

# Structural Conditions and Algorithms for Cycle–Star Decomposition of Complete Bipartite Graphs

Ilayaraja M.<sup>1</sup>; Abiramasundari C.<sup>2</sup>; Bhuvaneshwari R.<sup>3</sup>; Imran R.<sup>4</sup>

<sup>1,2,3,4</sup>Department of Mathematics Sona College of Arts and Science, Salem-5, India

Publication Date: 2026/03/12

**Abstract:** This paper establishes necessary and sufficient conditions for the existence of decompositions of complete bipartite graphs into cycles and stars in certain classes of cases. Explicit constructions are provided for all admissible parameter sets. In addition, a Python implementation is developed to verify the theoretical conditions and to generate corresponding decompositions.

**How to Cite:** Ilayaraja M.; Abiramasundari C.; Bhuvaneshwari R.; Imran R. (2026) Structural Conditions and Algorithms for Cycle–Star Decomposition of Complete Bipartite Graphs. *International Journal of Innovative Science and Research Technology*, 11(3), 250-253. <https://doi.org/10.38124/ijisrt/26mar118>

## I. INTRODUCTION

Graph decomposition is a central topic in graph theory, with applications in design theory, network analysis, and algorithmic graph theory. Complete bipartite graphs play an important role in decomposition problems. The complete bipartite graph  $K_{m,n}$  consists of two disjoint vertex sets of sizes  $m$  and  $n$ , where each vertex in one part is adjacent to every vertex in the other. Owing to their regular structure, such graphs serve as natural objects for studying both theoretical and constructive aspects of graph decompositions.

In this paper, we investigate decompositions of complete bipartite graphs into cycles and stars. A cycle is a connected 2-regular graph, and a star is a tree isomorphic to  $K_{1,t}$  for some integer  $t \geq 1$ . The main objective is to determine necessary and sufficient conditions for the existence of such decompositions and to provide explicit construction methods.

## II. PRELIMINARIES

In this section, we introduce basic definitions and notation used throughout the paper.

- Definition 2.1. A cycle is a connected 2-regular graph. A cycle of length  $k$  is denoted by  $C_k$ .
- Definition 2.2. A star is a tree isomorphic to the complete bipartite graph  $K_{1,t}$  for some integer  $t \geq 1$ . The unique vertex of degree  $t$  is called the center of the star.

- Definition 2.3. Let  $G = (V, E)$  be a graph. A decomposition of  $G$  is a collection of subgraphs  $\{G_1, G_2, \dots, G_k\}$  such that

$$E(G_i) \cap E(G_j) = \emptyset \text{ for all } i \neq j, \text{ and}$$

$$E(G) = \bigcup_{i=1}^k E(G_i)$$

- Definition 2.4. A cycle-star decomposition of a graph  $G$  is a decomposition in which each subgraph is either a cycle or a star.
- Definition 2.5. The complete bipartite graph  $K_{m,n}$  is a graph whose vertex set can be partitioned into two disjoint sets of sizes  $m$  and  $n$ , such that every vertex in one set is adjacent to every vertex in the other.

## III. MAIN RESULTS

In this section, we present necessary and sufficient conditions for the existence of cycle–star decompositions of complete bipartite graphs.

## IV. NECESSARY CONDITIONS FOR $\{PC_4, QS_4\}$ -DECOMPOSITION

- $K_{2,n}$  with  $n$  even:  $q$  must be even.
- $K_{4,n}$  with  $n \geq 4$ :  $p \neq 1$  and  $q \neq 1$ .
- $K_{m,n}$  with  $m, n \geq 6$  even:  $q \neq 1$ .
- Exactly one partition odd:
  - ✓  $m$  odd,  $n \equiv 0 \pmod{4}$ :  $q \geq n/4$ .
  - ✓  $n$  odd,  $m \equiv 0 \pmod{4}$ :  $q \geq m/4$ .

