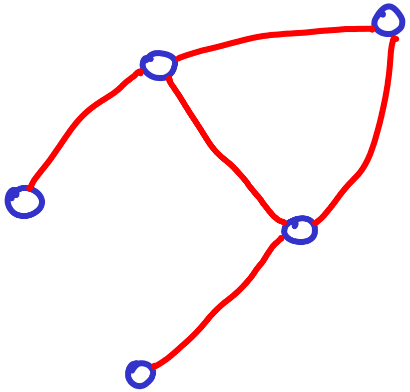
GRAPHS

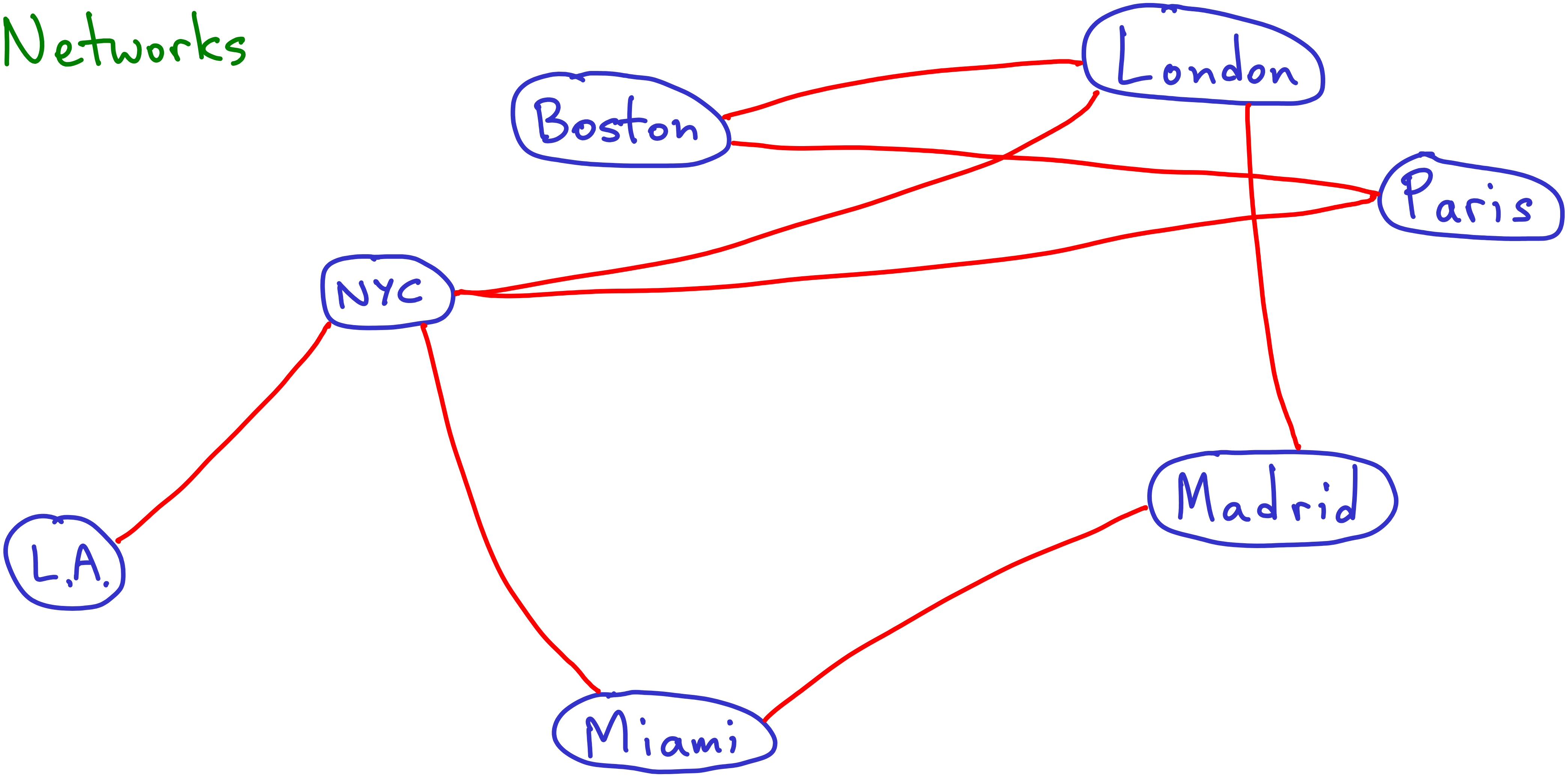
vertices and edges



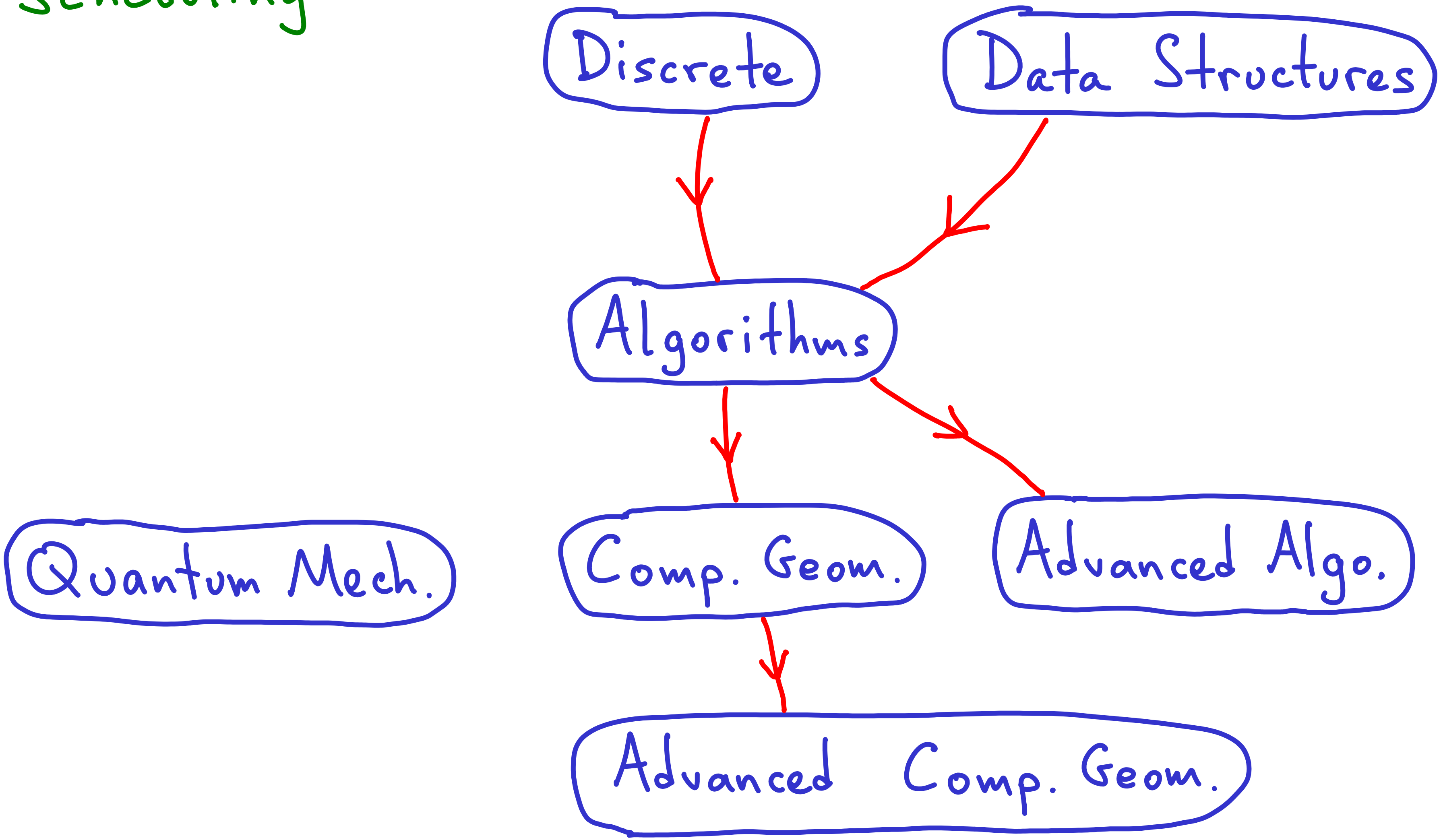
Terminology: one vertex • not "one vertice"

many vertices

Networks



Scheduling



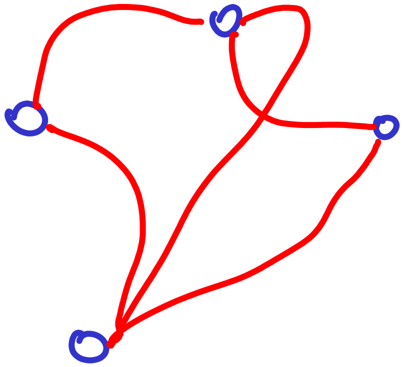
(Directed Graph)

Graphs can be "abstract" or "geometric" / "embedded"

