Partially ordered sets (continued)

Digraphs

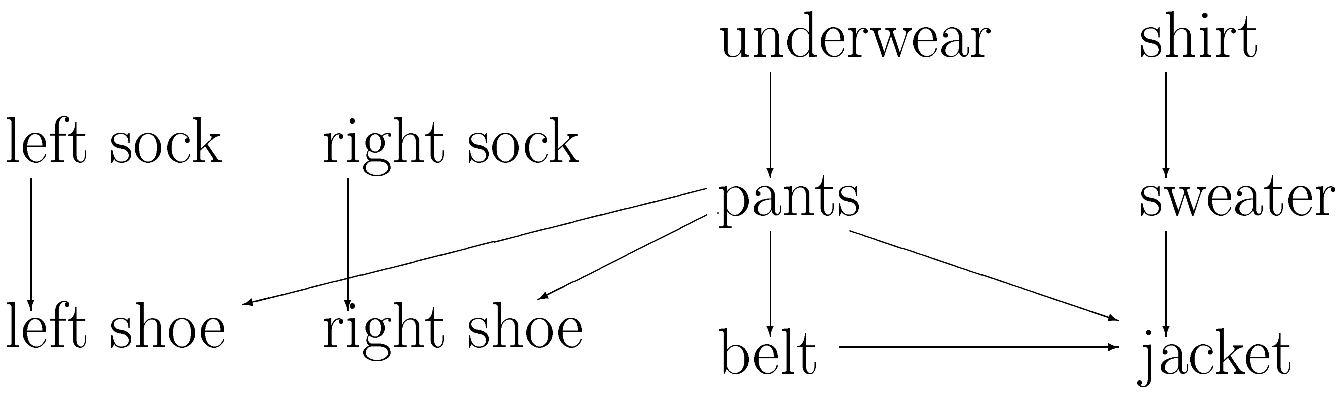
Graphs

Graph Coloring

Graph connectivity

Example for partially ordered set

Given a finite poset of tasks, how long does it take to do them all?
One time unit for a task,
Unlimited number of processors.
For example in the dressing example we could do all minimal elements first (left sock, right sock, underwear, shirt), remove and repeat.



Definition: A **chain** in a poset is a sequence of elements of the domain, each of which is smaller than the next in the partial order ($\preceq$ and $\neq$).

A longest chain is also known as a *critical path*.
Let t be the length of the longest chain.

Theorem: Given any finite poset $(A, \preceq)$ for which the longest chain has length t, it is possible to partition A into t subsets, $A_1, A_2, \cdots, A_t$ such that $(\forall i \in [1, t])(\forall a \in A_i)(\forall b \preceq a, b \neq a)[b \in A_1 \cup A_2 \cup \cdots \cup A_{i-1}]$.

proof: Put each a in set A_i, where i is the length of the longest chain ending at a.

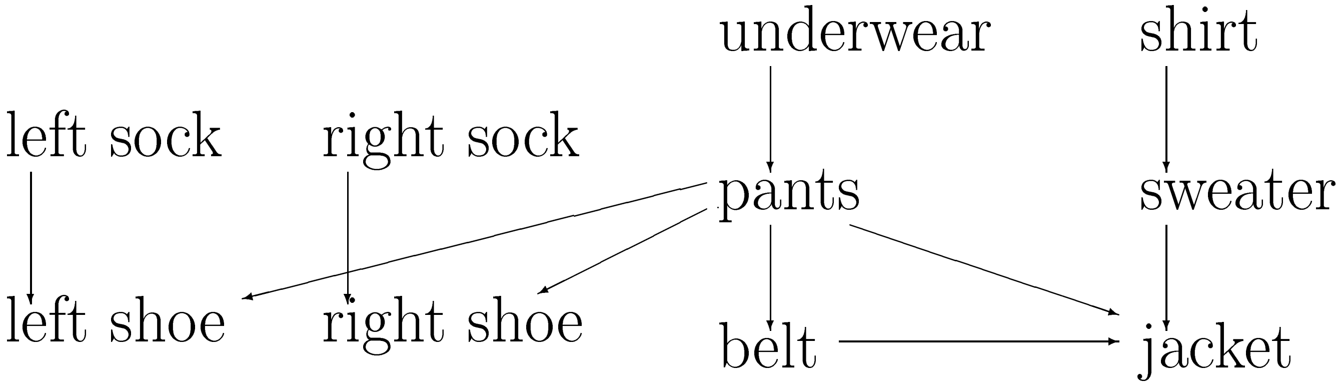
Need to show $(\forall i)(\forall a \in A_i)$
$(\forall b \preceq a, b \neq q)[b \in A_1 \cup A_2 \cup \cdots \cup A_{i-1}]$.

