Discrete Math and Analyzing Social Graphs

Seara

2021-07-01

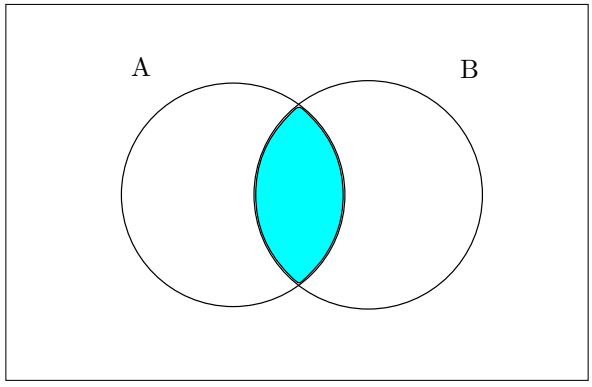
1 Week 1

1.1 Basic counting techniques

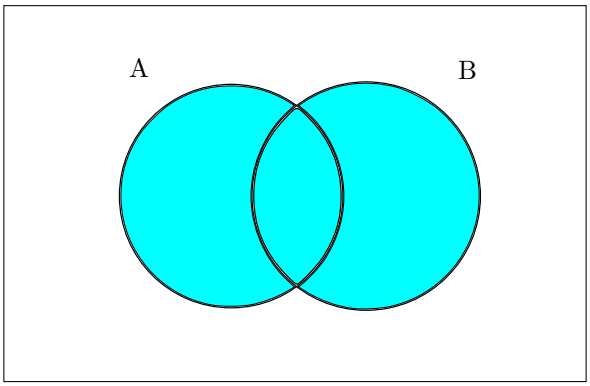
Rule of sum. If there are k objects of the first type and there are n objects of the second type, then there are $n + k$ objects of one of two types. In the rule of sum no object should belong to both classes!

Set. Set is an arbitrary group of arbitrary objects.

- The order of elements is not important: $\{0, 1, 2, 3\} = \{2, 0, 3, 1\}$
- Repetitions in the set of elements are not important: $\{0, 1, 2, 3\} = \{1, 0, 1, 3, 2, 3\}$
- The set $A \cap B$ is an intersection of sets: it consists of elements belonging to both sets.



- The set $A \cup B$ is a union of these sets: it consists of elements belonging to at least one of the sets.



- If every element of A is also an element of B, then A is a subset of B; we write $A \subseteq B$.
- If some object x is an element of A we write $x \in A$.
- The number of elements in the set A is $|A|$ (can be infinite).
- If some object x is an element of A we write $x \in A$.
- A set without elements is denoted by $\emptyset$ and is called empty set.

Rule of sum in the set language. If there is a set A with k elements, a set B with n elements and these sets do not have common elements, then the set $A \cup B$ has $n + k$ elements

Generalized rule of sum. If there are finite sets A and B, then

$$|A \cup B| = |A| + |B| - |A \cap B|$$

This covers the original rule, when $|A \cap B| = 0$.

Rule of product. If there are k objects of the first type and there are n objects of the second type, then there are $k \times n$ pairs of objects, the first of the first type and the second of the second type.

Rule of product in the set language. If there is a finite set A and a finite set B, then there are $|A| \times |B|$ pairs of objects, the first from A and the second from B.

1.2 Tuples and permutations

Cartesian product. By $|A| \times |B|$ we denote the set of all pairs (a, b), where $a \in A$ and $b \in B$. $|A| \times |B|$ is called Cartesian product of sets A and B. If A and B are finite, the number of elements in $|A| \cdot |B|$ is equal to $|A| \times |B|$.

- More generally, suppose we are given sets $A_1, A_2, ..., A_k$
- $A_1 \times A_2 \times ... \times A_k$ we denote the set of all tuples $(a_1, a_2, ..., a_k)$, where $a_1 \in A_1$, $a_2 \in A_2$ and so on
- $A_1 \times A_2 \times ... \times A_k$ is called Cartesian product of sets $A_1, A_2, ..., A_k$
- When $A_1 = A_2 = ... = A_k$ it is convenient to shortenthe notation to A^k
- That is, the set of tuples in which each coordinate is taken from the same set is denoted by A^k