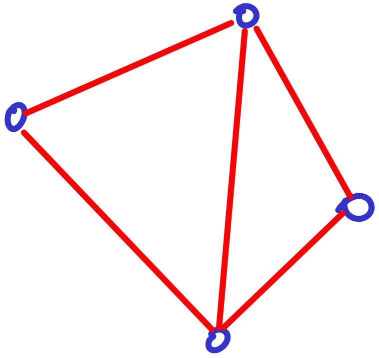
co-ordinates & drawings don't matter

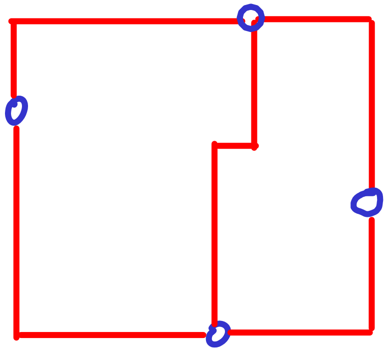
representing physical restrictions

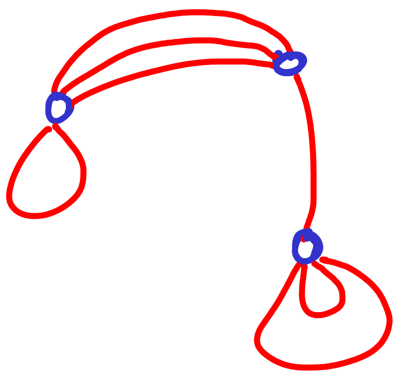
although sometimes it helps to draw & visualize



vs







Often it is assumed that there are no self-loops or multi-edges (duplicates)

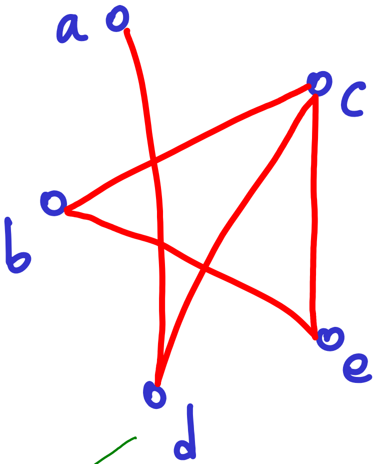
Assume this unless specified

$G = (V, E)$

$V \downarrow V(G)$ $\quad E \downarrow E(G)$

V : set of vertices

E : set of edges



$(\{a, b, c, d, e\}, \{ad, bc, be, ce, cd\})$

Terminology:

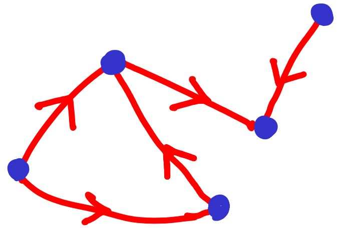
d is incident to two edges

Terminology:

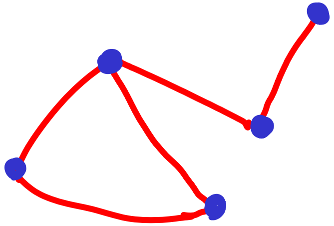
b & c
- are adjacent
- share an edge
- are neighbors