Proof by contradiction:
If $b \notin A_1 \cup A_2 \cup \cdots \cup A_{i-1}$ then there is a path of length $> i - 1$ ending in b,
which means there is a path of length $> i$ ending in a,
which means $a \notin A_i$.

We can schedule the tasks in t time units.

An **antichain** is a set of incomparable elements.

(Dilworth) $\forall t$, every poset with n elements must have a chain of size $> t$ or an antichain of size $\geq \frac{n}{t}$.



$n = 10$, $t = 3$ there exists a chain of length 4.

$t = 4$, no chain of length 4, but antichain of size $4 \geq \frac{10}{4}$.

(Dilworth) $\forall t$, every poset with n elements must have a chain of size $> t$ or an antichain of size $\geq \frac{n}{t}$.

Proof: Assume there is no chain of length $> t$.
So, longest chain has length $\leq t$.
Then by previous result, the n elements can be partitioned into $\leq t$ antichains.
(Cannot assume exactly t, since could have some empty.)
So there is an antichain with $\geq \frac{n}{t}$ elements.
As needed.

Corollary: Every poset with n elements has a chain of length $> \sqrt{n}$ or an antichain of size $\geq \sqrt{n}$.

Examine $t = \sqrt{n}$

Database of records with distinct numerical keys.
Suppose an insert requires comparisons.
Each comparison costs the time to perform it.
One extreme, database kept totally sorted or totally unsorted.
A search for a record with key k.
The worst-case number of comparisons depends on database partial order
Database totally sorted, start in the middle — Binary search
Approximately $\log n$ comparisons needed.
For any t, either have a chain of length $> t$, or an antichain of size $\geq \frac{n}{t}$.
Setting up a chain of length $> t$ requires at least t comparisons.
The antichain requires at least $\frac{n}{t}$ comparisons.
We have to pay the cost sometime – tradeoff – either we do $> t$ comparisons setting up the db, or at least $\frac{n}{t}$ in the worst case to search the db.

A distributed computation:
sequence of "steps" by each of n processes

local steps, *send* steps, *receive* steps

Processes are working independently,
interaction by messages

dependence among steps:
Successive steps of same process
Send vs. receive of same message

the reflexive transitive closure is a partial order

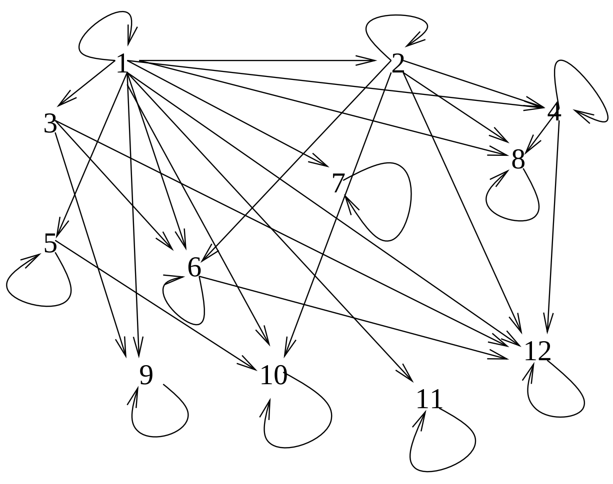
Basic theorem: Any topological sort of this partial order describes a possible sequence in which all the distributed steps could occur.
(No changes in what happens based on reordering incomparable steps.)

