

Foundations of Computer Science

Lecture 11

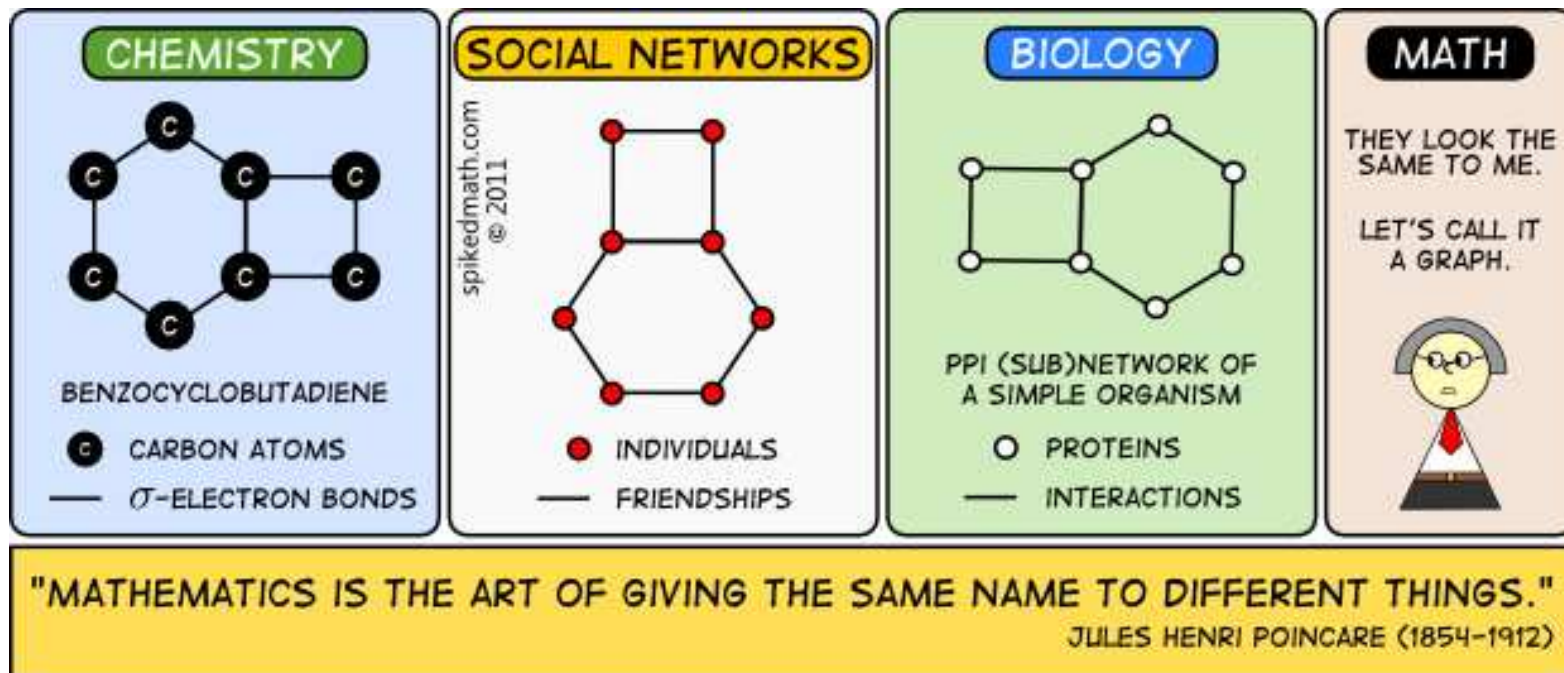
Graphs

Definition and Properties. Equivalence of Graphs.

Degree Sequences and The Handshaking Theorem.

Planar Graphs.

Different Types of Graphs: Multigraph, Weighted, Directed.



- ① Division, quotient and remainder. Properties of divisibility.
- ② Greatest common divisor and Euclid's algorithm.
 - ▶ Bezout's Identity: The GCD is the smallest linear combination.
 - ▶ Euclid's Lemma: $p|q_1 \cdots q_\ell \rightarrow p$ is one of the q_i .
- ③ Fundamental Theorem of Arithmetic Part II: Uniqueness of prime factorization.
- ④ Modular arithmetic
 - ▶ **Pop Quiz:** What is the last digit of 29^{29} .
- ⑤ RSA

Today: Graphs

- 1 Graph basics and notation
 - Equivalent graphs: isomorphism
- 2 Degree sequence
 - Handshaking Theorem
- 3 Trees
- 4 Planar graphs
- 5 Other types of graphs: multigraph, weighted, directed
- 6 Problem solving with Graphs

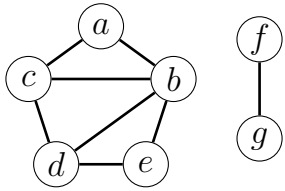
Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
 connectivity (e.g. cities linked by highways)
 conflicts (e.g. radio-stations with listener overlap)

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
 connectivity (e.g. cities linked by highways)
 conflicts (e.g. radio-stations with listener overlap)

Graph G



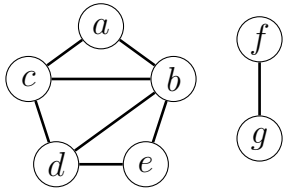
Vertices (aka nodes): a, b, c, d, e, f, g

$$V = \{a, b, c, d, e, f, g\}.$$

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
connectivity (e.g. cities linked by highways)
conflicts (e.g. radio-stations with listener overlap)

Graph G



Vertices (aka nodes): a, b, c, d, e, f, g

Edges: $(a,b), (a,c), (a,d), (b,c), (b,d), (b,e), (c,d), (d,e), (f,g)$

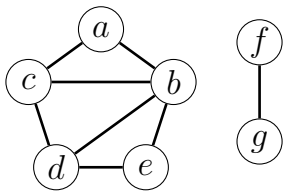
$V = \{a, b, c, d, e, f, g\}.$

$E = \{(a,b), (a,c), (a,d), (b,c), (b,d), (b,e), (c,d), (d,e), (f,g)\}.$

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
connectivity (e.g. cities linked by highways)
conflicts (e.g. radio-stations with listener overlap)

Graph G



Vertices (aka nodes): a, b, c, d, e, f, g

Edges: $(a,b), (a,c), (b,c), (b,d), (b,e), (c,d), (c,e), (f,g)$

Degree: Number of relationships

$$V = \{a, b, c, d, e, f, g\}.$$

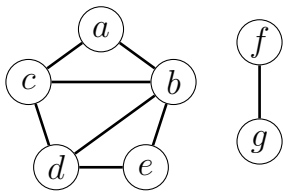
$$E = \left\{ (a, b), (a, c), (b, c), (b, d), (b, e), (c, d), (c, e), (f, g) \right\}.$$

$$e.g., \text{degree}(b) = 4.$$

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
connectivity (e.g. cities linked by highways)
conflicts (e.g. radio-stations with listener overlap)

Graph G



Vertices (aka nodes): $a\ b\ c\ d\ e\ f\ g$

Edges: $\begin{matrix} a & a & b & b & b & c & d & f \\ | & | & | & | & | & | & | & | \\ b & c & c & d & e & d & e & g \end{matrix}$

Degree: Number of relationships

Path: $a - c - b - e - d - b$

$$V = \{a, b, c, d, e, f, g\}.$$

$$E = \left\{ (a, b), (a, c), (b, c), (b, d), \right. \\ \left. (b, e), (c, d), (d, e), (f, g) \right\}.$$

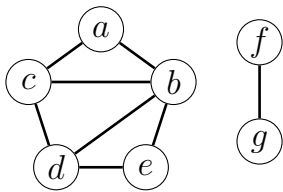
e.g., $\text{degree}(b) = 4.$

$$p = acbedb.$$

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
connectivity (e.g. cities linked by highways)
conflicts (e.g. radio-stations with listener overlap)

Graph G



Vertices (aka nodes): a, b, c, d, e, f, g

Edges: $(a,b), (a,c), (b,c), (b,d), (c,d), (d,e), (f,g)$

Degree: Number of relationships

Path: $a-c-b-e-d-b$

$$V = \{a, b, c, d, e, f, g\}.$$

$$E = \left\{ (a, b), (a, c), (b, c), (b, d), (b, e), (c, d), (d, e), (f, g) \right\}.$$

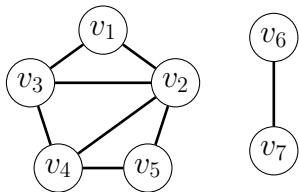
e.g., $\text{degree}(b) = 4$.

