

14. Elementare Graphalgorithmen

- Graphen sind eine der wichtigsten Modellierungskonzepte der Informatik
- Graphalgorithmen bilden die Grundlage vieler Algorithmen in der Praxis

Übersicht:

- Zunächst eine kurze Einführung in Graphen.
- Dann Darstellungen von Graphen
- Schließlich einfache Graphalgorithmen:
Breiten- und Tiefensuche, Zusammenhangskomponenten,
Minimalspannende Bäume

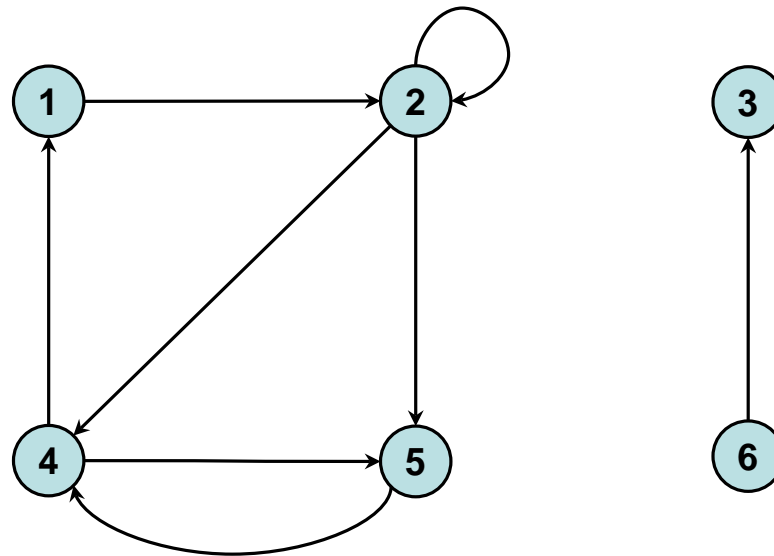
Wiederholung Graphen – Gerichtete Graphen

- Ein **gerichteter Graph** G ist ein Paar (V, E) , wobei V eine endliche Menge ist und $E \subseteq V \times V$.
- Elemente aus V heißen **Knoten**, Elemente aus E heißen **Kanten**. Entsprechend heißt V **Knotenmenge** und E heißt **Kantenmenge** von G .
- Kanten sind geordnete Paare von Knoten. Kanten der Form (u, u) , $u \in V$, sind zugelassen und heißen **Schleifen**.
- Ist $(u, v) \in E$, so sagen wir, dass die Kante **von u nach v führt**. Sagen auch, dass u und v **adjazent** sind. Müssen dann aber noch Richtung berücksichtigen.
- Der **Grad** eines Knotens v ist die Anzahl der Kanten (v, w) in E .

Illustration gerichteter Graph

$$V = \{1,2,3,4,5,6\}$$

$$E = \{(1,2), (2,2), (2,4), (2,5), (4,1), (4,5), (5,4), (6,3)\}$$



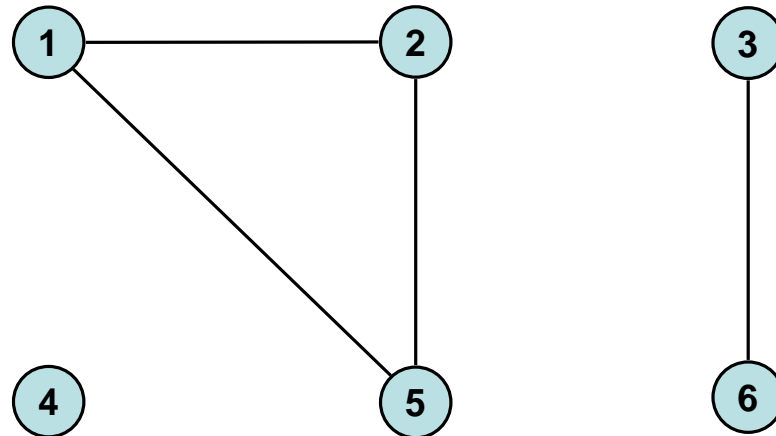
Wiederholung Graphen – Ungerichtete Graphen

- Ein **ungerichteter Graph** G ist ein Paar (V, E) , wobei V eine endliche Menge ist und E eine Menge von 2-elementigen Teilmengen von V ist.
- Elemente aus V heißen **Knoten**, Elemente aus E heißen **Kanten**. Entsprechend heißt V **Knotenmenge** und E heißt **Kantenmenge** von G .
- Kanten haben die Form $\{u, v\}$. Kanten der Form $\{u, u\}$ sind nicht zugelassen. Vereinfachend kann man statt $\{u, v\}$ auch die Notation (u, v) mit $u < v$ verwenden (klärt Speicherort der Kante), Standard ist aber $\{u, v\}$.
- Ist $\{u, v\} \in E$, so sagen wir, u und v **adjazent** sind. Der **Grad** von u ist die Anzahl der Kanten $\{u, v\}$ in E .

Illustration ungerichteter Graph

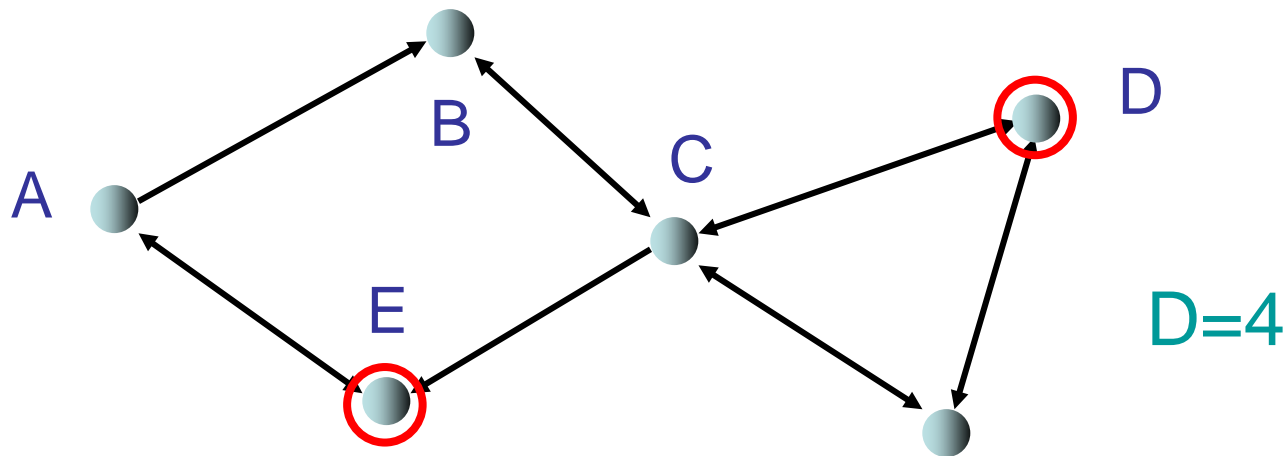
$V = \{1, 2, 3, 4, 5, 6\}$

$E = \{ \{1, 2\}, \{1, 5\}, \{2, 5\}, \{3, 6\} \}$



Graphentheorie

- n : Anzahl Knoten, m : Anzahl Kanten
- $\delta(v,w)$: **Distanz** von w zu v in G
 - gerichteter Graph: Anzahl Kanten eines kürzesten **gerichteten** Weges von v nach w
 - ungerichteter Graph: Anzahl Kanten eines kürzesten Weges von v nach w
- $D = \max_{v,w} \delta(v,w)$: **Durchmesser** von G

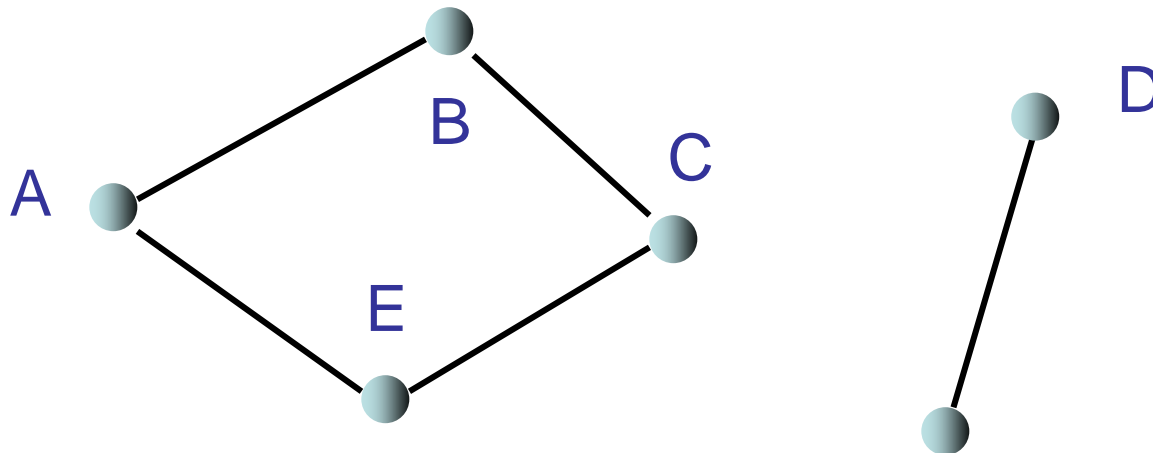


Graphentheorie

G ungerichtet:

- **G** ist **zusammenhängend**: Durchmesser **D** endlich (d.h. es gibt einen Weg von jedem Knoten zu jedem anderen Knoten in **G**)

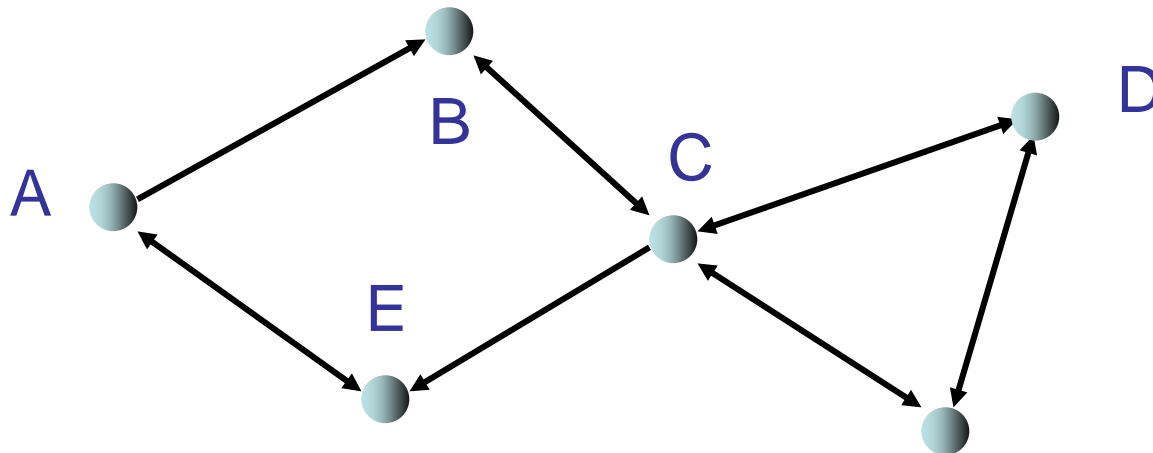
Beispiel: nicht zusammenhängender Graph



Graphentheorie

G gerichtet:

- **G** ist **schwach zusammenhängend**: Durchmesser **D** endlich, wenn alle Kanten als ungerichtet angesehen werden
- **G** ist **stark zusammenhängend**: **D** endlich

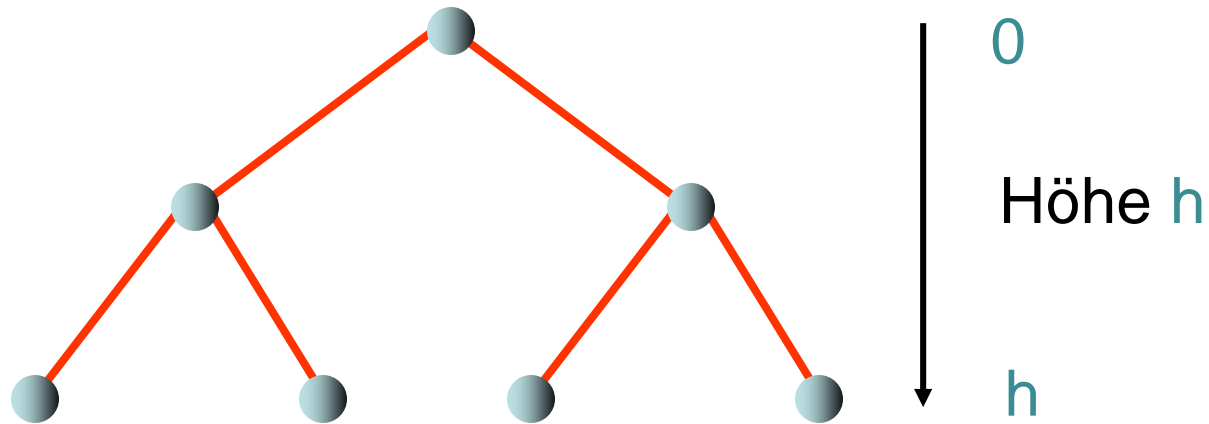


Lineare Liste:



- Grad 2
- Hoher Durchmesser ($n-1$ für n Knoten)

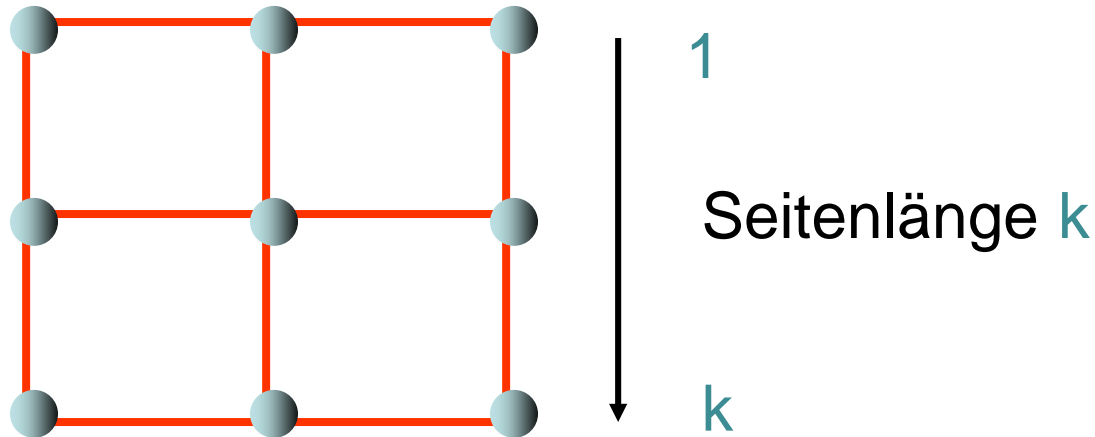
Vollständiger binärer Baum:



- $n=2^{h+1}-1$ Knoten, Grad 3
- Durchmesser ist $2h \sim 2 \log_2 n$

Graphentheorie

2-dimensionales Gitter:



- $n = k^2$ Knoten, maximaler Grad 4
- Durchmesser ist $2(k-1) \sim 2\sqrt{n}$

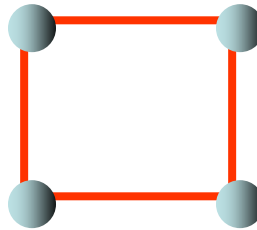
Graphentheorie

Hypercube:

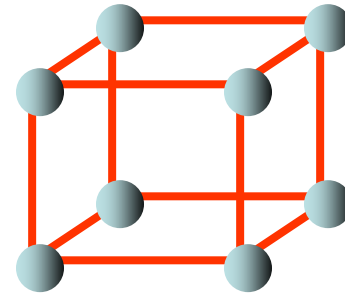
- Knoten: $(x_1, \dots, x_d) \in \{0, 1\}^d$
- Kanten: $\forall i: (x_1, \dots, x_d) \rightarrow (x_1, \dots, x_{i-1}, 1-x_i, x_{i+1}, \dots, x_d)$



d=1



d=2



d=3

Grad d , Durchmesser d

Operationen auf Graphen

$G=(V,E)$: Graph-Variable

Operationen:

- $\text{Insert}(G,e)$: $E:=E \cup \{e\}$
- $\text{Remove}(G,i,j)$: $E:=E \setminus \{e\}$ für die Kante $e=(v,w)$ mit $\text{key}[v]=i$ und $\text{key}[w]=j$
- $\text{Insert}(G,v)$: $V:=V \cup \{v\}$
- $\text{Remove}(G,i)$: sei $v \in V$ der Knoten mit $\text{key}[v]=i$.
 $V:=V \setminus \{v\}$, $E:=E \setminus \{(x,y) \mid x=v \vee y=v\}$
- $\text{Search}(G,i)$: gib Knoten v aus mit $\text{key}[v]=i$
- $\text{Search}(G,i,j)$: gib Kante (v,w) aus mit $\text{key}[v]=i$ und $\text{key}[w]=j$

Operationen auf Graphen

Anzahl der Knoten oft **fest**. In diesem Fall:

- $V = \{1, \dots, n\}$ (Knoten hintereinander nummeriert, identifiziert durch ihre Keys)

Relevante Operationen:

- $\text{Insert}(G, e): E := E \cup \{e\}$
- $\text{Remove}(G, i, j): E := E \setminus \{e\}$ für die Kante $e = (i, j)$
- $\text{Search}(G, i, j):$ gib Kante $e = (i, j)$ aus

Operationen auf Graphen

Menge und Schlüssel der Knoten **variabel**,
aber Anzahl nach oben begrenzt durch **n**:

- **Hashing** (z.B. Kuckuckshashing) kann verwendet werden, um Keys von **n** Knoten in Bereich $\{1, \dots, O(n)\}$ zu hashen.
- Damit kann variabler Fall auf den Fall einer statischen Knotenmenge reduziert werden. (Nur $O(1)$ -Vergrößerung gegenüber statischer Datenstruktur.)

Operationen auf Graphen

Im folgenden: Konzentration auf statische Anzahl an Knoten.

Parameter für Laufzeitanalyse:

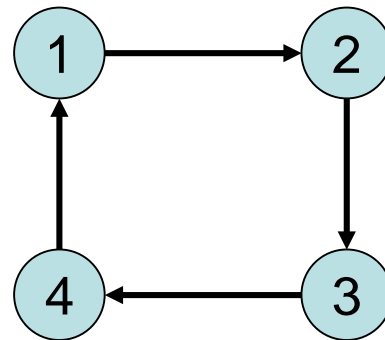
- n : Anzahl Knoten
- m : Anzahl Kanten
- d : maximaler Knotengrad (maximale Anzahl ausgehender Kanten von Knoten)

Graphrepräsentationen

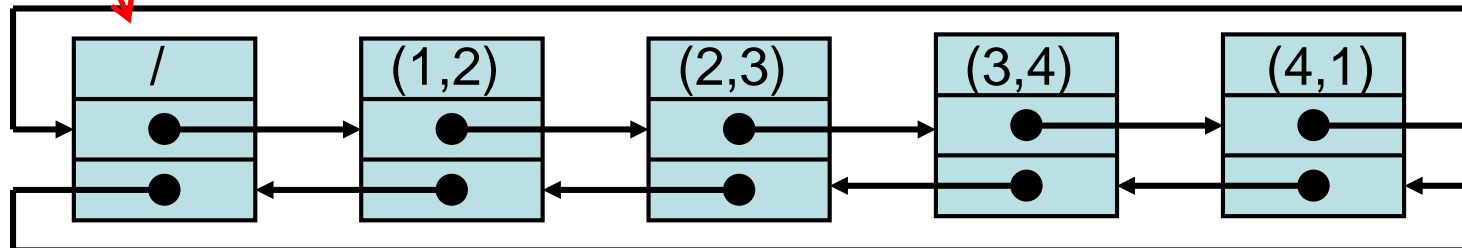
1. Sequenz von Kanten
2. Adjazenzfeld
3. Adjazenzliste
4. Adjazenzmatrix
5. Adjazenzliste + Hashtabelle
6. Implizite Repräsentationen

Graphrepräsentationen

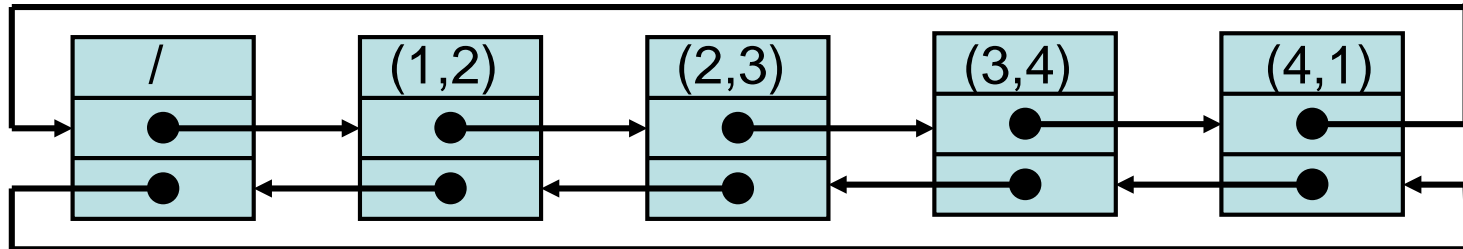
1: Sequenz von Kanten



Dummy



Graphrepräsentationen

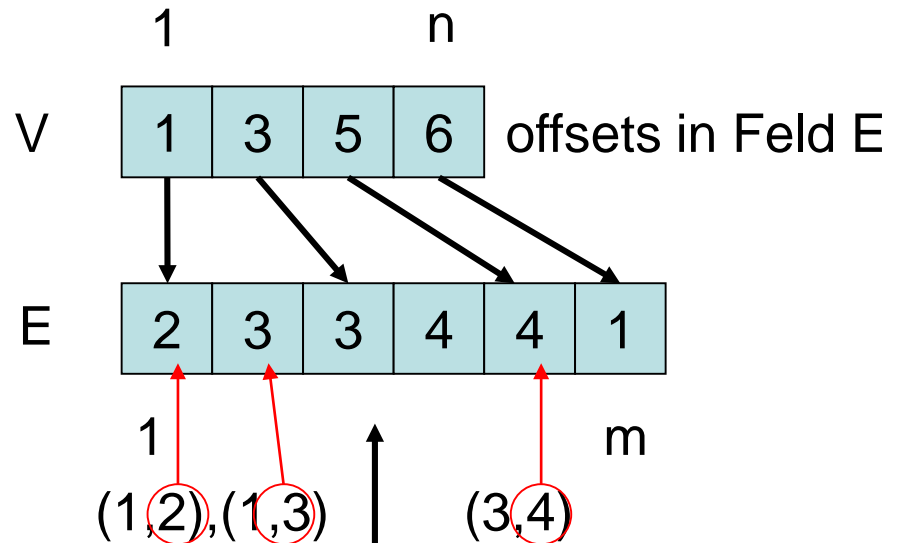
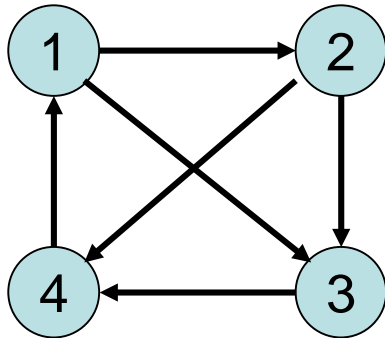


Zeitaufwand:

- $\text{Search}(G,i,j): \Theta(m)$ im worst case
- $\text{Insert}(G,e): O(1)$ (sofern e nicht in E)
- $\text{Remove}(G,i,j): \Theta(m)$ im worst case

Graphrepräsentationen

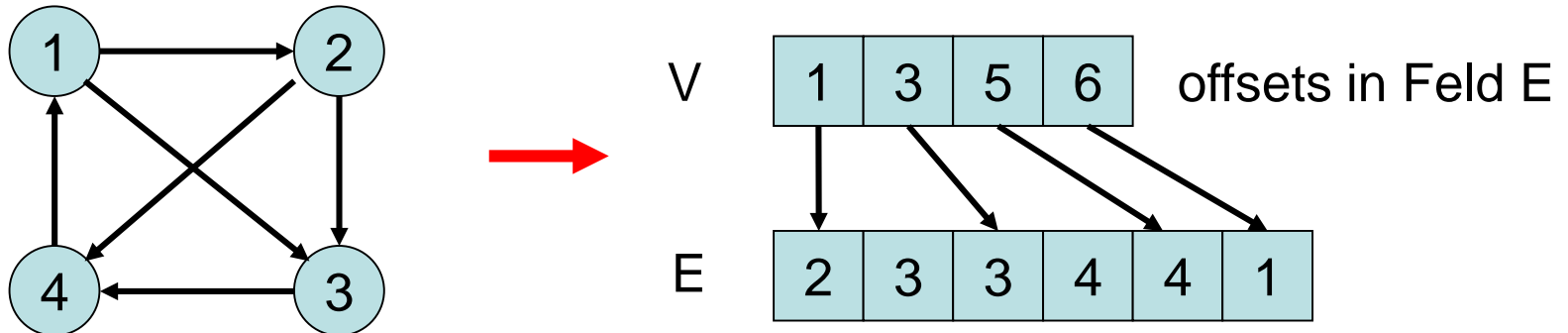
2: Adjazenzfeld



Hier: nur Zielkeys

In echter DS: **E**: Array [1..m] of **Edge**

Graphrepräsentationen

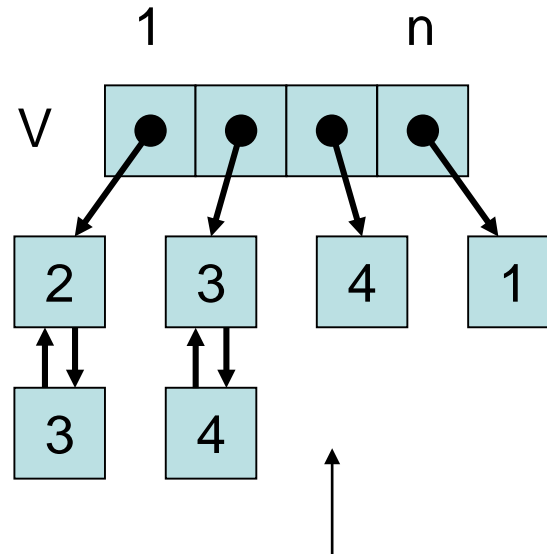
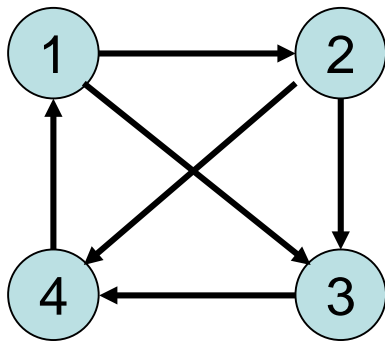


Zeitaufwand:

- $\text{Search}(G,i,j)$: Zeit $O(d)$
 - $\text{Insert}(G,e)$: Zeit $O(n+m)$ (worst case)
 - $\text{Remove}(G,i,j)$: Zeit $O(n+m)$ (worst case)
- wegen Verschiebungen in E

Graphrepräsentationen

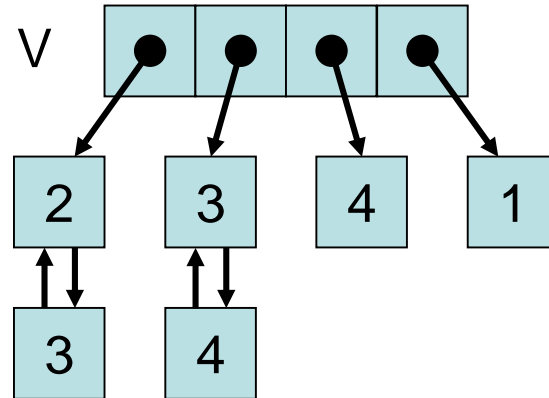
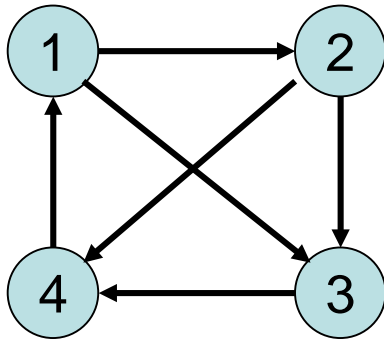
3: Adjazenzliste



Hier: nur Zielkeys

In echter DS: V : Array $[1..n]$ of List of Edge

Graphrepräsentationen



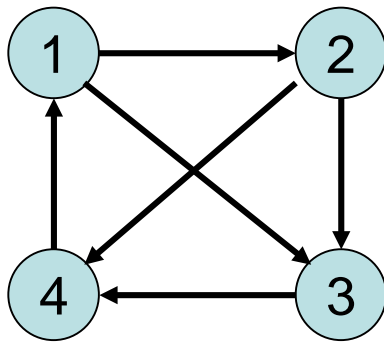
Zeitaufwand:

- $\text{Search}(G,i,j)$: Zeit $O(d)$
- $\text{Insert}(G,e)$: Zeit $O(d)$
- $\text{Remove}(G,i,j)$: Zeit $O(d)$

Problem: d kann
groß sein!