Some graph types

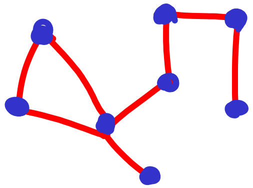
directed



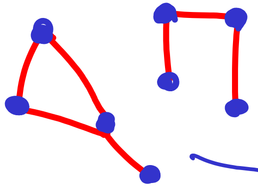
not directed



connected



not connected

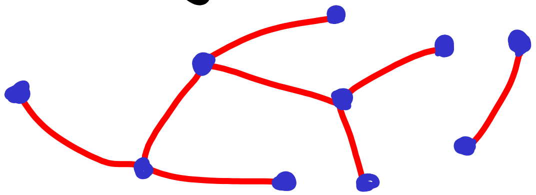


connected components

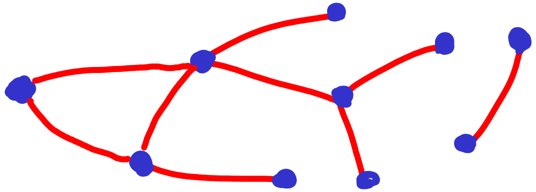
A connected acyclic graph is a tree

Forest of 2 trees →

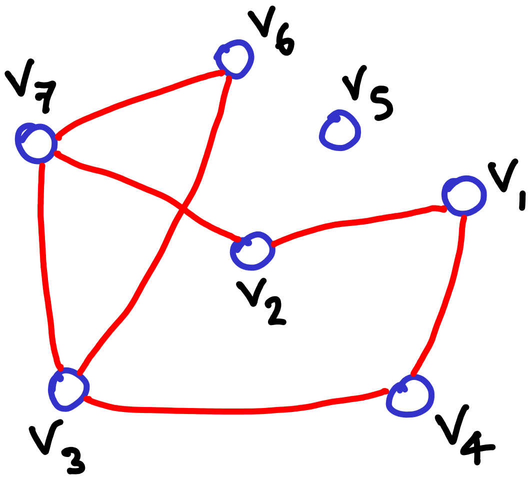
acyclic



not acyclic



Representation



$G = \{V, E\}$

	1	2	3	4	5	6	7
1	0	1	0	1	0	0	0
2	1	0	0	0	0	0	1
3	0	0	0	1	0	1	1
4	1	0	1	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	1	0	0	0	1
7	0	1	1	0	0	1	0

Adjacency matrix

size: $|V|^2$

(symmetric for undirected)

Adjacency list

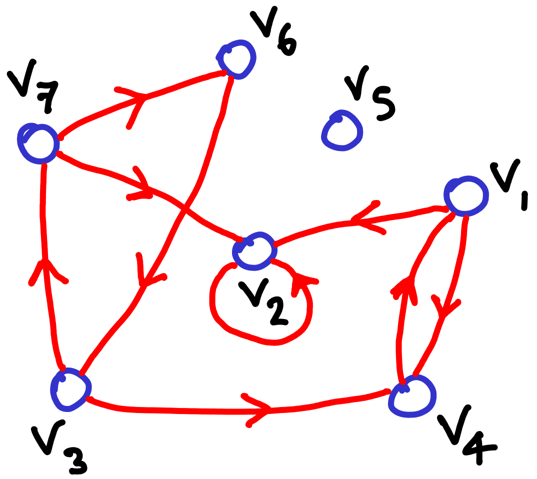
size: $|V| + 2|E|$

(undirected)

1 → 4 → 2
2 → 1 → 7
3 → 4 → 7 → 6
4 → 1 → 3
5
6 → 3 → 7
7 → 6 → 3 → 2

For every vertex y in primary column, store a list of neighbors of y

Representation



$$G = \{V, E\}$$

$$\begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 7 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{array}$$

Adjacency matrix

size: $|V|^2$

(directed or not)

Adjacency list

size: $|V| + |E|$

(directed)

$1 \to 2 \to 4$

$2 \to 2$

$3 \to 4 \to 7$

$4 \to 1$

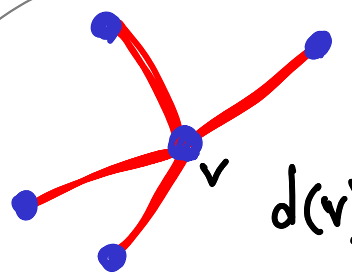
5

$6 \to 3$

$7 \to 6 \to 2$

Adjacency matrix size: $\Theta(V^2)$
Adjacency list size: $\Theta(V+E)$
} directed or not

Same for "dense" graphs, i.e. $E = \Theta(V^2)$
Adj. list uses less space for "sparse" graphs

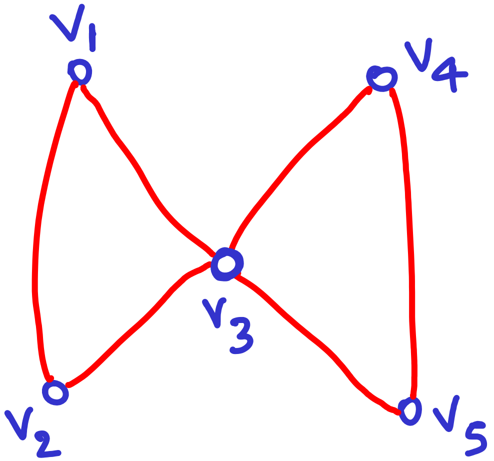


$d(v)=4$

- How much time to...
  - Query adjacency : (are v_i & v_j neighbors?)
    Matrix $\Theta(1)$
    List $O(V)$ ... but really $O(\text{degree}(v_i))$ or $d(v_j)$
  - Enumerate neighbors : (of one vertex, v_k)
    List $\Theta(d(v_k))$
    Matrix $\Theta(V)$