**V. MAIN THEOREM**

➤ Let  $m, n \in \mathbb{Z}^+$  with  $m \leq n$  and  $p, q \geq 0$ .  $K_{m,n}$  admits a  $\{pC_4, qS_4\}$ -decomposition if and only if one of the following holds:

- $m = 2$  even,  $n$  even:  $q$  even.
- $m = 4, n \geq 4$  even:  $p, q \neq 1$ .
- $m, n \geq 6$  even:  $q \neq 1$ .
- Exactly one partition odd:
- ✓  $m$  odd,  $n \equiv 0 \pmod{4}$ :  $q \geq n/4$ .
- ✓  $n$  odd,  $m \equiv 0 \pmod{4}$ :  $q \geq m/4$ .

**VI. ALGORITHM FOR  $\{PC_4, QS_4\}$ -DECOMPOSITION OF  $K_{M,N}$**

We describe an algorithm that determines whether a complete bipartite graph  $K_{m,n}$  admits a decomposition into  $p$  cycles  $C_4$  and  $q$  stars  $S_4$ , and constructs such a decomposition whenever possible.

➤ *Algorithm 1:  $\{pC_4, qS_4\}$ -Decomposition of  $K_{m,n}$*

- Require: Integers  $m, n$  (part sizes), integer  $q$  (number of stars)
- Ensure: Lists of cycles and stars forming a decomposition, or report impossible

```

Compute total number of edges: total_edges = m * n
Check divisibility: if (total_edges - 4q) mod 4 != 0, return "Decomposition impossible"
Set p = (total_edges - 4q) / 4
Check structural conditions (small cases, parity, forbidden values):
if conditions fail, return "Decomposition impossible"
Initialize vertex sets:
X = {x1, ..., xm}, Y = {y1, ..., yn}
Initialize empty sets: C4 list, S4 list, used edges
Construct p copies of C4:
for each of the p cycles do
Select two vertices xi, xi+1 in X and two vertices yj, yj+1 in Y
Form 4-cycle: (xi, yj), (xi+1, yj), (xi+1, yj+1), (xi, yj+1)
Add cycle to C4_list and mark edges as used
Update indices i, j with wrap-around as needed
end for
Construct q copies of S4:
Let remaining edges = edges not in C4 list
while remaining edges is not empty do
Find a vertex in X union Y with degree >= 4 in remaining edges
Form a star with 4 incident edges
Add star to S4_list and remove edges from remaining edges
if no such vertex exists then
Handle fallback (group remaining 4 edges if possible)
end if
end while
return C4_list, S4_list
    
```

➤ *Remarks on Correctness*

- The algorithm first checks the necessary edge-count condition  $mn = 4p + 4q$ . If this fails, decomposition is impossible.
- Structural feasibility conditions include parity checks, forbidden small cases (e.g.,  $p = 1$  or  $q = 1$  in small graphs), and minimum size requirements for  $m$  and  $n$ .
- Each  $C_4$  is constructed using disjoint pairs of vertices, ensuring edge-disjoint cycles.
- Remaining edges are grouped into  $S_4$  stars by selecting vertices with degree at least 4. This ensures all edges are covered.
- The algorithm terminates in  $O(mn)$  time.

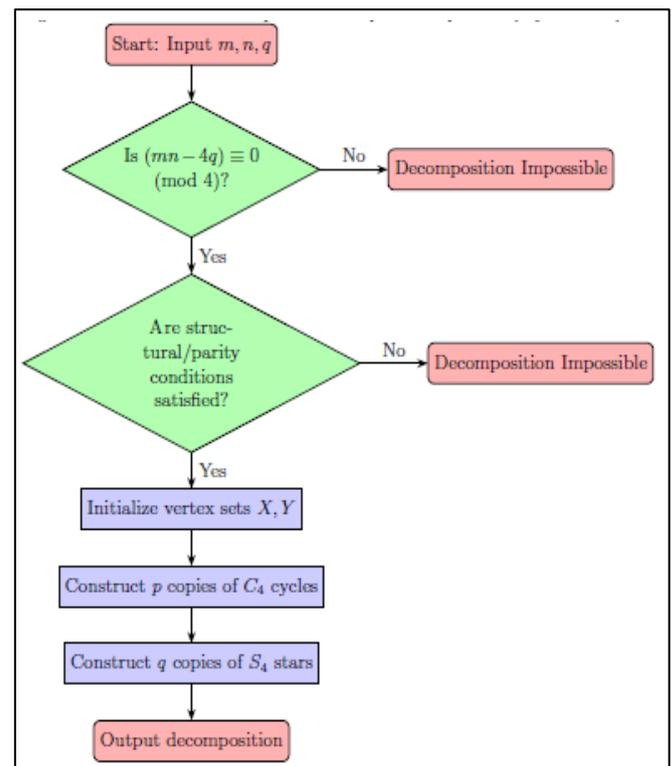


Fig 1 Flowchart for the Decomposition of a Bipartite Graph into  $C_4$  Cycles and  $S_4$  Stars

**VII. PYTHON PROGRAM FOR DECOMPOSITION**

In this section, we present a Python program that verifies the existence of a  $C_4$ - $S_4$  decomposition of the complete bipartite graph  $K_{m,n}$  and constructs such a decomposition whenever possible. The program first checks the necessary conditions and then explicitly generates the cycles and stars.

```

Python program
def is_decomposition_possible(m, n, q):
total_edges = m * n

# Check edge count condition
if (total_edges - 4 * q) % 4 != 0:
    return False
    
```

```

return False, None

p = (total_edges - 4 * q) // 4

# Special cases
if m == 2 and n % 2 == 0 and q % 2 == 0:
    p = n // 2
    return True, p

if m == 4 and n >= 4 and q != 1 and p != 1:
    return True, p

if m >= 6 and n >= 6 and q != 1:
    return True, p

if (m % 2 == 1 and n % 4 == 0 and q >= n // 4) or \
    (n % 2 == 1 and m % 4 == 0 and q >= m // 4):
    return True, p

return False, None

def decompose_Kmn_C4_S4(m, n, q):
    possible, p = is_decomposition_possible(m, n, q)
    if not possible:
        raise ValueError("Decomposition not possible for given
        parameters")

    # Vertex sets
    X = [f'x{i+1}' for i in range(m)]
    Y = [f'y{i+1}' for i in range(n)]

    # All edges of  $K_{m,n}$ 
    total_edges = {(x, y) for x in X for y in Y}
    used_edges = set()

    C4_list = []
    S4_list = []

    # Construct p copies of C4
    i = j = 0
    for _ in range(p):
        x1, x2 = X[i], X[(i + 1) % m] # Fixed wrap-around
        y1, y2 = Y[j], Y[(j + 1) % n]
        cycle = [(x1, y1), (x2, y1), (x2, y2), (x1, y2)]
        C4_list.append(cycle)
        used_edges.update(cycle)

        j += 2
        if j >= n:
            j = 0
            i += 2
        if i >= m:
            i = 0

    # Remaining edges for S4 stars
    remaining_edges = list(total_edges - used_edges)

    from collections import defaultdict
    adj_X = defaultdict(list)
    adj_Y = defaultdict(list)

```

```

for x, y in remaining_edges:
    adj_X[x].append(y)
    adj_Y[y].append(x)

# Construct S4 stars (simplified and fixed indexing issues)
while remaining_edges:
    made_star = False

    for x in X:
        if len(adj_X[x]) >= 4:
            leaves = adj_X[x][:4]
            star = [(x, leaf) for leaf in leaves]
            S4_list.append(star)
            for leaf in leaves:
                adj_X[x].remove(leaf)
                adj_Y[leaf].remove(x)
            remaining_edges.remove((x, leaf))
            made_star = True
            break

    if made_star:
        continue

    for y in Y:
        if len(adj_Y[y]) >= 4:
            leaves = adj_Y[y][:4]
            star = [(leaf, y) for leaf in leaves]
            S4_list.append(star)
            for leaf in leaves:
                adj_Y[y].remove(leaf)
                adj_X[leaf].remove(y)
            remaining_edges.remove((leaf, y))
            made_star = True
            break

    if not made_star and remaining_edges:
        # Fallback: take any 4 remaining edges as a "star" (for
        completeness)
        fallback = remaining_edges[:4]
        S4_list.append(fallback)
        for edge in fallback:
            remaining_edges.remove(edge)

return C4_list, S4_list

```

The function `is_decomposition_possible` implements the theoretical conditions established in Section 3. The function `decompose_Kmn_C4_S4` then constructs the corresponding decomposition by first generating  $C_4$  cycles and subsequently partitioning the remaining edges into  $S_4$  stars.

## VIII. EXAMPLES

➤ *Example 1:  $K_{2,6}$  with  $q = 0$*

```

m, n, q = 2, 6, 0
C4s, S4s = decompose_Kmn_C4_S4(m, n, q)
print("C4_ cycles :", C4s)

```

```
print ("S4_ stars :", S4s)
```

**Output:** The graph  $K_{2,6}$  admits a decomposition into three  $C_4$  cycles and no  $S_4$  stars.

➤ *Example 2:  $K_{4,4}$  with  $q = 1$*

```
m, n, q = 4, 4, 1
```

```
try:
```

```
C4s , S4s = decompose_Kmn_C4_S4 (m, n, q)
```

```
except ValueError as e:
```

```
print (e)
```

**Output:** No decomposition exists, as the parameters violate the necessary condition  $p, q \neq 1$

➤ *Example 3:  $K_{6,6}$  with  $q = 2$*

```
m, n, q = 6, 6, 2
```

```
C4s, S4s = decompose_Kmn_C4_S4(m, n, q)
```

```
print ("C4_ cycles :", C4s)
```

```
print ("S4_ stars :", S4s)
```

**Output:** The graph  $K_{6,6}$  admits a decomposition into eight  $C_4$  cycles and two  $S_4$  stars.

➤ *Example 4:  $K_{3,8}$  with  $q = 2$*

```
m, n, q = 3, 8, 2
```

```
C4s, S4s = decompose_Kmn_C4_S4(m, n, q)
```

```
C4s, S4s = decompose_Kmn_C4_S4(m, n, q)
```

```
print ("C4_ cycles :", C4s)
```

```
print ("S4_ stars :", S4s)
```

**Output:** Since one partite set has odd cardinality, the algorithm verifies the parity condition and produces a valid decomposition consisting of  $C_4$  cycles and two  $S_4$  stars

## IX. CONCLUSION

In this paper, we established necessary and sufficient conditions for the existence of  $\{pC_4, qS_4\}$ -decompositions of complete bipartite graphs in several special cases. These conditions were translated into an explicit algorithm implemented in Python.

The program verifies all edge-count and parity constraints and constructs the required decomposition whenever it exists. The examples confirm the correctness of the theoretical results and demonstrate the effectiveness of the algorithm. Future work may explore extensions to larger cycle lengths, other classes of trees, or more general bipartite graphs.

## REFERENCES

- [1]. Alspach, B., Gavlas, H. (2001). Cycle decompositions of  $K_n$  and  $K_n - I$ . Journal of Combinatorial Theory, Series B, 81(1), 77–99.
- [2]. Bondy, J. A., Murty, U. R. S. (1976). Graph Theory with Applications. Macmillan Press, New York.
- [3]. Jeevadoss, S., Muthusamy, A. (2014). Decomposition of complete bipartite graphs into

paths and cycles. Discrete Mathematics, 331, 98–108.

- [4]. M. Ilayaraja and A. Muthusamy, Decomposition of Complete Graphs, Indian Journal of Discrete Mathematics, 4 (2018), 13–33.
- [5]. M. Ilayaraja and A. Muthusamy, Decomposition of Complete Bipartite Graphs into Cycles and Stars with Four Edges, AKCE International Journal of Graphs and Combinatorics, 17. (2020), 697–702.