F11ME3 Algebra 3: graph theory
Outline lecture notes

Mark V Lawson
Heriot-Watt University, Edinburgh

8.IV.08

1 Introduction to graphs

It's often thought that applied mathematics has to involve calculus. This is quite wrong. In the past few decades, applications of mathematics have increasingly come to rely upon mathematics which does not necessarily involve calculus; this mathematics is often called discrete mathematics. One of the most important topics in discrete mathematics is graph theory. Graphs are special kinds of pictures that enable us to represent problems, and often to solve them. Graphs have both practical and theoretical applications.

The main goal of this course is to introduce you to the basic ideas of graph theory. In addition, I want also to introduce you to some proof methods, particularly induction, which you will find useful in your future study of mathematics.

These outline lecture notes contain all the definitions, statements of the main results, and most of the proofs. They do not contain the many examples and motivating material that will be discussed in the lectures.

1.1 First examples

A *graph* is a diagram consisting of *vertices* joined by *edges*. An edge must either join a vertex to itself or join two different vertices together.

This is the only meaning of the word 'graph' in this course. In the lectures, I shall give you some examples of graphs.

Each graph can be drawn in many different ways, but they are still the same graph. However, we usually draw graphs so that the information they contain is easy to see.

The vertices of a graph are often used to represent things and the edges to represent relationships between these things.

Graphs can be used to represent information and they can be used to help us solve problems. The important point to remember is that **a graph is known once we know its vertices and edges**. The following exercise illustrates this point.

Exercise 1.1 Consider the following 6 intervals of the real line

$$a = (0, 3),\ b = (2, 7),\ c = (-1, 1),\ d = (2, 3),\ e = (1, 4),\ f = (6, 8).$$

Draw a graph where the vertices are labelled a, b, c, d, e, f, and an edge joins vertex x to vertex y if and only if $x \neq y$ and the intervals x and y have non-empty intersection. ■

Example 1.2 The London Underground map. The London Underground is a convenient way of getting around the city. However, navigating your way through the Underground is potentially daunting: it is huge, and the central area where most of the tourists sights are is complicated. These problems were solved by Harry Beck in 1933 when he redesigned the Underground map. Rather than representing the railway lines as they really are he distorted them so that the whole map was easier to read. There are two reasons why such distortions don't matter: first, travelling underground means that there are no sights out of the window; second, a traveller is interested primarily in connections — that there is a line from station A to station B rather than the shape of that line. On the other hand, if you go walking in the countryside you need a 'real' map: you are not merely interested in the fact that there is a path up Ben Nevis but also in the shape of that path and what sights you can see from it. ■

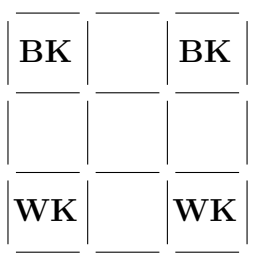
Example 1.3 'Six degrees of separation'. This example stems from the work of the American psychologist Stanley Milgram in the 1960's. It has recently become topical again and has connections with the surprising effectiveness of the internet.[1]

[1]The following book is worth reading for more information: Mark Buchanan, *Small world*, Phoenix, 2002.

Imagine all the people in the world now living. Some of these people you are directly acquainted with. Let's draw lines linking you to each of your acquaintances. Similarly, each of your acquaintances is directly acquainted with a number of people. Draw lines linking them to each of their acquaintances. Continue in this way until everyone in the world has been accounted for. We will obtain an enormous picture — an 'acquaintanceship graph'. If we pick two people, they might be directly acquainted: I'll say that they are *linked*. On the other hand, although they might not be directly acquainted with each other, they might each have a direct acquaintance in common: I'll say that they are separated by 2 links. And so on. Thus A is separated from B by 4 links if there are three other people X_1, X_2 and X_3 such that A is directly acquainted with X_1, X_1 is directly acquainted with X_2, X_2 is directly acquainted with X_3, X_3 is directly acquainted with B.

Milgram carried out some experiments that suggested that any two people in the world are only a few links apart: on average something like 6. This passed into everyday language as the phrase 'six degrees of separation'.[2] ■

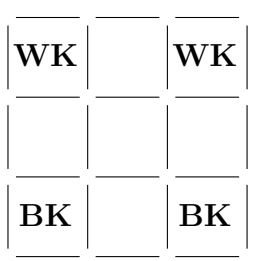
Example 1.4 The exchanging knights problem. On the surface, this problem appears to have nothing to do with graphs. The drawing below shows a 3×3 'chess-board'



on which are placed two black knights (BK) and two white knights (WK). The problem is to make the black knights and the white knights exchange places in the smallest number of moves using only allowable chess moves. So

[2] *Why* this should be so is an interesting question: read Buchanan's book for more information.

at the end the chess-board should look like this:



I shall ask you to think about this problem in the lecture and I shall show you how it may be solved easily using graphs.[3] ■

Example 1.5 Unrealistic Renaissance rabbits. An immature pair of rabbits is placed in a walled garden. How many rabbits will there be at the end of 6 months, if an immature pair of rabbits takes a month to mature, and if at the end of every month a mature pair of rabbits produces an immature pair? ■

1.2 Graph terms

Graph theory has a special terminology which enables us to talk about graphs unambiguously. In the lecture, I shall work through the definitions below giving you examples.

A *graph* is a digram consisting of points, called *vertices*, joined together by lines, called *edges*; each edge joins exactly two vertices or a vertex to itself.

- The terminology of graph theory is not completely standardised, so when using a book, check how the author uses technical words.[4]
- The word 'graph' is used with another, and quite different meaning in mathematics: we talk about 'graphs of functions'.
- Note that the point where edges cross is not a vertex.

[3] I learnt about this problem from David Wells book 'You are a mathematician' (Penguin, 1995), which I recommend.

[4] A good book is *Graphs: an introductory approach*, by R. J. Wilson and J. J. Watkins published by John Wiley, 1990.

- It is important to remember that each graph can be drawn in many ways; think of the edges as being elastic so that they can be stretched and deformed as much as you like. We usually try and draw a graph in the simplest possible way.

- A computer-friendly representation of graphs, the adjacency matrix, will be described later.

Two or more edges joining the same pair of vertices are called *multiple edges*. An edge joining a vertex to itself is called a *loop*. A graph with no loops or multiple edges is called a *simple graph*. A pair of vertices joined by an edge are said to be *adjacent*. A vertex is said to be *incident* to an edge if it forms one end of the edge. The *degree* of a vertex v, denoted $\deg v$, is the number of edges meeting at v. The following should be noted:

> **a loop at v contributes 2 to the degree of a vertex since a loop has two ends which meet at v.**

Warning! Make sure you get the definition of vertex degree correct.

A single vertex with no edges out of it is called an *isolated vertex*. It is often convenient to *list* the degrees of the vertices in a graph, and this is usually done by writing them in 'non-decreasing' order; this peculiar terminology means 'in increasing order, but allowing repeats where necessary'. Such a list is called a *degree-sequence*. We say that a graph is *regular* if all vertices have the same degree. In particular, if the degree of each vertex is r then we say the graph is *regular of degree* r.

