Mathematical Foundations of Computer Science

This Lecture: Graph Theory: Eulerian Graphs

Department of Mathematical and Computational Sciences
National Institute of Technology Karnataka, Surathkal

MA714 (Odd Semester [2025-26])

Eulerian graphs form an important area of graph theory, originating from the famous Königsberg bridge problem.

- Focus: Traversing edges of a graph exactly once.
- Applications: Network routing, DNA sequencing, optimization.

Trail: A walk in a graph in which no edge is repeated.

Eulerian Trail: A trail that visits every edge exactly once.

Eulerian Circuit: An Eulerian trail that starts and ends at the same vertex.

Eulerian Graph: A connected graph that contains an Eulerian circuit.

Important Results on Eulerian Graphs

1. **Euler's Theorem**: A connected graph is Eulerian if and only if every vertex has even degree.
2. **Eulerian Trail Criterion**: A connected graph has an Eulerian trail if and only if exactly two vertices have odd degree.
3. **Non-Eulerian Characterization**: More than two odd degree vertices implies no Eulerian trail or circuit.
4. **Fleury's Algorithm**: Method to construct an Eulerian trail by avoiding bridges unless necessary.

Fleury's Algorithm is a classical method to construct an Eulerian trail or circuit in a graph.

- Works for Eulerian and semi-Eulerian graphs.
- Based on careful edge selection to avoid breaking connectivity.
- Ensures every edge is traversed exactly once.

What is Fleury's Algorithm?

An algorithm to find an Eulerian trail or circuit in a graph.

Works when the graph is connected and has 0 or 2 odd-degree vertices.

Rule: Avoid using a bridge edge unless there is no alternative.

Ensures we don't get stuck before using all edges.

1. Start at a vertex of odd degree (if exists), else any vertex.
2. While unused edges remain:
   a) If possible, choose a non-bridge edge.
   b) If no choice, take the bridge.
   c) Traverse edge, delete it from graph.
3. The sequence of vertices is the Eulerian trail/circuit.

Fleury’s Algorithm (Informal)

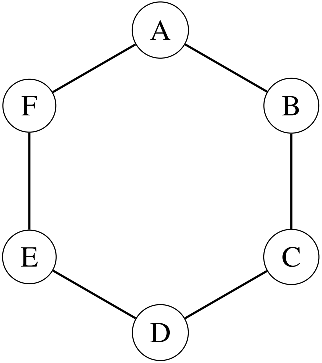
1. Choose a start vertex: any vertex if Eulerian; a vertex of odd degree if semi-Eulerian.
2. At each step, choose an edge incident to the current vertex that is *not* a bridge of the remaining graph, unless no such edge exists.
3. Traverse the chosen edge, remove it from the graph, and continue from the new vertex.
4. Repeat until all edges have been traversed.

Steps of Fleury's Algorithm (explained slightly differently)

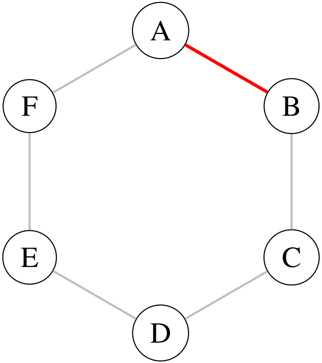
1. Start at a vertex with odd degree if an Eulerian trail exists, or any vertex if Eulerian circuit exists.
2. At each step, choose an edge that is not a bridge unless no alternative is available.
3. Delete the edge after traversing.
4. Repeat until all edges are used.

Example 1: Eulerian Circuit in a 6-cycle

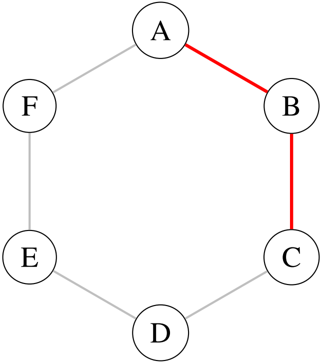
Graph: $A - B - C - D - E - F - A$



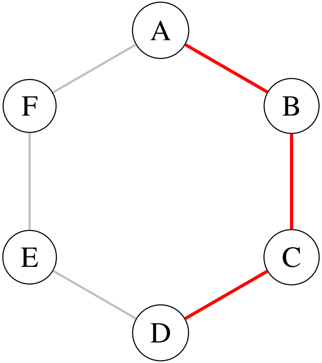
At A, choose A-B.



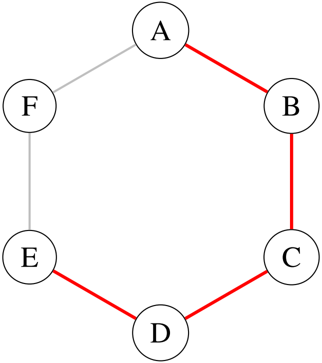
At B, must go B-C.



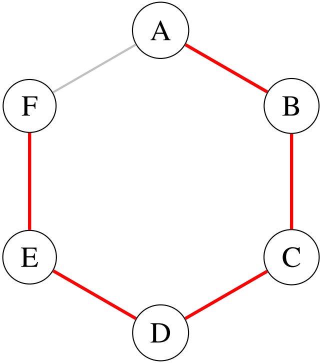
At C, go C-D.



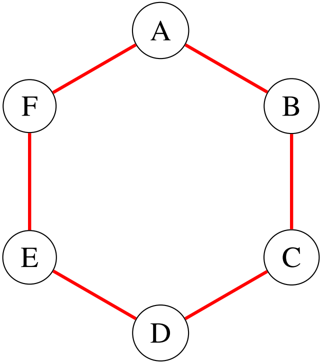
At D, go D-E.



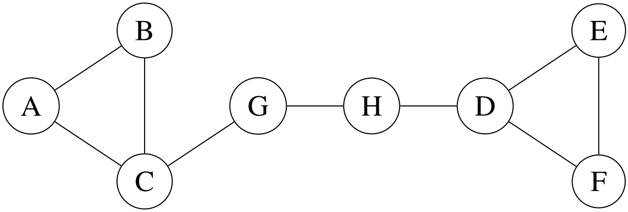
At E, go E-F.



At F, return F-A.



Graph: Two triangles linked by a path.



Fleury(G)

Input: connected graph $G = (V, E)$ that is Eulerian or semi-Eulerian.
Let v be a start vertex (odd degree if exists, else arbitrary).
While $E \neq \emptyset$:
    Choose an edge $e = (v, u)$ incident to v such that e is not a bridge of G unless it is the only incident edge.
    Output e; remove e from G; set $v \leftarrow u$.
EndWhile

Fleury's algorithm avoids bridges until forced.

Ensures no edges are left stranded.

Correct but not the fastest (O(E^2)).

Correctness sketch:

The algorithm never gets stuck prematurely because avoiding bridges preserves connectivity of the remaining graph components that still contain edges.

When only bridge edges remain from the current vertex, they must be part of the unique continuation of the trail.

In an Eulerian graph all vertices have even degree, so the trail returns to the start vertex (circuit).

Complexity:

Naïve implementation: at each step check whether an edge is a bridge by running a DFS — $O(E(E+V))$ worst-case.

With dynamic bridge-finding (e.g., maintaining DFS tree + lowlink) complexity can be improved, but Fleury is mainly pedagogical; Hierholzer's algorithm is preferred for efficiency ($O(E)$).

Problem 1: Is K_4 Eulerian?
Solution: Each vertex has degree 3 (odd). All 4 vertices are odd. Thus, not Eulerian.

Problem 2: Show that cycle graph C_n is Eulerian.
Solution: Each vertex has degree 2 (even). Hence C_n is Eulerian.

Problem 3: Graph with exactly two odd vertices has an Eulerian trail.
Solution: By Euler's theorem, it starts at one odd vertex and ends at the other.

Problem 4: Is the Petersen graph Eulerian?
Solution: All 10 vertices have degree 3 (odd). Not Eulerian.

Problem 5: Square with diagonals drawn.
Solution: Each vertex has degree 3 (odd). Not Eulerian. By adding an edge, all degrees can be made even.

Eulerian graphs are useful in:

1. **Network Routing**: Traverse every link once.
2. **DNA Sequencing**: Model genome assembly using Eulerian paths.
3. **Chinese Postman Problem**: Optimal street/trail coverage.
4. **Circuit Testing**: Test every connection once.
5. **Data Management**: Efficient traversal of structures.

Bondy, J.A. and Murty, U.S.R., *Graph Theory with Applications*, North-Holland, 1976.

Diestel, R., *Graph Theory*, Springer, 5th Edition, 2017.

West, D.B., *Introduction to Graph Theory*, Prentice Hall, 2nd Edition, 2001.

Deo, N., *Graph Theory with Applications to Engineering and Computer Science*, Prentice-Hall, 1974.

Eulerian graphs are characterized by vertex degrees.

Applications include optimization, routing, sequencing.

Simple but powerful concept in graph theory.