

Lesson 2. Accessibility and Connectivity.

1. Accessibility.
2. Computation of the Connected Components.
3. Euler trails and Euler tours.
4. Hamilton paths and Hamilton cycles.

1. Accessibility

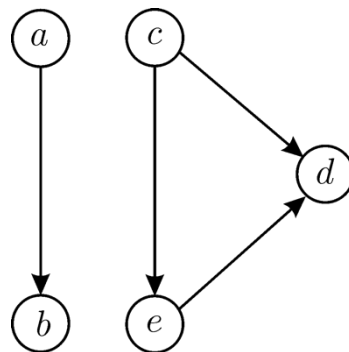
DEFINITIONS

Let $G = (V, A)$ be a directed graph.

1. Let $x_i, x_j \in V$, we say that x_j is **reachable** from x_i or x_i **reaches** to x_j if there exists a path from x_i to x_j .
2. The **reachable matrix** of G is the $n \times n$ square matrix defined by

$$R = [r_{ij}] / r_{ij} = \begin{cases} 1 & \text{if } v_i \text{ reaches } v_j \\ 0 & \text{otherwise} \end{cases}$$

EXAMPLE:



$$R = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

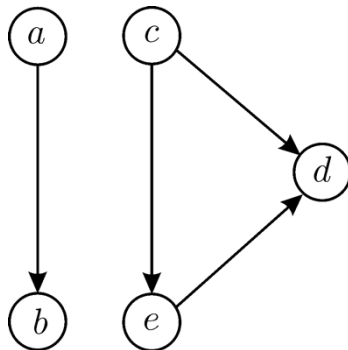
1. Accessibility

3. The **access matrix** of G is the $n \times n$ square matrix defined by

$$Q = [q_{ij}] / q_{ij} = \begin{cases} 1 & \text{if } v_j \text{ reaches } v_i \\ 0 & \text{otherwise} \end{cases}$$

PROPOSITION: $Q = R^T$.

EXAMPLE:



$$R = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

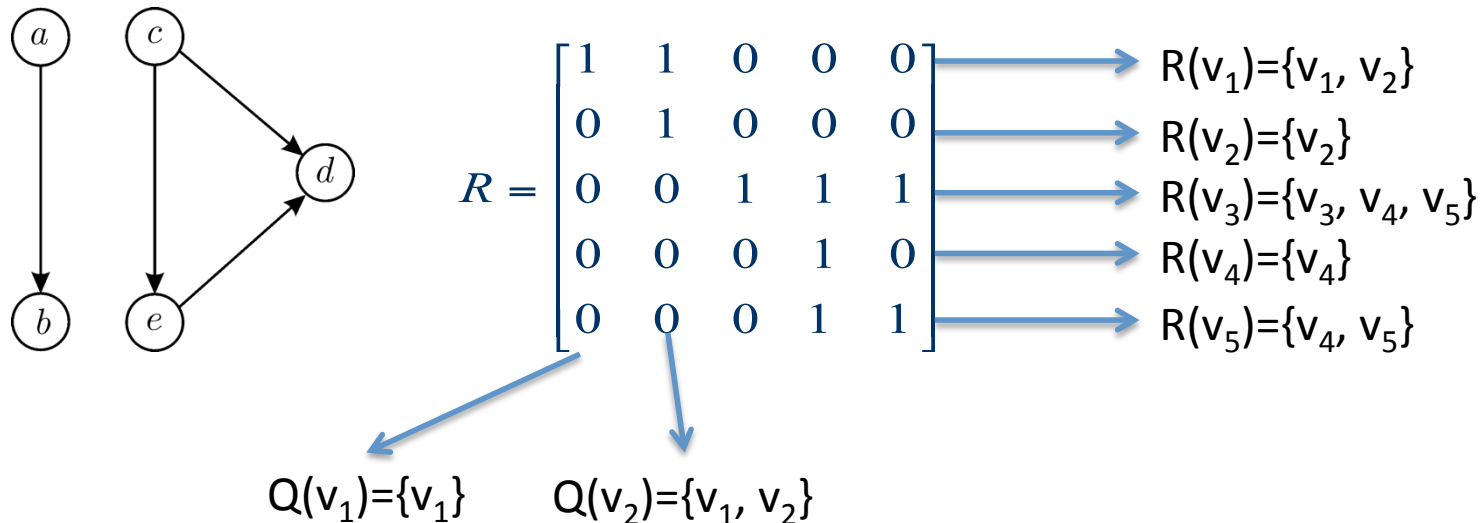
$$Q = R^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

1. Accessibility

Notation:

- $R(v_i)$ denotes the set of vertices that v_i reaches.
- $Q(v_i)$ denotes the set of vertices that reach to v_i .