$p = acbedb$.

Graph Isomorphism. Relabeling the nodes in G to v_1, \dots, v_7 .

$a \rightarrow v_1,$
 $b \rightarrow v_2,$
 $c \rightarrow v_3,$
 $d \rightarrow v_4,$
 $e \rightarrow v_5,$
 $f \rightarrow v_6,$
 $g \rightarrow v_7$

Relabeling of Graph G

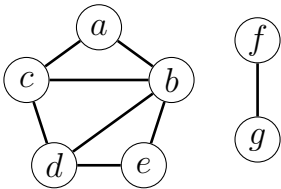


$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$
$$E = \left\{ (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4), (v_2, v_5), (v_3, v_4), (v_4, v_5), (v_6, v_7) \right\}.$$

Graph Basics and Notation

Graphs model relationships: friendships (e.g. social networks)
connectivity (e.g. cities linked by highways)
conflicts (e.g. radio-stations with listener overlap)

Graph G



Vertices (aka nodes): $\textcircled{a} \textcircled{b} \textcircled{c} \textcircled{d} \textcircled{e} \textcircled{f} \textcircled{g}$

Edges: $\begin{array}{ccccccc} \textcircled{a} & \textcircled{a} & \textcircled{b} & \textcircled{b} & \textcircled{b} & \textcircled{c} & \textcircled{d} & \textcircled{f} \\ | & | & | & | & | & | & | & | \\ \textcircled{b} & \textcircled{c} & \textcircled{c} & \textcircled{d} & \textcircled{e} & \textcircled{d} & \textcircled{e} & \textcircled{g} \end{array}$

Degree: Number of relationships

Path: $\textcircled{a} - \textcircled{c} - \textcircled{b} - \textcircled{e} - \textcircled{d} - \textcircled{b}$

$$V = \{a, b, c, d, e, f, g\}.$$
$$E = \left\{ (a, b), (a, c), (b, c), (b, d), \right. \\ \left. (b, e), (c, d), (d, e), (f, g) \right\}.$$

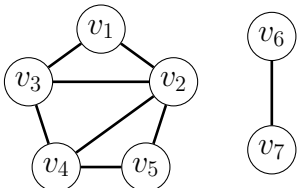
e.g., $\text{degree}(b) = 4$.

$$p = acbedb.$$

Graph Isomorphism. Relabeling the nodes in G to v_1, \dots, v_7 .

$$\begin{array}{l} a \rightarrow v_1, \\ b \rightarrow v_2, \\ c \rightarrow v_3, \\ d \rightarrow v_4, \\ e \rightarrow v_5, \\ f \rightarrow v_6, \\ g \rightarrow v_7 \end{array}$$

Relabeling of Graph G



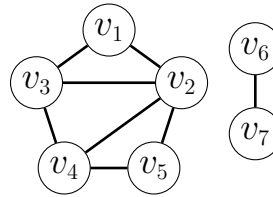
$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$
$$E = \left\{ (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4), \right. \\ \left. (v_2, v_5), (v_3, v_4), (v_4, v_5), (v_6, v_7) \right\}.$$

If two graphs can be relabeled with v_1, \dots, v_n , giving the *same* edge set, they are equivalent – *isomorphic*.

Practice. Pop Quiz 11.1; Exercise 11.2.

Paths and Connectivity

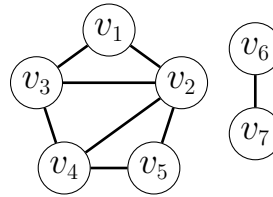
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$

Paths and Connectivity

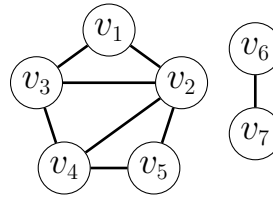
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.

Paths and Connectivity

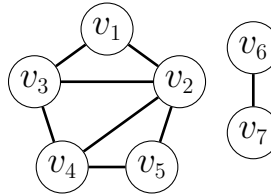
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.
- The length of a path is the number of edges traversed (5).

Paths and Connectivity

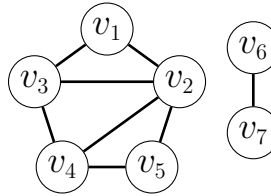
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.
- The length of a path is the number of edges traversed (5).
- *Cycle*: path that starts and ends at a vertex without repeating any edge: $v_1 v_2 v_3 v_1$

Paths and Connectivity

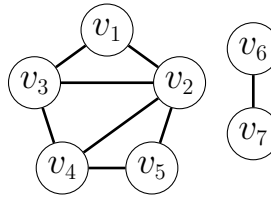
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.
- The length of a path is the number of edges traversed (5).
- *Cycle*: path that starts and ends at a vertex without repeating any edge: $v_1 v_2 v_3 v_1$
- v_1 and v_6 are not connected by a path.

Paths and Connectivity

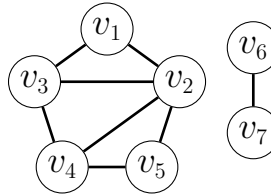
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.
- The length of a path is the number of edges traversed (5).
- *Cycle*: path that starts and ends at a vertex without repeating any edge: $v_1 v_2 v_3 v_1$
- v_1 and v_6 are not connected by a path.
- The graph G is *not* connected (*every* pair of vertices must be connected by a path).

Paths and Connectivity

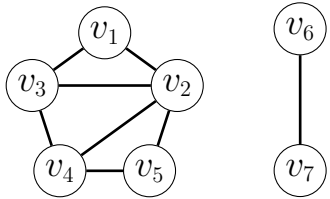
Graph, G



- A *path* from v_1 to v_2 is a sequence of vertices (start is v_1 and end is v_2): $v_1 v_3 v_2 v_5 v_4 v_2$
- There is an edge in the graph between consecutive vertices in the path.
 v_1 and v_2 are *connected*.
- The length of a path is the number of edges traversed (5).
- *Cycle*: path that starts and ends at a vertex without repeating any edge: $v_1 v_2 v_3 v_1$
- v_1 and v_6 are not connected by a path.
- The graph G is *not* connected (*every* pair of vertices must be connected by a path).
- How can we make G connected?

Graph Representation

Graph

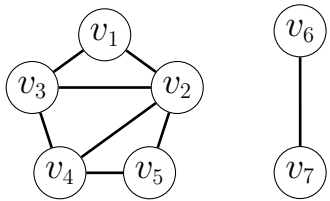


Adjacency List

v_1 : v_2, v_3
 v_2 : v_1, v_3, v_4, v_5
 v_3 : v_1, v_2, v_4
 v_4 : v_2, v_3, v_5
 v_5 : v_2, v_4
 v_6 : v_7
 v_7 : v_6

Graph Representation

Graph



Adjacency List

v_1 : v_2, v_3
 v_2 : v_1, v_3, v_4, v_5
 v_3 : v_1, v_2, v_4
 v_4 : v_2, v_3, v_5
 v_5 : v_2, v_4
 v_6 : v_7
 v_7 : v_6

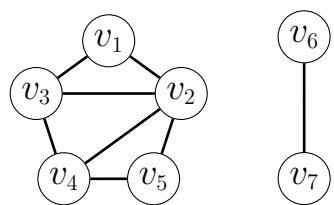
Adjacency Matrix

$$\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} \begin{array}{ccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \\ \left(\begin{array}{ccccccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array} \right)$$

More wasted memory; faster algorithms.