Lamport — have a logical time for every step
can be used for time-stamping update a shared file

Digraphs and binary relation are equivalent

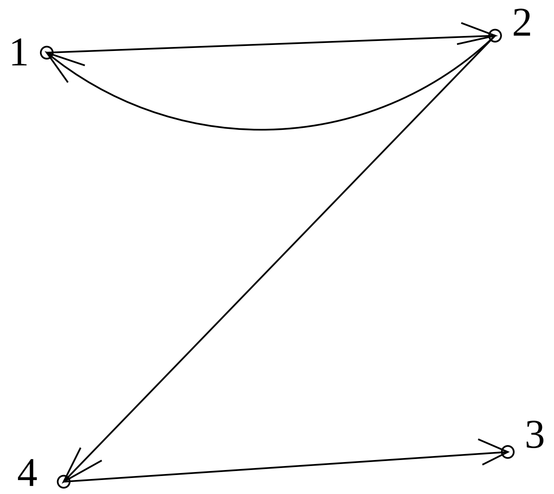
Edges are arrows
(Each edge is an ordered pair)

Precedence Constrains (relations)



A *directed graph* $G(V, E)$ is a pair of sets (V, E).
V is the set of *vertices*
E is the set of *edges* — an edge is a pair of distinct vertices

Diagram:



$V = \{1, 2, 3, 4\}$, $E = \{(1, 2), (2, 1), (2, 4), (4, 3)\}$

For a relation R on a finite set A,
$V = A$, the underlying set, and
$E = R$, the relation,
the set of ordered pairs (x, y) such that xRy.

Tournament digraphs (a directed edge between any two)

Computer networks:
V set of Computers,
E Communication links

Airline Connections:
V Airports,
E flights

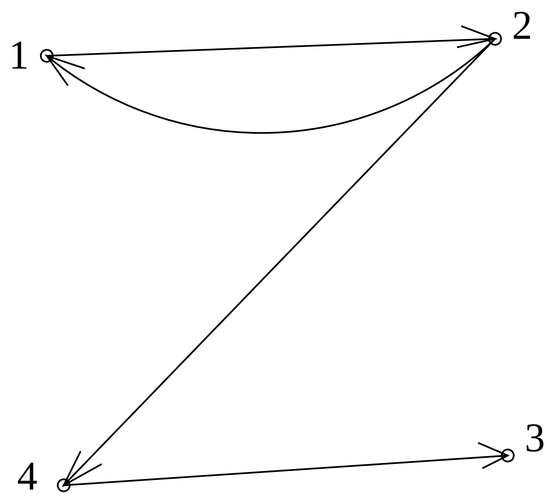
Precedence Constraints (e.g., dressing, prerequisites):
V Jobs,
E Constrains on job scheduling

Program Flowchart:
V step of computation,
E control flow

Weighted Graphs — A number is associated with each edge.
A *weight* $w(u, v)$ for each edge (u, v).
E.g., time for airline trip

x is the *initial vertex* of an edge (x, y)
y is the *terminal vertex*

in-degree of x, # of edges with x as terminal
out-degree of x, # of edges with x as initial



in-degree of vertex 2 is 1 out degree of vertex 2 is 2.

Theorem: Sum of the indegrees of all nodes in a graph = sum of outdegrees of all nodes in the graph = number of edges

Proof: induction on the number of edges,
Base: a graph with no edges
Step: consider a graph with $n + 1$ edges and remove an edge

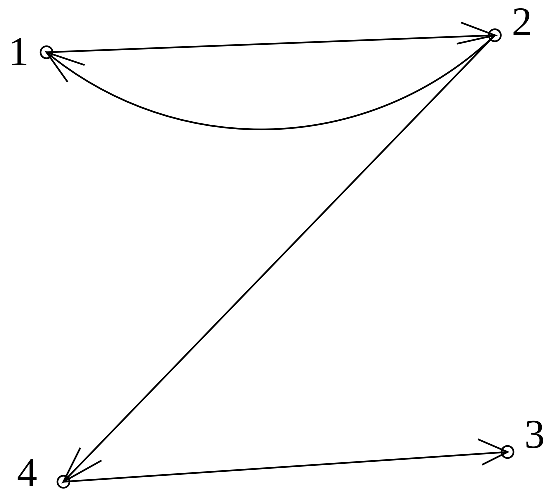
Path in a digraph: Same as path in the corresponding binary relation.