If G is a graph then a *subgraph* of G is a graph G' which is obtained from G by erasing some of the edges and some of the vertices (where 'some' can mean anything between 'none' and 'all').

A *path* in a graph is a succession of edges with the property that the second vertex of each edge is the first vertex of the next one. The *length* of a path is just the number of edges involved in the path (counting repeats). A path is said to be *closed* if it begins and ends at the same vertex. A *cycle* in a graph is a closed path with distinct vertices.

A graph which is all in one piece is said to be *connected*, whereas one which splits into several pieces is said to be *disconnected* — the connected pieces which make up a graph are called its *connected components*. Observe

that a graph is connected precisely when any two vertices can be joined by a path.

Certain kinds of graphs occur so frequently that they are given special names. I shall explain the meaning of the following terms in the lecture:

Null graphs N_n.

Complete graphs K_n.

Cycle graphs C_n.

Complete bipartite graphs $K_{m,n}$.

I conclude this section with our first result. It turns out to be surprisingly useful.

Theorem 1.6 (The handshaking lemma) *The sum of the vertex degrees of a graph is equal to twice the number of edges.*

Proof When we sum up the vertex degrees, each edge is counted twice since each edge has exactly two ends. ∎

1.3 Labelled versus unlabelled graphs

When I draw graphs, sometimes I leave the vertices as empty circles, and sometimes the vertices contain information such as a number or a letter. The former are called *unlabelled graphs* and the latter *labelled graphs*. A labelled graph can be converted into a labelled graph by erasing the labels. In most problems we are interested in labelled graphs. For example, a map of the London Underground would not be much use without the station names. Two labelled graphs are the same if we can deform one into the other (by moving vertices and stretching edges) in such a way that the labels match. For unlabelled graphs we don't have any labels to worry about. We say that two labelled graphs *have the same shape* or are *isomorphic* if the unlabelled graphs we get when we erase the vertex labels are the same. In most of this course we shall be interested in labelled graphs. But when you are asked to list graphs of a certain type then it is the *shapes* of the graphs which are important and so unlabelled graphs are required, because there are fewer graphs to worry about.

Exercise 1.7 Let $X = \{a, b, c, d, e\}$. Draw the following graph: the vertices are labelled by the two element subsets of X. If A and B are both two element subsets of X then we draw an edge between them if $A \cap B = \emptyset$, the empty set.

In addition, write down the degree sequence of the resulting graph. ■

Exercise 1.8 Draw all simple unlabelled graphs with 3 vertices. ■

Exercise 1.9 Draw a simple graph having 6 vertices, 6 edges, and degree sequence $(1, 1, 2, 2, 3, 3)$ ■

Exercise 1.10 Draw two different graphs with 6 vertices, 6 edges and regular of degree 2. ■

2 The adjacency matrix

In this section, we shall show how to construct a matrix from a graph and use it to obtain information about the graph.

2.1 Definition

To understand this section, you will need to know how to calculate positive powers of a square matrix.

Matrices are defined as rectangular arrays of numbers. If a matrix has m rows and n columns, we say that it is an $m \times n$ matrix. Matrices are usually denoted by Roman capitals. To pick out the element of the matrix in the ith row and jth column, we write $(A)_{ij}$.

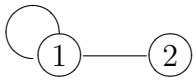
Let G be a graph with vertices $1, \ldots, n$. Then the *adjacency matrix*, $A(G)$, of G, is the $n \times n$ matrix such that $(A(G))_{ij}$ is the number of edges joining i to j.

> **Each loop at the vertex i counts one towards the value of $(A(G))_{i,i}$.**

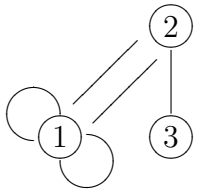
The adjacency matrix of the graph and the graph itself contain the same information as far as graph theory is concerned.

Examples 2.1 Write down the adjacency matrices of each of the following graphs.

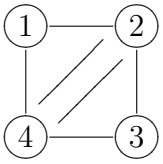
(i)



(ii)



(iii)



■

Examples 2.2 Draw the graphs described by each of the following adjacency matrices.

(i) $\begin{pmatrix} 1 & 2 \\ 2 & 2 \end{pmatrix}$

(ii) $\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

(iii) $\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$

■

Exercise 2.3 Write down the adjacency matrix of $K_{2,2}$. ■

Exercise 2.4 Write down the adjacency matrix of K_4. ■

We shall use adjacency matrices later, but we first I shall need to describe an important technique for proving certain kinds of results.

2.2 Proof by induction

This is a proof technique with applications throughout mathematics. The basis of this technique is the following idea:

> "I am thinking of a subset X of the infinite set $\{1, 2, 3, \ldots\}$. I tell you two things about X: first, $1 \in X$, and second if $n \in X$ then $n + 1 \in X$. What is X?"

The fact that these two pieces of information are enough to determine the set of positive integers is called the *principle of induction*. This principle can be used to prove results as follows.

Suppose we have an infinite number of statements $S_1, S_2, S_3, \ldots$ which we want to prove. By the principle of induction it is enough to do two things:

1. Show that S_1 is true.

2. Show that if S_n is true then S_{n+1} is also true.

It will follow that S_i is true for all positive i. This proof technique can only be learnt by attempting lots of examples.

Example 2.5 Prove by induction that

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}.$$

■

A proof by induction takes the following form:

Base step Show that the case $k = 1$ holds.

Induction hypothesis (IH) Assume that the case $k = n$ holds.

Proof bit Now use (IH) to show that the case $k = n + 1$ holds.

Exercise 2.6 Prove by induction that

$$1 + 3 + 5 + \ldots + (2n - 1) = n^2$$

for each $n \geq 1$. ■

Exercise 2.7 Prove by induction that $5^n - 1$ is exactly divisible by 4 for all natural numbers $n \geq 1$. ■

What I have described above I shall call 'basic' induction. There are numerous variations on basic induction. I shall describe two here:

1. Rather than starting the base step at $k = 1$ we might start at $k = 2$ or $k = 3$ and so on.

2. In basic induction we assume S_n and prove S_{n+1}. Sometimes we need to assume some or all of $S_1, \ldots, S_n$ to be true in order to prove S_{n+1} and in addition our base case may consist of several cases. This is often called 'strong induction'.

Example 2.8 Prove for all natural numbers $n \geq 4$ that $n^3 < 3^n$. ■

Example 2.9 A *prime number* is a natural number greater than or equal to 2 which is only exactly divisible by 1 and itself. **The number 1 is not a prime.** Prove that every natural number greater than or equal to 2 can be written as a product of primes. ■

Example 2.10 Let F_n denote the nth Fibonacci number. Thus $F_1 = 1$, $F_2 = 1$, $F_3 = 2$ etc. Prove that

$$F_n = \frac{1}{2^n\sqrt{5}}[(1+\sqrt{5})^n - (1-\sqrt{5})^n].$$

The base case here is the check that our formula gives 1 when $n = 1$ and 1 when $n = 2$.

