3), September 2019.

[27] J. P. Ortiz, P. Martinez, M. Tomova, and C. Wyels. Radio numbers of some generalized prism graphs. *Discuss. Math. Graph Theory*, 31:45–62, 2011.

[28] V. S. Reddy and V. K. Iyer. Upper bounds on the radio number of some trees. *Int. J. Pure and Applied Math.*, 71(2):207–215, 2011.

[29] A. Rosa. On certain valuations of the vertices of a graph. In *Theory of Graphs, International Symposium, Rome, July 1966*, pages 349–355. Gordon and Breach, 1967.

[30] L. Saha and P. Panigrahi. A graph radio k-coloring algorithm. In *Lecture Notes in Computer Science*, volume 7643, pages 125–129, 2012.

[31] L. Saha and P. Panigrahi. On the radio number of toroidal grids. *Australian J. Combin.*, 55:273–288, 2013.

[32] L. Saha and P. Panigrahi. A lower bound for radio k-chromatic number. *Discrete Appl. Math.*, 192:87–100, 2015.

[33] Diaa Salama, E. Badr, and Mostafa Farag. Fhe-chaos nhcp: Developing a novel secure framework for cloud computing environment. *Journal of Computing and Communication*, 1(1):12–26, 2022.

[34] M. Saraswathi and K. N. Meera. Radio mean labeled paths in cryptography. In *IEEE 4th PhD Colloquium on Emerging Domain Innovation and Technology for Society (PhD EDITS)*, 2022.

[35] H. Shabana, R. El-Shanawany, and S. R. Halawa. Graph design for data authentication over insecure communication channel. *Alexandria Engineering Journal*, 75:649–662, 2023.

[36] H. Shabana, R. El-Shanawany, and S. R. Halawa. Orthogonal Decompositions of Regular Graphs and Designing Tree-Hamming Codes. *European Journal of Pure and Applied Mathematics*, 17(4):3492–3516, 2024.

[37] H. Shabana and M. V. Volkov. Careful synchronization of partial deterministic finite automata. *Acta Informatica*, 59:479–504, 2022.

[38] G. S. Singh. A note on graceful prisms. *National Academy of Sciences Letters*, 15:193–194, 1992.

[39] P. Zhang. Radio labellings of cycles. *Ars Combin.*, 65:21–32, 2002.