Graphrepräsentationen

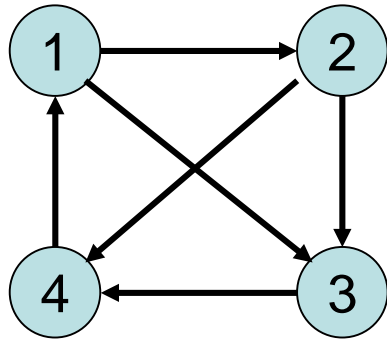
4: Adjazenzmatrix



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

- $A[i,j] \in \{0,1\}$ (bzw. Zeiger auf **Edge**)
- $A[i,j]=1$ genau dann, wenn $(i,j) \in E$

Graphrepräsentationen



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

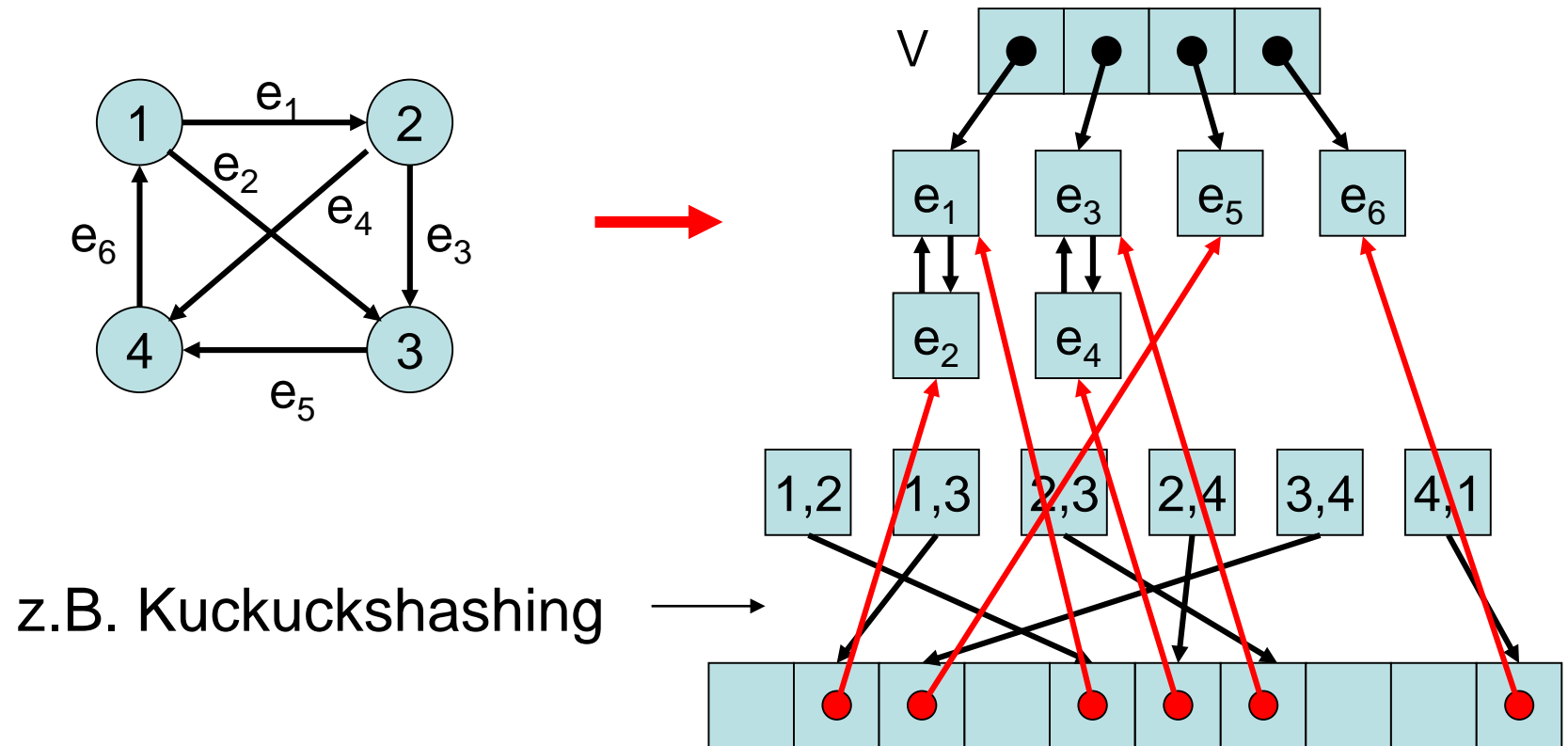
Zeitaufwand:

- $\text{Search}(G,i,j)$: Zeit $O(1)$
- $\text{Insert}(G,e)$: Zeit $O(1)$
- $\text{Remove}(G,i,j)$: Zeit $O(1)$

Aber: Speicher-
aufwand $\Theta(n^2)$

Graphrepräsentationen

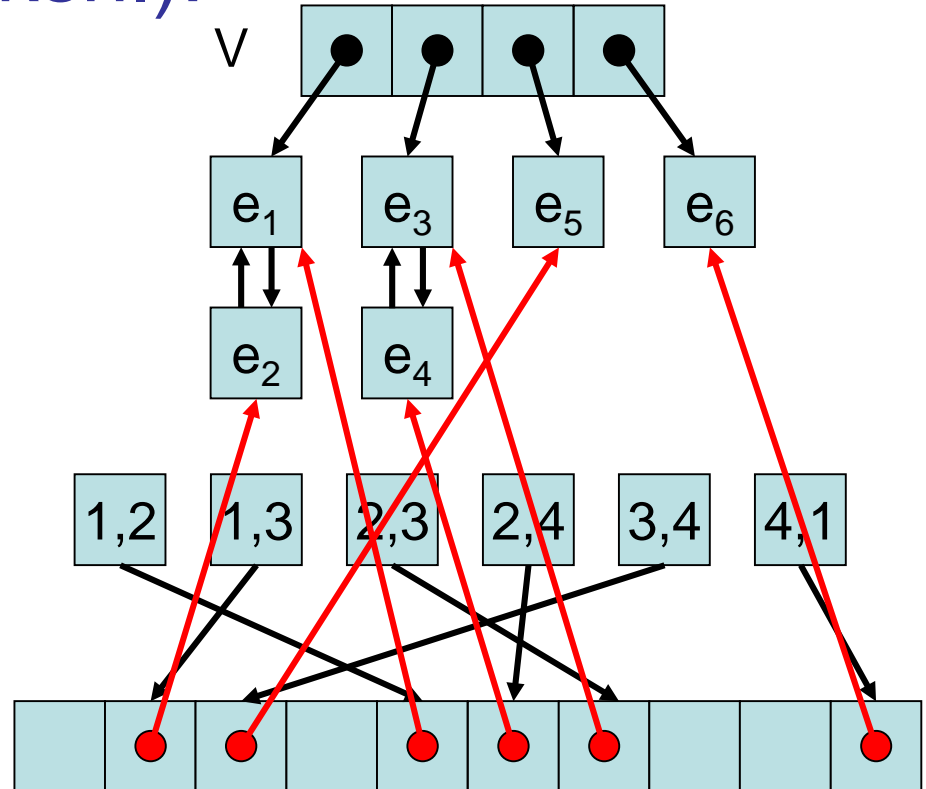
5: Adjazenzliste + Hashtabelle



Graphrepräsentationen

Zeitaufwand (Kuckucksh.):

- $\text{Search}(G,i,j)$:
 $O(1)$ (worst case)
- $\text{Insert}(G,e)$:
 $O(1)$ (erwartet)
- $\text{Remove}(G,i,j)$:
 $O(1)$ (worst case)
- Speicher: $O(n+m)$



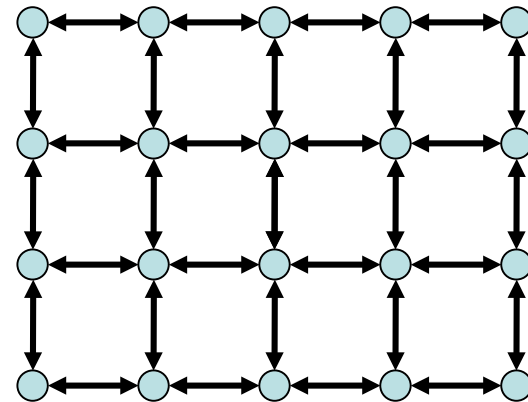
Graphrepräsentationen

6: Implizite Repräsentationen

(k,l) -Gitter $G=(V,E)$:

- $V=[k] \times [l]$ ($[a]=\{0, \dots, a-1\}$ für $a \in \mathbb{N}$)
- $E=\{((v,w),(x,y)) \mid (v=w \wedge |x-y|=1) \vee (w=y \wedge |v-x|=1)\}$

Beispiel: $(5,4)$ -Gitter



Graphrepräsentationen

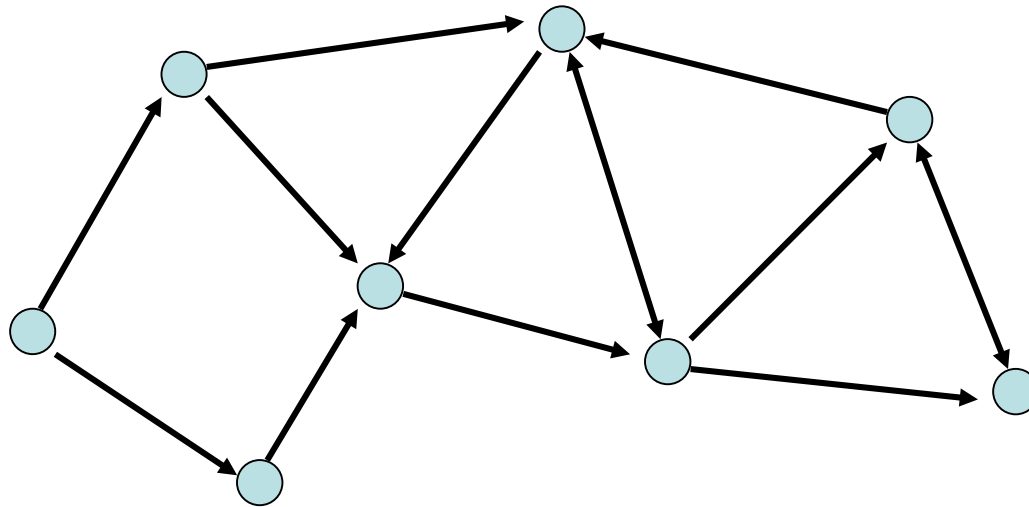
6: Implizite Repräsentationen

(k,l) -Gitter $G=(V,E)$:

- $V=[k] \times [l]$ ($[a]=\{0, \dots, a-1\}$ für $a \in \mathbb{N}$)
- $E=\{((v,w),(x,y)) \mid (v=x \wedge |w-y|=1) \vee (w=y) \wedge |v-x|=1)\}$
- Speicheraufwand: $O(\log k + \log l)$
(speichere Kantenregel sowie k und l)
- Search-Operation: $O(1)$ Zeit (reine Rechnung)

Graphdurchlauf

Zentrale Frage: Wie können wir die Knoten eines Graphen durchlaufen, so dass jeder Knoten mindestens einmal besucht wird?



Graphdurchlauf

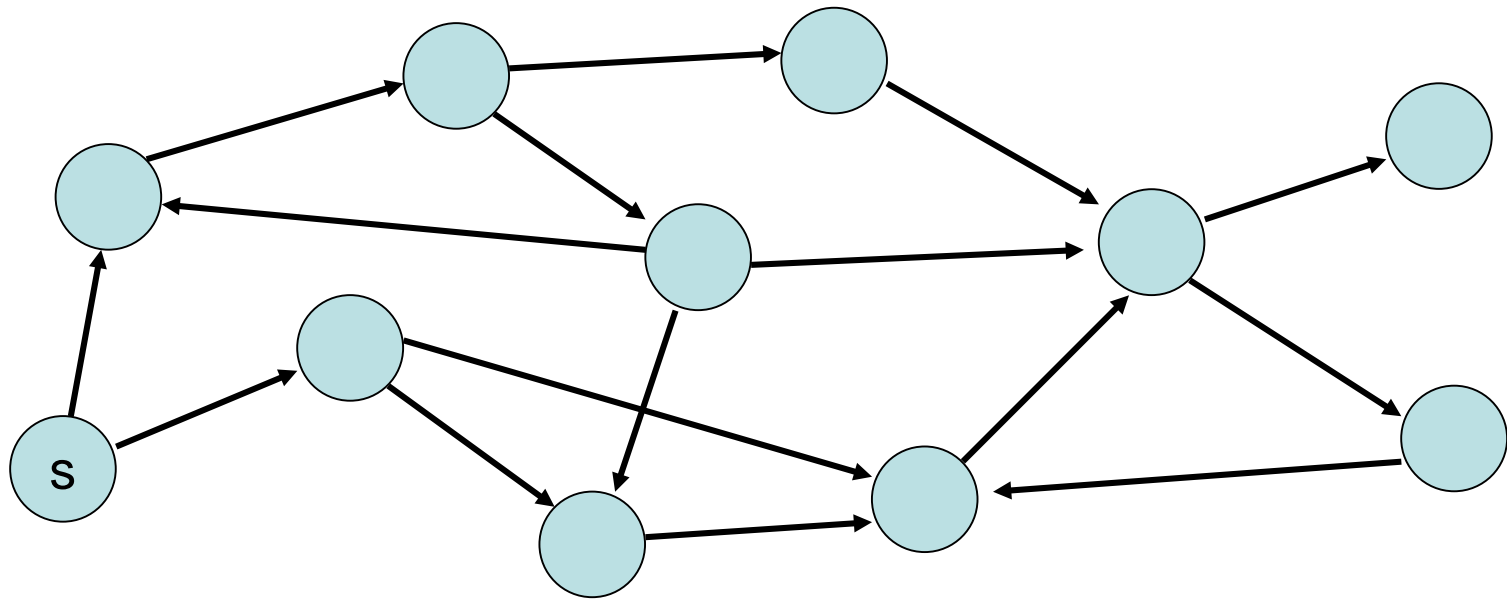
Zentrale Frage: Wie können wir die Knoten eines Graphen durchlaufen, so dass jeder Knoten mindestens einmal besucht wird?

Grundlegende Strategien:

- Breitensuche
- Tiefensuche

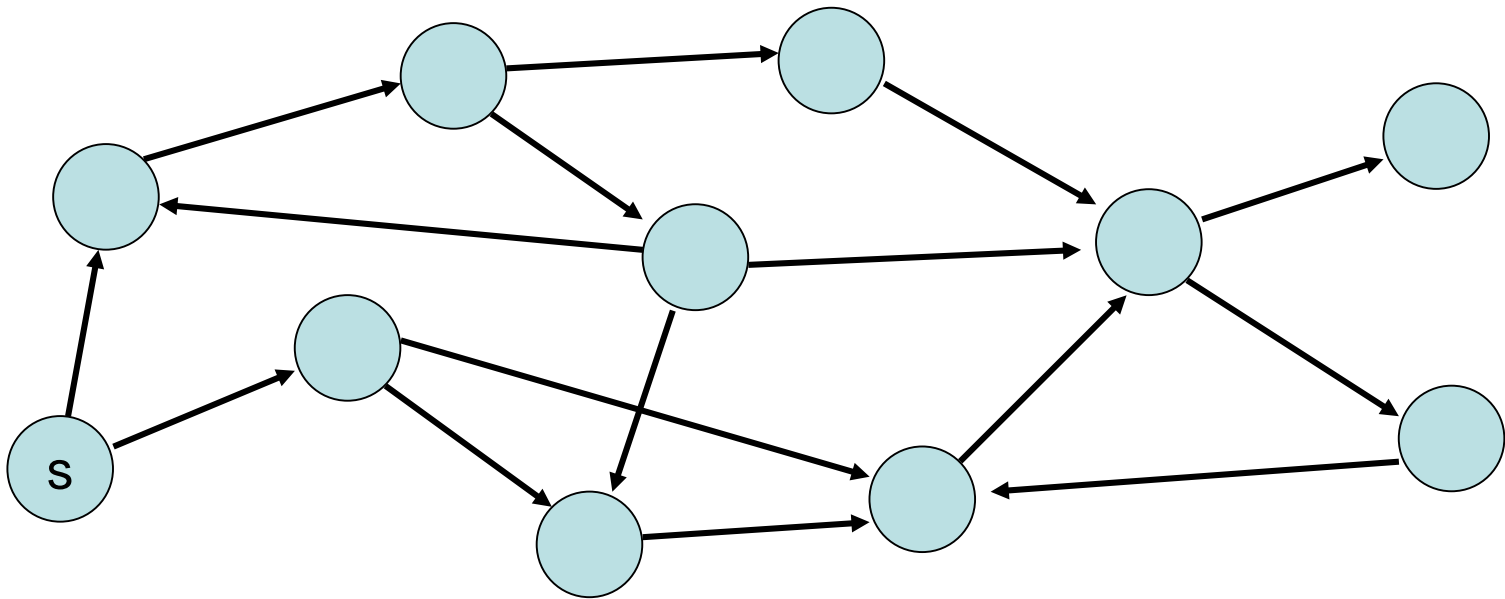
Breitensuche

- Starte von einem Knoten **s**
- Exploriere Graph Distanz für Distanz



Tiefensuche

- Starte von einem Knoten **s**
- Exploriere Graph in die Tiefe
(● : aktuell, ● : noch aktiv, ● : fertig)



Breitensuche – Überblick (1)

- Die *Breitensuche* ist ein Algorithmus, der die Grundlage vieler Graphenalgorithmen bildet.
- Ziel der Breitensuche ist es, bei einem Graphen $G=(V,E)$ und einer *Quelle* $s \in V$ alle Knoten $v \in V$ zu finden, die von s aus *erreichbar* sind. Dabei ist ein Knoten v von s aus erreichbar, wenn es in G einen Pfad von s nach v gibt.
- Die Breitensuche berechnet auch für alle Knoten v den *Abstand* $\delta(s,v)$ von s zu v . Dabei ist der Abstand von s zu v die minimale Anzahl von Kanten auf einem Pfad von s nach v .

Breitensuche - Überblick (2)

- Wird v entdeckt, während $\text{Adj}[u]$ nach neuen Knoten durchsucht wird, so heisst u *Vorgänger* von v . ($\text{Adj}[u]$: Menge der Nachbarn von u)
- Knoten sind entweder *weiß*, *grau* oder *schwarz*.
- Weiß sind alle noch nicht entdeckten Knoten.
- Grau sind alle entdeckten Knoten, deren Adjazenzliste noch nicht vollständig nach neuen Knoten durchsucht wurde.
- Schwarz sind alle anderen Knoten, d.h., schwarze Knoten wurden bereits entdeckt und ihre Adjazenzliste wurde vollständig durchsucht.

Pseudocode für Breitensuche

BFS(G,s)

```
1 for jeden Knoten  $u$  in  $V \setminus \{s\}$ 
2   do  $color[u] \leftarrow WHITE$ 
3      $d[u] \leftarrow \infty$ 
4      $\pi[u] \leftarrow NIL$ 
5  $color[s] \leftarrow GRAY$ 
6  $d[s] \leftarrow 0$ 
7  $\pi[s] \leftarrow NIL$ 
8  $Q \leftarrow \{ \}$ 
9 Enqueue(Q,s)
10 while  $Q \neq \{ \}$ 
11   do  $u \leftarrow Dequeue(Q)$ 
12     for  $v \in Adj[u]$ 
13       do if  $color[v] = WHITE$ 
14         then  $color[v] \leftarrow GRAY$ 
15            $d[v] \leftarrow d[u] + 1$ 
16            $\pi[v] \leftarrow u$ 
17           Enqueue(Q,v)
18    $color[u] \leftarrow BLACK$ 
```

- BFS benutzt Queue für graue Knoten.
- $color[u]$:= Feld für Farbe von v . Initial WHITE.
- $d[u]$:= Feld für bislang berechneten Abstand zu s . Initial ∞ .
- $\pi[u]$:= Feld für Vorgänger. Initial NIL.

Pseudocode für Breitensuche

BFS(G,s)

```
1 for jeden Knoten  $u$  in  $V \setminus \{s\}$ 
2   do  $color[u] \leftarrow WHITE$ 
3      $d[u] \leftarrow \infty$ 
4      $\pi[u] \leftarrow NIL$ 
5  $color[s] \leftarrow GRAY$ 
6  $d[s] \leftarrow 0$ 
7  $\pi[s] \leftarrow NIL$ 
8  $Q \leftarrow \{ \}$ 
9  $Enqueue(Q,s)$ 
10 while  $Q \neq \{ \}$ 
11   do  $u \leftarrow Dequeue(Q)$ 
12     for  $v \in Adj[u]$ 
13       do if  $color[v] = WHITE$ 
14         then  $color[v] \leftarrow GRAY$ 
15            $d[v] \leftarrow d[u] + 1$ 
16            $\pi[v] \leftarrow u$ 
17            $Enqueue(Q,v)$ 
18    $color[u] \leftarrow BLACK$ 
```

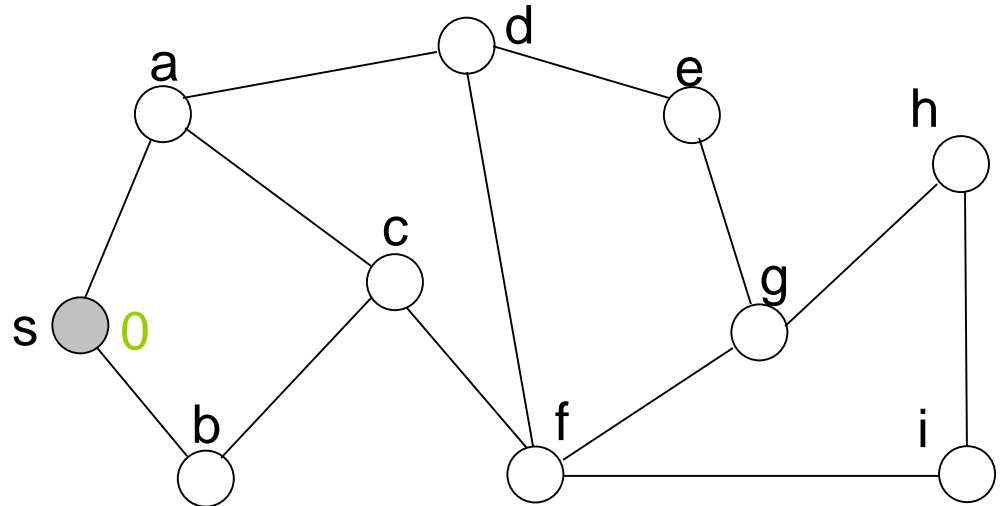
- BFS benutzt Queue für graue Knoten.
- $color[u]$:= Feld für Farbe von v . Initial WHITE.
- $d[u]$:= Feld für bislang berechneten Abstand zu s . Initial ∞ .
- $\pi[u]$:= Feld für Vorgänger. Initial NIL.

initialisiere BFS

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

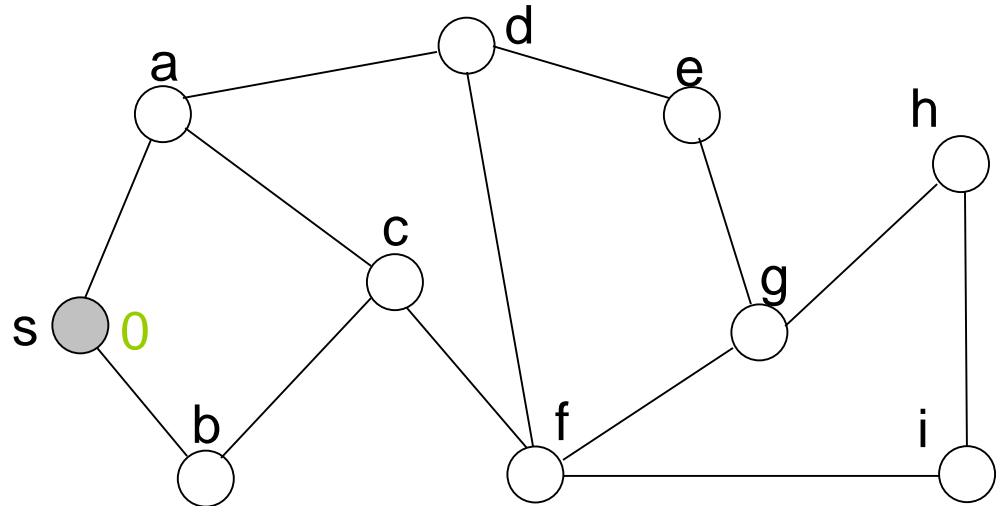


Q: s

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: s

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

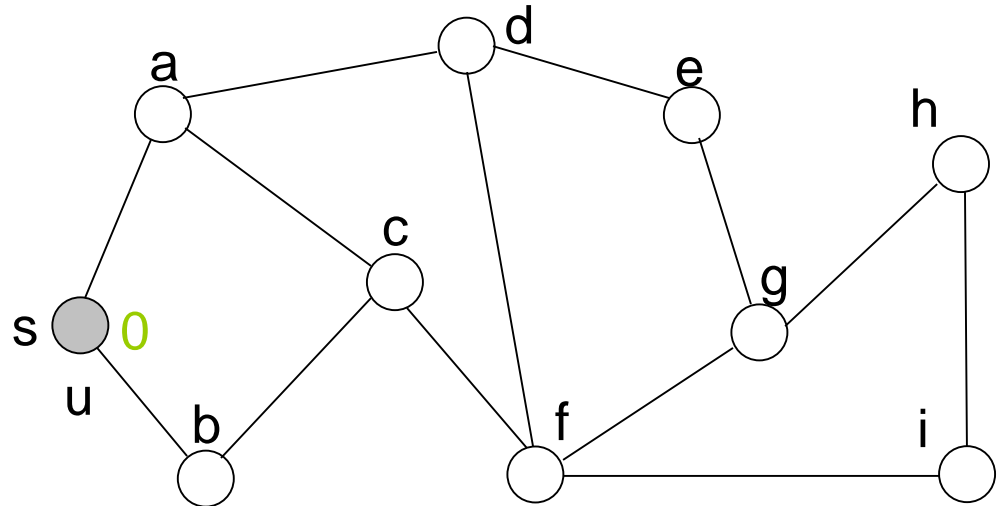
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$ Q:

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

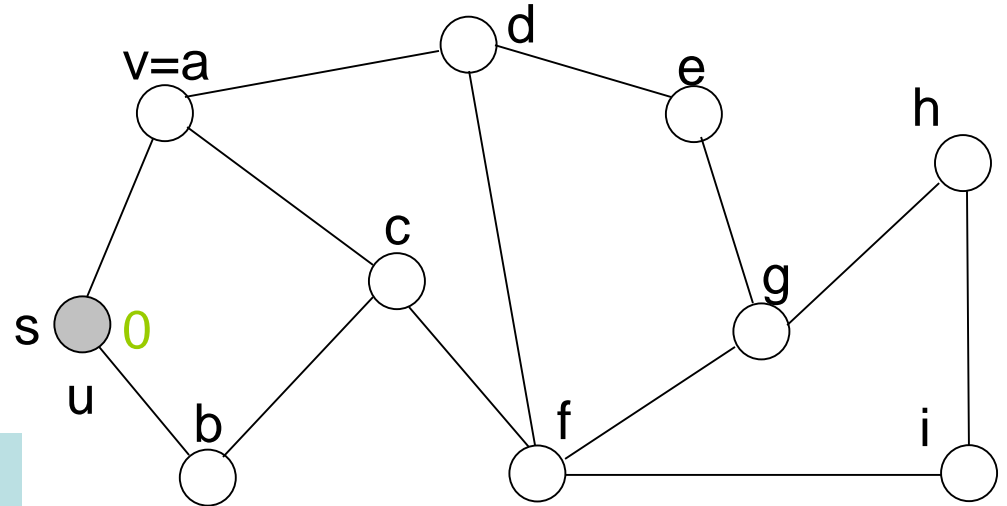
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$ $Q:$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

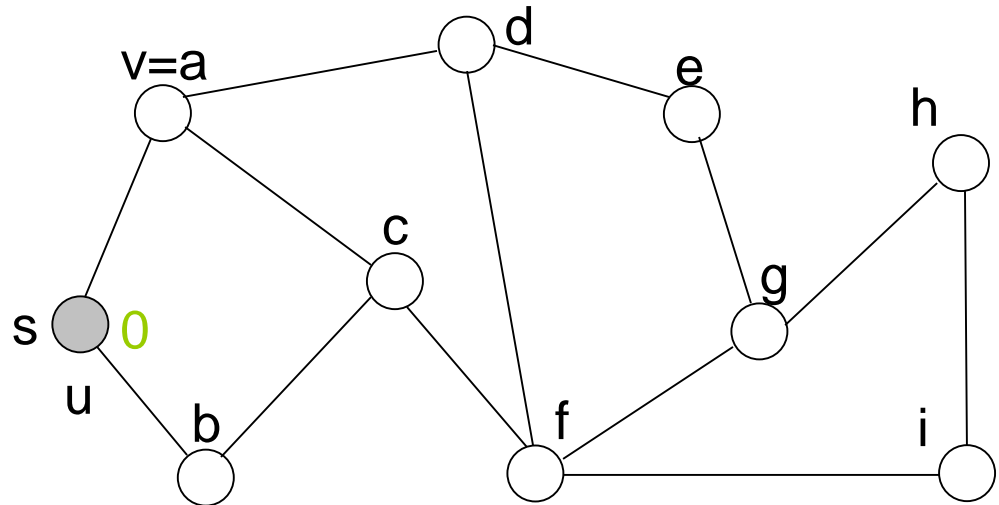
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$ Q:

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

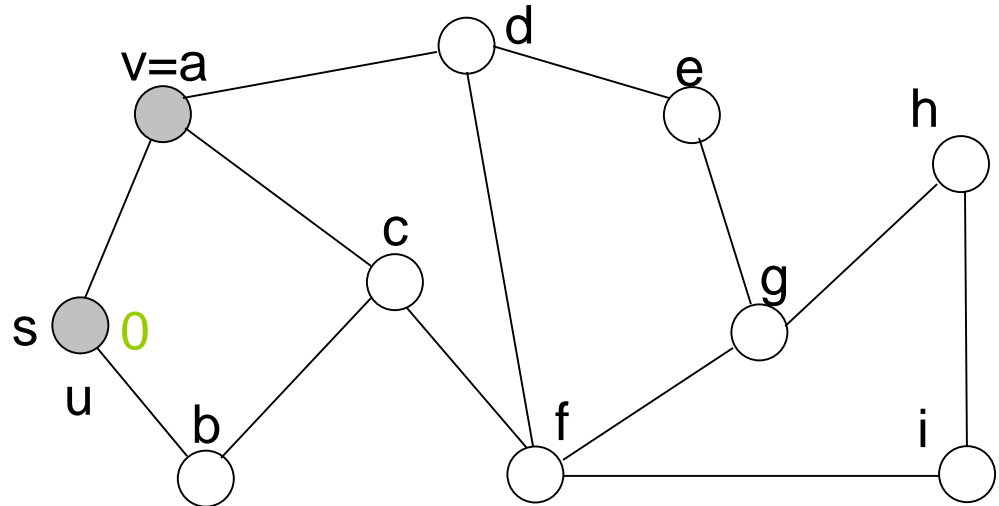
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$ Q:

8. $\text{Enqueue}(Q, v)$

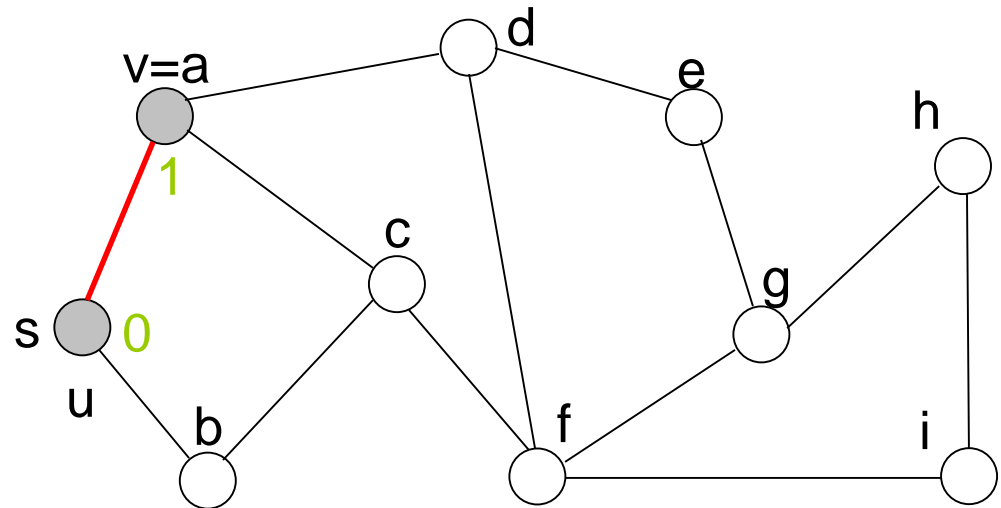
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

