Connectedness, Euler Paths and Hamiltonian Circuits

Paths and Cycles

A path in a graph is a sequence of nodes and arcs $n_1, a_1, n_2, a_2, \ldots, a_{k-1}, n_k$ such that each a_i joins n_i to n_{i+1}.

We describe paths by sequences of nodes, omitting the arc names.

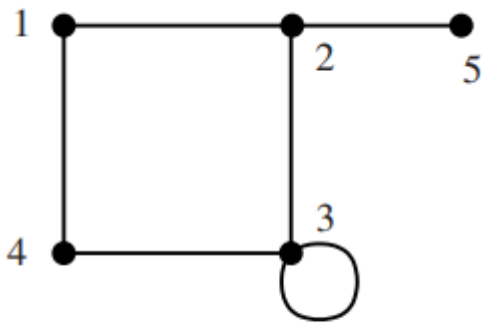

Figure 1.9: A graph

Possible paths include:

- 1,2,3,4
- 2,5,2,3,3,4

In the latter case, the arc joining 2 to 5 is used twice (in opposite directions).

The length of a path is the number of arcs in the path. If an arc is used twice it is counted twice.

Thus the length of 2,5,2,3,3,4 is 5. We allow zero-length paths.

A path is simple if it has no repeated nodes.

A cycle is a path which finishes where it started, i.e. $n_1 = n_k$, which has at least one arc, and which does not use the same arc twice.

Cycles are sometimes called circuits.

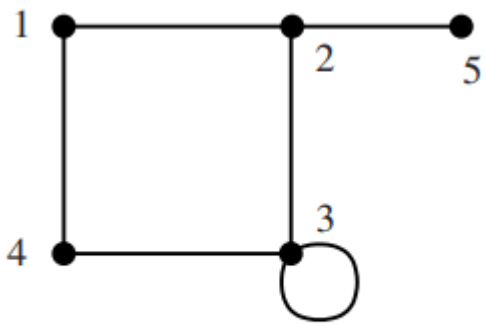

Figure 1.9: A graph

Possible cycles include:

- 1,2,3,4,1
- 3,3

If a graph has no cycles it is acyclic.

A graph is connected if for any two different nodes n, n' there is a path from n to n' .

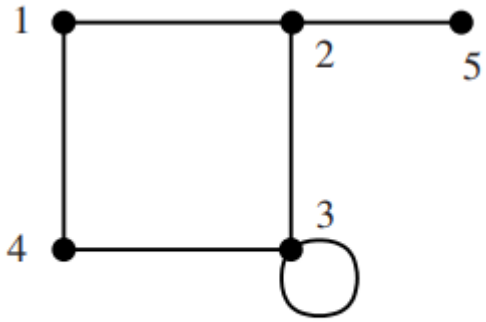

Figure 1.9: A graph

This graph is connected.

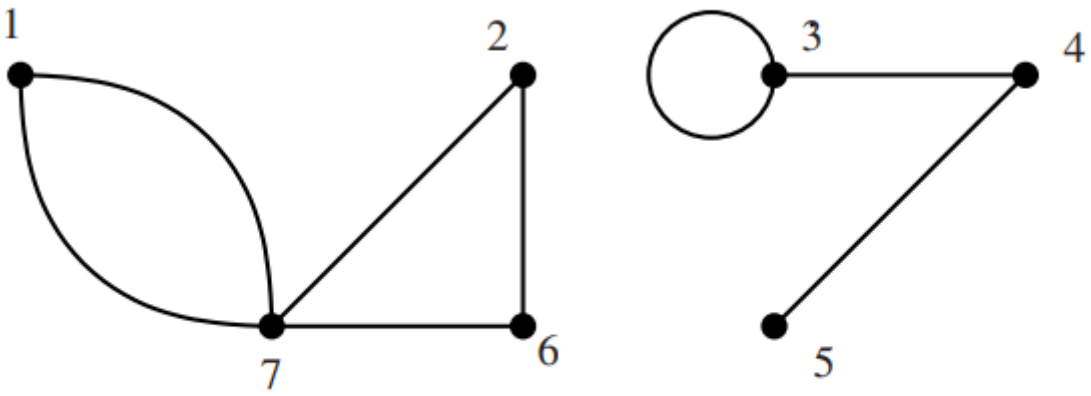

Figure 1.10: A disconnected graph

This graph is not, since there is no path from 2 to 3 (for instance).