EXAMPLE:



1. Accessibility

Notation:

$\Gamma^p(v_i)$ denotes the set of vertices reachable from v_i through a path of length p .

Remarks: It is obvious that the set $R(v_i)$ associated with a vertex v_i can be calculated using the sets $\Gamma^p(v_i)$, $p \leq n$:

$$R(v_i) = \{v_i\} \cup \Gamma(v_i) \cup \dots \cup \Gamma^p(v_i), \quad p \leq n$$

We can use the sets $R(v_i)$ to build the Reachable Matrix R . We will study another method, the Warshall's method, in the practical class.

EXAMPLE:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\Gamma^0(v_1) = \{v_1\}$$

$$\Gamma^1(v_1) = \{v_2\}$$

$$\Gamma^2(v_1) = \{v_3, v_6\}$$

$$\Gamma^3(v_1) = \{v_5, v_6\}$$

$$\Gamma^4(v_1) = \{v_3, v_5\}$$

$$R(v_1) = \{v_1, v_2, v_3, v_5, v_6\}$$

$$\Gamma^0(v_2) = \{v_2\}$$

$$\Gamma^1(v_2) = \{v_3, v_6\}$$

$$\Gamma^2(v_2) = \{v_5, v_6\}$$

$$\Gamma^3(v_2) = \{v_3, v_5\}$$

$$R(v_2) = \{v_2, v_3, v_5, v_6\}$$

$$\Gamma^0(v_3) = \{v_3\}$$

$$\Gamma^1(v_3) = \{v_6\}$$

$$\Gamma^2(v_3) = \{v_5\}$$

$$\Gamma^3(v_3) = \{v_3\}$$

$$R(v_3) = \{v_3, v_5, v_6\}$$

$$\Gamma^0(v_4) = \{v_4\}$$

$$\Gamma^1(v_4) = \{v_1, v_6\}$$

$$\Gamma^2(v_4) = \{v_2, v_5\}$$

$$\Gamma^3(v_4) = \{v_3, v_6\}$$

$$R(v_4) = \{v_1, v_2, v_3, v_4, v_5, v_6\}$$

$$\Gamma(v_5) = \{v_5\}$$

$$\Gamma^1(v_5) = \{v_3\}$$

$$\Gamma^2(v_5) = \{v_6\}$$

$$\Gamma^3(v_5) = \{v_5\}$$

$$R(v_5) = \{v_3, v_5, v_6\}$$

$$\Gamma(v_6) = \{v_6\}$$

$$\Gamma^1(v_6) = \{v_5\}$$

$$\Gamma^2(v_6) = \{v_3\}$$

$$\Gamma^3(v_6) = \{v_3, v_6\}$$

$$R(v_6) = \{v_3, v_5, v_6\}$$

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

2. Computation of the Connected Components

Let $G = (V, A)$ be a directed graph.

METHOD 1.

Step 1. Initialize $i \leftarrow 1$, $V^{(1)} = V$.

Step 2. Choose $v_i \in V^{(i)}$.

Step 3. Compute $R(v_i) \cap Q(v_i)$.

Let $V^{(i+1)} = V^{(i)} \setminus R(v_i) \cap Q(v_i)$.

Let $i \leftarrow i + 1$.

Step 4. If $V^{(i)} = \emptyset$, then STOP.

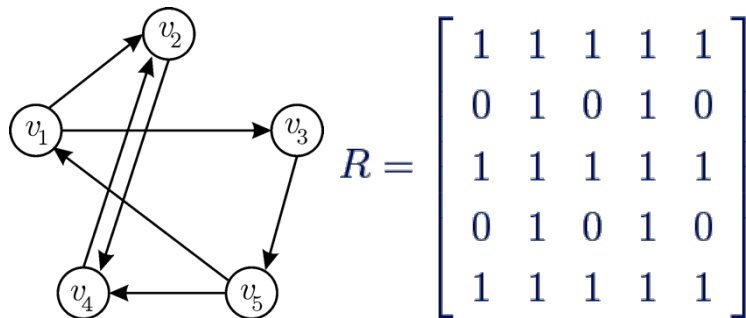
Else, goto Step 2.

Notation:

- $R(v_i)$ denotes the set of vertices that v_i reaches.
- $Q(v_i)$ denotes the set of vertices that reaches to v_i .