Our induction hypothesis is that the formula holds for $n = k-2$ and $n = k-1$ where $k \geq 3$. We prove that the formula holds for $n = k$. Thus we have to calculate

$$\frac{1}{2^{k-1}\sqrt{5}}[(1+\sqrt{5})^{k-1} - (1-\sqrt{5})^{k-1}] + \frac{1}{2^{k-2}\sqrt{5}}[(1+\sqrt{5})^{k-2} - (1-\sqrt{5})^{k-2}]$$

This is an exercise in highschool algebra. Observe that

$$3+\sqrt{5} = \frac{1}{2}(1+\sqrt{5})^2 \text{ and } 3-\sqrt{5} = \frac{1}{2}(1-\sqrt{5})^2.$$

Our sum becomes

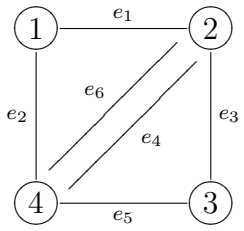
$$\frac{1}{2^k\sqrt{5}}[(1+\sqrt{5})^k - (1-\sqrt{5})^k],$$

as required. ■

2.3 Counting paths in a graph

We shall use induction to prove a theorem about the powers of the adjacency matrix of a graph.

Example 2.11 Consider the following graph whose edges I have labelled.



List all the paths of length 2 between any two vertices. List the paths of length 3 between any two vertices. ■

Our next theorem tells us how to calculate the number of paths of any given length between any two given vertices.

Theorem 2.12 *Let A be the adjacency matrix of the graph G. Then the number of paths of length m from vertex i to vertex j is $(A^m)_{ij}$.*

Proof I shall assume that the vertices of the graph G are numbered $1, 2, \ldots$. The theorem will be proved by induction.

Base case: for $m = 1$ the result is immediate because the paths of length 1 are just the edges.

Induction hypothesis: assume that the number of paths of length n from vertex i to vertex j is $(A^n)_{ij}$. We shall prove that $(A^{n+1})_{ij}$ is the number of paths of length $n + 1$ from vertex i to vertex j.

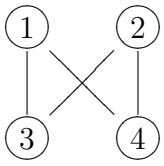
Idea: every path of length $n + 1$ is a path of length n with an extra edge attached. Thus the number of paths of length $n+1$ from i to j is equal to the sum of the following: the number of paths of length n from i to some vertex k, times the number of edges from k to j. But by the induction hypothesis the number of paths of length n from i to some vertex k is $(A^n)_{ik}$, and the number of edges from k to j is $(A)_{kj}$. Thus the number of paths of length $n + 1$ from i to j is:

$$\sum_{k=1}^{n} (A^n)_{ik}(A)_{kj}$$

but this is precisely $(A^{n+1})_{ij}$, as required. ■

I'll refer back to the example above to show you that this theorem makes sense.

Exercise 2.13 The graph $K_{2,2}$ is



Write down the adjacency matrix of this graph. How many paths of length 2 are there joining vertex 1 to vertex 2? How many paths of length 3 are there joining vertex 4 to vertex 1? ■

3 Trees

A *tree* is a connected graph containing no cycles. I'll give some examples in the lecture to make this definition clearer. As you will see, the term 'tree' is well-earned.

The key property on which this section is based is the following: *amongst all connected graphs with n vertices, trees have the smallest number of edges:* $n-1$. This is proved in Section 3.1, and its practical consequences are explored in Sections 3.2 and 3.3. It is the practical consequences which are most important to us.

3.1 The 'most efficient' connected graphs

In this section, I shall describe some theoretical results about connected graphs. Remember that a graph is *connected* if it is all in one piece. A better definition is to say that a graph is connected if any two vertices are joined by a path in the graph. If a graph is not connected we say it is *disconnected*. A disconnected graph consists of pieces each of which is connected. These pieces are called the *connected components* of the graph.

(i) Given n vertices, what is the minimum number of edges you need to draw so that you get a connected graph on those vertices?

Examples 3.1 Given 4,5 or 6 vertices, what are the minimum number of edges needed to make a connected graph having those vertices? ■

These examples suggest that a *connected* graph with n vertices must have at least $n-1$ edges.

Proposition 3.2 *Let G be a connected graph having n vertices. Then the smallest number of edges it can have is $n-1$.*

Proof We begin with some preliminary results.

Let G be a connected graph. We claim that if an edge of G is erased, then the resulting graph has at most two components. To see why, let G' be the graph that results by erasing an edge. Suppose G' had at least three components. Then replacing the erased edge would join at most two components back together, which is a contradiction. More generally, we claim

that if we erase r edges from a graph the resulting graph will have at most $r+1$ components. We prove this by induction. We have already proved that if we erase one edge we get at most two components. Induction hypothesis: suppose for all $k \leq n$ if we erase k edges we get at most $k+1$ components. Suppose now we erase $n+1$ edges. To do this, we first erase n edges and that gives us at most $n+1$ components. The final edge will be erased from one of the connected components of the resulting graph which will split into at most two components. This gives the result.

We are now ready to prove the proposition. Let G be a connected graph with n vertices. Assume that each graph having $k \leq n$ vertices has at least $k-1$ edges. Let G be a graph with $n+1$ vertices. Choose any vertex v. Because the graph is connected the degree of v is at least 1. Suppose that the edges incident with v are $e_1, \ldots, e_s$. Erasing these edges gives a graph with the vertex v forming one component on its own and at most s other components by the result above. Let the number of vertices in each of these components be $v_1, \ldots, v_s$. Then $v_1 + \ldots + v_s = n$ and by the induction hypothesis the component with v_i vertices has at least $v_i - 1$ edges. Thus the original graph consists of at least $s + (v_1 - 1) + \ldots + (v_s - 1) = n$ edges, as required. ∎

(ii) What can we say about those connected graphs with n vertices having exactly $n-1$ edges?

Proposition 3.3 *A connected graph with n vertices and $n-1$ edges is a tree.*

Proof Recall that a tree is a connected graph having no cycles. Suppose G is a graph with n vertices and exactly $n-1$ edges and it is not a tree. Then it must contain a cycle. Choose such a cycle and erase one of its edges. The graph G' that results has n vertices and $n-2$ edges and is still connected, but this is impossible by Proposition 3.2. Thus G really cannot contain cycles and so must be a tree. ∎

(iii) How many edges does a tree with n vertices have?

Proposition 3.4 *A tree with n vertices has exactly $n-1$ edges.*

Proof We prove this by induction on the number of vertices. If $n = 1$ there are no edges and so the result holds.

