Today we will discuss two sections:

Section 10.4 *Connectivity*
(paths and isomorphism)

Section 10.5 *Euler and Hamilton Paths*

10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

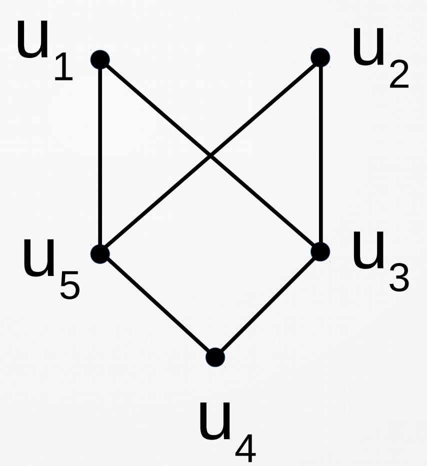
Example:

10.4 *Connectivity*

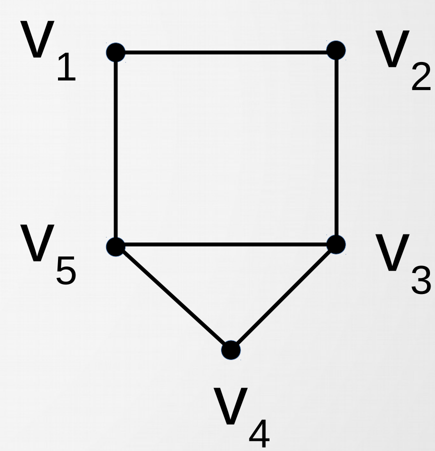
Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

G_2 and G_1 are not isomorphic

10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:

All three invariants hold:

$|U| = |V|$ $|E_1| = |E_2|$

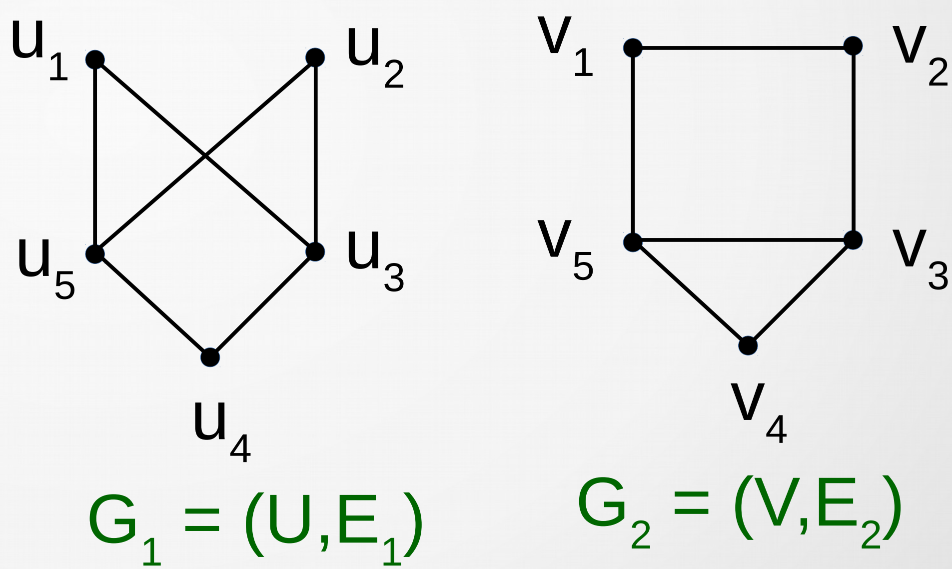
$deg(u_5) = deg(u_3) = 3$

$deg(v_5) = deg(v_3) = 3$

$deg(u_1) = deg(u_2) = 2$

$deg(v_1) = deg(v_2) = 2$

$deg(u_4) = deg(v_4) = 1$



10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:

All three invariants hold:

$|U| = |V|$ $\quad$ $|E_1| = |E_2|$

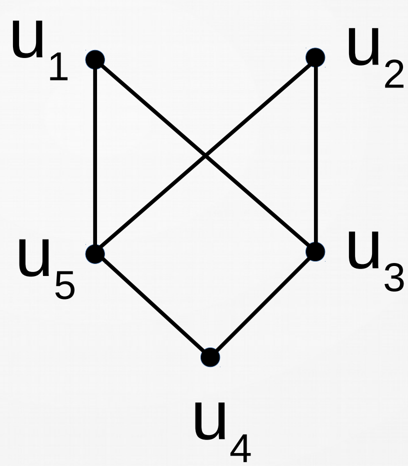
$deg(u_5) = deg(u_3) = 3$

$deg(v_5) = deg(v_3) = 3$

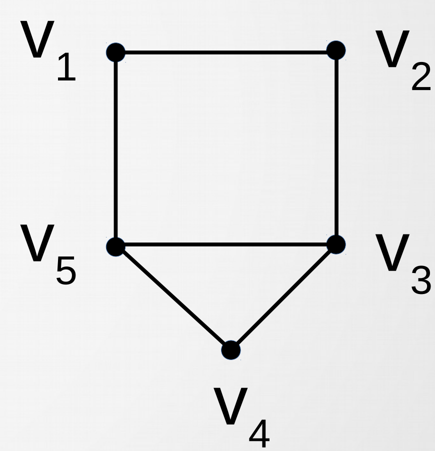
$deg(u_1) = deg(u_2) = 2$

$deg(v_1) = deg(v_2) = 2$

$deg(u_4) = deg(v_4) = 1$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

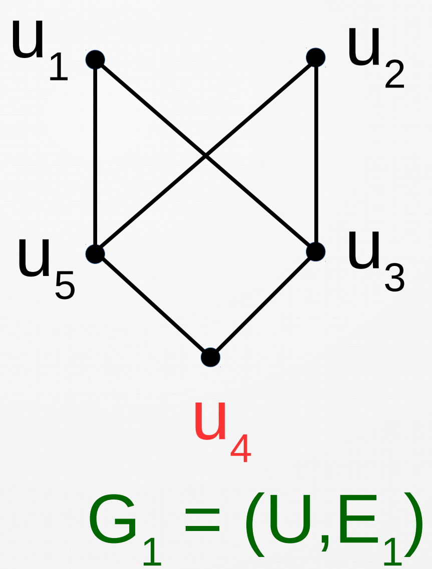
However, G_2 has a simple circuit of length 3, and G_1 doesn't.

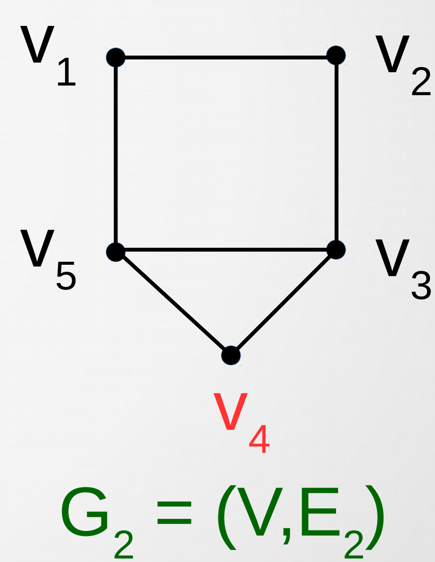
10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:



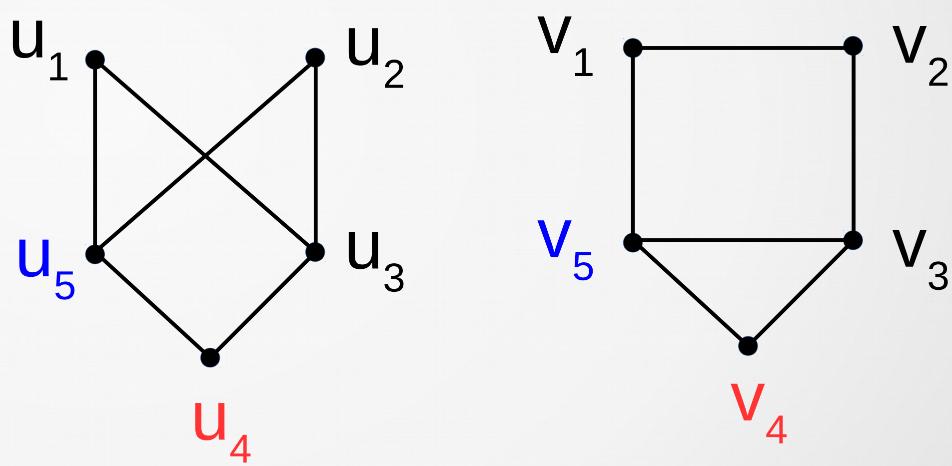


10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:



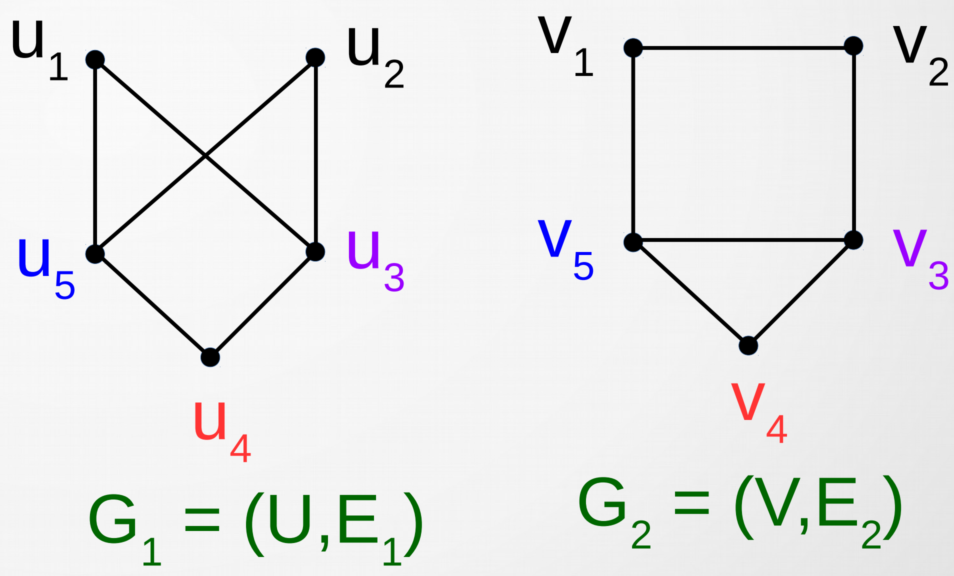
$G_1 = (U,E_1)$ $G_2 = (V,E_2)$

10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:

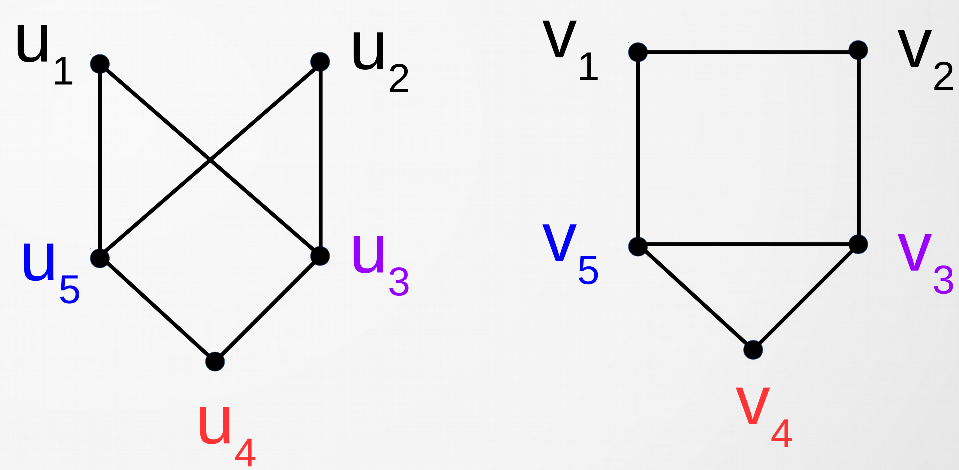


10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:



$G_1 = (U,E_1)$ $G_2 = (V,E_2)$

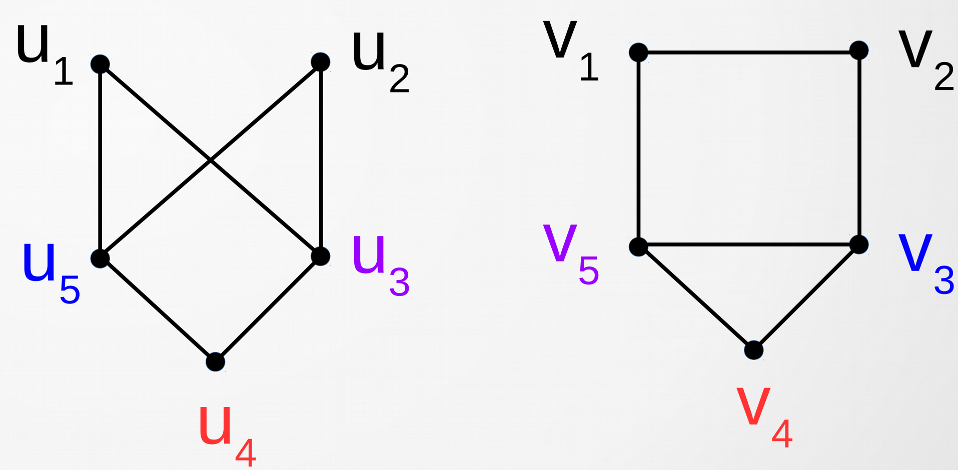
$(v_5,v_3) \in E_2$, but $(u_5, u_3) \notin E_1$.