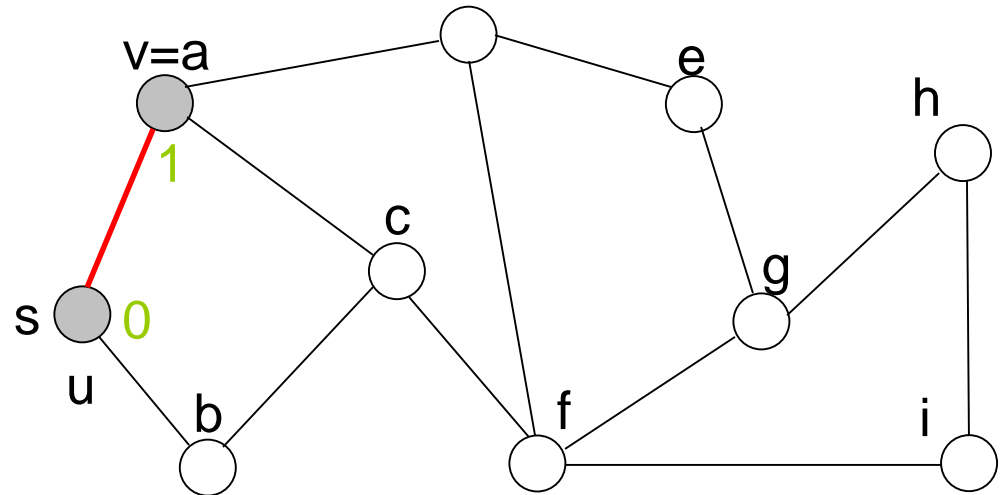


Q:

Breitensuche - Beispiel

BFS(G,s)

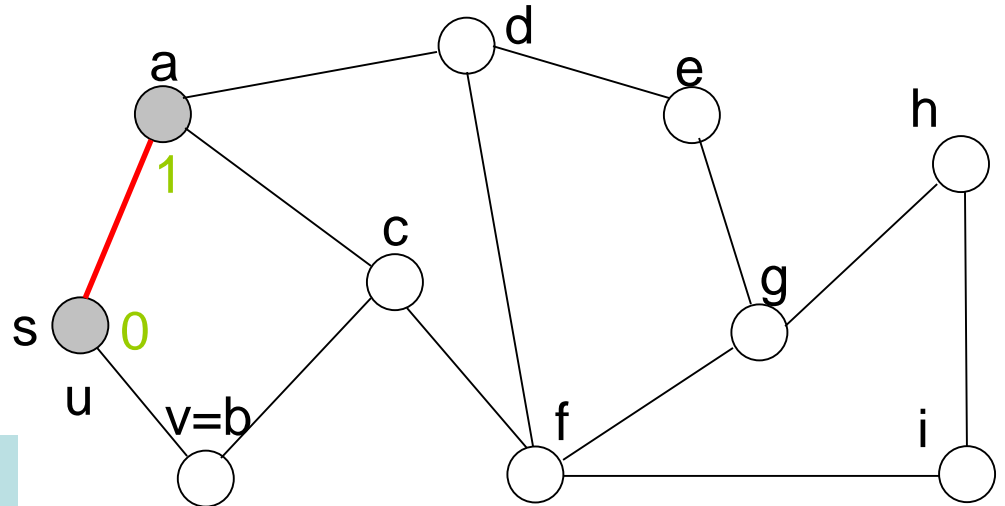
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. **Enqueue**(Q,v)
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

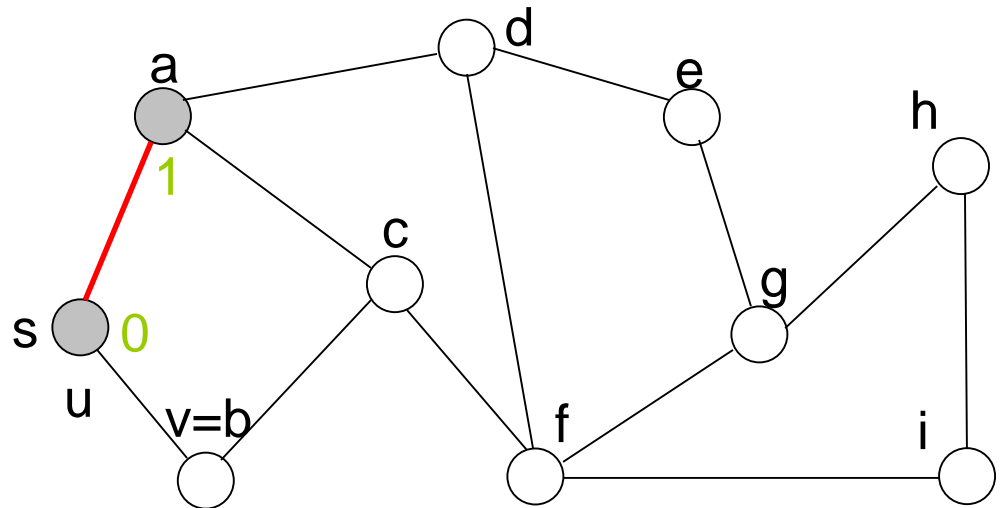
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

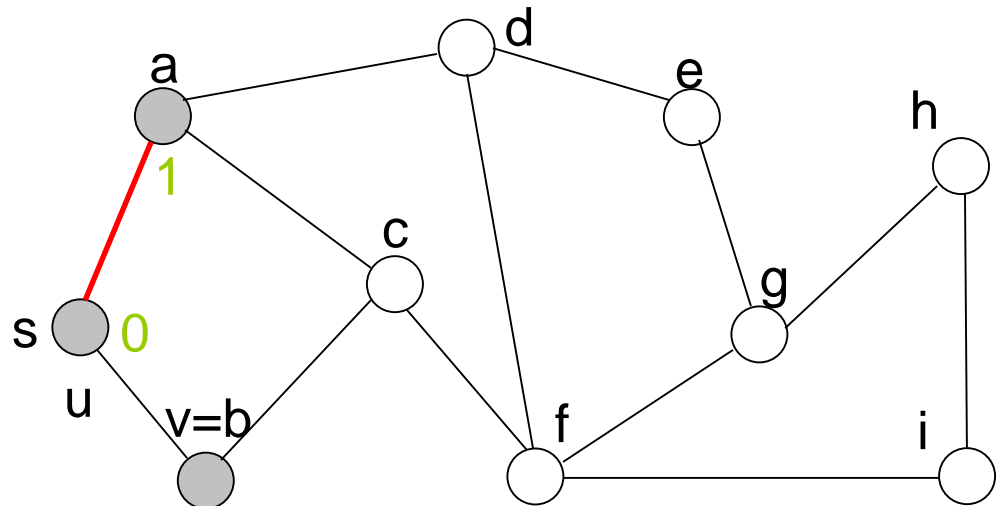
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

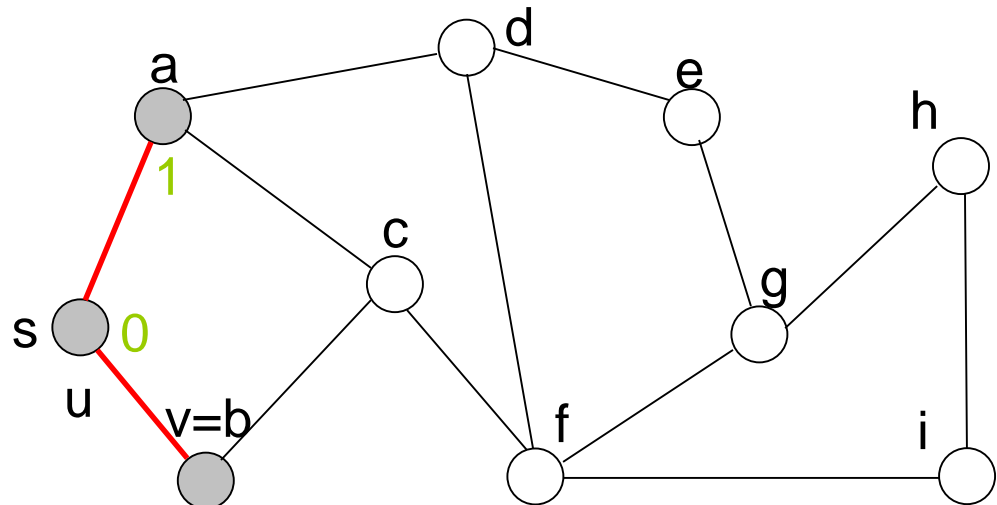
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$

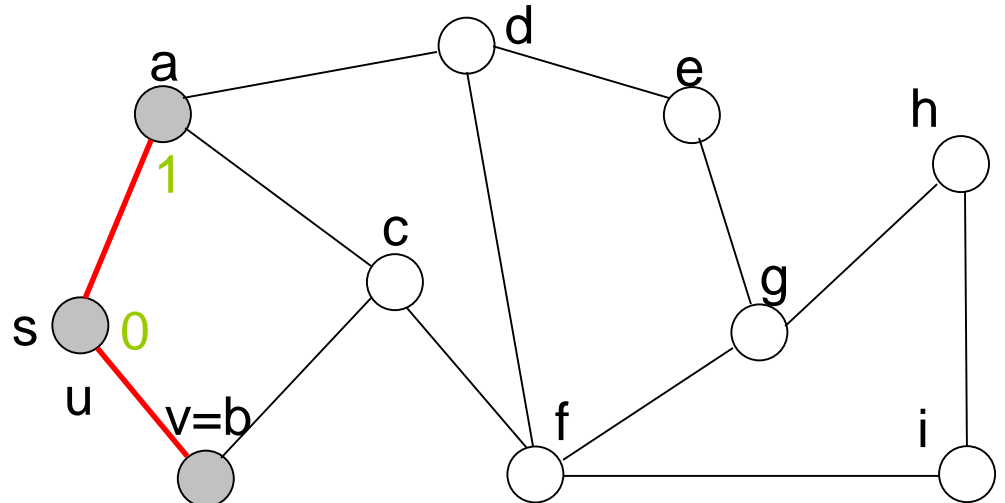


Q: a

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. **Enqueue**(Q,v)
9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: a, b

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

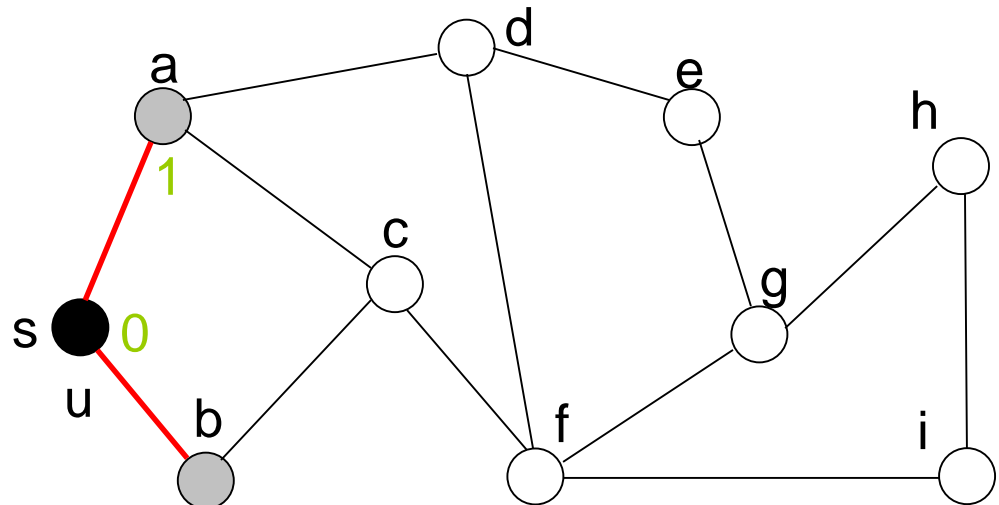
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: a, b

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

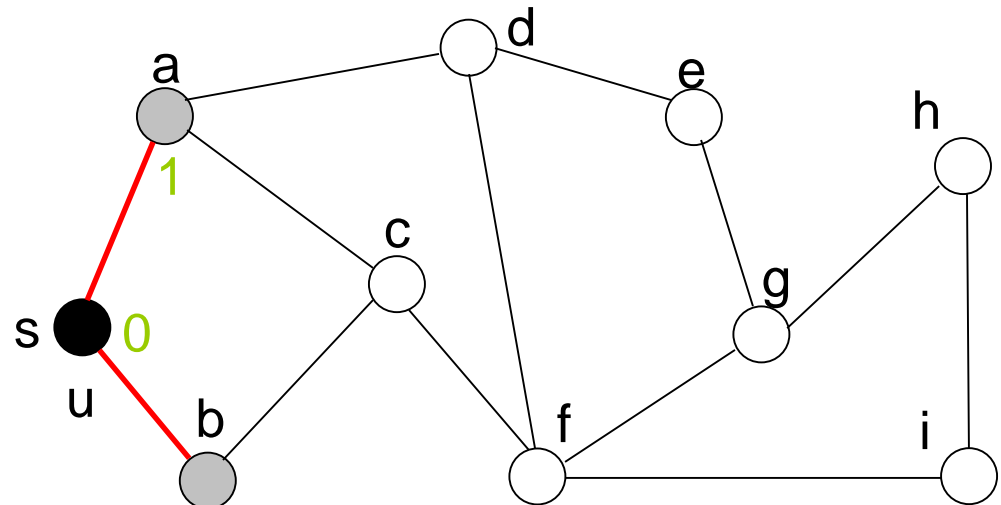
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$

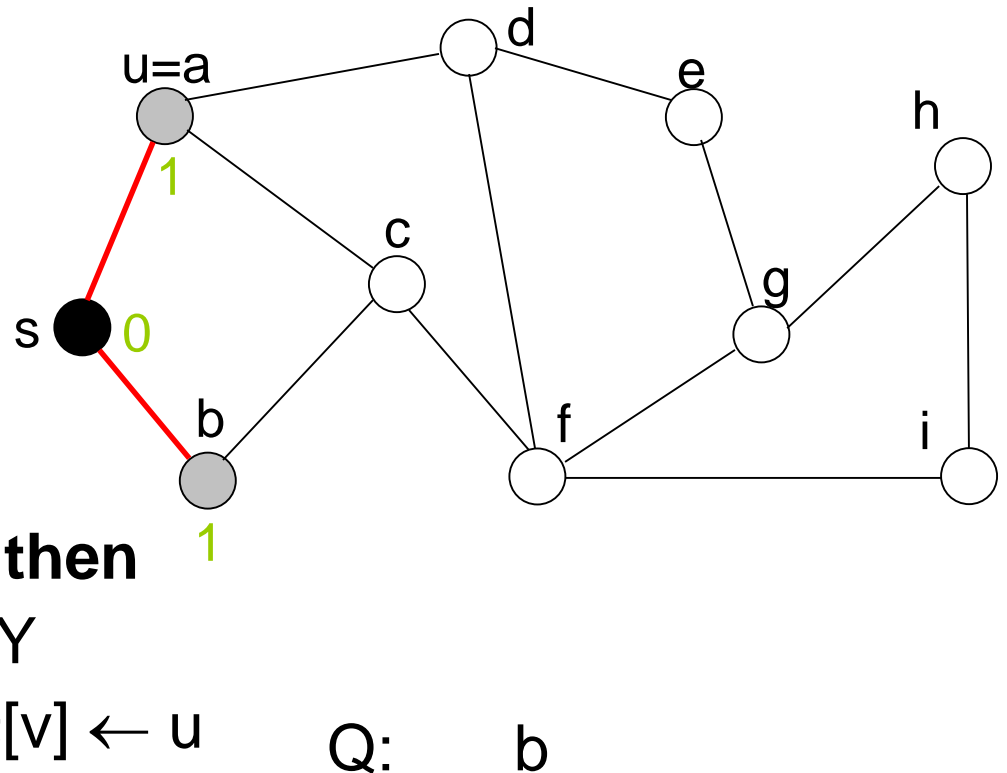


Q: a, b

Breitensuche - Beispiel

BFS(G,s)

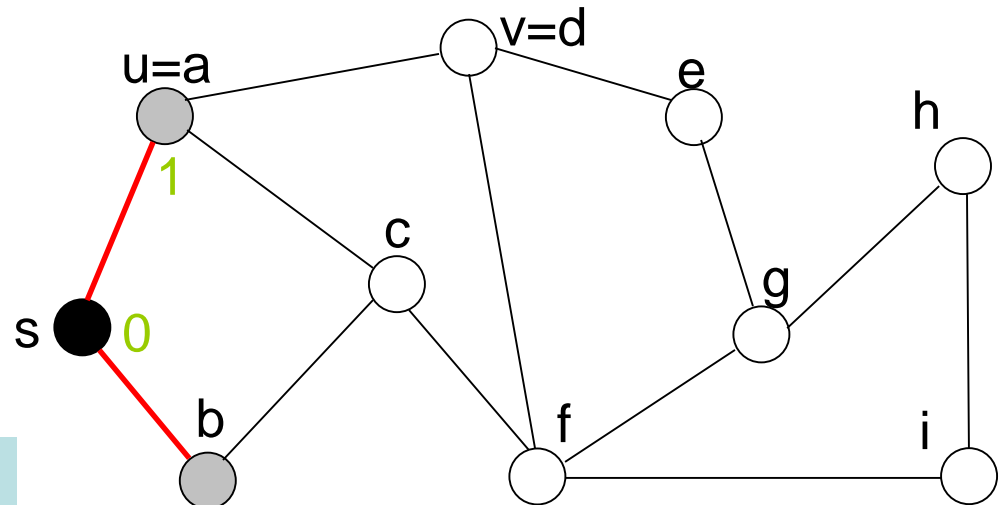
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

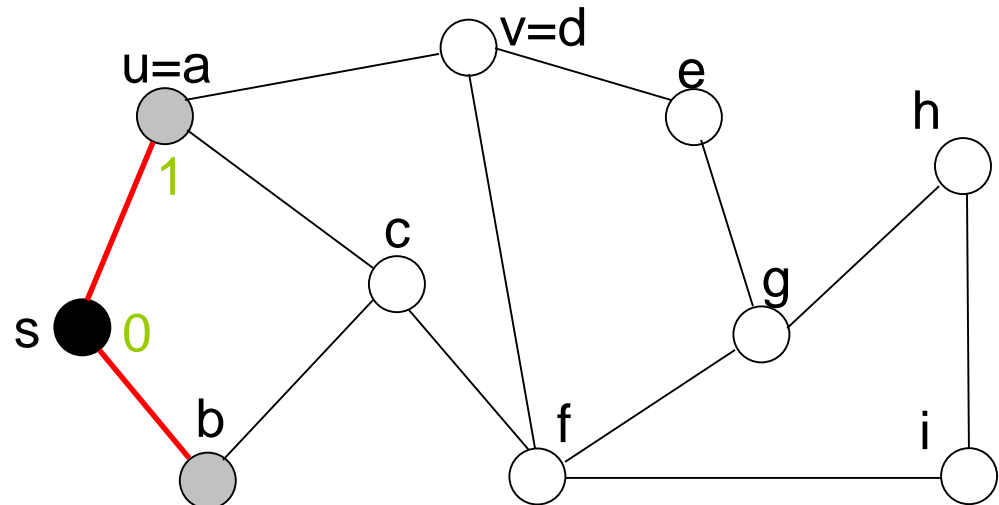
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

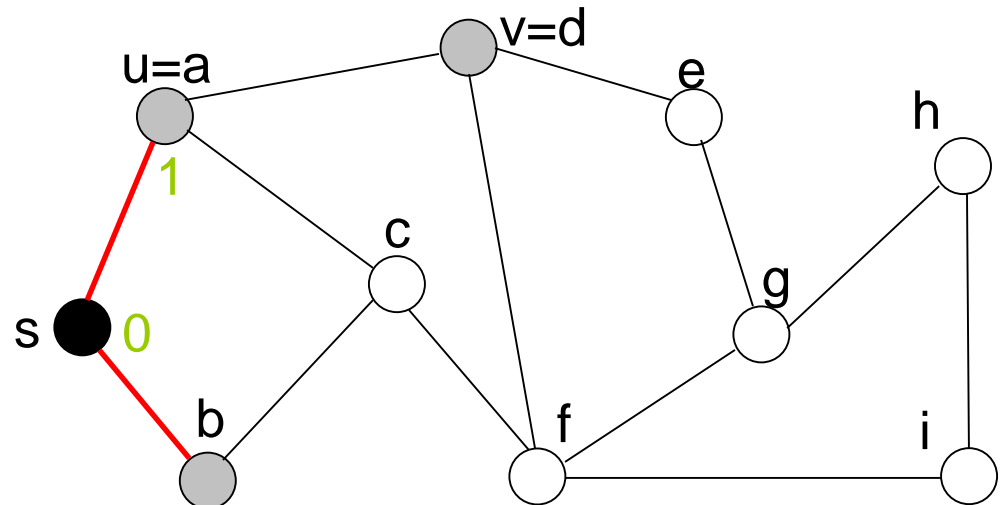
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

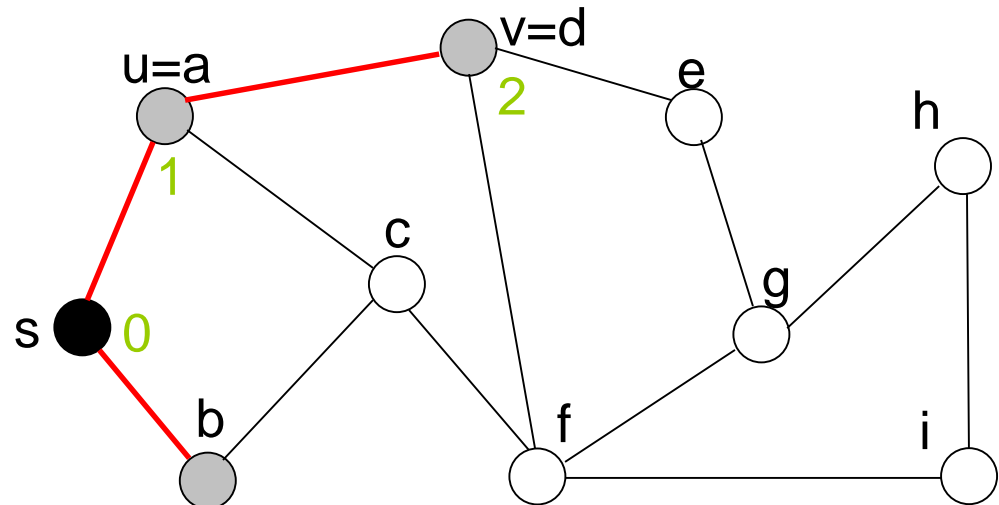
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

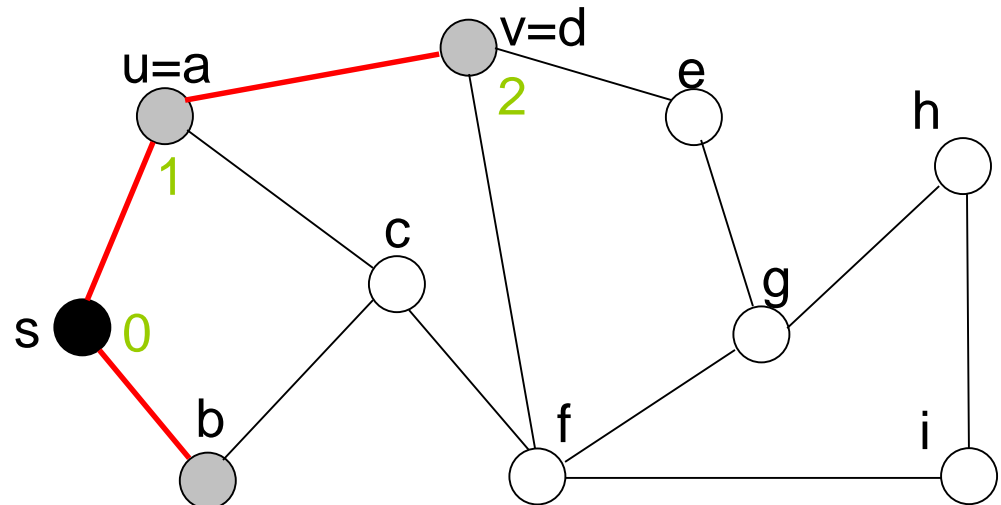
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. **Enqueue**(Q,v)
9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: b, d

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

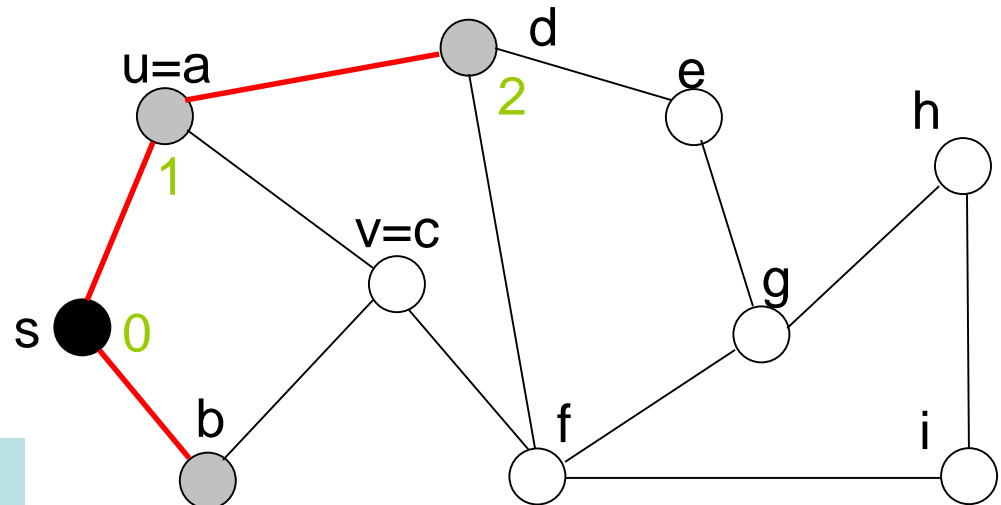
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: b, d

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

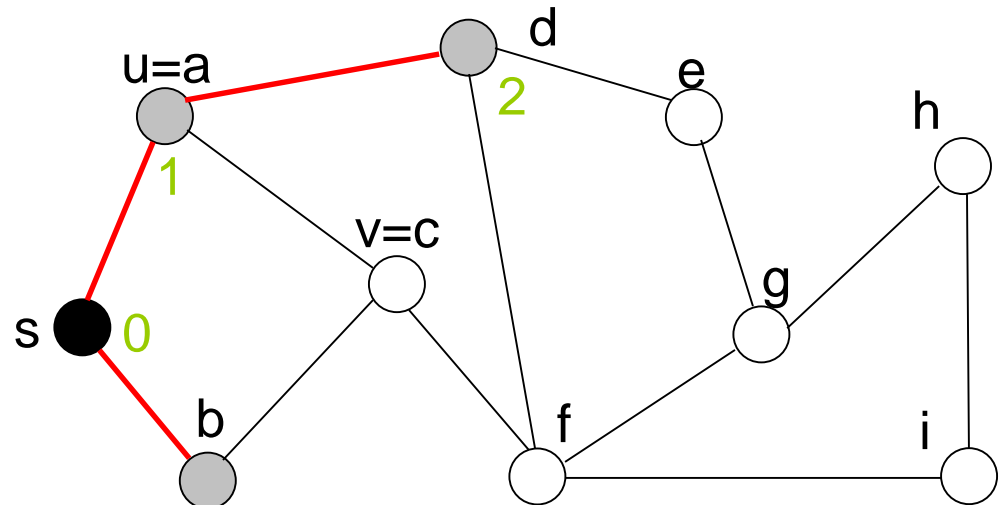
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

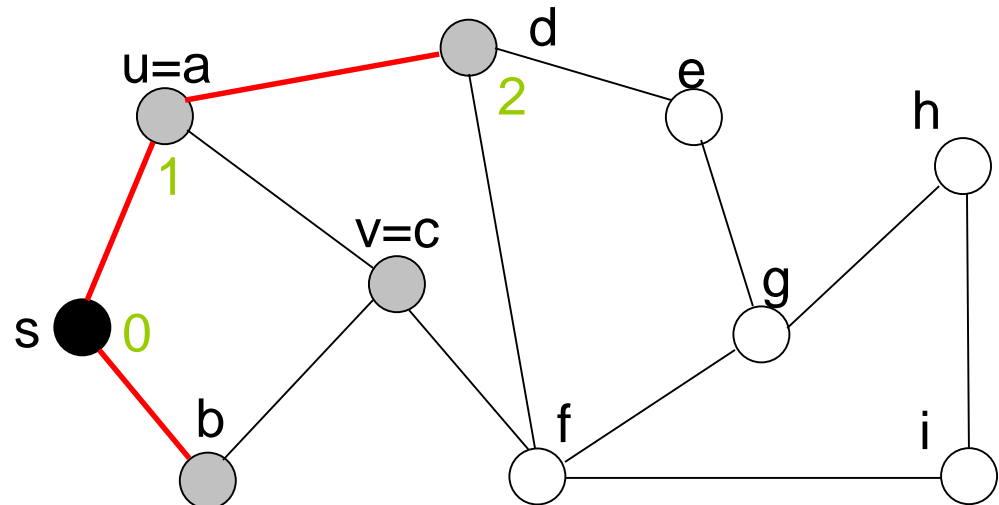
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

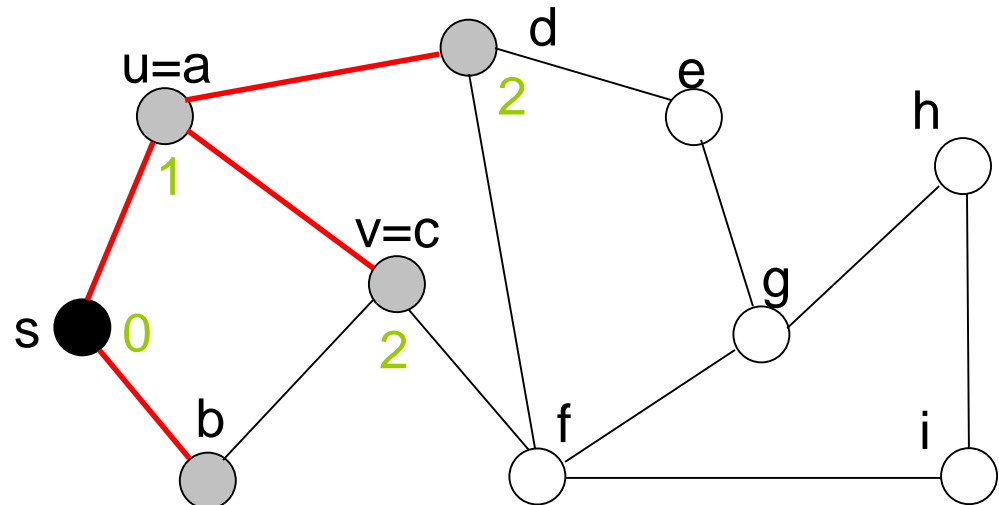
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