We have seen in the previous video that for finite A

$$|A^k| = |A|^k$$

Permutations. Tuples of length k are called k-permutations.

$$k - permutations\ of\ n = \frac{n!}{(n-k)!}$$

1.3 Combinations

Combinations. For a set S its k-combination is a subset of S of size k

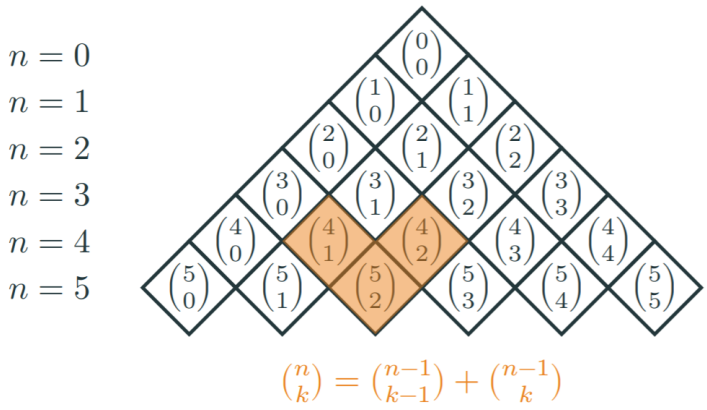
$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

- The number of k-combinations of n element set is denoted $\binom{n}{k}$
- Pronounced 'n choose k'

2 Week 2

2.1 Binomial theorem

Pascal triangle.



Formula:

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

is much better for computing binomials.
Pascal triangle is symmetric:

$$\binom{n}{k} = \binom{n}{n-k}$$

Comparing binomials. If $k \leq n/2$

$$\binom{n}{k-1} < \binom{n}{k}$$

If $k > n/2$

$$\binom{n}{k-1} > \binom{n}{k}$$

This means that binomials grow in the middle of Pascal Triangle.

Binomial theorem.

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k$$

One of the consequences:

$$2^n = \sum_{k=0}^{n} \binom{n}{k}$$

2.2 Combinations with repetitions

Combinations with repetitions. The number of combinations of size k of n objects with repetitions is equal to $\binom{k+n-1}{n-1}$
n ingredients mean $n-1$ delimiters; choosing $(n-1)$ element in the line of $k+(n-1)$ elements

Final table. We considered selections of k items out of n possible options.

	With repetitions	Without repetitions
Ordered	Tuples n^k	k-permutations $\frac{n!}{(n-k)!}$
Unordered	Combinations with repetitions $\binom{k+n-1}{n-1}$	Combinations $\binom{n}{k}$

3 Week 3

3.1 The notion of event

Random experiment. For example, toss a coin. Two possible outcomes - HEAD or TAIL. Sample space $\Omega = \{H, T\}$

Event. Event is a set of outcomes of an experiment. Events obey the laws of boolean logic. Example: X, Y - events, $X \subset \Omega$, $Y \subset \Omega$

- $X \cap Y$ - both events happen;
- $X \cup Y$ - at least one of the events happen;
- $X \subset Y$ - if X happens, Y also happens;
- $\overline{X}$ - X do not happen.

3.2 Calculating probabilities

Classical probability.

$$P(A) = \frac{|A|}{|\Omega|},$$

where $|A|$ is the length of the event set.

Probability of union.

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

This comes from:

$$P(A \cup B) = \frac{|A \cup B|}{|\Omega|} = \frac{|A| + |B| - |A \cap B|}{|\Omega|} = \frac{|A|}{|\Omega|} + \frac{|B|}{|\Omega|} - \frac{|A \cap B|}{|\Omega|}$$

Mutually exclusive events. If $A \cap B = \emptyset \Rightarrow P(A \cup B) = P(A) + P(B)$

Useful fact about probabilities:.

$$\overline{P(A)} = 1 - P(A)$$

Analysis of bagging procedure. Bagging chooses elements from original dataset one by one with repetitions to form another dataset for ML purposes. It turns out that the number of all possible datasets that can be formed is $|\Omega| = n^n$ (Cartesian product). Outcomes $(y_1, ..., y_n) \in \{x_1, ..., x_n\}$ Fun fact: the chance for one object not to be in newly formed dataset is $\frac{(n-1)^n}{n^n} = (1 - \frac{1}{n})^2 \approx \frac{1}{2.7}$

Outcomes with non-equal probabilities. Imagine having a sample space $\Omega = \{\omega_1, ..., \omega_n\}$ of length $|\Omega| = n$, where for each ω_k where $k \in [1, n]$ there is corresponding probability P_k.

