

# MAT 7003 : Mathematical Foundations

(for Software Engineering)

J Paul Gibson, A207

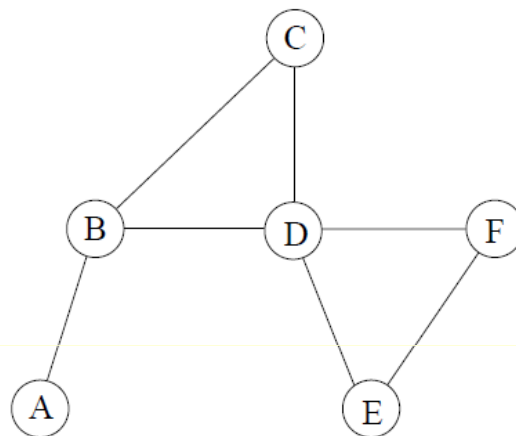
[paul.gibson@it-sudparis.eu](mailto:paul.gibson@it-sudparis.eu)

<http://www-public.it-sudparis.eu/~gibson/Teaching/MAT7003/>

## Graph Theory

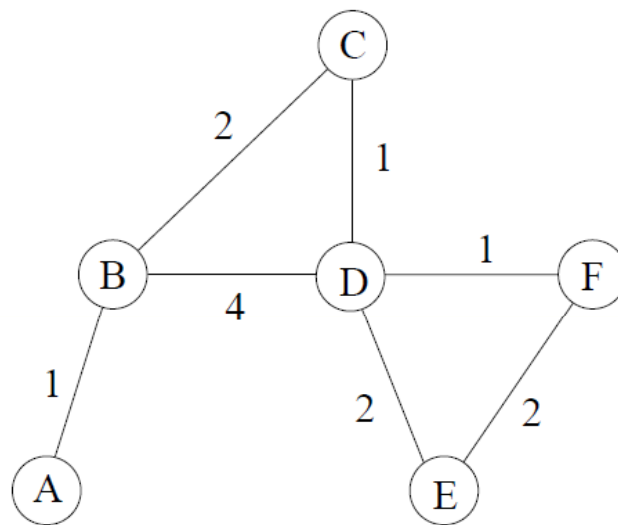
<http://www-public.it-sudparis.eu/~gibson/Teaching/MAT7003/L9b-GraphTheory.pdf>

What is a graph?



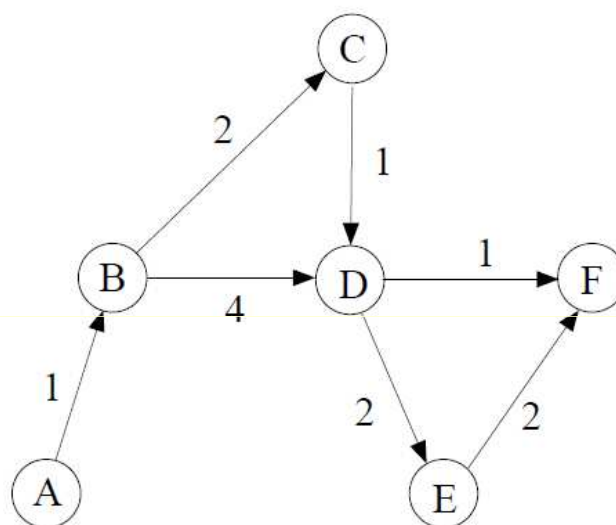
- ▶ Graphs have *vertices* and *edges*.
- ▶ An edge normally connects two vertices.
- ▶ There are different types of graphs, which we will look at later...

## Weighted graphs



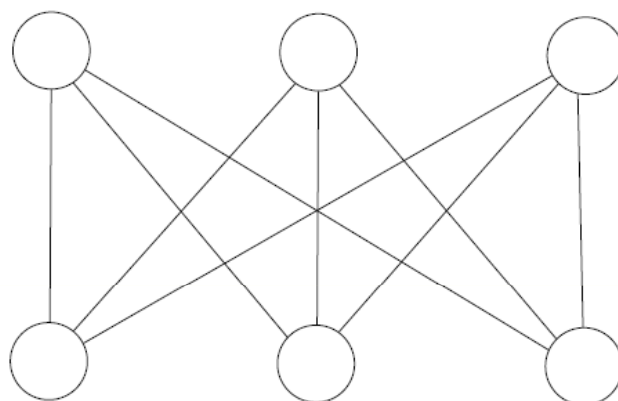
- ▶ Can extend graphs by associating a *weight* with each edge.
- ▶ Might represent e.g. the cost of travelling between two points.

