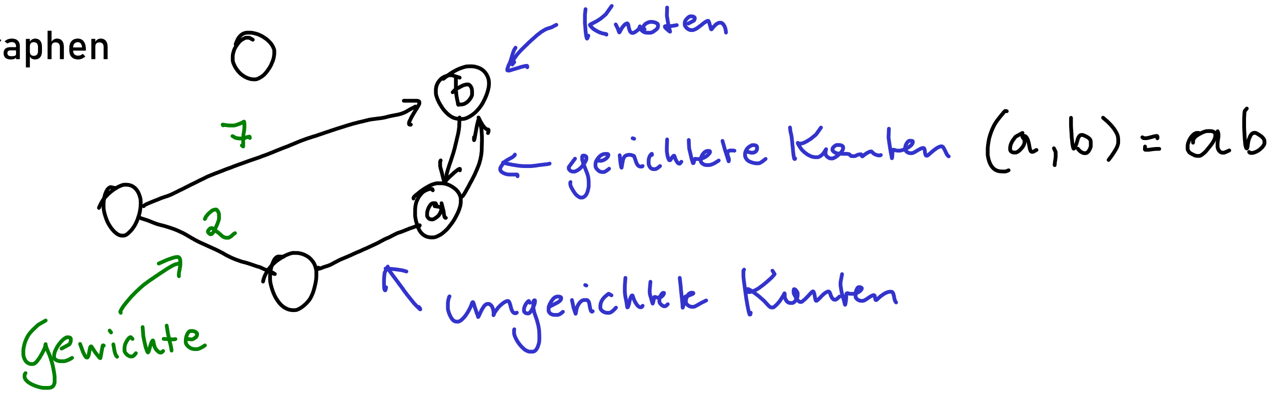
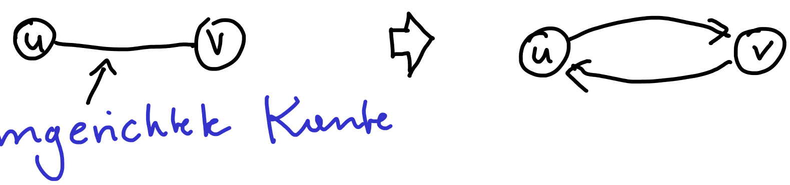
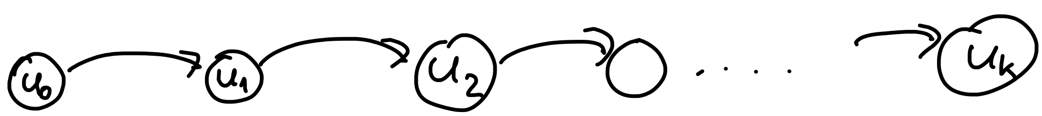
Graphen



• Bei Graphenalgorithmen sind gerichtete Graphen der Normalfall.



• (Gerichteter) Weg der Länge k von u_0 nach u_k.

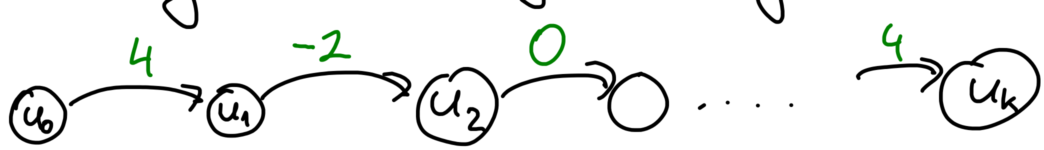


Kreis $u_0 = u_k$. einfacher Weg | Kreis : keine mehrfachen Knoten

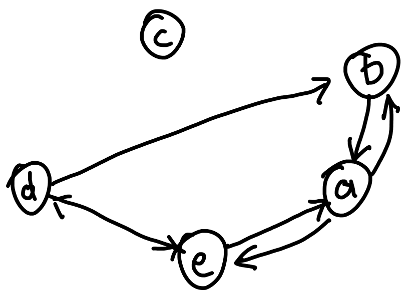
• Abstand von u nach v

= Länge des kürzesten Weges von u nach v
(oder ∞, falls v von u nicht erreichbar ist)

• Bei Kantengewichten: Länge eines Weges = Summe der Kantenlängen



Speicherung von Graphen



$\boxed{\begin{array}{l} n \text{ Knoten} \\ m \text{ Kanten} \end{array}}$

a) <u>Adjazenz</u> <u>matrix</u>

	a	b	c	d	e
a	0	1	0	0	1
b	1	0	0	0	0
c	0	0	0	0	0
d	0	1	0	0	1
e	1	0	0	1	0

$O(n^2)$

b) Adjazenz <u>liste</u>

Jeder Knoten speichert eine Liste der (ausgehenden) Kanten.

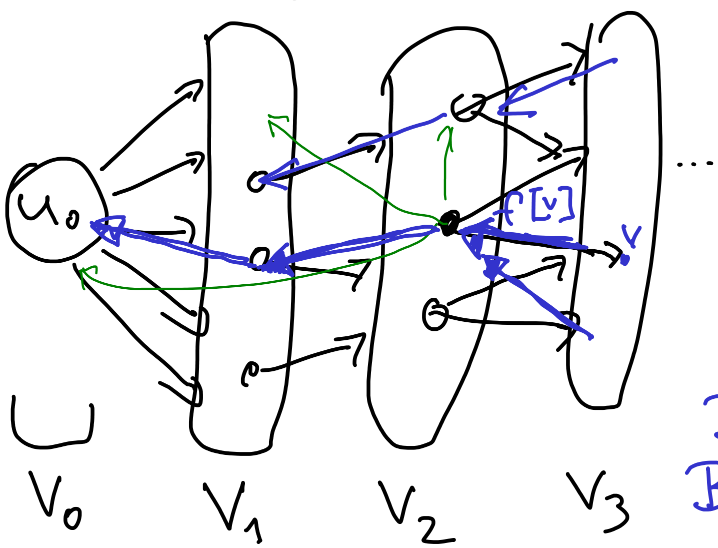
Speicherbedarf $O(m+n)$

a : b, e
b : a ← hier nur: Endknoten der Kanten
c : —
d : b, e ← kann verkettete Liste oder Feld (oder ...) sein.
e : a, d

<u>Grundoperation</u>: Durchlaufe alle Kanten, die von einem Knoten <u>u</u> ausgehen.

<u>Erreichbarkeit</u>: Breitensuche

↳ Besuche alle Knoten, die von einem <u>Startknoten u_0</u> aus erreichbar sind.



V_i := Knoten im Abstand i von v_0

Jeder erreichbare Knoten v hat einen Vorgängerzeiger zu einem Knoten $u = f[v]$ mit $d[u] = d[v] - 1$.

Diese Zeiger bilden einen gerichteten Baum mit Wurzel u_0, den Breitensuchbaum.

Jeder Knoten v hat eine Markierung $d[v]$ (anfangs None).

Am Ende: $d[v]$ = Abstand von u_0

$d[u_0] := 0$; $V_0 := \{u_0\}$; $i := 0$;

<u>Schleife</u> <u>while</u> $V_i \neq \emptyset$:

Besuche alle Nachbarn von Knoten in V_i, baue V_{i+1} auf:

INVARIANTE:

$V_k \equiv \{v \in V \mid d[v] = k\}$

$V_{i+1} := \emptyset$

for u in V_i:

  <u>for</u> v <u>in</u> $E[u]$: # alle Kanten (u,v), die von u ausgehen

    <u>if</u> $d[v]$ == None:

      $d[v] := i+1$

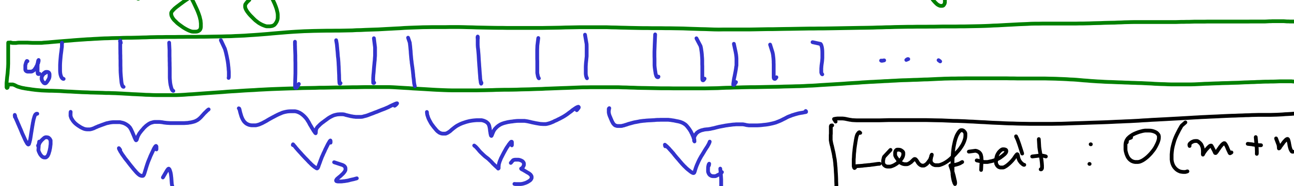
      $V_{i+1} := V_{i+1} \cup \{v\}$; $f[v] := u$

$i := i+1$

einfach als Liste verwalten. Warteschlange

Vorgängerzeiger $f[v]$ = der Knoten u, dem der Knoten v das Einfügen in V_{i+1} verdankt.

eine einzige gemeinsame Warteschlange Q



Laufzeit : $O(m+n)$
breadth-first-search
BFS

$Q := \{u_0\}$; $d[u_0] = 0$; Q

<u>Schleife</u> <u>while</u> $\cancel{V_i} \neq \emptyset$:

~~for u in V_i~~: Entferne das erste Element u von Q.

  <u>for</u> v <u>in</u> $E[u]$: # alle Kanten (u,v), die von u ausgehen

    <u>if</u> $d[v]$ == None:

      $d[v] := \cancel{i+1}\ d[u]+1$;

~~$i := i+1$~~ $Q\ \cancel{V_{i+1}} := Q\ \cancel{V_{i+1}} \cup \{v\}$; $f[v] := u$

$Q := \{u_0\}$; $d[u_0] := 0$; ($d[v]$ auf <u>None</u> initialisiert für $v \neq u_0$)

<u>while</u> $Q \neq \emptyset$:

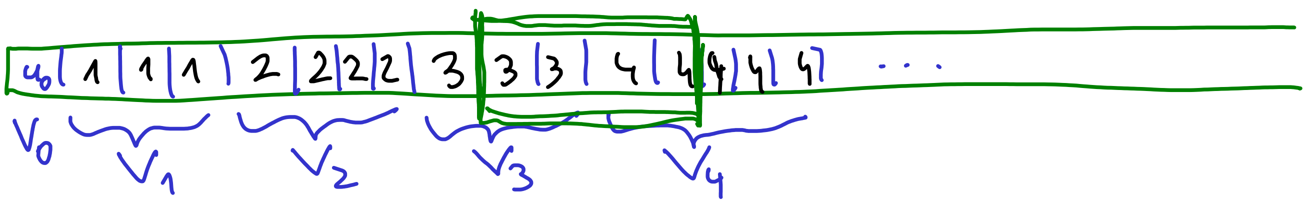
  Entferne das erste Element u von Q.

  <u>for</u> v <u>in</u> $E[u]$: # alle Kanten (u,v), die von u ausgehen

    <u>if</u> $d[v]$ is None:

      $d[v] := d[u] + 1$

      $Q := Q \cup \{v\}$



INVARIANTE.

Die Knoten $u \in Q$ haben höchstens zwei verschiedene Werte $d[u]$,

und zwar zwei aufeinander folgende Zahlen <u>i</u> und <u>$i+1$</u>.

Die Knoten u mit $d[u] = i$ stehen vor den anderen.

KORREKTHEITSBEWEIS.

(1) Wenn $d[v] := k'$ gesetzt wird, dann gibt es einen Weg $u_0 \to \dots \to v$ der Länge k'. $(\Rightarrow \underbrace{d(u_0, v)}_{k} \leq k' = d[v])$

INDUKTION nach k'. $k' = 0$. ✓ $k' \geq 1$:

$$u_0 \overset{\leq k'-1}{\rightsquigarrow} \underset{d[u] = k'-1}{u} \to v$$

✓

(2) Wenn $d(u_0, v) = k$ ist, dann ist v zu Beginn der k-ten Schleife in V_k.
$(\Leftrightarrow d[v] = k)$

$$u_0 \to u_1 \to u_2 \to \dots \to u_{k-1} \to u_k = v$$

Induktion nach k.

$k = 0$ ✓

$d(u_0, u_{k-1}) = k-1$

I.V. $\Rightarrow u_{k-1} \in V_{k-1} \Leftrightarrow d[u_{k-1}] = k-1$

$\Rightarrow$ Es wird $d[v] := k$ gesetzt, sofern $d[v]$ nicht vorher schon einen anderen Wert hat.

$d[v] > k$? ✗ $d[v] < k$? ✗ (Widerspruch zu (1))

$d[v] = k$? ✓