Chapter 10
Paths and Circuits

10.1 Eulerian Circuits

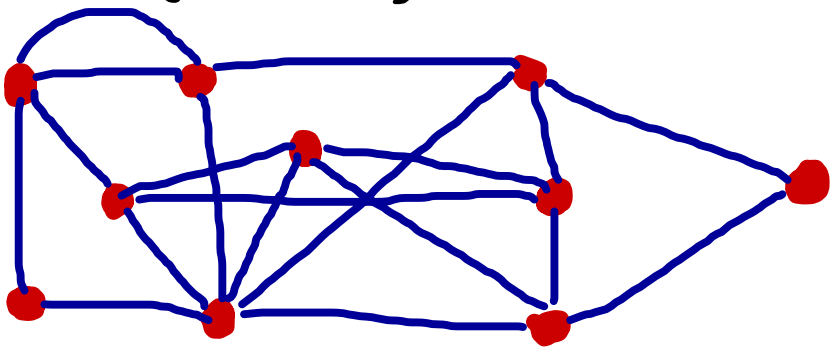
Eulerian Circuits

A walk is an alternating sequence of vertices and edges

$$v_1 e_1 v_2 e_2 \cdots v_n e_n v_1$$

where $e_i = v_i v_{i+1}$ and the e_i are distinct.

An Eulerian circuit in a pseudograph is a walk that crosses each edge exactly once and ends where it started.



Find one!

If a pseudograph G admits an Eulerian circuit, we say that G is Eulerian.

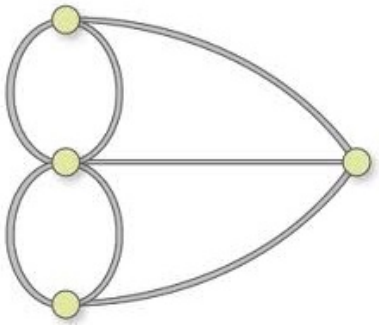
Euler is pronounced Oiler.

The Königsberg Bridge Problem

The Bridges of Konigsberg

Is it possible to take a walk, cross each bridge exactly once, and return to where you started?

Or: Is the following pseudograph Eulerian?



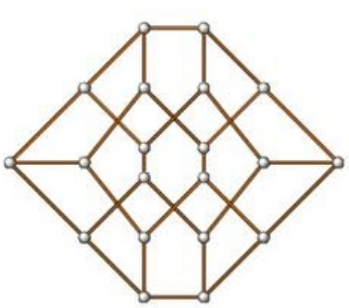
Answer: No! Each vertex needs to have even degree.

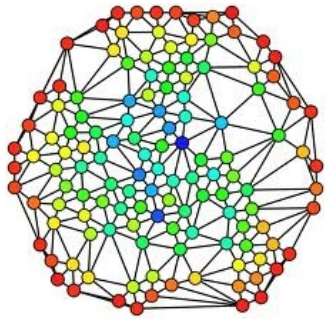
Connectivity

We just argued that Eulerian graphs have no vertices of odd degree.

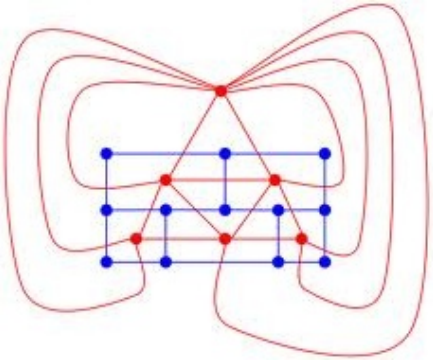
What else? Eulerian graphs must also be connected.

A pseudograph is connected if there is a walk between any two vertices.





connected



not connected

Eulerian Pseudographs

Theorem. A pseudograph is Eulerian if and only if it is connected and every vertex has even degree.

Proof. We already know Eulerian pseudographs are connected with no odd vertices.

Say G is connected and has no odd vertices.
Start at any vertex, walk across an adjacent edge.
Because every vertex is even, can keep walking. When we run out of options for continuing, we must be back at starting point.

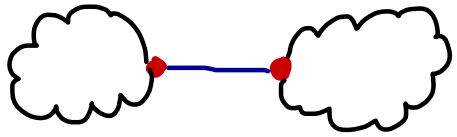
If we haven't used all edges, then repeat the process.

At the end, since G is connected, we can amalgamate these walks into one walk.

Eulerian Pseudographs

Theorem. A pseudograph is Eulerian if and only if it is connected and every vertex has even degree.