Induction hypothesis: each tree with k vertices where $k \leq n$ has $k - 1$ edges. Now consider a tree with $n + 1$ vertices. If we erase an edge of a tree the resulting graph must split into two connected components having k_1 and k_2 vertices. By the induction hypothesis, these have $k_1 - 1$ and $k_2 - 1$ edges respectively. Thus the original tree has $1 + (k_1 - 1) + (k_2 - 1) = n$ edges, as required. ∎

Theorem 3.5 *Let G be a connected graph with n vertices. Then G is a tree if and only if it has exactly $n - 1$ edges.*

Proof Let G be a connected graph with n vertices. Suppose first that G is a tree. Then by Proposition 3.4, it has $n-1$ edges. Now suppose that G has $n - 1$ edges. Then by Proposition 3.3 it is a tree. ∎

Theorem 3.6 *Amongst all connected graphs with n vertices, the trees have the smallest number of edges: $n - 1$.*

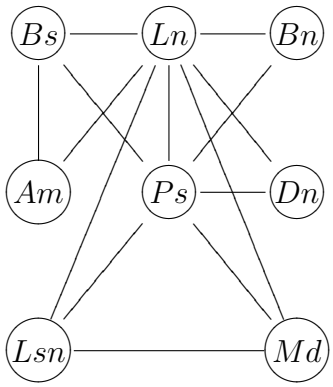
Proof Let G be a connected graph with n vertices. By Proposition 3.2, the smallest number of edges it can have is $n-1$. By Proposition 3.3 a connected graph with n vertices and $n - 1$ edges is a tree. By Theorem 3.5, trees are characterised as connected graphs with n vertices and $n - 1$ edges. ∎

In other words, trees are the 'most efficient' connected graphs.

3.2 Spanning trees

Before stating this problem formally, I shall begin with an example that illustrates the ideas involved.

Example 3.7 Below is a graph showing fictional airline routes in part of Europe between certain cities.



Rather unrealistically, I shall assume that the cost of each direct route between any two cities is the same. If you run an airline you want to be able to do two things: guarantee that you can fly passengers between any two cities, possibly with intermediate stops, and minimise the cost to you of doing this. ■

This problem can be formulated mathematically without reference to airline routes. Given a graph G find a subgraph G' that satisfies the following three conditions:

(i) The graph G' has the same vertices as G.

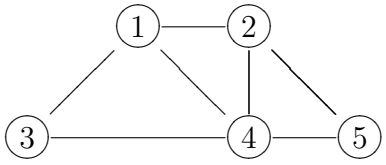
(ii) The graph G' is connected.

(iii) The graph G' has the smallest number of edges possible amongst all subgraphs of G satisfying (i) and (ii).

Now (ii) and (iii) tell us that G' must be a tree by Theorem 3.6. Thus we are looking for a subgraph of G that is a tree and satisfies (i); such a subgraph of G is called a 'spanning tree' of G.

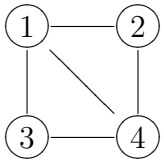
That is: given a graph G a *spanning tree* for G is a subgraph of G that is a tree and includes all the vertices of G.

Example 3.8 I shall give some examples of spanning trees of the following graph in the lectures



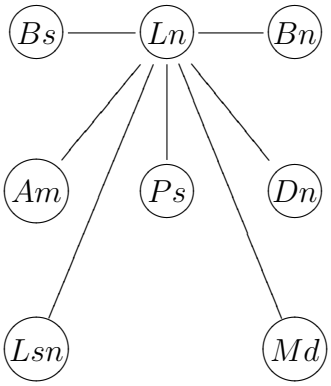
■

Exercise 3.9 Find *all* spanning trees of the following graph



■

Example 3.10 If we return to our original problem, we can now easily find solutions. Here is one (there are lots of others):



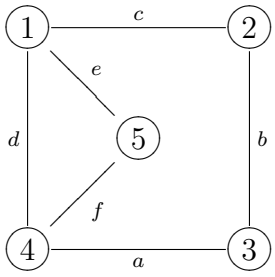
We have found the solution to the original problem. ■

Finding spanning trees of small graphs is easy and can be carried out by inspection, but we need to be more systematic if we are going to find spanning trees of real graphs. To this end, I shall now describe an algorithm for finding spanning trees.

Algorithm: spanning tree
Input: A connected graph G.
Output: A spanning tree T for G.
Procedure: The set T is initially empty.

1. Choose any vertex of G and put it into T.
2. Repeat the following procedure until it is no longer possible to choose an edge satisfying the conditions, at which point the algorithm terminates: choose an edge of G that joins a vertex in T to a vertex not in T and put the edge in T. ■

Example 3.11 In the lecture, I shall apply the algorithm **spanning tree** to the following graph. I have labelled the edges for clarity.



■

Theorem 3.12 *The algorithm* **spanning tree** *works. Hence every connected graph has a spanning tree.*

Proof We show first that all vertices are used. We prove this using contradiction. Suppose the algorithm terminates and there is a vertex u not in T. Since G is connected there is a path in G joining u to a vertex v in T. On this path there must be an edge having one vertex in T and the other not in T, but this contradicts the fact that the algorithm terminated. The fact that T contains no cycles follows from the fact that we never add an edge between vertices already in T so we can never make a cycle. It remains to show that T is connected but this follows from the fact that we only join edges to pre-existing edges in T. ■

Exercise 3.13 Apply the algorithm **spanning tree** to the graph of Exercise 3.9 twice over: first, starting at vertex 1, and second at vertex 2. Show all steps in the application of the algorithm ■

3.3 Prim's algorithm

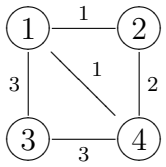
My airline problem above was unrealistic because different airline routes are likely to cost different amounts, so not only do we need to find a spanning tree we need to find one where the total cost is as small as possible. This can be formalised as follows.

A *weighted graph* is one whose edges are labelled by positive numbers, called *weights*. These numbers might represent distances, if the vertices were towns and the edges were roads, or they might represent costs.

The *total weight* of a spanning tree in a weighted graph is the sum of all the weights on all the edges in the tree. A *minimum spanning tree* is a spanning tree whose total weight is less than or equal to the total weight of any other spanning tree.

Returning to my airline example, we might make the graph a weighted graph whose weights were costs. A minimum spanning tree would then give me the cheapest solution to the problem of connecting any two cites by means of airline routes.

Exercise 3.14 List all the spanning trees of the following weighted graph and so find the minimum weight spanning trees.



■

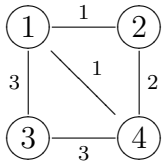
I shall now describe an algorithm which will find such a minimum spanning tree. It generalises the algorithm for finding a spanning tree which is only slightly changed.

Prim's algorithm: minimum spanning tree
Input: A connected weighted graph G.
Output: A minimum spanning tree T for G.
Procedure: The set T is initially empty.

1. Choose any vertex of G and put it into T.

2. Repeat the following procedure until it is no longer possible to choose an edge satisfying the conditions, at which point the algorithm terminates: choose an edge of G of *smallest weight* that joins a vertex in T to a vertex not in T and put the edge in T. ■

Exercise 3.15 Apply Prim's algorithm to the following weighted graph and so find the minimum weight spanning trees. Show all steps in the application of the algorithm.



■

Remark Prim invented his algorithm in 1957 to help solve a problem in computer network design, but it had already been discovered by the Czech mathematician Jarnik in 1930 and so ought to be called the 'Jarnik-Prim algorithm'.

Theorem 3.16 [5] *Prim's algorithm works.*

Proof We prove that at each step in the algorithm, T is a subgraph of a minimum spanning tree.