10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:



$G_1 = (U,E_1)$ $G_2 = (V,E_2)$

same problem

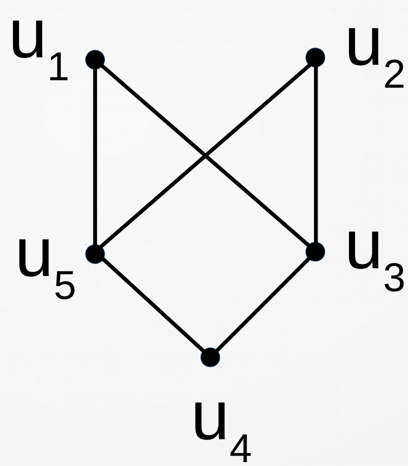
10.4 *Connectivity*

Path and isomorphism

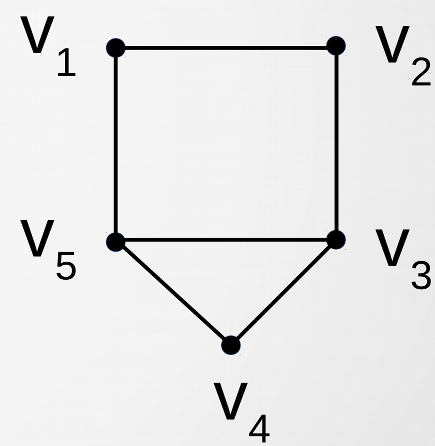
Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:

The existence of simple circuit of a particular length can be used to show that two graphs are not isomorphic



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

G_2 and G_1 are not isomorphic

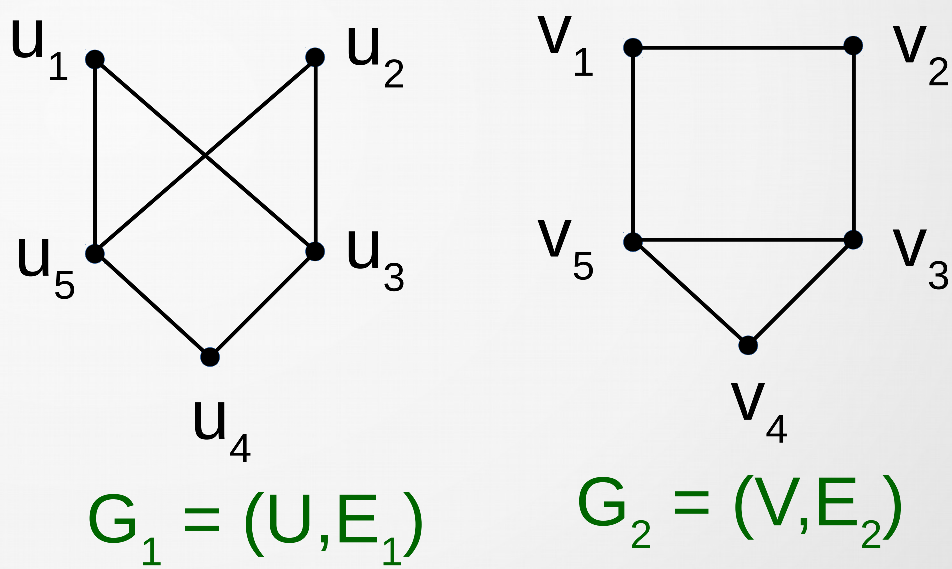
10.4 *Connectivity*

Path and isomorphism

Paths and *circuits* can help determine whether two graphs are isomorphic.

Example:

The existence of simple circuit of a particular length (a fourth invariant) can be used to show that two graphs are not isomorphic



G_2 and G_1 are not isomorphic

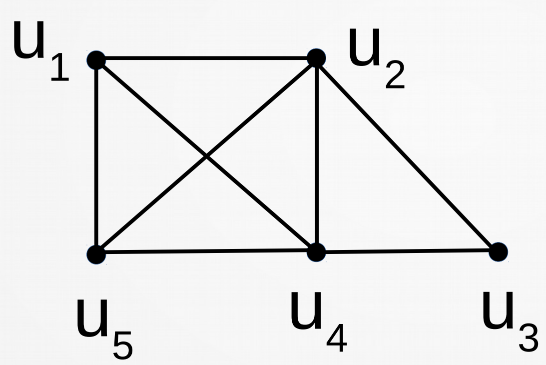
(needs to be proved)

10.4 *Connectivity*

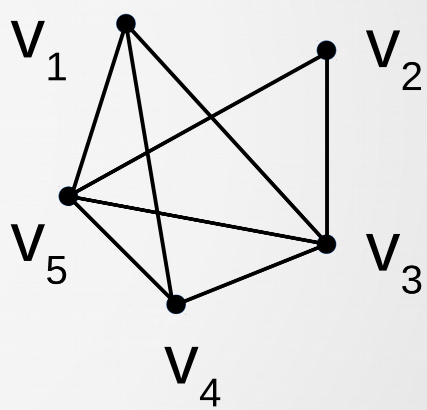
Path and isomorphism

<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

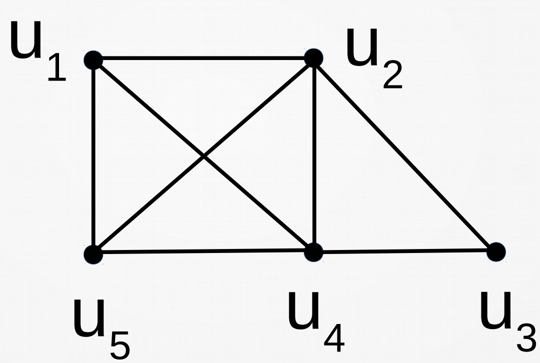
10.4 *Connectivity*

Path and isomorphism

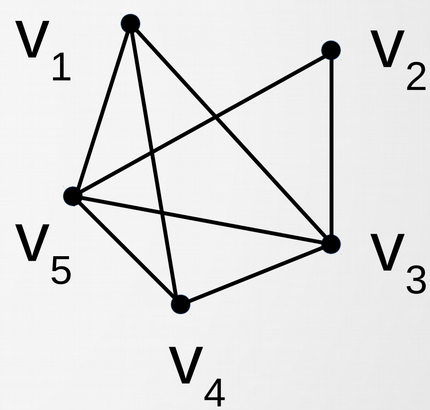
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$

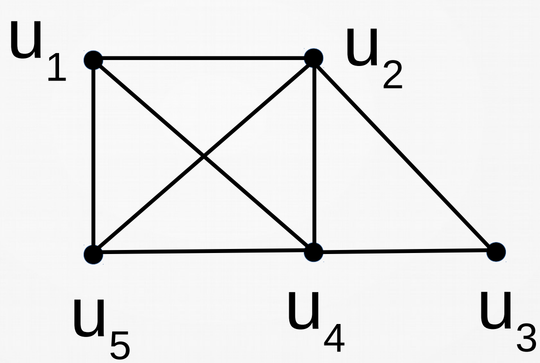
10.4 *Connectivity*

Path and isomorphism

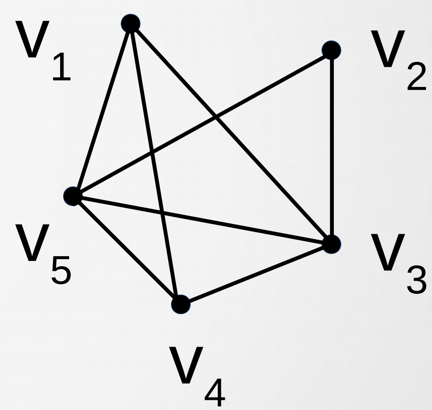
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$

let's build circuits in G_1 and G_2 through corresponding vertices

10.4 *Connectivity*

Path and isomorphism

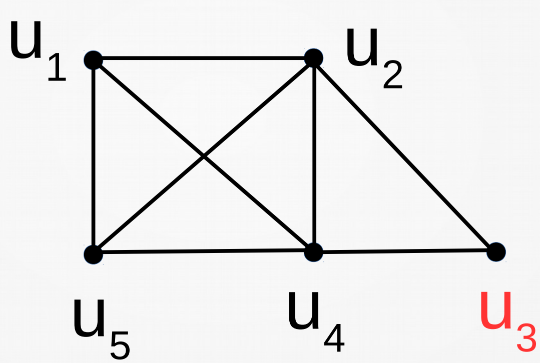
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

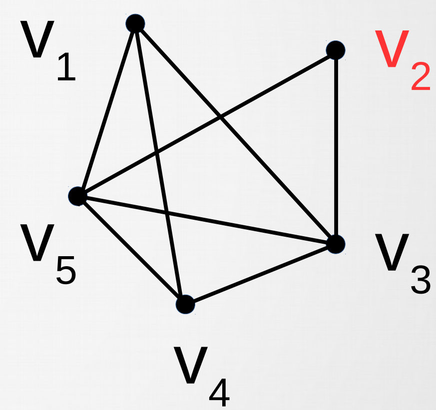
$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

let's build circuits in G_1 and G_2 through corresponding vertices

10.4 *Connectivity*

Path and isomorphism

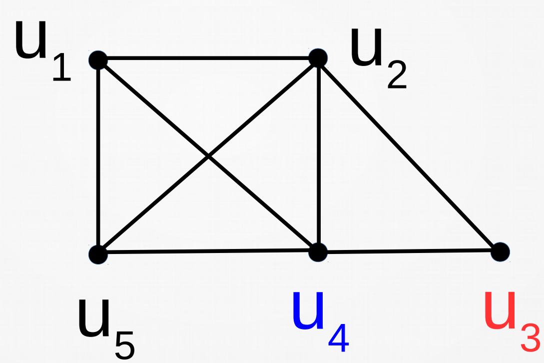
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

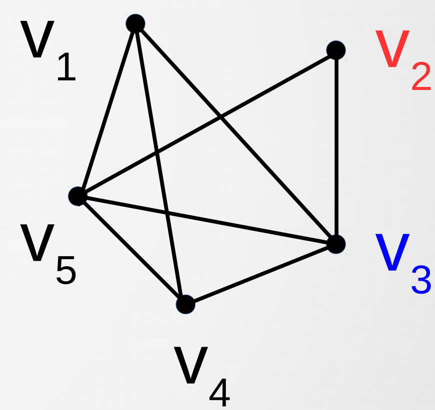
$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

let's build circuits in G_1 and G_2 through corresponding vertices

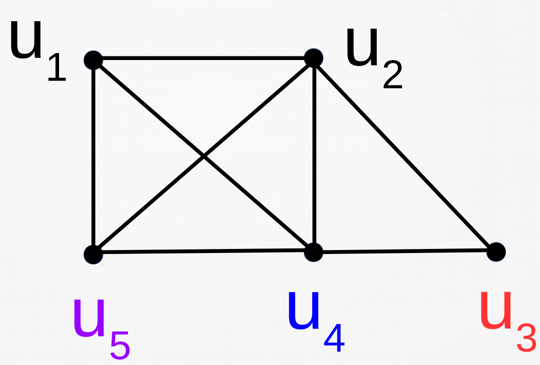
10.4 *Connectivity*

Path and isomorphism

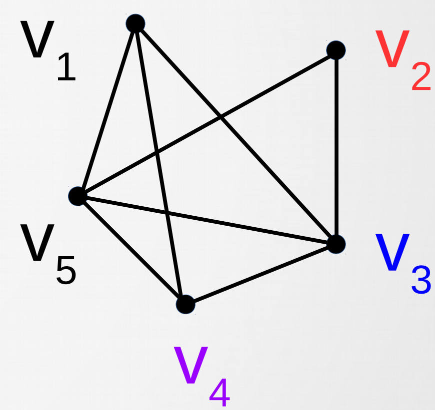
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$

let's build circuits in G_1 and G_2 through corresponding vertices

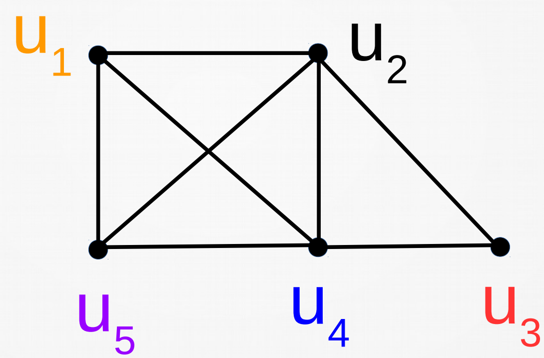
10.4 *Connectivity*

Path and isomorphism

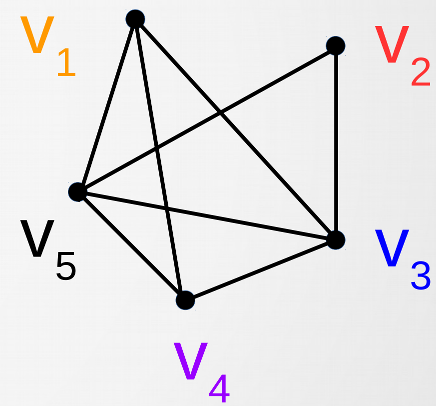
Graph invariants: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$\deg(u_1) = \deg(u_5) = 3$
$\deg(u_2) = \deg(u_4) = 4$
$\deg(u_3) = 2$, $\deg(v_2) = 2$

$\deg(v_1) = \deg(v_4) = 3$
$\deg(v_3) = \deg(v_5) = 4$

let's build circuits in G_1 and G_2 through corresponding vertices

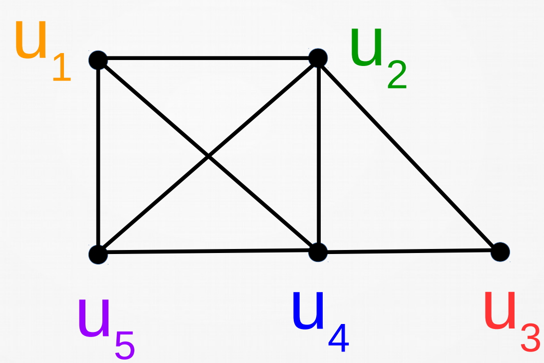
10.4 *Connectivity*

Path and isomorphism

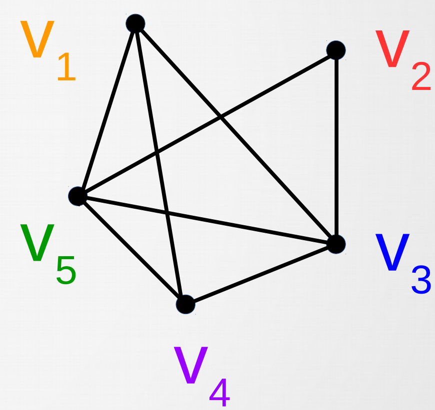
<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$

let's build circuits in G_1 and G_2 through corresponding vertices

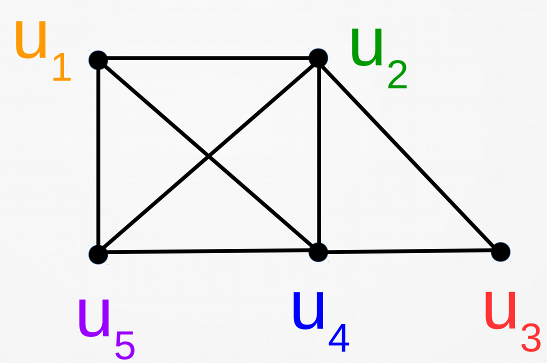
10.4 *Connectivity*

Path and isomorphism

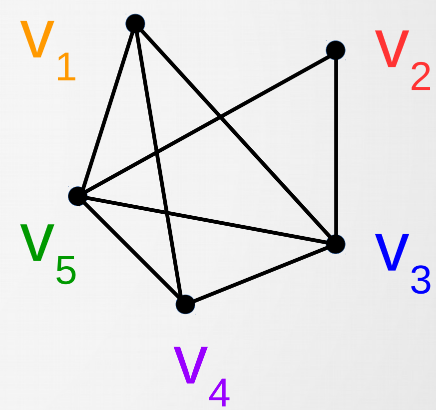
Graph invariants: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:

$|U| = |V| = 5$,
$|E_1| = |E_2| = 8$



$G_1 = (U,E_1)$



$G_2 = (V,E_2)$

$deg(u_1) = deg(u_5) = 3$
$deg(u_2) = deg(u_4) = 4$
$deg(u_3) = 2$, $deg(v_2) = 2$

$deg(v_1) = deg(v_4) = 3$
$deg(v_3) = deg(v_5) = 4$

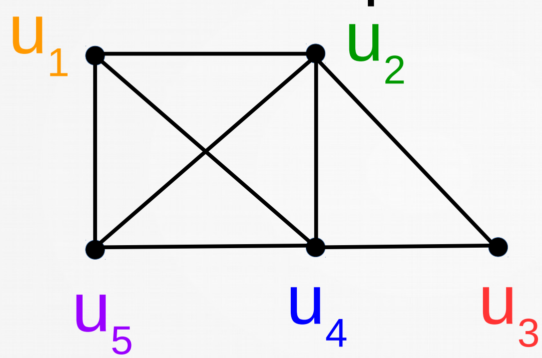
It looks like a possible isomorphism... just need to check it.

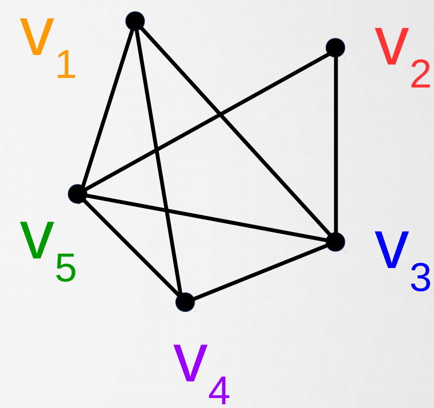
10.4 *Connectivity*

Path and isomorphism

Graph invariants: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:





Let's use adjacency matrix representation of graphs

	u_3	u_4	u_5	u_1	u_2
u_3	0	1	0	0	1
u_4	1	0	1	1	1
u_5	0	1	0	1	1
u_1	0	1	1	0	1
u_2	1	1	1	1	0

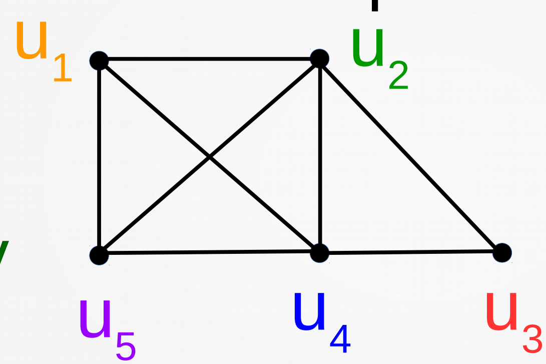
	v_2	v_3	v_4	v_1	v_5
v_2	0	1	0	0	1
v_3	1	0	1	1	1
v_4	0	1	0	1	1
v_1	0	1	1	0	1
v_5	1	1	1	1	0

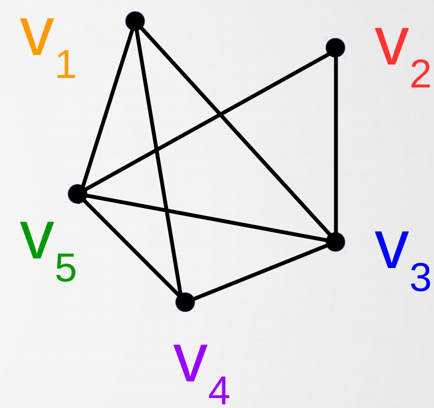
10.4 *Connectivity*

Path and isomorphism

<u>Graph invariants</u>: $|V_1| = |V_2|$, $|E_1| = |E_2|$, degrees of vertices, simple circuits of particular length

Example:





Their adjacency matrices are equal, therefore our mapping is an isomorphism and graphs are isomorphic.

	u_3	u_4	u_5	u_1	u_2
u_3	0	1	1	0	0
u_4	1	0	1	1	1
u_5	0	1	0	1	1
u_1	0	1	1	0	1
u_2	1	1	1	1	0

	v_2	v_3	v_4	v_1	v_5
v_2	0	1	1	0	0
v_3	1	0	1	1	1
v_4	0	1	0	1	1
v_1	0	1	1	0	1
v_5	1	1	1	1	0

10.5 *Euler and Hamilton Paths*

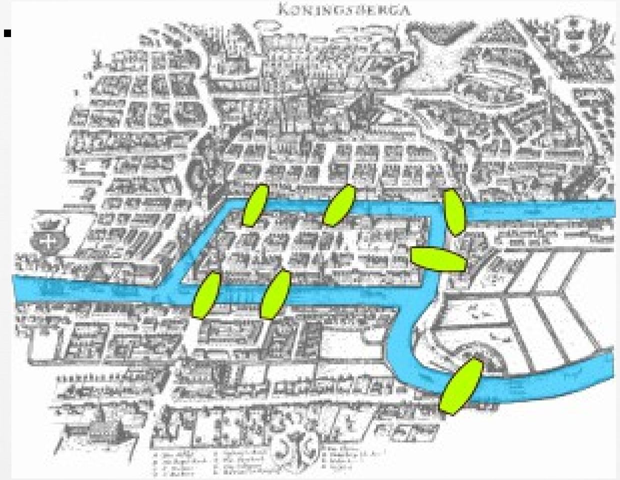
Can we travel long the edges of a graph starting at a vertex and returning to the same vertex by traversing <u>each edge</u> of the graph <u>exactly once</u>?

Can we travel long the edges of a graph starting at a vertex and returning to the same vertex by visiting <u>each vertex</u> in the graph <u>exactly once</u>?

10.5 *Euler and Hamilton Paths*

Town of Königsberg

The city of Königsberg in Prussia (now Kaliningrad, Russia) was set on both sides of the Pregel River, and included two large islands which were connected to each other, or to the two mainland portions of the city, by seven bridges.

