

# COMP9020 Week 7

## Term 3, 2019

### Graph Theory

- Textbook (R & W) - Ch. 3, Sec. 3.2; Ch. 6, Sec. 6.1–6.5
- A. Aho & J. Ullman. *Foundations of Computer Science in C*, p. 522–526 (Ch. 9, Sec. 9.10)

# Summary of topics

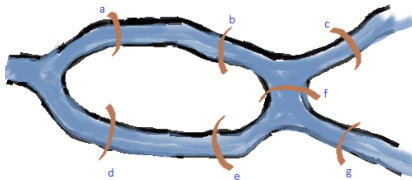
- Motivation and applications
- Terminology and notation
- Graph traversals
- Properties of graphs

# Summary of topics

- Motivation and applications
- Terminology and notation
- Graph traversals
- Properties of graphs

# Graph theory: Historical Motivation

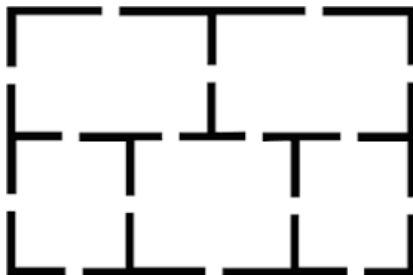
Bridges of Königsberg problem



Can you find a route which crosses each bridge exactly once?

# Graph theory: Historical Motivation

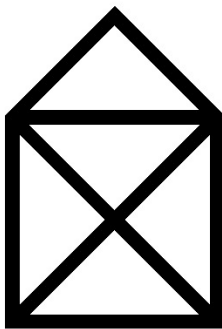
Five rooms problem



Can you find a route which passes through each door exactly once?

# Graph theory: Historical Motivation

Crossed house problem



Can you draw this without taking your pen off the paper?

# Graph theory: Historical Motivation

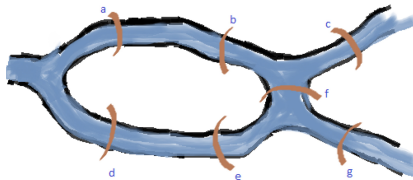
Three utilities problem



Can you connect all utilities to all houses without crossing connections?

# Graph theory: Historical Motivation

Bridges of Königsberg problem

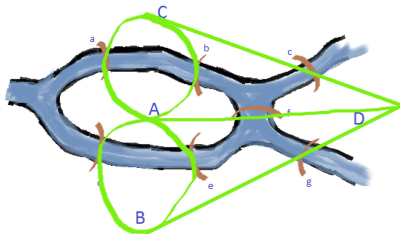


Can you find a route which crosses each bridge exactly once?



# Graph theory: Historical Motivation

Bridges of Königsberg problem



Can you find a route which crosses each bridge exactly once?

# Graphs in Computer Science

## Examples

- 1 The WWW can be considered a massive graph where the nodes are web pages and arcs are hyperlinks.
- 2 The possible states of a program form a directed graph.
- 3 Circuit components and their connections form a graph.
- 4 Social networks can be viewed as a graph where the nodes are users and the edges are connections.
- 5 The map of the earth can be represented as an undirected graph where edges delineate countries.

# Graphs in Computer Science

Applications of graphs in Computer Science are abundant, e.g.

- route planning in navigation systems, robotics
- optimisation, e.g. timetables, utilisation of network structures, bandwidth allocation
- compilers using “graph colouring” to assign registers to program variables
- circuit layout ([Untangle game](#))
- determining the significance of a web page (Google's pagerank algorithm)
- modelling the spread of a virus in a computer network or news in social network

# Summary of topics

- Motivation and applications
- Terminology and notation
- Graph traversals
- Properties of graphs

# Graphs

Terminology (the most common; there are many variants):

**Graph** — pair  $(V, E)$  where  $V$  — set of vertices (or nodes)  
 $E$  — set of edges

**Undirected graph:** Every edge  $e \in E$  is a two-element set of vertices, i.e.  $e = \{x, y\} \subseteq V$  where  $x \neq y$

**Directed graph:** Every edge (or arc)  $e \in E$  is an ordered pair of vertices, i.e.  $e = (x, y) \in V \times V$ , note  $x$  may equal  $y$ .