Vertex degree

Handshaking theorem



v_3 has 4 neighbors $\longleftrightarrow$ $d(v_3) = 4$

$\sum d(v) = 2+2+4+2+2 = 12$

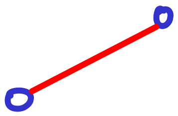
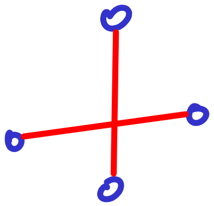
$|E| = 6$

$$\boxed{\sum_{v \in G} d(v) = 2 \cdot |E|}$$

(doublecounting every edge)

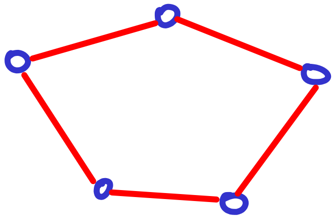
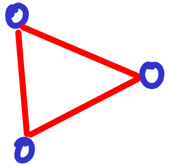
Regular graphs : all vertices have the same degree

$d=1$?

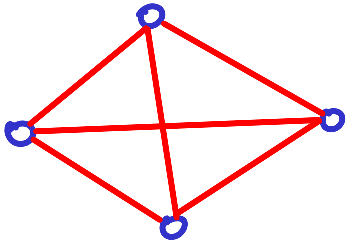
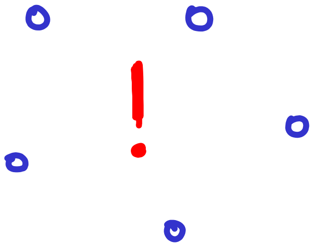
1-regular

$d=2$?

2-regular

$d=3$?

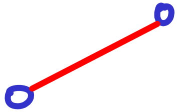



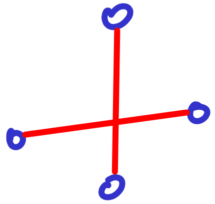
need $\sum d(v) = 15$

but $2E$ is even

Regular graphs : all vertices have the same degree

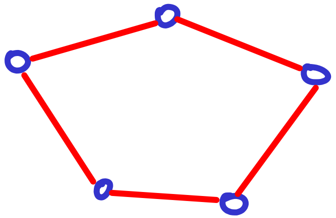
d = 1 ?

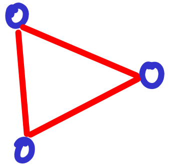




1-regular

d = 2 ?

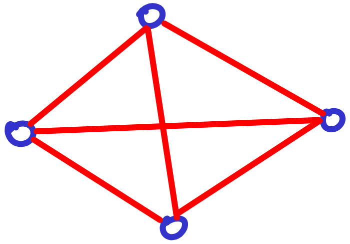


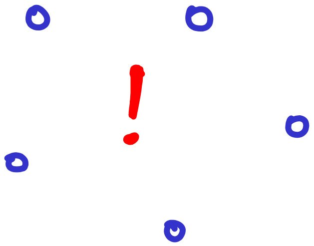


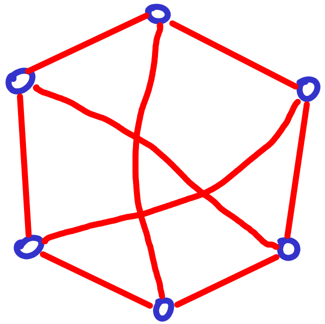
2-regular

3-regular

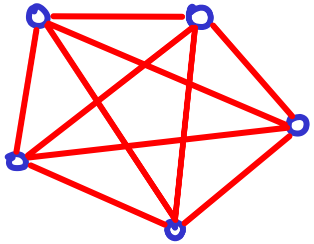
d = 3 ?







$V = 5$



4-regular

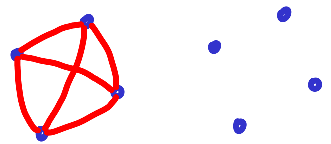
($V-1$ regular) $\rightarrow$ complete graph : K_5