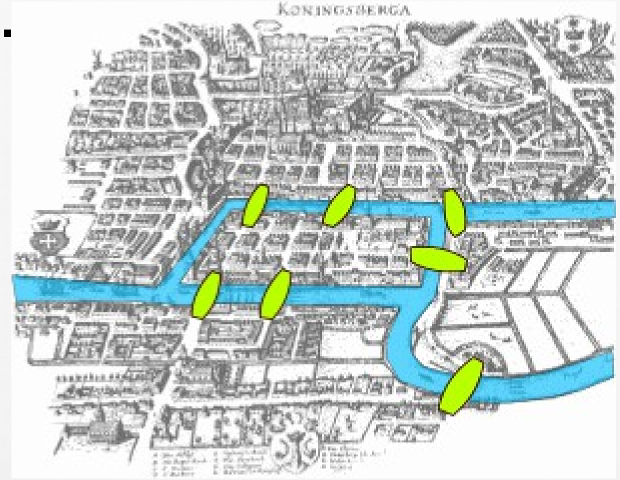


The problem was to devise a walk through the city that would cross each of those bridges once and only once.

10.5 *Euler and Hamilton Paths*

Town of Königsberg

The city of Königsberg in Prussia (now Kaliningrad, Russia) was set on both sides of the Pregel River, and included two large islands which were connected to each other, or to the two mainland portions of the city, by seven bridges.



In 1736 Euler proved that the problem has no solution.

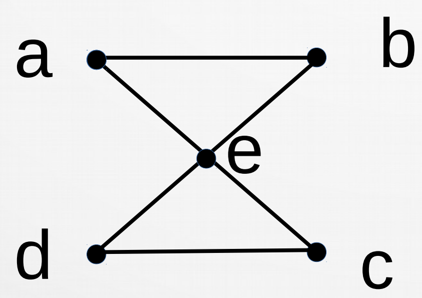
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

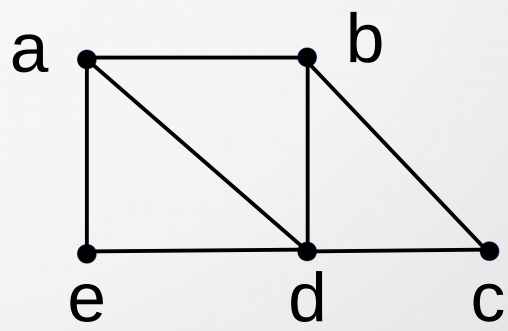
[Def] An *Euler circuit* in a graph G is a simple circuit containing every edge of G.

[Def] An *Euler path* in G is a simple path containing every edge of G

Examples:



G



H

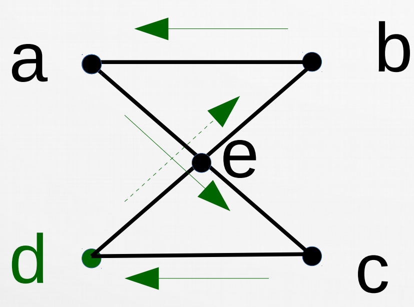
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

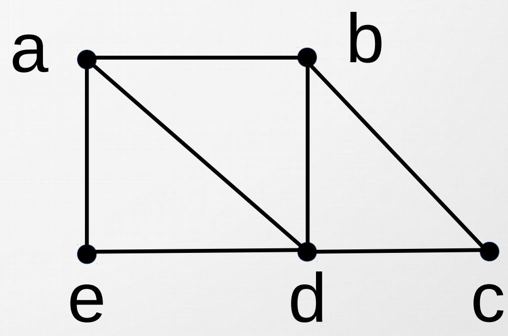
[Def] An *Euler circuit* in a graph G is a simple circuit containing every edge of G.

[Def] An *Euler path* in G is a simple path containing every edge of G

Examples:



G



H

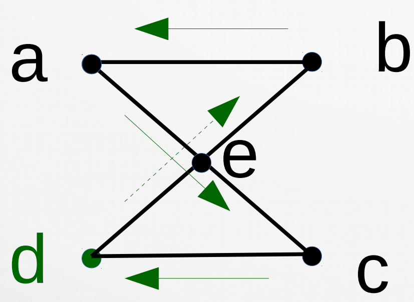
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

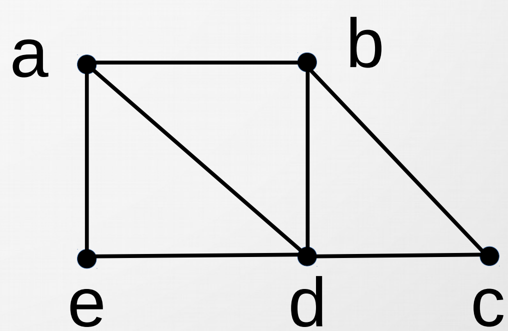
[Def] An *Euler circuit* in a graph G is a simple circuit containing every edge of G.

[Def] An *Euler path* in G is a simple path containing every edge of G

Examples:



G



H

G has an Euler circuit

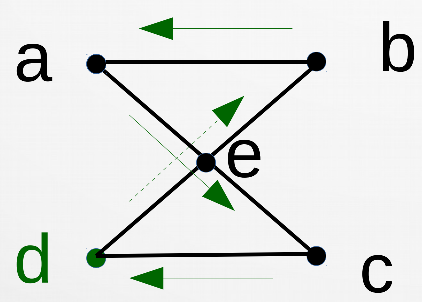
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

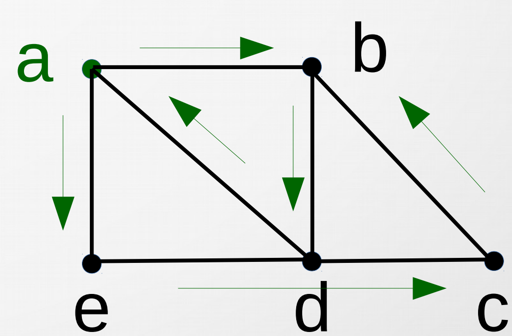
[Def] An *Euler circuit* in a graph G is a simple circuit containing every edge of G.

[Def] An *Euler path* in G is a simple path containing every edge of G

Examples:



G



H

G has an Euler circuit

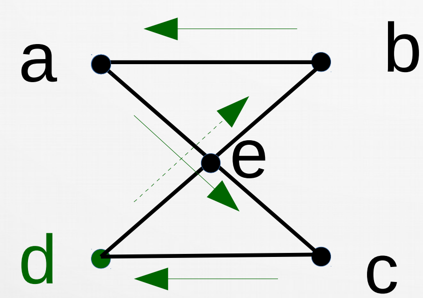
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Def] An *Euler circuit* in a graph G is a simple circuit containing every edge of G.

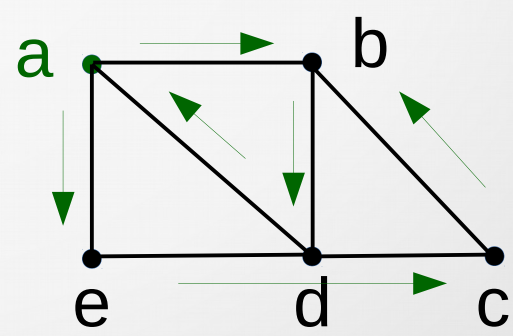
[Def] An *Euler path* in G is a simple path containing every edge of G

Examples:



G

G has an Euler circuit



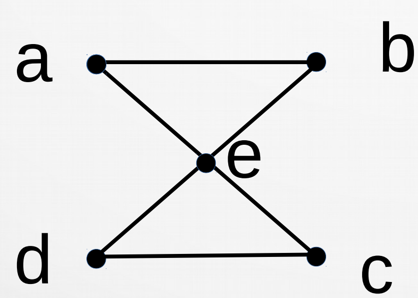
H

H has an Euler path

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph with at least two vertices has an *Euler circuit* if and only iff (iff) each of its vertices has even degree.



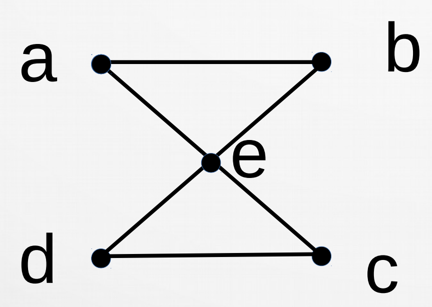
G

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph with at least two vertices has an *Euler circuit* if and only iff (iff) each of its vertices has even degree.

Why is it so?



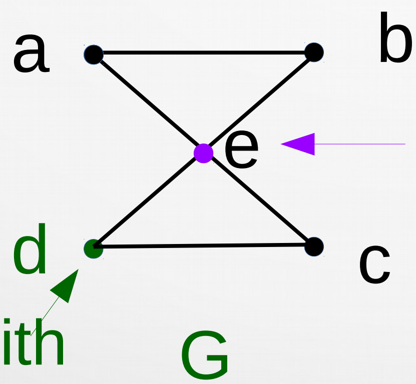
G

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph with at least two vertices has an *Euler circuit* if and only iff (iff) each of its vertices has even degree.

Why is it so?



every time we pass from a vertex we enter it (1) and leave it (1), contributing 2 to its degree.

begin and end with the same vertex (1+1)

10.5 *Euler and Hamilton Paths*

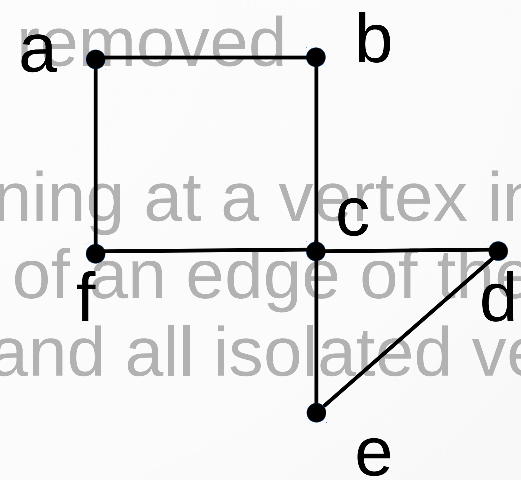
Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate Vertex
return *circuit* {*circuit* is an Euler circuit}

10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

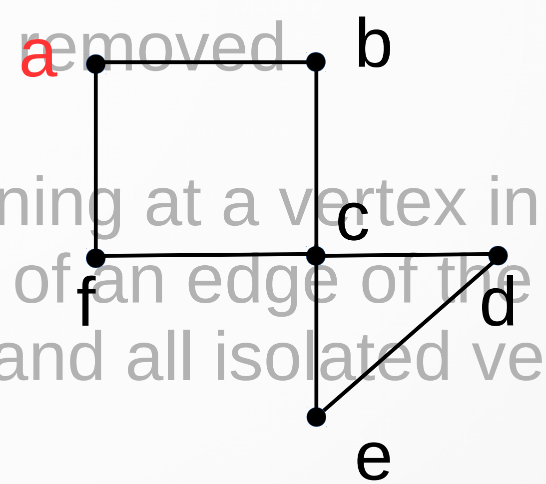
procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
H := H with edges of *subcircuit* and all isolated vertices removed
circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)

pick a vertex, say a

circuit := pick any simple circuit in G that starts at a

H := G with the edges of the *circuit* removed

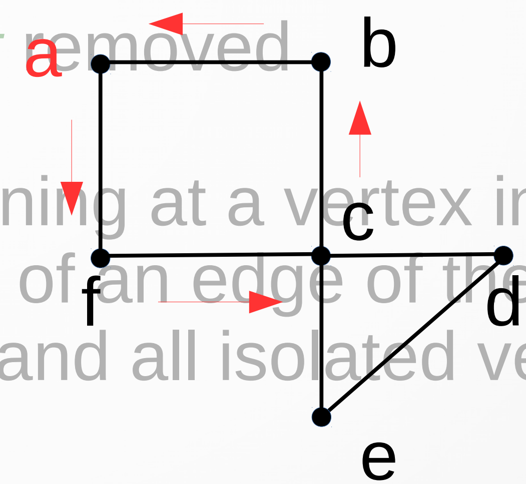
while H has edges

  subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.

  H := H with edges of *subcircuit* and all isolated vertices removed

  circuit := *circuit* with *subcircuit* inserted at the appropriate vertex

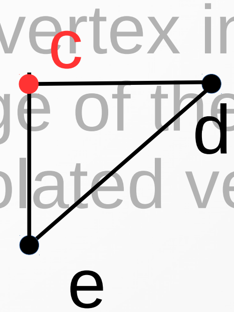
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

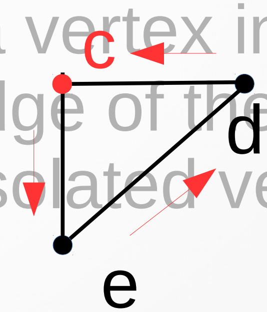
procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
  subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
  H := H with edges of *subcircuit* and all isolated vertices removed
  circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

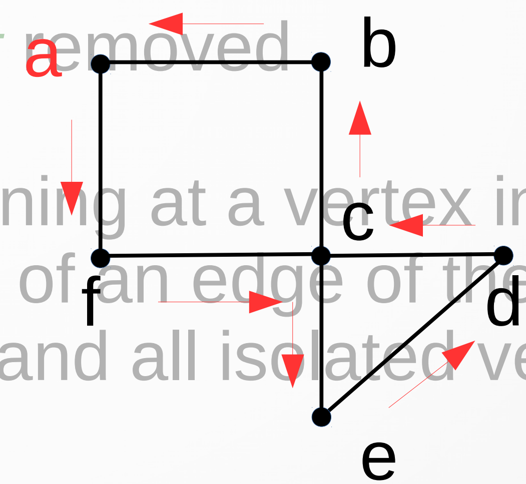
Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}

10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}



10.5 *Euler and Hamilton Paths*

Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)

pick a vertex, say a

circuit := pick any simple circuit in G that starts at a

H := G with the edges of the *circuit* removed

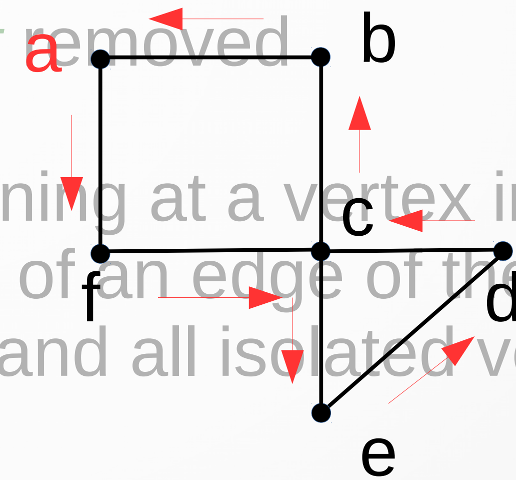
while H has edges

    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.

    H := H with edges of *subcircuit* and all isolated vertices removed

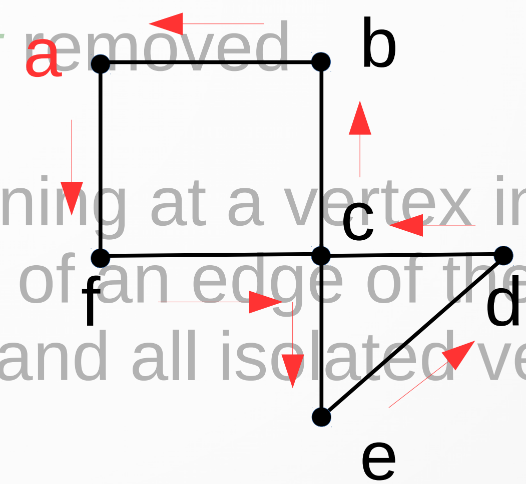
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex

return *circuit* {*circuit* is an Euler circuit}



Algorithm Constructing Euler circuits

procedure Euler(G: connected multigraph with all vertices of even degree)
pick a vertex, say a
circuit := pick any simple circuit in G that starts at a
H := G with the edges of the *circuit* removed
while H has edges
    subcircuit := a circuit in H beginning at a vertex in H that also is an endpoint of an edge of the *circuit*.
    H := H with edges of *subcircuit* and all isolated vertices removed
    circuit := *circuit* with *subcircuit* inserted at the appropriate vertex
return *circuit* {*circuit* is an Euler circuit}

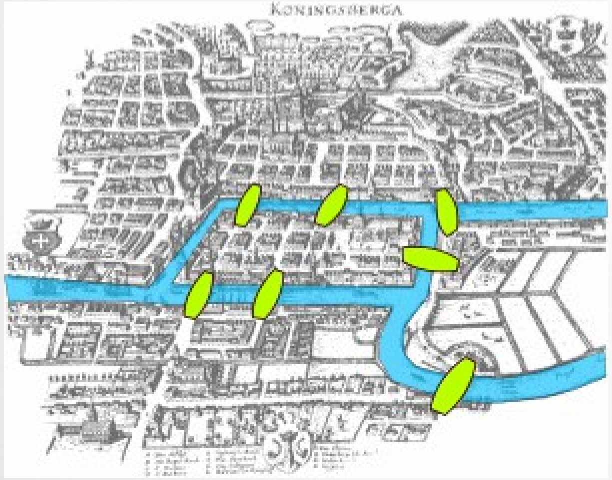


Running time: in worst case: Cm+D, where C, D $\in Z^{+}\{0\}$ const.

10.5 *Euler and Hamilton Paths*

Town of Königsberg

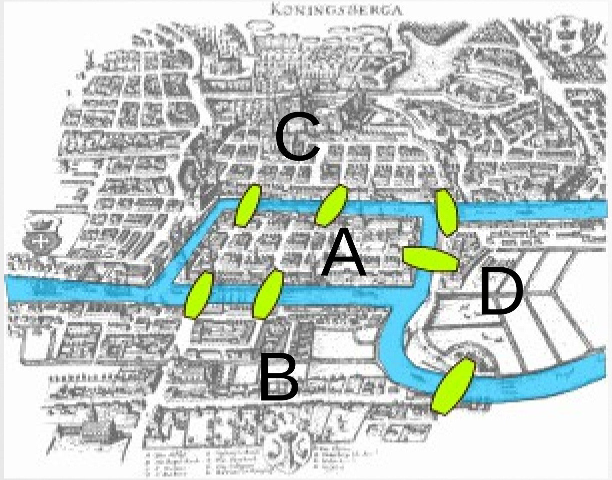
devise a walk through the city that would cross each of those bridges once and only once

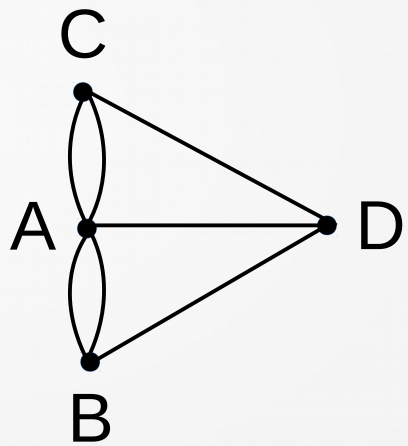


10.5 *Euler and Hamilton Paths*

Town of Königsberg

devise a walk through the city that would cross each of those bridges once and only once



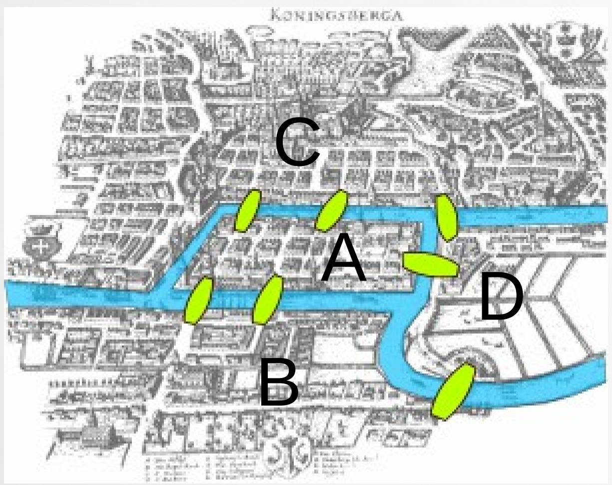


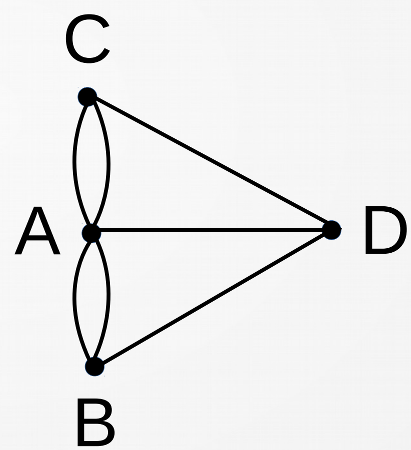
multigraph model of the
Town of Königsberg

10.5 *Euler and Hamilton Paths*

Town of Königsberg

devise a walk through the city that would cross each of those bridges once and only once





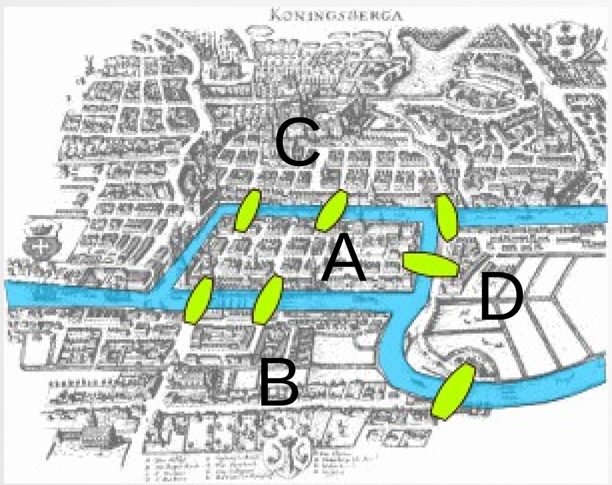
deg(A) = 5
deg(C) = 3
deg(B) = 3
deg(D) = 3

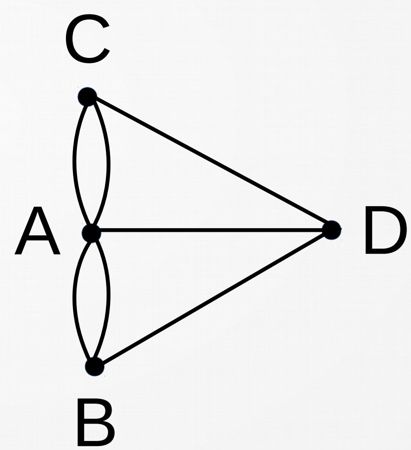
multigraph model of the
Town of Königsberg

10.5 *Euler and Hamilton Paths*

Town of Königsberg

devise a walk through the city that would cross each of those bridges once and only once





deg(A) = 5
deg(C) = 3
deg(B) = 3
deg(D) = 3

multigraph model of the Town of Königsberg

There is no Euler circuit, hence no such walk through the city.

10.5 *Euler and Hamilton Paths*

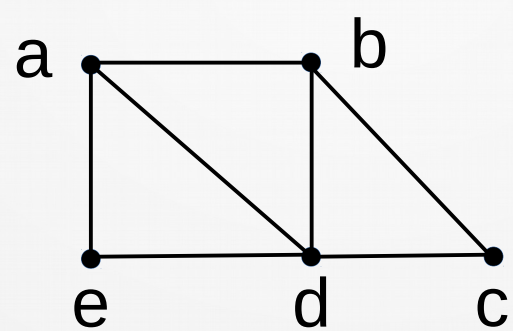
Algorithm Constructing Euler circuits

Another algorithm for constructing Euler circuit is described in Exercise 50.

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph has an *Euler path* but not an *Euler circuit* iff it has exactly two vertices of odd degree.

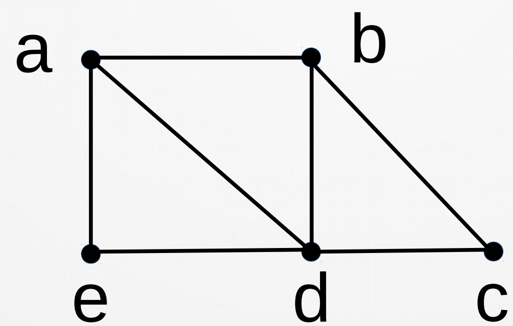


H

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph has an *Euler path* but not an *Euler circuit* iff it has exactly two vertices of odd degree.



H

deg(a) = 3
deg(b) = 3
deg(c) = 2
deg(d) = 4
deg(e) = 2

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

[Theorem] A connected multigraph has an *Euler path* but not an *Euler circuit* iff it has exactly two vertices of odd degree.

Why is it so? Check page 697 for the explanation

10.5 *Euler and Hamilton Paths*

Euler circuits and paths

Another type of puzzle: draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.

The book shows the example with Mohammed's Scimitar (see page 697).

We will consider Witch's knot.

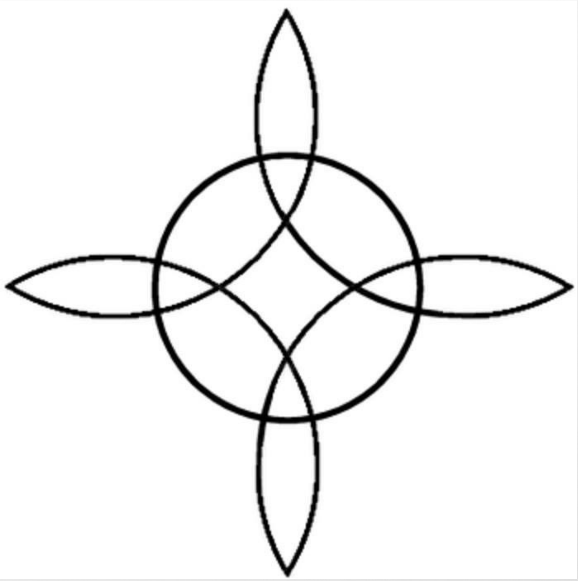
10.5 *Euler and Hamilton Paths*

Euler circuits and paths

Another type of puzzle: draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.

The book shows the example with Mohammed's Scimitar (see page 697).

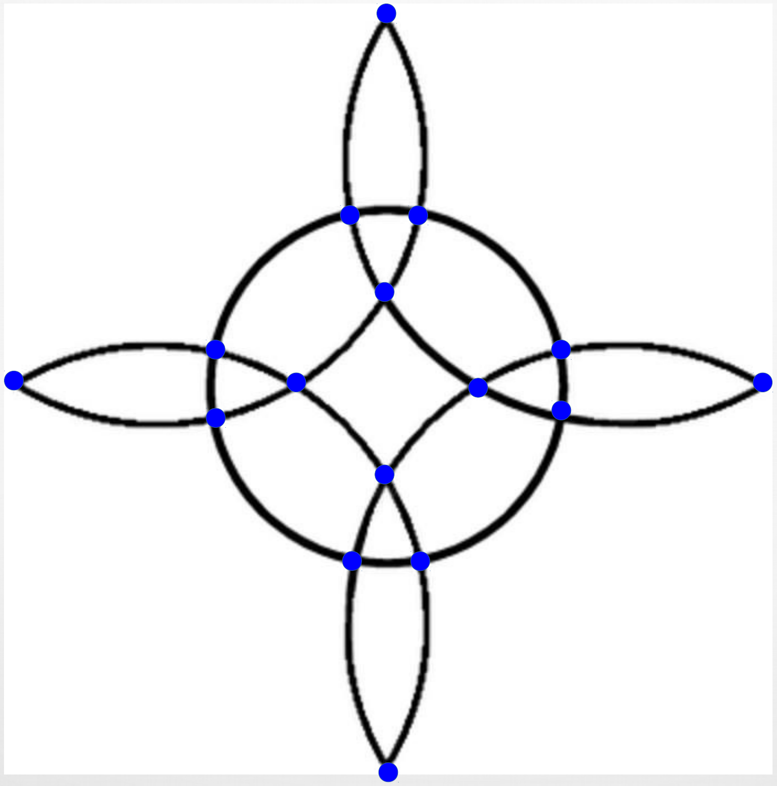
We will consider Witch's knot.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

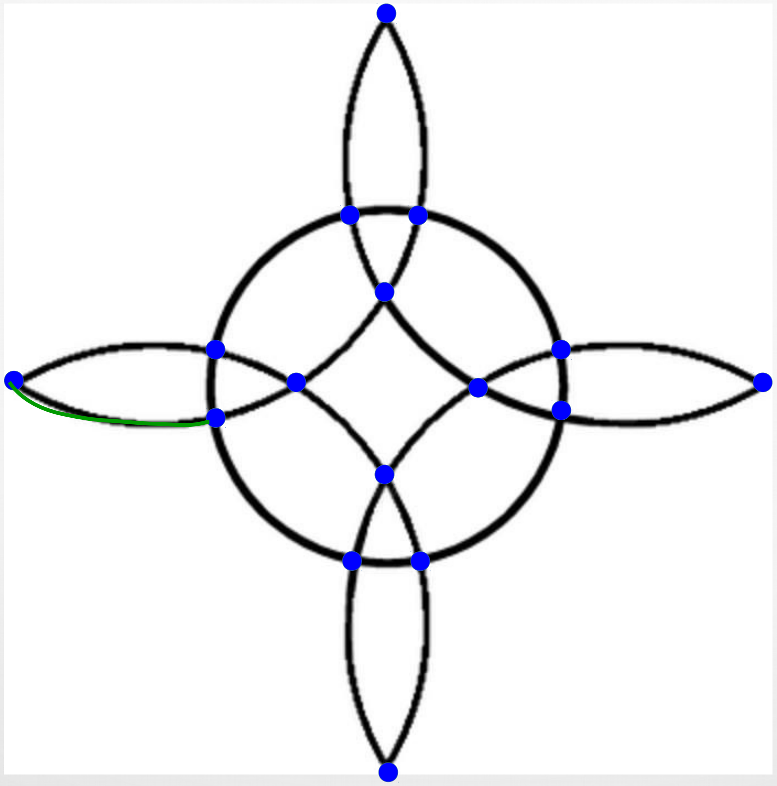
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

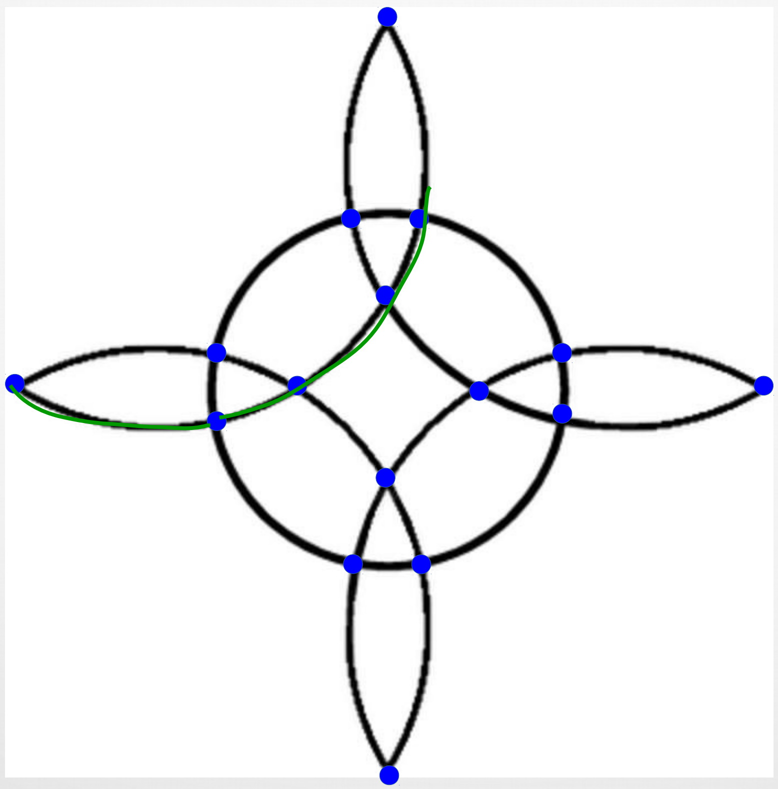
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

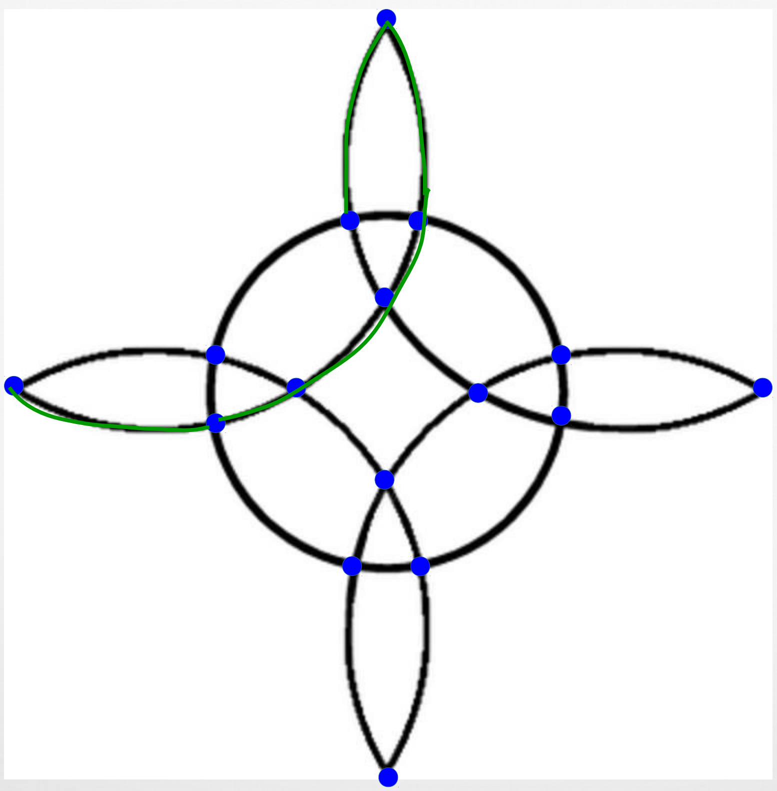
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

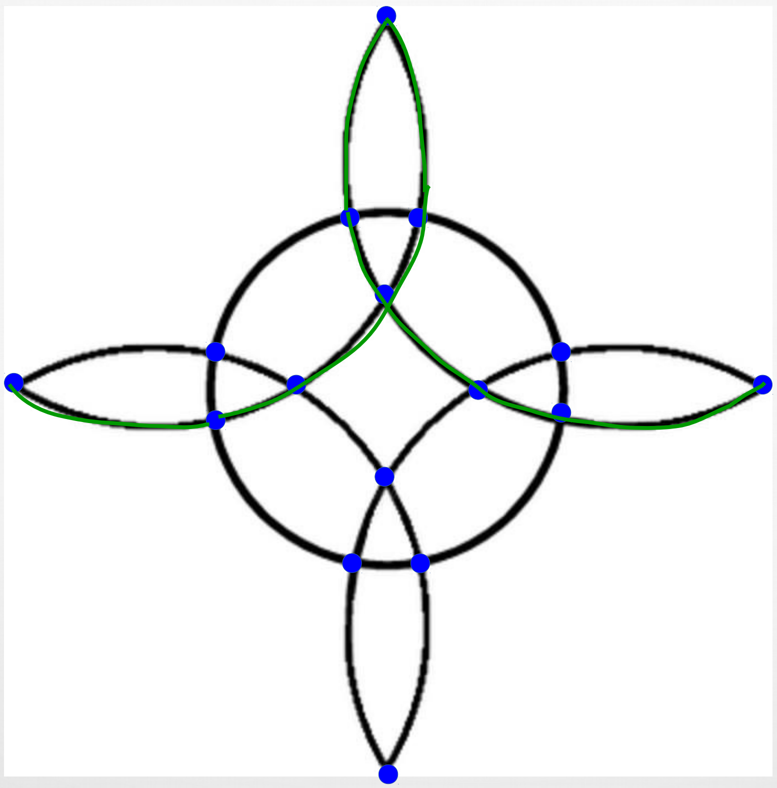
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