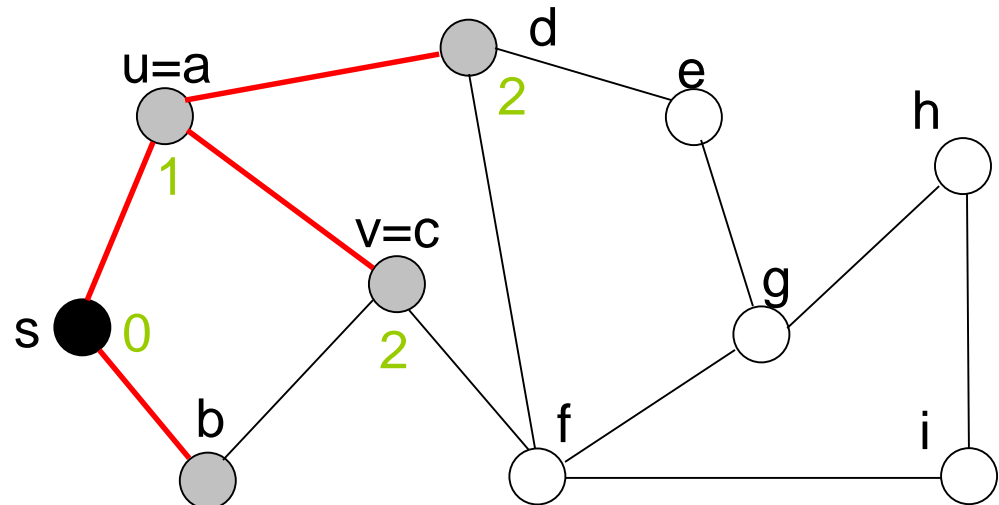


Q: b, d

Breitensuche - Beispiel

BFS(G,s)

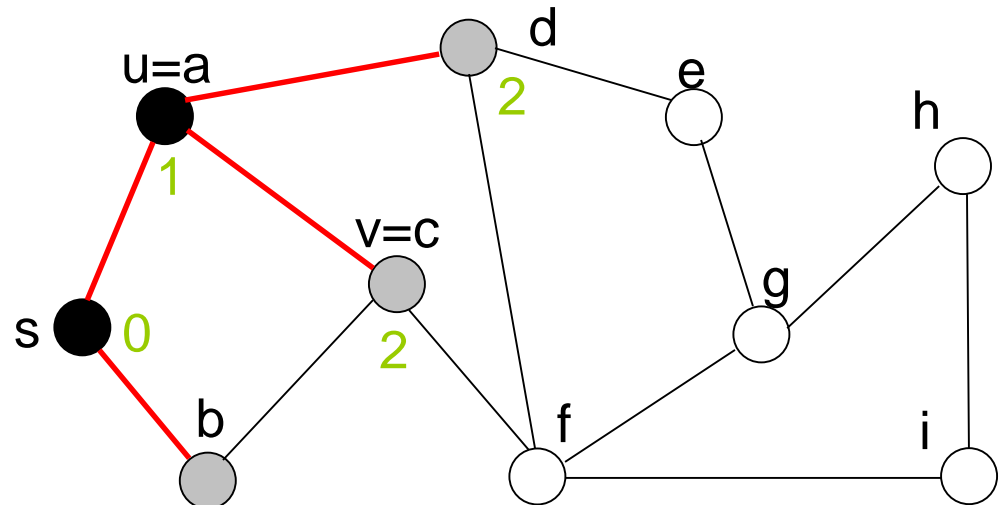
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. **Enqueue**(Q,v)
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

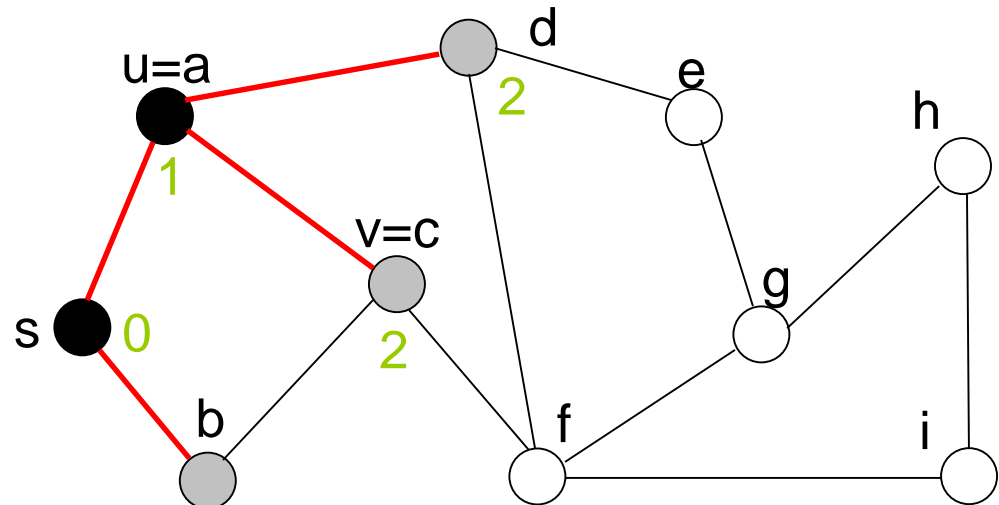


Q: b, d, c

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

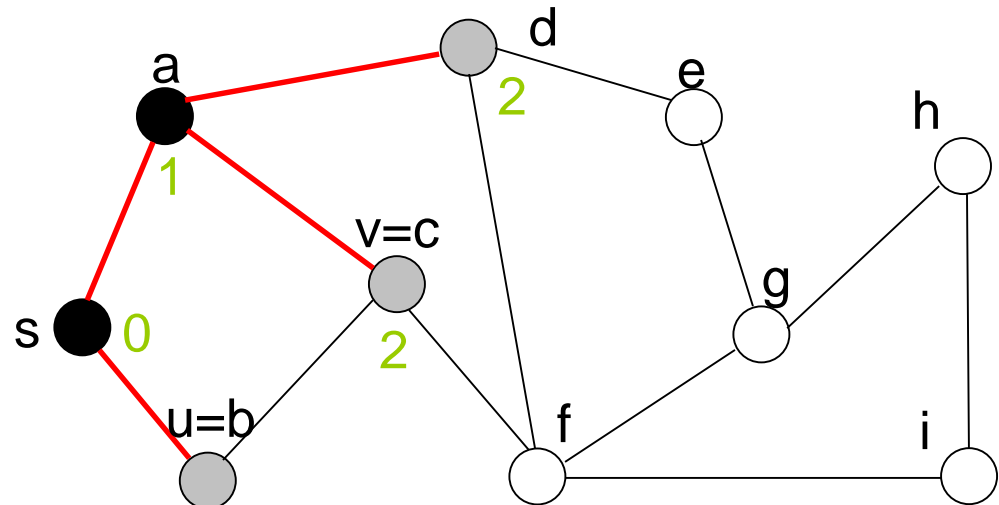


Q: b, d, c

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

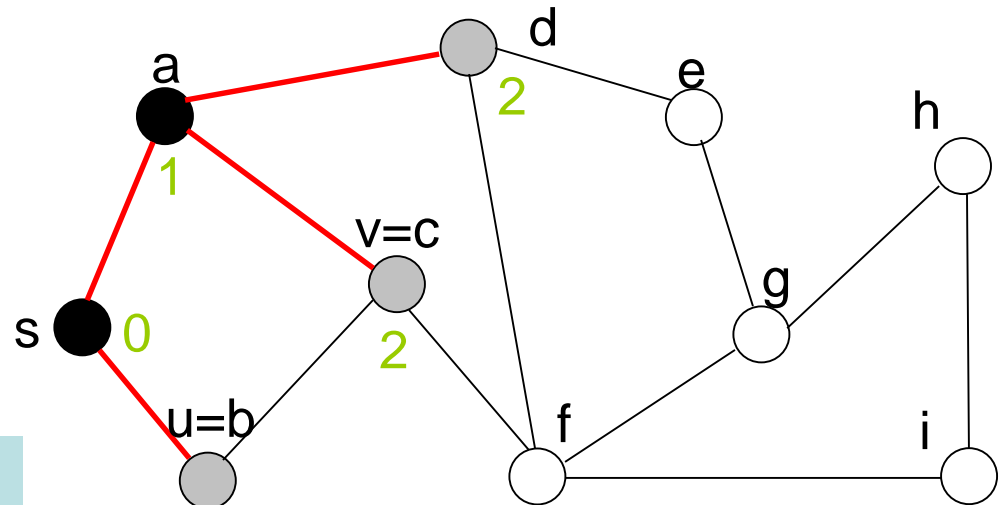


Q: d, c

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

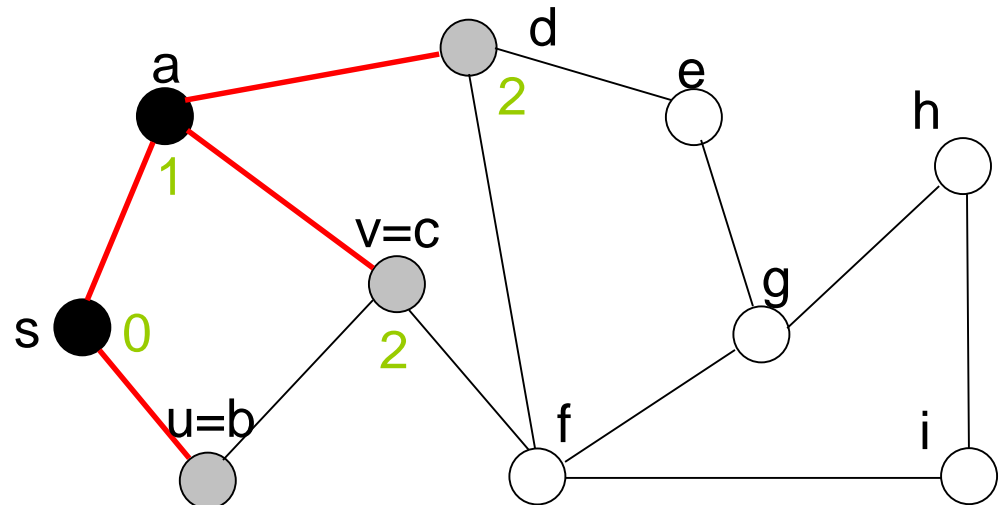


Q: d, c

Breitensuche - Beispiel

BFS(G,s)

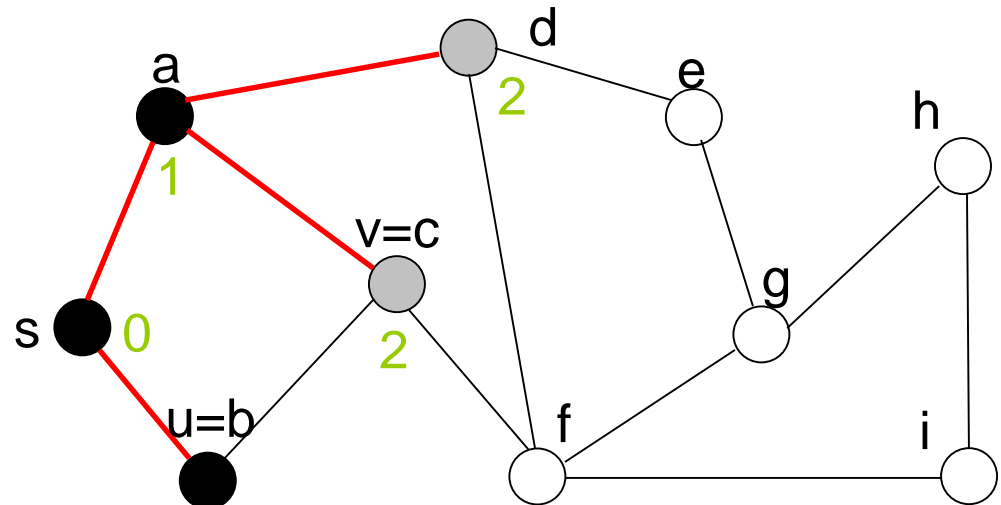
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

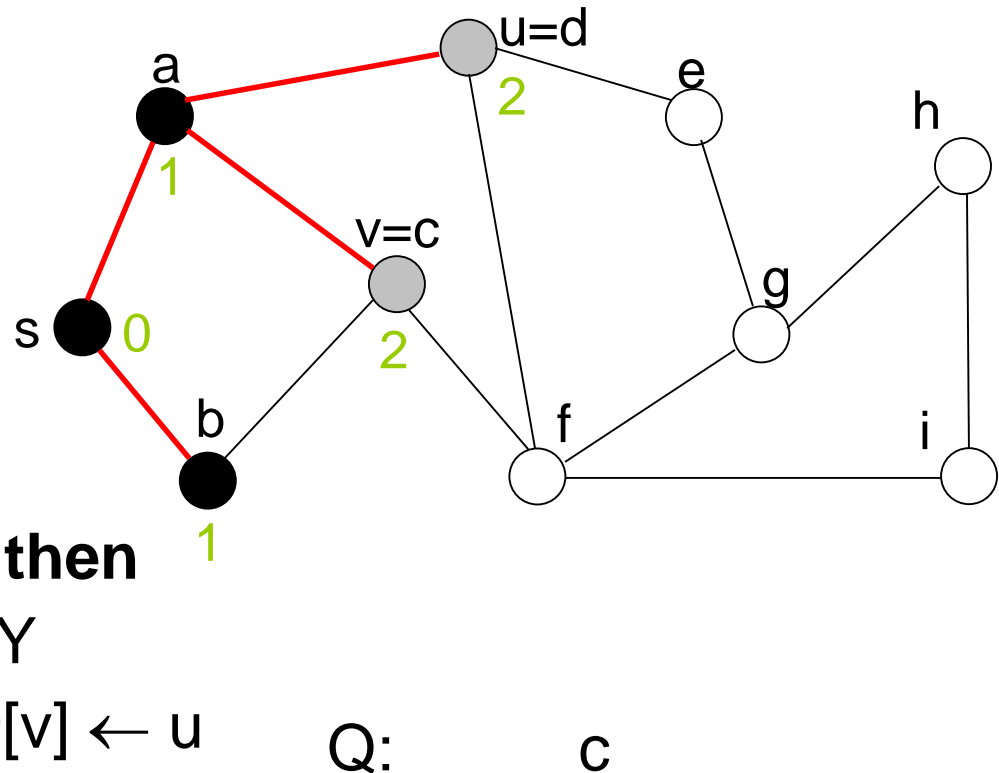
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

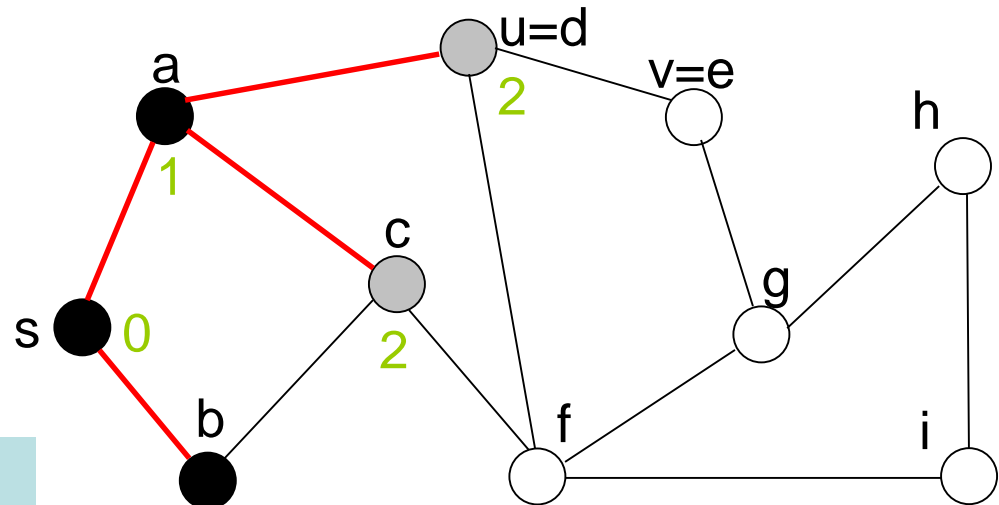
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

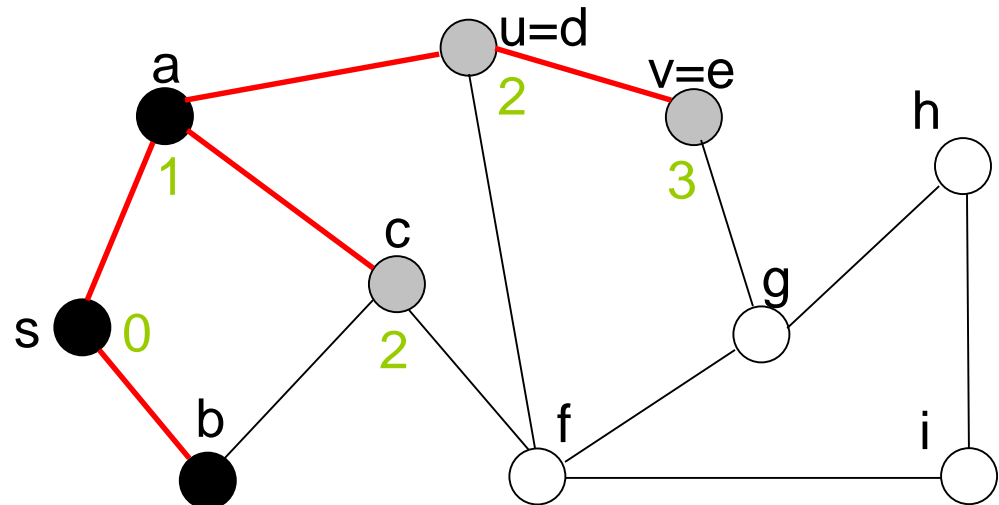
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

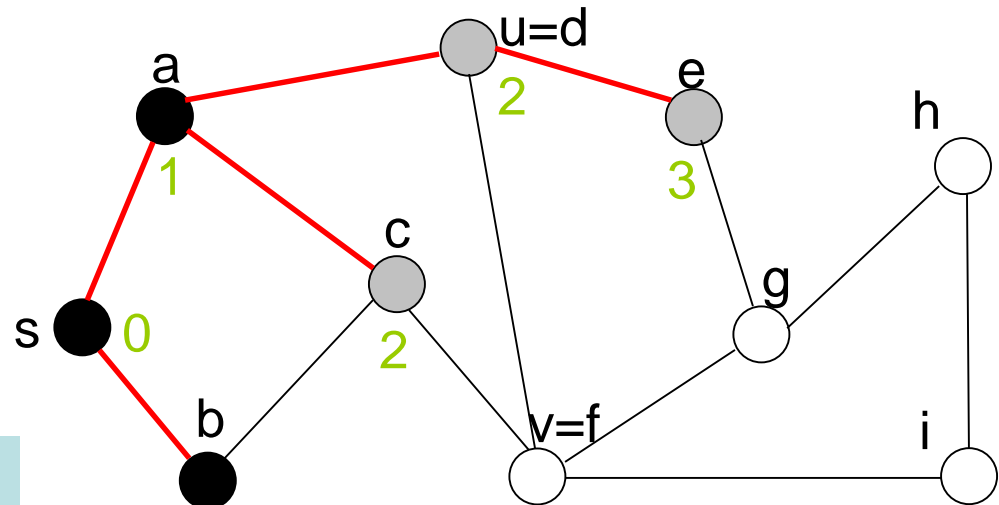


Q: c, e

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

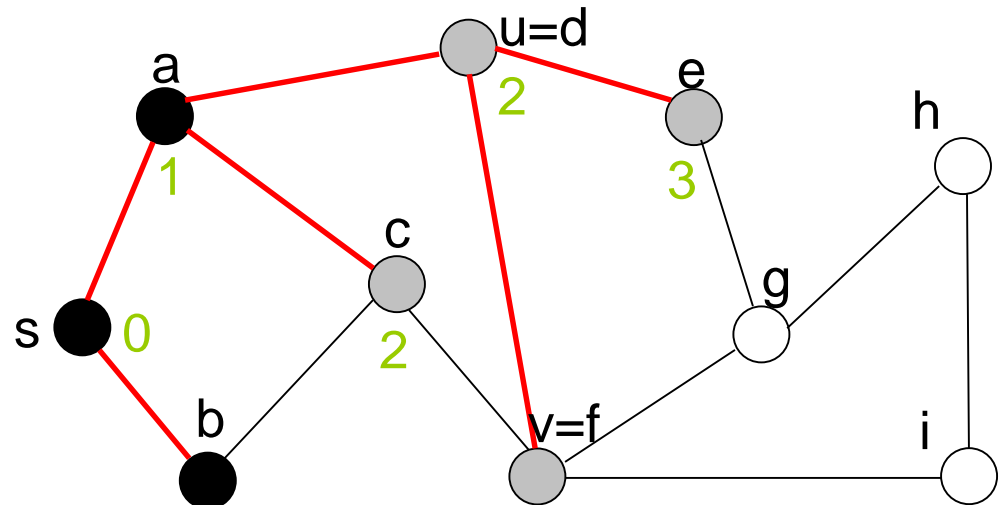


Q: c, e

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Q: c, e, f

Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

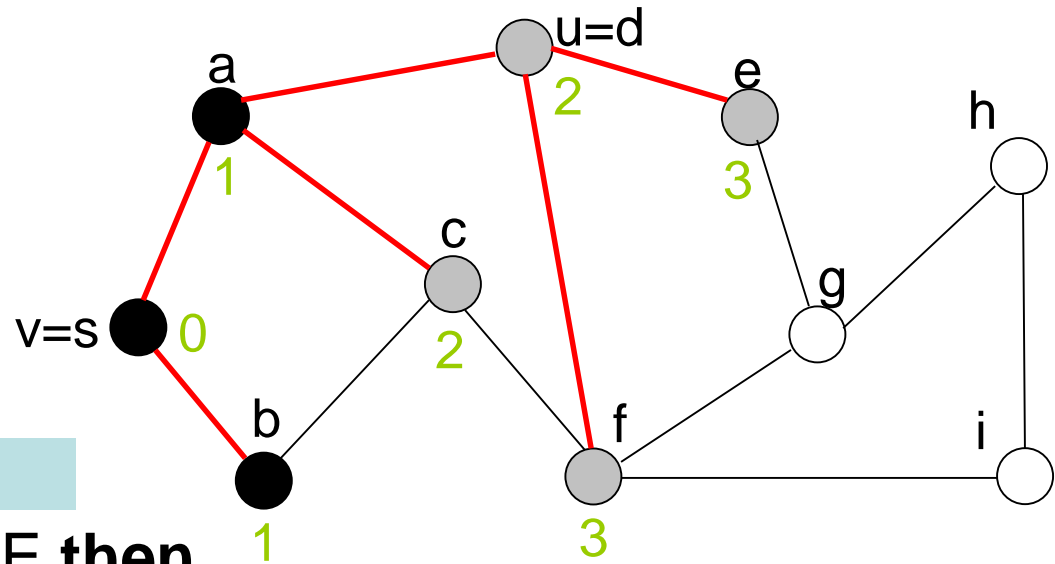
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

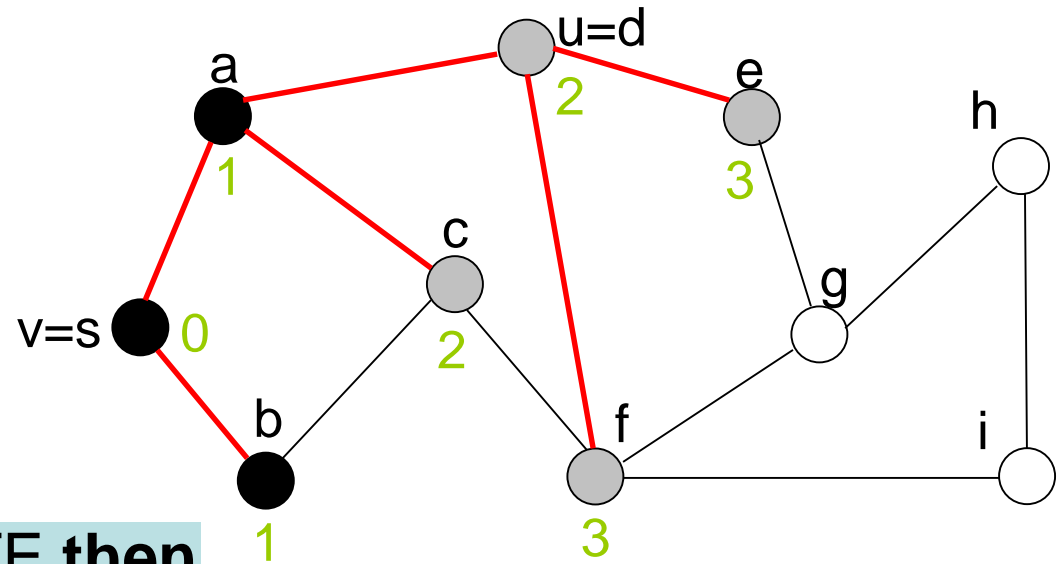
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

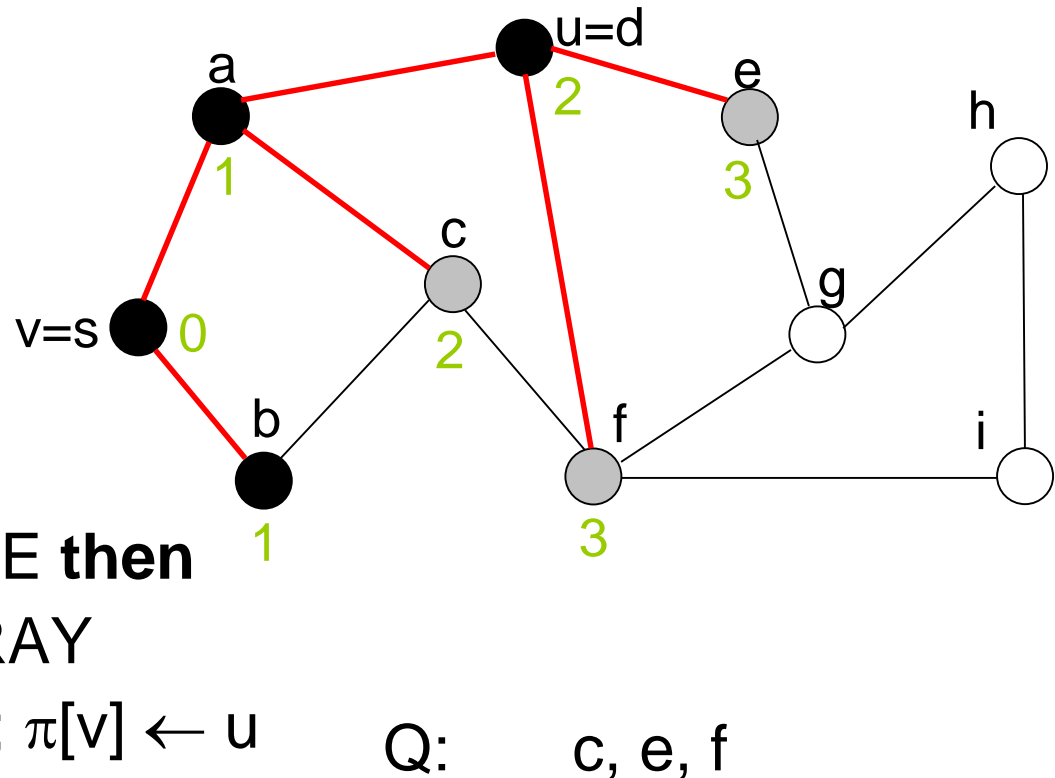


Q: c, e, f

Breitensuche - Beispiel

BFS(G,s)

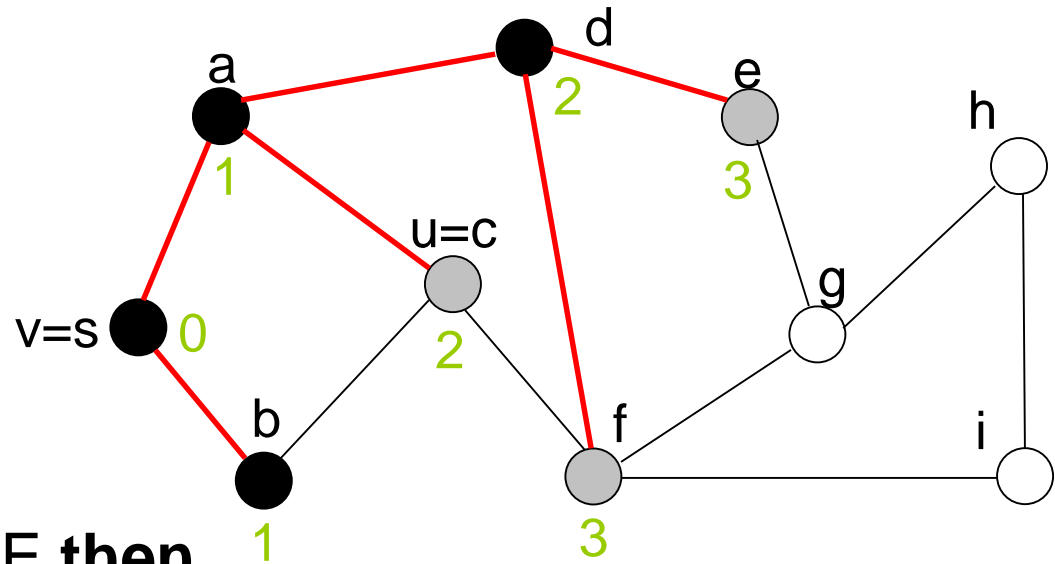
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

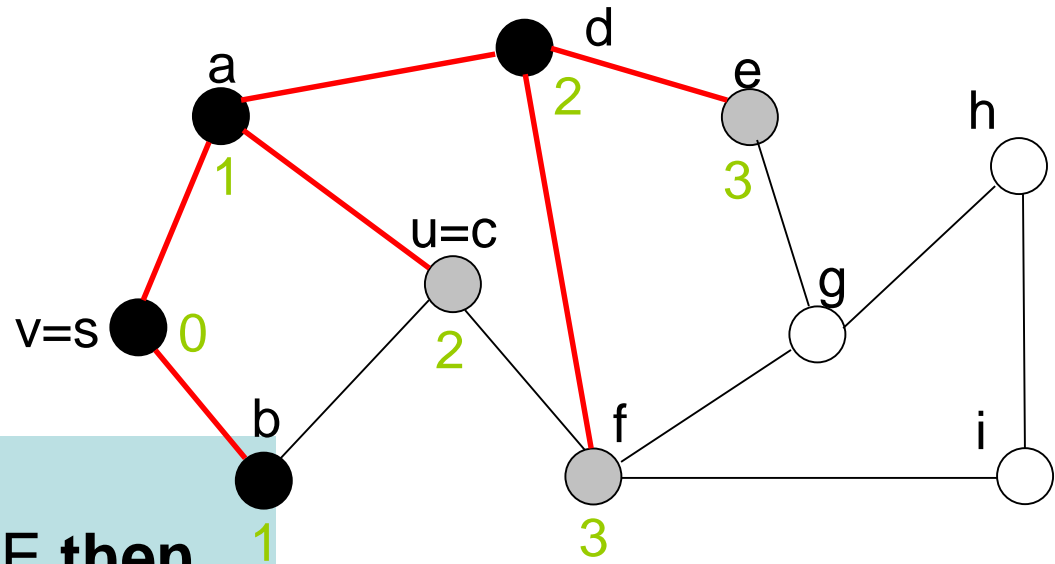


Q: e, f

Breitensuche - Beispiel

BFS(G,s)