#edges? $\quad V-1 + V-2 + V-3 + \cdots + 3 + 2 + 1 = \binom{V}{2}$

$$= \sum_{i=1}^{V-1} i = \frac{V\cdot(V-1)}{2}$$

Also: $\sum d(v) = V\cdot(V-1)$

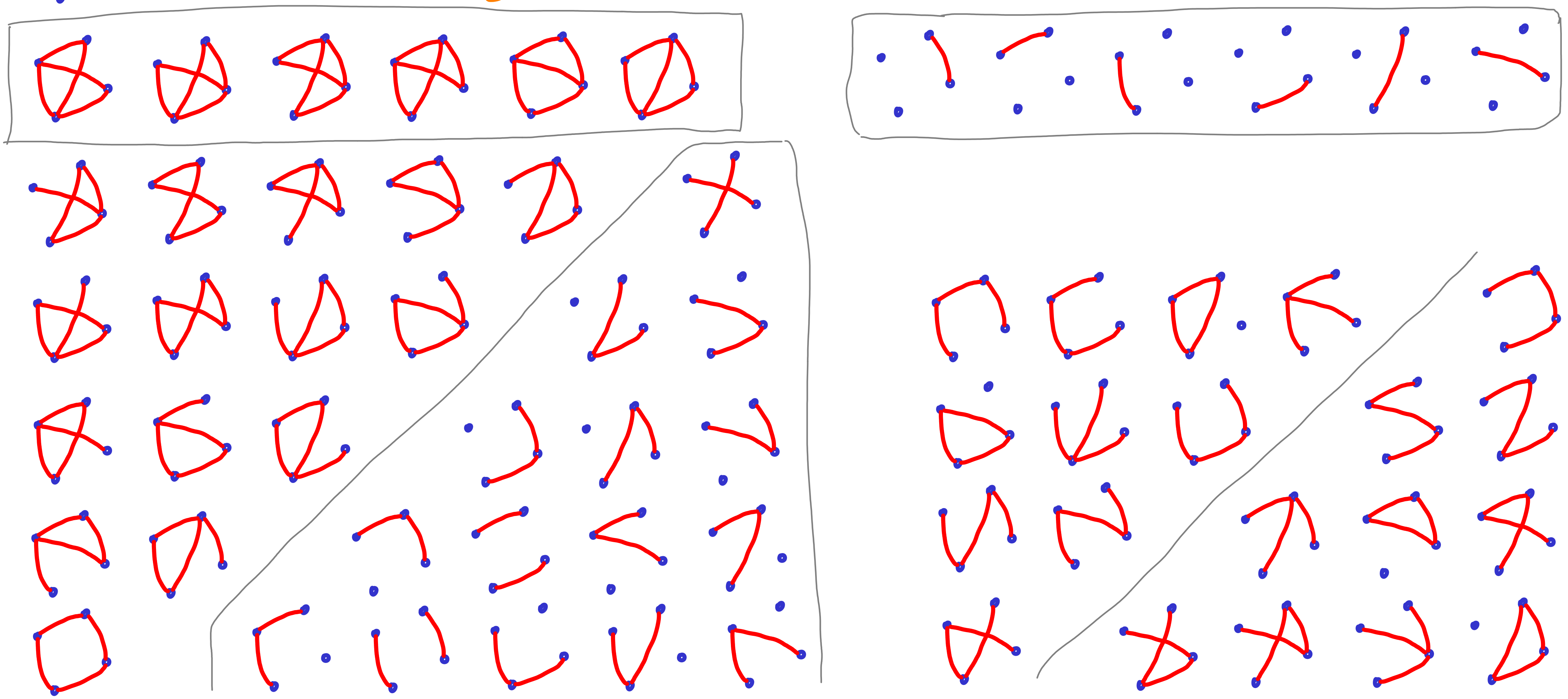
by Handshaking, $E = \dfrac{V\cdot(V-1)}{2}$



\# possible graphs on V vertices = $2^{\binom{V}{2}}$ $\{0,1\}^{\binom{V}{2}}$

(each edge is IN or OUT)