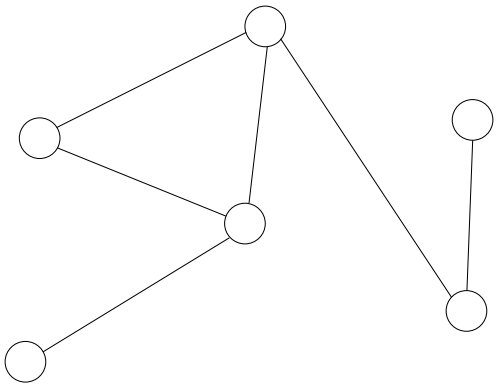
- $P(\{\omega_i\}) = P_i$
- $P_i \geq 0$
- $P_1 + P_2 + ... + P_n = 1$
- $P(\{\omega_1, \omega_2\}) = P_1 + P_2$

All the properties of classical probability also works with this type.

4 Week 4

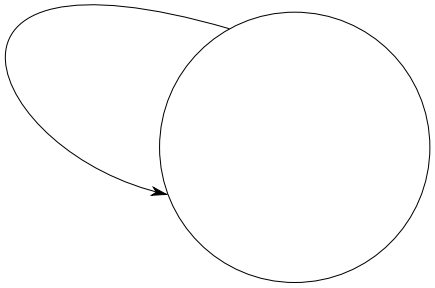
4.1 Graphs

Graph. Graph is a set of vertices, some of which are connected by edges.

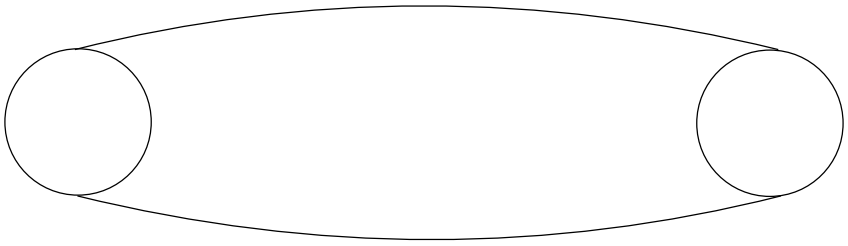


Graphs can be directed and undirected. There are two special kinds of edges in graphs:

- Loop: a vertex connected to itself.



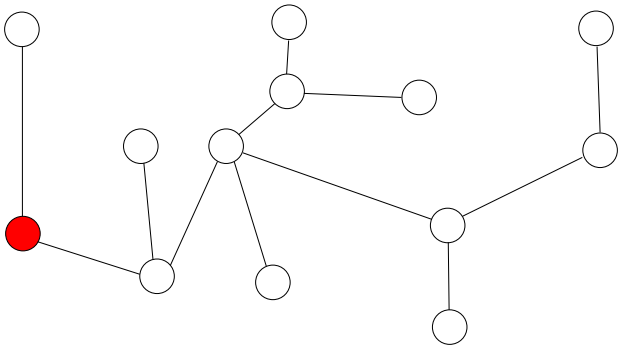
- Parallel edges: two vertices connected by more than one edge(undirected).



By default, loops and parallel edges are disallowed. A graph with parallel edges is called a *multigraph*. A graph with parallel edges and loops is called a *pseudograph*.

Trees. A path in a graph is a sequence of vertices $v_0, v_1, ..., v_n$, such that v_i is connected to v_{i+1} by *edges*. If $v_n = v_0$, then this path is a *cycle*. A cycle is simple, if $v_0, v_1, ..., v_n$ are different vertices and $n \geq 3$

Rooted trees. One can pick an arbitrary vertex of a tree and declare it as a root.