## Directed graphs (digraphs)



- ▶ Can also make edges *directional*.
- ▶ This might now represent, for example, a network of one-way streets.

## Bipartite graphs



- ▶ In this type of graph, the vertices are divided into two sets.
- ▶ There are no edges between vertices in the same set.
- ▶ This graph is also *complete* (all possible edges are present).

## Trees

Trees are graphs which are:

- Acyclic
- Connected
- Simple

Every graph has a number of edges,  $\varepsilon$ .

Every graph also has a number of vertices,  $\nu$ .

Exercise: prove that for a tree,

$$\varepsilon = \nu - 1$$

Hint: think about the simplest tree first, then add vertices one by one (proof by induction).

Therefore, if any simple, connected graph has  $\nu - 1$  edges, it must be a tree. Can you prove this?

## The degree of a vertex

- ▶ The degree of a vertex  $d(v)$  is the number of edges which are incident to it.
- ▶ Exercise: show that

$$\sum_{v \in V} d(v) = 2\varepsilon$$

**TO DO:** Given the previous result, show that there are at least two vertices in any tree with degree 1.

**A cut edge is an edge which would disconnect a graph if it was removed.**

**QUESTION:** How many cut edges are there in a tree?

**Minimum spanning tree** - a **subgraph which** connects every vertex in the graph using the lowest number of edges.

**QUESTION:** Why is this sure to be a tree?

## When are two graphs equivalent?

- ▶ If two graphs have the same structure, but with different labels, or drawn in a different way, they are **isomorphic**.
- ▶ Sometimes it is easy to tell if two graphs are not isomorphic (e.g. because they have a different number of vertices) . . .
- ▶ . . . but sometimes it is difficult.

## Some confusing terminology

- ▶ An *edge cut* is a subset of the edges of a graph, such that removing them disconnects the graph.
- ▶ A *vertex cut* is a subset of the vertices of a graph, such that removing them disconnects the graph.
- ▶ Don't get confused with cut edges and cut vertices!
- ▶ Cut edges are sometimes referred to as *bridges*.

## Connectivity

- ▶ The connectivity  $\kappa(G)$  is the smallest  $k$  for which there is a  $k$ -vertex cut in  $G$ .
- ▶ In a communications network, for example, this would tell us how resilient the network is:  $k$  nodes have to be removed before there is any disconnection.

## Edge connectivity

- ▶ The edge connectivity  $\kappa'(G)$  is a similar concept for the smallest number of edges which can be removed to disconnect a graph.
- ▶ We say a graph is  $k$ -edge connected if it takes  $k$  edges to be removed from the graph to disconnect it.
- ▶ So this is another measure of how strongly connected the graph is.

## Euler tours

- ▶ *Trails* are walks which never traverse the same edge twice.
- ▶ *Tours* or *cycles* are trails which end up at the place they started.
- ▶ An *Euler tour* traverses every edge of a graph exactly once.
- ▶ If a graph contains an Euler tour, it is *eulerian*.
- ▶ Graphs are eulerian iff all vertices are of even degree.  
**Why?**

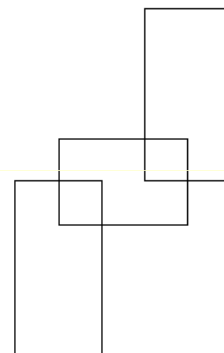
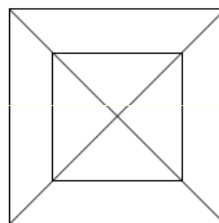
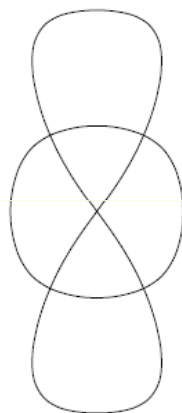
## Proof: an eulerian graph must have all vertices of even degree

1. Let  $C$  be an Euler tour of graph  $G$ , which starts and ends at vertex  $u$ .
2. Each time a vertex is included in the tour  $C$ , two edges incident to that vertex are used up.
3. Every edge in  $G$  is included in the tour. So every vertex other than  $u$  must have even degree.
4. The tour starts and ends at  $u$ , so it must also have even degree.