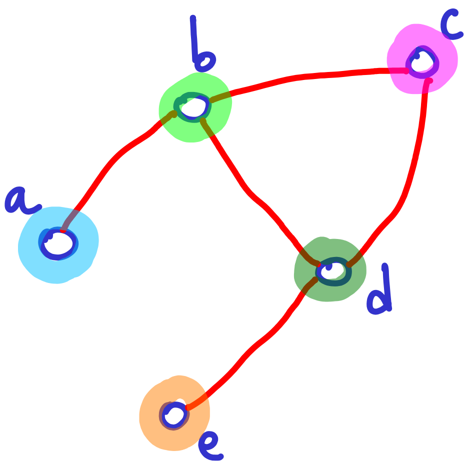
↳ but there are similar shapes

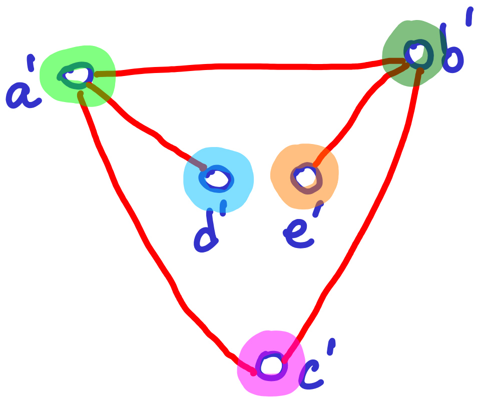


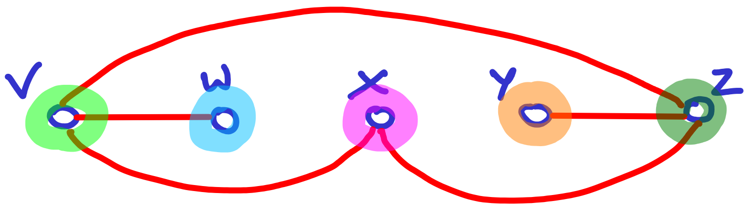
isomorphism → map vertices of graph G to vertices of graph H.

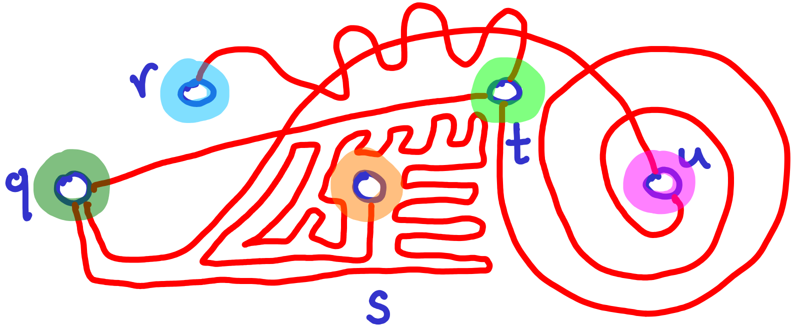
vertices $\alpha, \beta \in G$ share an edge in G

IFF vertices $m(\alpha), m(\beta) \in H$ share an edge in H.









a	d'	w	r
b	a'	v	t
c	c'	x	u
d	b'	z	q
e	e'	y	s

difficult – complicated – inefficient*

(not just because drawings look complicated)

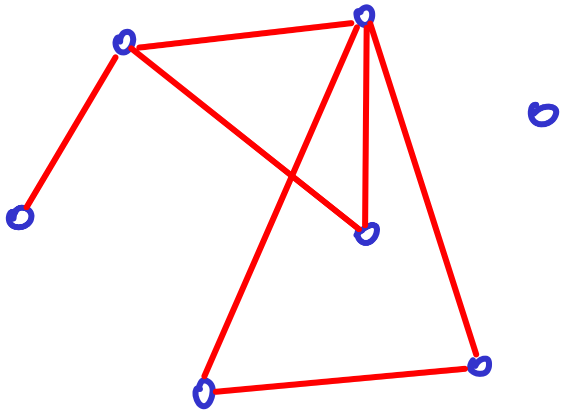
- Determining if two graphs are isomorphic (without a mapping)

- Counting # possible graphs without "doublecounting" isomorphs

* time complexity as function of V,E

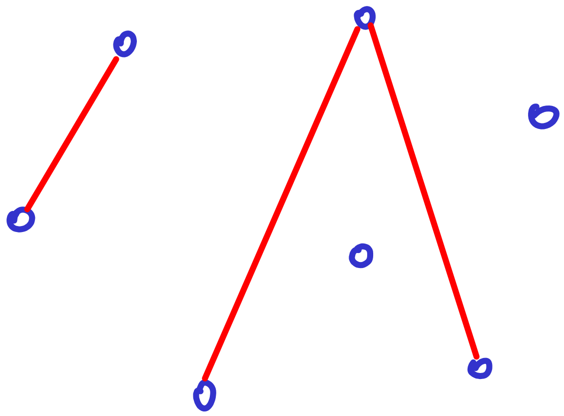
Subgraphs

G is a subgraph of H if $\left\{ \begin{array}{l} V(G) \subseteq V(H) \\ E(G) \subseteq E(H) \end{array} \right.$ $\Big\}$ if equal then it's a "spanning" subgraph