Graph Representation

Graph



Adjacency List

v_1 : v_2, v_3
 v_2 : v_1, v_3, v_4, v_5
 v_3 : v_1, v_2, v_4
 v_4 : v_2, v_3, v_5
 v_5 : v_2, v_4
 v_6 : v_7
 v_7 : v_6

Adjacency Matrix

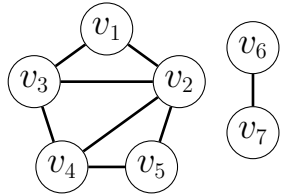
	v_1	v_2	v_3	v_4	v_5	v_6	v_7
v_1	0	1	1	0	0	0	0
v_2	1	0	1	1	1	0	0
v_3	1	1	0	1	0	0	0
v_4	0	1	1	0	1	0	0
v_5	0	1	0	1	0	0	0
v_6	0	0	0	0	0	0	1
v_7	0	0	0	0	0	1	0

More wasted memory; faster algorithms.

Small redundancy: every edge is “represented” twice.

Degree Sequence

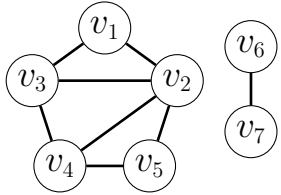
Graph



degree δ_i = number of v_i 's neighbors
 $= \sum_{j=1}^n A_{ij}.$

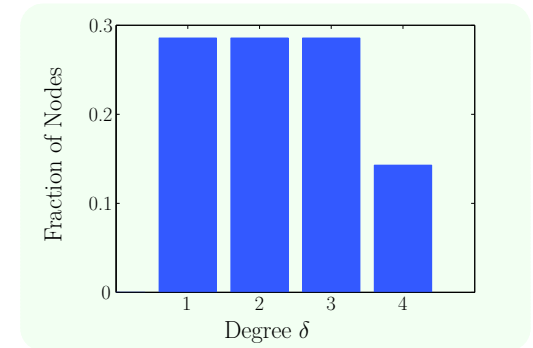
Degree Sequence

Graph



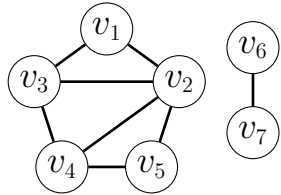
degree δ_i = number of v_i 's neighbors
$$= \sum_{j=1}^n A_{ij}.$$

$$\boldsymbol{\delta} = \begin{bmatrix} & v_2 & v_3 & v_4 & v_1 & v_5 & v_6 & v_7 \\ 4 & 3 & 3 & 2 & 2 & 1 & 1 \end{bmatrix}$$



Degree Sequence

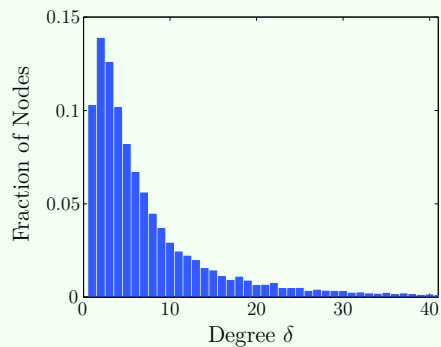
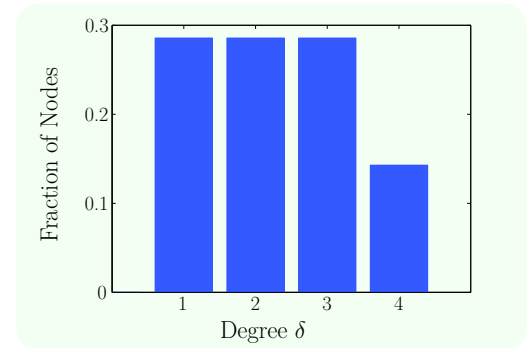
Graph



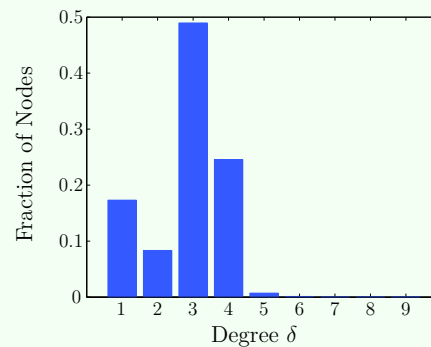
degree δ_i = number of v_i 's neighbors

$$= \sum_{j=1}^n A_{ij}.$$

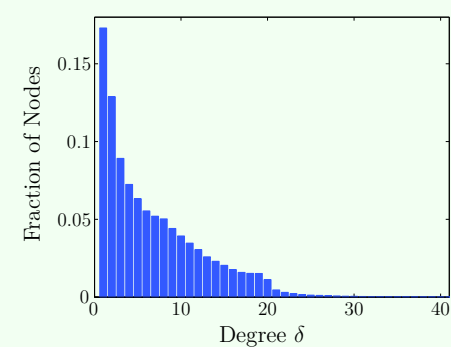
$$\boldsymbol{\delta} = \begin{bmatrix} & v_2 & v_3 & v_4 & v_1 & v_5 & v_6 & v_7 \\ 4 & 3 & 3 & 2 & 2 & 1 & 1 \end{bmatrix}$$



Co-author network



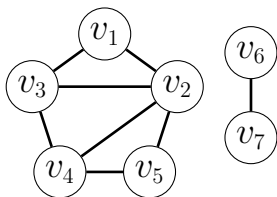
PA road network



Web graph

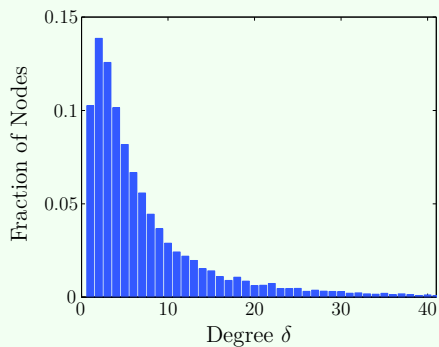
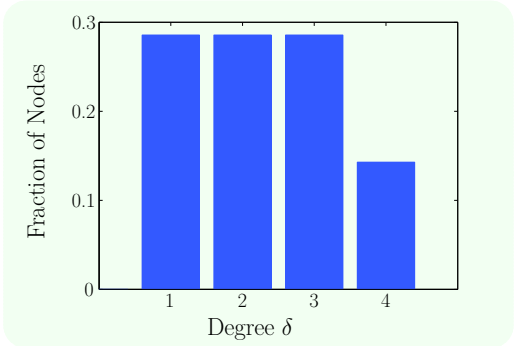
Degree Sequence

Graph

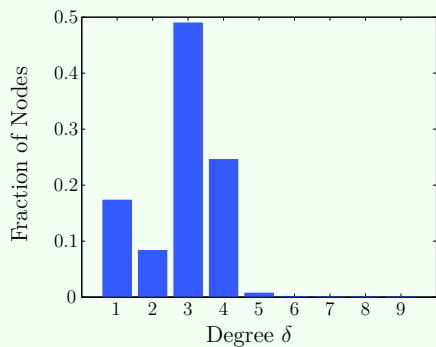


degree δ_i = number of v_i 's neighbors
$$= \sum_{j=1}^n A_{ij}.$$

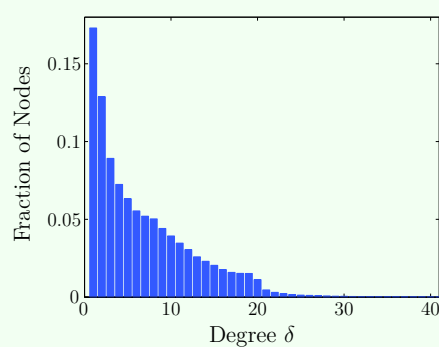
$$\boldsymbol{\delta} = \begin{bmatrix} & v_2 & v_3 & v_4 & v_1 & v_5 & v_6 & v_7 \\ 4 & 3 & 3 & 2 & 2 & 1 & 1 \end{bmatrix}$$