A *path* from a vertex u to a vertex v is a sequence of vertices $u = u_0, u_1, \ldots, u_k = v$ such that each consecutive pair (u_i, u_{i+1}) form an edge of the digraph.

The *length* of the path is k (the number of edges).

A path of length 0, is a single vertex u_0.
(Rosen's definition: paths have at least one edge).

A path may contain the same edge or vertex multiple times.



$u = 1, 2, 1, 2, 4, 3$ is a path of length 5

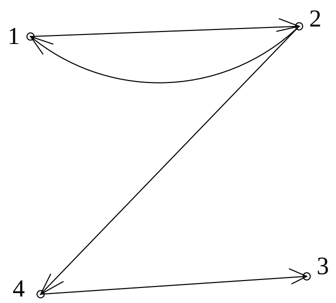
A path is *simple* if no vertex appears more than once,
Possible exception the first vertex may be the same as the last
(Rosen: defines "simple" to mean "no repeated edges".)

$1, 2, 4, 3$ and $2, 1, 2$ are simple paths,
while $2, 1, 2, 4$ is not simple.

Theorem: If $|V| = n$, then any simple path has length at most $n - 1$.

Theorem: If $|V| = n$, and a path from u to v exists in G,
then there is a simple path (of length $n - 1$) from u to v

Proof: Consider the shortest path from u to v
If not a simple path—so some vertex is repeated.
The path $u_0, \ldots, u_i, \ldots, u_j, \ldots, u_k$ has $u_i = u_j$.
$u_0, \ldots, u_i, u_{j+1}, \ldots, u_k$ is a shorter path
A contradiction.



$u = 1, 2, 1, 2, 4, 3$ can be shorten to $1, 2, 4, 3$

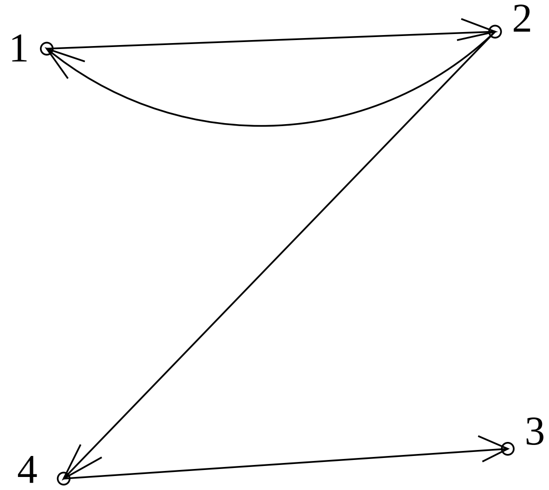
Circuit (also known as a *cycle*)
a path that begins and ends at the same vertex.

There is a path of length k from u to v in G if and only if $(u, v) \in E^k$. (E is the relation corresponding to the edges.)

There is a path from u to v in G if and only if (u, v) is in the reflexive transitive closure of E. (In this case, we say that u is *connected to* v in G.)

Adjacency matrix representation can be used to calculate the needed products and take the needed closures.

Euler circuit in digraph G: A circuit in G that traverses each edge of G exactly once.



An Euler circuit exists if we add the edge (3,2).

Digraph G is *strongly connected* if there is a path from every vertex to every other vertex.

Digraph G is *weakly connected* if there is a path between any two vertices in the underlying undirected graph

Theorem: Suppose G is a weakly connected digraph, Then G has an Euler circuit if and only if the in-degree and out-degree of each vertex are equal.