## NB

*Binary relations on finite sets correspond to directed graphs.  
Symmetric, antireflexive relations correspond to undirected graphs.*

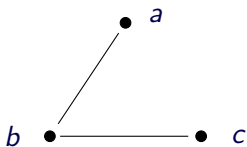
# Graph representations

Graph:

$$V = \{a, b, c\}$$

$$E = \{\{a, b\}, \{b, c\}\}$$

Pictorially:

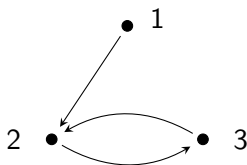


Directed graph:

$$V = \{1, 2, 3\}$$

$$E = \{(1, 2), (2, 3), (3, 2)\}$$

Pictorially:



# Graph representations

Graph:

$$V = \{a, b, c\}$$

$$E = \{\{a, b\}, \{b, c\}\}$$

Adjacency matrix:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Directed graph:

$$V = \{1, 2, 3\}$$

$$E = \{(1, 2), (2, 3), (3, 2)\}$$

Adjacency matrix:

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

# Graph representations

Graph:

$$V = \{a, b, c\}$$

$$E = \{\{a, b\}, \{b, c\}\}$$

Adjacency list:

$a : b$   
 $b : a, c$   
 $c : b$

Directed graph:

$$V = \{1, 2, 3\}$$

$$E = \{(1, 2), (2, 3), (3, 2)\}$$

Adjacency list:

$1 : 2$   
 $2 : 3$   
 $3 :$



# Graph representations

Graph:

$$V = \{a, b, c\}$$

$$E = \{\{a, b\}, \{b, c\}\}$$

Incidence matrix

(vertices=rows,  
edges=columns):

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}$$

Directed graph:

$$V = \{1, 2, 3\}$$

$$E = \{(1, 2), (2, 3), (3, 2)\}$$

Incidence matrix

(vertices=rows,  
edges=columns):

$$\begin{pmatrix} -1 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix}$$

# Vertex Degrees (Undirected graphs)

- **Degree** of a vertex

$$\deg(v) = |\{ w \in V : \{v, w\} \in E \}|$$

i.e., the number of edges attached to the vertex

- **Regular graph** — all degrees are equal
- *Degree sequence*  $D_0, D_1, D_2, \dots, D_k$  of graph  $G = (V, E)$ , where  $D_i$  = no. of vertices of degree  $i$

## Question

What is  $D_0 + D_1 + \dots + D_k$ ?

- $\sum_{v \in V} \deg(v) = 2 \cdot e(G)$ ; thus the sum of vertex degrees is always even.
- There is an even number of vertices of odd degree (6.1.8)

# Vertex Degrees (Directed graphs)

- **Out-degree** of a vertex

$$\text{outdeg}(v) = |\{ w \in V : (v, w) \in E \}|$$

i.e., the number of edges going out of the vertex

- **In-degree** of a vertex

$$\text{indeg}(v) = |\{ w \in V : (w, v) \in E \}|$$

i.e., the number of edges going in to the vertex

- $\sum_{v \in V} \text{outdeg}(v) = \sum_{v \in V} \text{indeg}(v) = e(G).$

# Paths

- A **(directed) path** in a (directed) graph  $(V, E)$  is a sequence of edges that link up

$$v_0 \xrightarrow{\{v_0, v_1\}} v_1 \xrightarrow{\{v_1, v_2\}} \dots \xrightarrow{\{v_{n-1}, v_n\}} v_n$$

where  $e_i = \{v_{i-1}, v_i\} \in E$  (or  $e_i = (v_{i-1}, v_i) \in E$ )

- **length** of the path is the number of edges:  $n$   
neither the vertices nor the edges have to be all different
- Subpath of length  $r$ :  $(e_m, e_{m+1}, \dots, e_{m+r-1})$
- Path of length 0: single vertex  $v_0$
- **Connected graph (undirected)** — each pair of vertices joined by a path
- **Strongly connected graph (directed)** — each pair of vertices joined by a directed path in both directions

# Exercises

## Exercises

6.1.13(a) Draw a connected, regular graph on four vertices, each of degree 2

6.1.13(b) Draw a connected, regular graph on four vertices, each of degree 3

6.1.13(c) Draw a connected, regular graph on five vertices, each of degree 3

6.1.14(a) Graph with 3 vertices and 3 edges

6.1.14(b) Two graphs each with 4 vertices and 4 edges

# Exercises

## Exercises

?