Co-author network



PA road network



Web graph

Complete, K_5



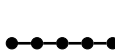
$[4, 4, 4, 4, 4]$

Bipartite, $K_{3,2}$



$[3, 3, 2, 2, 2]$

Line, L_5



$[2, 2, 2, 1, 1]$

Cycle, C_5



$[2, 2, 2, 2, 2]$

Star, S_6



$[5, 1, 1, 1, 1, 1]$

Wheel, W_6



$[5, 3, 3, 3, 3, 3]$

Handshaking Theorem

Pop Quiz. Construct a graph with degree sequence $\delta = [3, 3, 3, 2, 1, 1]$.

Handshaking Theorem

Pop Quiz. Construct a graph with degree sequence $\delta = [3, 3, 3, 2, 1, 1]$.

Theorem. Handshaking Theorem

For any graph the sum of vertex-degrees equals twice the number of edges, $\sum_{i=1}^n \delta_i = 2|E|$.

Handshaking Theorem

Pop Quiz. Construct a graph with degree sequence $\delta = [3, 3, 3, 2, 1, 1]$.

Theorem. Handshaking Theorem

For any graph the sum of vertex-degrees equals twice the number of edges, $\sum_{i=1}^n \delta_i = 2|E|$.

Proof. Every edge contributes 2 to the sum of degrees. (Why?)

Handshaking Theorem

Pop Quiz. Construct a graph with degree sequence $\delta = [3, 3, 3, 2, 1, 1]$.

Theorem. Handshaking Theorem

For any graph the sum of vertex-degrees equals twice the number of edges, $\sum_{i=1}^n \delta_i = 2|E|$.

Proof. Every edge contributes 2 to the sum of degrees. (Why?)
If there are $|E|$ edges, their contribution to the sum of degrees is $2|E|$. ■

Exercise. Give a formal proof by induction on the number of edges in the graph.

Handshaking Theorem

Pop Quiz. Construct a graph with degree sequence $\delta = [3, 3, 3, 2, 1, 1]$.

Theorem. Handshaking Theorem

For any graph the sum of vertex-degrees equals twice the number of edges, $\sum_{i=1}^n \delta_i = 2|E|$.

Proof. Every edge contributes 2 to the sum of degrees. (Why?)
If there are $|E|$ edges, their contribution to the sum of degrees is $2|E|$. ■

Exercise. Give a formal proof by induction on the number of edges in the graph.

Pop Quiz (Answer). Can't be done: sum of degrees is $3 + 3 + 3 + 2 + 1 + 1 = 13$ (odd).

Exercise. At a party a person is odd if they shake hands with an odd number of people. Show that the number of odd people is even.

Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

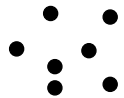
Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

step 0

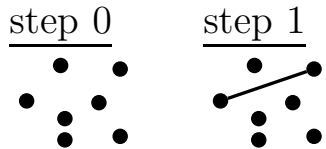


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

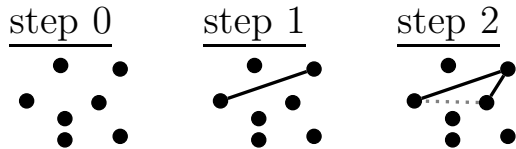


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

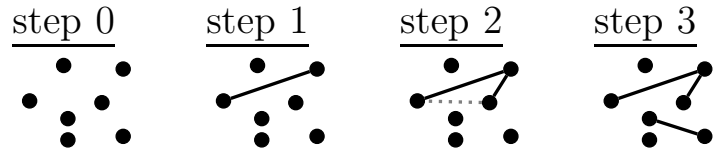


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

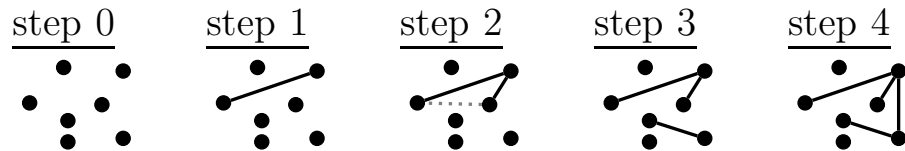


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

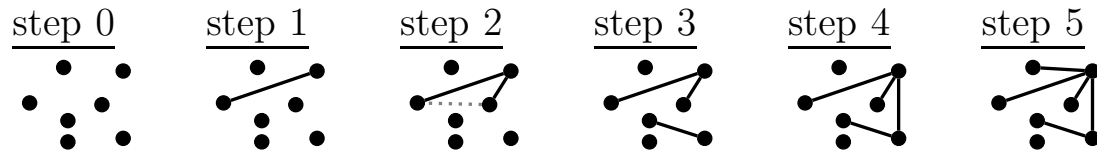


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

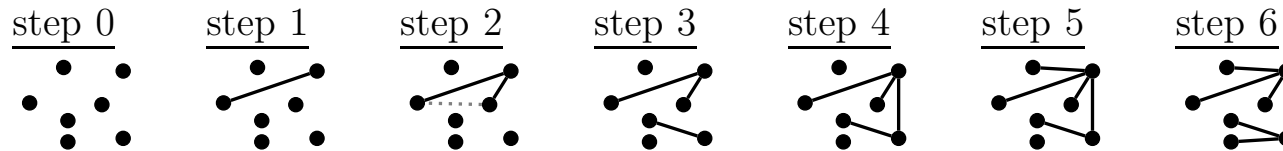


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.

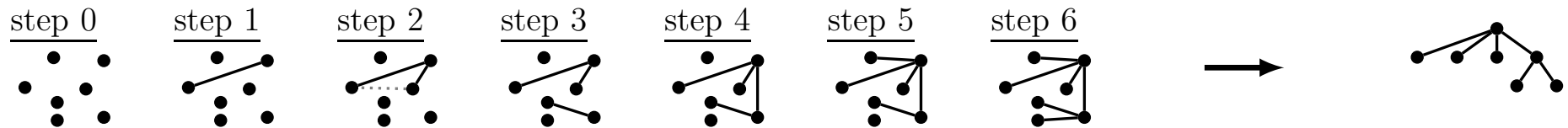


Trees (More General than RBTs)

Definition: General Tree.

A tree is a *connected* graph with no cycles.

Building a tree, one edge at a time.



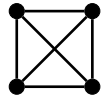
Exercise 11.6. Every tree with n vertices has $n - 1$ edges.

(We proved this for RBTs.)

Planar Graphs

A graph is planar if you can draw it without edge crossings.

Complete graph K_4 :

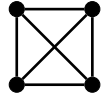


non planar drawing

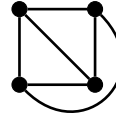
Planar Graphs

A graph is planar if you can draw it without edge crossings.

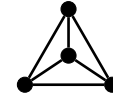
Complete graph K_4 :



non planar drawing



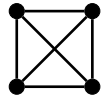
planar drawings $\rightarrow K_4$ is planar



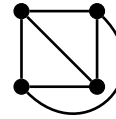
Planar Graphs

A graph is planar if you can draw it without edge crossings.