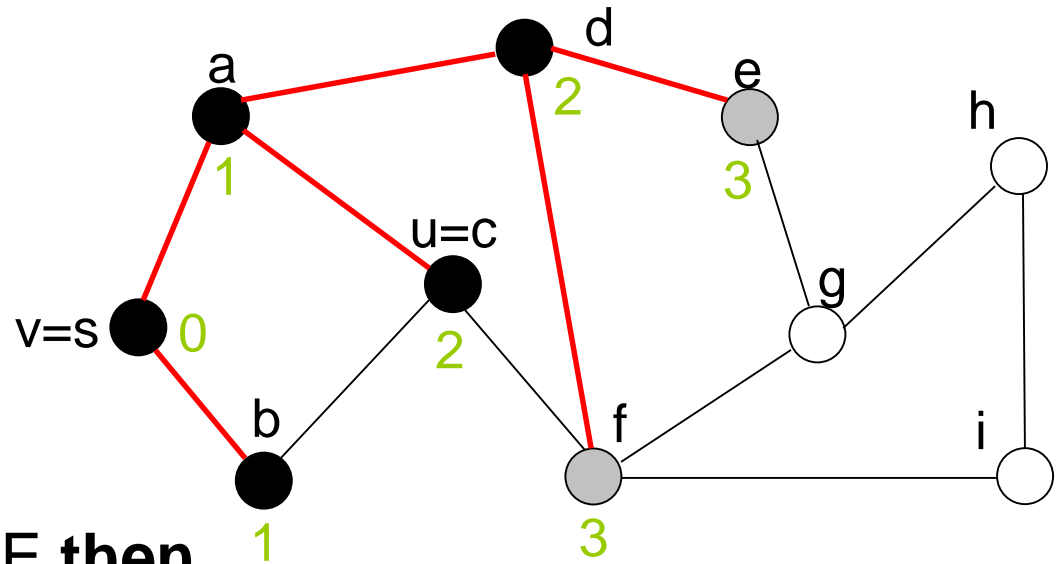
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$

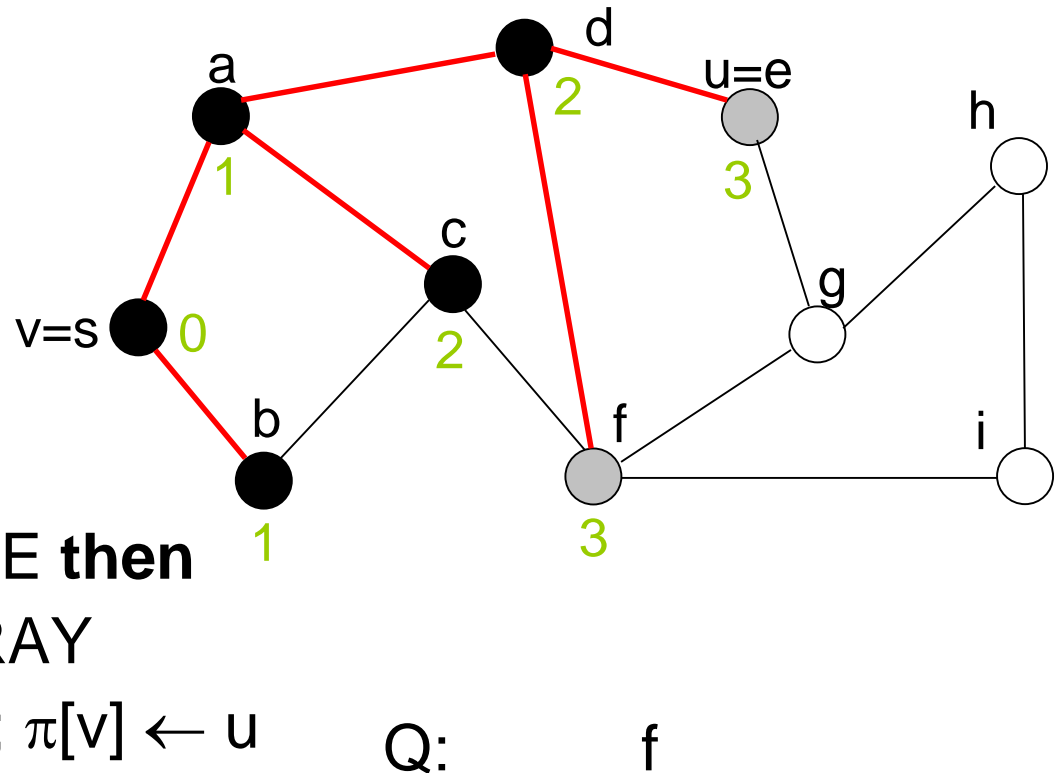


Q: e, f

Breitensuche - Beispiel

BFS(G,s)

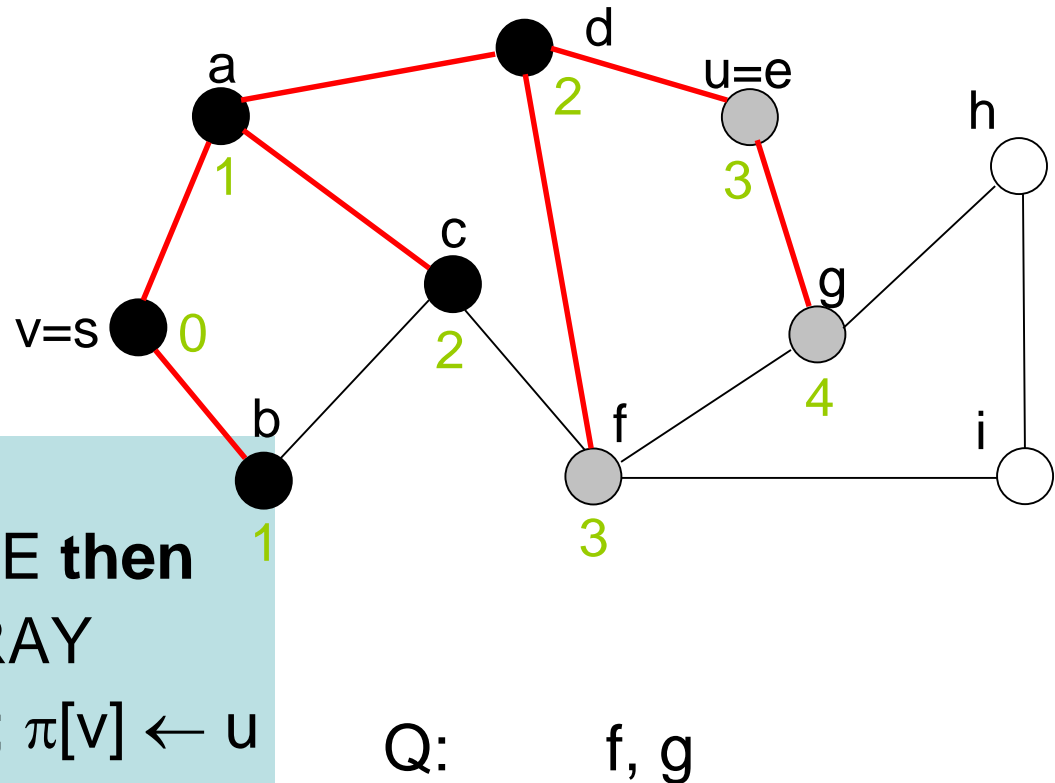
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

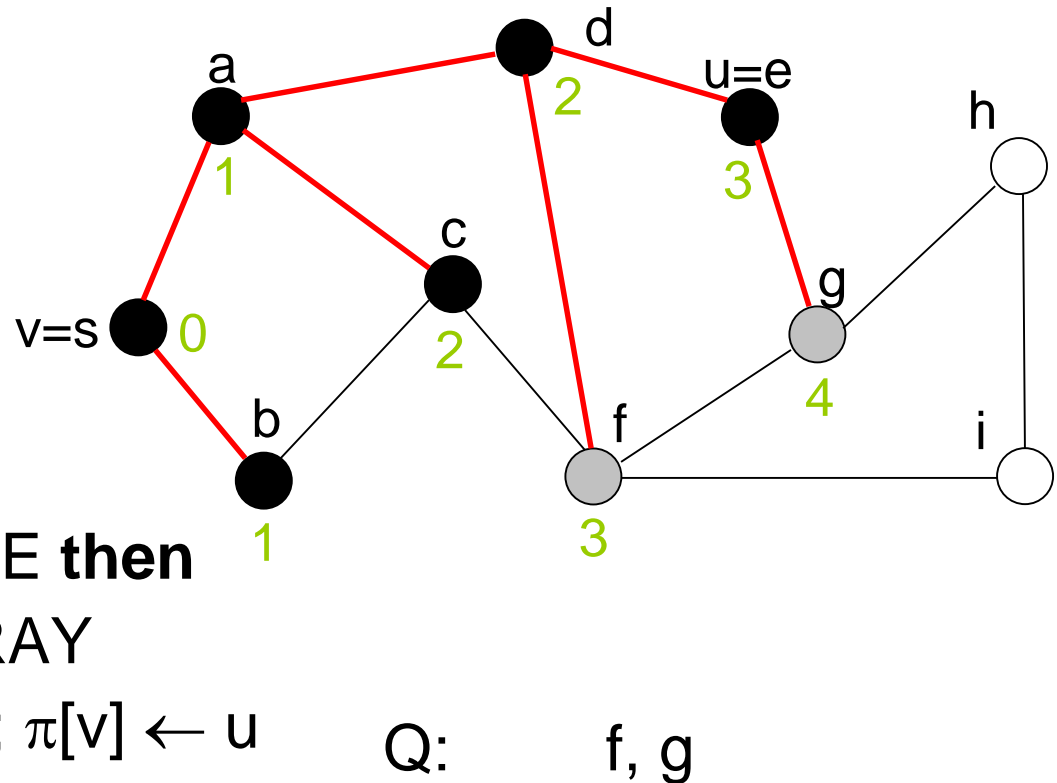
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

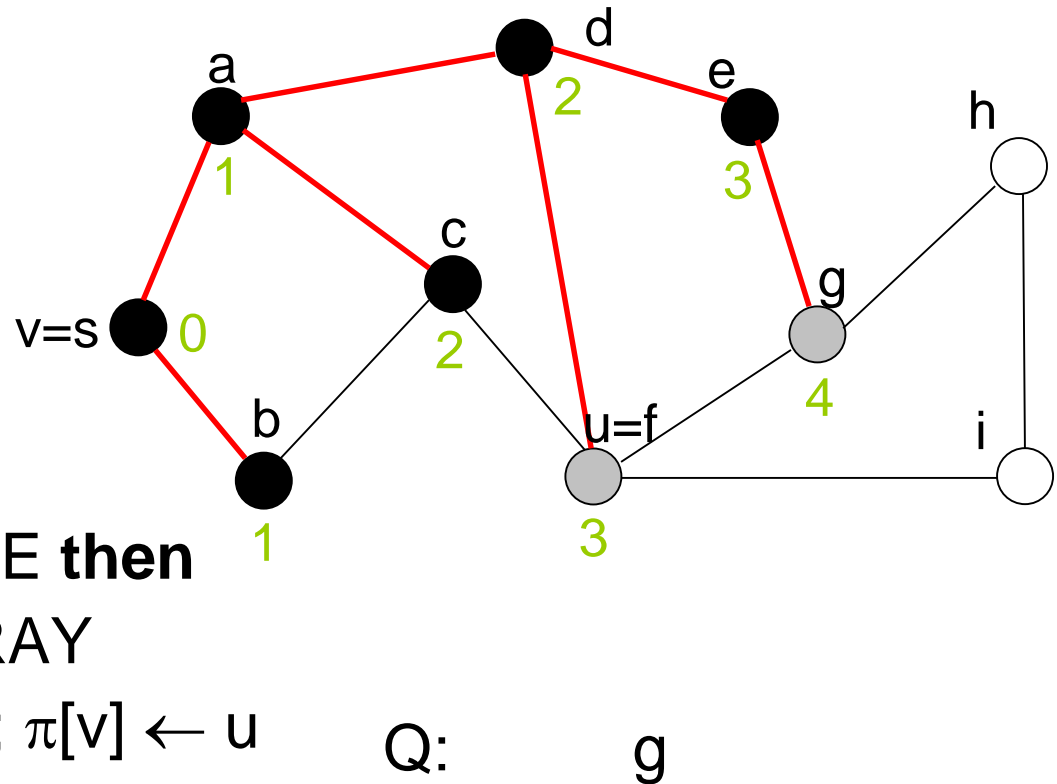
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

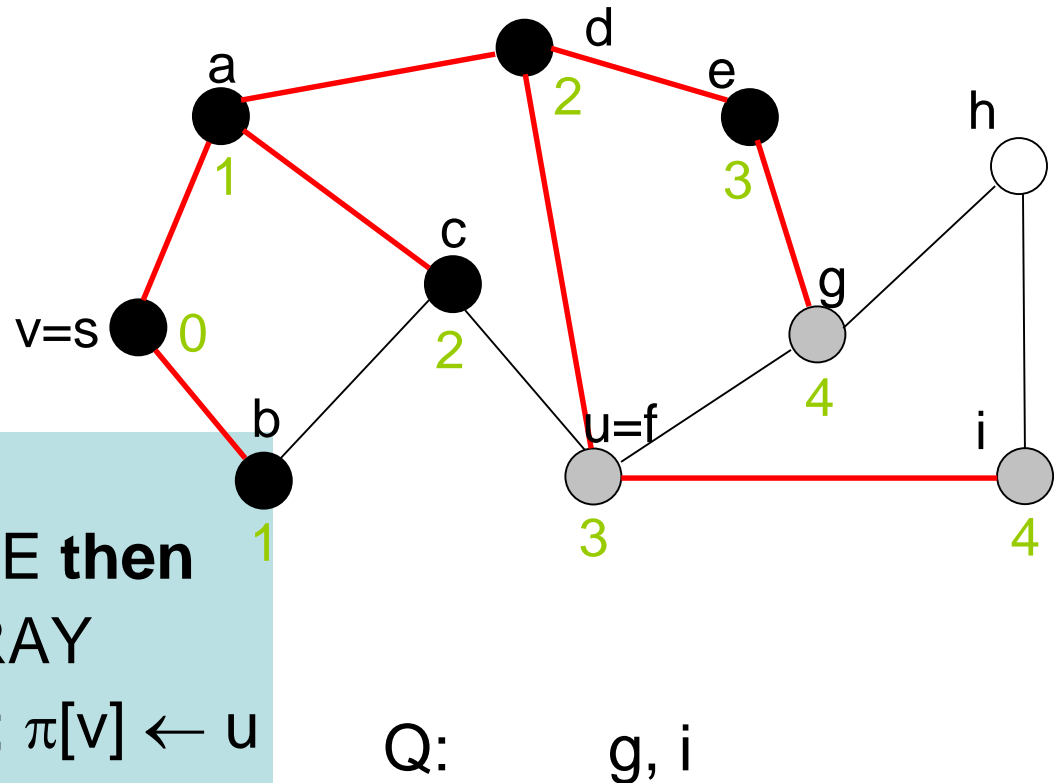
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

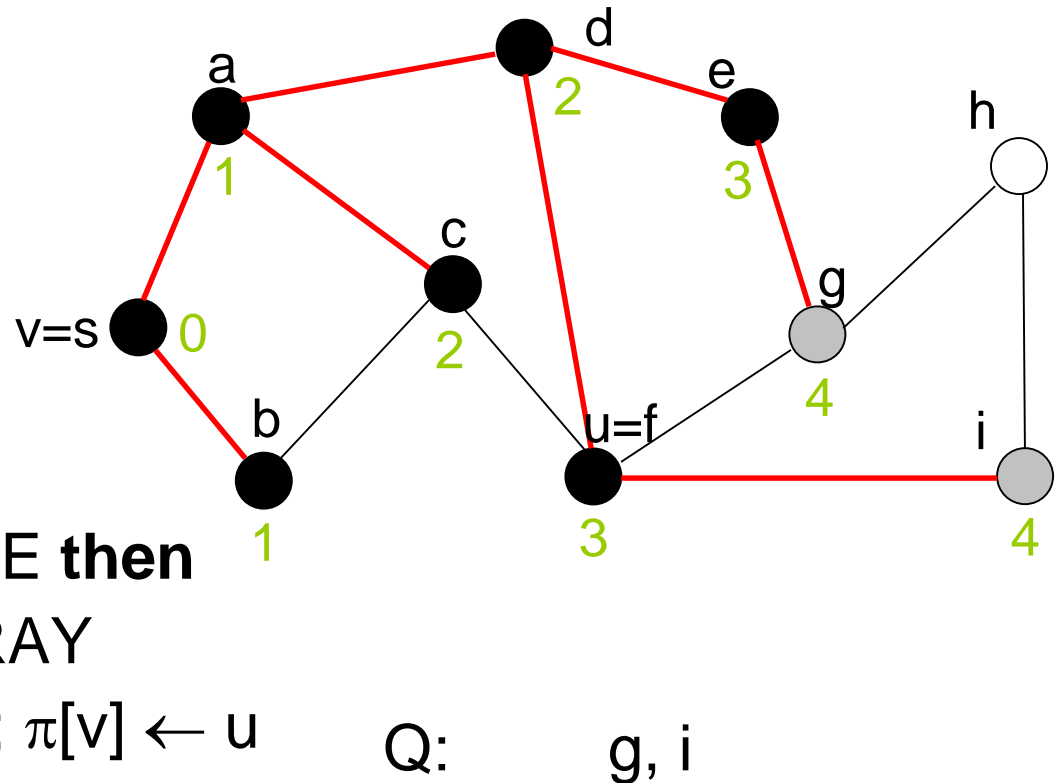
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

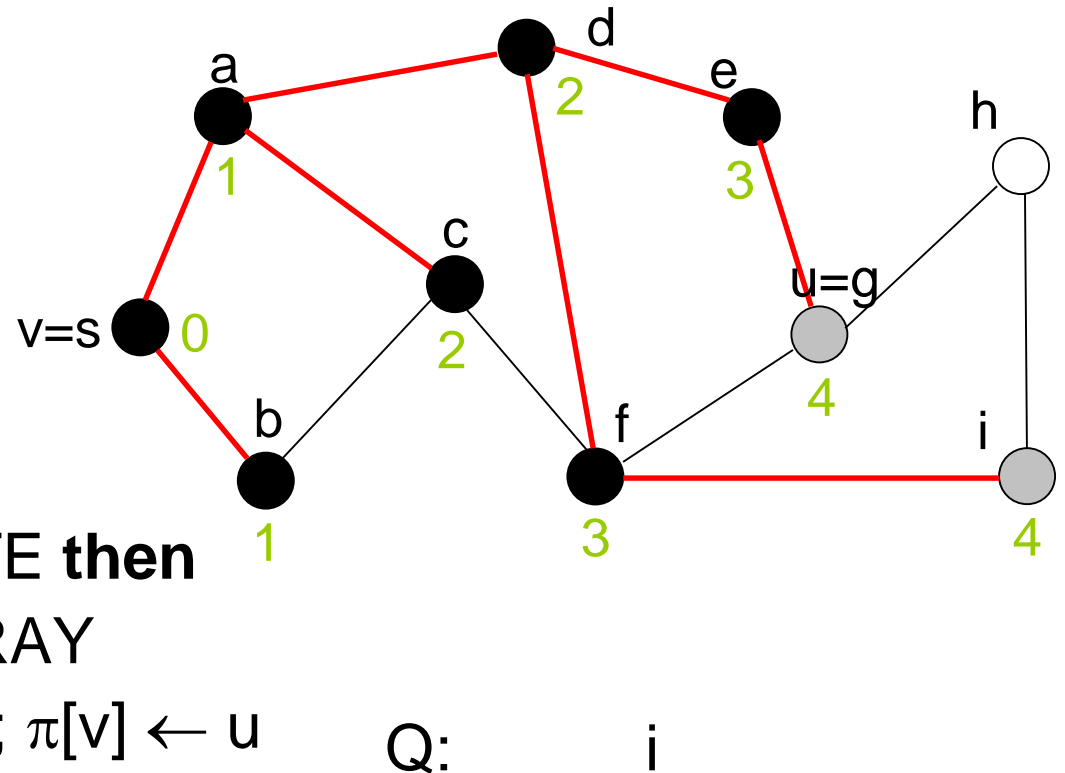
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

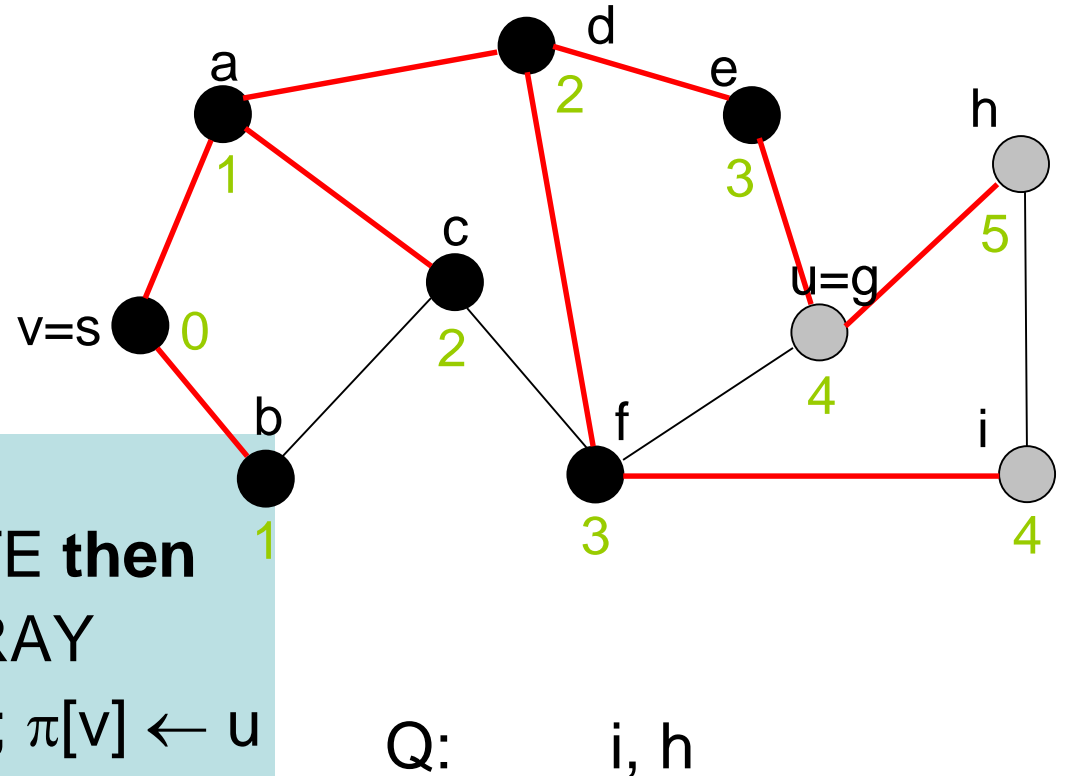
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

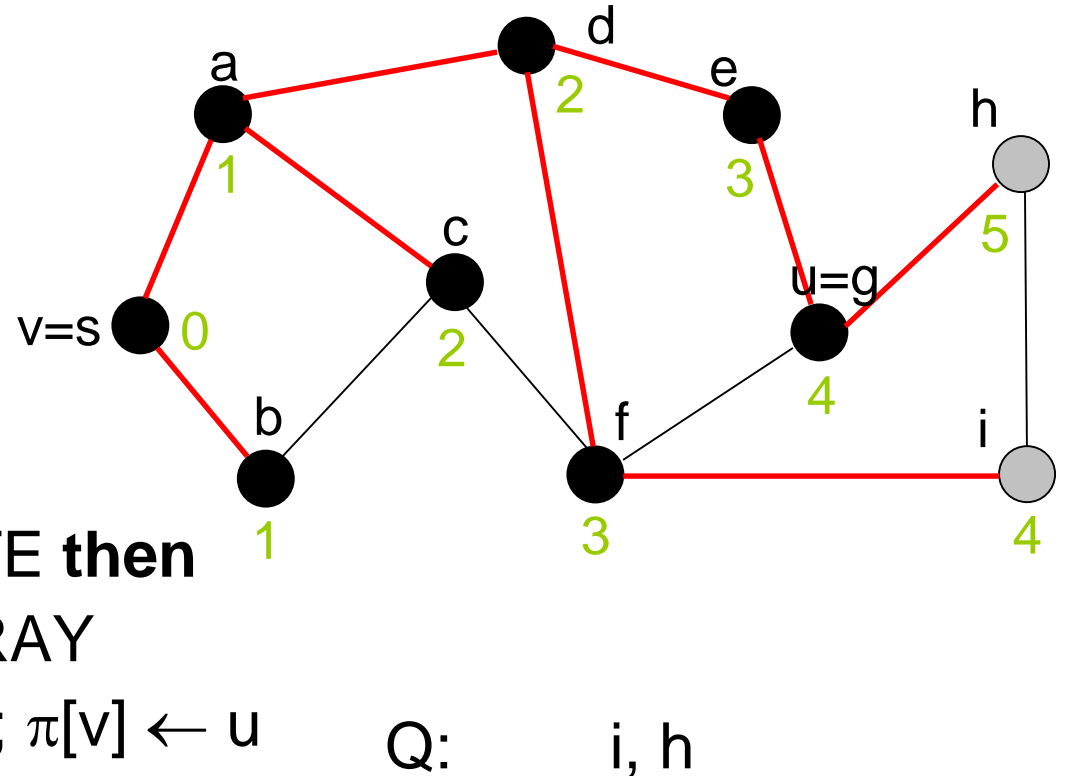
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

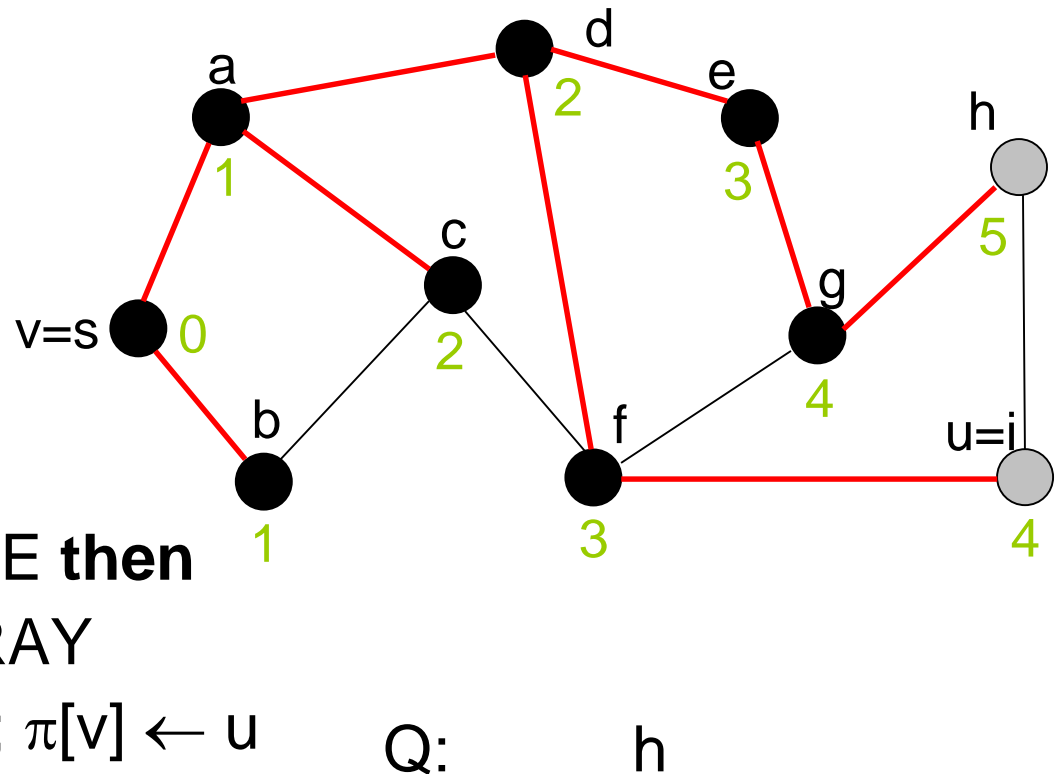
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

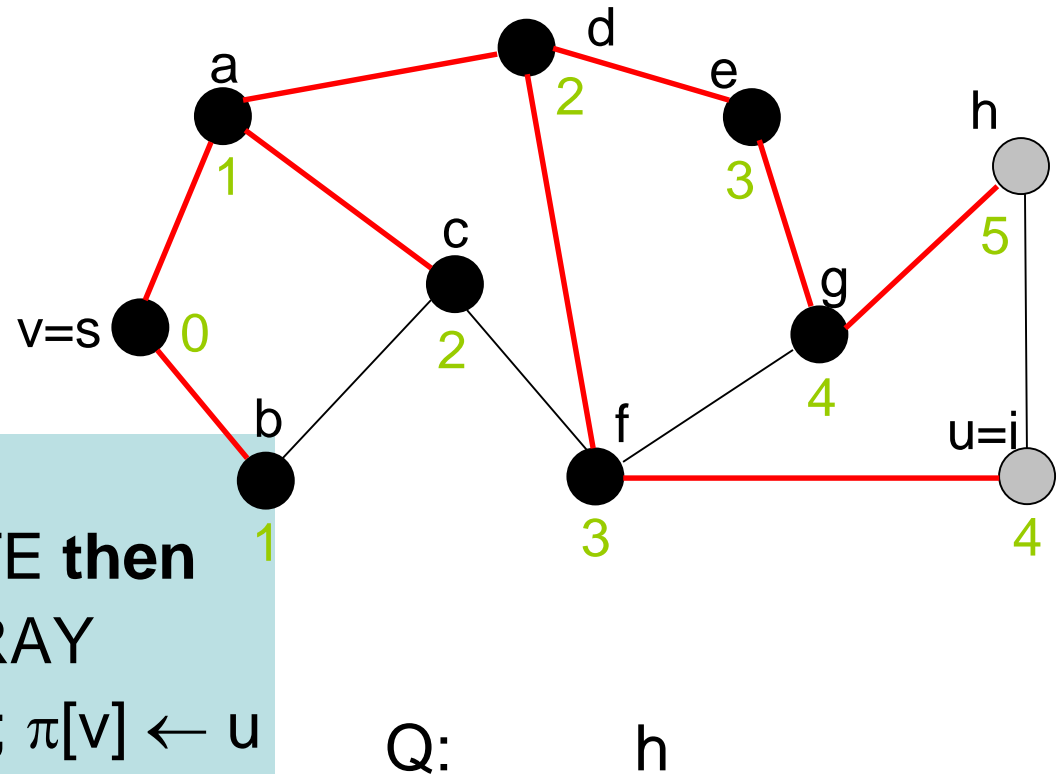
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