## Proof: a graph with all vertices of even degree must be eulerian

1. **Assume the opposite:**  $G$  is a noneulerian graph with all vertices of even degree.
2.  $G$  must contain a closed trail. Let  $C$  be the largest possible closed trail in the graph.
3. Because of our assumption,  $C$  must have missed out some of the graph  $G$ , call this  $G'$ .
4.  $C$  is eulerian, so has no vertices of odd degree.  $G'$  therefore also has no vertices of odd degree.
5.  $G'$  must have some tour  $C'$ , and there must be a common vertex  $v$  in both  $C$  and  $C'$ .
6.  $CC'$  therefore makes a larger trail than  $C$ , which violates our assumption in (2). **Contradiction.**

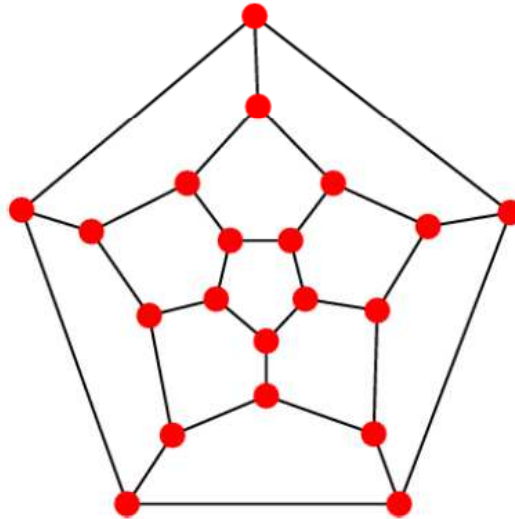
Which of these shapes are eulerian?



(i.e. which can you draw without removing your pen from the page, and without covering any lines twice?)

## Hamilton paths

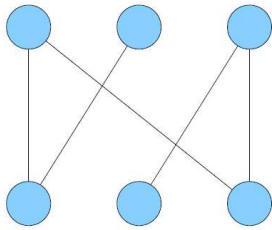
- ▶ A Hamilton path visits every vertex (once).
- ▶ A Hamilton cycle does the same, and finishes at the place it started. The Dodecahedron has many Hamilton cycles:



## Matchings

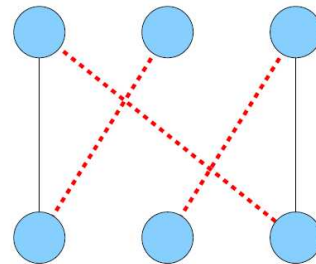
- ▶ A *matching* is a subset of edges in a graph which have no common vertices.
- ▶ For each edge  $M$  in a matching, the two vertices at either end are matched.
- ▶ A *maximum matching* is one in which as many vertices are matched as possible.
- ▶ A *perfect matching* is one in which every vertex is matched.
- ▶ An  *$M$ -alternating path* in a graph is one in which the edges are alternately in  $M$  and  $G \setminus M$ .

Example: this bipartite graph...

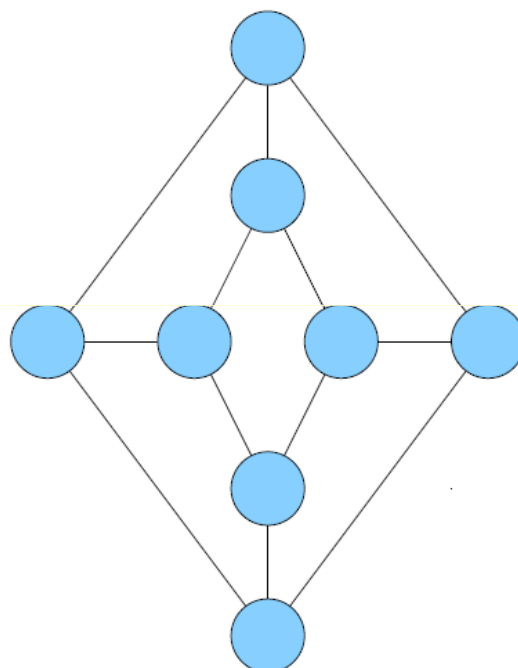


... has this (perfect) matching.