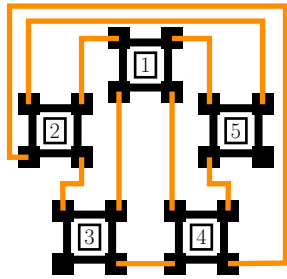
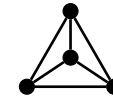
Complete graph K_4 :



non planar drawing



planar drawings $\rightarrow K_4$ is planar

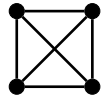


Chip design: CPUs
must be connected
without wire-crossings.

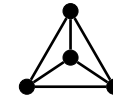
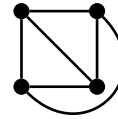
Planar Graphs

A graph is planar if you can draw it without edge crossings.

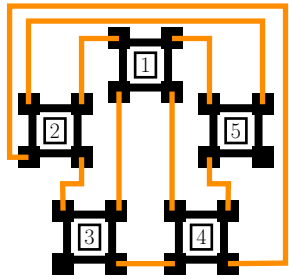
Complete graph K_4 :



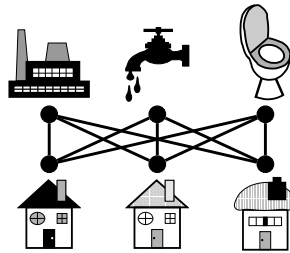
non planar drawing



planar drawings $\rightarrow K_4$ is planar



Chip design: CPUs must be connected without wire-crossings.

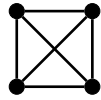


Town planing: connect utilities to homes without pipe-crossings.

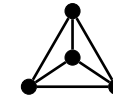
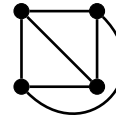
Planar Graphs

A graph is planar if you can draw it without edge crossings.

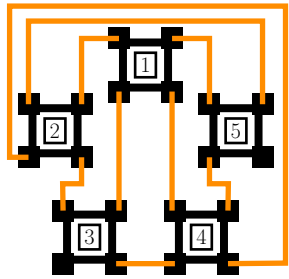
Complete graph K_4 :



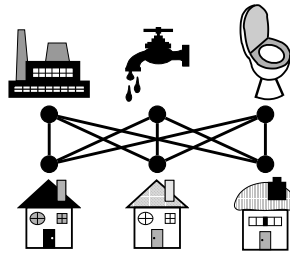
non planar drawing



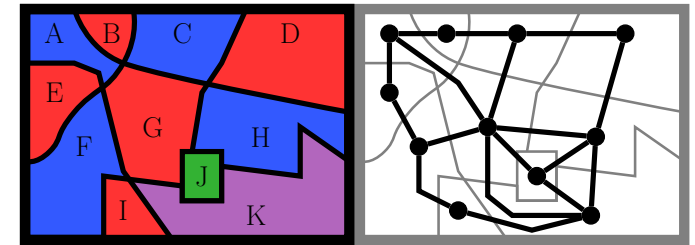
planar drawings $\rightarrow K_4$ is planar



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.

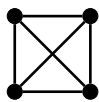


Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

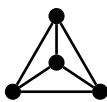
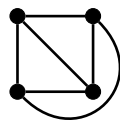
Planar Graphs

A graph is planar if you can draw it without edge crossings.

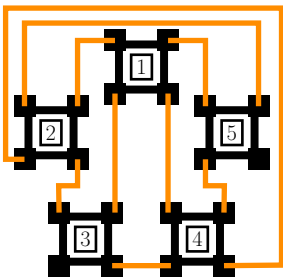
Complete graph K_4 :



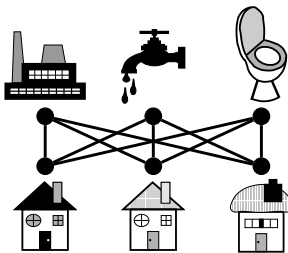
non planar drawing



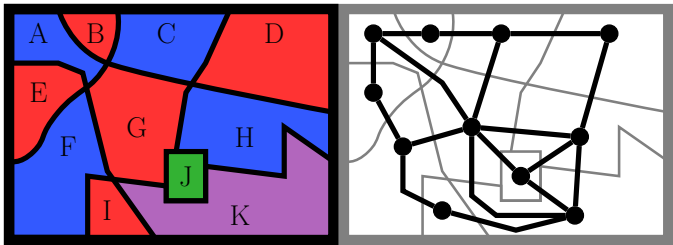
planar drawings $\rightarrow K_4$ is planar



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.



Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

Exercise 11.7. Euler's Invariant Characteristic: $F + V - E = 2$.

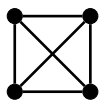
(Faces, F : outer region or region enclosed by a cycle.)

	V	E	F	$F + V - E$
planar K_4	4	6	4	$4 + 4 - 6 = 2$ ✓

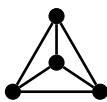
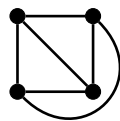
Planar Graphs

A graph is planar if you can draw it without edge crossings.

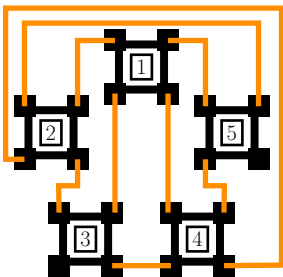
Complete graph K_4 :



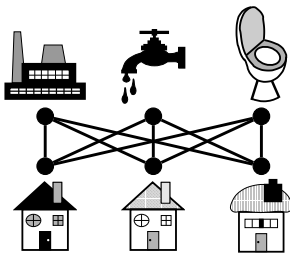
non planar drawing



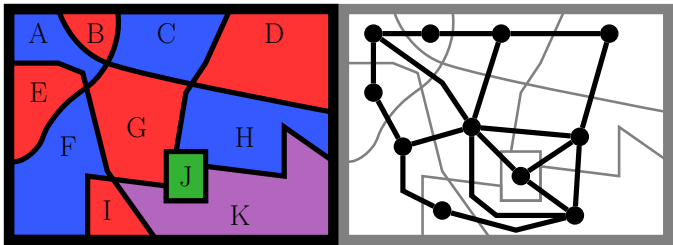
planar drawings $\rightarrow K_4$ is planar



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.



Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

Exercise 11.7. Euler's Invariant Characteristic: $F + V - E = 2$.

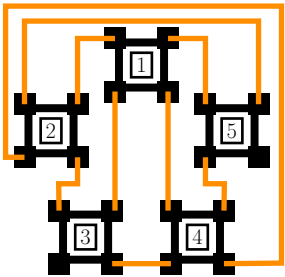
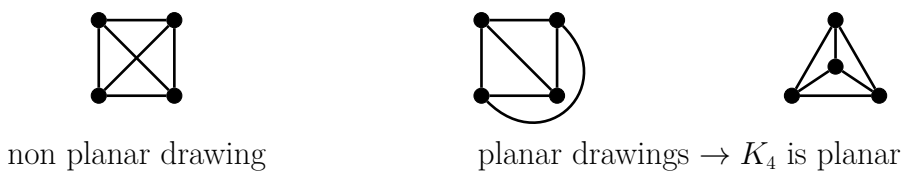
(Faces, F : outer region or region enclosed by a cycle.)

	V	E	F	$F + V - E$
planar K_4	4	6	4	$4 + 4 - 6 = 2$ ✓
planar map	11	17	8	$8 + 11 - 17 = 2$ ✓

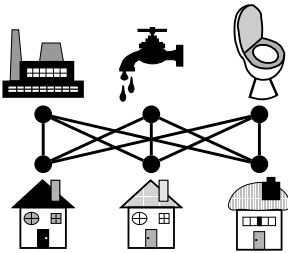
Planar Graphs

A graph is planar if you can draw it without edge crossings.

