ifp

11.12.2019

- Graphs
  - A Graph G(N, E) consists of a set of Nodes N and Edges E
  - A node is on the position where an edge starts or ends or several edges meet
  - Every edge is a connection between two nodes. Every edge has a start and an end node

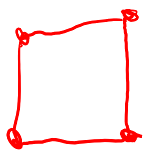
Universität Stuttgart

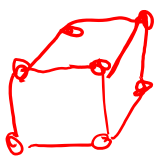
ifp

Topological Primitives

 node dimension 0

 edge " 1

 mesh/face " 2

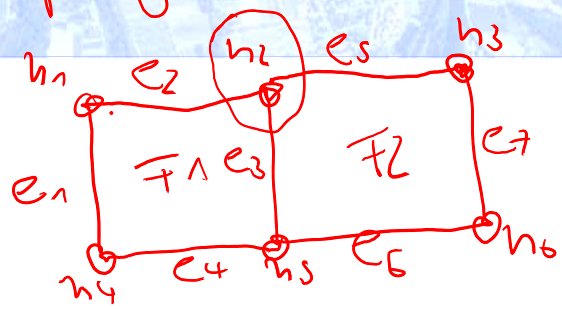
 body " 3

Universität Stuttgart

ifp

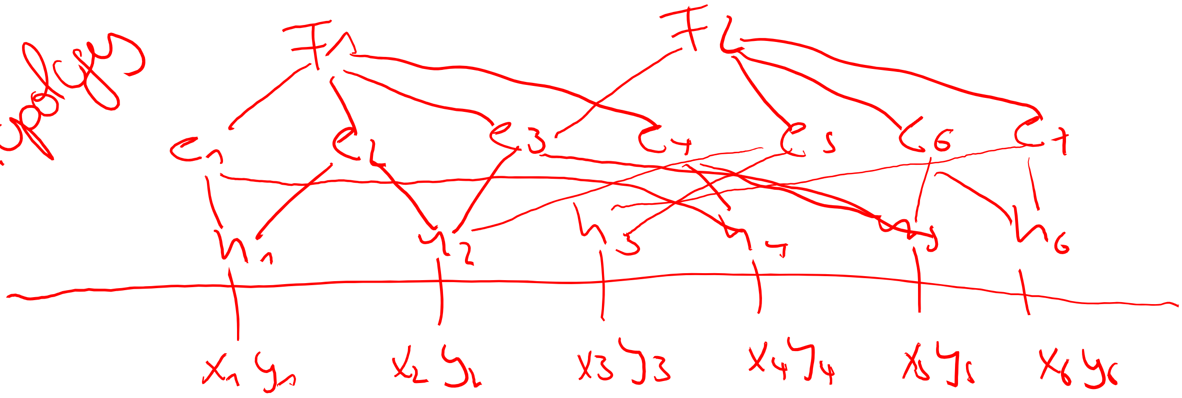
Universität Stuttgart

Topological structured data



redundance free storage of coordinates

Topology



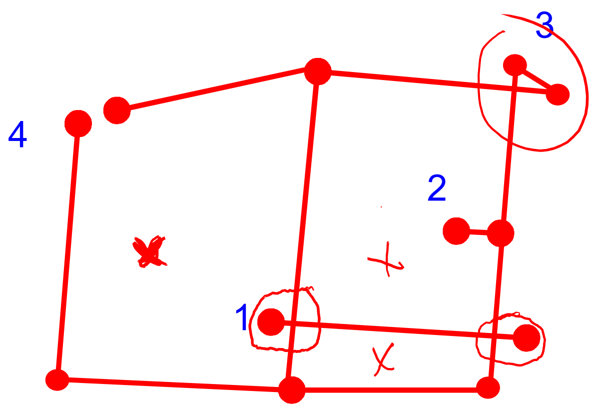
---

ifp

Universität Stuttgart

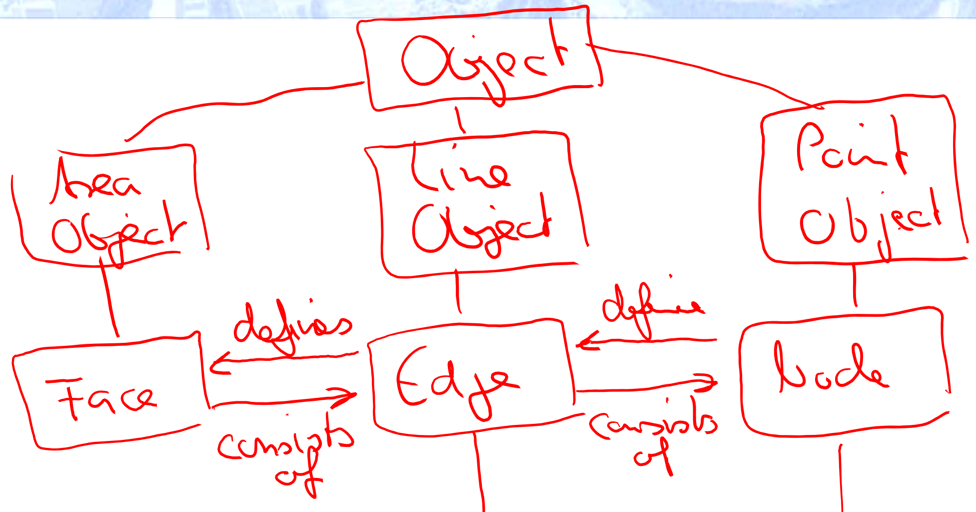
Importance of Topology

- description of relations between elementary objects (node, edge, area)
- important for consistency checks:
  - under- / overshoots (1)
  - dead ends (2)
  - weird polygons (3)
  - closed polygons (4), ...



ifp

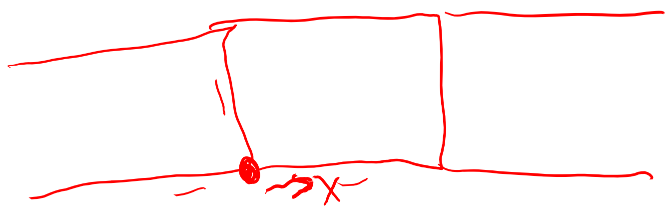
