Computer Science Fundamentals

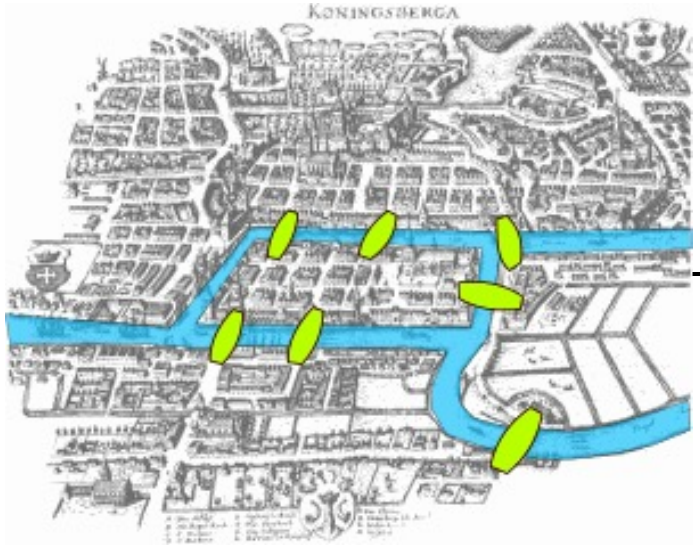
Graph Theory – Introduction

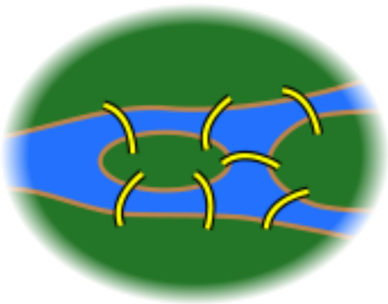
Technische Hochschule Rosenheim
Winter 2021/22
Prof. Dr. Jochen Schmidt

Seven Bridges of Königsberg (*Königsberger Brückenproblem*)

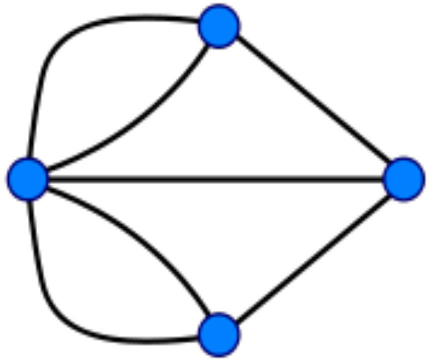
Technische Hochschule **Rosenheim**

Euler 1736:
Is there a circular route through Königsberg that crosses each of the seven bridges over the Pregel exactly once?

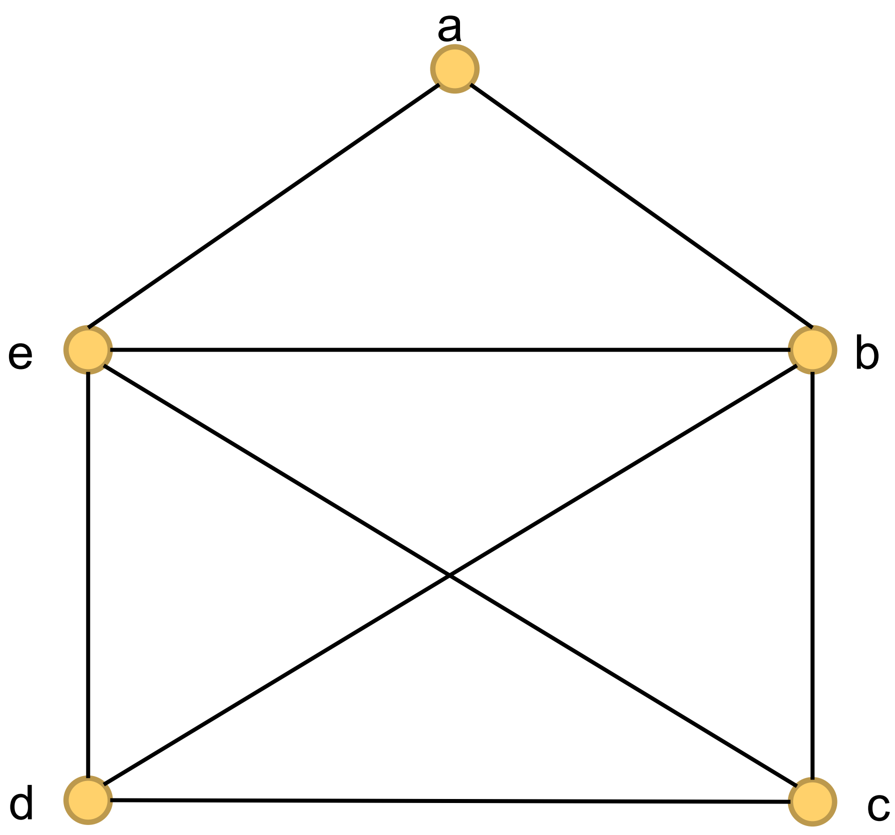




Is there a circular route that contains each edge exactly once?

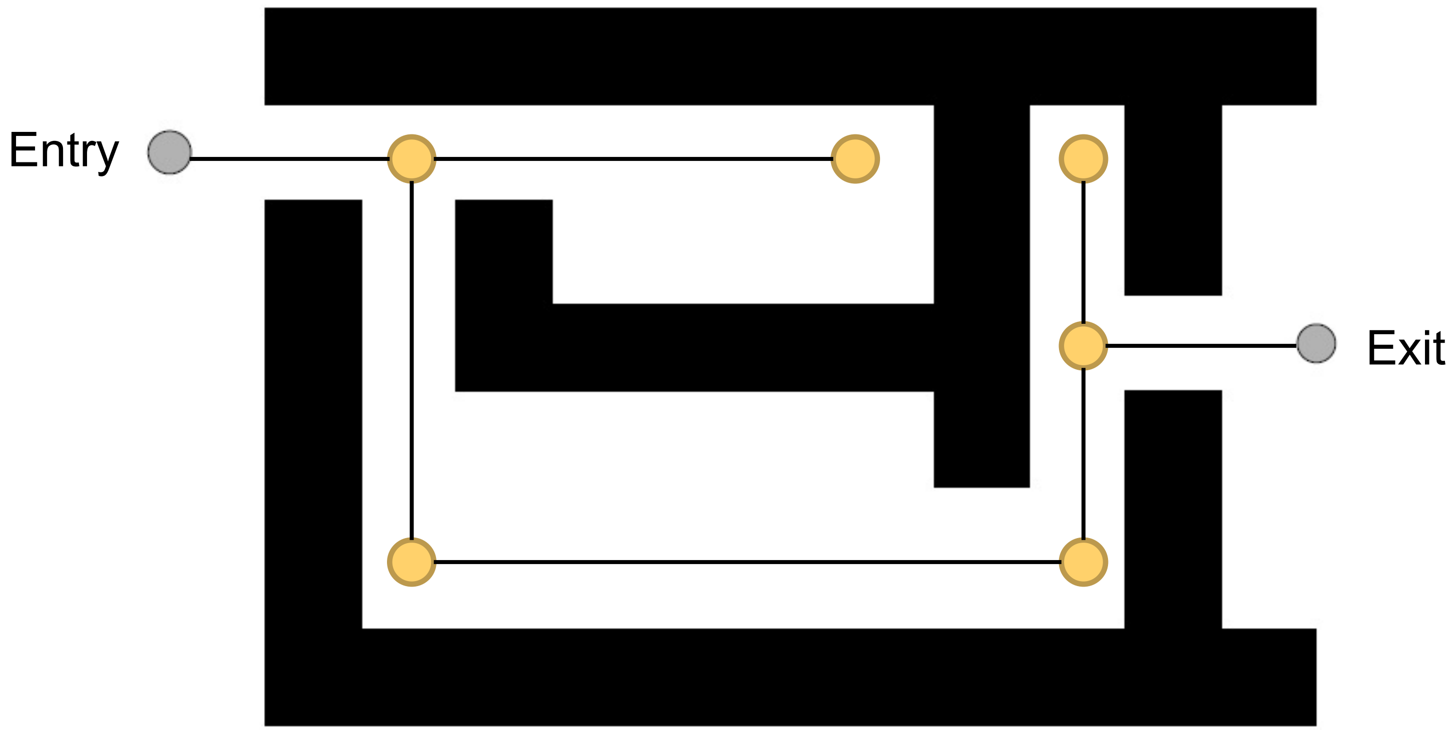


Das Haus vom Nikolaus



Labyrinth

Find a way through the labyrinth!