Proof:
Only if – we cannot traverse all edges of a node otherwise,
In-degree greater than out-degree then cannot exit,
Out-degree greater than in-degree then cannot enter sufficient number of times

If — first the existence of a circuit:
Start with any edge, and trace nodes,
Until arriving to a node with all outgoing edges traced. (This is the node we started in since whenever we reach another node, the number of the in-going edges that were traversed is one smaller than the number of out-going edges that we traversed.)

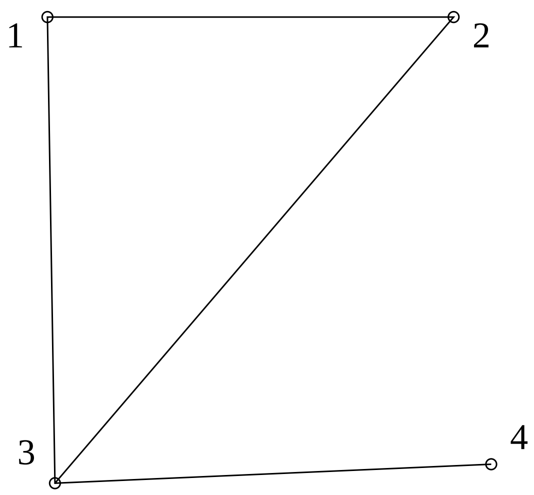
Choose a maximum length circuit C with no repeating edge,
If it does not include an edge, then
H is subgraph of G
Consisting all edges in G not in C.

There exists a node u in H that is also in C.
For if not, G is not weakly connected:
No edge connects a node in H with a node in C.

u has a non traversed (ingoing our outgoing) edge
u must have an outgoing edge that was not traversed
(the number of traversed ingoing and outgoing edges must be equal).

Starting with that outgoing edge, construct a cycle in H, eventually, it gets back to u.

Now we have two cycles with disjoint edge sets and a common vertex, u.
Can splice them together to get a larger cycle than C. Contradicts choice of C as maximum.



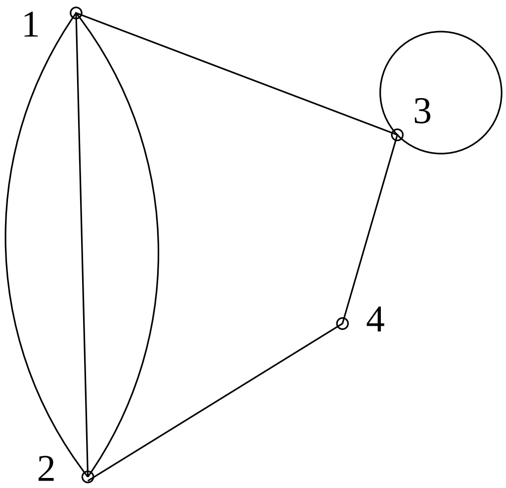
Undirected graphs correspond to symmetric relations — if (u, v) exists then (v, u) exists.

$V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (1, 3), (2, 3), (3, 4)\}$ (unordered pairs)