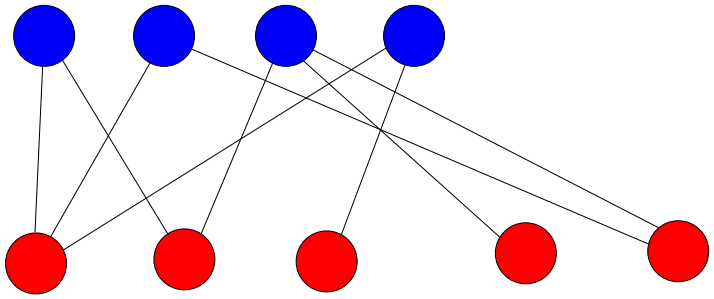


For any vertex (except the root), there is exactly one vertex, which is next to it on the path towards the root. This is the *parent* of the given vertex. Other vertices are children. Vertices without children are called *leaves*. If a tree has n vertices and m edges, then $m = n - 1$

Vertex colorings. In a coloring, each vertex is marked by a color taken from a finite set of possible colors. A coloring is *proper*, if endpoints of any edge have different colors.

The four color theorem. Any map can be colored in 4 colors, such that adjacent areas have different colors. If a graph can be drawn on a plane without intersections, then it is 4-colorable.

Bipartite graphs. 2-colorable graphs are also called bipartite. Vertices in a bipartite graph can be split into two parts such that edges go only between parts.



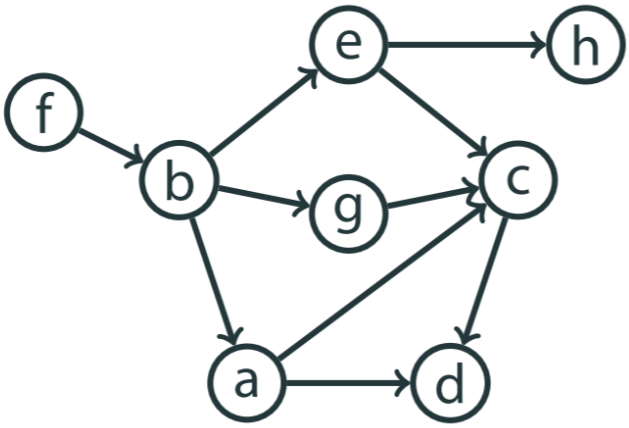
2-coloring is impossible, if there is a cycle with an odd number of vertices.

4.2 Cycles in graphs

Euler paths and cycles. A *Euler path* is a path in the graph which visits each edge exactly once. A *Euler cycle* is a Euler path which starts and ends at the same vertex. An odd vertex is a vertex which has an odd number of edges adjacent to it. If there exists a Euler path, then the graph has at most two odd vertices. If there exists a Euler cycle, then the graph has no odd vertices.

Hamiltonian paths and cycles. A *Hamiltonian path* must visit each vertex exactly once. A *Hamiltonian cycle* is a Hamiltonian path which starts and ends at the same vertex.

Directed acyclic graphs. A directed graph without directed cycles is called acyclic. Directed acyclic graphs are called DAGs.

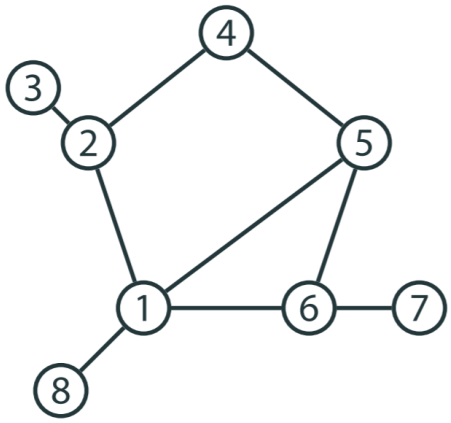


Topological sorting. In directed acyclic graphs, vertices can be enumerated in such a way that all edges go forward. For graphs with directed cycles topological sorting is impossible.

Traversing graphs. Traversing graphs means visiting its vertices in a specific order. Trees are usually traversed recursively. If we cut root, the tree splits into several subtrees, each with its own root. We run our traversing function recursively for each of these new trees. This is the *depth-first* traversing algorithm for trees. But when to visit the root? In the *pre-order*, we visit the root before traversing the sub-trees(like from left to right, top to bottom). In the *post-order*, we visit the root after traversing the sub-trees(like left to right, bottom to top). For binary trees, the third traversing order is available. In the *in-order*, we first traverse the left sub-tree, then visit the root, and then traverse the right sub-tree.