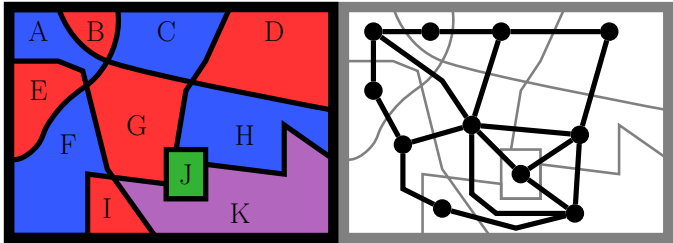
Complete graph K_4 :



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.



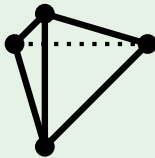
Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

Exercise 11.7. Euler's Invariant Characteristic: $F + V - E = 2$.

(Faces, F : outer region or region enclosed by a cycle.)

	V	E	F	$F + V - E$
planar K_4	4	6	4	$4 + 4 - 6 = 2$ ✓
planar map	11	17	8	$8 + 11 - 17 = 2$ ✓
pyramid	4	6	4	$4 + 4 - 6 = 2$ ✓

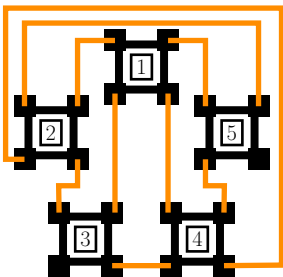
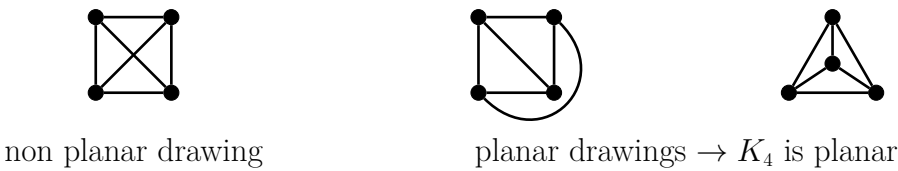
Pyramid



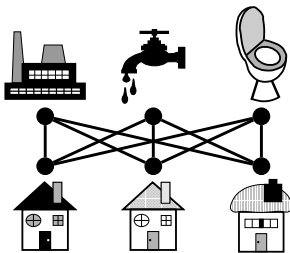
Planar Graphs

A graph is planar if you can draw it without edge crossings.

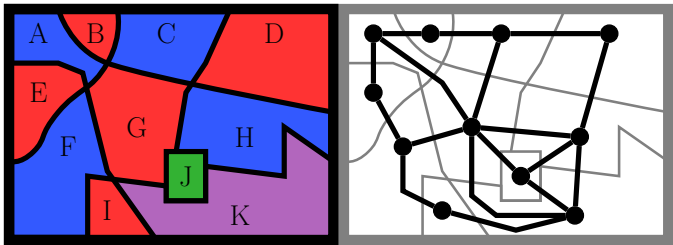
Complete graph K_4 :



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.



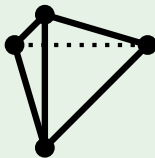
Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

Exercise 11.7. Euler's Invariant Characteristic: $F + V - E = 2$.

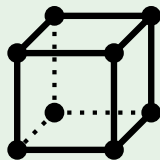
(Faces, F : outer region or region enclosed by a cycle.)

	V	E	F	$F + V - E$
planar K_4	4	6	4	$4 + 4 - 6 = 2$ ✓
planar map	11	17	8	$8 + 11 - 17 = 2$ ✓
pyramid	4	6	4	$4 + 4 - 6 = 2$ ✓
cube	8	12	6	$6 + 8 - 12 = 2$ ✓

Pyramid



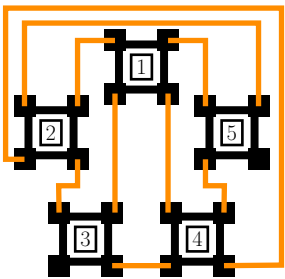
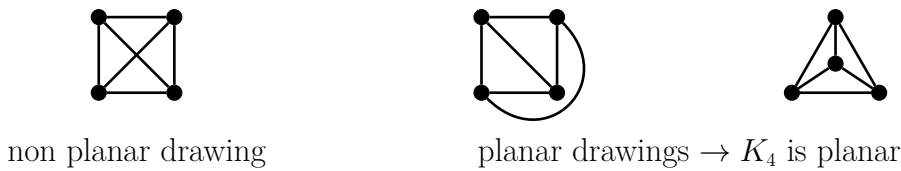
Cube



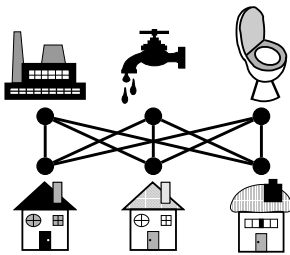
Planar Graphs

A graph is planar if you can draw it without edge crossings.

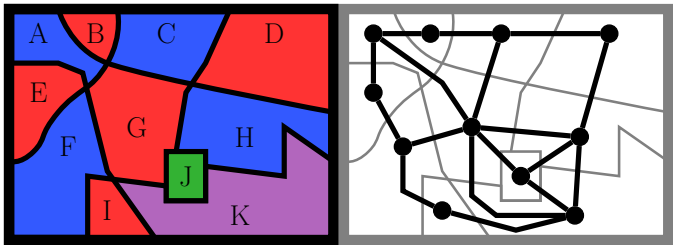
Complete graph K_4 :



Chip design: CPUs must be connected without wire-crossings.



Town planing: connect utilities to homes without pipe-crossings.



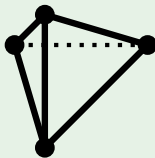
Map coloring: adjacent countries sharing a border must have different colors. The map corresponds to a planar graph.

Exercise 11.7. Euler's Invariant Characteristic: $F + V - E = 2$.

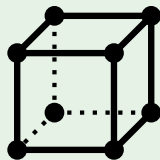
(Faces, F : outer region or region enclosed by a cycle.)

	V	E	F	$F + V - E$
planar K_4	4	6	4	$4 + 4 - 6 = 2$ ✓
planar map	11	17	8	$8 + 11 - 17 = 2$ ✓
pyramid	4	6	4	$4 + 4 - 6 = 2$ ✓
cube	8	12	6	$6 + 8 - 12 = 2$ ✓
octohedron	6	12	8	$8 + 6 - 12 = 2$ ✓

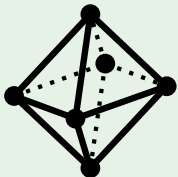
Pyramid



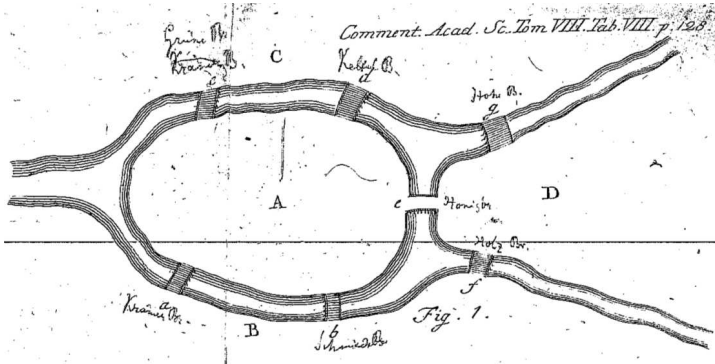
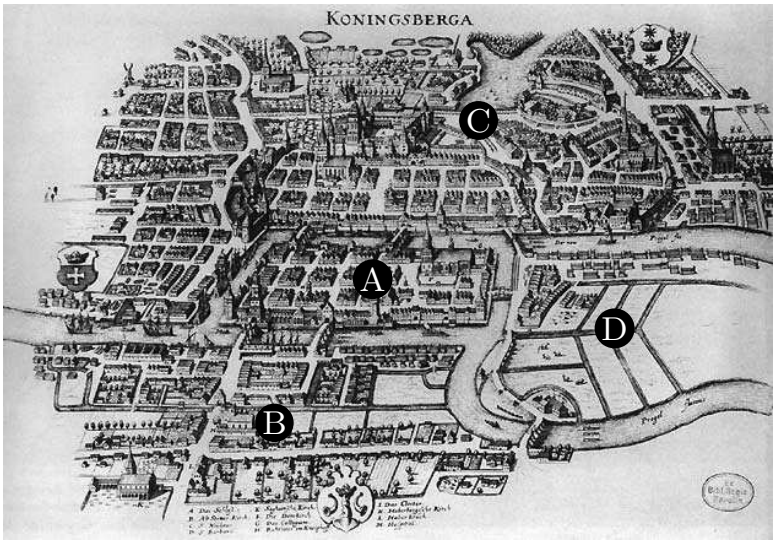
Cube