# Exercises

## NB

*We use the notation*

$n = v(G) = |V|$  for the no. of vertices of graph  $G = (V, E)$

$m = e(G) = |E|$  for the no. of edges of graph  $G = (V, E)$

## Exercises

6.1.20(a) Graph with  $e(G) = 21$  edges has a degree sequence  $D_0 = 0, D_1 = 7, D_2 = 3, D_3 = 7, D_4 = ?$   
Find  $v(G)$

6.1.20(b) How would your answer change, if at all, when  $D_0 = 6$ ?

# Exercises

## NB

*We use the notation*

$n = v(G) = |V|$  for the no. of vertices of graph  $G = (V, E)$

$m = e(G) = |E|$  for the no. of edges of graph  $G = (V, E)$

## Exercises

6.1.20(a) Graph with  $e(G) = 21$  edges has a degree sequence  
 $D_0 = 0, D_1 = 7, D_2 = 3, D_3 = 7, D_4 = ?$   
Find  $v(G)$

?

6.1.20(b) How would your answer change, if at all, when  $D_0 = 6$ ?

?



# Cycles

Recall paths  $v_0 \xrightarrow{e_1} v_1 \xrightarrow{e_2} \dots \xrightarrow{e_n} v_n$

- *simple path* —  $e_i \neq e_j$  for all edges of the path ( $i \neq j$ )
- *closed path* —  $v_0 = v_n$
- **cycle** — closed path, all other  $v_i$  pairwise distinct and  $\neq v_0$
- *acyclic path* —  $v_i \neq v_j$  for all vertices in the path ( $i \neq j$ )

## NB

- ①  $C = (e_1, \dots, e_n)$  is a cycle iff removing any single edge leaves an acyclic path. (Show that the 'any' condition is needed!)
- ②  $C$  is a cycle if it has the same number of edges and vertices and no proper subpath has this property.  
(Show that the 'subpath' condition is needed, i.e., there are graphs  $G$  that are **not** cycles and  $|E_G| = |V_G|$ ; every such  $G$  must contain a cycle!)

# Trees

- **Acyclic graph** — graph that doesn't contain any cycle
- **Tree** — connected acyclic [undirected]graph
- A graph is acyclic *iff* it is a *forest* (collection of disjoint trees)

## NB

*Graph  $G$  is a tree iff*

- $\Leftrightarrow$  *it is acyclic and  $|V_G| = |E_G| + 1$ .  
(Show how this implies that the graph is connected!)*
- $\Leftrightarrow$  *there is exactly one simple path between any two vertices.*
- $\Leftrightarrow$   *$G$  is connected, but becomes disconnected if any single edge is removed.*
- $\Leftrightarrow$   *$G$  is acyclic, but has a cycle if any single edge on already existing vertices is added.*

# Trees

A tree with one vertex designated as its *root* is called a *rooted tree*. It imposes an ordering on the edges: 'away' from the root — from parent nodes to children. This defines a *level number* (or: *depth*) of a node as its distance from the root.

Another very common notion in Computer Science is that of a *DAG* — a *directed, acyclic graph*.

## Exercise (Supplementary)

### Exercises

**6.7.3** (Supp) Tree with  $n$  vertices,  $n \geq 3$ .

Always true, false or could be either?

- (a)  $e(T) \stackrel{?}{=} n$
- (b) at least one vertex of deg 2
- (c) at least two  $v_1, v_2$  s.t.  $\deg(v_1) = \deg(v_2)$
- (d) exactly one path from  $v_1$  to  $v_2$

## Exercise (Supplementary)

### Exercises

**6.7.3** (Supp) Tree with  $n$  vertices,  $n \geq 3$ .

Always true, false or could be either?

(a)  $e(T) \stackrel{?}{=} n$  ?

(b) at least one vertex of deg 2 ?

(c) at least two  $v_1, v_2$  s.t.  $\deg(v_1) = \deg(v_2)$  ?