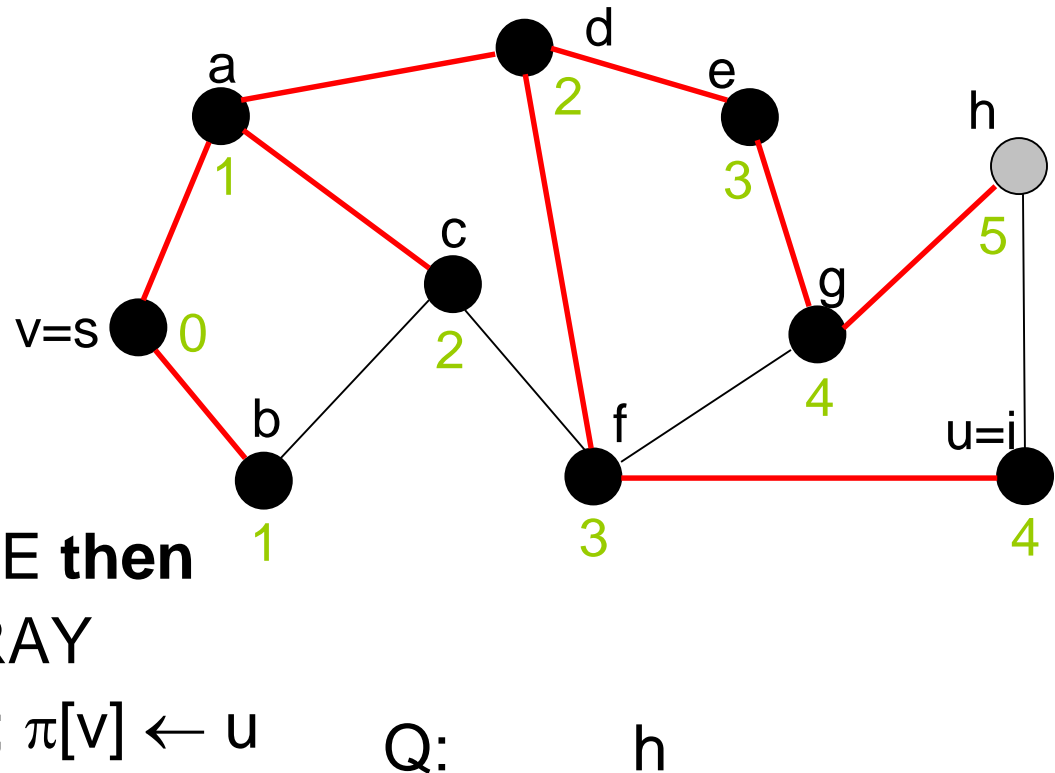
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“

2. **while** $Q \neq \emptyset$ **do**

3. $u \leftarrow \text{Dequeue}(Q)$

4. **for** $v \in \text{Adj}[u]$ **do**

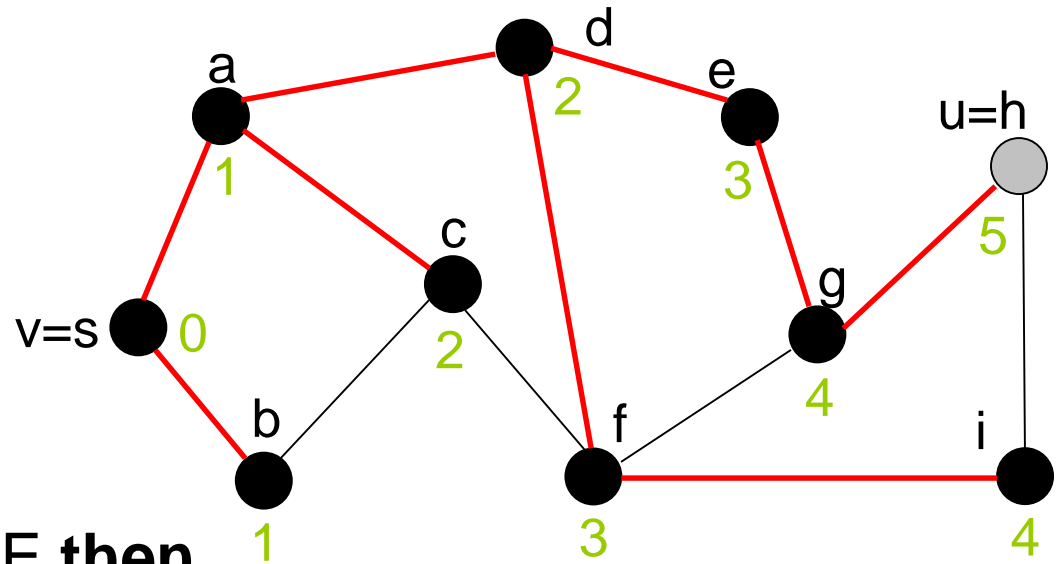
5. **if** $\text{color}[v] = \text{WHITE}$ **then**

6. $\text{color}[v] \leftarrow \text{GRAY}$

7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$

8. $\text{Enqueue}(Q, v)$

9. $\text{color}[u] \leftarrow \text{BLACK}$

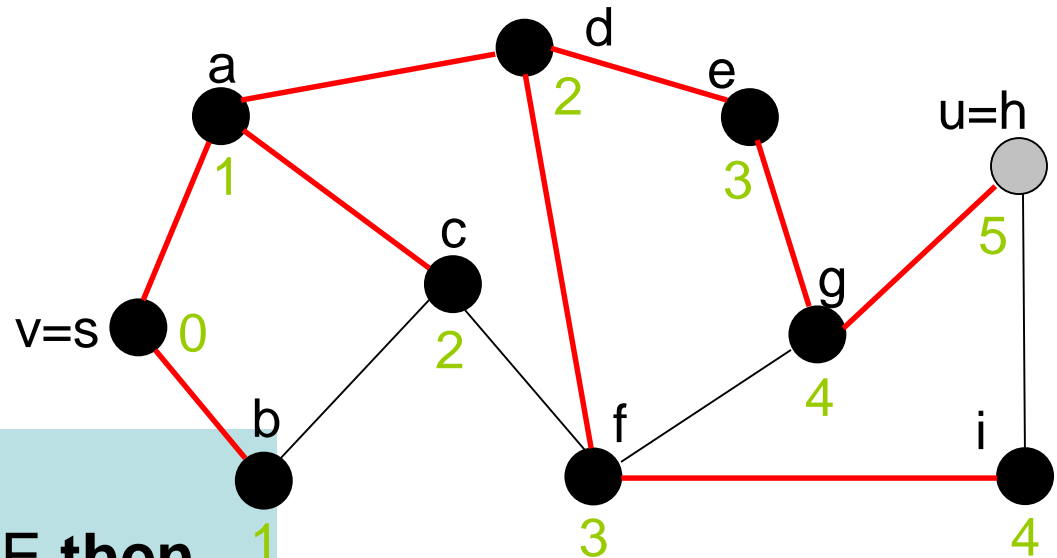


Q:

Breitensuche - Beispiel

BFS(G,s)

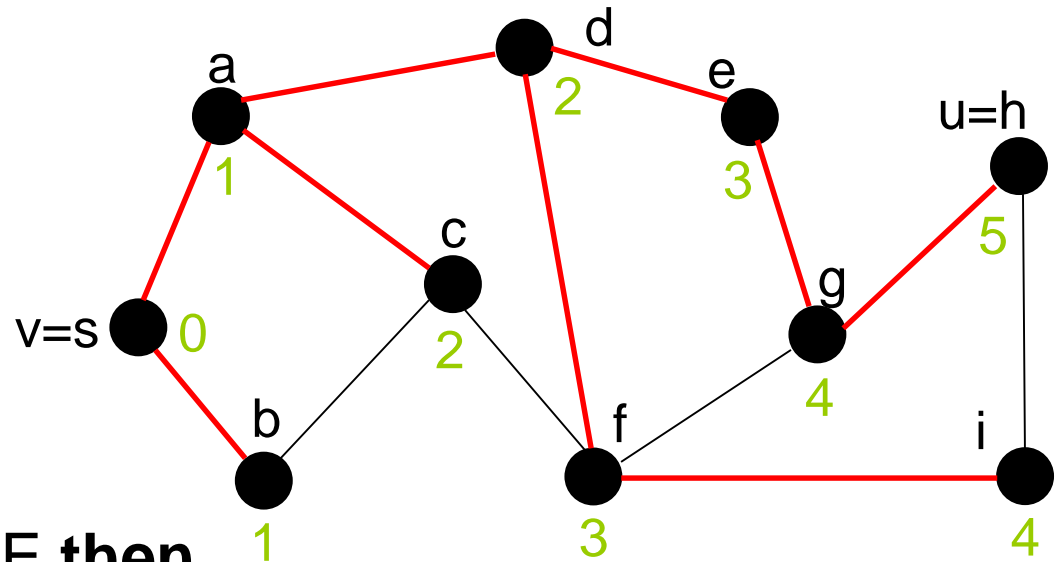
1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche - Beispiel

BFS(G,s)

1. „initialisiere BFS“
2. **while** $Q \neq \emptyset$ **do**
3. $u \leftarrow \text{Dequeue}(Q)$
4. **for** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{WHITE}$ **then**
6. $\text{color}[v] \leftarrow \text{GRAY}$
7. $d[v] \leftarrow d[u] + 1; \pi[v] \leftarrow u$
8. $\text{Enqueue}(Q, v)$
9. $\text{color}[u] \leftarrow \text{BLACK}$



Breitensuche – Laufzeitanalyse

- Zeilen 2-4 jeweils konstante Zeit. Werden $|V| - 1$ mal durchlaufen. Damit Zeit $O(|V|)$.
- Zeilen 5-9 insgesamt konstante Zeit.
- Jeder Knoten wird nur einmal in Queue eingefügt und gelöscht.
- Schleife in Zeilen 12 -17 wird für jeden Eintrag v in Adjazenzlisten nur einmal durchlaufen.
- Zeilen 12-17 pro Durchlauf Zeit $O(1)$.

Breitensuche – Laufzeitanalyse

- Gesamtzeit für Durchläufe der Schleife in Zeilen 11-18 insgesamt $O(|E|)$. Denn Gesamtgröße aller Adjazenzlisten $\Theta(|E|)$.

Satz 14.1: Bei Eingabe von Graph $G=(V,E)$ und Quelle s besitzt Algorithmus BFS Laufzeit $O(|V| + |E|)$

Lemma 14.2: Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph. Sei $s \in V$ beliebig. Für jede Kante $(u, v) \in E$ gilt

$$\delta(s, v) \leq \delta(s, u) + 1.$$

Lemma 14.3: Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph. Sei $s \in V$ beliebig und s, G Eingabe für BFS. Nach Beendigung von BFS gilt für jeden Knoten $v \in V$

$$d[v] \geq \delta(s, v).$$

Lemma 14.4: Die Queue Q enthalte zu einem beliebigen Zeitpunkt des Ablaufs von BFS die Knoten (v_1, \dots, v_r) , wobei v_1 der Kopf $\text{head}[Q]$ und v_r das Ende $\text{tail}[Q]$ sei. Dann gilt

$$d[v_r] \leq d[v_1] + 1 \quad \text{und} \quad d[v_i] \leq d[v_{i+1}], \quad i = 1, \dots, r - 1.$$

Korollar 14.5: Knoten v_i werde während des Ablaufs von BFS vor Knoten v_j in die Queue Q eingefügt. Zum Zeitpunkt, an dem v_j in die Queue Q eingefügt wird, gilt

$$d[v_i] \leq d[v_j]$$

Satz 14.6: Sei $G=(V,E)$ ein gerichteter oder ungerichteter Graph. Sei $s \in V$ beliebig und s,G Eingabe für BFS. Nach Beendigung von BFS gilt für jeden Knoten $v \in V$

$$d[v] = \delta(s, v).$$

Insbesondere sind die von s aus erreichbaren Knoten v die Knoten mit $d[v] < \infty$.

Weiter ist für jeden von s aus erreichbaren Knoten $v \neq s$ ein kürzester Pfad von s zu v gegeben durch einen kürzesten Pfad von s zum Vorgänger $\pi[v]$ von v erweitert um die Kante $(\pi[v], v)$.

Breitensuchbäume

Betrachten nach BFS mit Eingabe G, s den Graphen

$G_\pi = (V_\pi, E_\pi)$ mit

$$V_\pi = \{v \in V \mid \pi[v] \neq \text{NIL}\} \cup \{s\}.$$

und

$$E_\pi := \{(\pi[v], v) \mid v \in V_\pi \setminus \{s\}\}.$$

Dann gilt

Satz 14.7: $G_\pi = (V_\pi, E_\pi)$ ist ein Baum. V_π enthält genau die von s aus erreichbaren Knoten in G . Für jeden Knoten $v \in V_\pi$ ist der eindeutige Pfad von s zu v in G_π ein kürzester Pfad von s zu v in G .

Tiefensuche

- Suche zunächst „tiefer“ im Graph
- Neue Knoten werden immer vom zuletzt gefundenen Knoten entdeckt
- Sind alle adjazenten Knoten des zuletzt gefundenen Knoten v bereits entdeckt, springe zurück zum Knoten, von dem aus v entdeckt wurde
- Wenn irgendwelche unentdeckten Knoten übrigbleiben, starte Tiefensuche von einem dieser Knoten

Invariante Tiefensuche:

- Zu Beginn: alle Knoten weiß
- Entdeckte Knoten werden grau
- Abgearbeitete Knoten werden schwarz
- Zwei Zeitstempel: $d[v]$ und $f[v]$ (liegen zwischen 1 und $2|V|$)
- $d[v]$: v ist entdeckt
- $f[v]$: v ist abgearbeitet

Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$

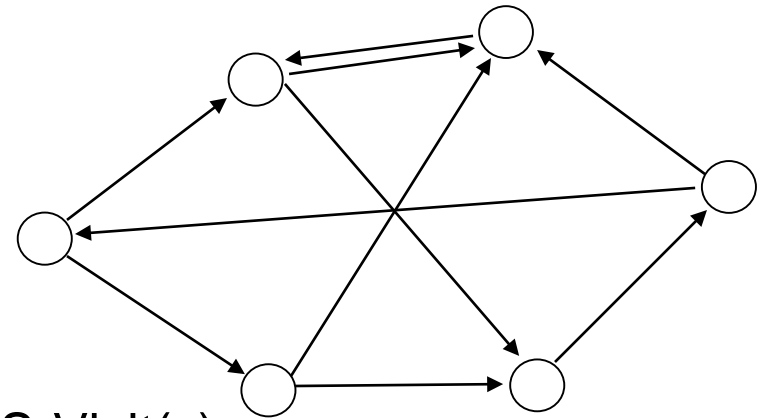
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



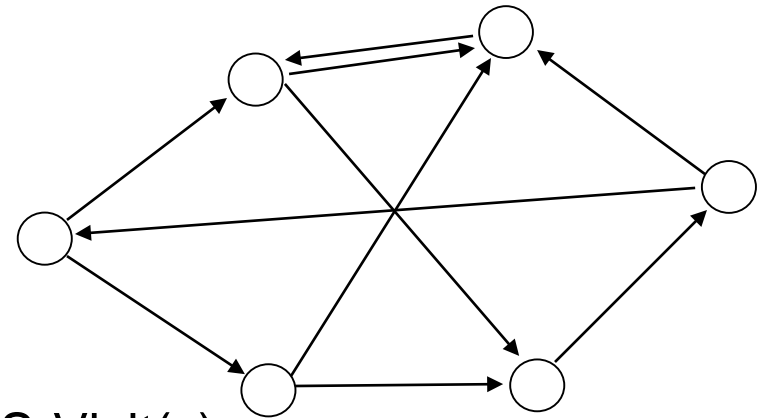
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



Tiefensuche

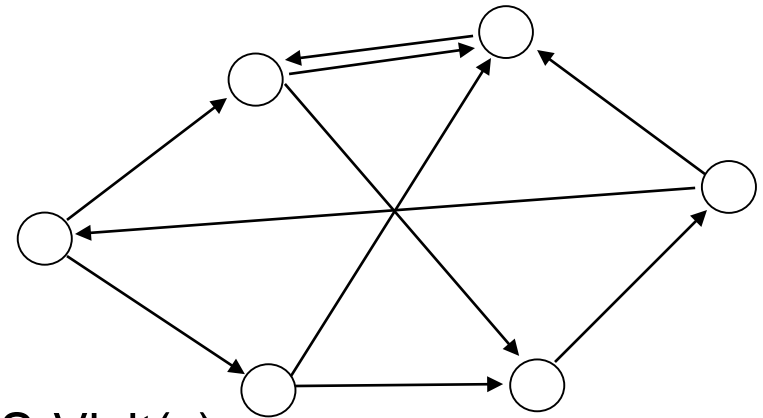
DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

time=0

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



Tiefensuche

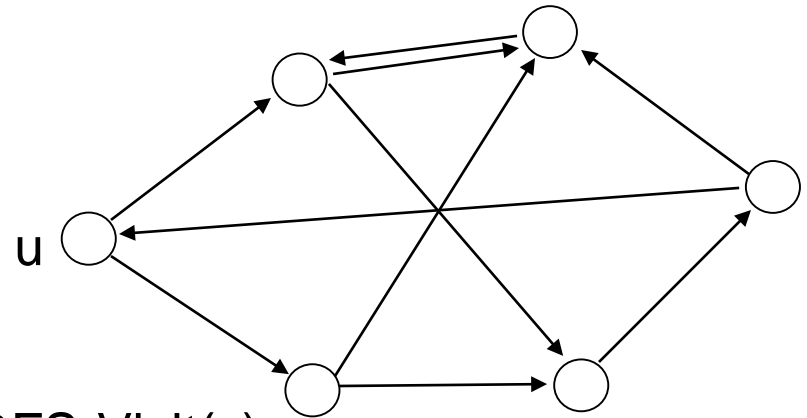
DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

time=0

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



Tiefensuche

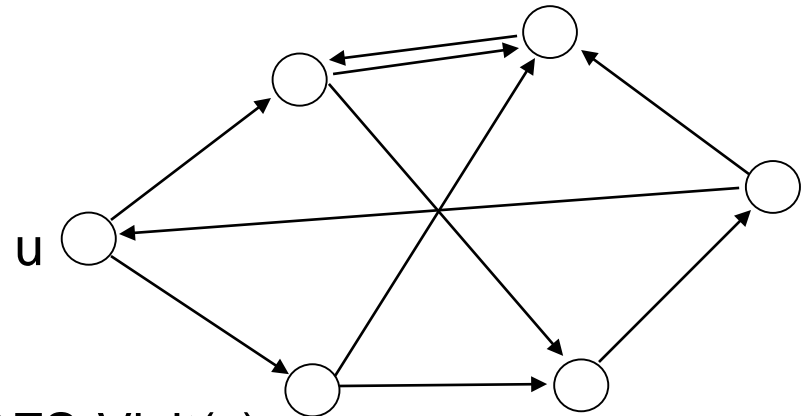
DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

time=0

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



Tiefensuche

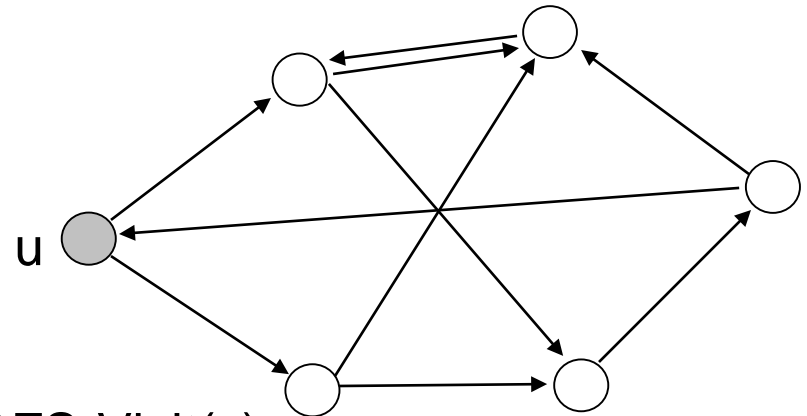
DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

time=0

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



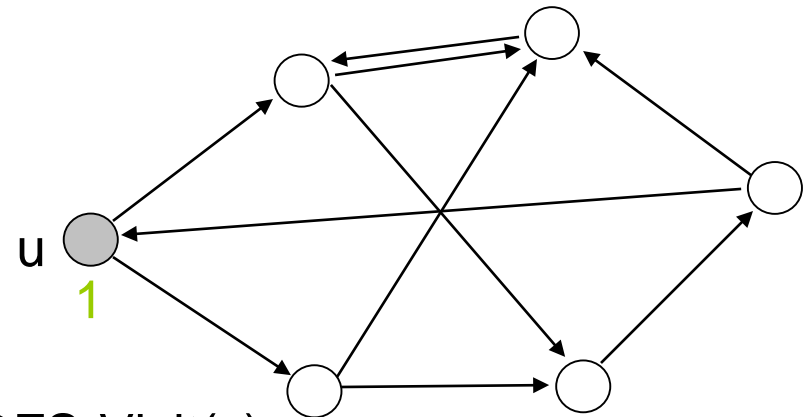
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



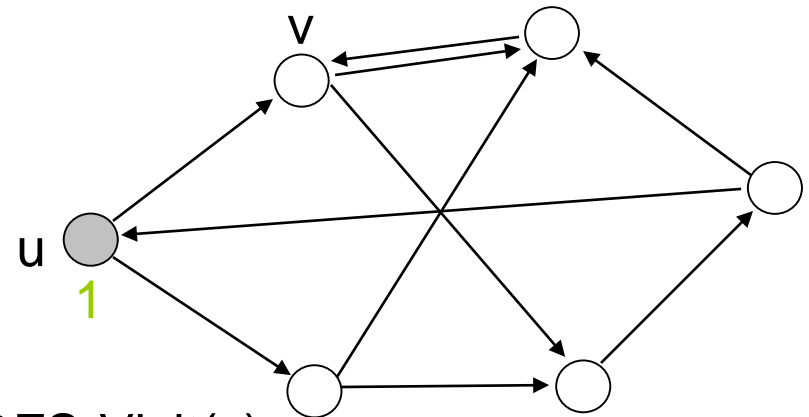
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



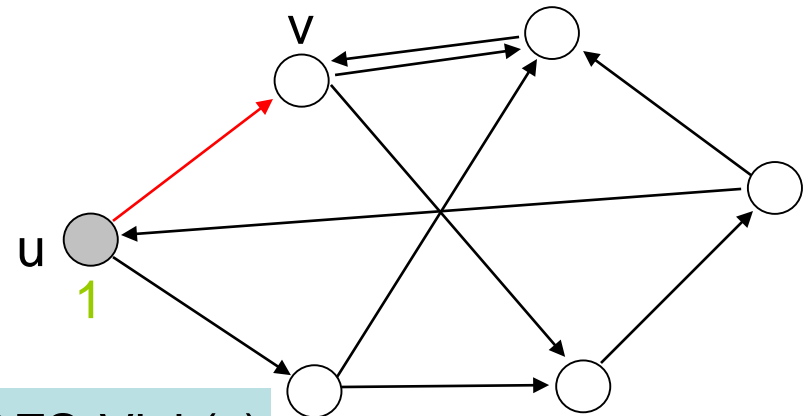
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



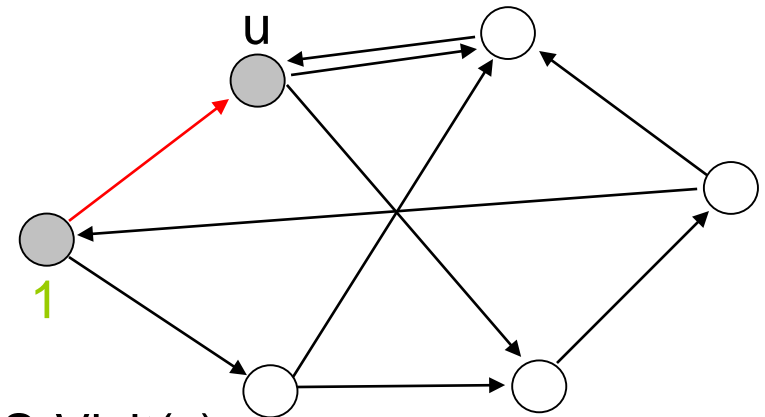
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



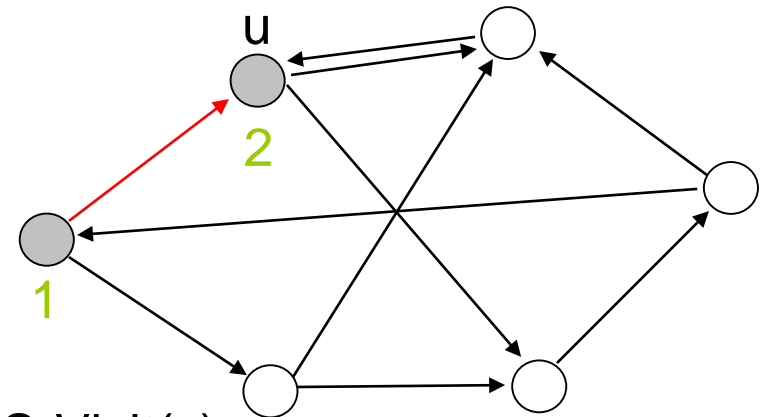
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



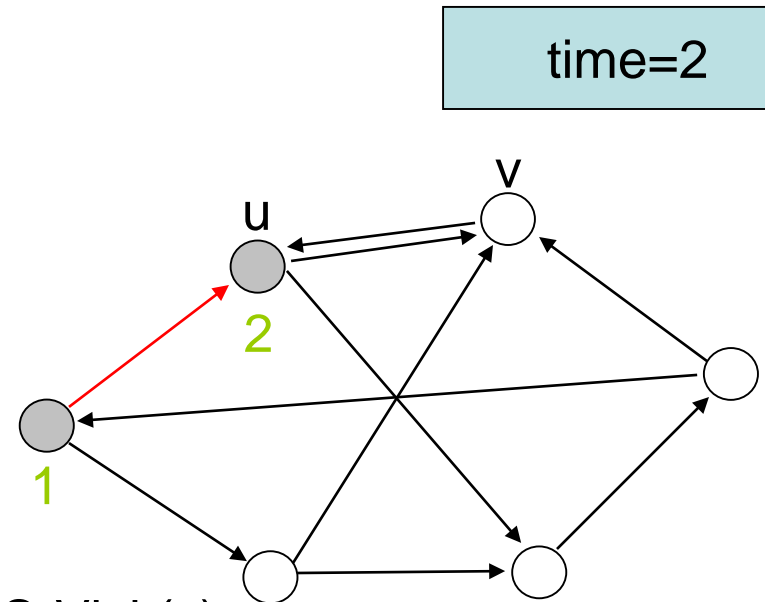
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



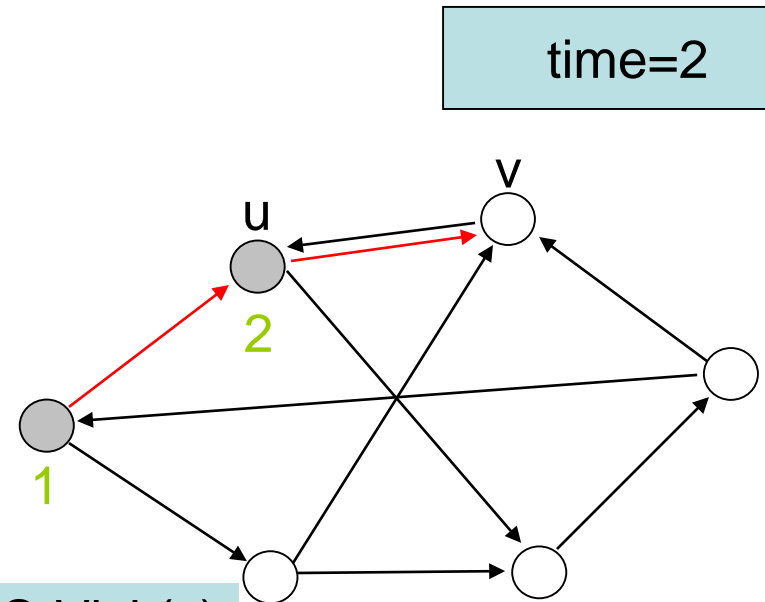
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



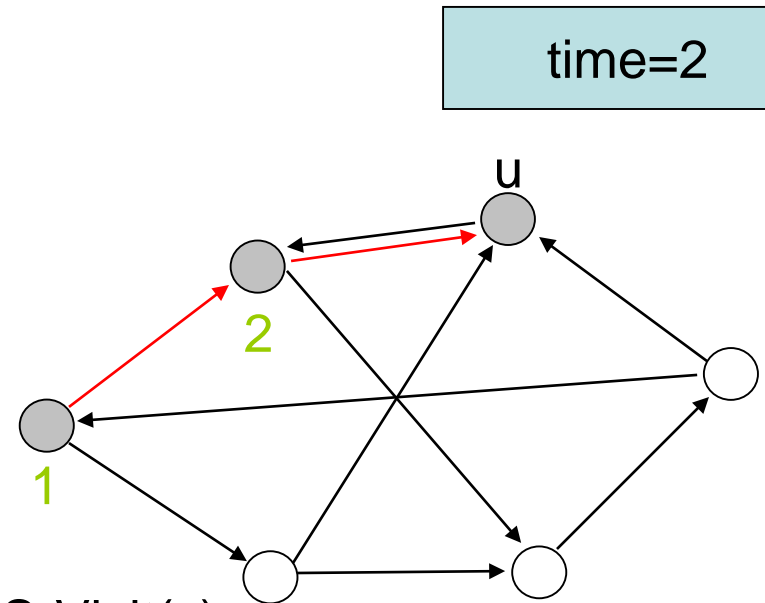
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