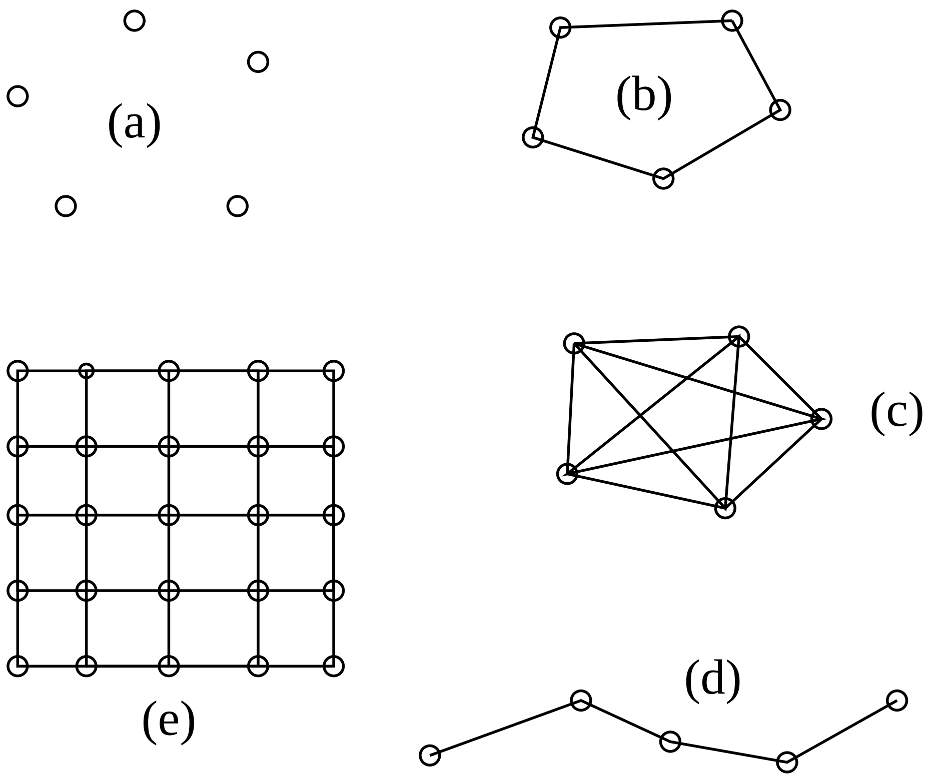
More Terms:
Vertex 1 is *adjacent* to vertex 2, but not to vertex 4.
Edge (u, v) is *incident* or *attached* to vertices u and v.
Vertices u and v are *adjacent* or *neighbors*.
"an edge joins u and v", "the edge between u and v"

Multigraph — Neighbors can be connected by several edges.

Self-Loops — an edge that connects a vertex to itself.



A multigraph with a self-loop.
three edges connecting vertices 1 and 2 and
a self-loop on vertex 3.

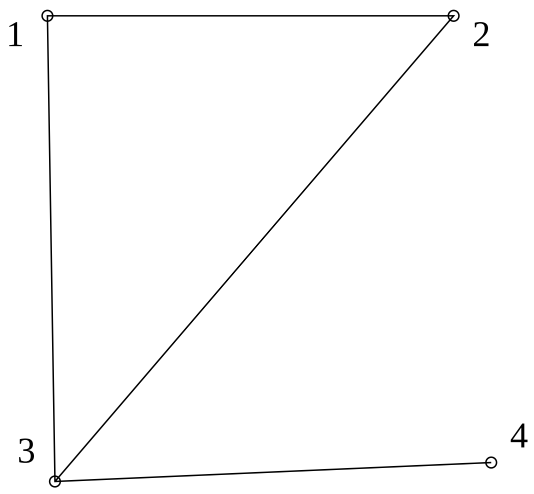


(a) The empty graph on five vertices, E_5. An empty graph has no edges at all.
(b) The cycle (or ring) on five vertices, C_5.
(c) The complete graph on five vertices, K_5.
(d) A five-vertex line graph.
(e) A 5×5 2-dimensional mesh.

The *adjacency matrix*

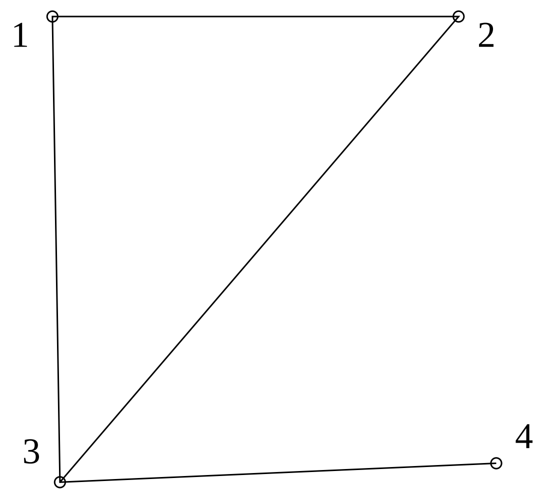
	1	2	3	4
1	0	1	1	0
2	1	0	1	0
3	1	1	0	1
4	0	0	1	0

Represents:



What does the following weight-matrix $w(i,j)$ represents?