Matching  $M \subseteq E$  shown as dashed lines:



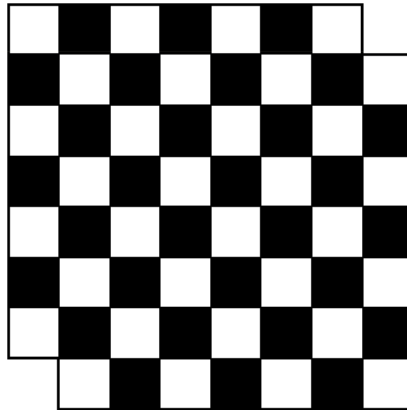
**TO DO:** Does this graph have a matching?



**TO DO :** Prove that a tree has a maximum of one perfect match

## Matches in bipartite graphs

- ▶ An  $8 \times 8$  square grid has the corner tiles removed.
- ▶ Is it possible to cover the whole of the remaining grid with  $2 \times 1$  blocks?



## Colourings

- ▶ A *colouring* of a graph  $\mathcal{C}$  is an assignment of  $k$  colours to the edges of the graph.
- ▶ In a *proper* colouring, no two adjacent edges are the same colour.
- ▶ If  $G$  can be coloured with  $k$  colours, then we say it is *k-edge-colourable*.
- ▶ If  $k$  is the minimum number of colours for which this is possible, the graph is *k-edge-chromatic*.
- ▶ In this case,  $k$  is the *edge chromatic number*,  $\chi(G)$ .

## A colouring as a set of matchings

- ▶ Think of a colouring as a partitioning of the edges into subsets.
- ▶ Each subset contains no vertex no more than once.
- ▶ Therefore each subset is a matching.

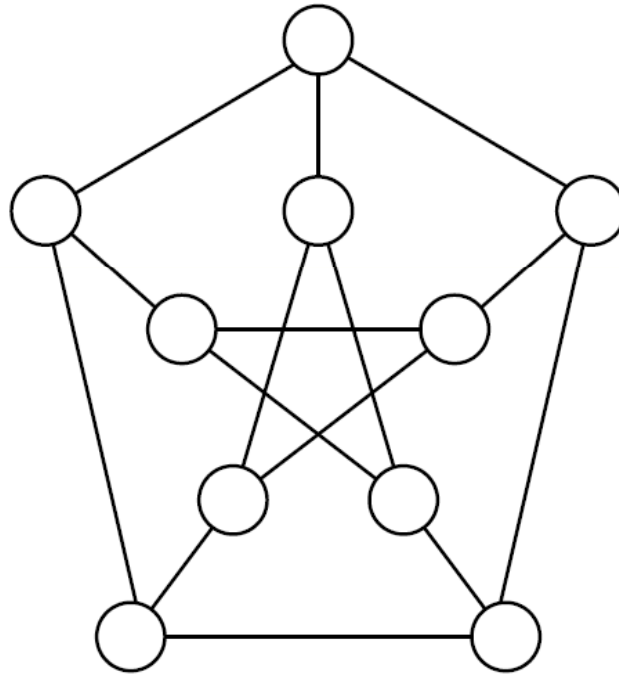
## What's the smallest edge colouring?

- ▶  $\chi(G)$  can't be less than the largest degree in the graph  $\Delta$ .
- ▶ It also cannot be more than  $\Delta + 1$  (Vizing's theorem).
- ▶ Therefore  $\chi(G)$  is always equal to either  $\Delta$  or  $\Delta + 1$ .
- ▶ A bipartite graph is always  $\Delta$ -edge-chromatic.

## TO DO:

### Edge colouring in the Petersen graph

Show that this graph is 4-edge-chromatic:

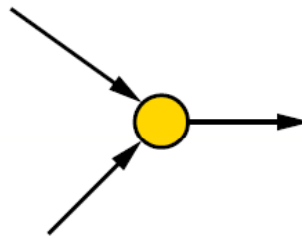


## Directed graphs

- ▶ Also called *digraphs*
- ▶  $D = (V, A, \psi)$   
where  $V$  are vertices,  $A$  are arcs,  $\psi$  is an incidence function giving an ordered pair of vertices for each arc.
- ▶ If  $\psi(a) = (u, v)$ , then arc  $a$  is a connection from  $u$  to  $v$ .
- ▶ Every digraph  $D$  has an underlying undirected graph  $G$ .
- ▶ We say that  $\psi$  is an *orientation* of  $G$ .
- ▶ **Exercise: how many orientations does a graph  $G$  have?**