General Example/Terms

Vertices (*Knoten*): $V = \{a, b, c, d\}$

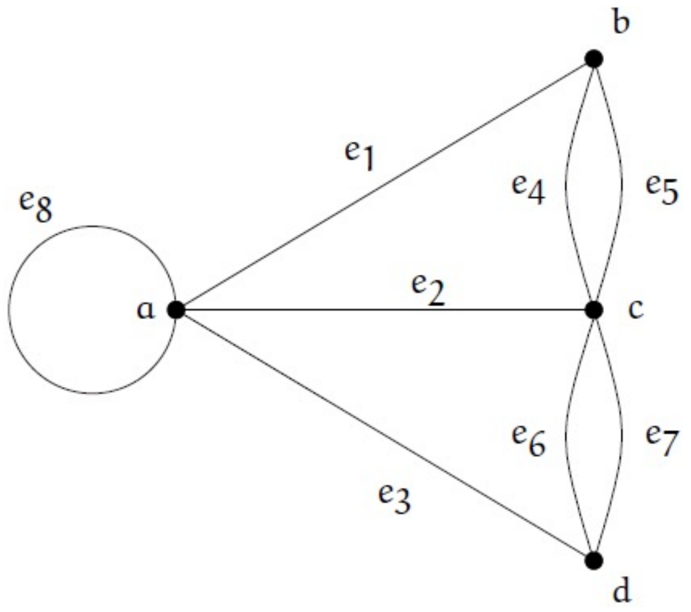
Edges (*Kanten*): $E = \{e_1, e_2, \dots, e_8\}$

Incidence mapping (*Inzidenzabbildung*):

$$I = \{(e_1, \{a, b\}), (e_2, \{a, c\}), \dots, (e_8, \{a\})\}$$

defines which edges connect which vertices

Loop (*Schlinge*):
Edge is incident on a single vertex



parallel edges:
Edges are incident on the same vertex

Simple (*schlichter*) graph: has neither loops nor parallel edges

Definition: Graph

A (non-directed) graph G consists of a

- set of vertices (or nodes, *Knoten*) V
- set of edges (*Kanten*) E
- incidence mapping I that maps edges to vertices

- Adjacency (*Adjazenz*)
  - the two nodes a, b of an edge e are called adjacent
- Incidence (*Inzidenz*)
  - the edge e that connects the nodes a, b, is incident on a and b

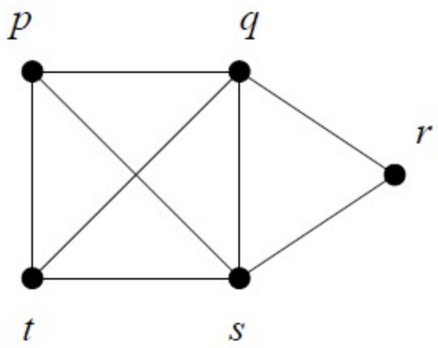
- If V is countably infinite, then G is called an infinite graph

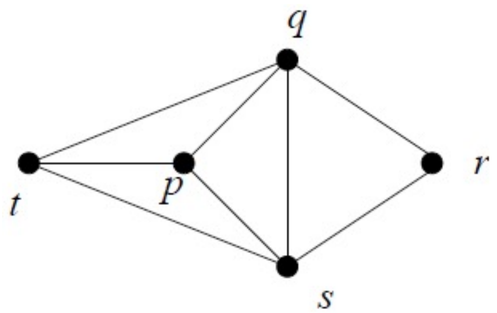
Visualization: Graph Drawing

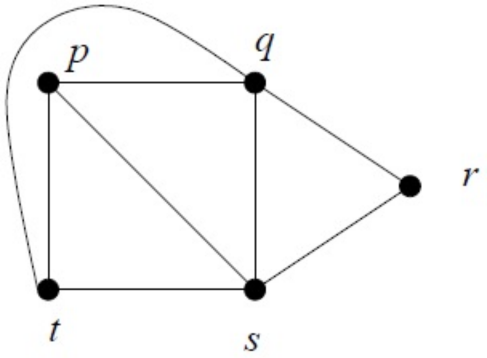
- Graphs are visualized by drawing vertices and edges
- There are many different diagrams for the same graph – don’t confuse the graph with its drawing

$V = \{p, q, r, s, t\}$

$E = \{\{p, q\}, \{p, s\}, \{p, t\}, \{q, r\}, \{q, s\}, \{q, t\}, \{r, s\}, \{s, t\}\}$ Simplified notation for simple graphs



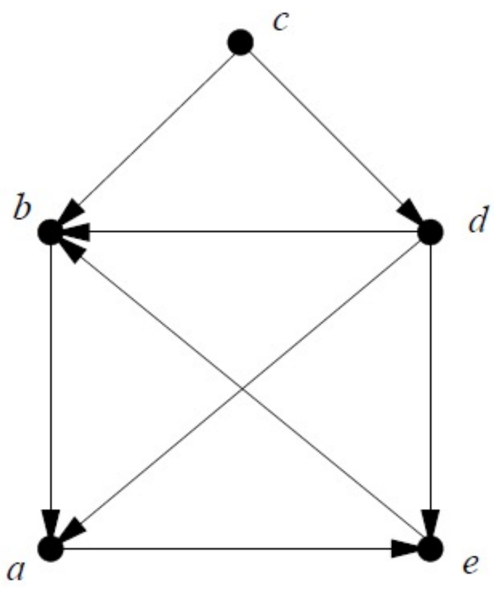




Directed Graphs

A directed (*gerichteter*) graph G consists of a

- set of vertices V
- set of directed edges E
  - consisting of ordered pairs of vertices (a, b) $\in V \times V$
  - a is called start node
  - b is called end node

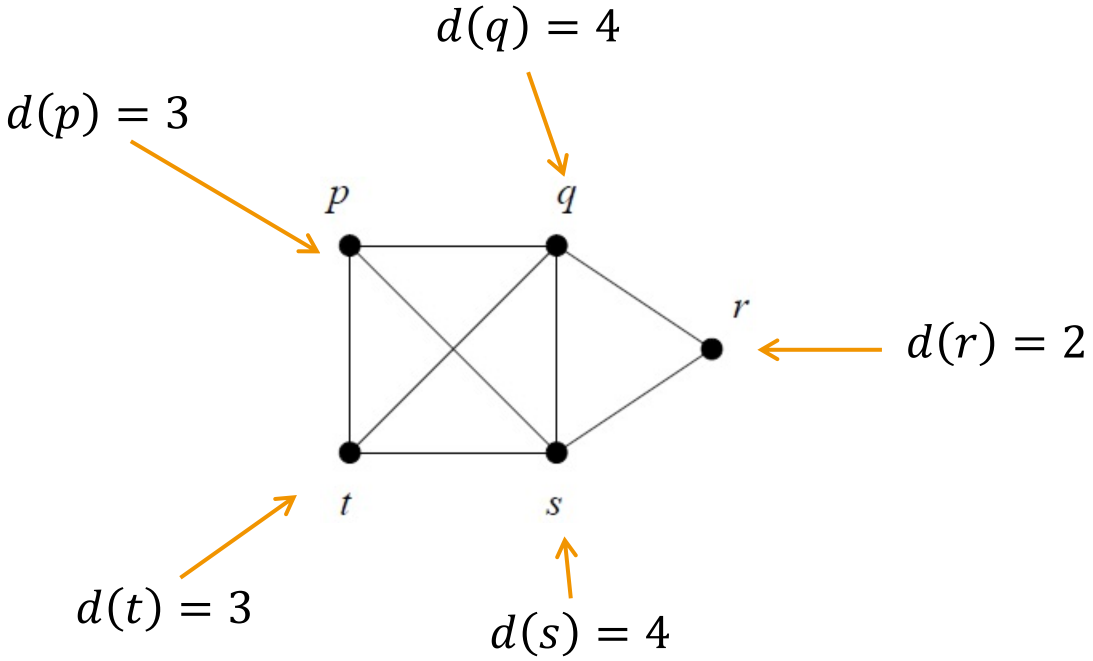


Degree of a Vertex – Undirected Graphs

- degree (*Grad*) of vertex x_i:
  $d(x_i)$ = number of incident edges

- Degree sum formula
  for a graph with n vertices and k edges:

$$\sum_{i=1}^{n} d(x_i) = 2k$$



Degree of a Vertex – Directed Graphs

- outdegree (*Ausgangsgrad*)
  $d^+(x_i)$ = number of edges starting at x_i
- indegree (*Eingangsgrad*)
  $d^-(x_i)$ = number of edges ending at x_i