	1	2	3	4
1	0	4	7	0
2	4	0	1	0
3	7	1	0	9
4	0	0	9	0



$G' = (V', E')$ is a *subgraph* of $G = (V, E)$ iff $V' \subseteq V$ and $E' \subseteq E$.

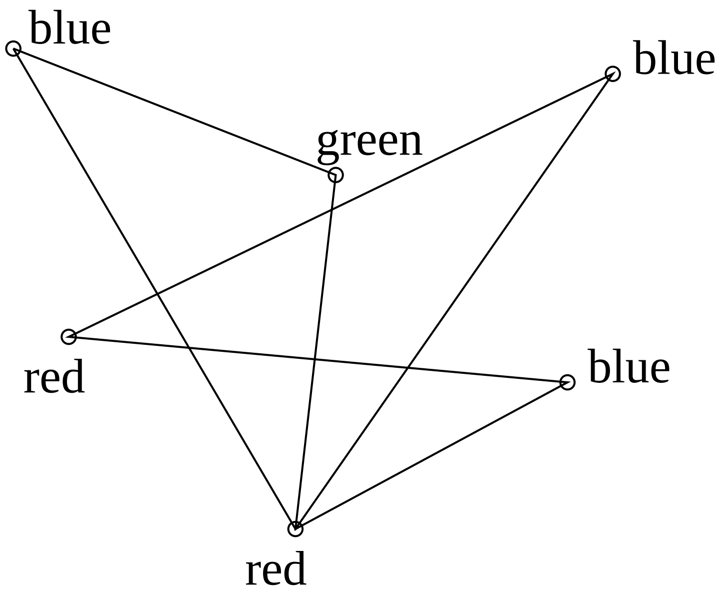
A *degree* of a vertex v, $d(v)$ is the number of edges incident to v.

What is the degree of a vertex in:
Empty graph?
Cycle?
Complete graph?

In a graph with m edges the sum of the degrees is $2m$

In every graph there are even number of vertices with odd degree

Exam scheduling problem.
Each vertex represents a course.
An edge (c_1, c_2) exists iff there is a student that takes both c_1 and c_2.
Each time slot for an exam is represented by a distinct color: Monday 9-12 by red, Monday 1-4 by blue, etc.
No two neighboring vertices receive the same color.

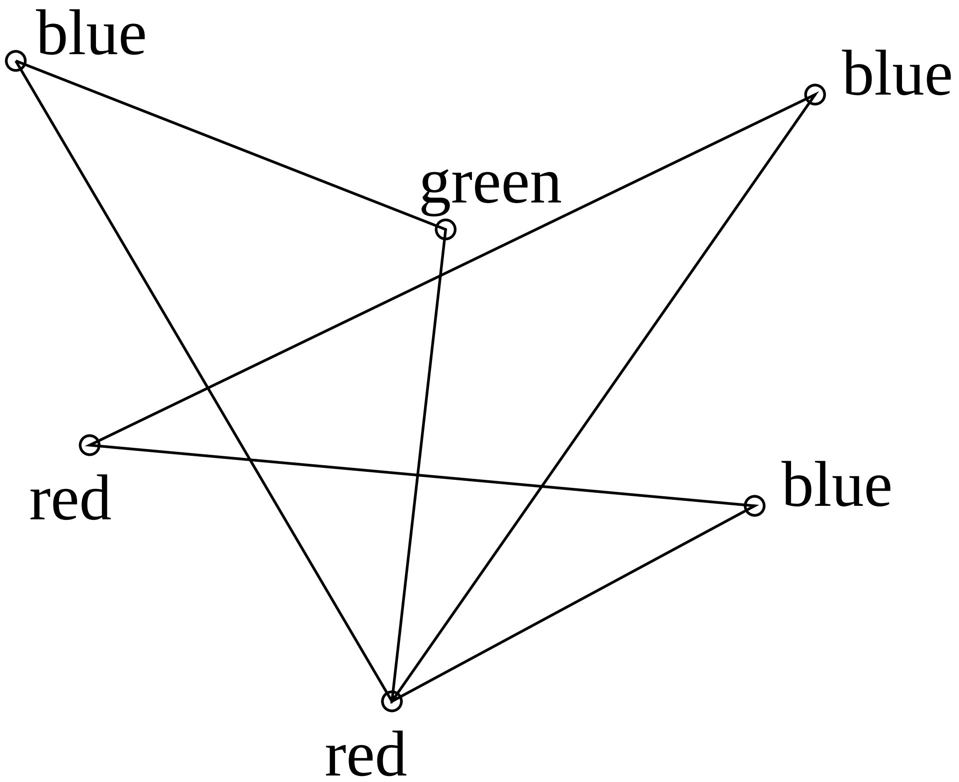


$\chi(G)$ the *chromatic number* is the minimal number of colors needed to color G so that no two adjacent vertices receive the same color.

$\chi(G)$ of a triangle graph is 3.

Upper bounds for a graph with n nodes $\chi(G) \leq n$.

In fact $\Delta + 1$, where Δ is the maximal vertex degree.



Connectivity related questions:
Is a graph all in one piece or composed of several pieces?
What is the length of the shortest path from one vertex to another?
Are there many different routes or just one?

A *path* from a vertex u to a vertex v is a sequence of vertices $u = u_0, u_1, u_2, \ldots, u_k = v$, where each (u_i, u_{i+1}) is an edge

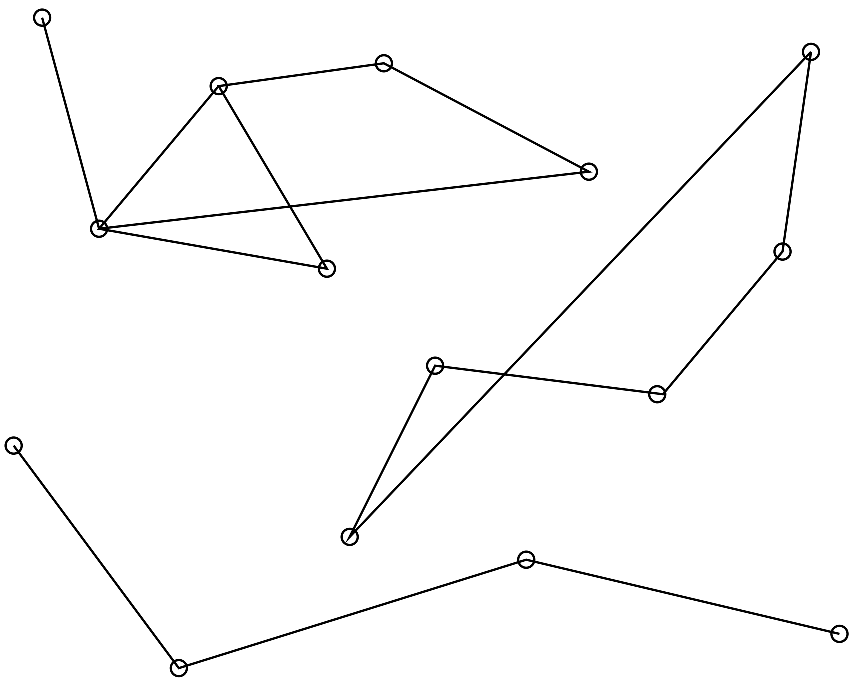
A path may contain the same edge or vertex multiple times.

A *simple path*, no vertex is crossed more than once.
$u_0 = u_k$ is permitted.

A *circuit* or a *cycle* is a path from a vertex back to itself.

A graph is *connected* if there is a path between every pair of distinct vertices.

A connected component is a subset of the vertices such that
each can reach any other in the subset,
but can't reach any vertex not in the subset.



Connectedness is an equivalence relation —
reflexive — (if we allow length 0 paths).
symmetric — because we can reverse paths in an undirected graph.
transitive — because we can concatenate paths.

Connectedness defines a partition of the vertices into equivalence classes.

Each equivalence class is called a *connected component.*