Topological structured data model



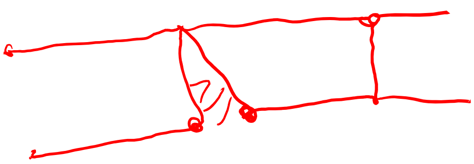
Universität Stuttgart

ifp

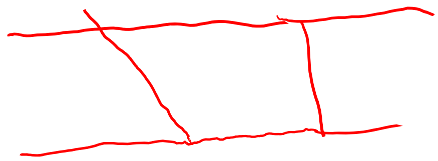
Redundance free storage of coordinate



without topology



without topology



Universität Stuttgart

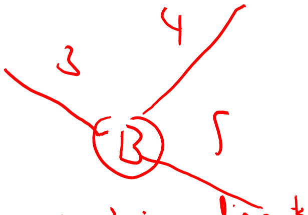
ifp

How can topology be stored in a computer

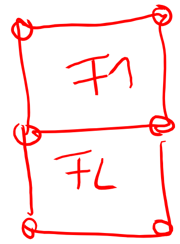
- Topologic spaces can be described with incidence and adjacency relations
- Incidence names the meeting of different topologic primitives
- Adjacency names the meeting of same topologic primitives

Examples

edge 1 incidences node A
edge 2 " " A
edge 1 is adjacent to edge 2



edge 3 is adjacent to edge 4 and 5
node B incidences edge 3,4,5



face 1 is adjacent to face 2

Universität Stuttgart

ifp

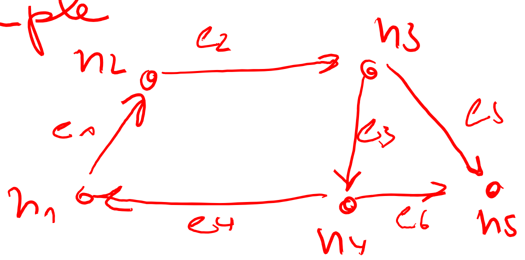
Incidence Matrix

$B(i,j) = -1$ if edge i starts at node j

$B(i,j) = 1$ if edge i end at node j

$B(i,j) = 0$

Example



	n_1	n_2	n_3	n_4	n_5
e_1	-1	+1	0	0	0
e_2	0	-1	1	0	0
e_3	0	0	-1	1	0
e_4	1	0	0	-1	0
e_5	0	0	-1	0	1
e_6	0	0	0	-1	0