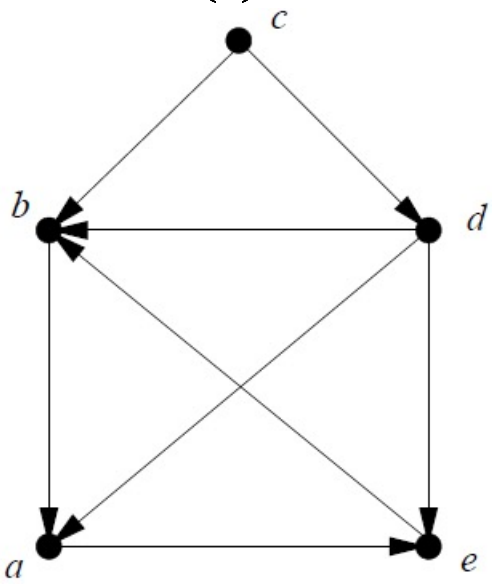
- Degree sum formula
  for a graph with n vertices and k edges:

$$\sum_{i=1}^{n} d^+(x_i) = \sum_{i=1}^{n} d^-(x_i) = k$$

$d^+(c) = 2$
$d^-(c) = 0$

$d^+(b) = 1$
$d^-(b) = 3$

$d^+(d) = 1$
$d^-(d) = 3$



$d^+(a) = 1$
$d^-(a) = 2$

$d^+(e) = 1$
$d^-(e) = 2$

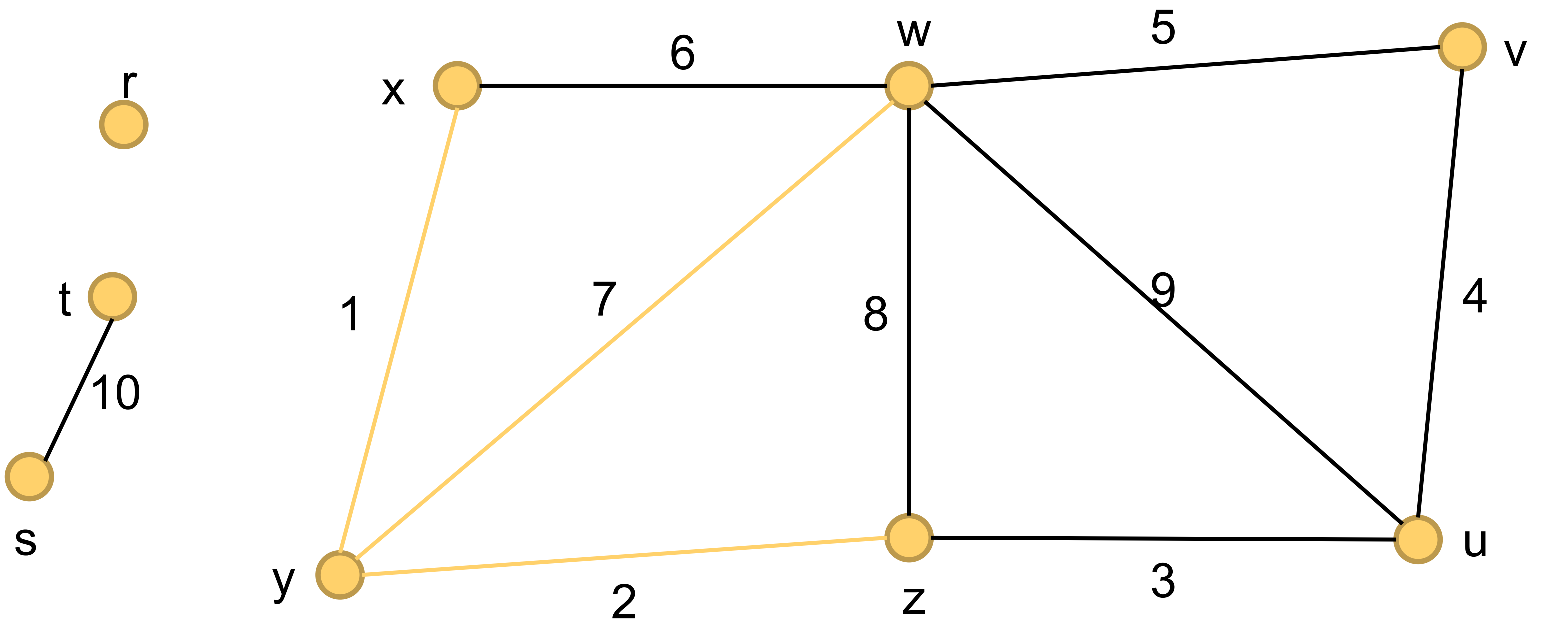
Complete Graph

- a graph is called complete (*vollständig*) if there is an edge from each node to each other
- a complete (undirected) graph with n nodes has $\binom{n}{2}$ edges

Walks

A sequence of adjacent edges from vertex v_0 to v_n $(v_0, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n)$ is called a walk (*Kantenfolge, Kantenzug*) of length n

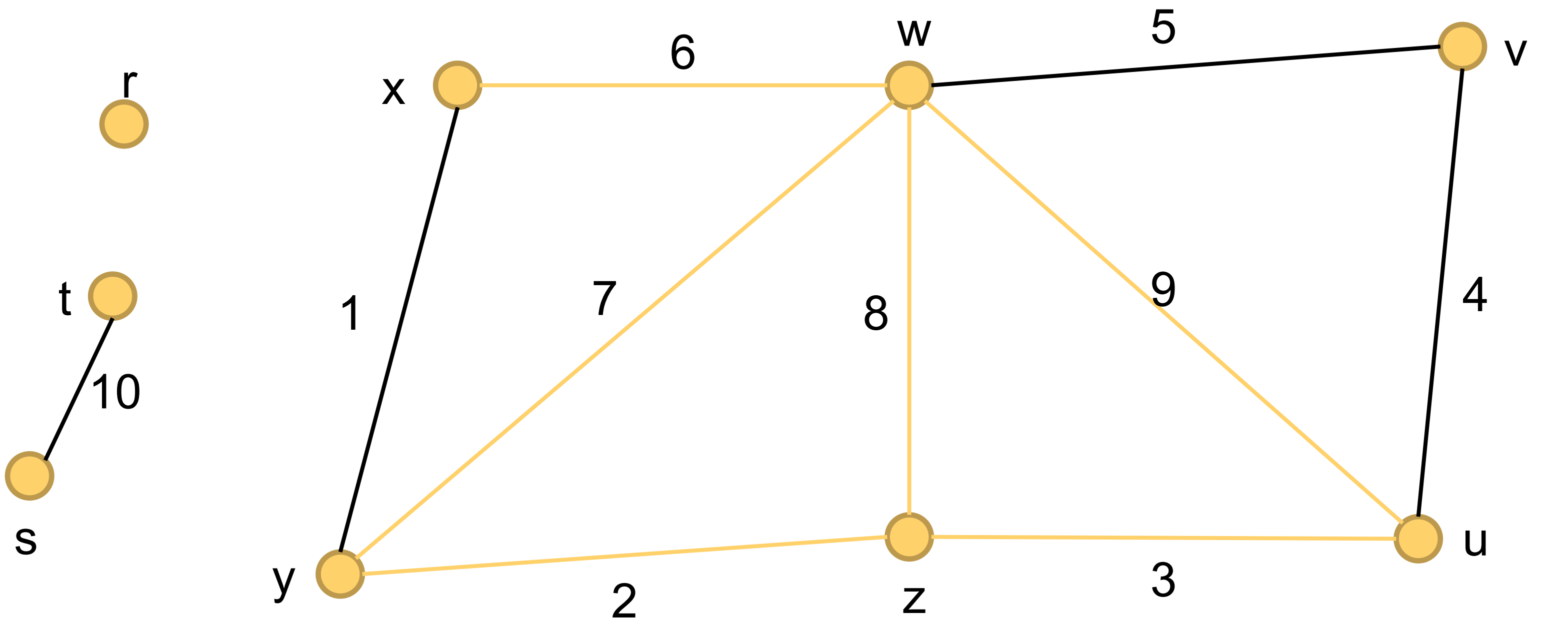
- Edges and vertices may be repeated
- closed walk: $v_0 = v_n$



Walk from x to z (but not a trail or path): 1, 7, 7, 2 (x, y, w, y, z)

Trails

Trail (*Weg*): A walk, where all *edges* are pairwise disjoint

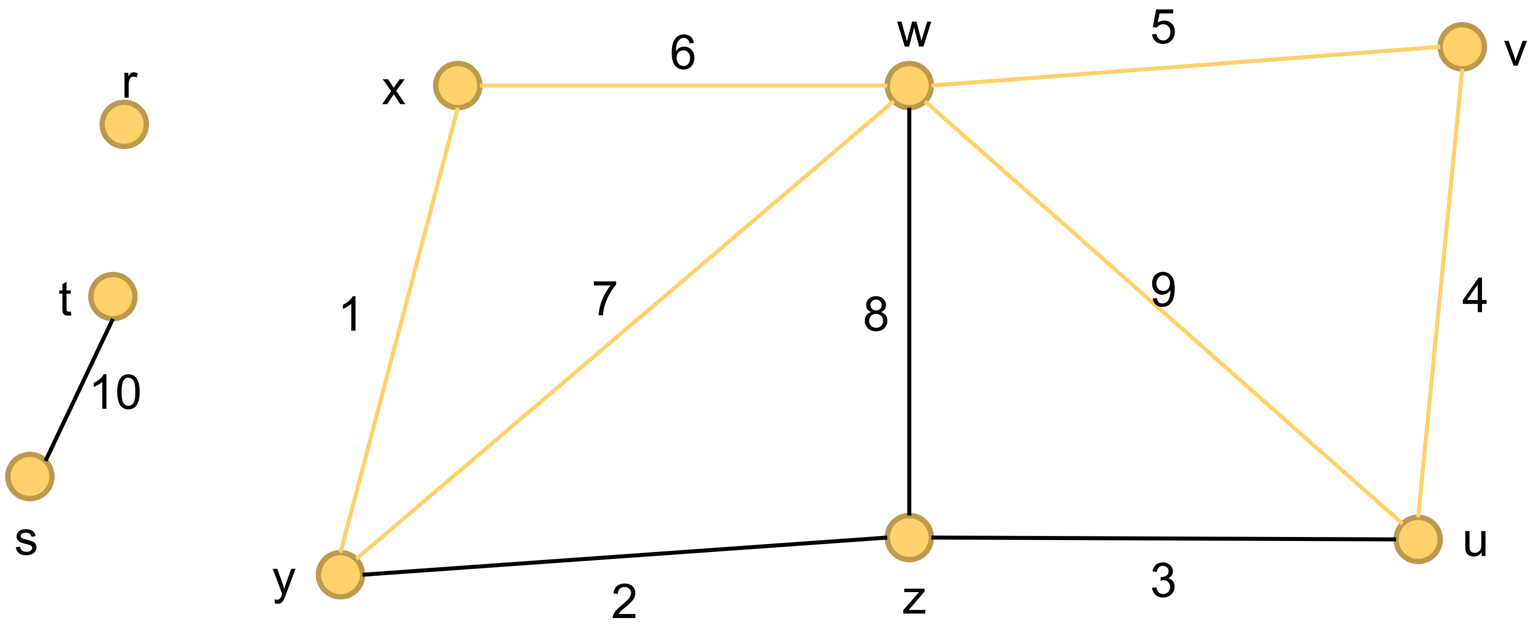


Trail from x to z (but not a path): 6, 8, 3, 9, 7, 2 (x, w, z, u, w, y, z)

Closed Trails

Trail (*Weg*): A walk, where all *edges* are pairwise disjoint

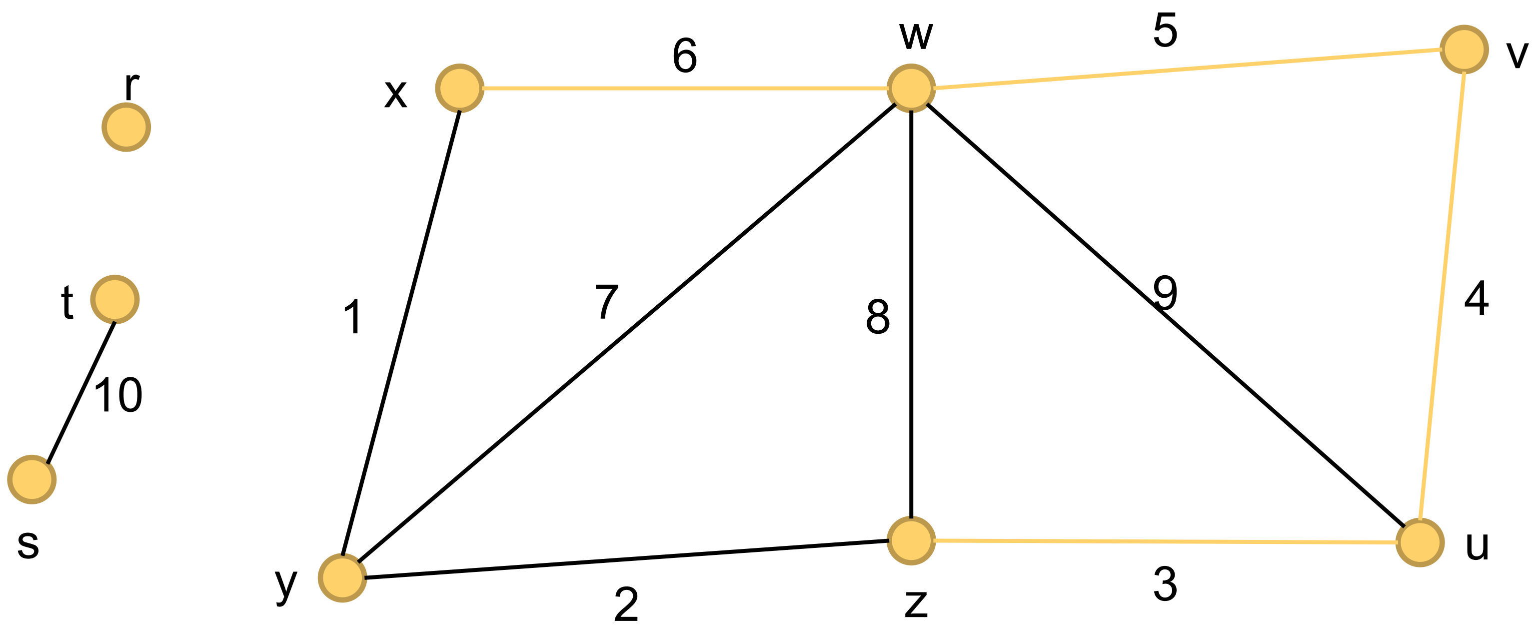
- Closed trail (*Kreis*): $v_0 = v_n$



Closed trail (but not a cycle): 6, 5, 4, 9, 7, 1 (x, w, v, u, w, y, x)

Paths

Path (*Pfad*): A walk, where all *vertices* are pairwise disjoint

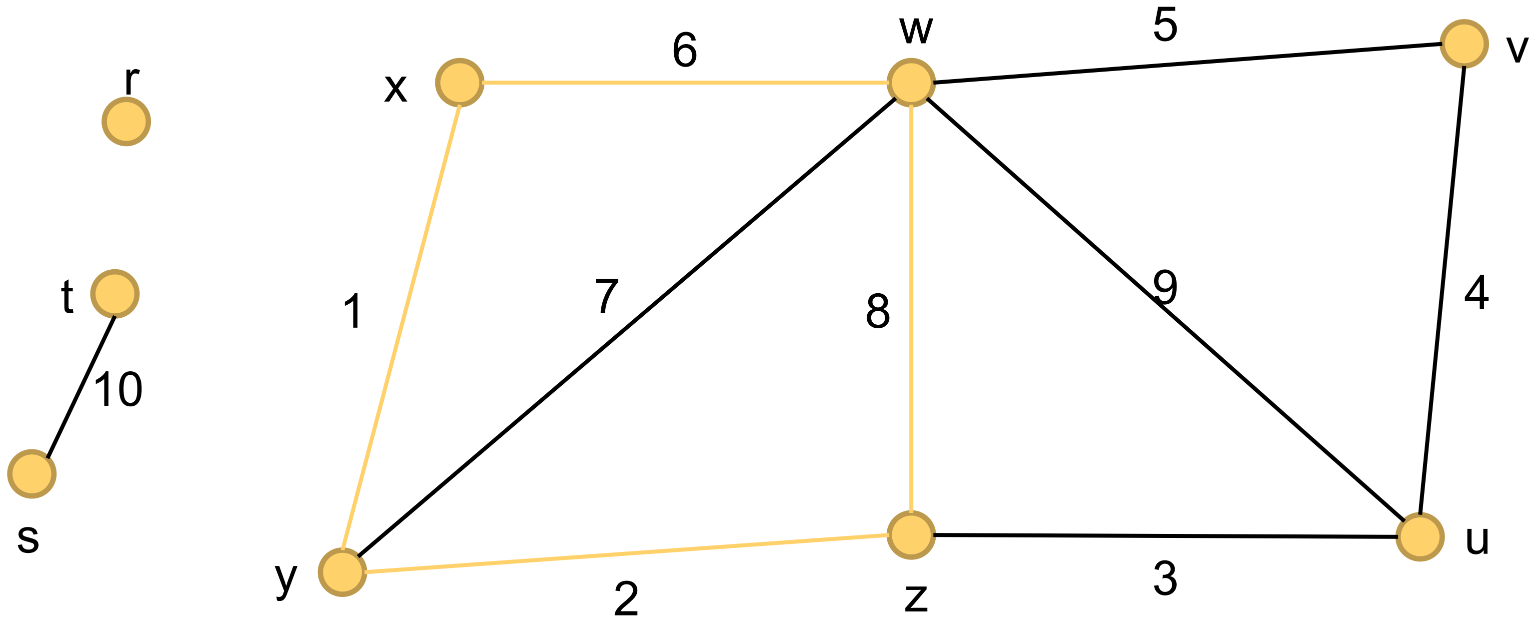


Path from x to z: 6, 5, 4, 3 (x, w, v, u, z)

Cycles

Path (*Pfad*): A walk, where all *vertices* are pairwise disjoint

- Cycle (*Zyklus*): closed path $v_0 = v_n$
  (Start-/end nodes are exempt from the rule that all vertices must be pairwise disjoint)



Cycle: 6, 8, 2, 1 (x, w, z, y, x)

Walks – Summary

- A sequence of adjacent edges from vertex v_0 to v_n $(v_0, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n)$ is called a walk (*Kantenfolge, Kantenzug*) of length n
  - edges and vertices may be repeated
  - closed walk: $v_0 = v_n$

- Trail (*Weg*): A walk, where all *edges* are pairwise disjoint
  - Closed trail (*Kreis*): $v_0 = v_n$

- Path (*Pfad*): A walk, where all *vertices* are pairwise disjoint
  - Cycle (*Zyklus*): closed path $v_0 = v_n$
    (Start-/end nodes are exempt from the rule that all vertices must be pairwise disjoint)

Note: These terms are not used consistently in the literature

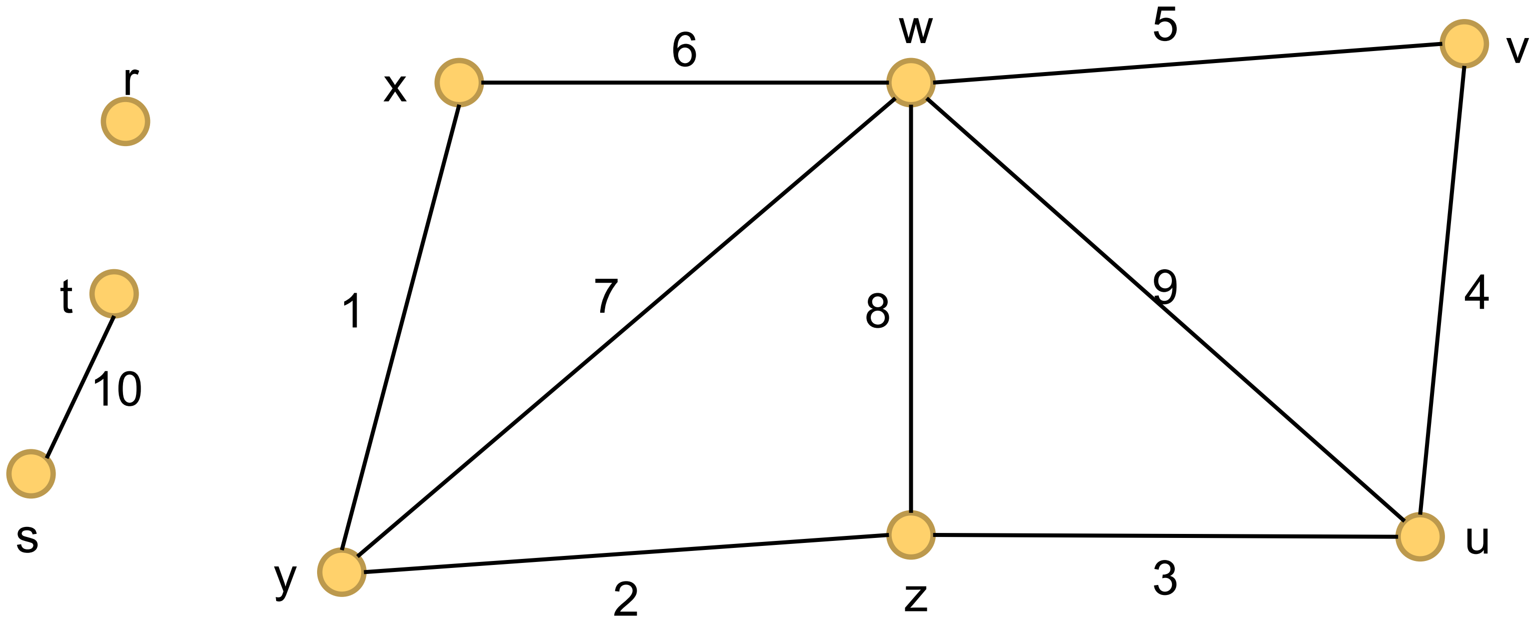
Connection

- Two nodes *v, w* are called connected (*verbunden*) if there is a path from *v* to *w*

- A graph G is said to be connected (*zusammenhängend*) if and only if all pairs of vertices of G are connected
  - each connected graph with n vertices has at least $n-1$ edges

- A connected component (*Zusammenhangskomponente*) of G is a connected subgraph $G(U)$ induced by a set of vertices $U \subseteq V$ that has the maximum number of vertices possible

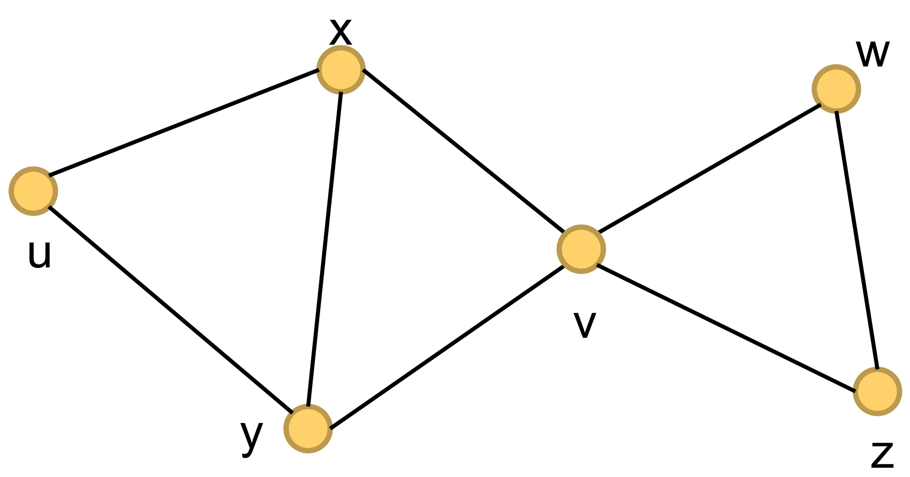
Connection – Example

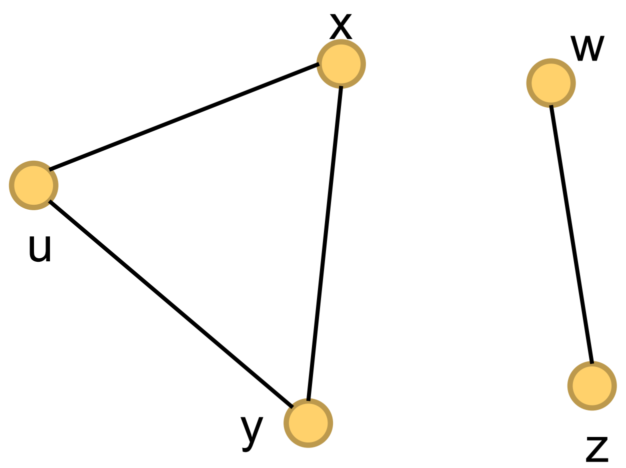


- r is an isolated vertex
- s and t are connected
- s and y are not connected
- This is **one** graph. The graph is disconnected
- it consists of three connected components
  - {r}
  - {s, t}
  - {x, y, z, u, v, w}

Vertex Separator

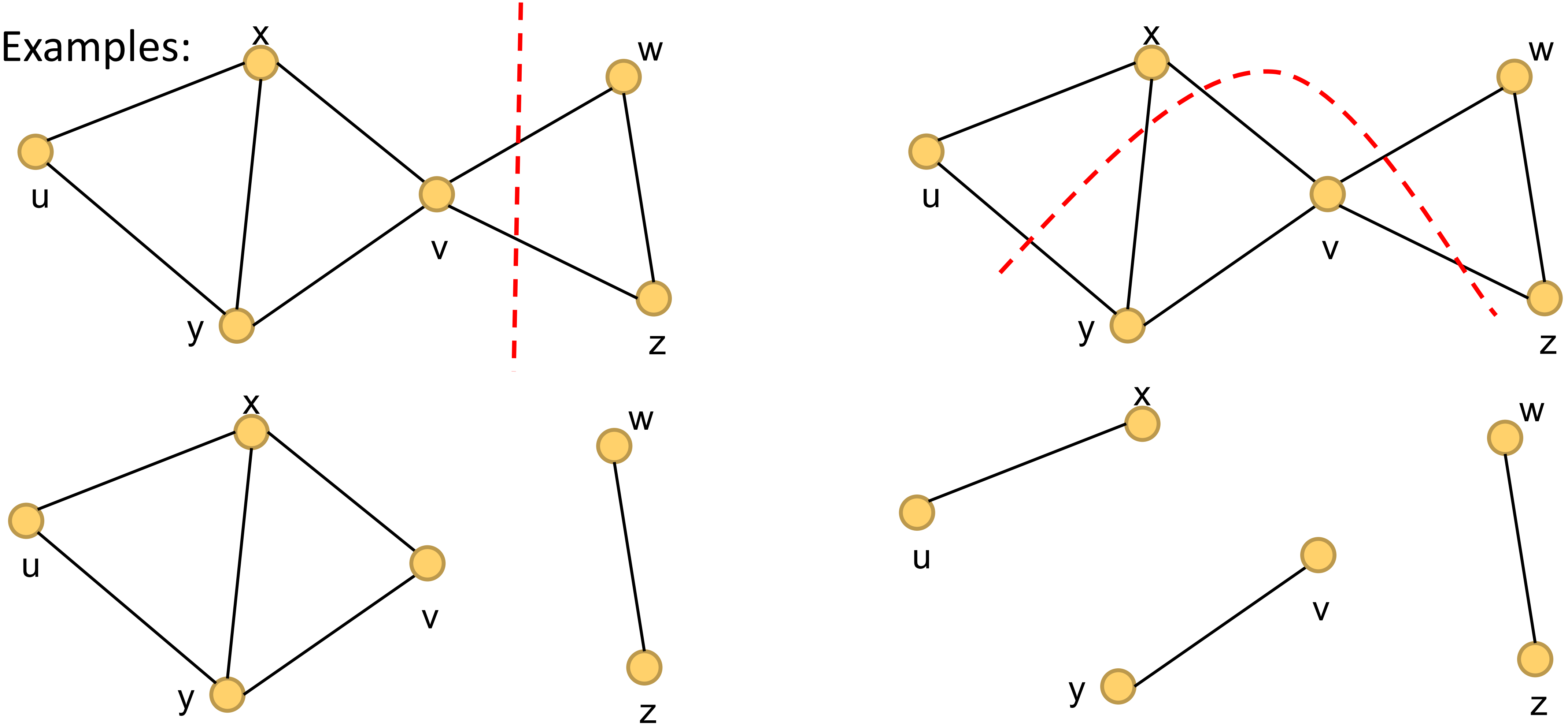
- A vertex separator (*Trenner*) is a subset of vertices of a graph that separates the graph into distinct connected components if removed (together with the incidental edges).
- Special case: A single vertex is called separating (*trennend*) if, after removing this vertex (and the incidental edges), the residual graph has more components than before.
- Examples
  - there are no separating nodes in the graph on the previous slide
  - in the following graph, only v is a separating vertex:





Cut

- A cut (*Schnitt*) is a partitioning of a graph induced by removing a set of edges (cut-set)
- Examples:



Graph Isomorphism

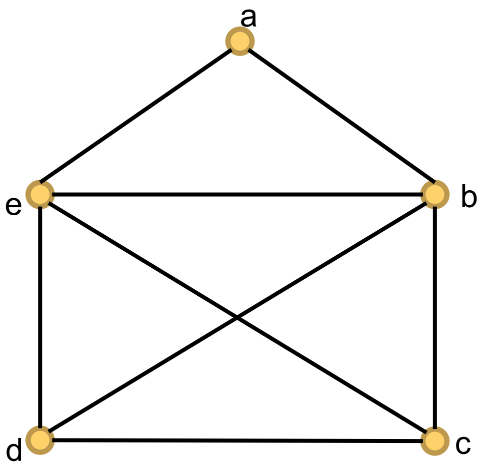
- Two graphs G_1 and G_2 are said to be isomorphic (*isomorph*) if and only if there exists a bijective mapping h of vertex set V_1 to V_2 such that

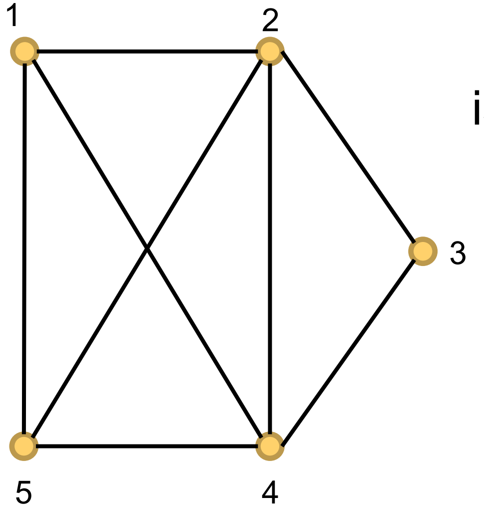
  $\forall v, w \in V_1 : \{v, w\} \in E_1 \Leftrightarrow \{\mathrm{h}(v), \mathrm{h}(w)\} \in E_2$

  - i.e., if (v, w) is an edge of G_1 then $(\mathrm{h}(v), \mathrm{h}(w))$ is an edge of G_2
  - G_2 emerges from G_1 by renaming the vertices
  - isomorphic graphs have the same properties

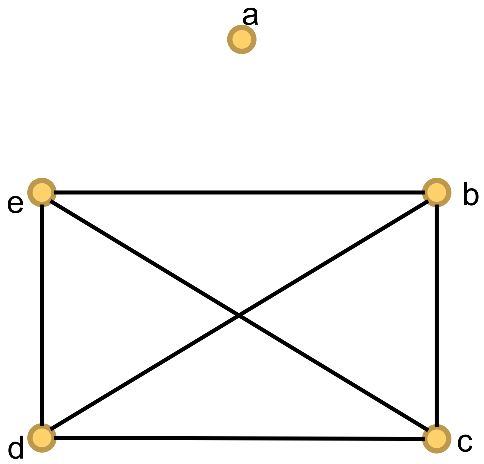
- h is called isomorphism and denoted as $G_1 \simeq G_2$

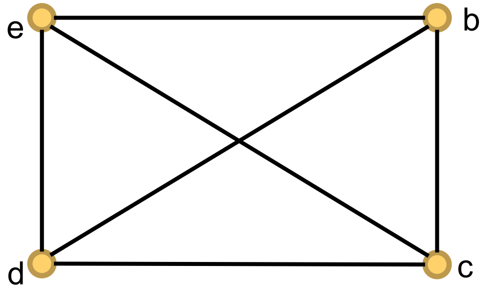
Graph Isomorphism – Example





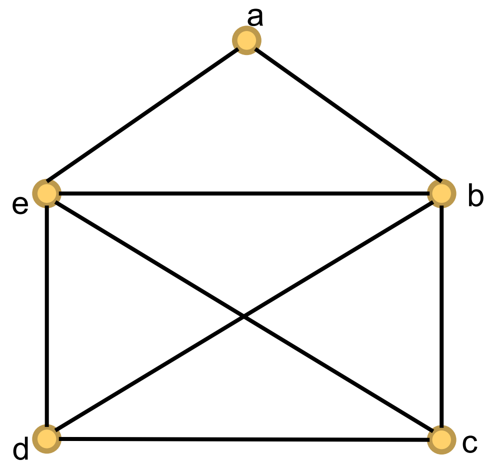
isomorphic: nodes renamed (and drawn rotated)

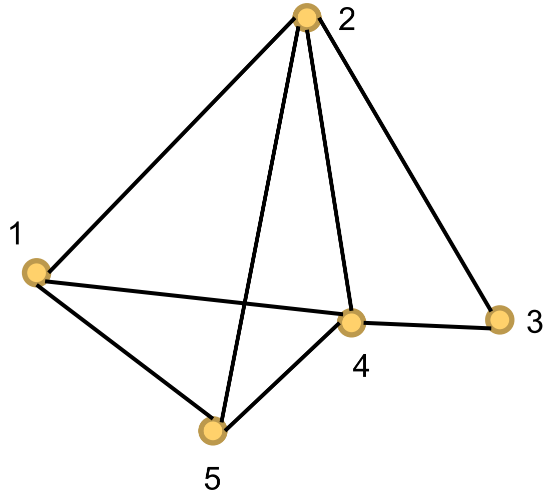




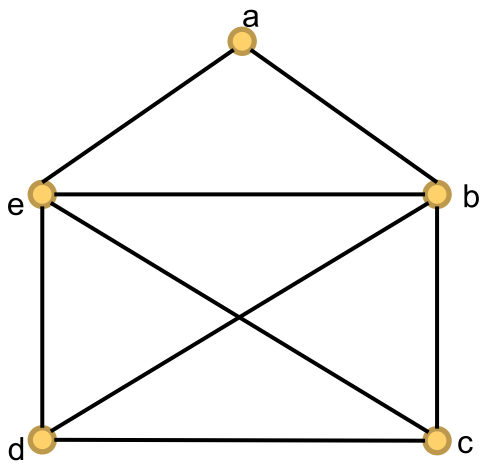
not isomorphic:
different number of vertices and edges

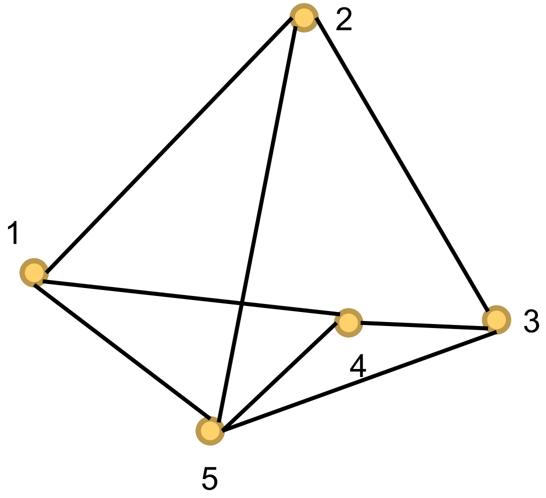
Graph Isomorphism – Example





isomorphic: vertices renamed and moved



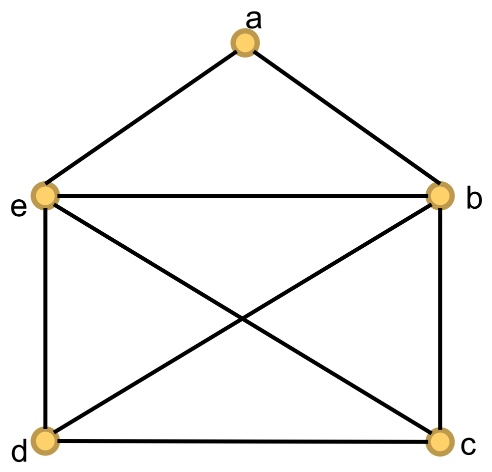


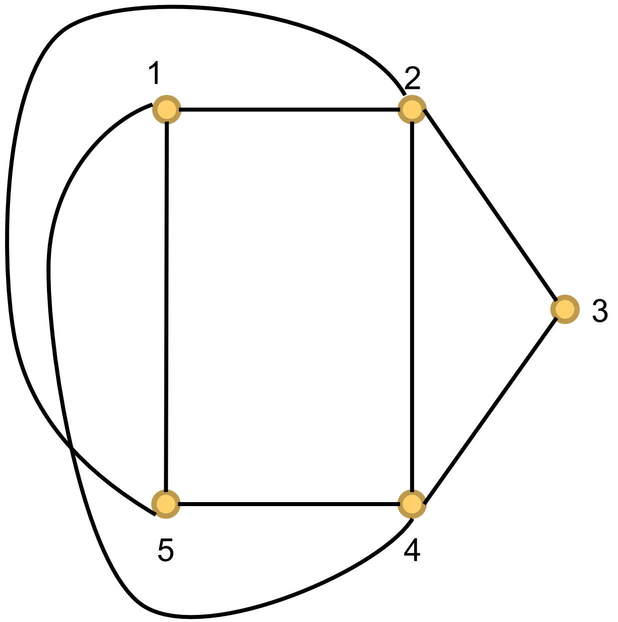
not isomorphic:
same number of vertices & edges,
but connected differently

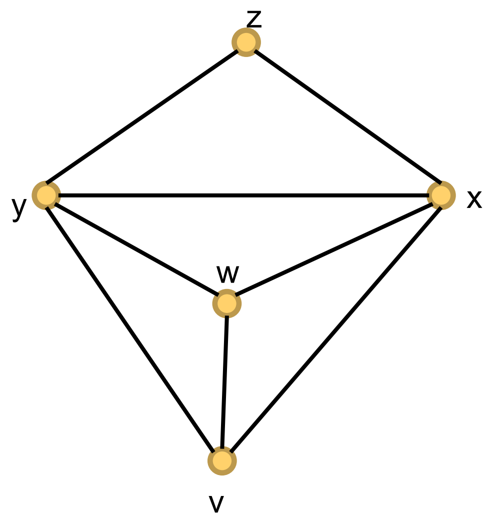
Graph Isomorphism – Exercise

Technische Hochschule Rosenheim

Which of the following graphs are isomorphic?
In case of isomorphism, give a bijective mapping of the vertices!

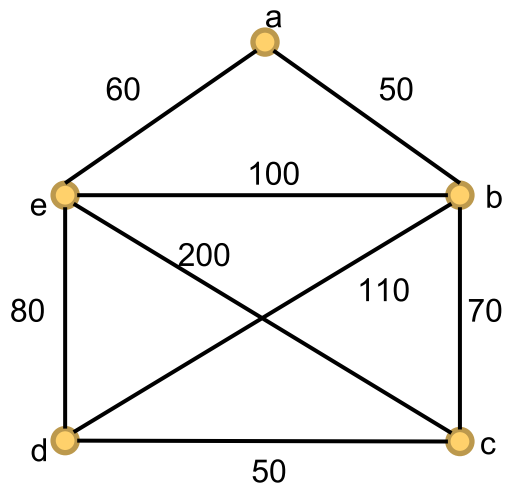






Weighted Graphs

- Assign values to the edges of a graph: this is called a weighted (gewichteter) graph
- Examples:
  - Distances/Lengths
  - Time
  - Costs
  - Probabilities

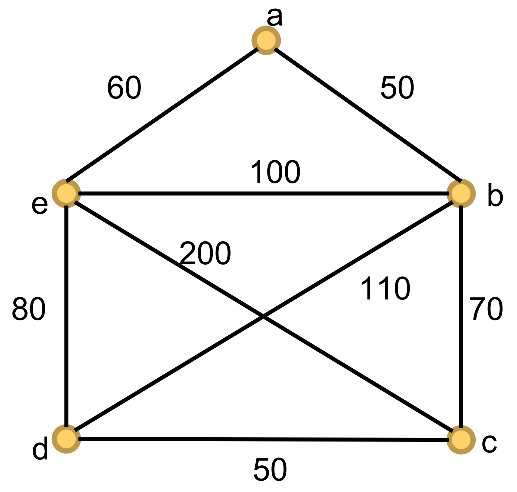


- in some cases, negative weights can be useful
  - however, these lead to problems with distance calculations
  - therefore, it is assumed here that weights are not negative
- unweighted graph: special case, all weights equal 1

Weighted Graphs

- Length of a walk: Sum of all edge weights

- Distance $d(v, w)$ of two vertices v, w:
  - Minimum of all walks from v to w
  - If there is no walk: $d(v, w) = \infty$