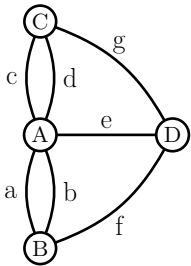
Octohedron



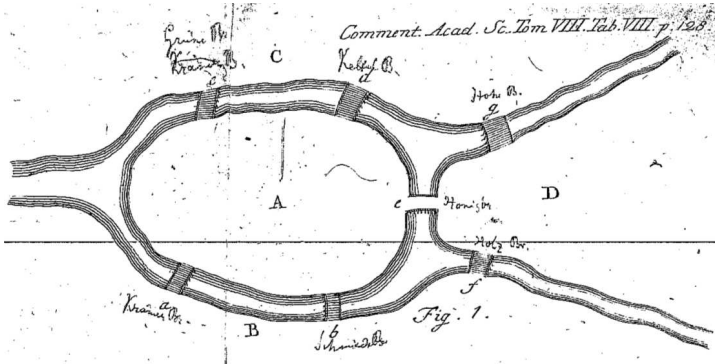
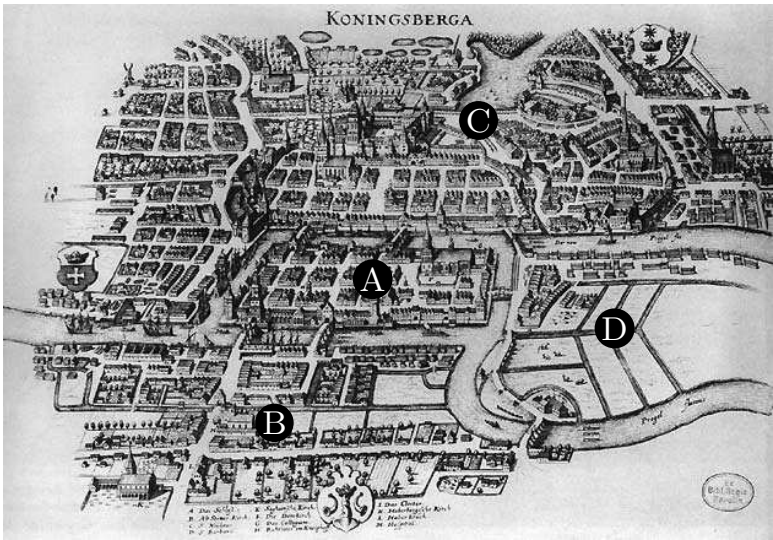
Other Types of Graphs: Multigraph, Weighted, Directed



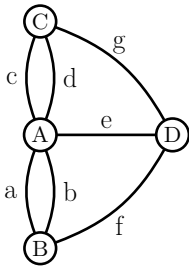
Euler's Multigraph



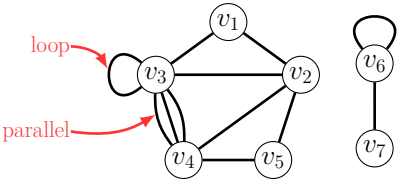
Other Types of Graphs: Multigraph, Weighted, Directed



Euler's Multigraph



Multigraph (NOT simple)

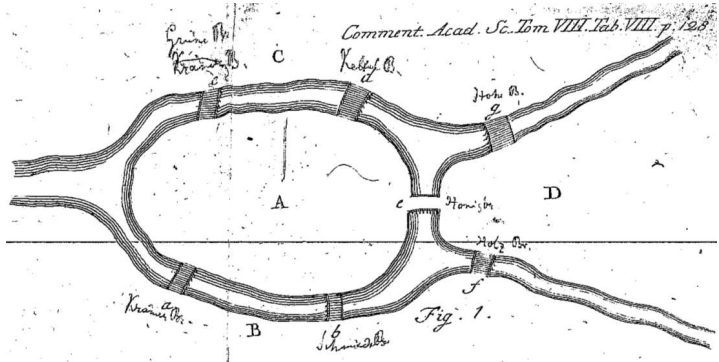
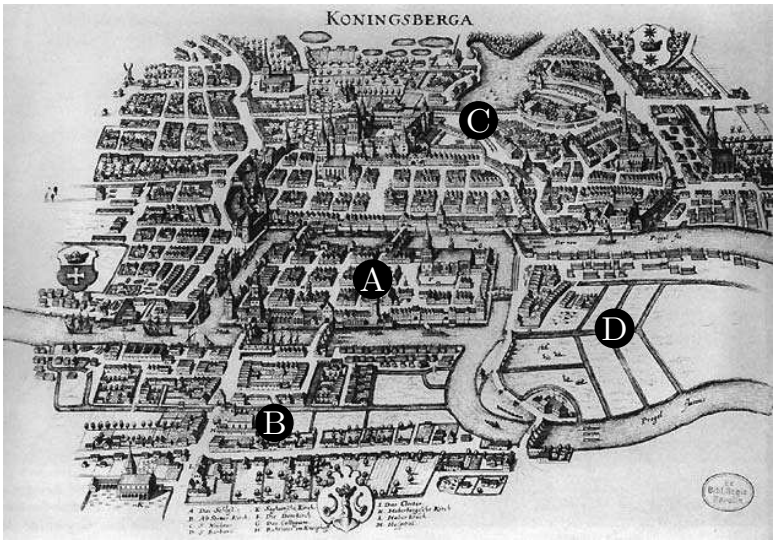


$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$

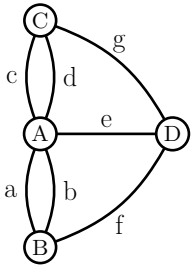
$$E = \left\{ (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4), (v_2, v_5), (v_3, v_4), (v_3, v_4), (v_3, v_4), (v_4, v_5), (v_6, v_7), (v_3, v_3), (v_6, v_6) \right\}$$

Handshaking Theorem still valid.