2. Computation of the Connected Components

EXAMPLE:



METHOD 1.

Step 1. Initialize $i \leftarrow 1$, $V^{(1)} = V$.

Step 2. Choose $v_i \in V^{(i)}$.

Step 3. Compute $R(v_i) \cap Q(v_i)$.

Let $V^{(i+1)} = V^{(i)} \sim R(v_i) \cap Q(v_i)$.

Let $i \leftarrow i+1$.

Step 4. If $V^{(i)} = \emptyset$, then STOP.

Else, goto Step 2.

Notation:

- $R(v_i)$ denotes the set of vertices that v_i reaches.
- $Q(v_i)$ denotes the set of vertices that reaches to v_i .

ITERATION 1.

Step 1. $i \leftarrow 1$, $V^{(1)} = V = \{v_1, v_2, v_3, v_4, v_5\}$.

Step 2. Choose a vertex from $V^{(1)}$, for example v_2 .

Step 3. First Connected Component: $R(v_2) \cap Q(v_2) = \{v_2, v_4\} \cap \{v_1, v_2, v_3, v_4, v_5\} = \{v_2, v_4\}$.

$V^{(2)} = V^{(1)} \sim R(v_2) \cap Q(v_2) = \{v_1, v_2, v_3, v_4, v_5\} \sim \{v_2, v_4\} = \{v_1, v_3, v_5\}$.

$i \leftarrow 2$.

Step 4. $V^{(i)} \neq \emptyset$, then goto Step 2.

ITERATION 2.

Step 2. Choose a vertex from $V^{(2)}$, for example v_1 .

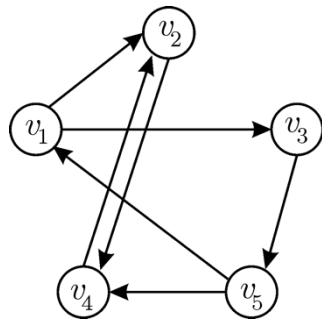
Step 3. Second Connected Component: $R(v_1) \cap Q(v_1) = \{v_1, v_2, v_3, v_4, v_5\} \cap \{v_1, v_3, v_5\} = \{v_1, v_3, v_5\}$.

$V^{(3)} = V^{(2)} \sim R(v_1) \cap Q(v_1) = \{v_1, v_3, v_5\} \sim \{v_1, v_3, v_5\} = \emptyset$.

Step 4. $V^{(3)} = \emptyset$ then STOP.

2. Computation of the Connected Components

EXAMPLE:

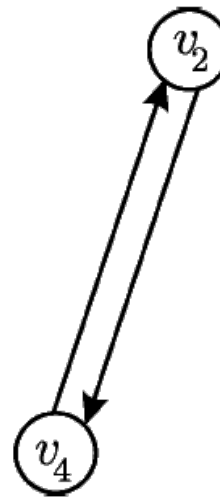
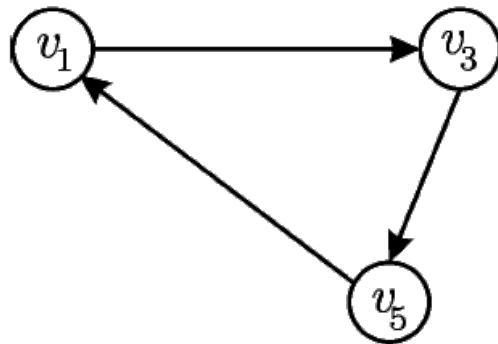


$$R = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The connected component are:

$$\{v_2, v_4\}, \{v_1, v_3, v_5\}.$$

Graphically

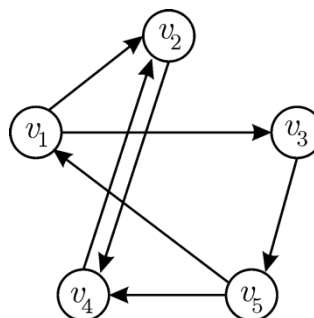


2. Computation of the Connected Components

METHOD 2.

Another way to compute the connected components is to compute $R \otimes Q$. The connected component of x_i is the set of vertices whose columns have a 1 in row i .

EXAMPLE:

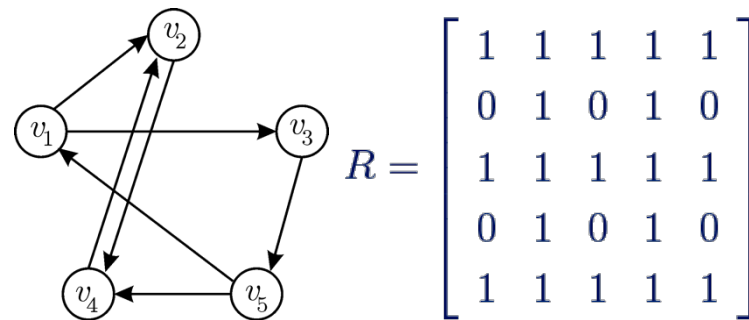


$$R = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$R \otimes Q = R \otimes R^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

2. Computation of the Connected Components

EXAMPLE:



$$R \otimes Q = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Looking at rows 1st, 3rd and 5th, we conclude that a connected component is

$$\{v_1, v_3, v_5\}$$

2nd and 4th rows indicate the other connected component

$$\{v_2, v_4\}$$

2. Computation of the Connected Components

Remark: In the undirected case is obvious that the connected component associated with a vertex x_i can be calculated by obtaining the set:

$$R(v_i) = \{v_i\} \cup \Gamma(v_i) \cup \dots \cup \Gamma^p(v_i), \quad p \leq n$$

3. Euler trails and Euler tours.

The Seven Bridges of Königsberg. The town of Königsberg, Prussia (now called Kaliningrad and part of the Russian republic), was divided into four sections by the branches of the Pregel River. These four sections included the two regions on the banks of the Pregel, Kneiphof Island, and the region between the two branches of the Pregel. In the eighteenth century seven bridges connected these regions. Figure 1 depicts these regions and bridges. The townspeople took long walks through town on Sundays. They wondered whether it was possible to start at some location in the town, travel across all the bridges without crossing any bridge twice, and return to the starting point.

The Swiss mathematician Leonhard Euler solved this problem. His solution, published in 1736, may be the first use of graph theory. Euler studied this problem using the multigraph obtained when the four regions are represented by vertices and the bridges by edges. This multigraph is shown in Figure 2. The problem of traveling across every bridge without crossing any bridge more than once can be rephrased in terms of this model. The question becomes: Is there a closed trail (no edge is repeated) in this multigraph that contains every edge?

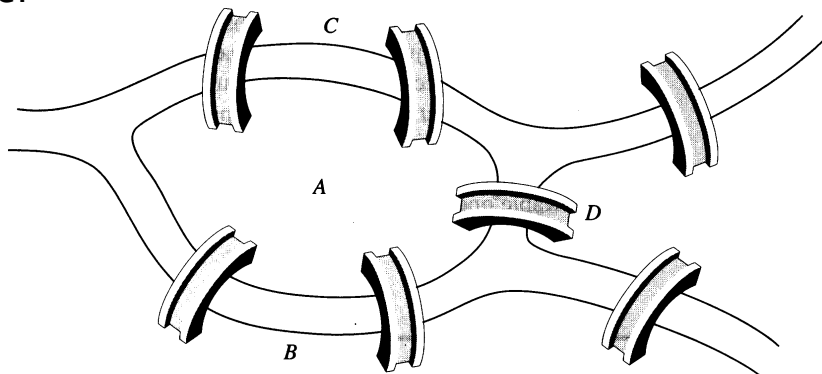


FIGURE 1. The Seven Bridges of Königsberg.

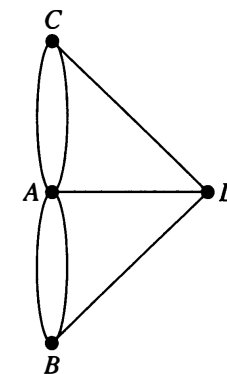


FIGURE 2. Multigraph Model of the Town of Königsberg.

3. Euler trails and Euler tours.

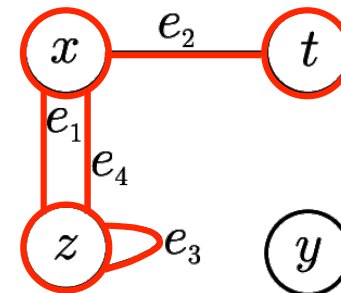
DEFINITIONS:

Let G be a graph not necessarily simple.

1. A **tour** in G is a closed walk that traverses every edge of the graph at least once.

EXAMPLE:

The walk $te_2xe_1ze_3ze_4xe_2t$ is a tour.