Universität Stuttgart

ifp

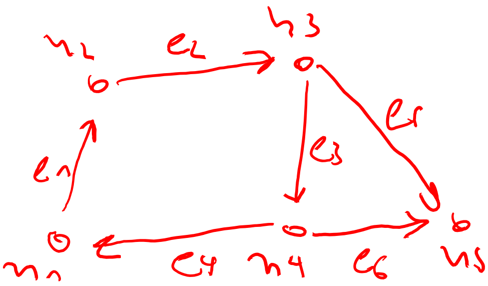
Universität Stuttgart

Adjacency Matrix

The main diagonal contains the number of adjacent nodes

nodes which are adjacent to each other ~~are~~ marked with -1

Adjacency matrix is symmetrical



	n_1	n_2	n_3	n_4	n_5
n_1	2	-1	0	-1	0
n_2	-1	2	-1	0	0
n_3	0	-1	3	-1	-1
n_4	-1	0	-1	3	-1
n_5	0	0	-1	-1	2

ifp

Universität Stuttgart

Adjacency Matrix

- The adjacency matrix $\underline{A}$ can be calculated directly from the incidence matrix: $\underline{A} = \underline{B}^T\underline{B}$

Falsch Schema

$$\begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} = \underline{B}$$

$$B^T = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 & 0 & -1 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 \\ -1 & 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{bmatrix} = \underline{A}$$

ifp

Universität Stuttgart

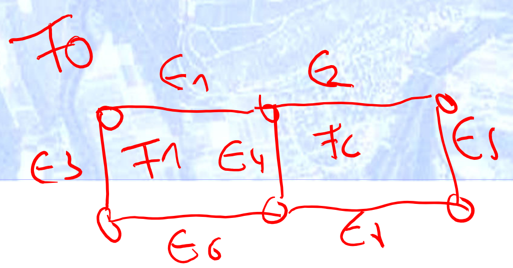
Edge/Node Incidence Matrix

	n1	n2	n3	n4	n5
e1	-1	+1	0	0	0
e2	0	-1	1	0	0
e3	0	0	-1	1	0
e4	0	0	0	-1	1

Node Adjacency

	n1	n2	n3	n4	n5
n1	1	-1	0	0	0
n2	-1	2	-1	0	0
n3	0	-1	2	-1	0
n4	0	0	-1	2	-1
n5	0	0	0	-1	1

ifp
Universität Stuttgart



Edge Adjacency Matrix

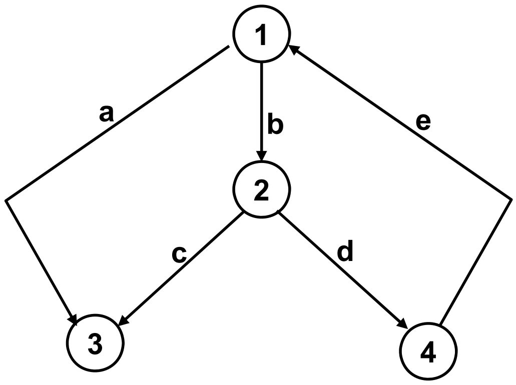
	E1	E2	E3	E4	E5	E6	E7
E1	3	-1	-1	-1	0	0	0
E2	-1	3	0	-1	-1	0	0
E3	-1	0	2	0	0	-1	0
E4	-1	-1	0	4	0	-1	-1
E5	0	-1	0	0	2	0	-1
E6	0	0	-1	-1	0	3	-1
E7	0	0	0	-1	-1	-1	3

Face Adjacency

	F0	F1	F2
F0	2	-1	-1
F1	-1	2	-1
F2	-1	-1	2

Exercise

- Derive the incidence and adjacency matrix from the following graph. Afterwards control the result by calculating A from B

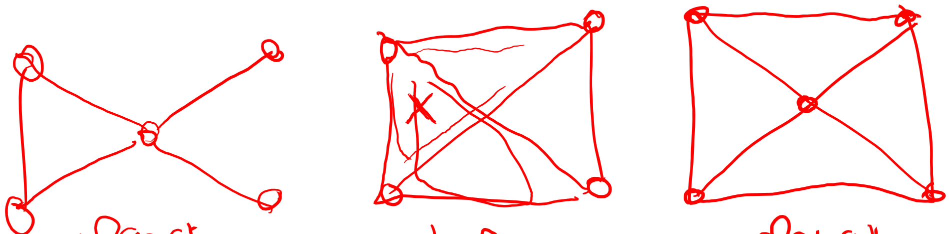


ifp

Universität Stuttgart

Planar graph

A graph is planar if it can be drawn in 2D intersection-free



planar — not planar — planar

Euler equation

For every planar graph:

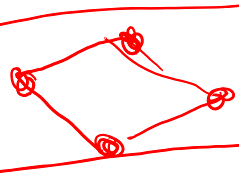
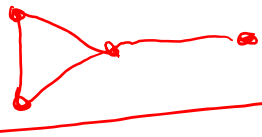
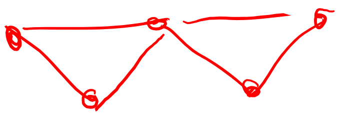
$$C = n - e + f = 2$$

C .. characteristic of ~~the graph~~

n .. number of nodes

e .. " " edges

f .. " " face

graph	n	e	f	n-e+f
	1	0	1	2
	2	1	1	2
	4	4	2	2
	4	4	2	2
	5	6	3	2

Minimal Spanning Tree

- A spanning tree connects all nodes of a graph. A minimal spanning tree is that tree where the length (costs) of the spanning tree is minimal
- Example: communication net between cities. The cities are represented with nodes and the communication net with edges. We are searching for that communication net which is the cheapest.
- A minimal spanning tree has no cycles. That means that there are no redundancies.

Kruskal Algorithm

n ... number of nodes

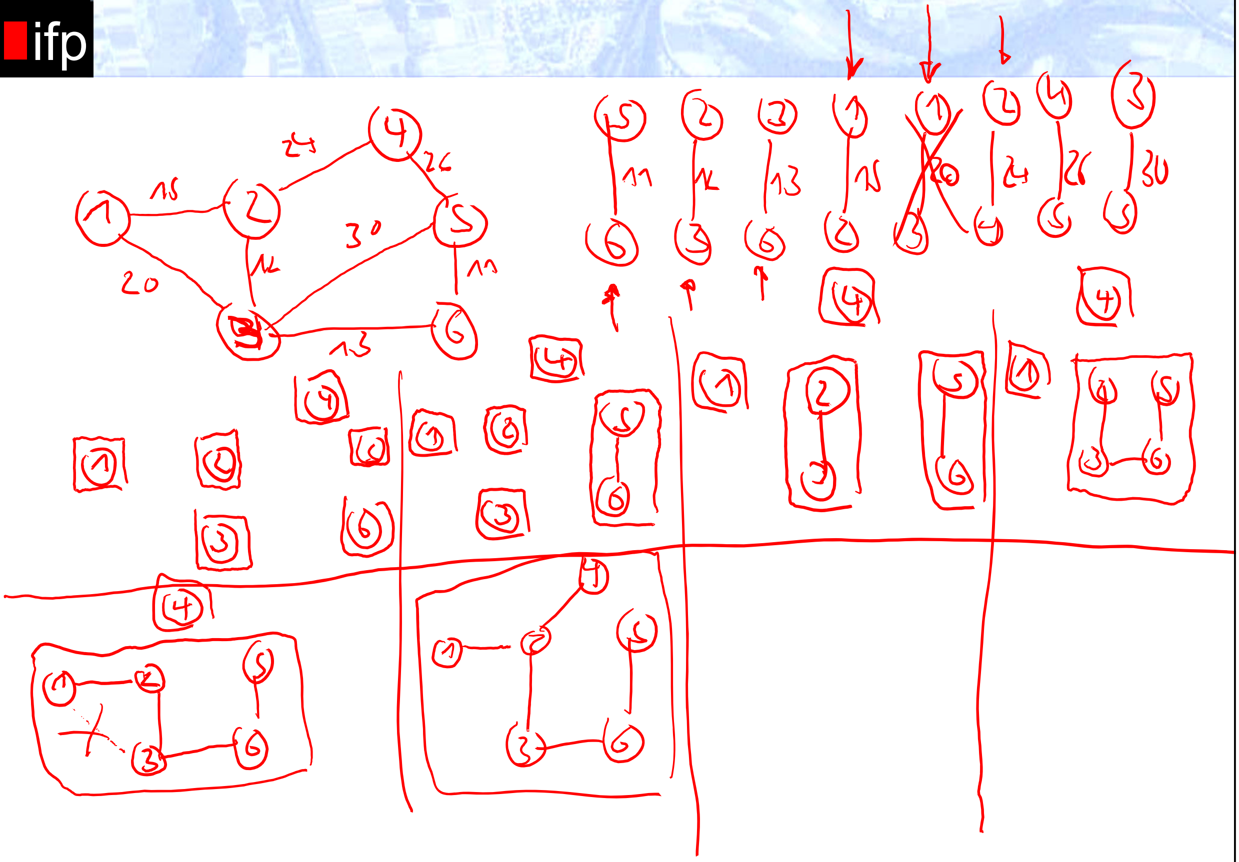
- The algorithm merge successive nodes which are connected by a minimal distance. At the beginning we have *n* sets with one node
- Now we take the shortest edge and check if the two nodes of that edge are in the same set.
  - if yes: this would lead to a cycle and the edge can be discarded
  - if no: merge the two sets which contain the beginning and end node of the edge
- With that algorithm we merge successive all edges until all nodes are connected. That means we have at the end only one set left