H

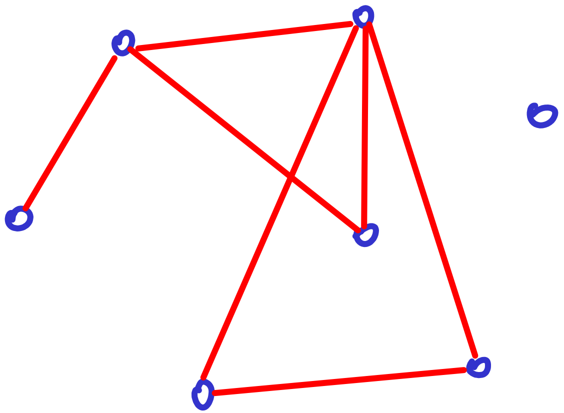
G



spanning subgraph of H

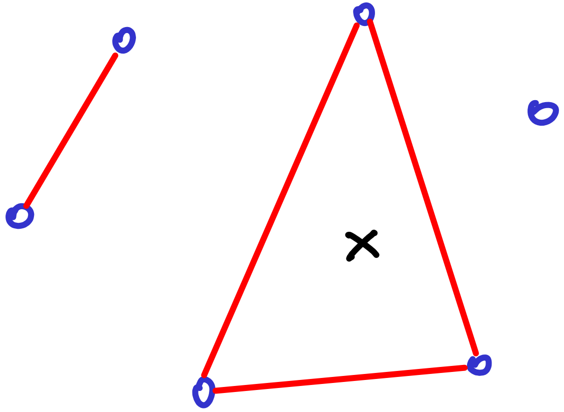
Subgraphs

G is a subgraph of H if $\begin{cases} V(G) \subseteq V(H) \\ E(G) \subseteq E(H) \end{cases}$ if equal then it's a "spanning" subgraph



H

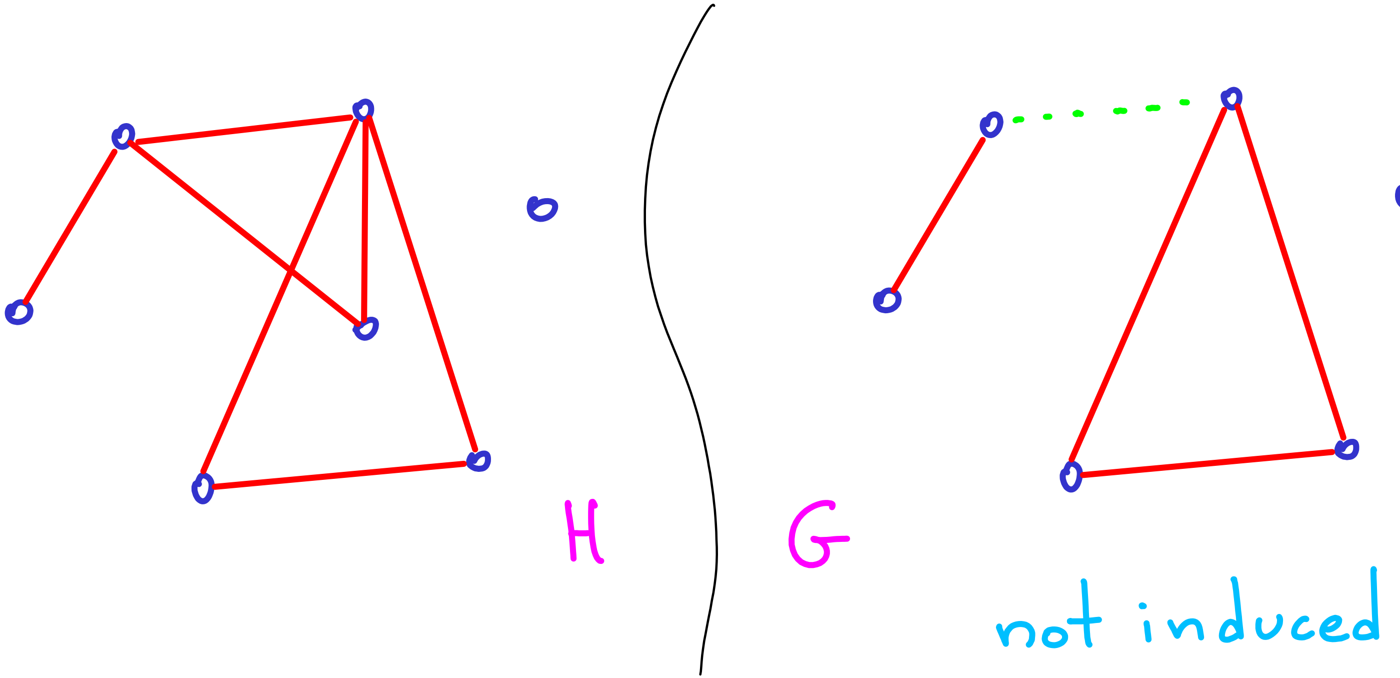
G



not a spanning subgraph of H

Subgraphs

G is a subgraph of H if $\begin{cases} V(G) \subseteq V(H) \\ E(G) \subseteq E(H) \end{cases}$ } if equal then it's a "spanning" subgraph



If you only remove edges as a result of removing vertices then G is an "induced" subgraph