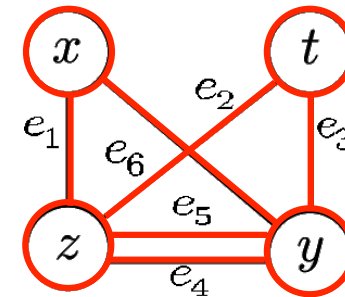


3. Euler trails and Euler tours.

2. An **Euler tour** (or Eulerian tour) in G is a closed walk that traverses every edge of the graph exactly once.

EXAMPLE:

The walk $te_3ye_4ze_1xe_6ye_5ze_2t$ is an Euler tour.



3. A graph G is said to be an **Eulerian graph** if G has an Euler tour.

EXAMPLE: The above graph is an Eulerian graph.

3. Euler trails and Euler tours.

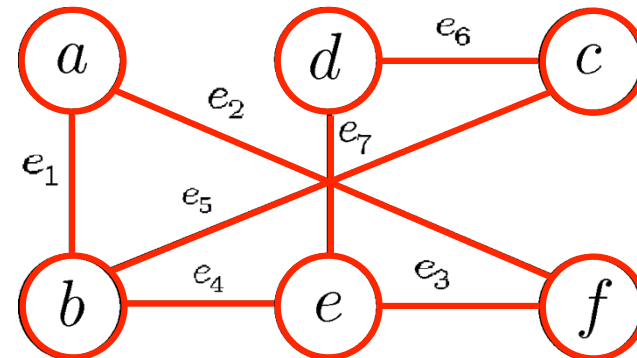
4. If there is an open walk from a to b in G and this walk traverses each edge in G exactly once, the trail is called an **Euler trail**.

EXAMPLE:

The walk

$be_1ae_2fe_3ee_4be_5ce_6de_7e$

is an Euler trail.



3. Euler trails and Euler tours.

THEOREM

Let G be an undirected and connected graph.

1. G is an Eulerian graph if and only if it has not vertices of odd degree.
2. G has a Euler trail if and only if it has exactly two vertices of odd degree.

THEOREM

Let $G = (V, A)$ be a directed and weakly connected graph.

1. G is an Eulerian graph if and only if, for all $v \in V$, $d_{in}(v) = d_{out}(v)$.
2. G has an Euler trail if and only if

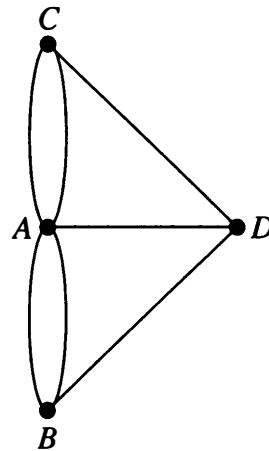
$$\begin{aligned} d_{in}(v) &= d_{out}(v), \quad \forall v \neq p, q. \\ d_{in}(p) &= d_{out}(p) - 1, \quad d_{in}(q) = d_{out}(q) + 1. \end{aligned}$$

Where p and q are the starting and ending vertices, respectively, of the path.

3. Euler trails and Euler tours.

EXAMPLE:

We can now solve the Königsberg bridge problem. Because the multigraph representing these bridges, has four vertices of odd degree, it does not have an Euler tour. There is no way to start at a given point, cross each bridge exactly once, and return to the starting point.



Moreover, there is no an Euler trail.

3. Euler trails and Euler tours.

FLEURY'S ALGORITHM

The following algorithm, called Fleury's algorithm, constructs an Euler tour or an Euler trail in an undirected connected graph.

If the graph is Eulerian, starting with an arbitrary vertex of G , we will form a walk by choosing edges successively. Once an edge is chosen, it is removed. Edges are chosen successively so that each edge begins where the last edge ends, and so that this edge is not a cut edge unless there is no alternative.

After this process, ie, when we have used all the edges, we have obtained an Euler tour.

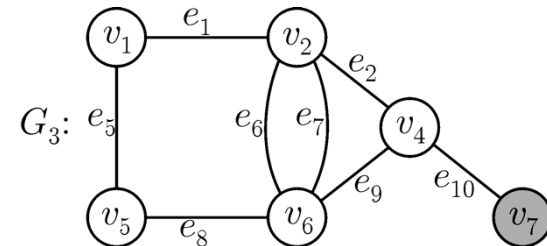
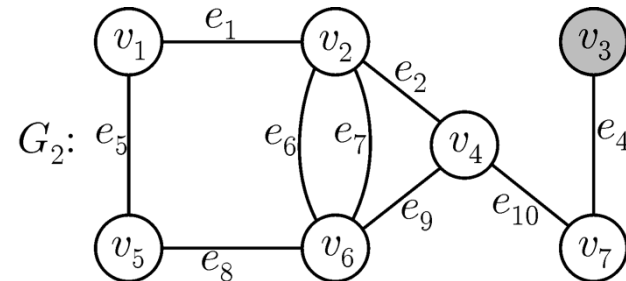
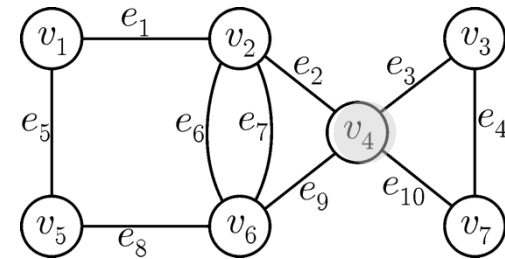
If the graph has an Euler trail we will start with a vertex of odd degree following the process described.

3. Euler trails and Euler tours.

EXAMPLE: The following graph is connected and all the vertices have even degree. Therefore, it has an Euler tour. We will use Fleury's algorithm.

Starting with v_4 we can choose 4 possible edges; none of which disconnects the graph. Choosing edge e_3 , we remove it from the graph and we add it to the $T=e_3$.

Vertex v_3 is incident only with e_4 . We remove e_4 and v_3 . We add e_4 to the tour: $T=e_3e_4$.



3. Euler trails and Euler tours.

Current situation: $T = e_3 e_4$.

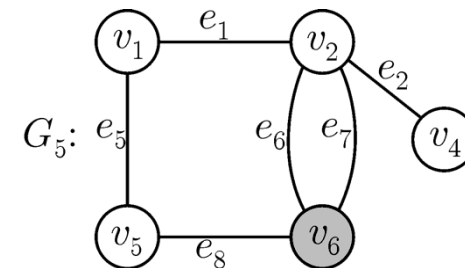
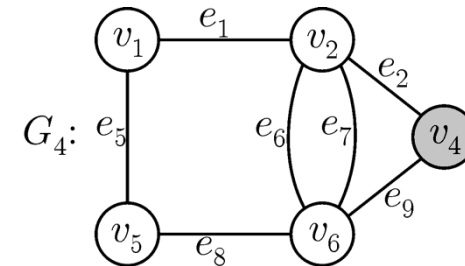
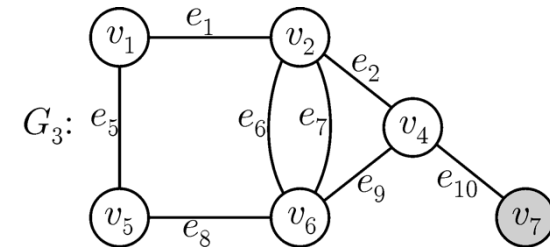
Vertex v_7 is incident only with e_{10} .

We remove e_{10} and v_7 .

We add e_{10} to the tour: $T = e_3 e_4 e_{10}$.

From v_4 we can choose 2 edges and none of them disconnect the graph: we choose e_9 .

We remove e_9 and add it to the tour: $T = e_3 e_4 e_{10} e_9$.



3. Euler trails and Euler tours.

Current situation: $T = e_3 e_4 e_{10} e_9$.

From v_6 we can choose 3 edges and none of them disconnect the graph: we choose e_7 .

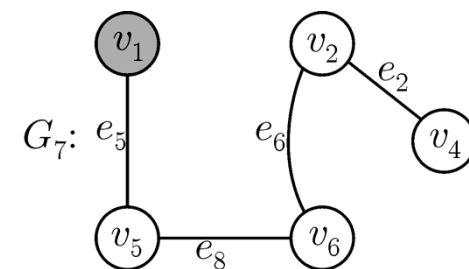
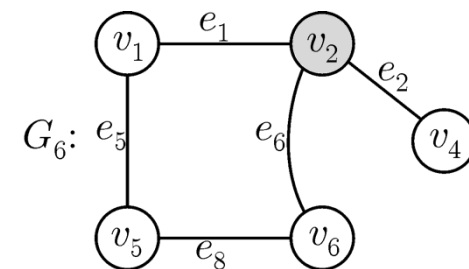
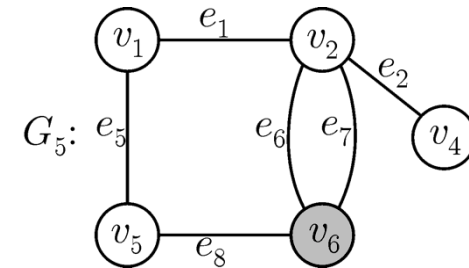
We remove e_7 and add it to the tour:

$T = e_3 e_4 e_{10} e_9 e_7$.