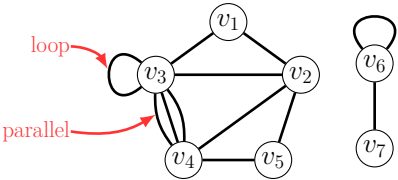
Other Types of Graphs: Multigraph, Weighted, Directed



Euler's Multigraph



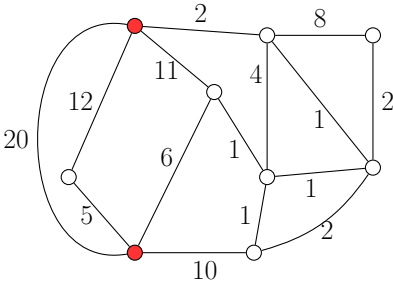
Multigraph (NOT simple)



$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$
$$E = \left\{ (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4), (v_2, v_5), (v_3, v_4), (v_3, v_4), (v_3, v_4), (v_4, v_5), (v_6, v_7), (v_3, v_3), (v_6, v_6) \right\}$$

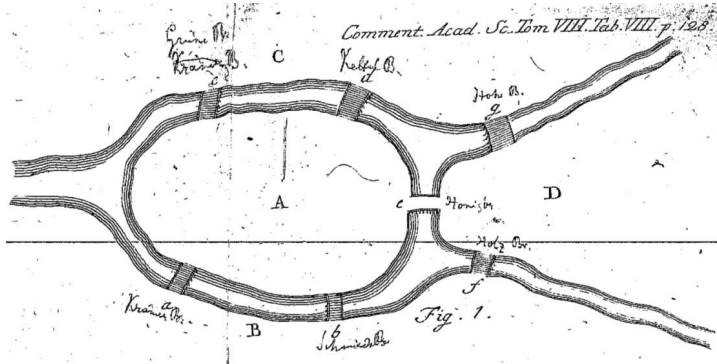
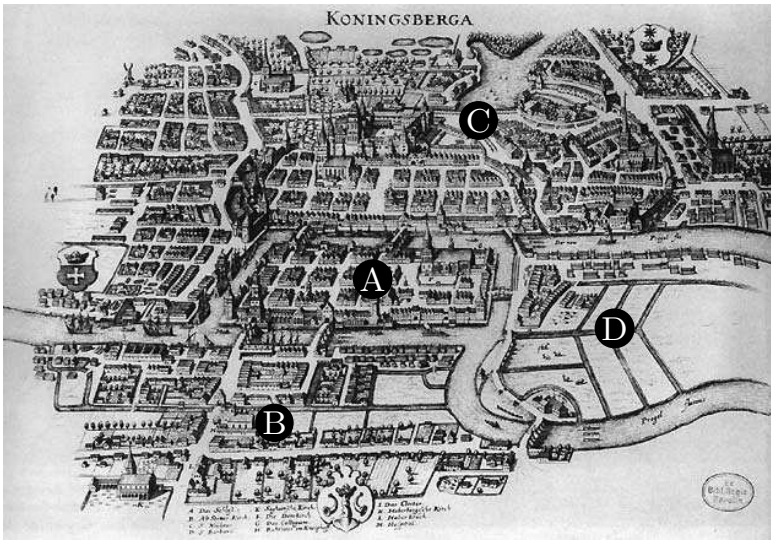
Handshaking Theorem still valid.

Weighted

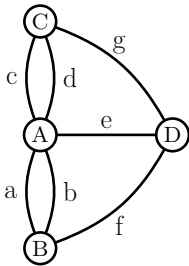


How quickly can one route between the red ISPs?

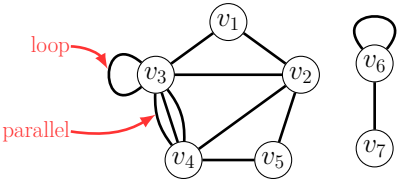
Other Types of Graphs: Multigraph, Weighted, Directed



Euler’s Multigraph



Multigraph (NOT simple)

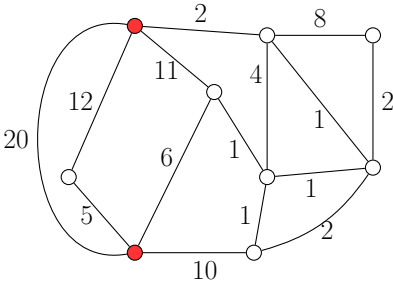


$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$

$$E = \left\{ (v_1, v_2), (v_1, v_3), (v_2, v_3), (v_2, v_4), (v_2, v_5), (v_3, v_4), (v_3, v_4), (v_3, v_4), (v_4, v_5), (v_6, v_7), (v_3, v_3), (v_6, v_6) \right\}$$

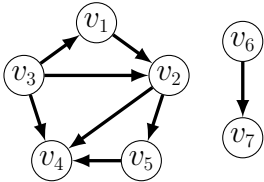
Handshaking Theorem still valid.

Weighted



How quickly can one route between the red ISPs?

Directed Graphs



$$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}.$$

$$E = \left\{ (v_1 \rightarrow v_2), (v_3 \rightarrow v_1), (v_3 \rightarrow v_2), (v_2 \rightarrow v_4), (v_2 \rightarrow v_5), (v_3 \rightarrow v_4), (v_5 \rightarrow v_4), (v_6 \rightarrow v_7) \right\}.$$

Ancestry graphs, tournaments, one-way streets, partially ordered sets (Example 11.6), ...

Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

On the right is elevation data in a park.

One unit of rain falls on each grid-square.

Water flows to a neighbor of lowest elevation (e.g. $17 \rightarrow 1$)

3	2	17	11	12
4	1	18	10	7
21	22	23	16	8
20	13	5	19	9
25	24	6	14	15

Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

On the right is elevation data in a park.

One unit of rain falls on each grid-square.

Water flows to a neighbor of lowest elevation (e.g. $17 \rightarrow 1$)

Where should we install drains and what should their capacities be?

3	2	17	11	12
4	1	18	10	7
21	22	23	16	8
20	13	5	19	9
25	24	6	14	15

Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

On the right is elevation data in a park.

One unit of rain falls on each grid-square.

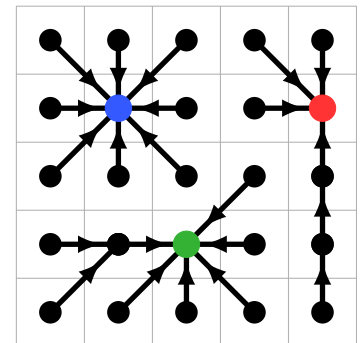
Water flows to a neighbor of lowest elevation (e.g. $17 \rightarrow 1$)

Where should we install drains and what should their capacities be?

3	2	17	11	12
4	1	18	10	7
21	22	23	16	8
20	13	5	19	9
25	24	6	14	15

Model the problem as a directed graph.

Directed edges indicate how water flows: three disjoint trees.



Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

On the right is elevation data in a park.

One unit of rain falls on each grid-square.

Water flows to a neighbor of lowest elevation (e.g. $17 \rightarrow 1$)

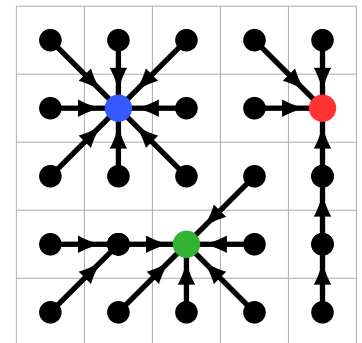
Where should we install drains and what should their capacities be?

3	2	17	11	12
4	1	18	10	7
21	22	23	16	8
20	13	5	19	9
25	24	6	14	15

Model the problem as a directed graph.

Directed edges indicate how water flows: three disjoint trees.

The red, green and blue vertices are “sinks” (no out-going arrow).



Problem Solving with Graphs

Graphs are everywhere because relationships are everywhere.

On the right is elevation data in a park.

One unit of rain falls on each grid-square.

Water flows to a neighbor of lowest elevation (e.g. $17 \rightarrow 1$)

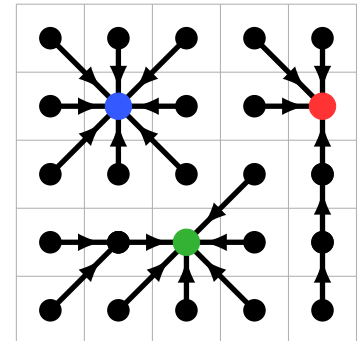
Where should we install drains and what should their capacities be?

3	2	17	11	12
4	1	18	10	7
21	22	23	16	8
20	13	5	19	9
25	24	6	14	15

Model the problem as a directed graph.

Directed edges indicate how water flows: three disjoint trees.

The red, green and blue vertices are “sinks” (no out-going arrow).



Place drains at the sinks.

Drain capacities: **blue**=9 units, **red**=7 units and **green**=9 units.

The solution pops out once we formulate the problem as a graph.