Second Proof (Fleury's Algorithm). Pick a starting vertex.
While the pseudograph has at least one edge:
traverse any edge that is not a bridge



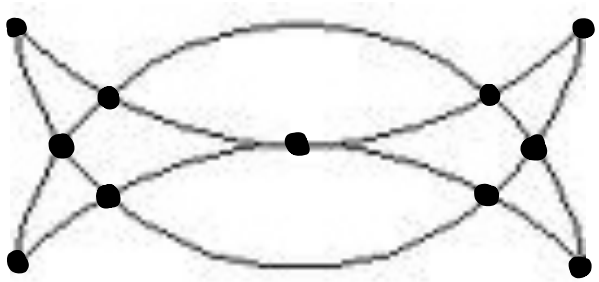
and delete that edge.

Prove that this always results in an Eulerian circuit.

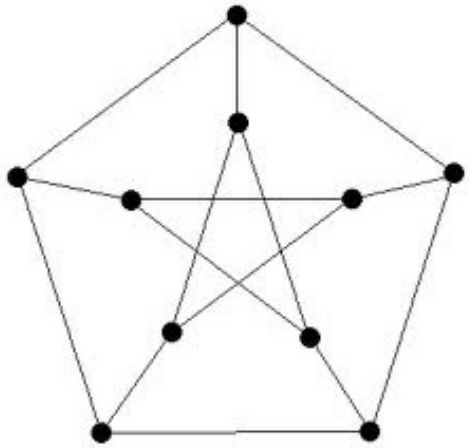
Eulerian Pseudographs

For each pseudograph, find an Eulerian circuit if it exists.

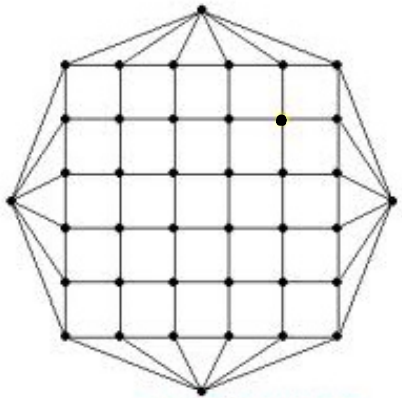
(i)



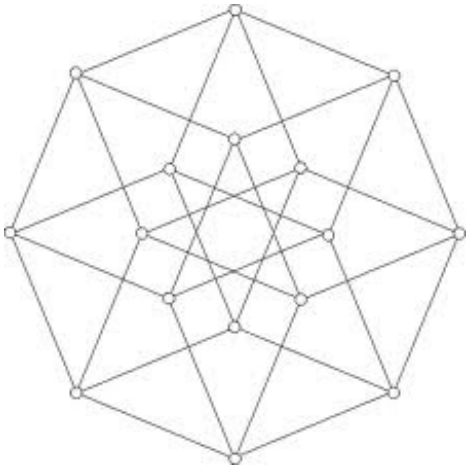
(ii)



(iii)



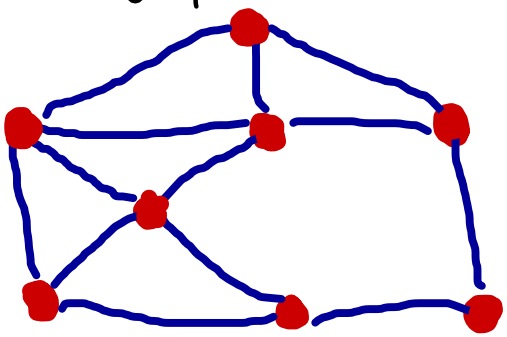
(iv)



10.2 Hamiltonian Cycles

Hamiltonian Cycles

A Hamiltonian cycle in a pseudograph is a walk that visits each vertex exactly once:



Find one!

If a pseudograph has a Hamiltonian cycle, we say the pseudograph is Hamiltonian.

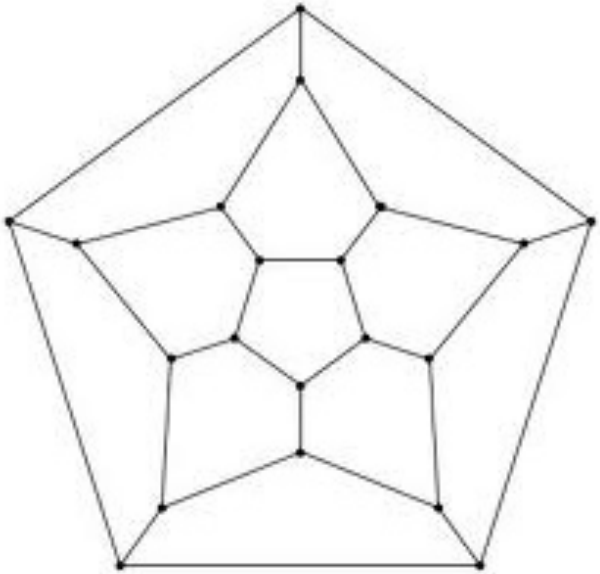
Euler: each edge once
Hamilton: each vertex once