Structure Chart Kruskal Algorithm

sort all edges		
store every node in a separate set		
do for all edges (shortest first)		
	are beginning node and end node in different sets?	
	yes	no
	merge sets	discard edge

ifp

Universität Stuttgart



ifp

Universität Stuttgart

Bubblesort

- Input: List L with n not sorted numbers

Repeat n -1 times

For all s from 1 to n-1

L[s] > L[s+1] ?

Yes	No
exchange L[s] with L[s+1]	

- Output: sorted list L

Exampl

L(1) = 14

L(2) = 2

L(3) = 91

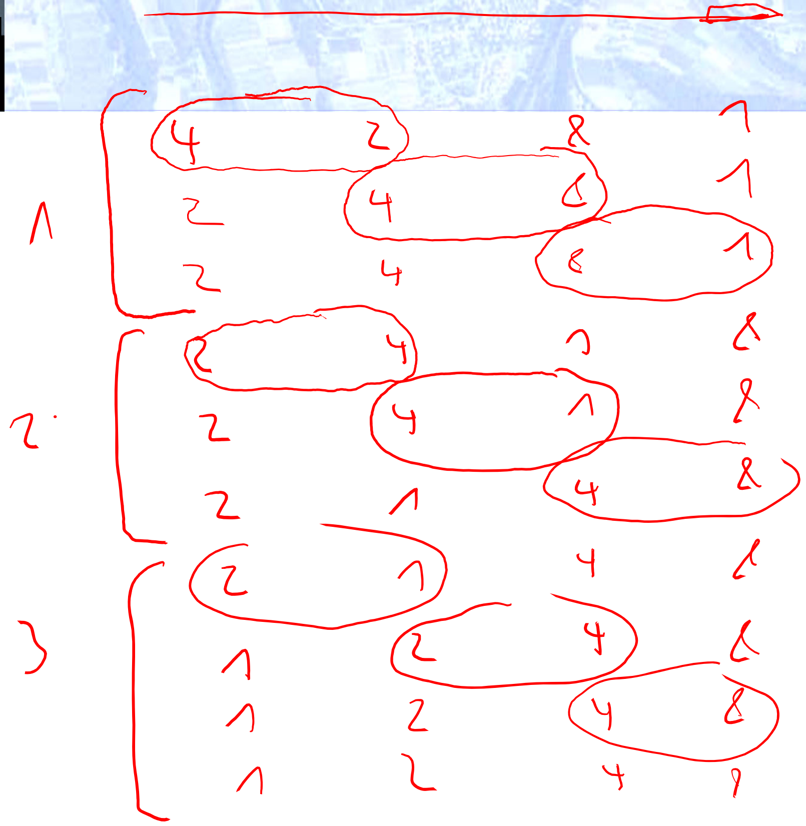
L(4) = 7

L(5) = 9

⋮

ifp

Universität Stuttgart

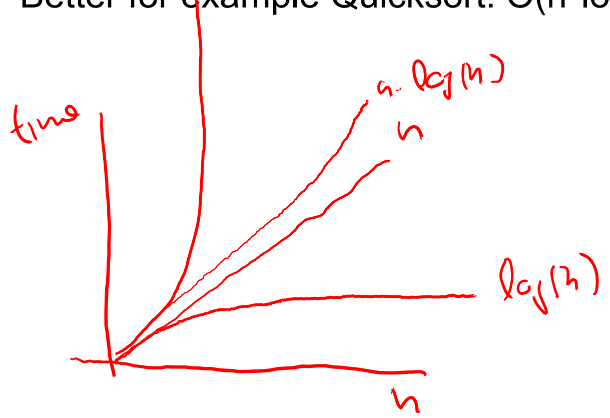


ifp

Universität Stuttgart

Complexity

- $O(n^2)$ Time n^2
- Better for example Quicksort: $O(n*log(n))$ Time

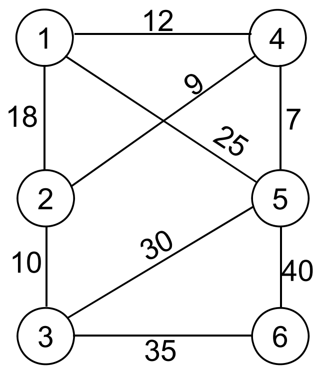


ifp

Universität Stuttgart

Exercise

- Identify the minimal spanning tree with the Kruskal algorithm:

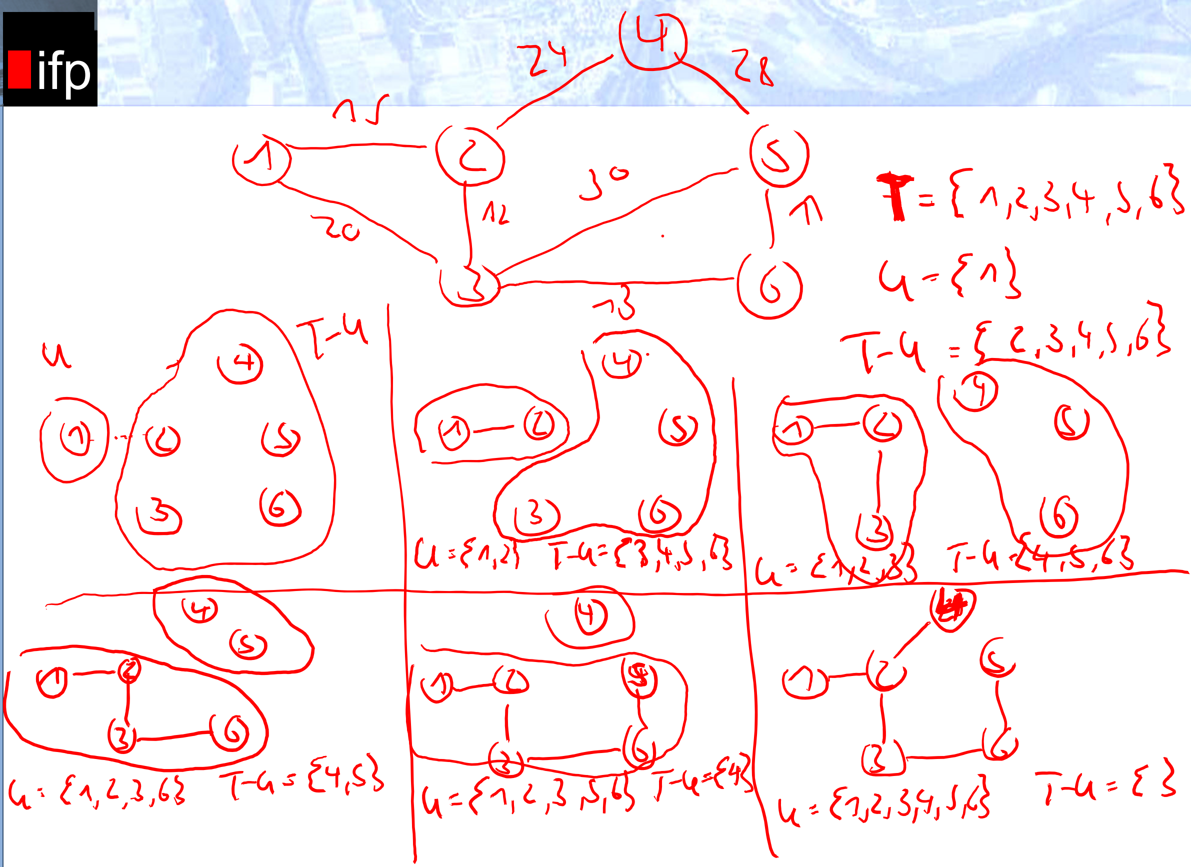


ifp

Universität Stuttgart

Prim Algorithm

T = set that contains all nodes $\{n_1, n_2, \ldots, n_n\}$	
U = set that contains one node $\{n_1\}$	
Repeat n -1 times (n = number of nodes)	
	search for node $n \in \{T \text{ minus } U\}$ with minimum distance to U
	edge from n to U is part of the Minimum Spanning Tree
	add node n to set U



Exercise

- Identify the Minumum Spanning Tree with the Prim algorithm: