

# Binomial Labeling of Intersection Mirror Lagoon Step Graph and Intersection Opposite Lagoon Step Graph with Python Implementation

<sup>1</sup>J. Vasanthi, <sup>2</sup>N. Ramya

<sup>1</sup>Research Scholar, Department of Mathematics, Bharath Institute of Higher Education & Research, Chennai, Tamil Nadu, India.

<sup>2</sup>Professor, Department of Mathematics, Bharath Institute of Higher Education & Research, Chennai, Tamil Nadu, India.

**Article History:**

**Received:** 27-10-2024

**Revised:** 01-12-2024

**Accepted:** 28-12-2024

**Abstract:**

Assume  $G = \{V, E\}$  be the graph with  $n$  vertices and  $e$  edges. The binomial labeling classification has been used for Intersection Mirror Lagoon Step Graph  $IM L_n S_m$  and Intersection Opposite Lagoon Step Graph  $IO L_n S_m$  in this paper. The concepts stated below in the paper have been implemented using Python coding.

**Keywords:** Intersection Mirror Lagoon Step Graph, Intersection Opposite Lagoon Step Graph, Binomial labeling.

## 1. Introduction

Rosa [1] first proposed the idea of graph labeling in 1967. Gallian [2] updates a dynamic overview of graph labeling techniques on a regular basis, and the Electronic Journal of Combinatory publishes it. We adhere to Bondy and Murthy's [3] notation and nomenclature for several aspects of graph theory. Binomial labeling introduced by Chandrakala and Sekar [4] et.al. The Binomial labeling for Intersection Mirror Lagoon Step Graph  $IM L_n S_m$  and the Intersection Opposite Lagoon Step Graph  $IO L_n S_m$  are constructed here using Python code [5][6].

## 2. Preliminaries

**Definition 1 :** A Lagoon Step graph  $L_n S_m$  is obtained from Lagoon  $L_n$  (L Shape) graph with vertices  $n$  ( $n \geq 3$ , must be an odd) joining with step graph  $S_m$  ( $m \geq 1$ ,  $m$  is number of steps with  $m = (n-1)/2$ ), the graph consists of  $n+2m-1$  vertices and  $n+2m-1$  edges and is described below

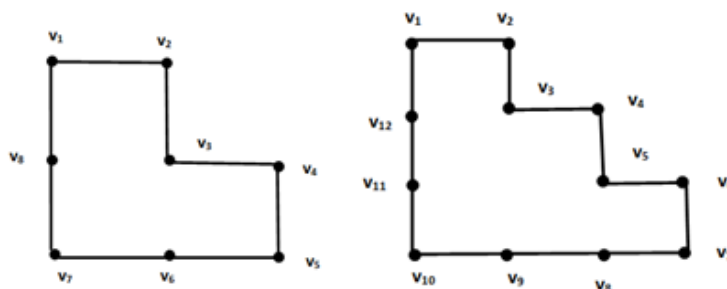


Figure 1:  $L_5S_2$  &  $L_7S_3$

**Definition 2 :** Intersection Mirror Lagoon Step graph  $IM L_n S_m$  obtained by attaching Lagoon step graph with Mirror image Lagoon step graph consists of  $3n+m-4$  vertices and  $3n+m-3$  edges (with  $n \geq 3$  must be an odd and  $m = (n-1)/2$ ,  $m \geq 1$ ) described below

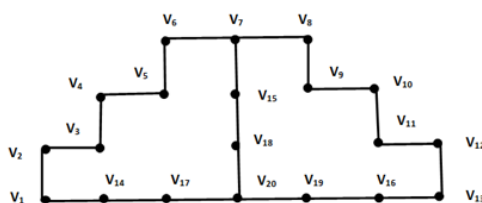


Figure 2: IM L<sub>7</sub>S<sub>3</sub>

**Definition 3:** Intersection Opposite Lagoon Step graph IO L<sub>n</sub>S<sub>m</sub> obtained by attaching Lagoon step graph with Opposite direction of Lagoon step graph consists of vertices 2n+4m-4 vertices and edges 2n+4m-3 (with n ≥ 3 must be an odd and m = (n-1)/2, m ≥ 1) described below

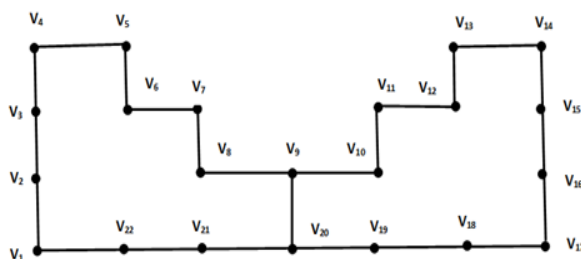


Figure 3: IO L<sub>7</sub>S<sub>3</sub>

**Definition 4 :** For positive integers n and r such that 0 ≤ r ≤ n, then Binomial coefficient  ${}^n C_r = n!/(r!(n-r)!)$  is always an integer.

**Definition 5:** Consider G = (p, q) be a graph. If the graph is an injective map f: V(G) → {1, 2, 3, ..., q+1} where the induced f\*: E(G) → N is provided by, then G has a Binomial Labeling by f\*(u v) =  ${}^M C_m = M!/(m!(M-m)!)$  where M = max {f(u), f(v)}, m=min {f(u), f(v)} assigns distinct labels for the edges.

**Definition 6:** The term "binomial graph" refers to a graph that has the Binomial Labeling.

### 3. Main Results

**3. 1 Theorem:** Intersection Mirror Lagoon step graph IM L<sub>n</sub> S<sub>m</sub> is a Binomial graph.

**Proof:** Consider the graph G be the intersection Mirror Lagoon step graph IM L<sub>n</sub> S<sub>m</sub> with 3n+m-4 vertices and 3n+m-3 edges when n = 3, 5, 7, 9... then m = 1, 2, 3... respectively.

**Case i :** When n = 5, 7, 9... then m = 2, 3,4... respectively.

Let v<sub>i</sub> be the outer vertices starts from left step corner to right step corner, w<sub>i</sub> be the vertical inner vertices and x<sub>i</sub> be the base vertices. Classify f: V (G) → {1, 2, 3... 3n+m-2} is given below

f(v<sub>i</sub>) = i, i = 1, 2 ... n+2m-2, when n = 5, 7, 9..., then m = 2, 3, 4... respectively,

f(w<sub>i</sub>) = 2n+2m-j-1, which can be expressed as

i	n	m	j
1	5	2	0
1, 2	7	3	0, 1
1, 2, 3	9	4	0, 1, 2

.	.	.	.
.	.	.	.
.	.	.	.

respectively.

$f(x_i) = n+2m+i-2$  which can be expressed as

n	m	i
5	2	1, 2, 3, 4, 5
7	3	1, 2, 3, 4, 5, 6, 7
9	4	1, 2, 3, 4, 5, 6, 7, 8, 9
.	.	.
.	.	.
.	.	.

respectively.

The equation  $f^*(e=uv) = {}^R C_r = R!/(r!(R-r)!)$  yields  $f^*: E(G) \rightarrow N$ , wherein  $R = \max\{f(u), f(v)\}$  and  $r = \min\{f(u), f(v)\}$  for every  $uv \in E(G)$ . Every edge in this case has a unique label.

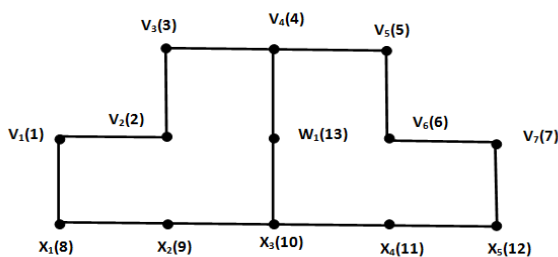


Figure 4: Binomial graph  $IML_5S_2$

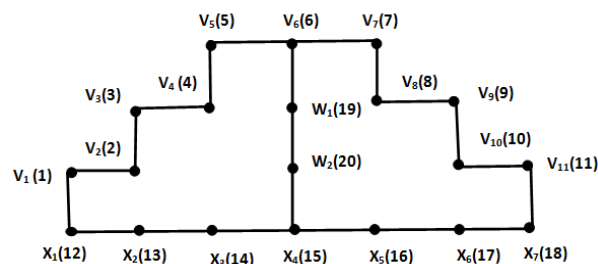


Figure 5: Binomial graph  $IM L_7S_3$

**Special case  $IM L_3S_1$ :** Consider  $v_1, v_2, v_3, v_4, v_5$  and  $v_6$  be the six vertices of  $IM L_3S_1$ . Vertex labeling are  $f(v_i) = i, i = 1, 2, 3, 4, 5$  and  $f(v_i) = i+1, i = 6$ .

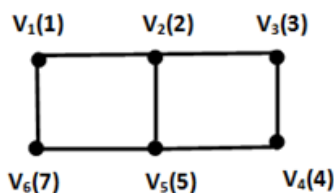


Figure 6: Binomial graph  $IML_3S_1$

Every edge in this case has a unique label.  $IML_nS_m$  is a binomial graph as a result.

**3. 2 Theorem :** Intersection Opposite Lagoon step graph  $IO L_nS_m$  is a Binomial graph.

**Proof :** Consider the graph  $G$  be the Intersection Opposite Lagoon Step graph with  $2n+4m-4$  vertices and  $2n+4m-3$  edges when  $n = 3, 5, 7, \dots$ , then  $m = 1, 2, 3, 4, \dots$  respectively.

**Case i:** When  $n = 5, 7, 9, \dots$  then  $m = 2, 3, 4, \dots$  respectively.

Let  $v_i$  be the outer vertices except last vertex  $v_{i1}$  where  $v_{i1}$  be the last vertex.

Classify  $f: V(G) \rightarrow \{1, 2, 3 \dots 2n+4m-2\}$  as follows

$f(v_i) = i, i = 1, 2, 3 \dots 2n+4m-5$  when  $n = 5, 7, 9 \dots$  then  $m = 2, 3, 4 \dots$  respectively,

$f(v_{i1}) = i + 1$ , where  $i = 2n+4m-4$  when  $n = 5, 7, 9 \dots$  then  $m = 2, 3, 4 \dots$  respectively.

$f^*: E(G) \rightarrow N$  is given by  $f^*(e=uv) = {}^R C_r = R!/(r!(R-r)!)$  for all  $uv \in E(G)$  where  $R = \max \{f(u), f(v)\}$  and  $r = \min \{f(u), f(v)\}$ . Here all the edges have distinct labels.

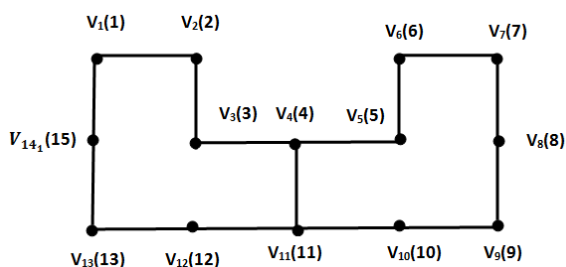


Figure 7: Binomial graph  $IOL_5S_2$

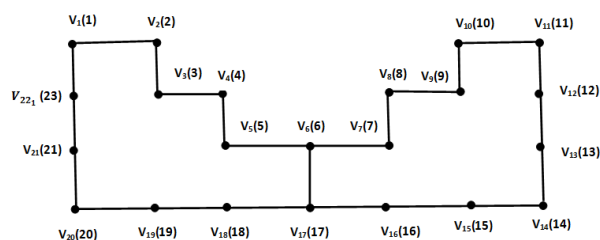


Figure 8: Binomial graph  $IOL_7S_3$

**Special case  $IO L_3S_1$ :** Let  $v_1, v_2, v_3, v_4, v_5$  and  $v_6$  be the six vertices of  $IO L_3S_1$ . Vertex labelings are  $f(v_i) = i, i = 1, 2, 3, 4, 5$  and  $f(v_i) = i+1, i = 6$ .

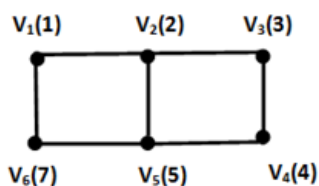


Figure 9: Binomial graph  $IOL_3S_1$

Every edge in this case has a unique label.  $IOL_nS_m$  is a binomial graph as a result.

#### 4. Verification Using Python Coding

##### 4.1 Verification of the Binomial of $IM L_9 S_4$ and $IM L_{11} S_5$ using Python Implementation

```

binomial labeling IMLnSm.py
binomial labeling IMLnSm.py > IMLnSm
1 def IMLnSm(n):
2     if n < 3 or n % 2 == 0:
3         print("Error: n must be an odd number and greater than or equal to 3.")
4         return
5     m = (n - 1) // 2
6     print(f"Calculated value of m: {m}")
7     v_start = 1
8     v_end = 3 * n + m - 4
9     V = list(range(v_start, v_end + 1))
10    print(f"Possible values of V (vertices): {V}")
11    e_start = 1
12    e_end = 3 * n + m - 3
13    E = list(range(e_start, e_end + 1))
14    print(f"Possible values of E (edges): {E}")
15    print("\nOuter vertices starting with left step corner to right step corner:")
16    o_start = 1
17    o_end = n + 2 * m - 2
18    outer_vertices = list(range(o_start, o_end + 1))
19    print(f"f(vi) = {outer_vertices}")
20    print("\nVertical inner vertices:")
21    for i in range(1, m):
22        j = i - 1
23        inner_vertex = 2 * n + 2 * m + j - 1
24        print(f"f(wi) = {inner_vertex}")
25    print("\nBase vertices:")
26    for i in range(1, n + 1):
27        base_vertex = n + 2 * m + i - 2
28        print(f"f(x(i)) = {base_vertex}")
    
```

```

binomial labeling IMLnSm.py x
binomial labeling IMLnSm.py > IMLnSm
1 def IMLnSm(n):
2     inner_vertex = 2 * n + 2 * m + 3 - 1
24     print(f"u(i) = {inner_vertex}")
25     print("\nBase vertices:")
26     for i in range(1, n + 1):
27         base_vertex = n + 2 * m + i - 2
28         print(f"x(i) = {base_vertex}")
29     n = int(input("Enter an odd number greater than or equal to 3: "))
30     IMLnSm(n)
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Lenovo\Desktop\vas\WPH PhD> python -i "C:\Users\Lenovo\Desktop\vas\WPH PhD\binomial labeling IMLnSm.py"
Enter an odd number greater than or equal to 3: 9
Calculated value of m: 4
Possible values of V (vertices): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]
Possible values of E (edges): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28]
Outer vertices starting with left step corner to right step corner:
f(vi) = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
Vertical inner vertices:
f(u1) = 25
f(u2) = 26
f(u3) = 27
Base vertices:
f(x1) = 16
f(x2) = 17
f(x3) = 18
f(x4) = 19
f(x5) = 20
f(x6) = 21
f(x7) = 22
f(x8) = 23
f(x9) = 24
Ln 1, Col 15 Spaces: 4 UTF-8 CR LF Python 3.12.1 Go Live

```

```

binomial labeling IMLnSm.py x
binomial labeling IMLnSm.py > IMLnSm
1 def IMLnSm(n):
2     inner_vertex = 2 * n + 2 * m + 3 - 1
24     print(f"u(i) = {inner_vertex}")
25     print("\nBase vertices:")
26     for i in range(1, n + 1):
27         base_vertex = n + 2 * m + i - 2
28         print(f"x(i) = {base_vertex}")
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Lenovo\Desktop\vas\WPH PhD> python -i "C:\Users\Lenovo\Desktop\vas\WPH PhD\binomial labeling IMLnSm.py"
Enter an odd number greater than or equal to 3: 11
Calculated value of m: 5
Possible values of V (vertices): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]
Possible values of E (edges): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]
Outer vertices starting with left step corner to right step corner:
f(vi) = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
Vertical inner vertices:
f(u1) = 31
f(u2) = 32
f(u3) = 33
f(u4) = 34
Base vertices:
f(x1) = 20
f(x2) = 21
f(x3) = 22
f(x4) = 23
f(x5) = 24
f(x6) = 25
f(x7) = 26
f(x8) = 27
f(x9) = 28
f(x10) = 29
f(x11) = 30
Ln 1, Col 15 Spaces: 4 UTF-8 CR LF Python 3.12.1 Go Live

```

### 4. 2 Verification of the Binomial of IO L<sub>9</sub> S<sub>4</sub> and IO L<sub>11</sub> S<sub>5</sub> using Python Implementation

```

binomial labeling IOLnSm.py x binomial labeling IMLnSm.py
1 def IOLnSm(n):
2     if n < 3 or n % 2 == 0:
3         print("Error: n must be an odd number and greater than or equal to 3.")
4         return
5     m = (n - 1) // 2
6     print(f"Calculated value of m: {m}")
7     v_start = 1
8     v_end = 2 * n + 4 * m - 4
9     V = list(range(v_start, v_end + 1))
10    print(f"Possible values of V (vertices): {V}")
11    e_start = 1
12    e_end = 2 * n + 4 * m - 3
13    E = list(range(e_start, e_end + 1))
14    print(f"Possible values of E (edges): {E}")
15    print("\nOuter vertices except last vertex:")
16    o_start = 1
17    o_end = 2 * n + 4 * m - 5
18    outer_vertices = list(range(o_start, o_end + 1))
19    print(f"f(vi) = {outer_vertices}")
20    print("\nLast vertex:")
21    i = 2 * n + 4 * m - 4
22    last_vertex = i + 1
23    print(f"f(v(i)) = {last_vertex}")
24    n = int(input("Enter an odd number greater than or equal to 3: "))
25    IOLnSm(n)
Ln 1, Col 15 Spaces: 4 UTF-8 CR LF Python 3.12.1 Go Live

```

```

1 def IOLnSm(n):
2     if n < 3 or n % 2 == 0:
3         print("Error: n must be an odd number and greater than or equal to 3.")
4         return
5     m = (n - 1) // 2
6     print(f"Calculated value of m: {m}")
7     v_start = 1
8     v_end = 2 * n + 4 * m - 4
9     V = list(range(v_start, v_end + 1))
10    print(f"Possible values of V (vertices): {V}")
11    e_start = 1
12    e_end = 2 * n + 4 * m - 3
13    E = list(range(e_start, e_end + 1))
14    print(f"Possible values of E (edges): {E}")
15    print("\nOuter vertices except last vertex:")
16    o_start = 1
17    o_end = 2 * n + 4 * m - 5
18    outer_vertices = list(range(o_start, o_end + 1))

```

Terminal Output:

```

PS C:\Users\Lenovo\Desktop\vas\MDM PHD> python -u "C:\Users\Lenovo\Desktop\vas\MDM PHD\binomial labeling IOLnSm.py"
Enter an odd number greater than or equal to 3: 9
Calculated value of m: 4
Possible values of V (vertices): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
Possible values of E (edges): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31]
Outer vertices except last vertex:
f(vi) = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
Last Vertex:
f(v30) = 31

```

```

1 def IOLnSm(n):
2     if n < 3 or n % 2 == 0:
3         print("Error: n must be an odd number and greater than or equal to 3.")
4         return
5     m = (n - 1) // 2
6     print(f"Calculated value of m: {m}")
7     v_start = 1
8     v_end = 2 * n + 4 * m - 4
9     V = list(range(v_start, v_end + 1))
10    print(f"Possible values of V (vertices): {V}")
11    e_start = 1
12    e_end = 2 * n + 4 * m - 3
13    E = list(range(e_start, e_end + 1))
14    print(f"Possible values of E (edges): {E}")
15    print("\nOuter vertices except last vertex:")
16    o_start = 1
17    o_end = 2 * n + 4 * m - 5
18    outer_vertices = list(range(o_start, o_end + 1))

```

Terminal Output:

```

PS C:\Users\Lenovo\Desktop\vas\MDM PHD> python -u "C:\Users\Lenovo\Desktop\vas\MDM PHD\binomial labeling IOLnSm.py"
Enter an odd number greater than or equal to 3: 11
Calculated value of m: 5
Possible values of V (vertices): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38]
Possible values of E (edges): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39]
Outer vertices except last vertex:
f(vi) = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37]
Last Vertex:
f(v39) = 39

```

## 5. Conclusion

The demonstration of graphs like Intersection Mirror Lagoon step graph and Intersection Opposite Lagoon step graph exist when it comes to Binomial labeling. This paper also discusses the limitations and possible uses of this type of labeling. Python coding for Binomial labeling has been implemented for the above graphs. In the near future, more research under the Lagoon Step graph using various labeling strategies is needed.

## References

- [1] A. Rosa, On Certain Valuations of the Vertices of a Graph, In Theory of Graphs (Internat. Sympos. Rome. (1966) (1967), 349–359, Gordan and Breach. Newyork. Dunod. Paris.
- [2] J.A. Gallian, A Dynamic Survey of Graph Labeling, Electronic Journal of Combinatorics, 17 (DS6) (2016).
- [3] Bondy .J.A and Murthy U.S.R, “Graph Theory and Application” (North Holland). New York (1976). [4].S. Chandrakala and C.Sekar, “Binomial Labeling of Some Graphs”,International Journal for Research in Engineering Application & Management ISSN: 2454-9150 ,Vol.04,Issue. 08, Nov 2018.
- [5] D. Amuthavalli, O. V. Shanmuga Sundaram, Super Fibonacci graceful anti – magic labeling for flower graphs and python coding,Tuijin Jishu/Journal of Propulsion Technology, Vol. 44 No. 3,2023.
- [6] J.A.Gadhiya and R.Solanki, “ Labeling of Some Graphs Using Python Programming,” vol.20,no.17,pp.2288 - 2294,2022.