From v_2 we can choose e_1 , e_2 or e_6 . e_2 is a cut edge, then we can't choose it: we choose e_1 which is not a cut edge.

We remove e_1 and add it to the tour:

$T = e_3 e_4 e_{10} e_9 e_7 e_1$.

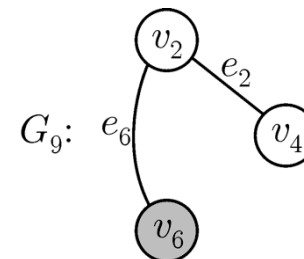
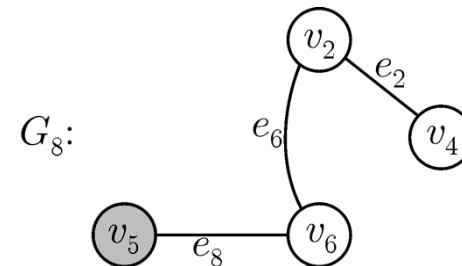
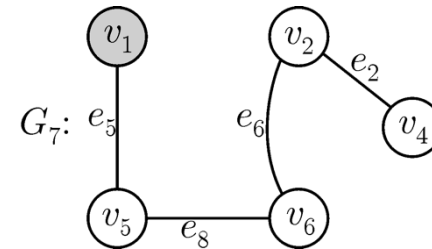


3. Euler trails and Euler tours.

Current situation: $T = e_3 e_4 e_{10} e_9 e_7 e_1$.

Vertex v_1 is incident only with e_5 .
We remove e_5 and v_1 . We add e_5 to the tour:
 $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5$.

Vertex v_5 is incident only with e_8 .
We remove e_8 and v_5 . We add e_8 to the tour:
 $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5 e_8$.



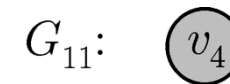
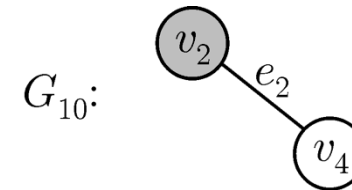
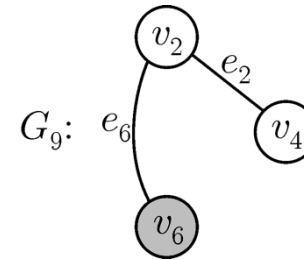
3. Euler trails and Euler tours.

Current situation: $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5 e_8$.

Vertex v_6 is incident only with e_6 . We remove e_6 and v_6 . We add e_6 to the tour:
 $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5 e_8 e_6$.

Vertex v_2 is incident only with e_2 . We remove e_2 and v_2 . We add e_2 to the tour:
 $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5 e_8 e_6 e_2$.

Since there is no more edges, we have obtained an Euler tour $T = e_3 e_4 e_{10} e_9 e_7 e_1 e_5 e_8 e_6 e_2$.



3. Euler trails and Euler tours.

FLEURY'S ALGORITHM IN DIRECTED GRAPHS

The following algorithm constructs an Euler tour or an Euler trail in a directed connected graph.

If the graph is Eulerian, starting with an arbitrary vertex of G , we will form a walk by choosing arcs successively. Once an edge is chosen, it is removed, arcs are chosen successively so that each arc begins where the last arc ends, and so that this arc is not a cut edge of the undirected associated graph unless there is no alternative.

If the graph has an Euler trail we will start with a vertex p such that $d_{in}(\mathbf{p}) = d_{out}(\mathbf{p}) - 1$, following the process described.

3. Euler trails and Euler tours.

EXAMPLE:

Let us consider the following directed graph $G=(V,A)$:

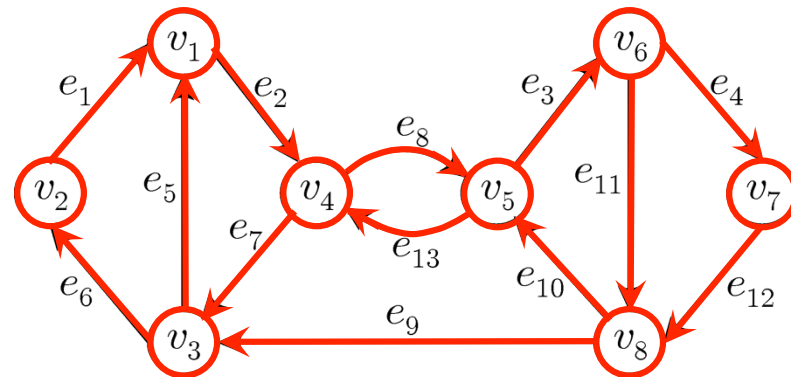
The graph is weakly connected and satisfy:

$$d_{in}(\mathbf{v}_i) = d_{out}(\mathbf{v}_i) \text{ for all } i \neq 1, 6$$

$$d_{in}(\mathbf{v}_6) = d_{out}(\mathbf{v}_6) - 1,$$

$$d_{in}(\mathbf{v}_1) = d_{out}(\mathbf{v}_1) + 1$$

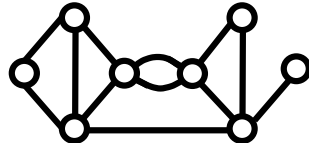
Then this graph has an Euler trail starting in \mathbf{v}_6 and ending in \mathbf{v}_1 .



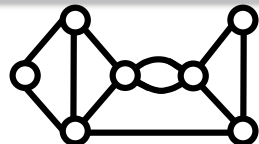
3. Euler trails and Euler tours.

FLEURY'S ALGORITHM:

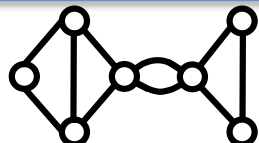
Iteration 1: $T = e_4$



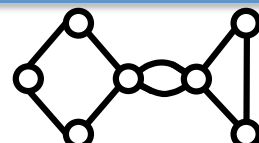
Iteration 2: $T = e_4 e_{12}$



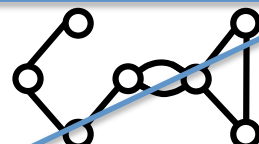
Iteration 3: $T = e_4 e_{12} e_9$



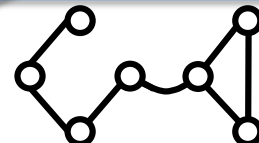
Iteration 4: $T = e_4 e_{12} e_9 e_5$



Iteration 5: $T = e_4 e_{12} e_9 e_5 e_2$



Iteration 6:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8$



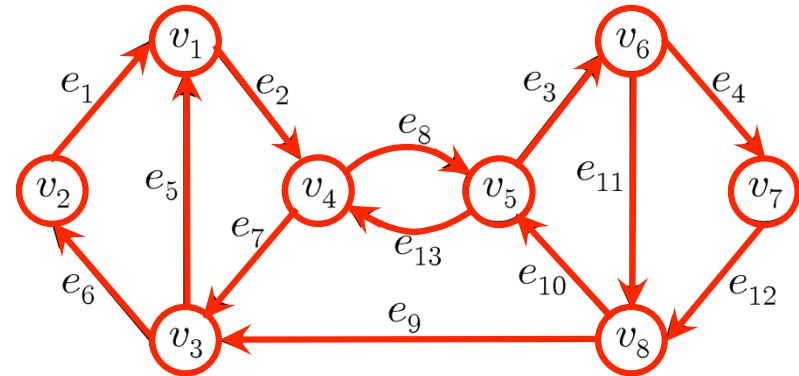
Iteration 7:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3$



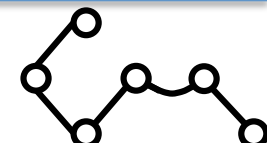
At this point, if we choose the arc e_7 , this arc is a cut edge of the undirected associated graph. Then, we have to choose e_8 .

At this point, if we choose the arc e_{13} , this arc is a cut edge of the undirected associated graph. Then, we have to choose e_3 .

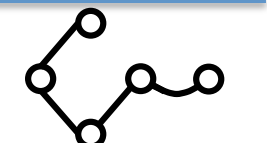
Euler trail



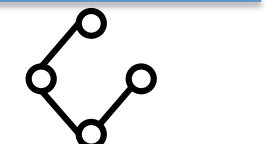
Iteration 8:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11}$



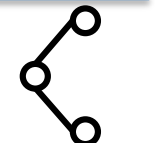
Iteration 9:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11} e_{10}$



Iteration 10:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11} e_{10} e_{13}$



Iteration 11:
 $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11} e_{10} e_{13} e_7$



Iteration 12: $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11} e_{10} e_{13} e_7 e_6$



Iteration 13: $T = e_4 e_{12} e_9 e_5 e_2 e_8 e_3 e_{11} e_{10} e_{13} e_7 e_6 e_1$