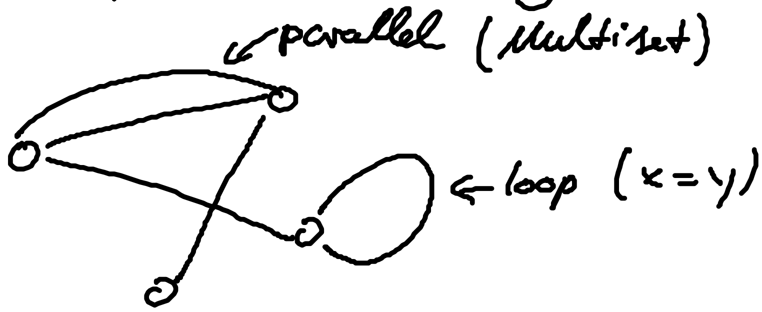
# Netzwerke / Graphen

Def: Ein Graph $G$ ist ein Paar $(V, E)$ bestehend aus einer endlichen Menge $V$ und einer Teilmenge $E$ der Menge aller Paare aus $V$:

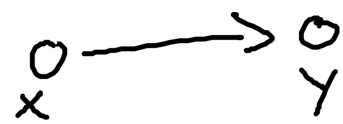
$$G = (V, E) \quad , \quad E \subseteq \{ \underline{\{x, y\}} \mid x, y \in V \}$$

$E$ heißt Kantenmenge, $E(G)$

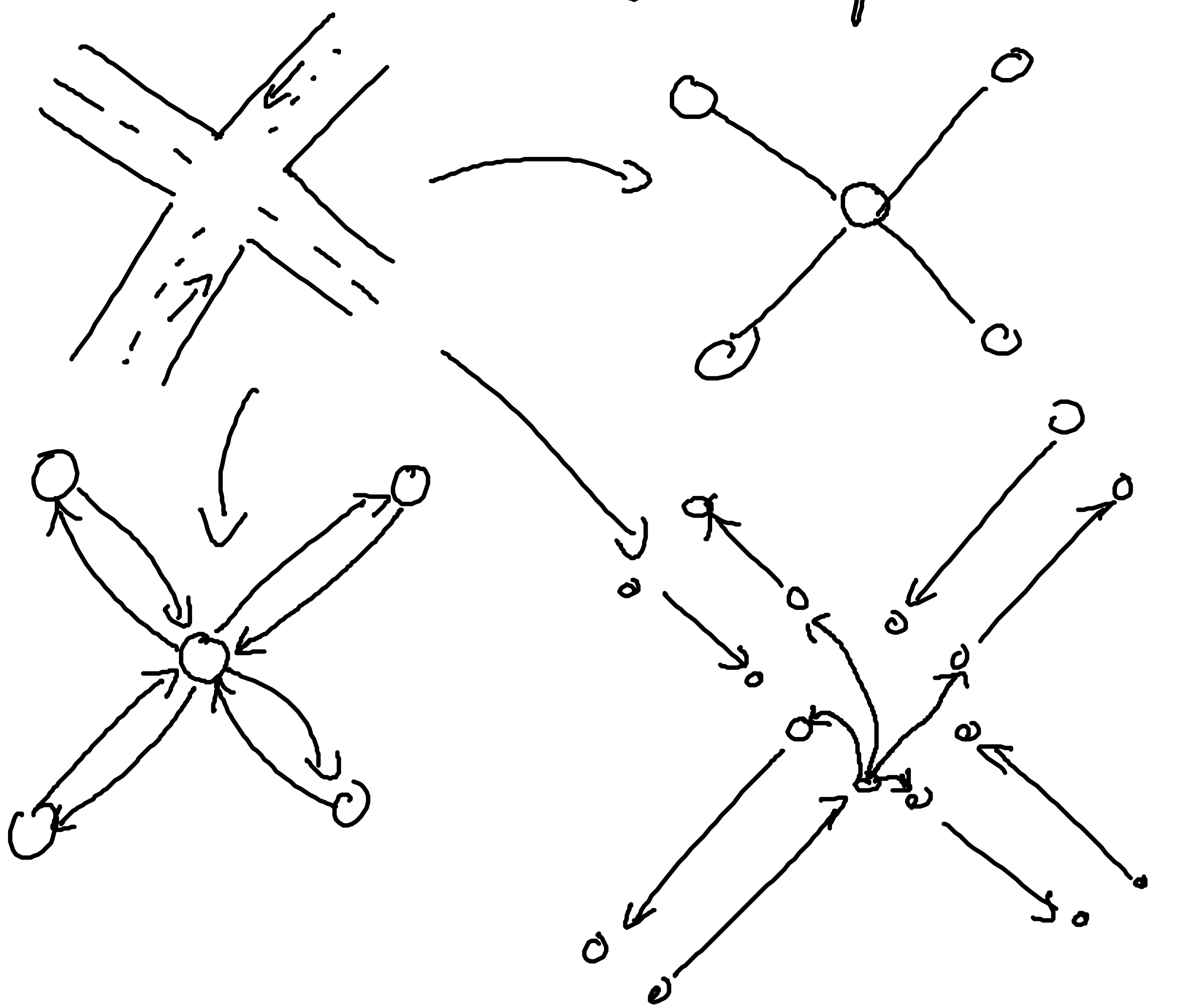
$V$ heißt Knotenmenge, $V(G)$.




1. gerichtetes Graph: $(x, y) \quad (\neq (y, x))$



2. einfacher Graph: hat keine par. Kanten & keine Loops.



Graphen → Netzwerke

Zusatzinformation an Knoten und Kanten:

$l : E \to \mathbb{R}_+$ (z.B. Länge der Kante)

$l(e)$

$w : V \to \mathbb{R}_+$

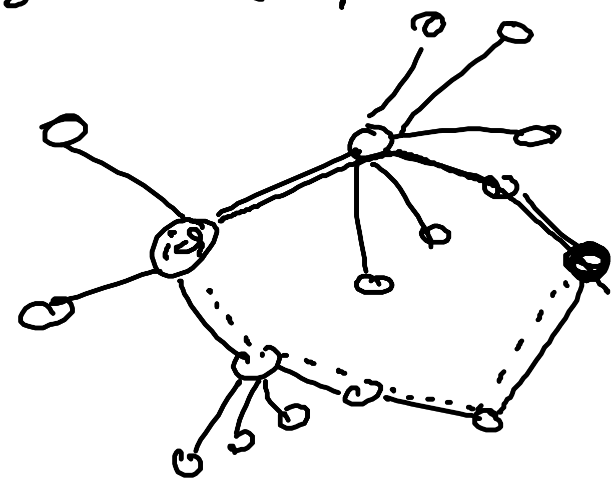
$c : V \to [n] := \{1, 2, \dots, n\}$

Stanley Milgram (60iger):

2 96 „Starter" ⟶ Small World

heute: „soziales Netzwerk"

wie weit ist ein anderes Mitglied von mir entfernt? (Hopdistanz)



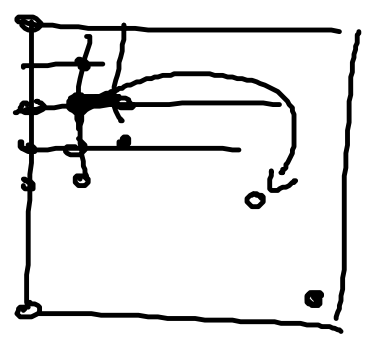

nochmal zu Milgram:

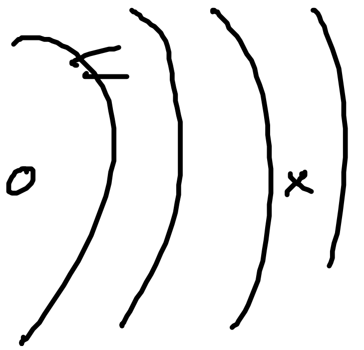
- es gibt viele kurze Wege
- diese Wege ⟶ sozialen Koordinaten

Jon Kleinberg, 1999 / Nature 2000:

Modell reicht aus:

- viele kurze Wege
- findbar

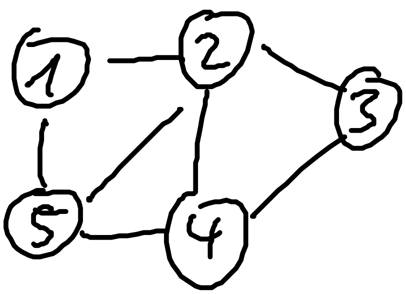




Beschleunigungsidee:
Bidirektional

Wie repräsentieren wir den Graphen?

1. Möglichkeit: Adjazenzmatrix



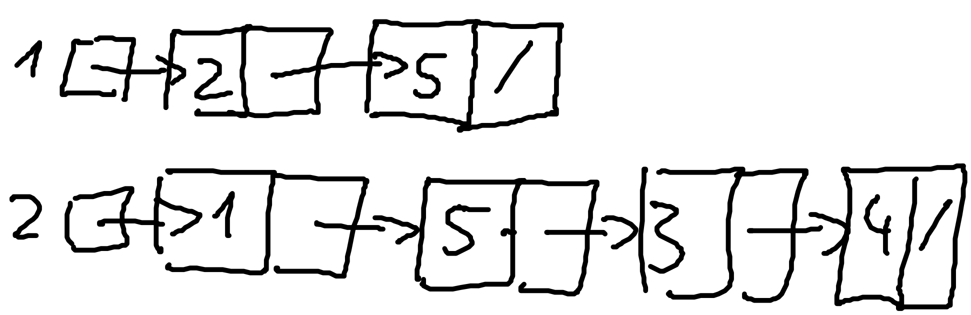

Knoten ist Knoten <u>adjazent</u>.

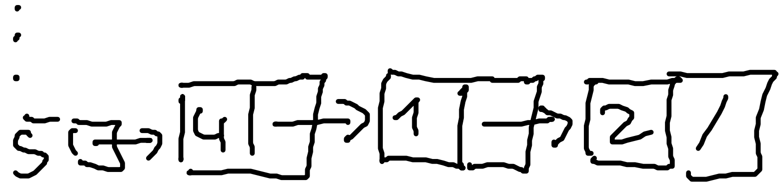
Knoten ist zu einer Kanten <u>inzident</u>.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 | 1 |
| 3 | 0 | 1 | 0 | 1 | 0 |
| 4 | 0 | 1 | 1 | 0 | 1 |
| 5 | 1 | 1 | 0 | 1 | 0 |

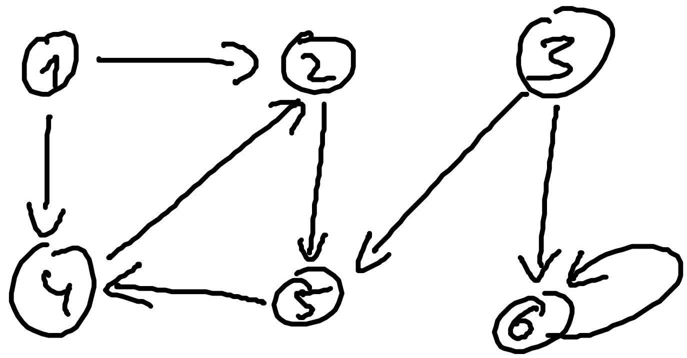
Graphen sind in der Praxis häufig „dünn"!

Liste

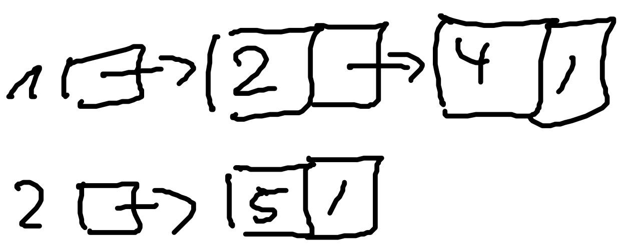




Gerichteter Graph:



Adjazenzmatrix i.A. <u>nicht</u> symmetrisch.



$$\begin{array}{c|cccc} & 1 & 2 & & \\ \hline 1 & 0 & 1 & \cdots & \\ 2 & 0 & 0 & \cdots & \\ \vdots & \vdots & \vdots & \ddots & \end{array}$$

Wir verwalten die Knoten während der Suche mit einer Queue: Q.

2 Operationen: 1. Enqueue (Q, e)
Element e <u>hinten</u> anfügen.
2. Dequeue (Q)
erstes Element zurückgeben (und entfernen)

Färben der Knoten während der Suche:

weiß: Grundzustand

grau: „entdeckt"

Schwarz: „abgearbeitet"

## Breitensuche:

Input: Graphen $G$
und $s \in V(G)$

Output: Distanz $dist(s,v)$ für alle Knoten $v$
& für jeden Knoten $v$ Vorgänger $pred(v)$
auf einem kürzesten Weg.

---

Initialisiere $color(v) = white \quad \forall v \in V(G) \setminus \{s\}$

$d(v) = \infty$ " " $\setminus \{s\}$

$pred(v) = null$ " "

$d(s) = 0$

$color(s) = grey$

$Q = \emptyset$

Enqueue$(Q, s)$

WHILE $Q \neq \emptyset$

DO $u$ = Dequeue$(Q)$

Adjazenzliste von u

FOR ALL $v \in Adj(u)$

DO (IF (color(v) = white)

| i. Level Nachbarn | i+1 Level Nachbarn |
|---|---|
| | |

then color(v) = grey

$d(v) = d(u)+1$

$pred(v) = u$

Enqueue(Q,v)

color(u) = black