## Incidence

- ▶ The notion of “degree” needs to be generalised in a digraph. We talk about indegree and outdegree:  $d^-(v)$ , and  $d^+(v)$ .



$$d^-(v) = 2$$
$$d^+(v) = 1$$

- ▶ The maximum indegree and outdegree are denoted by  $\Delta^-$  and  $\Delta^+$ , while the minima are denoted by  $\delta^-$  and  $\delta^+$ .
- ▶ **Exercise:** show that  $\sum_{v \in V} d^-(v) = \varepsilon = \sum_{v \in V} d^+(v)$ .

## Counting people in a queue

- ▶ While waiting in a huge queue at Stanbic bank, you might want to try and calculate how many people are in the queue (it goes outside the door and round the corner, so it is not possible to count everyone by sight).

Start of queue

You

End of queue



- ▶ A simple way to do this is to announce with a loudspeaker that everyone in the queue is to shout out their name. You then note down all the responses and count how many there are.
- ▶ However, this method is not very efficient or reliable—you have a lot of communication and processing to do.

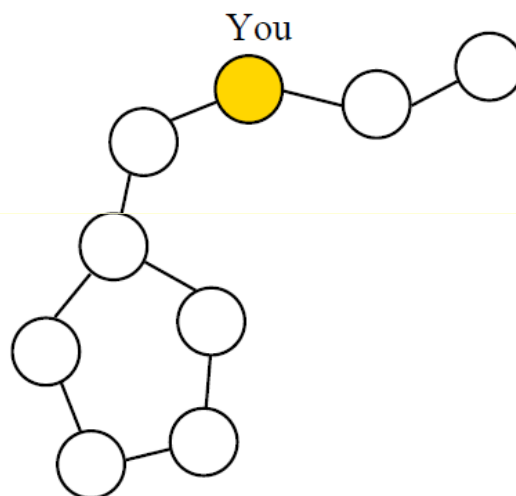
## A solution using message passing

The following method is more efficient:

1. If a person is at either end of a queue, they send the message "1" to their neighbour.
2. If a person has received a number from his neighbour, he adds one to that number and passes that message on to the neighbour on the other side.
3. When anyone has received messages from both their neighbours, they can calculate how many people there are in the queue by adding both messages and then adding one (for themselves).

## Cyclic graphs

The solution does not apply when there are cycles. Consider trying to count people standing in the following arrangement:



There are two problems: a vertex with degree three (for which we have no rule defined), and the presence of the cycle.

## Separable problems

- ▶ The problems which we can solve using message-passing algorithms have a property of *separability*.
- ▶ The simplest example is for counting the number of people in the queue. The number of people in front is separate from the number of people behind; when there is a cycle the separability property no longer holds.
- ▶ Trees also have the separability property, because every vertex is a cut vertex.

## Counting vertices in a tree with message passing

Each person should follow these rules:

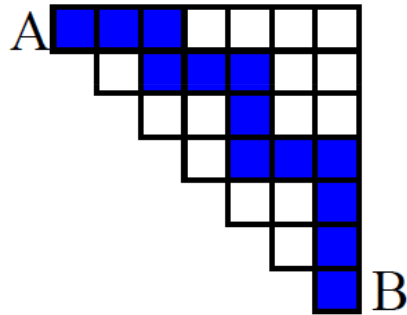
1. Count your neighbours,  $d$  (the degree of your vertex).
2. Keep count of how many messages you have received. When you have received  $d - 1$  messages, add them together, then add 1 and pass this message to the remaining neighbour (the one who has not sent you a message).
3. If you have received  $d$  messages:
  - 3.1 The sum of the messages plus 1 is the answer.
  - 3.2 Send to neighbour  $n$  the total answer minus  $v_n$  (the message from that neighbour).

Note that step 2 applies to people who are at the end of a line (vertices with degree 1).

## TO DO - PROBLEM: Using Message Passing in a graph

### Counting paths through a grid

- ▶ Say that valid paths are made up of rightward and downward moves.



- ▶ How many possible paths are there between A and B?