Given a graph G, we can define a relation on $nodes(G)$ by: $n \sim n'$ if and only if there is a path from n to n'.

This is an equivalence relation (reflexive, symmetric and transitive).

The equivalence classes of $\sim$ are called the (connected) components of G. Each component is a connected graph, but there is no path from a node in one component to a node in another component.

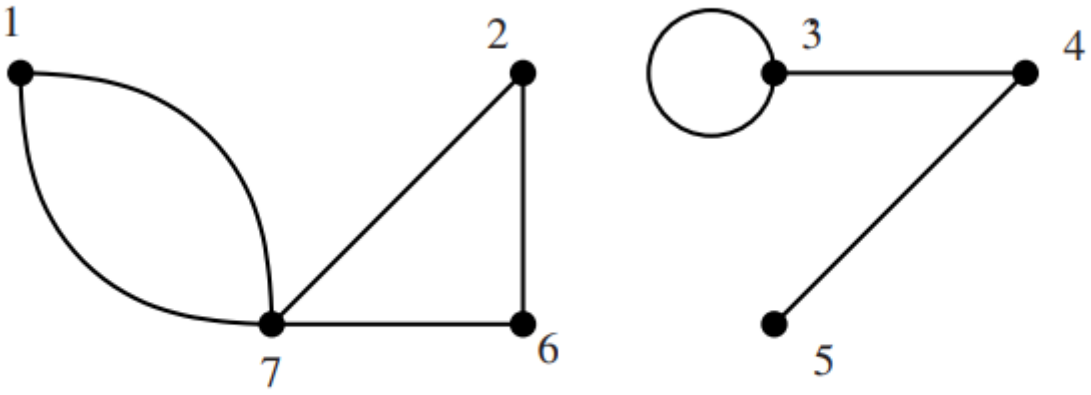

Figure 1.10: A disconnected graph

There are two connected components, one with nodes 1,2,6,7 and one with nodes 3,4,5.

Euler Paths and Circuits

An Euler path is a path which uses every arc in the graph exactly once.

An Euler circuit (Euler cycle) is a cycle which uses every arc in the graph exactly once.

Suppose that a graph G has an Euler path P starting at node n_0 and ending at n_k. If P passes through a node n, it must enter n as many times as it leaves it. Hence the number of arcs incident on n which are used by P must be **even**.

Every node of G must have an even degree except for n_0 and n_k

If n_0 and n_k are different then they must each have an odd degree, and if they are the same (P is a cycle) then $n_0 = n_k$ must have an even degree.

Therefore, a connected graph has an Euler path if and only if the number of odd nodes is either 0 or 2. A connected graph has an Euler circuit if and only if there are 0 odd degree nodes.

This gives us a very simple algorithm to check whether a connected graph has an Euler path:

- Count the number of odd nodes

This can be done in $O(n^2)$ time, where n is the number of nodes:

- Make a single pass through the adjacency matrix row by row
- Maintain a counter for the number of odd nodes
- Maintain a counter for the degree of the node corresponding to the current row

An algorithm for finding an Euler path where 2 nodes (n, n') have odd degree

- Start at node n
- Follow any path until you can go no further. You must have reached n'
  - You can't have finished at n, since you would have used up evenly many arcs incident on n, and there would be a spare arc.
  - You can't have stopped at n'' different from n, n' , since you would have used up an odd number of arcs incident on n'', and n'' has even degree.

- Notice that now every node has an even number of unused arcs
- Call the path so far P
- If you have an Euler path then stop
- Otherwise take some node n'' on your path so far which has unused arcs, and start a new path there
- This must return to its start n'' to form a cycle C
    - If it stopped at any other node there would be an unused arc
- Now merge P and C to form a longer path (see Figure 1.12)
- Keep on doing this until all arcs are exhausted
- This gives the desired Euler path

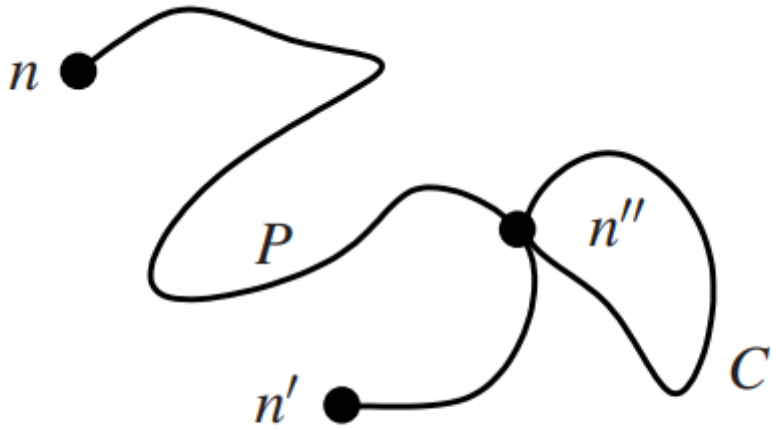

Figure 1.12: Finding an Euler path

Hamiltonian Circuits

A Hamiltonian path is a path which visits every node in a graph exactly once. A Hamiltonian circuit is a cycle which visits every node exactly once.

The Hamiltonian circuit problem (HCP) is: given a graph G, determine whether G has a Hamiltonian circuit. The Hamiltonian path problem is defined similarly.

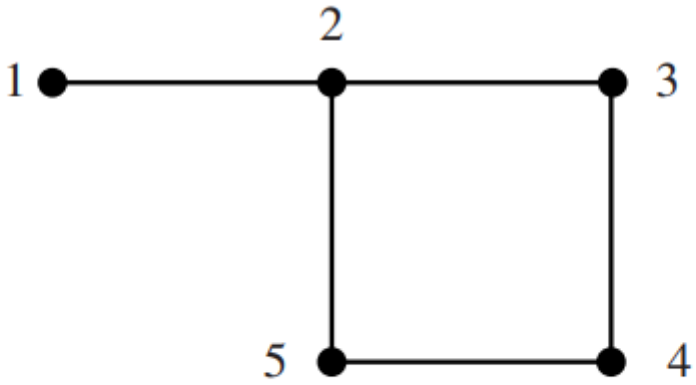

Figure 1.13: A graph with a Hamiltonian path

This graph has a Hamiltonian path, but no Hamiltonian circuit (once node 1 is reached, which must be from 2, we cannot leave it without repeating node 2).

Finding whether a graph has a Hamiltonian circuit

Firstly, we only need to consider simple graphs, since removing loops and parallel arcs makes no difference to whether a circuit exists.

2 necessary conditions:

1. The graph must be connected (also needed for a Hamiltonian path)
2. Each node must have degree at least 2, since we must approach it and leave it via two different nodes

These are not sufficient conditions.

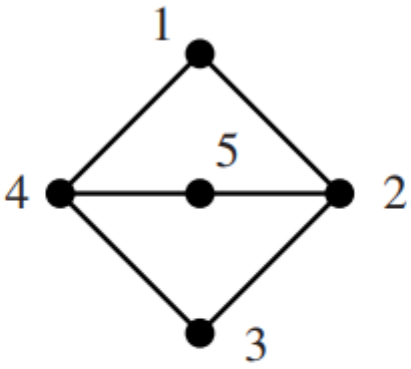

Figure 1.14: A graph with no Hamiltonian circuit

This graph does not have a Hamiltonian circuit.

Since graphs are finite, we can check each possible circuit in turn and see whether it is Hamiltonian. Suppose that G has n nodes labelled $1, \ldots, n$. Possible Hamiltonian circuits involve all n nodes exactly once. So they can be described by a permutation π of $1, \ldots, n$. There are $n!$ different permutations of $1, \ldots, n$. We generate and check each possible circuit π to see whether it is an actual circuit in G by asking (for each $i = 1, \ldots, n$) is $\pi(i)$ adjacent to $\pi(i + 1)$?

The checking of each π can be done quickly ($O(n)$). However, the number of permutations to be checked is $n!$ (bad).

A better algorithm is known (it can solve HCP in $O(n^2 2^n)$). This is still bad.

A faster algorithm has not been found.