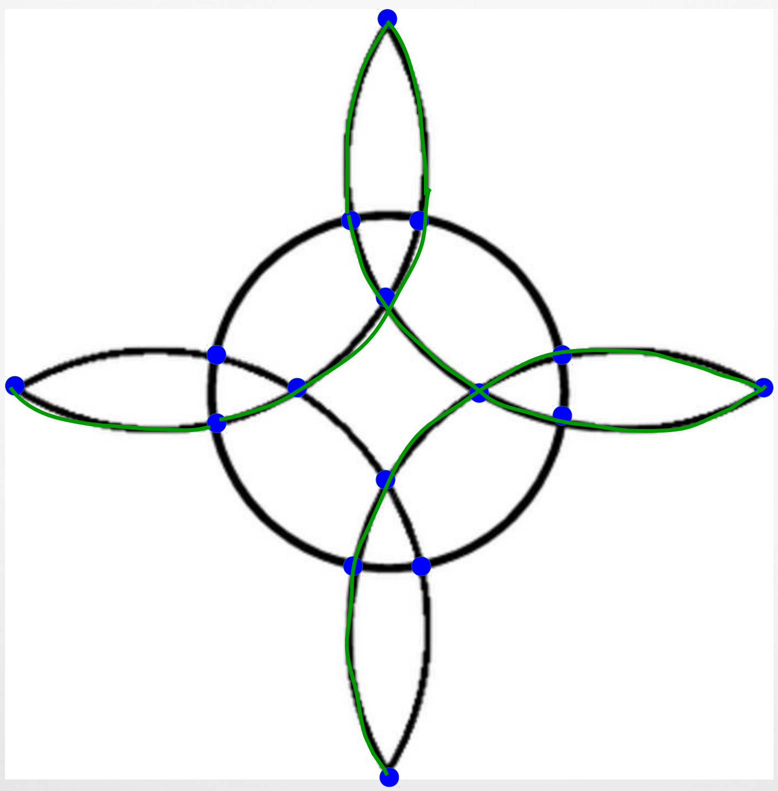
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

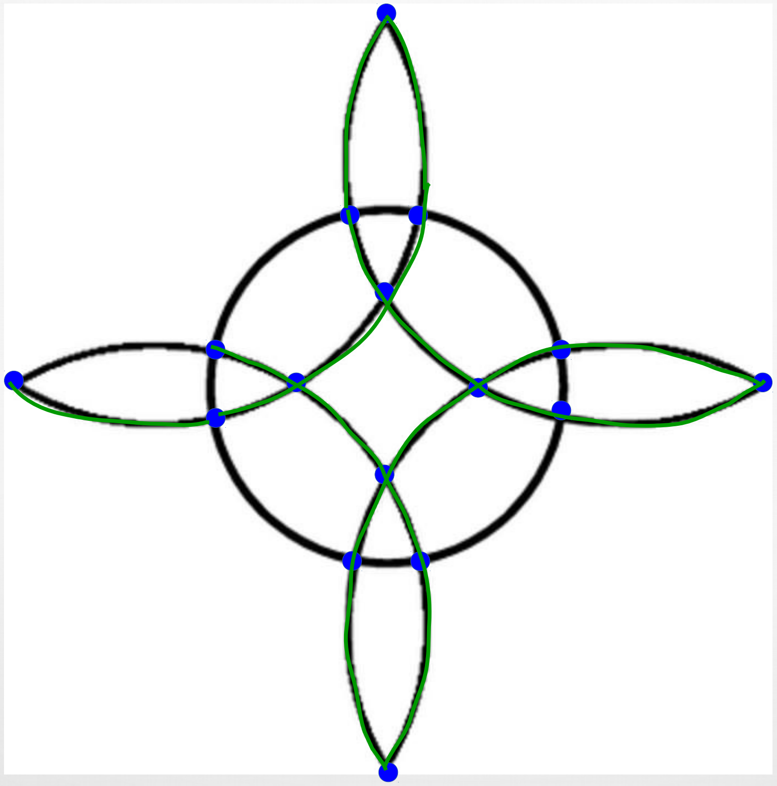
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

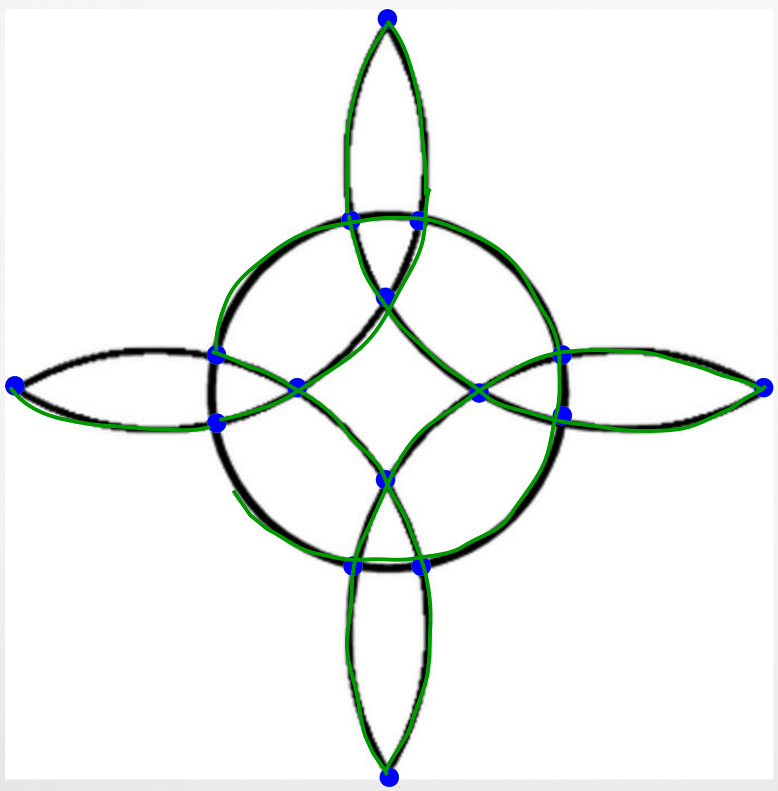
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

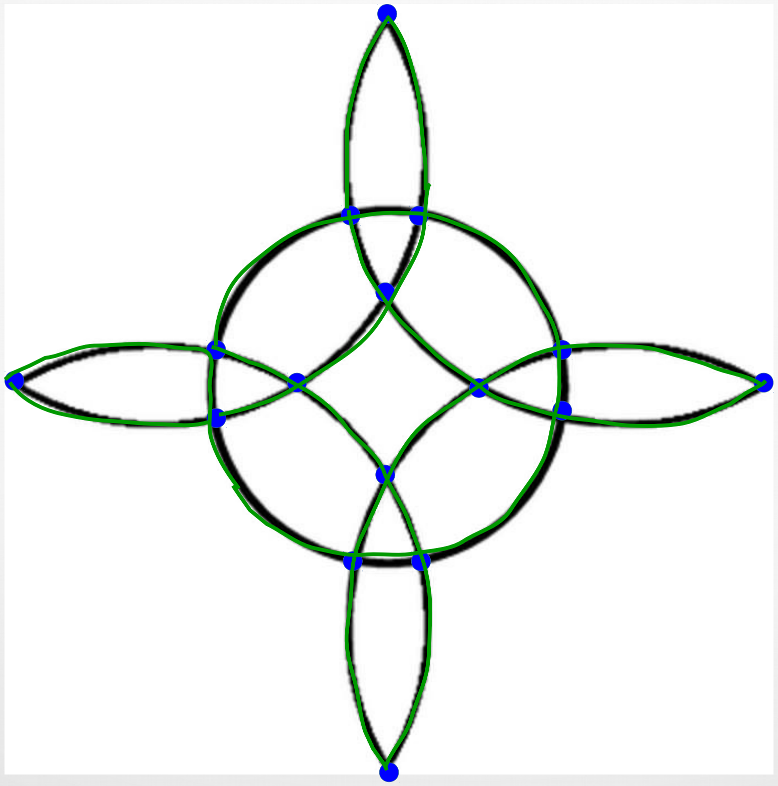
draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Euler circuits and paths

draw a picture in a continuous motion without lifting a pencil, so that no part of the picture is retraced.



10.5 *Euler and Hamilton Paths*

Applications of Euler Paths and Circuits

Chinese postman problem:
In honor of Guan Meigu, who posed it in 1962.

Suppose there is a mailman who needs to deliver mail to a certain neighbourhood. The mailman is unwilling to walk far, so he wants to find the shortest route through the neighborhood, that meets the following criteria:

- It is a closed circuit (it ends at the same point it starts).
- He needs to go through every street at least once.

Ideally: a street will be passed once! (that is Euler circuit)

10.5 *Euler and Hamilton Paths*

Applications of Euler Paths and Circuits:

- Garbage collection
- Molecular Biology: Euler paths are used in the DNA sequencing
- Electrical Engineering: layout of circuits
- See more on page 698

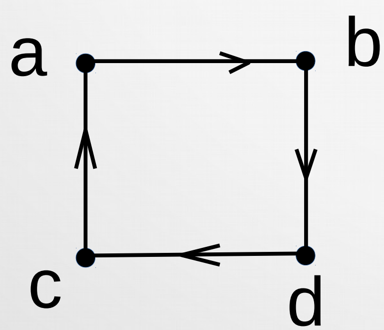
10.4 *Connectivity*

Strongly and weakly connected directed graphs

[Def] A directed graph is *strongly connected* if there is a path from a to b and from b to a whenever a and b are vertices in the graph.

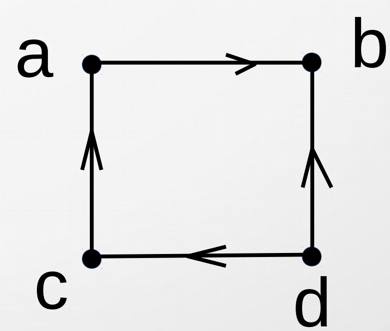
[Def] A directed graph is *weakly connected* if there is a path between any two vertices in the underlying *underlying undirected graph*.

H



strongly connected

G



weakly connected

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler circuit iff:

- All vertices with nonzero degree belong to a single strongly connected component.
- In-degree and out-degree of every vertex is same.

[Theorem] A directed graph has an Euler path iff

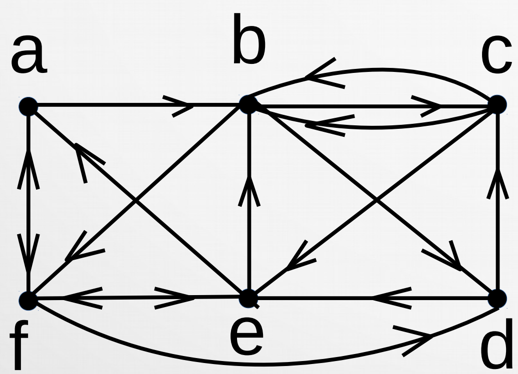
- at most one vertex has (out-degree) − (in-degree) = 1, at most one vertex has (in-degree) − (out-degree) = 1,
- every other vertex has equal in-degree and out-degree, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler circuit iff:

- All vertices with nonzero degree belong to a single strongly connected component.
- In-degree and out-degree of every vertex is same.

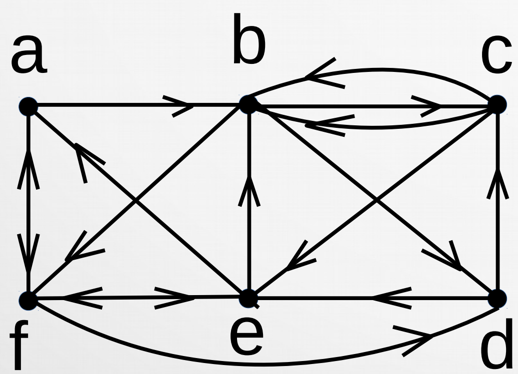


10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler circuit iff:

- All vertices with nonzero degree belong to a single strongly connected component.
- In-degree and out-degree of every vertex is same.



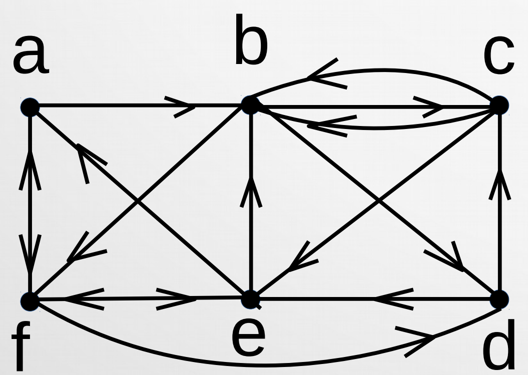
$deg^-(a) = 2$, $deg^+(a) = 2$
$deg^-(b) = 4$, $deg^+(b) = 3$
STOP!
No Euler circuit!

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.

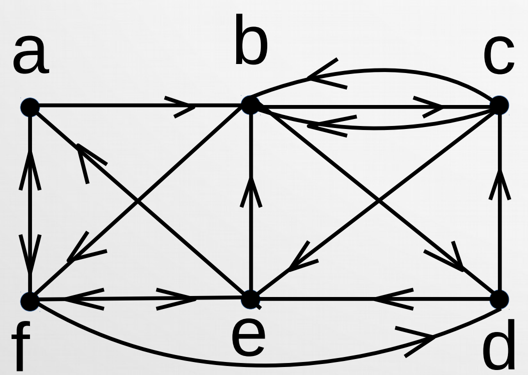


10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



graph has Euler path

$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

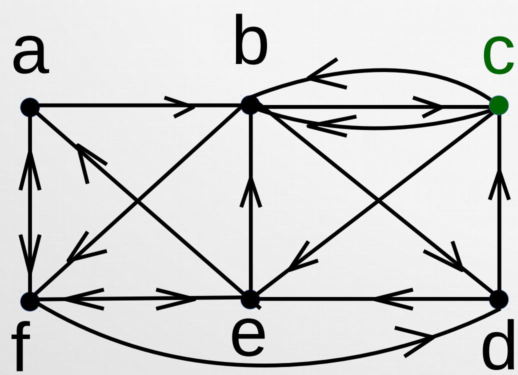
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

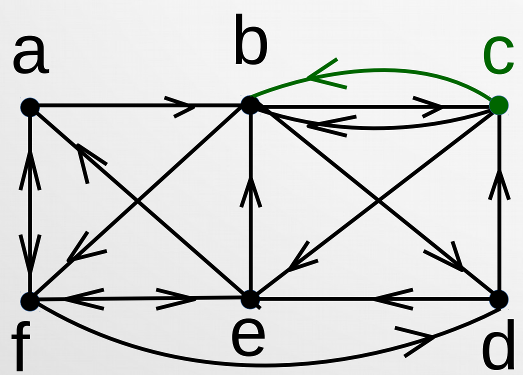
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



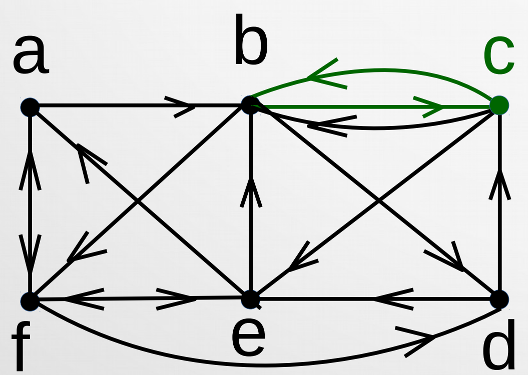
$deg^-(a) = 2, deg^+(a) = 2$
$deg^-(b) = 4, deg^+(b) = 3,$
$deg^-(b) - deg^+(b) = 1$
$deg^-(c) = 2, deg^+(c) = 3,$
$deg^+(c) - deg^-(c) = 1$
$deg^-(d) = 2, deg^+(d) = 2$
$deg^-(e) = 3, deg^+(e) = 3$
$deg^-(f) = 3, deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



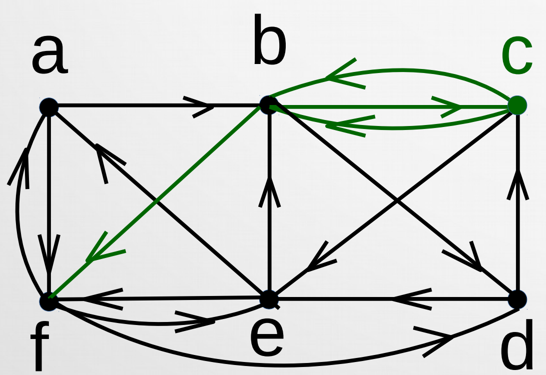
$deg^-(a) = 2$, $deg^+(a) = 2$
$deg^-(b) = 4$, $deg^+(b) = 3$,
$deg^-(b) - deg^+(b) = 1$
$deg^-(c) = 2$, $deg^+(c) = 3$,
$deg^+(c) - deg^-(c) = 1$
$deg^-(d) = 2$, $deg^+(d) = 2$
$deg^-(e) = 3$, $deg^+(e) = 3$
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2, deg^+(a) = 2$

$deg^-(b) = 4, deg^+(b) = 3,$

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2, deg^+(c) = 3,$

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2, deg^+(d) = 2$

$deg^-(e) = 3, deg^+(e) = 3$

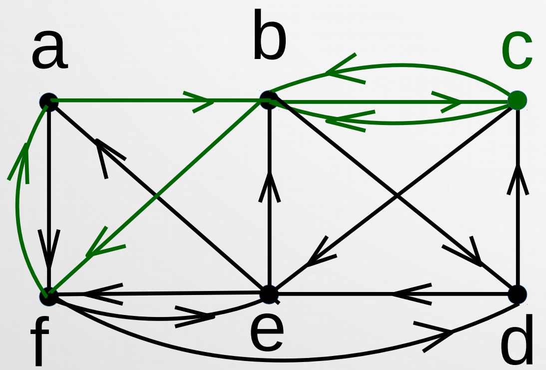
$deg^-(f) = 3, deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

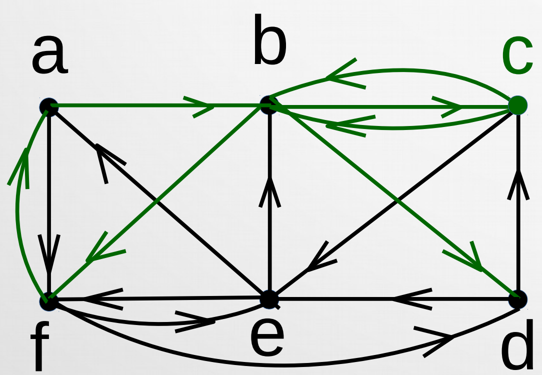
$deg^-(e) = 3$, $deg^+(e) = 3$

$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff
- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



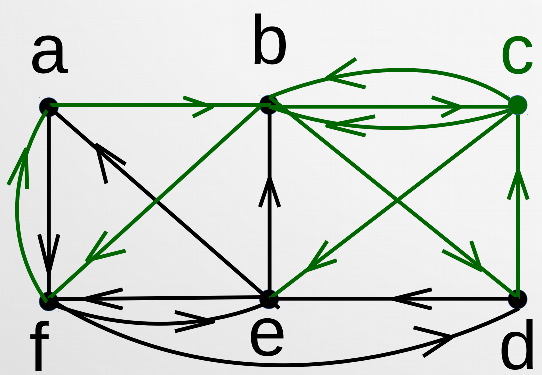
$deg^-(a) = 2, deg^+(a) = 2$
$deg^-(b) = 4, deg^+(b) = 3,$
$deg^-(b) - deg^+(b) = 1$
$deg^-(c) = 2, deg^+(c) = 3,$
$deg^+(c) - deg^-(c) = 1$
$deg^-(d) = 2, deg^+(d) = 2$
$deg^-(e) = 3, deg^+(e) = 3$
$deg^-(f) = 3, deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2, deg^+(a) = 2$

$deg^-(b) = 4, deg^+(b) = 3,$

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2, deg^+(c) = 3,$

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2, deg^+(d) = 2$

$deg^-(e) = 3, deg^+(e) = 3$

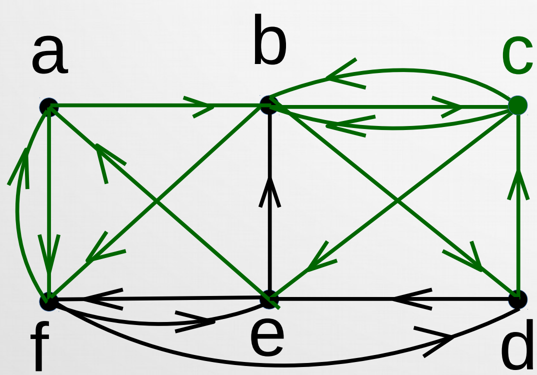
$deg^-(f) = 3, deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

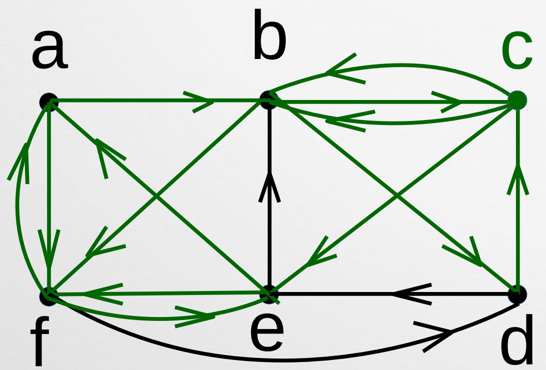
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

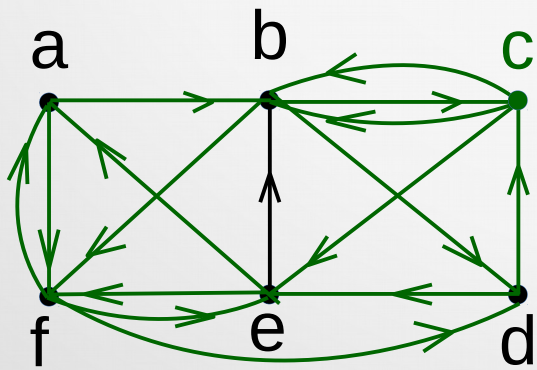
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

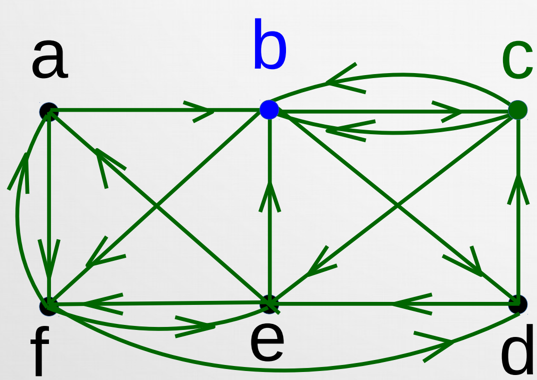
$deg^-(f) = 3$, $deg^+(f) = 3$

10.4 *Connectivity*

Euler circuit in directed graphs

[Theorem] A directed graph has an Euler path iff

- at most one vertex has (*out-degree*) − (*in-degree*) = 1, at most one vertex has (*in-degree*) − (*out-degree*) = 1,
- every other vertex has *in-degree* = *out-degree*, and
- all of its vertices with nonzero degree belong to a single connected component of the underlying undirected graph.



graph has Euler path

$deg^-(a) = 2$, $deg^+(a) = 2$

$deg^-(b) = 4$, $deg^+(b) = 3$,

$deg^-(b) - deg^+(b) = 1$

$deg^-(c) = 2$, $deg^+(c) = 3$,

$deg^+(c) - deg^-(c) = 1$

$deg^-(d) = 2$, $deg^+(d) = 2$

$deg^-(e) = 3$, $deg^+(e) = 3$

$deg^-(f) = 3$, $deg^+(f) = 3$