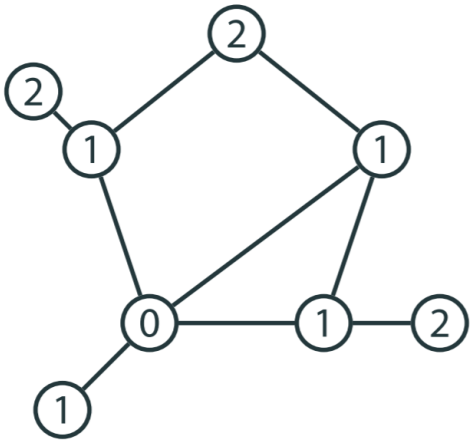
Traversing Graphs: DFS and BFS.

- Depth-first search



  DFS does not always find the shortest way to a vertex.

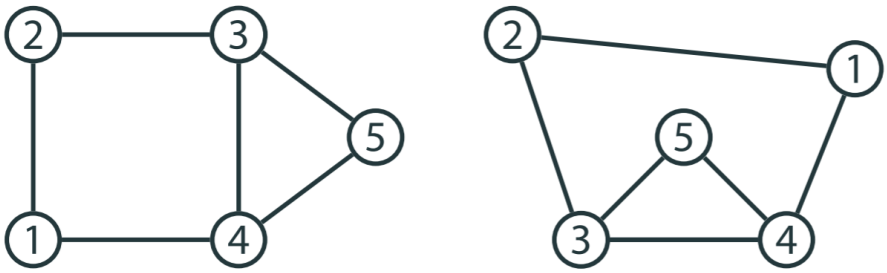
- Breadth-first search



5 Week 5

5.1 Degrees and distances

Isomorphic graphs. Two graphs, G and H, are *isomorphic*, if they have the same number n of vertices and vertices of each graph can be enumerated by numbers from 1 to n, so that vertices with numbers i and j are connected in G if and only if vertices with these numbers are connected in H.



The *isomorphism* itself is the correspondance between vertices with the same number.

Graph invariants. It's a parameter of the graph, which keeps the same among isomorphic versions of the same graph. For example, number of vertices, number of edges, acyclicity, the number of isolated vertices.

Vertex degree. The degree of a vertex is the number of edges connected to it. This is always the case:

$$number\ of edges\ = \frac{sum\ of\ vertex\ degrees}{2}$$

So, the sum of vertex degress should always be even, because we divide it by two.

Handshaking lemma. The number of people who shook hands with an odd number of other people is even. The handshaking lemma is not the criterion of existence of a graph with given degrees of vertices.

Clustering coefficients. A *triplet* is a pair of edges going from one vertex. A *triange* is a triple of interconnected vertices. Global clustering coefficient:

$$GCC(graph) = \frac{3 * (number\ of\ triangles)}{number\ of\ triplets}$$

Each triangle includes three triplets. That's why we multiple it by 3. In the local clustering coefficient, we count only triplets with a given A as the central vertex. If the degree of A is k, then the total number of triplets with A as the central is $k \cdot (k-1)/2$.

$$LCC(vertex) = \frac{number\ of\ pairs\ (B, C)\ which\ form\ a\ triangle\ with\ A}{k \cdot (k-1)/2}$$

If A is an isolated vertex ($degree = 0$), then $LCC(A)$ is undefined (zero-by-zero division).

Distances. Diameter. Eccentricity. The *length* of the path is the number of *edges* in it: $n - 1$. The *distance* between two vertices is the length of the shortest path connecting them. If the is no path then the distance is infinite:

$$d(s, t) = \infty$$

The distance from a vertex to itself is zero:

$$d(s, s) = 0$$

Triangle inequality:

$$d(s, t) \leq d(s, q) + d(d, t)$$

for any vertices s, t, q.
The *eccentricity* of a given vertex is the maximal distance from this vertex to another one: $ecc(u) = max_{v \in V} d(u, v)$. Calculating distance is an application of BFS. Once BFS reaches the other given vertex t, we know the distance $d(s, t)$. If BFS finishes without reaching t then t is not reachable from s, i.e, $d(s, t) = \infty$. If BFS reaches all vertices, then $ecc(s)$ is the number of layers produced by BFS (not counting the 0-th one). Otherwise, $ecc(s) = \infty$.
The *diameter* of a graph is the maximal distance possible in this graph:

$$diam(graph) = \max_{u,v \in V} d(u, v)$$

Equivalently, the diameter is the maximal eccentricity:

$$diam(graph) = \max_{u \in V} ecc(u)$$

The *radius* is the minimal eccentricity:

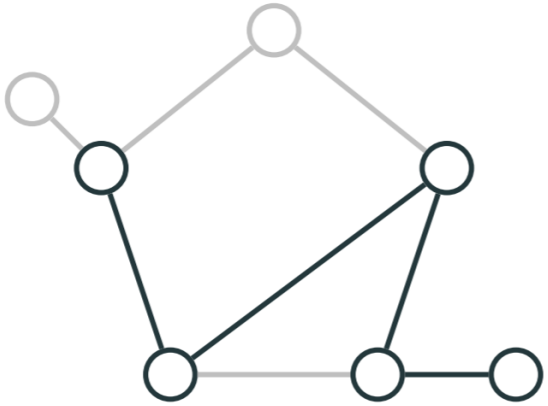
$$r(graph) = \min_{u \in V} ecc(u)$$

It is easy to see that:

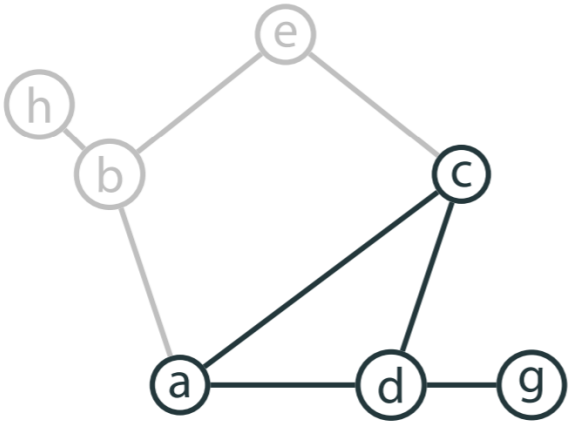
$$r(graph) \leq diam(graph) \leq 2 \cdot r(graph)$$

5.2 Subgraphs

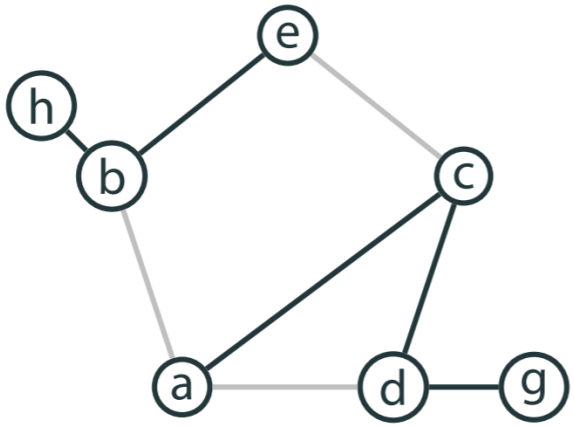
Cliques, independent sets. A *subgraph* is a part of a graph which is obtained by taking a subset of vertices and subset of edges. The vertex subset should cover the edge subset.



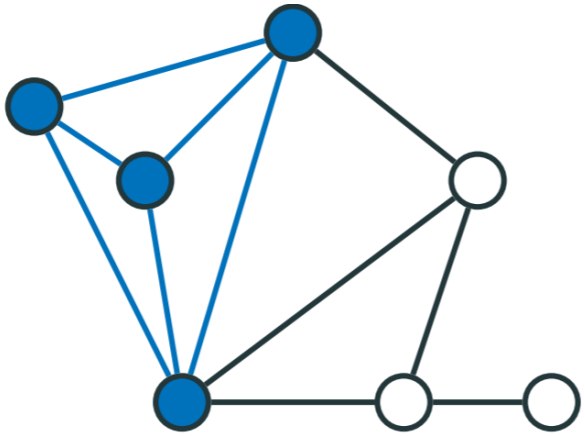
- An *induced* subgraph includes all the edges of the original graph, whose endpoint are in the vertex subset.



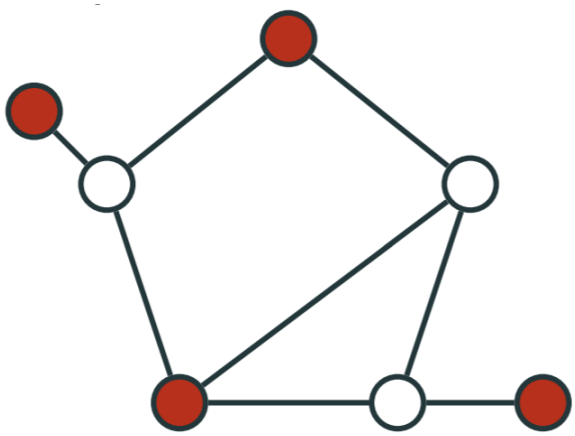
- A *spanning* subgraph includes all vertices of the original graph(but maybe not all edges).



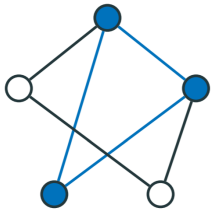
- A *clique* is a complete subgraph such that every two distinct vertices in the clique are adjacent.

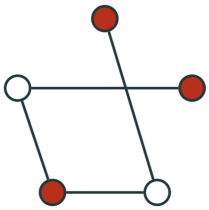


- An *independent set* is an empty induced subgraph. No pair of vertices from an independent set could be connected.

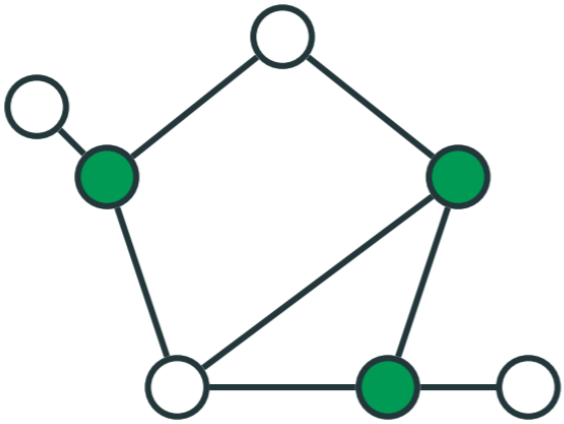


Cliques in a graph are exactly independent sets in the complement(inverse) graph.





Vertex cover. A *vertex cover* is a subset C of vertices such that for any edge at least one its endpoint belongs to C. A vertex cover is *optimal*, if it contains the smallest possible number of vertices.



If a set C of vertices is a vertex cover if and only if its complement, $V - C$, is an independent set.

5.3 Connectedness

Connected vertices. Two vertices are connected, if there exists a path between them. In other words, u and v are connected, if $d(u, v) < \infty$. The whole graph is connected, if any two vertices are connected.

Connected components. Vertex v belongs to the connected component of vertex u, if v and u are connected. Each graph gets split into several disjoint connected components. The number of connected components is a graph invariant.

Inequasion on the number of connected components. Let us estimate the minimal number of connected components in a graph with n vertices and m edges. A connected graph on n vertices should have at least $n-1$ edges(minimal example: tree). If $m < n-1$, then the graph has at least two connected components(meaning that the graph consists of at least two parts which are not connected). More generally, if a graph on n vertices has k connected components, it should have at least $n-k$ edges. In other words, $k \geq n-m$. However, even if $m \geq n-1$, the graph could be disconnected. If a connected component includes n_1 vertices, then the number of edges is less or equal than $n_1 \cdot (n_1-1)/2$. If a graph has two (or more) connected components, $n_1+n_2 = n$, then $m \leq n_1 \cdot (n_1-1)/2 + n_2 \cdot (n_2-1)/2$. If m is greater, the graph should be connected. The maximum value of this sum is reached when $n_1 = 1$ and $n_2 = n-1$. So, if $m \geq (n-1) \cdot (n-2)/2$, then the graph should be connected.

Circuit rank. The circuit rank is the graph invariant and is equal to number of edges which should be removed from the graph to break all its cycles. Its explisit formula:

$$r = m - n + c,$$

where m is the number of edges, n is the number of vertices, and c is the number of connected components. For a connected graph:

$$r = m - n + 1$$

If we remove $m-n+1$ edges, then the number of edges becomes $n-1$, which is the minimal possible for a connected graph. If we remove less than $m-n+1$ edges, the graph could not become a tree or forest and it would stil contain cycles.