(d) exactly one path from  $v_1$  to  $v_2$  ?

## Special Graphs

- **Complete graph  $K_n$**

$n$  vertices, all pairwise connected,  $\frac{n(n-1)}{2}$  edges.

- **Complete bipartite graph  $K_{m,n}$**

Has  $m + n$  vertices, partitioned into two (disjoint) sets, one of  $n$ , the other of  $m$  vertices.

All vertices from different parts are connected; vertices from the same part are disconnected. No. of edges is  $m \cdot n$ .

- **Complete  $k$ -partite graph  $K_{m_1, \dots, m_k}$**

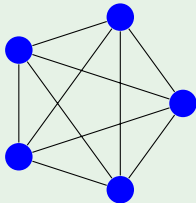
Has  $m_1 + \dots + m_k$  vertices, partitioned into  $k$  disjoint sets, respectively of  $m_1, m_2, \dots$  vertices.

No. of edges is  $\sum_{i < j} m_i m_j = \frac{1}{2} \sum_{i \neq j} m_i m_j$

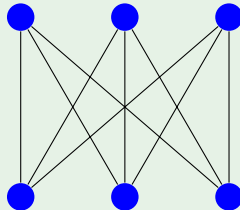
- These graphs generalise the complete graphs  $K_n = \underbrace{K_{1, \dots, 1}}_n$

## Example

$K_5$  :



$K_{3,3}$  :



## Graph Isomorphisms

$\phi : G \longrightarrow H$  is a *graph isomorphism* if

- (i)  $\phi : V_G \longrightarrow V_H$  is a bijection
- (ii)  $(x, y) \in E_G$  iff  $(\phi(x), \phi(y)) \in E_H$

Two graphs are called *isomorphic* if there exists (at least one) isomorphism between them.



# Graph Isomorphisms

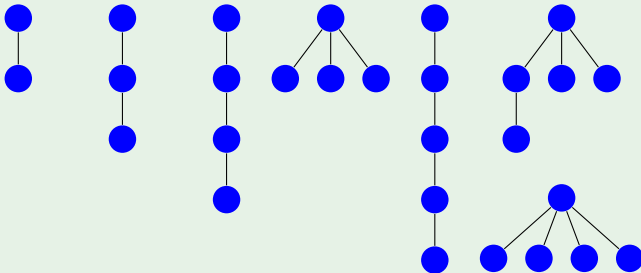
$\phi : G \longrightarrow H$  is a *graph isomorphism* if

- (i)  $\phi : V_G \longrightarrow V_H$  is a bijection
- (ii)  $(x, y) \in E_G$  iff  $(\phi(x), \phi(y)) \in E_H$

Two graphs are called *isomorphic* if there exists (at least one) isomorphism between them.

## Example

All nonisomorphic trees on 2, 3, 4 and 5 vertices.



# Automorphisms and Asymmetric Graphs

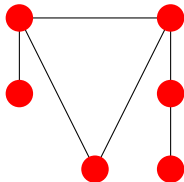
An isomorphism from a graph to itself is called *automorphism*.

Every graph has at least the trivial automorphism;

(trivial meaning  $\phi(v) = v$  for all  $v \in V_G$ )

Graphs with no non-trivial automorphisms are called *asymmetric*.

The smallest non-trivial asymmetric graphs have 6 vertices.



(Can you find another one with 6 nodes? There are seven more.)

# Summary of topics

- Motivation and applications
- Terminology and notation
- **Graph traversals**
- Properties of graphs

# Edge Traversal

## Definition

- **Euler path** — path containing every edge exactly once
- **Euler circuit** — closed Euler path

## Characterisations

- $G$  (connected) has an Euler circuit iff  $\deg(v)$  is even for all  $v \in V$ .
- $G$  (connected) has an Euler path iff either it has an Euler circuit (above) or it has exactly two vertices of odd degree.

## NB

- *These characterisations apply to graphs with loops as well*
- *For directed graphs the condition for existence of an Euler circuit is  $\text{indeg}(v) = \text{outdeg}(v)$  for all  $v \in V$*

# Exercises

## Exercises

**6.2.11** Construct a graph with vertex set  $\{0, 1\} \times \{0, 1\} \times \{0, 1\}$  and with an edge between vertices if they differ in exactly two coordinates.