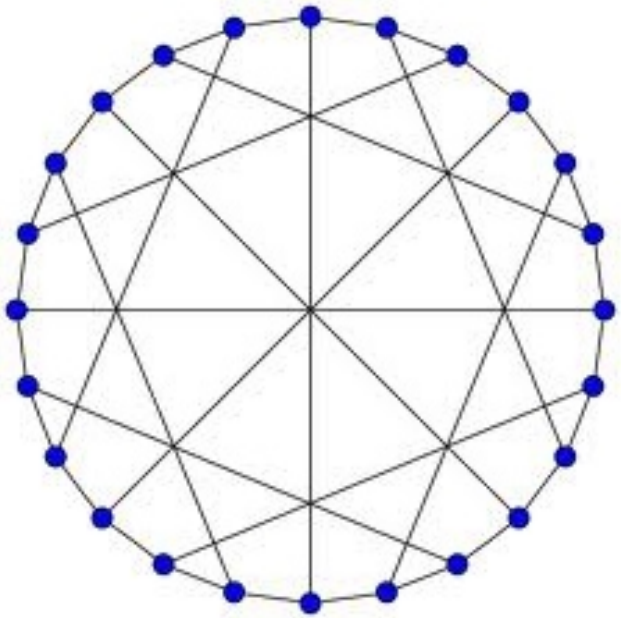
Sir William Rowan Hamilton

Note: A Hamiltonian cycle is isomorphic to an n-cycle.

Hamiltonian Cycles

Show that the following graphs are Hamiltonian.





In other words, find a Hamiltonian cycle in each.

Hamiltonian Graphs

We saw that it is easy to tell if a graph is Eulerian or not.
To prove a graph is Hamiltonian, just find a Hamiltonian cycle.
But there is no easy method for showing a graph is not Hamiltonian.

You could check all paths of length $|V|$. Takes too long!

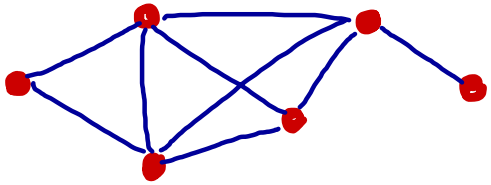
Better to use some basic facts:

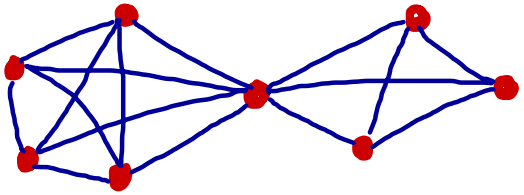
Let H be a Hamiltonian cycle in a pseudograph G

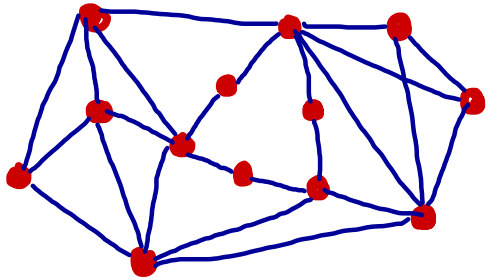
1. Every vertex of G has exactly two edges of H passing through it.
2. The only cycle contained in H is H.

Hamiltonian Graphs

Prove that the following graphs are not Hamiltonian.