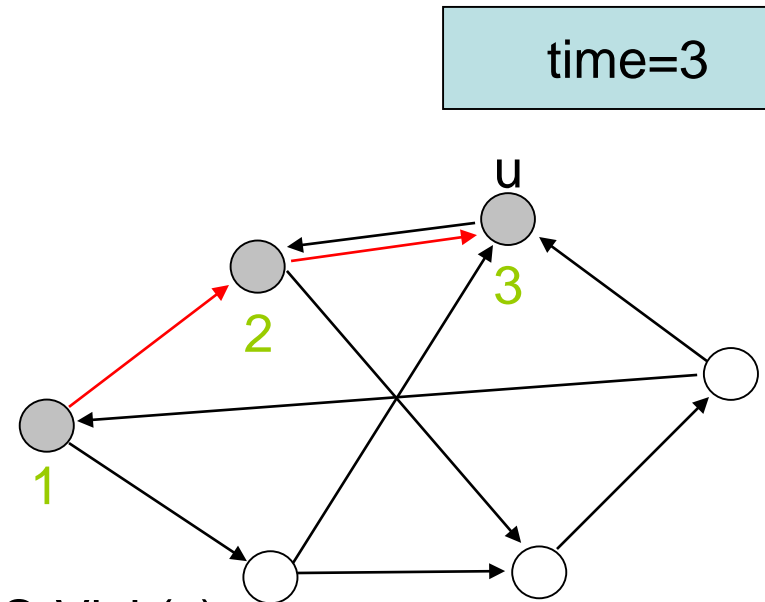
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



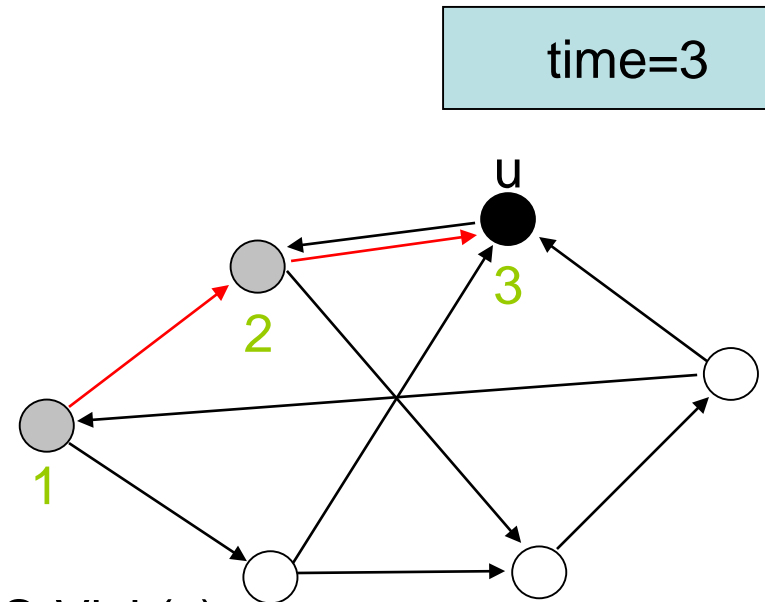
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



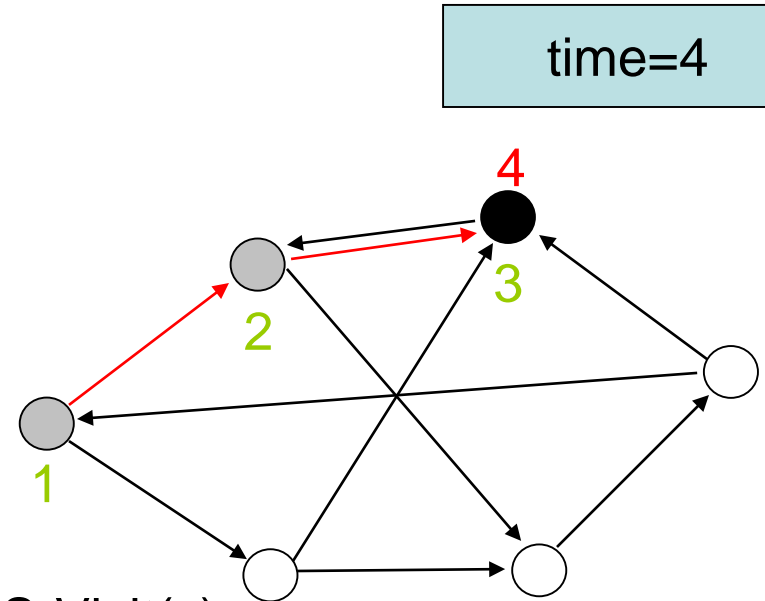
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



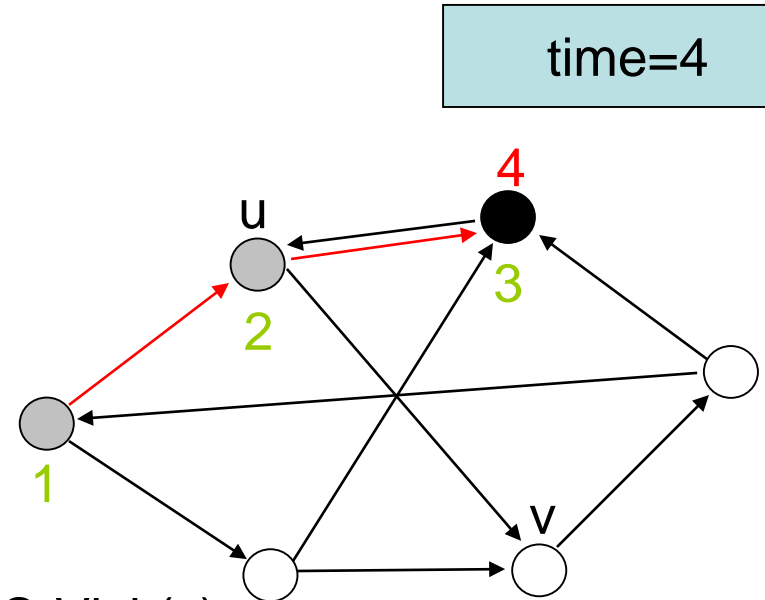
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



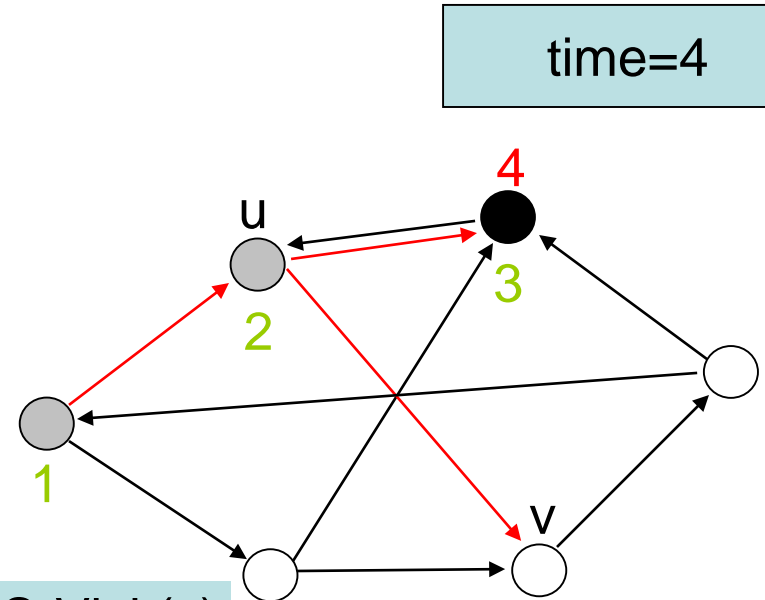
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



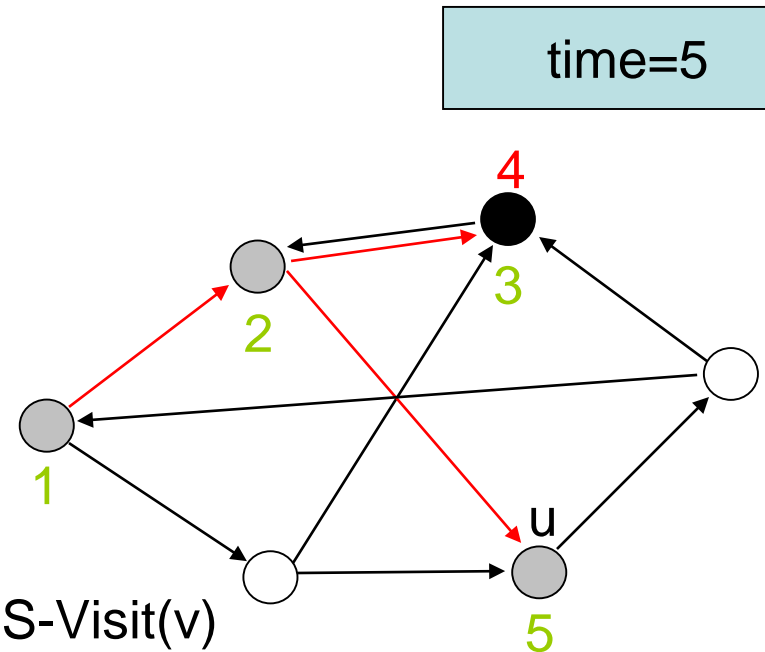
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



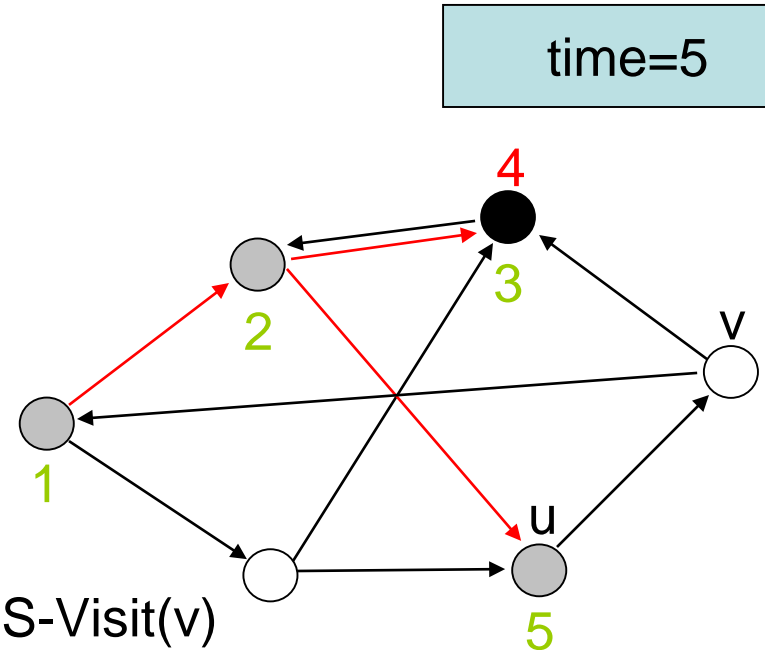
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



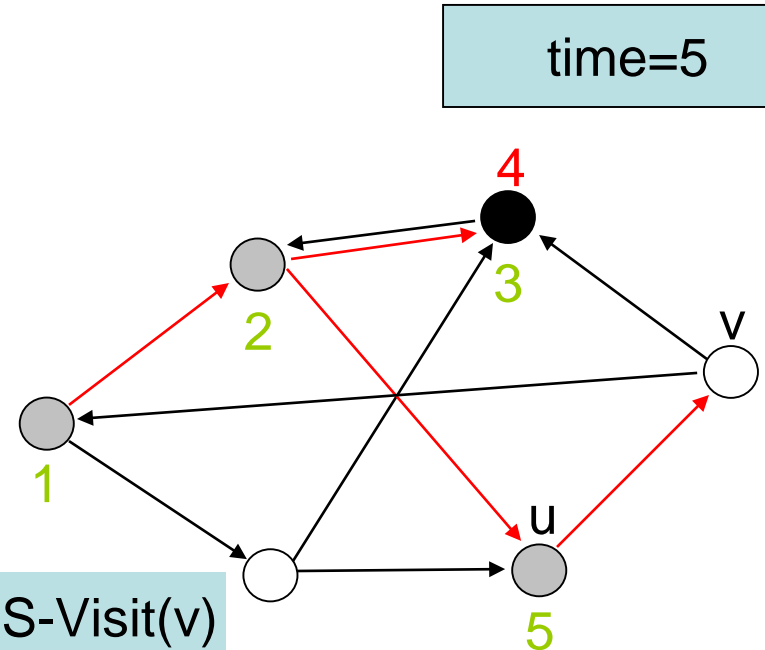
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** color[u] \leftarrow weiß ; $\pi[u] \leftarrow$ nil
2. time \leftarrow 0
3. **for each** vertex $u \in V$ **do**
4. **if** color[u]=weiß **then** DFS-Visit(u)

DFS-Visit(u)

1. color[u] \leftarrow grau
2. time \leftarrow time +1; d[u] \leftarrow time
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** color[v] = weiß **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. color[u] \leftarrow schwarz
7. time \leftarrow time+1; f[u] \leftarrow time



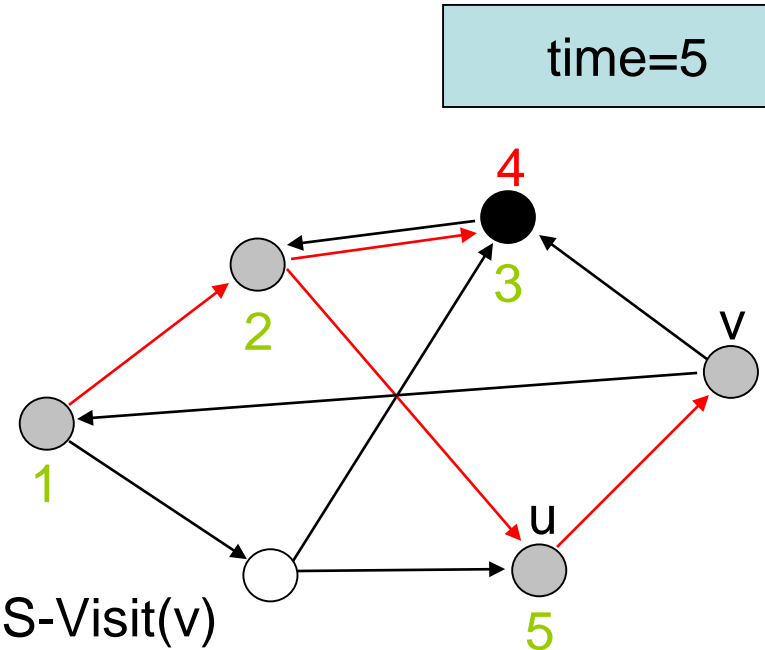
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



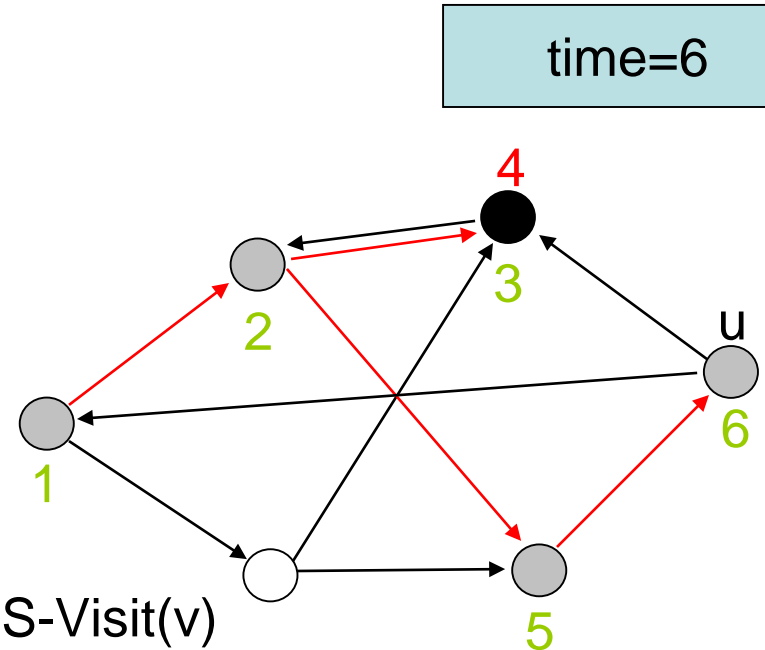
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



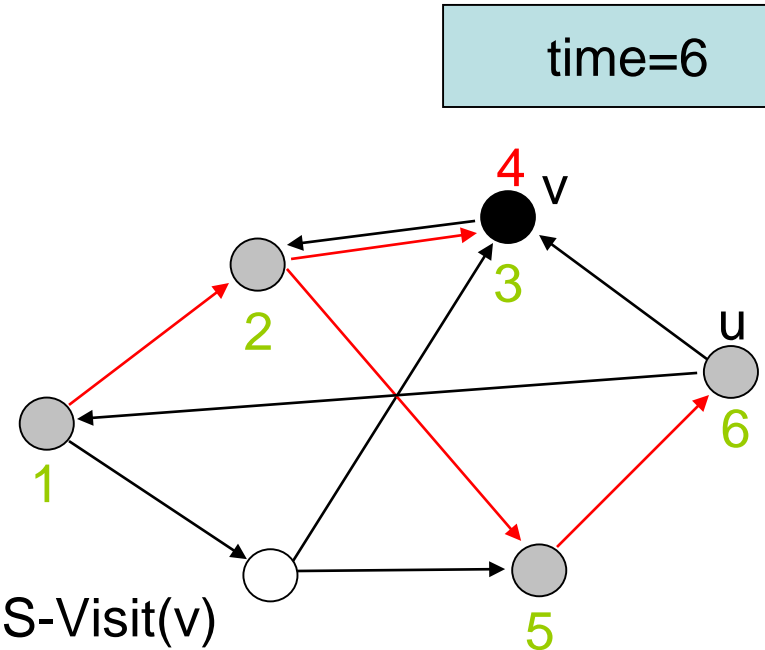
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



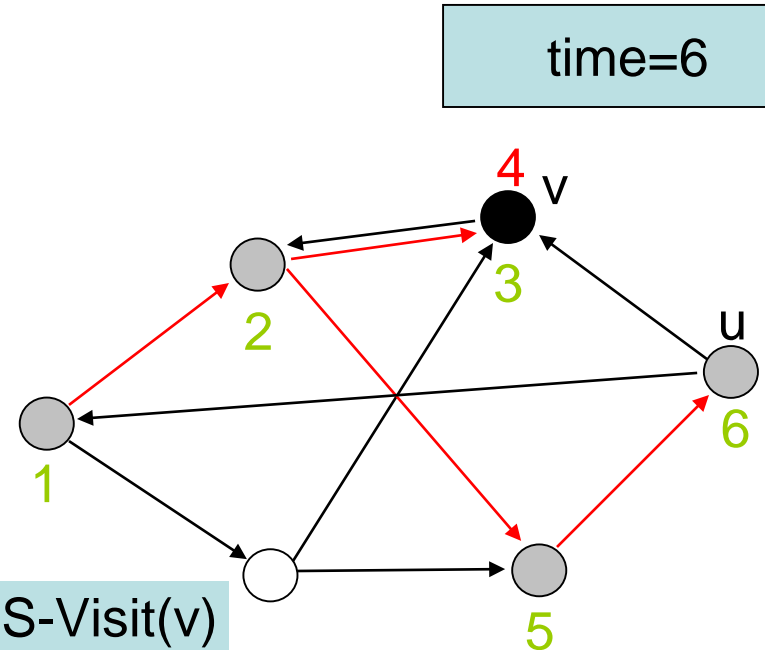
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



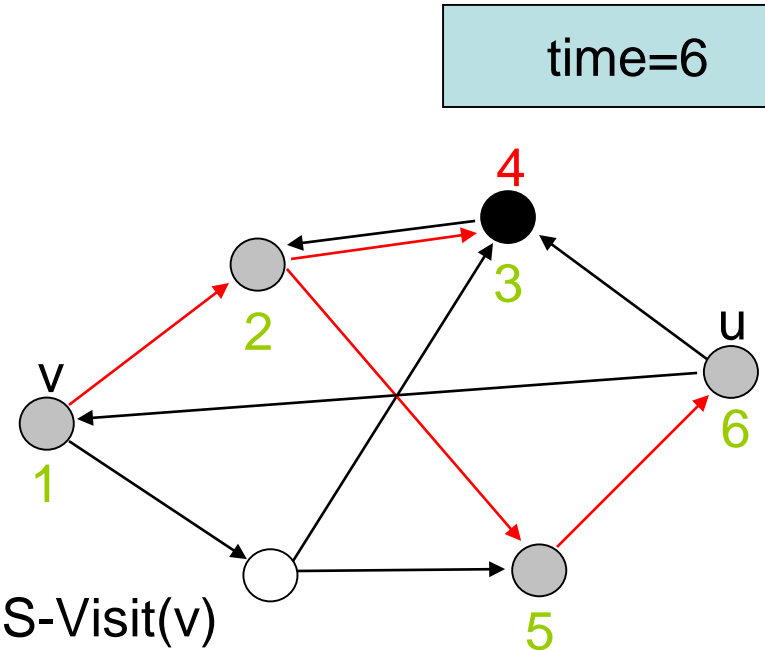
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** color[u] \leftarrow weiß ; $\pi[u] \leftarrow$ nil
2. time \leftarrow 0
3. **for each** vertex $u \in V$ **do**
4. **if** color[u]=weiß **then** DFS-Visit(u)

DFS-Visit(u)

1. color[u] \leftarrow grau
2. time \leftarrow time +1; d[u] \leftarrow time
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** color[v] = weiß **then** $\pi[v] \leftarrow$ u ; DFS-Visit(v)
6. color[u] \leftarrow schwarz
7. time \leftarrow time+1; f[u] \leftarrow time



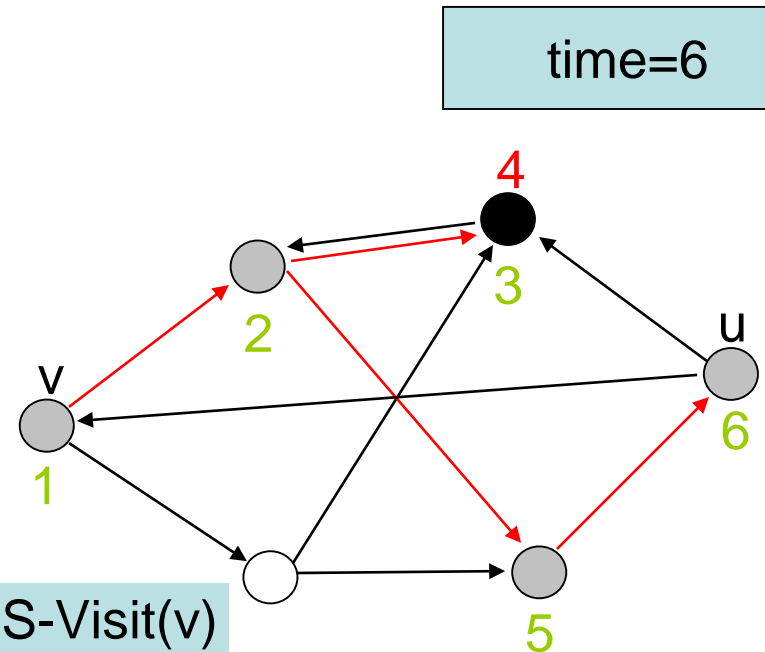
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



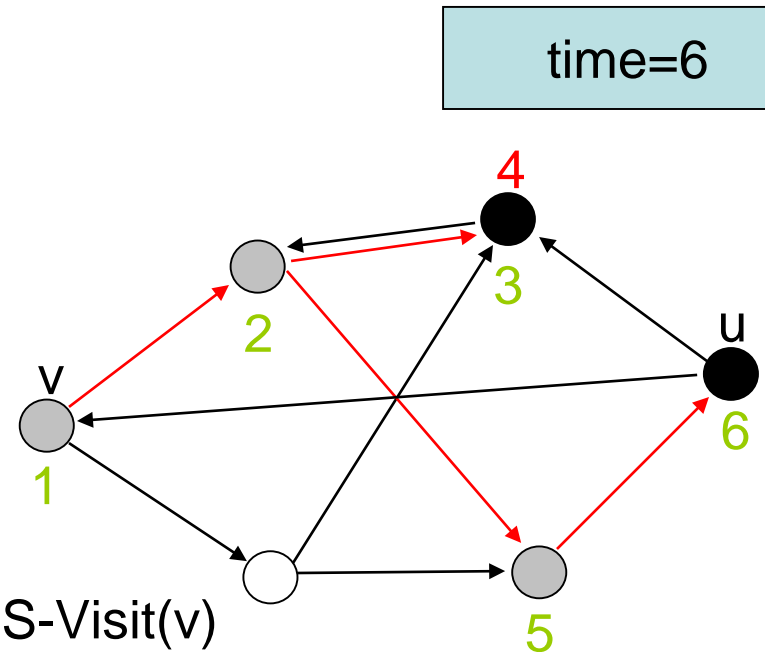
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



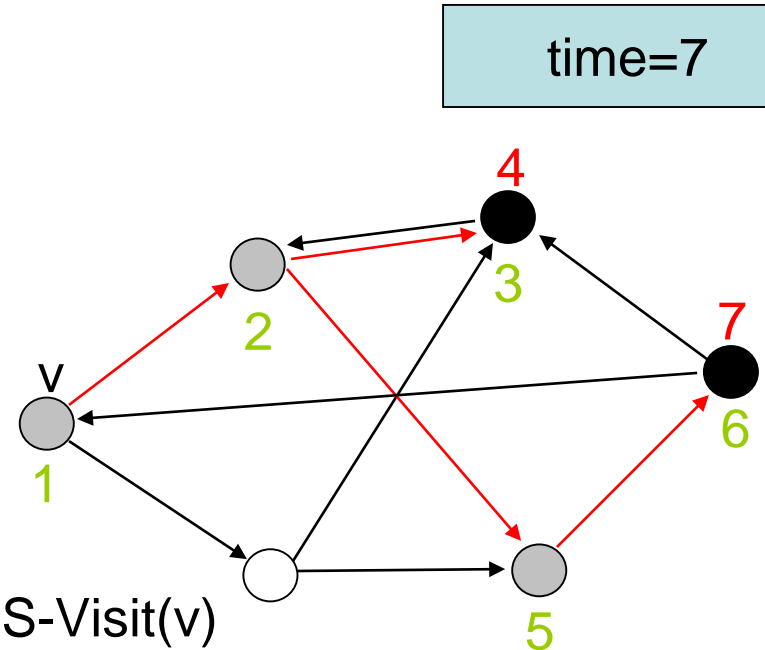
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. **if** $\text{color}[v] = \text{schwarz}$ **then** $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



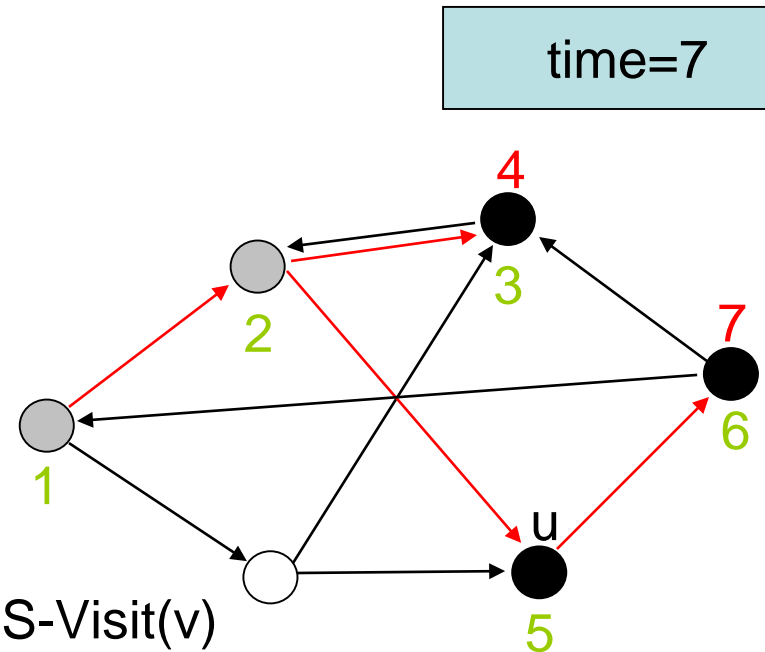
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



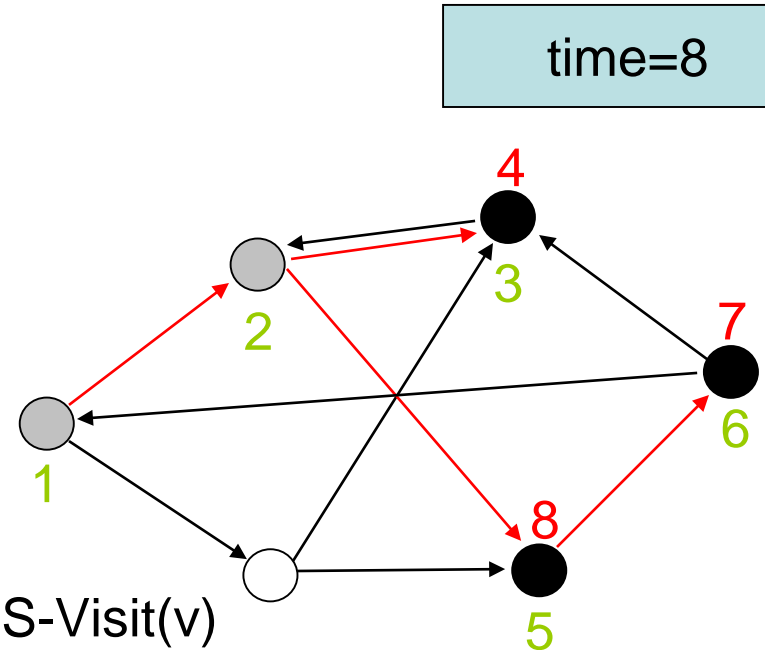
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



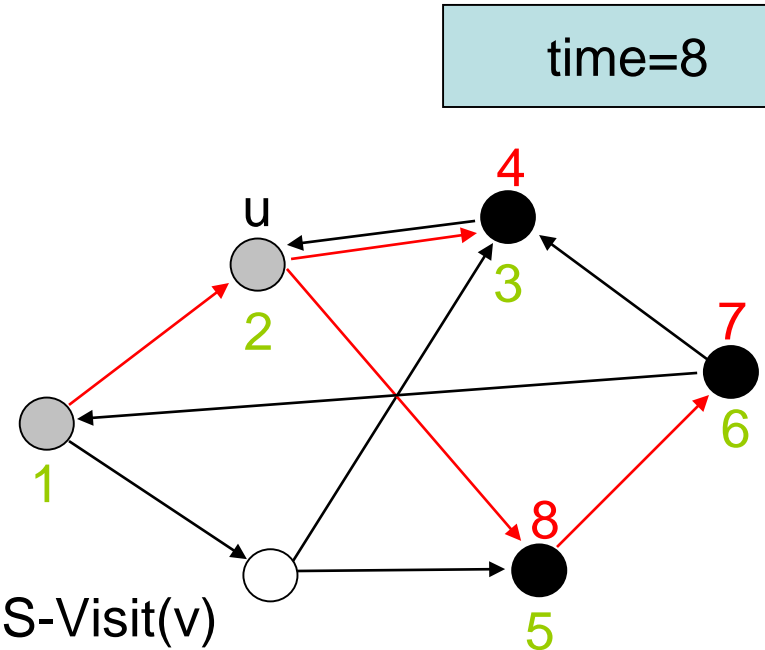
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