- (a) How many components does this graph have?
- (b) How many vertices of each degree?
- (c) Euler circuit?

**6.2.12** As Ex. 6.2.11 but with an edge between vertices if they differ in two or three coordinates.

# Exercises

## Exercises

?

# Exercises

## Exercises

6.2.14 Which complete graphs  $K_n$  have an Euler circuit?  
When do bipartite, 3-partite complete graphs have an Euler circuit?

# Exercises

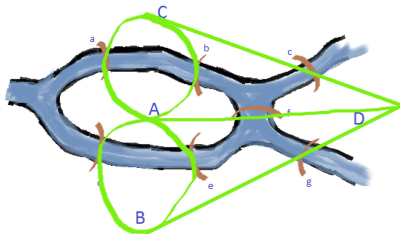
## Exercises

6.2.14 Which complete graphs  $K_n$  have an Euler circuit?  
When do bipartite, 3-partite complete graphs have an Euler circuit?  
?



# Bridges of Königsberg

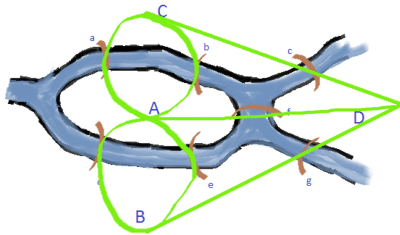
Bridges of Königsberg problem



Can you find a route which crosses each bridge exactly once?

# Bridges of Königsberg

Bridges of Königsberg problem



Can you find a route which crosses each bridge exactly once? No!

# Vertex Traversal

## Definition

- **Hamiltonian path** visits every vertex of graph exactly once
- **Hamiltonian cycle** visits every vertex exactly once except the last one, which duplicates the first

## NB

*Finding such a cycle, or proving it does not exist, is a difficult problem — the worst case is NP-complete.*

## Examples (when the cycle exists)

- All five regular polyhedra (verify!)
- $n$ -cube; Hamiltonian circuit = *Gray code*
- $K_m$  for all  $m$ ;  $K_{m,n}$  iff  $m = n$ ;  $K_{a,b,c}$  iff  $a, b, c$  satisfy the triangle inequalities:  $a + b \geq c$ ,  $a + c \geq b$ ,  $b + c \geq a$
- Knight's tour on a chessboard (incl. rectangular boards)

Examples when a Hamiltonian cycle does not exist are much harder to construct.

Also, given such a graph it is nontrivial to verify that indeed there is no such a cycle: there is nothing obvious to specify that could assure us about this property.

In contrast, if a cycle is given, it is immediate to verify that it is a Hamiltonian cycle.

These situations demonstrate the often enormous discrepancy in difficulty of 'proving' versus (simply) 'checking'.

# Exercise

## Exercise

6.5.5(a) How many Hamiltonian cycles does  $K_{n,n}$  have?

# Exercise

## Exercise

6.5.5(a) How many Hamiltonian cycles does  $K_{n,n}$  have?  
?

# Summary of topics

- Motivation and applications
- Terminology and notation
- Graph traversals
- Properties of graphs

## Colouring

Informally: assigning a “colour” to each vertex (e.g. a node in an electric or transportation network) so that the vertices connected by an edge have different colours.

Formally: A mapping  $c : V \longrightarrow [1..n]$  such that for every  $e = (v, w) \in E$

$$c(v) \neq c(w)$$

The minimum  $n$  sufficient to effect such a mapping is called the **chromatic number** of a graph  $G = (E, V)$  and is denoted  $\chi(G)$ .

### NB

*This notion is extremely important in operations research, esp. in scheduling.*

*There is a dual notion of ‘edge colouring’ — two edges that share a vertex need to have different colours. Curiously enough, it is much less useful in practice.*



# Properties of the Chromatic Number

- $\chi(K_n) = n$
- If  $G$  has  $n$  vertices and  $\chi(G) = n$  then  $G = K_n$

## Proof.

Suppose that  $G$  is 'missing' the edge  $(v, w)$ , as compared with  $K_n$ . Colour all vertices, except  $w$ , using  $n - 1$  colours. Then assign to  $w$  the same colour as that of  $v$ . □

- If  $\chi(G) = 1$  then  $G$  is totally disconnected: it has 0 edges.
- If  $\chi(G) = 2$  then  $G$  is bipartite.
- For any tree  $\chi(T) = 2$ .
- For any cycle  $C_n$  its chromatic number depends on the parity of  $n$  — for  $n$  even  $\chi(C_n) = 2$ , while for  $n$  odd  $\chi(C_n) = 3$ .

# Cliques

Graph  $(V', E')$  *subgraph* of  $(V, E)$  —  $V' \subseteq V$  and  $E' \subseteq E$ .

## Definition

A **clique** in  $G$  is a *complete subgraph of  $G$* . A clique of  $k$  nodes is called  *$k$ -clique*.

The size of the largest clique is called the *clique number* of the graph and denoted  $\kappa(G)$ .

## Theorem

$$\chi(G) \geq \kappa(G).$$

## Proof.

Every vertex of a clique requires a different colour, hence there must be at least  $\kappa(G)$  colours. □

However, this is the only restriction. For any given  $k$  there are graphs with  $\kappa(G) = k$ , while  $\chi(G)$  can be arbitrarily large.

## NB

*This fact (and such graphs) are important in the analysis of parallel computation algorithms.*

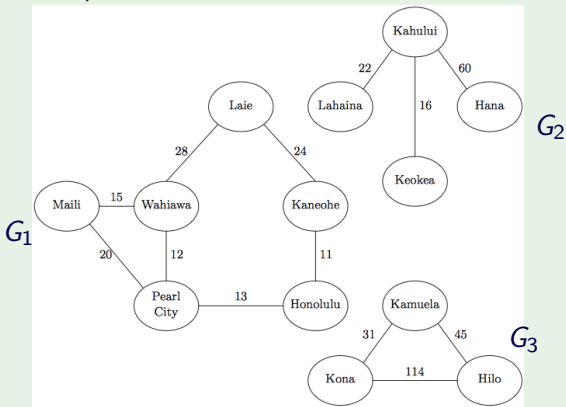
- $\kappa(K_n) = n$ ,  $\kappa(K_{m,n}) = 2$ ,  $\kappa(K_{m_1, \dots, m_r}) = r$ .
- If  $\kappa(G) = 1$  then  $G$  is totally disconnected.
- For a tree  $\kappa(T) = 2$ .
- For a cycle  $C_n$   
 $\kappa(C_3) = 3$ ,  $\kappa(C_4) = \kappa(C_5) = \dots = 2$

The difference between  $\kappa(G)$  and  $\chi(G)$  is apparent with just  $\kappa(G) = 2$  — this does not imply that  $G$  is bipartite. For example, the cycle  $C_n$  for any odd  $n$  has  $\chi(C_n) = 3$ .

# Exercise

## Exercise

9.10.1 (Ullmann)

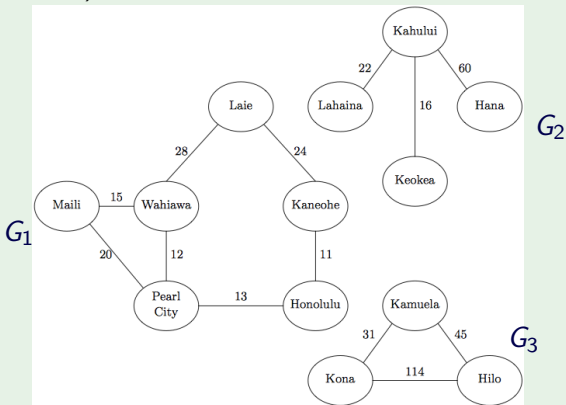


$\chi(G_i)? \kappa(G_i)?$

# Exercise

## Exercise

9.10.1 (Ullmann)



?

# Exercise

## Exercise

9.10.3 (Ullmann) Let  $G = (V, E)$  be an undirected graph. What inequalities must hold between

- the maximal  $\deg(v)$  for  $v \in V$
- $\chi(G)$
- $\kappa(G)$

# Exercise

## Exercise

9.10.3 (Ullmann) Let  $G = (V, E)$  be an undirected graph. What inequalities must hold between

- the maximal  $\deg(v)$  for  $v \in V$
- $\chi(G)$
- $\kappa(G)$

?

# Planar Graphs

## Definition

A graph is **planar** if it can be embedded in a plane without its edges intersecting.

## Theorem

*If the graph is planar it can be embedded (without self-intersections) in a plane so that all its edges are straight lines.*

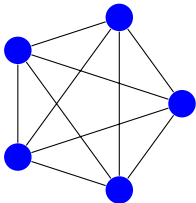
## NB

*This notion and its related algorithms are extremely important to VLSI and visualizing data.*

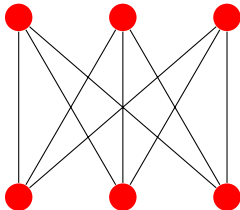


## Two minimal nonplanar graphs

$K_5$  :



$K_{3,3}$  :



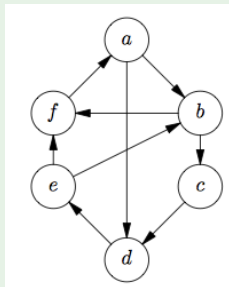
Try out  $K_5$

Try out  $K_{3,3}$

# Exercise

## Exercise

9.10.2 (Ullmann)



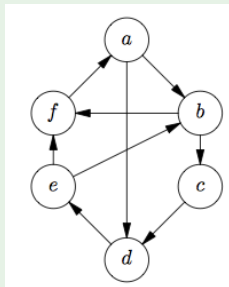
Is (the undirected version of) this graph planar?

Try it out

# Exercise

## Exercise

9.10.2 (Ullmann)



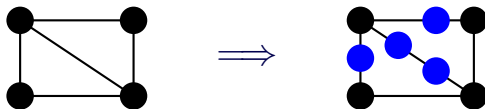
Is (the undirected version of) this graph planar? ?

Try it out

## Theorem

*If graph  $G$  contains, as a subgraph, a nonplanar graph, then  $G$  itself is nonplanar.*

For a graph, *edge subdivision* means to introduce some new vertices, all of degree 2, by placing them on existing edges.



We call such a derived graph a *subdivision* of the original one.

## Theorem

*If a graph is nonplanar then it must contain a subdivision of  $K_5$  or  $K_{3,3}$ .*

### Theorem

$K_n$  for  $n \geq 5$  is nonplanar.

### Proof.

It contains  $K_5$ : choose any five vertices in  $K_n$  and consider the subgraph they define. □

### Theorem

$K_{m,n}$  is nonplanar when  $m \geq 3$  and  $n \geq 3$ .

### Proof.

They contain  $K_{3,3}$  — choose any three vertices in each of two vertex parts and consider the subgraph they define. □

## Question

*Are all  $K_{m,1}$  planar?*

### Question

*Are all  $K_{m,1}$  planar?*

### Answer

*Yes, they are trees of two levels — the root and  $m$  leaves.*

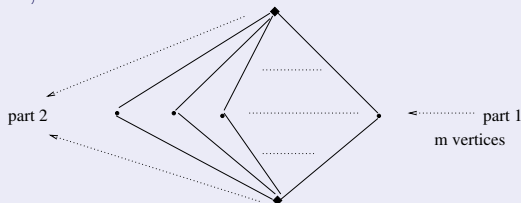
## Question

Are all  $K_{m,2}$  planar?

## Answer

Yes; they can be represented by “glueing” together two such trees at the leaves.

Sketching  $K_{m,2}$

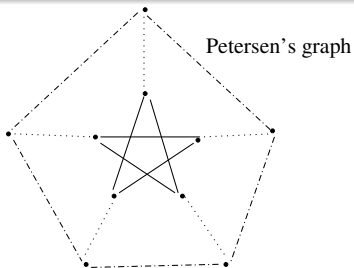


Also, among the  $k$ -partite graphs, planar are  $K_{2,2,2}$  and  $K_{1,1,m}$ . The latter can be depicted by drawing one extra edge in  $K_{2,m}$ , connecting the top and bottom vertices.



## NB

*Finding a 'basic' nonplanar obstruction is not always simple*



It contains a subdivision of both  $K_{3,3}$  and  $K_5$  while it does not directly contain either of them.