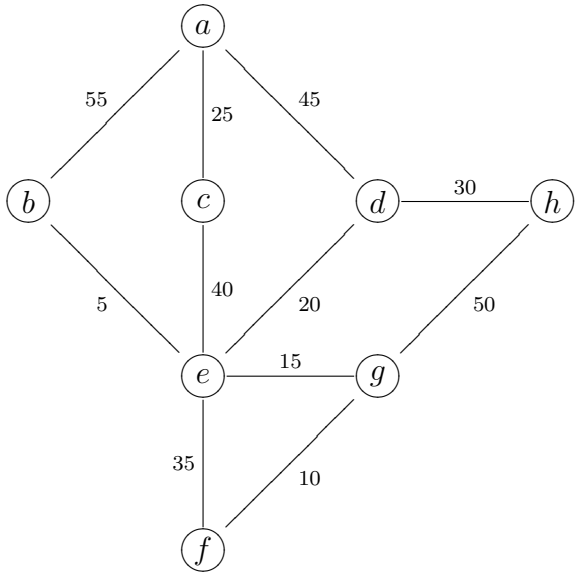
At the beginning of the algorithm, T consists of a single vertex and this must belong to every minimum spanning tree.

Now suppose the algorithm proceeds and at some stage T is contained in a minimum weight spanning tree S. We show that after the next iteration of the algorithm, the new tree T' is also contained in a minimum spanning tree. The tree T' is obtained by adding an edge e from u to v where u is a vertex of T and v is not, such that the weight of e is minimum amongst all edges that have exactly one of their vertices in T. If e is contained in S then S is a minimum spanning tree containing T', as required. Suppose that e is not contained in S. Then adding e to S will make a cycle C. Choose an edge $e' \neq e$ in C with the property that one vertex of e' is in T and one is not. Then the weight of e is less than or equal to the weight of e', from the

[5] This is a hard proof, and I've only included it for the sake of completeness.

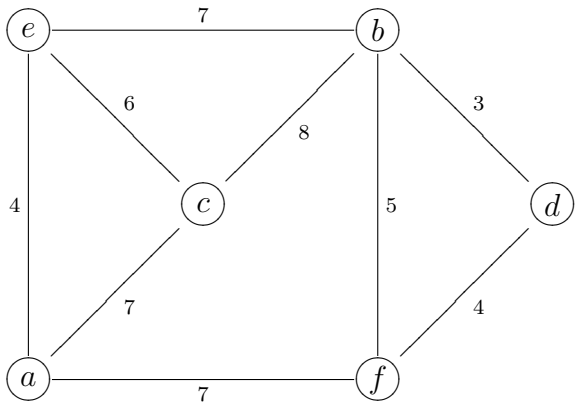
way we chose e'. Let S' be obtained from S by erasing e' and adding e. By construction S' is a spanning tree of G and its total weight is less than or equal to the total weight of S. But S was a minimum spanning tree. Hence S' is a minimum spanning tree. Furthermore, S' contains T', as required. ■

Exercise 3.17 Apply Prim's algorithm to the weighted graph below two times starting first at f and then again at a.



■

Exercise 3.18 Apply Prim's algorithm to the weighted graph below two times starting first at b and then again at a.



■

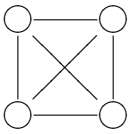
4 Planar graphs

A graph is said to be "planar" if it can be drawn without edges crossing. Planar graphs arise naturally from maps: the countries are represented by the vertices and two vertices are joined by an edge if the countries share a common border. They also arise from polyhedra: the points of the polyhedron are the vertices and the edges of the polyhedron form the edges of the graph. The Platonic solids give rise to the Platonic graphs, and the 'Buckyballs' of chemistry give rise to graphs with faces (see below) being either 5-cycles or 6-cycles. As we shall see, some graphs are planar and some aren't. The problem is: how to distinguish the planar from the non-planar ones?

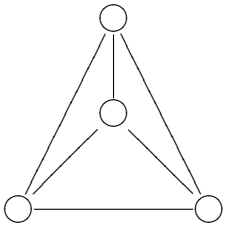
4.1 Faces

A graph is *planar* if it can be drawn without edges crossing.

Example 4.1 The graph



is planar because it can be redrawn as follows



■

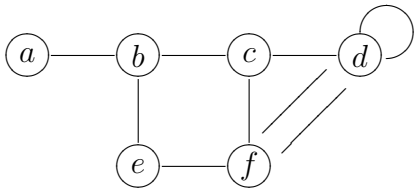
Exercise 4.2 Is the graph $K_{3,3}$ planar? ■

For planar graphs, drawn so that edges don't intersect, there are some new numbers we can calculate in addition to the number of vertices, edges and the vertex degrees. If we draw a planar graph in such a way that edges don't cross, the plane is divided into a number of regions, including the outside region, called *faces*. Each face has a *face degree* this is the total number of edges we meet in circumnavigating the boundary of the face: this may

involve counting the same edge twice — once on each side.

Warning! To find faces and face degrees you must first draw the graph so that no edges cross.

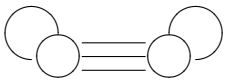
Example 4.3 Consider the following planar graph drawn so that edges don't cross.



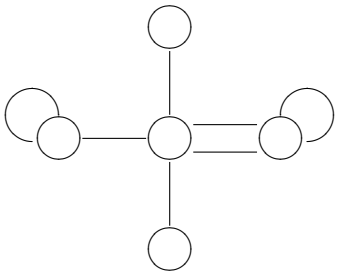
There are five faces: face 1 is the outside region, face 2 the square, face 3 the triangle, face 4 is bounded by two edges, and face 5 is the face bounded by the loop. For each face we shall calculate the face degree. In addition, we shall add up the face degrees. ■

Exercise 4.4 For each of the following graphs find the number of edges (e), the number of vertices (v), the number of faces (f), all face degrees, and the sum of the face degrees.

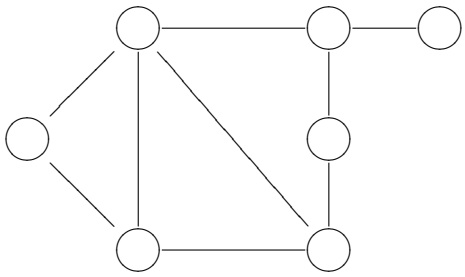
(i)



(ii)



(iii)



■

Lemma 4.5 (Face degrees) *In a planar connected graph, the sum of the face degrees is twice the number of edges.*

Proof Each edge is on the boundary of exactly two faces or occurs twice when we travel around a single face. ■

4.2 Euler's formula

For a planar graph the number of faces, the number of edges and the number of vertices are connected.

Theorem 4.6 (Euler's Formula) *Let G be a planar connected graph having f faces, e edges, and v vertices. Then*

$$f - e + v = 2.$$

Proof Let T be a spanning tree for G. Then T has one face, v vertices and $v - 1$ edges. Thus $1 - (v - 1) + v = 2$. Now let's see what happens as we add the remaining edges of G back to T one at a time. This will give us the proof of the theorem, as I shall show in the lecture. ■

Proposition 4.7 *In a simple, connected planar graph having v vertices and e edges, where $v \geq 3$, we have that*

$$e \leq 3v - 6.$$

Proof Let the graph have face degrees $\phi_1 \leq \phi_2 \leq \ldots \leq \phi_f$. Then by the result on face degrees

$$2e = \sum_{i=1}^{f} \phi_i.$$

The graph is simple and so there are no loops or multiple edges. In addition, there are at least 3 vertices. This implies that the smallest a face degree can be is 3. Thus $\phi_i \geq 3$ for all i. Since there are f faces we have that

$$2e \geq 3f.$$

By Euler's Formula

$$f - e + v = 2$$

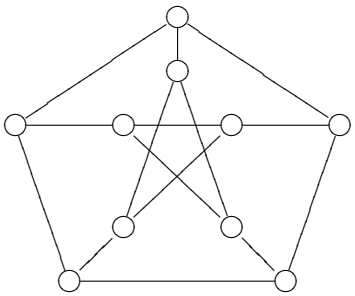
and so $f = 2+e-v$. Combining our two results we have that $2e \geq 6+3e-3v$. Rearranging this inequality we get

$$e \leq 3v - 6$$

as required. ■

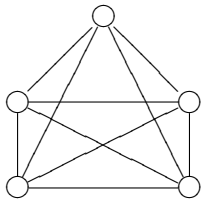
4.3 Deciding planarity

Given a graph, how would you prove that it was planar or, failing that, that it was not planar? To convince someone that a graph is planar, you simply have to show them how to deform the graph as drawn to one in which the edges don't cross. But how would you convince someone that a graph wasn't planar? This is much more difficult: just because a few attempts fail to show that a graph is planar does not mean it is non-planar: after all, maybe you missed a trick. A good example is the Petersen graph below: it seems to be non-planar, but how can we be certain?

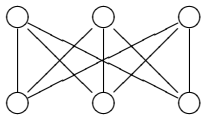


What we would like is to find some *simpler* property of a graph that is easy to check and that will tell us precisely whether a graph is planar or not.

Two graphs play an important role in answering this question. The first is K_5



and the second is $K_{3,3}$



Proposition 4.8 *The graph K_5 is not planar.*

Proof Suppose that K_5 were planar. In this graph $v = 5$ and $e = 10$. Thus by Euler's formula $f = e - v + 2 = 10 - 5 + 2 = 7$. Let the face degrees be $f_1 \leq \ldots \leq f_7$. By the Lemma on face degrees we know that

$$20 = \sum_{i=1}^{7} f_i.$$

The graph is simple and so the smallest possible face degree is 3 (and indeed there are triangles in this graph). It follows that

$$20 = \sum_{i=1}^{7} f_i \geq 7 \times 3 = 21.$$

This is nonsense and so the graph cannot be planar. ∎

Proposition 4.9 *The graph $K_{3,3}$ is not planar.*

Proof Suppose that $K_{3,3}$ were planar. In this graph $v = 6$ and $e = 9$. Thus by Euler's formula $f = e - v + 2 = 9 - 6 + 2 = 5$. Let the face degrees be $f_1 \leq \ldots \leq f_5$. By the Lemma on face degrees we know that

$$18 = \sum_{i=1}^{5} f_i.$$

The graph is simple and contains no triangles and so the smallest possible face degree is 4 (and indeed there are cycles of length 4 in this graph). It follows that

$$18 = \sum_{i=1}^{5} f_i \geq 5 \times 4 = 20.$$

This is nonsense and so the graph cannot be planar. ■

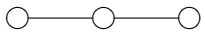
We can use these two results to prove that other graphs are not planar.

Proposition 4.10 *A graph G that contains a subgraph G' which is non-planar is itself non-planar.*

Proof If G were planar then G' would be too, which is a contradiction. ■

Thus graphs containing K_5 or $K_{3,3}$ as subgraphs cannot be planar.
Next, we introduce a new operation on graphs. Let G be a graph, and let

be any edge. Replace this edge by



The resulting graph G' has one extra vertex and one extra edge. It is called a *subdivision* of G; more generally, any graph that arises from G by repeated applications of this construction is called a subdivision of G.

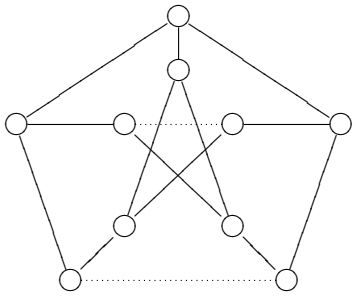
Proposition 4.11 *Subdivisions of $K_{3,3}$ and K_5 are also non-planar.*

Proof If the subdivisions were planar then the original graphs would also be planar, which is a contradiction. ■

From Propositions 4.10 and 4.11 we get the following useful result:

Theorem 4.12 *If a graph contains a subgraph that is a subdivision of K_5 or $K_{3,3}$ then it is not planar.* ■

Example 4.13 I shall prove in the lecture that the Petersen graph, below, is not planar. To start the proof off, I have dotted a couple of edges.



■

Is it true that a non-planar graph must contain as a subgraph a subdivision of K_5 or $K_{3,3}$? We cannot answer this question on the basis of what we have proved so far — it may well be that there are other graphs that also cause non-planarity. Remarkably, however, this complication does not arise: it can be proved that:

> if a graph is non-planar then it must contain as a subgraph a subdivision of K_5 or $K_{3,3}$.

This is much harder to prove then the results above, so I will not include the proof here.

If we combine the two main results above, we get the following, a result due to Kazimierz Kuratowski, a Polish mathematician who studied engineering at Glasgow University.

Theorem 4.14 (Kuratowksi's Theorem) *A graph is planar if and only if it does not contain as a subgraph a subdivision of K_5 or $K_{3,3}$.* ■

5 Vertex colouring

We shall now look at the problem of how to colour the vertices of a graph in such a way that adjacent vertices have different colours. This problem arose from the Victorian children's pastime of colouring-in maps; gave rise to the 'four colour conjecture' only proved in 1976 and even then only with the help of a computer, and led to the question of whether it is possible to decide 'quickly' whether a graph can be 3-coloured, a question known to be 'NP-complete' and so leading to one of the great unsolved questions of mathematics and theoretical computer science: 'is $P = NP$?' In addition, vertex colouring is essentially what Sudoku puzzles are, and the same idea is used to allocate frequencies to mobile phones. This is a long way from crayons and colouring books, but mathematics is not snobbish about where it finds its problems — the only thing that is important is whether they are interesting or not.

5.1 Introduction

Let me begin by describing two problems that on the face of it have nothing in common.

Example 5.1 A scheduling problem: A mathematics department plans to offer seven graduate courses next semester in: combinatorics (C); group theory (G); linear programming (L); numerical analysis (N); probability (P); statistics (S); and topology (T). The 12 graduate students make the following choices:

Adam C, L, T

Beth C, G, S

Charles G, N

Delia C, L

Ed L, N

Ffion C, G

George N, P

Hariot G, L

Ifor C, T

Jane C, S, T

Kelvin P, S

Lois P, T

The problem is to find the minimum number of time periods needed for these seven courses. ■

Example 5.2 Map colouring: Given a map of the world colour the countries in such a way that neighbouring countries (countries that share a border) are coloured differently so that they show up. In addition, try to do this by using the smallest number of different colours. ■

Neither problem appears to have anything to do with graphs, but they do. In the lectures, I shall explain how a map gives rise to a graph: the vertices of the graph are the countries and two countries are joined by an edge iff they share a border. Colouring the map corresponds to colouring the vertices of the graph in such a way that adjacent vertices are coloured differently.

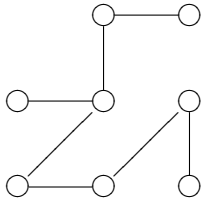
Now let me return to the scheduling problem. This too can be regarded as a graph problem. Two courses cannot be run at the same time if the same student attends both courses. We can represent this information by an *incompatibility graph*: the vertices are the courses — C,G,L,N,P,S,T. We join two vertices (ie two courses) by an edge iff there is at least one student taking both courses. In the lectures, I shall ask you to draw such a graph. The reason to do this is that now we can forget about the students and just concentrate on the relationships between the courses as represented by the graph; the information is now being presented in a way that is easier to use. Let the time periods be 1,2,3, ... We need to assign numbers to the vertices in such a way that adjacent vertices have different numbers. At the same time, we need to minimise the number of numbers used. We shall see that in this case, 4 time periods are needed.

5.2 Vertex colouring

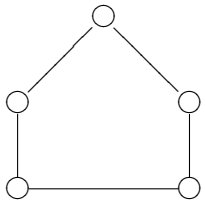
Both map colouring and scheduling are examples of the following graph-theoretic ideas. Let G be a simple graph. A *vertex colouring* of G is an assignment of colours to the vertices of G in such a way that adjacent vertices have different colours. The *chromatic number* of G, denoted $\chi(G)$, is the **smallest** number of colours needed amongst all vertex colourings. A vertex colouring using $\chi(G)$ colours is called a *minimum colouring*.

Exercise 5.3 Calculate the chromatic numbers of the following graphs and justify your answers.

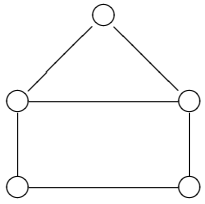
(i)



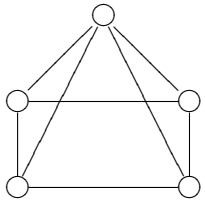
(ii)



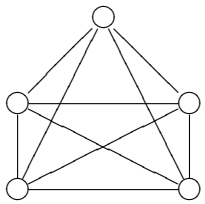
(iii)



(iv)



(v)



■

Finding the chromatic number of a graph is difficult in general: algorithms for computing it do little more than find all possible vertex colourings and pick the best, which is a very time-consuming process. It is possible to get some bounds. The following result gives us an upper bound — which needn't be very good.

Proposition 5.4 *Let G be a simple graph whose maximum vertex degree is d. Then $\chi(G) \leq d + 1$.*

Proof We shall prove this result by induction on n the number of vertices of the graph G. If n is one then $\chi(G) = 1$ and $d = 0$, so the result holds.

Induction hypothesis: suppose result is true for all graphs having at most n vertices. We prove that the result is true for all graphs having $n + 1$ vertices. Let G be a simple graph having $n+1$ vertices. Let v be a vertex having maximum vertex degree d. Let G' be the graph that arises by erasing v and all edges incident to it. The graph G' has n vertices and maximum vertex degree at most d, so by the induction hypothesis has chromatic number at most $d + 1$. We can now colour G by carrying over the colouring from G' and colouring v with the colour not used in its d neighbours. ■

We can sometimes get lower bounds. The proof of the following is immediate once you note that K_m has chromatic number m.

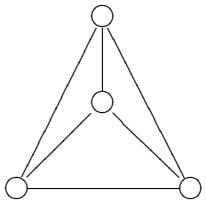
Proposition 5.5 *Let G be a simple graph that contains K_m as a subgraph. Then $m \leq \chi(G)$.* ■

5.3 The four colour theorem

One of the most famous theorems in graph theory is the following. It implies that every map can be coloured with at most four colours.

Theorem 5.6 (The four colour theorem) *Every simple planar graph has chromatic number at most four.* ■

This result had been conjectured about 150 years ago. It was finally proved in 1976. It is unique in mathematics in that the only known proofs require the use of a computer.[6] Notice that we can't do better than 4 because the following planar graph has chromatic number 4.



Although I can't prove the four colour theorem for you, I can prove a slightly weaker result: the six colour theorem! To do this, I need first the following result which is interesting in its own right.

Proposition 5.7 *A connected planar simple graph contains at least one vertex of degree 5 or less.*

Proof Let the vertex degrees be $d_1 \leq \ldots \leq d_v$. Then by the Handshaking Lemma $2e = \sum_{i=1}^{v} d_i$. Suppose the degree of every vertex was at least 6. Then by the Handshaking Lemma, we would have $2e \geq 6v$. Thus $e \geq 3v$. But by Proposition 4.7, $e \leq 3v - 6$. We therefore get a contradiction. ■

Theorem 5.8 (The six colour theorem) *Every simple planar graph has chromatic number at most six.*

Proof We don't lose any generality by assuming that the graph is connected (why not?) We will prove the theorem by induction on the number of vertices. The result is clearly true if the graph has only one vertex. We assume

[6] See R. Wilson, *Four colours suffice*, Penguin, 2002 for the full story.

that all connected simple planar graphs with at most n vertices can be 6-coloured. Let G be a connected planar simple graph with $n+1$ vertices. By Proposition 5.7, there is a vertex v of degree at most 5. Erase the vertex v from G together with all edges incident with v to obtain a new graph G'. This graph has n vertices and so can be 6-coloured by the induction hypothesis. Now replace v and its incident edges. Because v has degree at most 5 we add back in at most 5 edges. Colour v with a colour different from all the vertices adjacent to v: this is possible, because the worst that can happen is that v is connected to 5 vertices and these 5 vertices are all coloured differently. In this case, we still have a sixth colour to play with.■

It's possible to do better than Theorem 5.7 by using the same basic proof combined with a very clever idea.

Theorem 5.9 (The five colour theorem) *Every simple planar graph has chromatic number at most five.*

Proof We don't lose any generality by assuming that the graph is connected as in the proof of Theorem 5.8. We now prove the theorem by induction on the number of vertices. We assume that all connected simple planar graphs with at most n vertices can be 5-coloured. Let G be a connected planar simple graph with $n+1$ vertices. By Proposition 5.7, there is a vertex v of degree at most 5. If v has degree at most 4 then the proof is simple. We shall therefore assume that v has degree exactly 5. Erase v and the edges incident with it to obtain a new graph G'. By induction, this graph can be 5-coloured. Now replace v and its incident edges. We shall show how to colour v by modifying the colouring of G'. Let the vertices adjacent to v be $v_1, \ldots, v_5$, coloured 1,2,3,4,5, respectively.

Look at the subgraph H of G' that consists of all vertices coloured 1 or 3 and the edges that join them. There are two cases we have to consider.

(Case 1) Suppose that v_1 and v_3 are in different connected components of H. Then interchange the colours 1 and 3 in the connected component containing v_3. This means that v_3 is now coloured 1 and so the colour 3 is available to colour v and we are done.

(Case 2) Suppose instead that v_1 and v_3 are in the same connected component of H, then this trick will not work. However, it follows that if we look at the subgraph of G' coloured 2 or 4 then the vertices v_2 and v_4 must belong to different connected components because they are separated by a

path joining v_1 and v_3 whose vertices are coloured 1 or 3. Thus we can apply our recolouring procedure to v_2 and v_4 instead.

In both cases, we get a 5-colouring of G, as required. ■

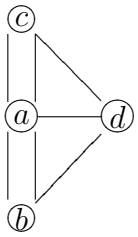
6 Eulerian graphs

In this section, I shall describe the Königsberg bridge problem and its consequences. This problem was historically one of the origins of graph theory.

6.1 The Königsberg bridge problem

The city of Königsberg was built around a river that contained an island connected to either bank by a total of seven bridges. The people of the city of Königsberg used to like to walk around their city crossing and recrossing some of the bridges over the river. The question arose of whether it was possible to do a round-trip crossing each bridge exactly once. This problem came to the attention of the mathematician Euler (pronounced 'Oiler').

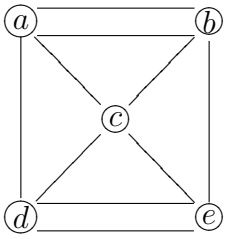
Example 6.1 The graph of the Königsberg Bridge Problem.



■

A closed path in a graph which uses each edge exactly once is called an *Euler tour* of the graph. A graph which has an Euler tour is said to be *Eulerian*. Note that we can cross and recross a given vertex many times — it is the edges that must be used exactly once.

Exercise 6.2 Find an Euler tour in the following graph. Ensure that your tour is indicated unambiguously.



■

Proposition 6.3 *If a graph G has an Euler tour then G is connected and every vertex has even degree.*

Proof It is clear that the graph must be connected. To show that each vertex degree is even, assume that an Euler tour has been found. Then at each vertex, there is for each incoming edge a distinct outgoing edge in the tour. Thus the number of edges at a vertex is even. ∎

It turns out that the converse of this statement is true: if G is connected and every vertex has even degree then G has an Euler tour. To prove this we shall need the following lemma.

Lemma 6.4 *Let G be a connected graph in which each vertex has even degree. Then G has at least one cycle.*

Proof Without loss of generality we may assume that the graph is simple and consists of more than one vertex. Pick any vertex v_0, and construct a path starting at v_0 using distinct edges which is as long as possible. Let the vertices used be $v_0, v_1, \ldots, v_n$. Now the vertex v_n has even degree and so must be joined to a vertex different from v_{n-1}. But the path was supposed to be as long as possible having distinct edges and so v_n must be joined by an edge to one of the vertices $v_0, \ldots, v_{n-2}$. Suppose it is joined to v_i. Then $v_i, v_{i+1}, \ldots, v_n, v_i$ forms a cycle, as required. ∎

Theorem 6.5 *A graph is Eulerian if and only if it is connected and every vertex has even order.*

Proof We have already proved that an Eulerian graph is connected and every vertex degree is even. It remains to be proved that a connected graph in which every vertex has even degree is Eulerian. This is proved by induction on the number of edges m. If $m = 0$ then the result is clear. So suppose the result is true for all appropriate graphs with at most m edges. Let G be a connected graph with $m+1$ edges in which every vertex has even degree. By Lemma 6.4, there is a cycle C in G. Erase the edges of C from G to get a new graph G'. The connected components of G' will each have the property that all vertex degrees are even, so by the induction hypothesis each will have an Euler tour. We now find an Euler tour of G. Start at any vertex u of C and traverse the edges of C until we meet a vertex v belonging to one

of the components of G'. Take the Euler tour of this component returning to v. Continue in this way along C until we return to u. ∎

In the proof of the above theorem, the cycle is used to link together the Euler tours on each of the components.

6.2 An algorithm

Theorem 6.5 tells us precisely *when* a graph is Eulerian, but it does not tell us *how* to find an Euler tour. I shall now describe an algorithm that takes as input a connected graph in which every vertex has even degree and produces as output an Euler tour. The algorithm is an application of the idea that makes the proof of Theorem 6.5 work.

The algorithm comes in two stages:

Stage 1 The graph G is split into cycles no two of which have an edge in common. I shall say that such a set of cycles is a 'complete set of disjoint cycles'.

Stage 2 The cycles constructed in (stage 1) are glued together to make an Euler tour.

Algorithm: disjoint cycles
Input: A connected graph G in which each vertex has even degree.
Output: A complete set of disjoint cycles DC for G. We regard DC as a list.

Procedure:

1. Find a cycle C in G and put it in DC. Let G' be the graph obtained by erasing the edges of C from G and any isolated vertices that result.

2. Repeat the following procedure until G' is empty: find a cycle C' in G'; add C' to DC; erase the edges of C' from G' and any isolated vertices that result; call the new graph that results G'. ∎

I shall prove in the lecture that this algorithm works.

The algorithm for the second stage (**algorithm: glueing disjoint cycles**) is best explained visually; I think of it as the 'wheels within wheels' algorithm for reasons that will become obvious during the lecture.

Algorithm: glueing disjoint cycles
Input: The graph G above and the output, DC, of the algorithm **disjoint cycles**.
Output: An Euler tour for G.

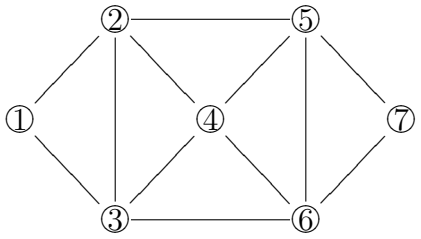
Procedure:

1. Choose C_1 from DC. Remove C_1 from DC. Choose a start vertex in C_1. Traverse C_1 clockwise. This is a partial Euler tour (PET).

2. Repeat the following procedure until there are no more cycles left in DC, at which point output PET: traverse PET from the start vertex until you meet a vertex v that is also shared by a cycle C' in DC. **Add the detour** C' to PET, and erase C' from DC.

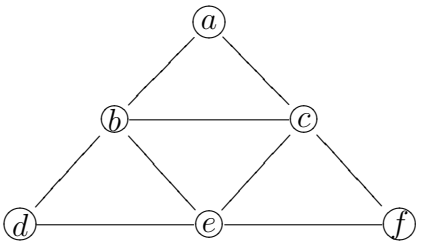
The only part of the algorithm that needs explaining is the procedure I've called **adding the detour**. This works as follows. We already have our partial Euler tour PET. Whilst traversing PET, we meet a vertex v shared by a cycle C' in DC. We enlarge PET as follows: traverse PET from the start vertex until v is met; add the traverse of C' that starts at v and travels clockwise around C' until we return to v; now continue on the original tour. We have therefore enlarged PET by adding a detour via C'. ■

Example 6.6 Apply the algorithms above to find an Euler tour of the following graph.



■

Exercise 6.7 Apply the algorithms above to find an Euler tour of the following graph.



■