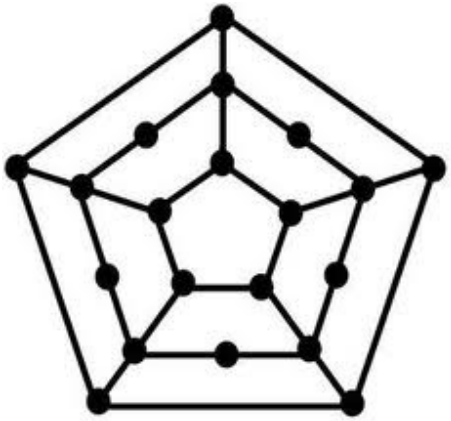


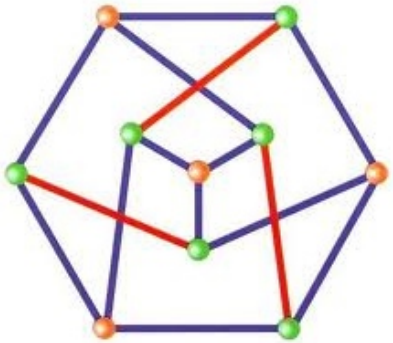


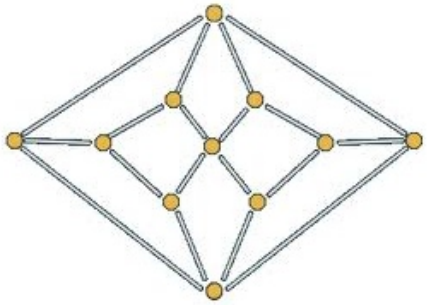


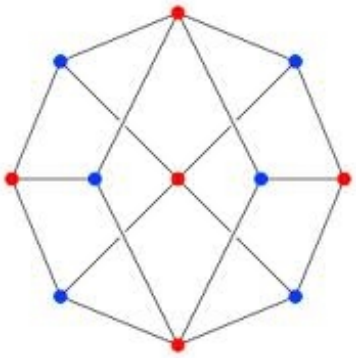
Hamiltonian Graphs

Which of the following graphs are Hamiltonian?

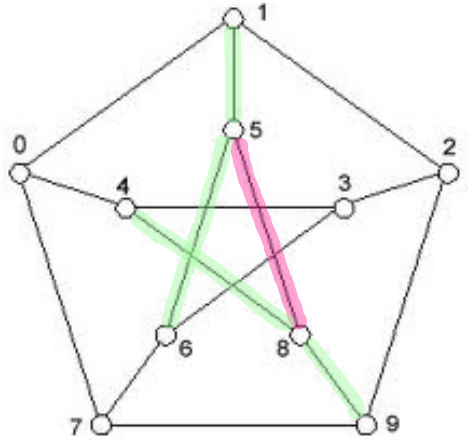








The Petersen Graph



Proposition. The Petersen graph is not Hamiltonian.

Proof. Say H is a Hamiltonian cycle. H must contain an edge connecting the inner and outer pentagons, say 15. From 5, need either 56 or 58, say 56. But then 58 is not in H, so 89 and 84 are in H. See ↖

Now there are two cases for the other edge of H at 4:

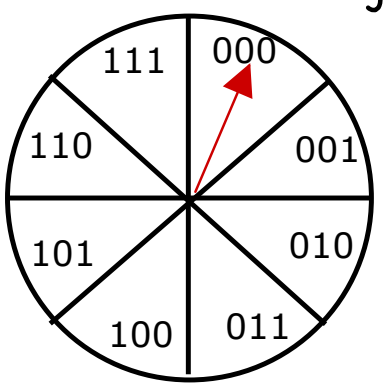
Case 1. 40 in H.

In this case, 43 is not in H $\rightsquigarrow$ 32 & 36 in H
$\rightsquigarrow$ 76 not in H $\rightsquigarrow$ 70 & 79 in H
$\rightsquigarrow$ The cycle 07984 in H, contradicting Fact 2.

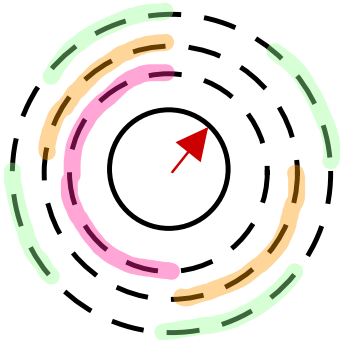
Case 2. 43 in H. Similar. ▨

Gray Codes

We can record the position of a rotating pointer with a bit string:



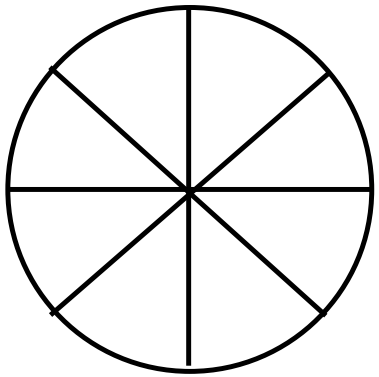
Can read the position of the arrow with 3 sets of contacts:



Problem: A small error could give 100 instead of 011
⟶ all 3 bits wrong!

Gray Codes

To fix this, want to number so that adjacent regions differ by one bit.



At first, not obvious how to do this.
But: such a numbering is just a Hamiltonian cycle in the n-cube.