Distance between a and d: 140

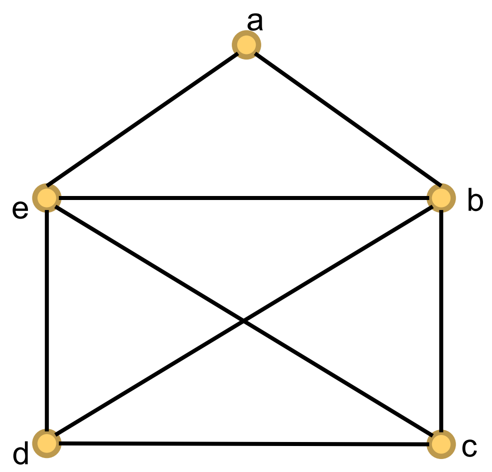
Adjacency Matrix

- Representation of a graph in matrix form
- Graph with n edges results in an n x n matrix $\boldsymbol{A}$
- The elements a_{ij} of $\boldsymbol{A}$ for a given labeling of the vertices are

$$a_{ij} = \begin{cases} 1, \text{if } (x_i, x_j) \text{is an edge of the graphen} \\ 0 \qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases}$$

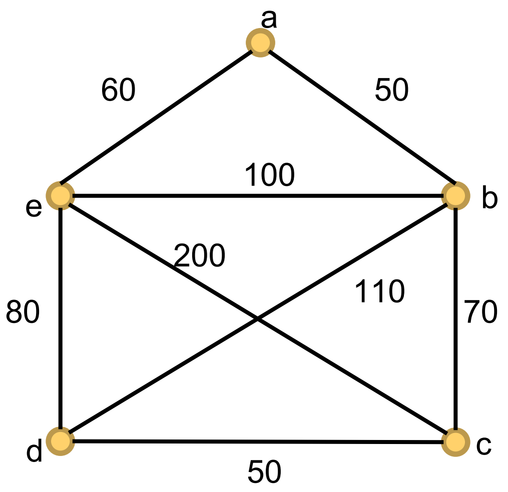
- $\boldsymbol{A}$ is called adjacency matrix (*Adjazenzmatrix*) of the graph
  - symmetric for non-directional graphs
  - in general asymmetric for directed graphs
  - weighted graphs: Use edge weights instead of 0 and 1

Adjacency Matrix – Example



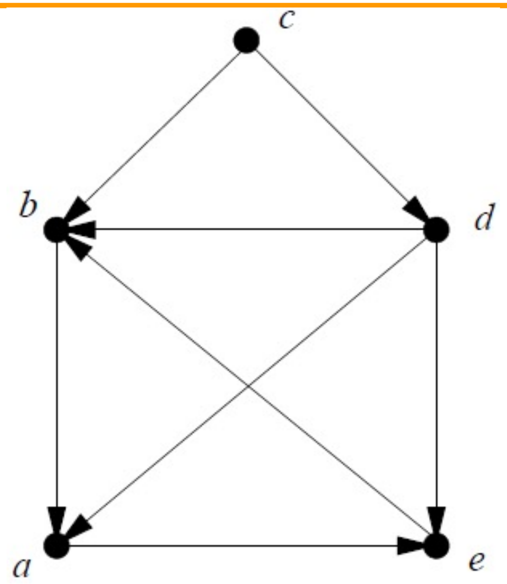
$$
\begin{array}{cc}
 & \begin{array}{ccccc} a & b & c & d & e \end{array} \\
A = & \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix} \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array}
\end{array}
$$

order of vertices is arbitrary!

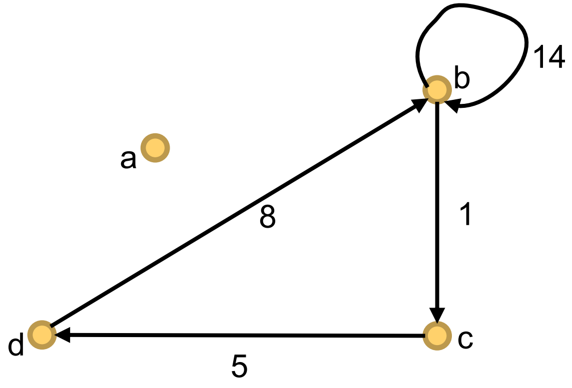


$$
A = \begin{pmatrix} 0 & 50 & 0 & 0 & 60 \\ 50 & 0 & 70 & 110 & 100 \\ 0 & 70 & 0 & 50 & 200 \\ 0 & 110 & 50 & 0 & 80 \\ 60 & 100 & 200 & 80 & 0 \end{pmatrix}
$$

Adjacency Matrix – Example/Exercise



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



$$A = ?$$

Adjacency Matrix – Directed Graphs

- Powers $\boldsymbol{A}^r$ of the adjacency matrix $\boldsymbol{A}$ give us information about existence and number of walks in directed graphs

- Number of different walks of length r from x_i to x_j = element a_{ij} of matrix $\boldsymbol{A}^r$

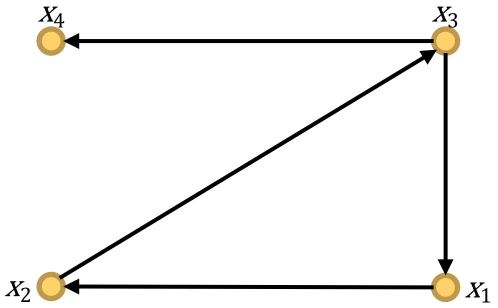
- Graph with n vertices is acyclic (*azyklisch*), if there exists an r with $1 \leq r < n$ such that:
  $\boldsymbol{A}^r \neq \mathbf{0}$, but $\boldsymbol{A}^s = \mathbf{0} \ \forall \ s > r$

  This is said to be a Directed Acyclic Graph (DAG)

Powers of the Adjacency Matrix – Example

Technische Hochschule Rosenheim

Graph with n vertices is acyclic (*azyklisch*), if there exists an r with $1 \leq r < n$ such that $\boldsymbol{A}^r \neq \mathbf{0}$, but $\boldsymbol{A}^s = \mathbf{0}\ \forall\ s > r$



$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\boldsymbol{A}^2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \boldsymbol{A}^3 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \boldsymbol{A}^4 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

all powers $\boldsymbol{A}, \boldsymbol{A}^2, \boldsymbol{A}^3, \boldsymbol{A}^4$ are unequal $\mathbf{0} \Leftrightarrow$ graph has cycles

Path Matrix

- The path matrix (*Wegematrix*) $\boldsymbol{W}$ indicates whether a path from x_i to x_j exists:

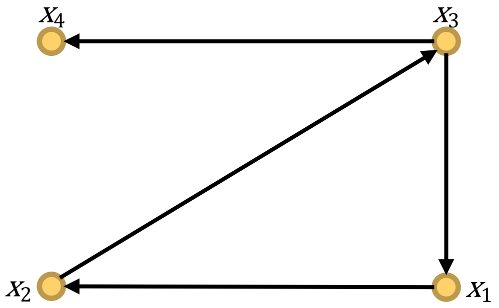
$$w_{ij} = \begin{cases} 1, \text{if there exists a path from } x_i \text{ to } x_j \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad \text{otherwise} \end{cases}$$

- $\boldsymbol{W}$ can be obtained by adding up all relevant powers of the adjacency matrix

$$\boldsymbol{A} + \boldsymbol{A}^2 + \boldsymbol{A}^3 + \ldots + \boldsymbol{A}^n$$

  and replacing all non-zero elements by 1

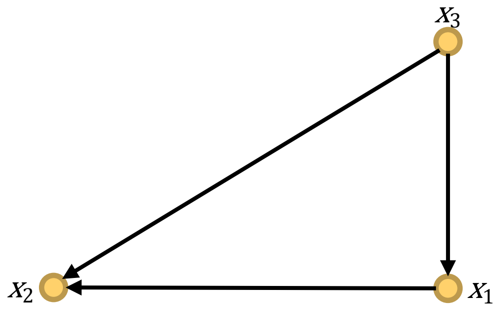
Path Matrix – Example



$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A^2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A^3 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad A^4 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$A + A^2 + A^3 + A^4 = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 1 & 1 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix} \longrightarrow W = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Path Matrix – Exercise



1. Determine the adjacency matrix and its powers.
2. What can be said about the graph from these results?
3. Determine the path matrix

Data Structures

- Adjacency matrix
  - undirected graphs: it is sufficient to store half the matrix
  - often contains many zeros

- Adjacency list
  - Linked list of vertices
  - for each vertex: contains a linked list of its neighbors
  - more compact than the adjacency matrix

Adjacency List – Example