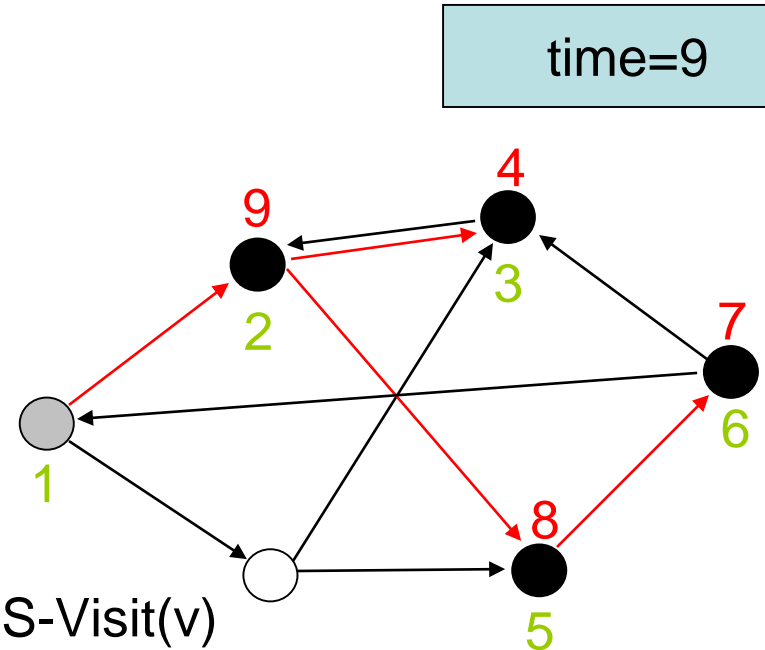
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



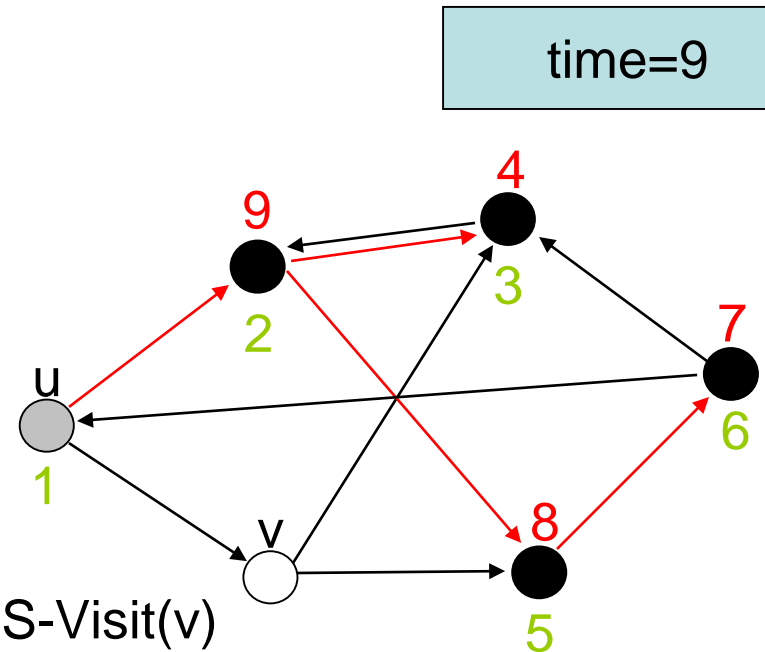
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** color[u] \leftarrow weiß ; $\pi[u] \leftarrow$ nil
2. time \leftarrow 0
3. **for each** vertex $u \in V$ **do**
4. **if** color[u]=weiß **then** DFS-Visit(u)

DFS-Visit(u)

1. color[u] \leftarrow grau
2. time \leftarrow time +1; d[u] \leftarrow time
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** color[v] = weiß **then** $\pi[v] \leftarrow$ u ; DFS-Visit(v)
6. color[u] \leftarrow schwarz
7. time \leftarrow time+1; f[u] \leftarrow time



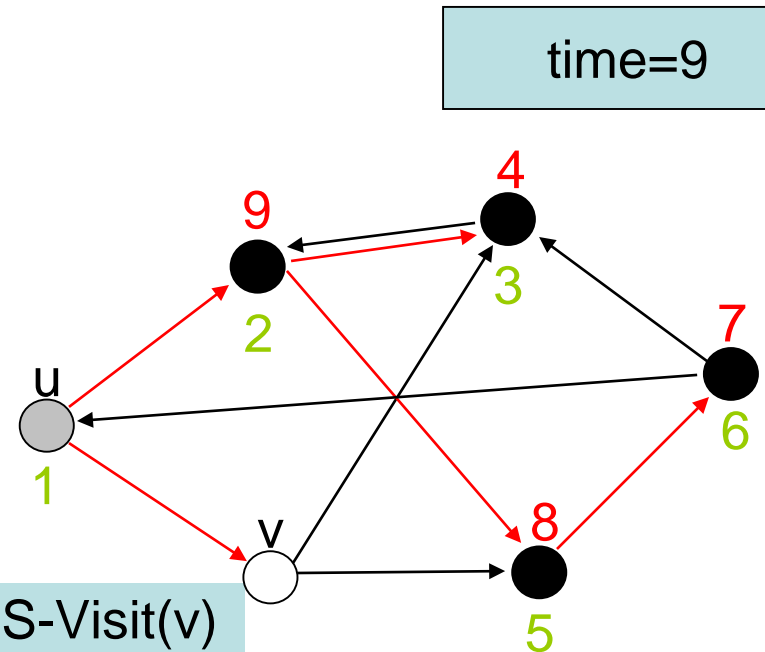
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



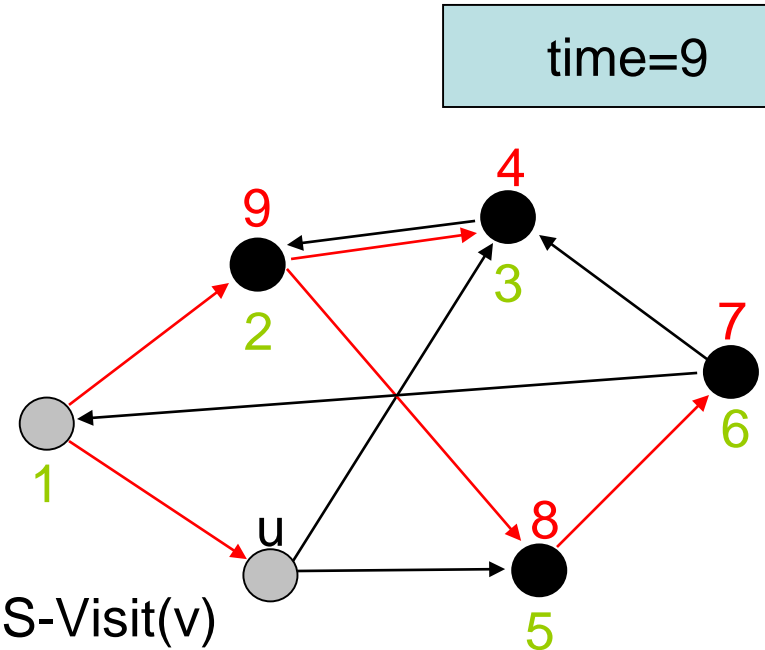
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



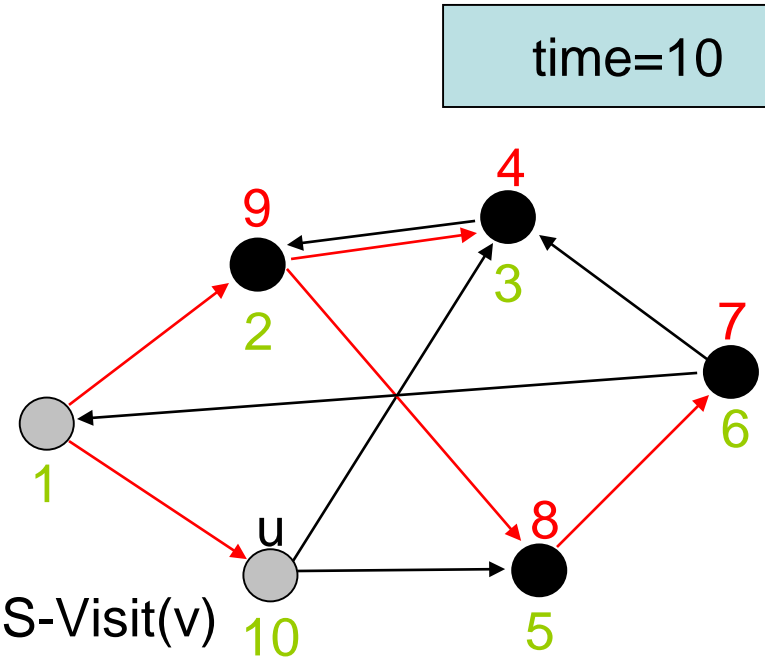
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



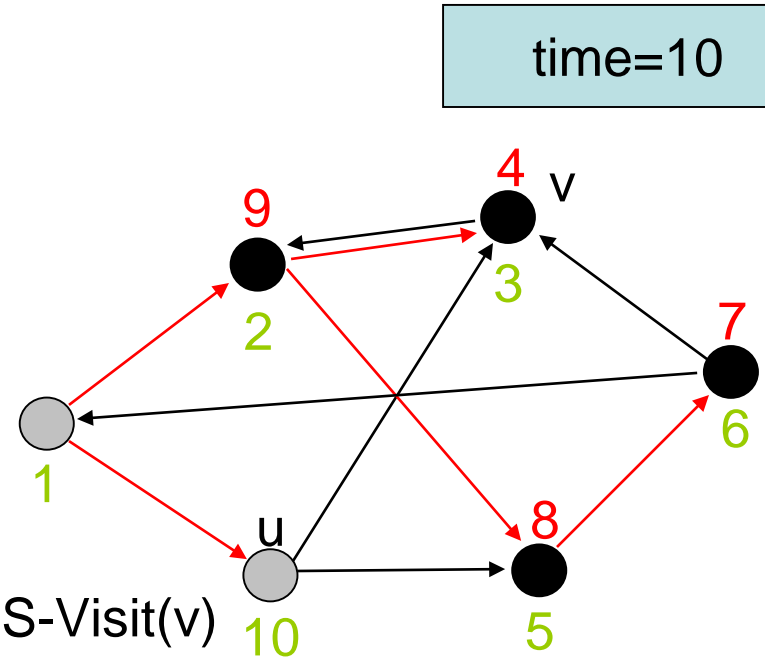
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



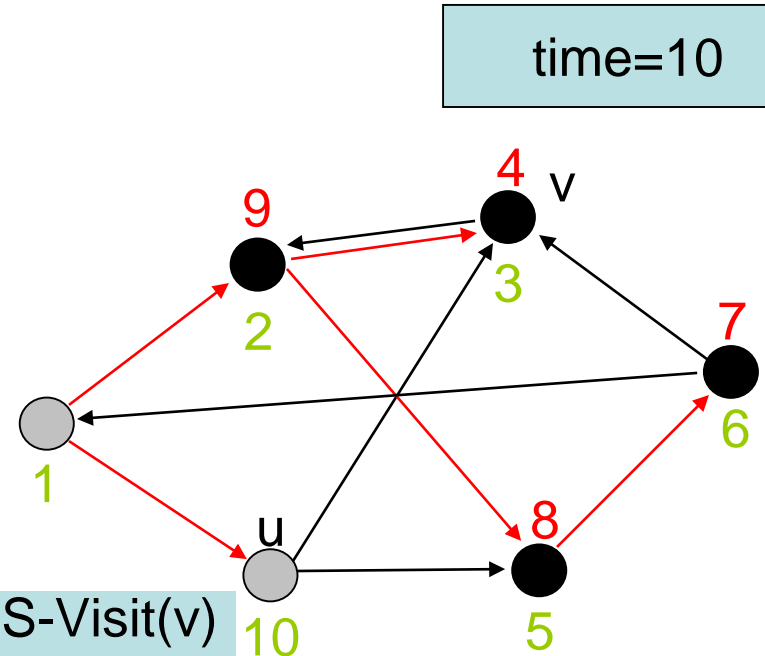
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



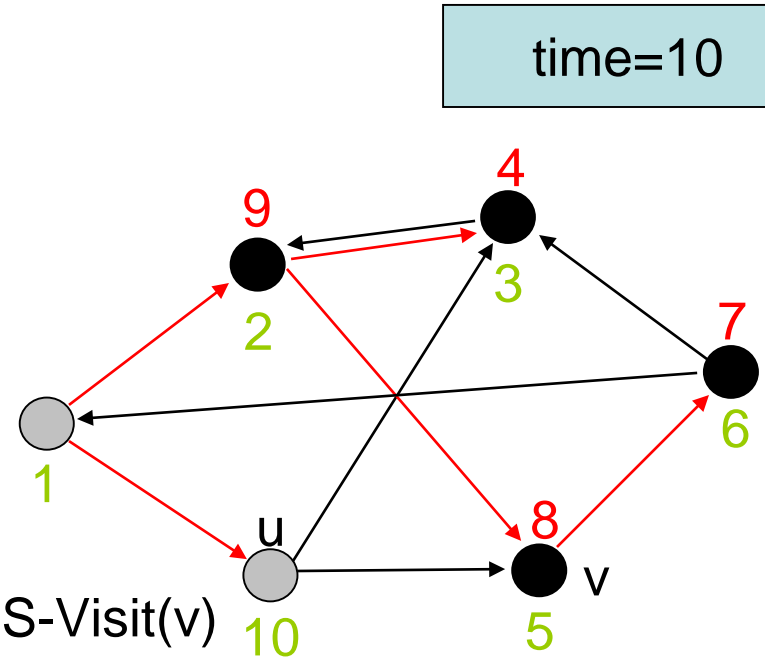
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



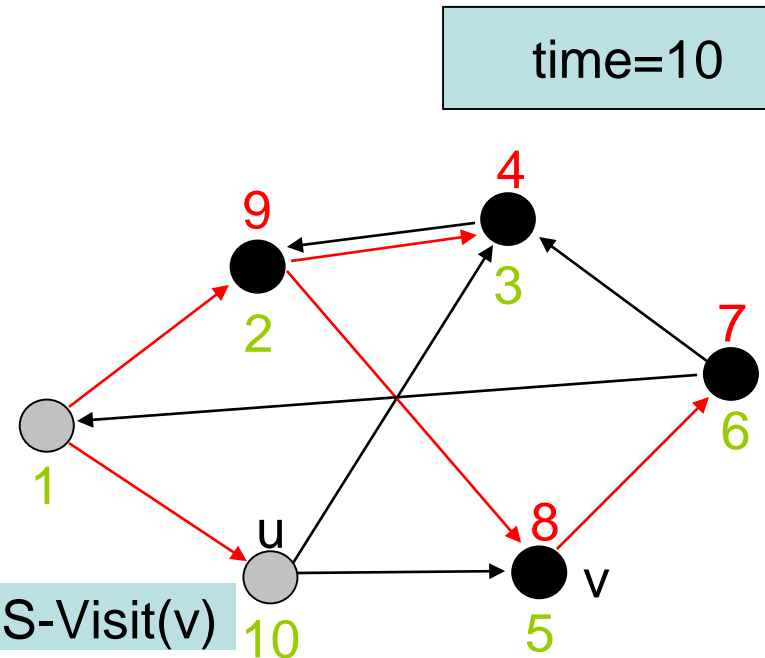
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



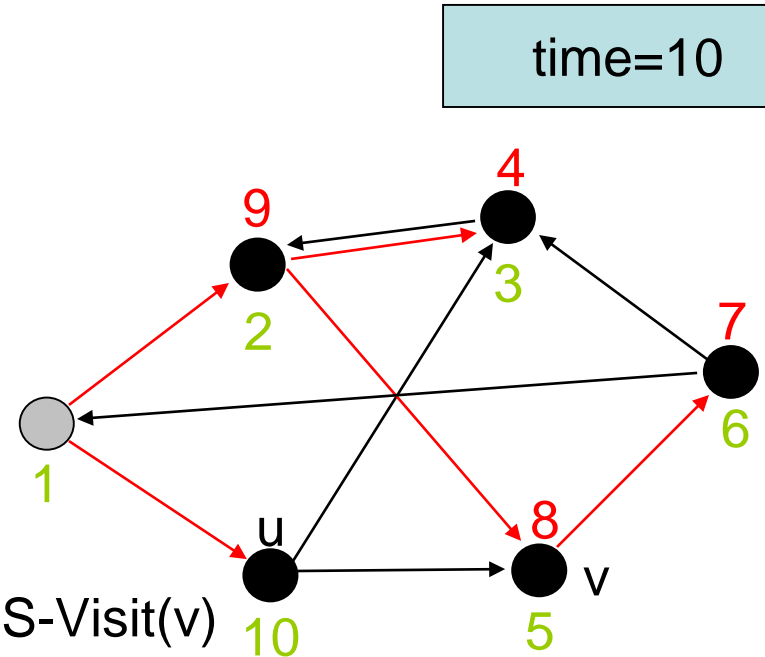
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



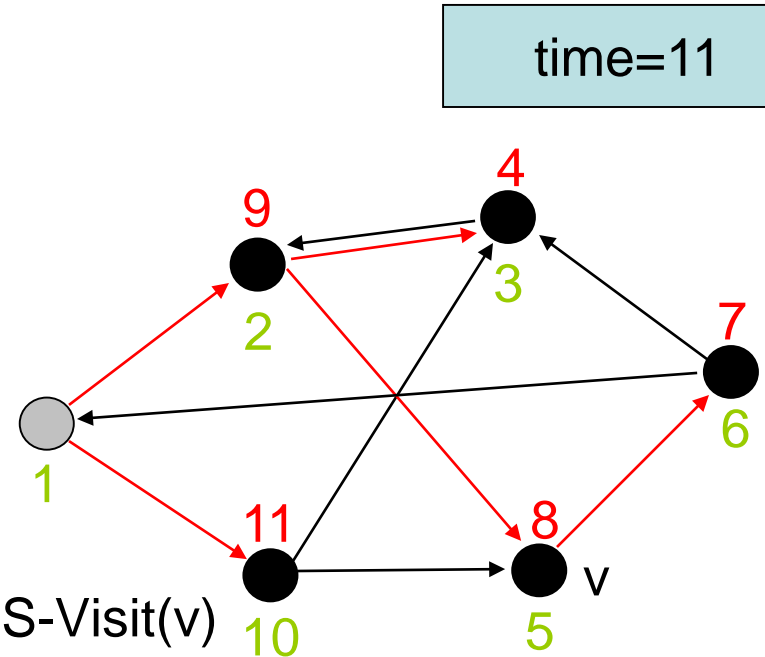
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
4. **for each** $v \in \text{Adj}[u]$ **do**
5. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
6. $\text{color}[u] \leftarrow \text{schwarz}$
7. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



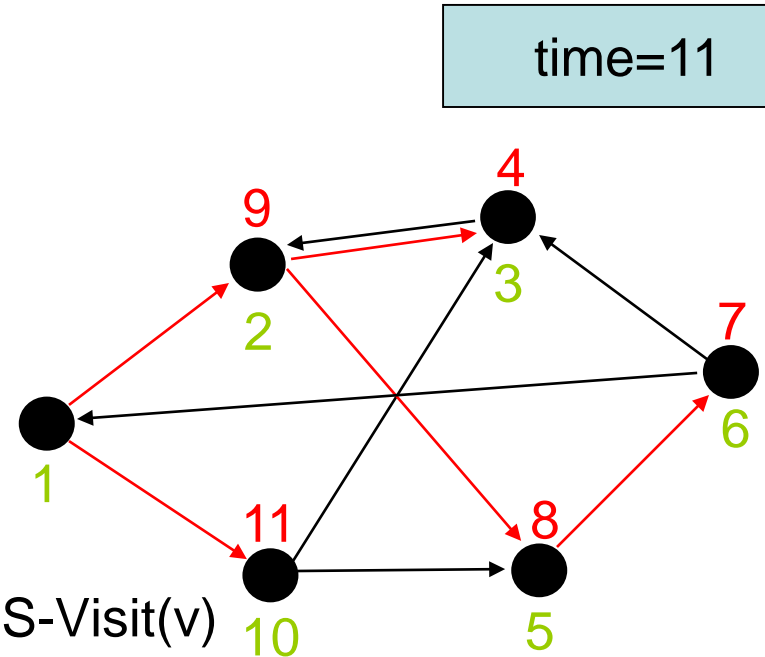
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



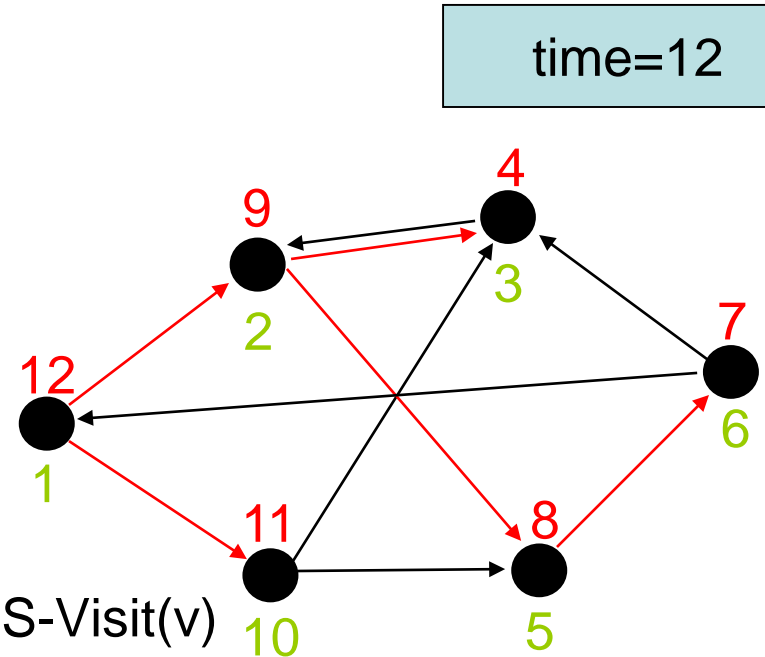
Tiefensuche

DFS(G)

1. **for each** vertex $u \in V$ **do** $\text{color}[u] \leftarrow \text{weiß}$; $\pi[u] \leftarrow \text{nil}$
2. $\text{time} \leftarrow 0$
3. **for each** vertex $u \in V$ **do**
4. **if** $\text{color}[u] = \text{weiß}$ **then** DFS-Visit(u)

DFS-Visit(u)

1. $\text{color}[u] \leftarrow \text{grau}$
2. $\text{time} \leftarrow \text{time} + 1$; $d[u] \leftarrow \text{time}$
3. **for each** $v \in \text{Adj}[u]$ **do**
4. **if** $\text{color}[v] = \text{weiß}$ **then** $\pi[v] \leftarrow u$; DFS-Visit(v)
5. $\text{color}[u] \leftarrow \text{schwarz}$
6. $\text{time} \leftarrow \text{time} + 1$; $f[u] \leftarrow \text{time}$



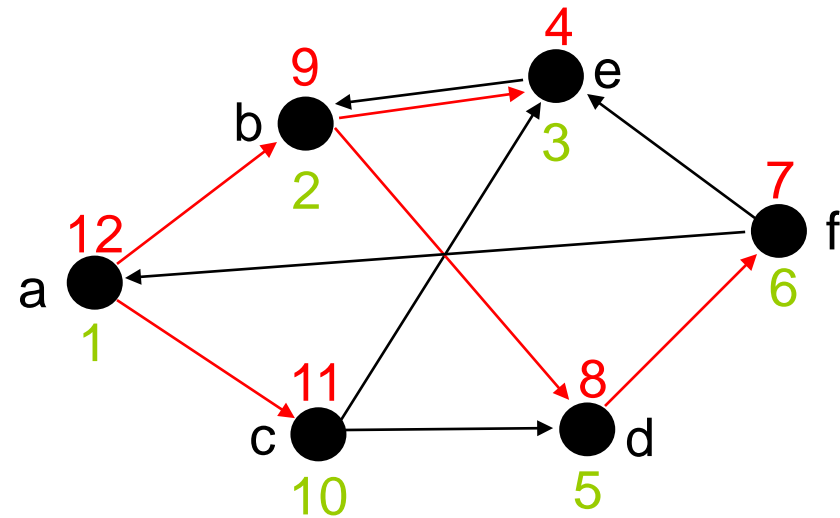
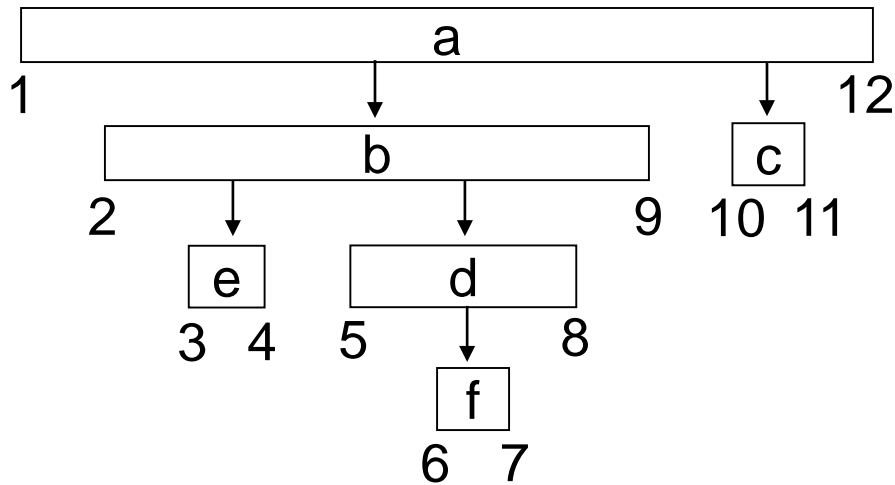
Satz 14.8 (Klammersatz zur Tiefensuche):

In jeder Tiefensuche eines gerichteten oder ungerichteten Graphen gilt für jeden Knoten u und v genau eine der folgenden drei Bedingungen:

- Die Intervalle $[d[u], f[u]]$ und $[d[v], f[v]]$ sind vollständig disjunkt
- Intervall $[d[u], f[u]]$ ist vollständig im Intervall $[d[v], f[v]]$ enthalten und u ist Nachfolger von v im DFS-Baum
- Intervall $[d[v], f[v]]$ ist vollständig im Intervall $[d[u], f[u]]$ enthalten und v ist Nachfolger von u im DFS-Baum

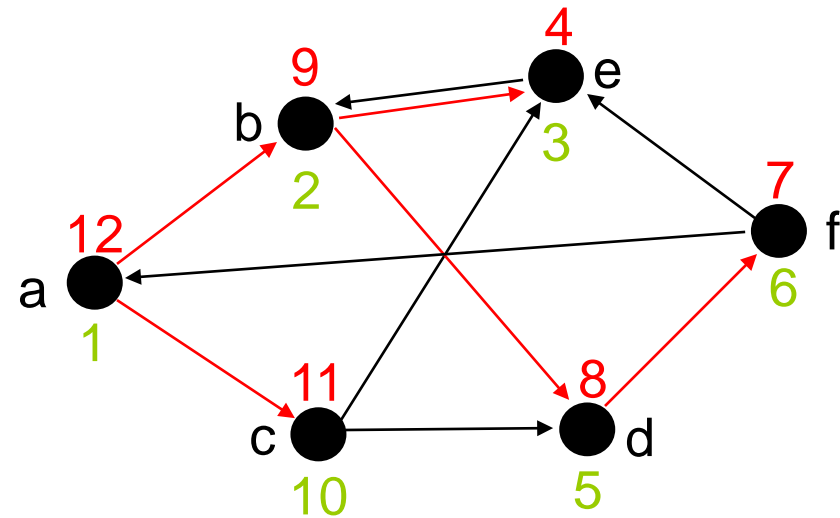
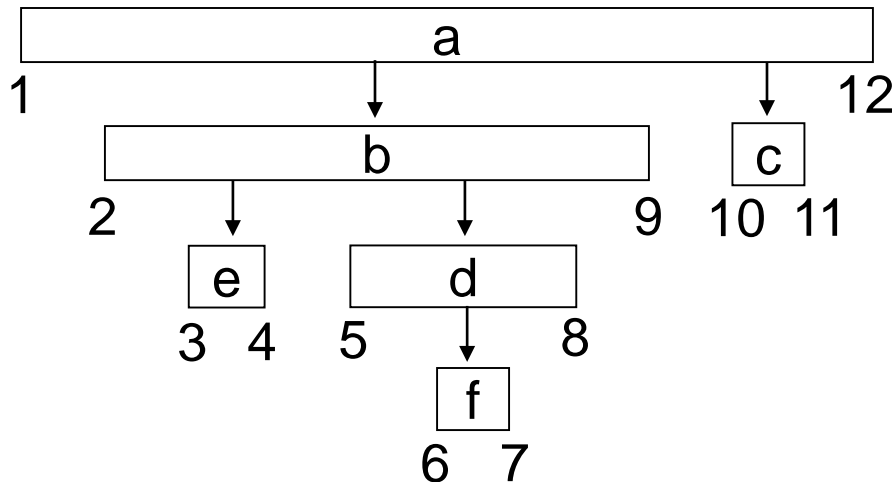
Tiefensuche

Beispiel:



Tiefensuche

Beispiel:



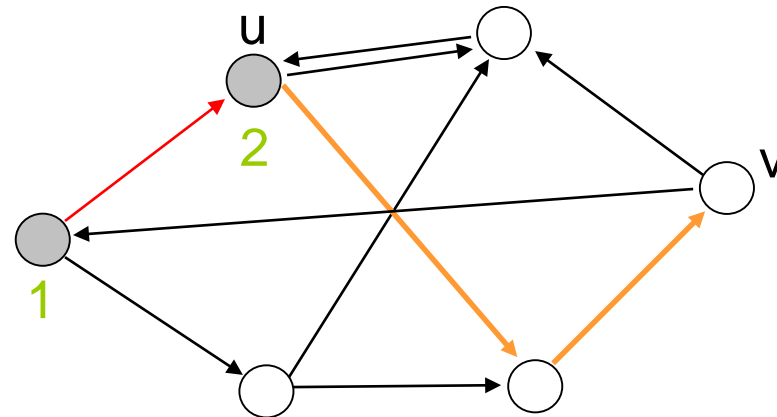
Korollar 14.9:

Knoten v ist echter Nachfolger von Knoten u im DFS-Baum von G , gdw. $d[u] < d[v] < f[v] < f[u]$.

Tiefensuche

Satz 14.10 (Satz vom weißen Weg)

In einem DFS-Wald eines gerichteten oder ungerichteten Graph G ist Knoten v ein Nachfolger von Knoten u , gdw. zum Zeitpunkt $d[u]$ v über einen Pfad weißer Knoten erreicht werden kann.



Tiefensuche - Laufzeitanalyse

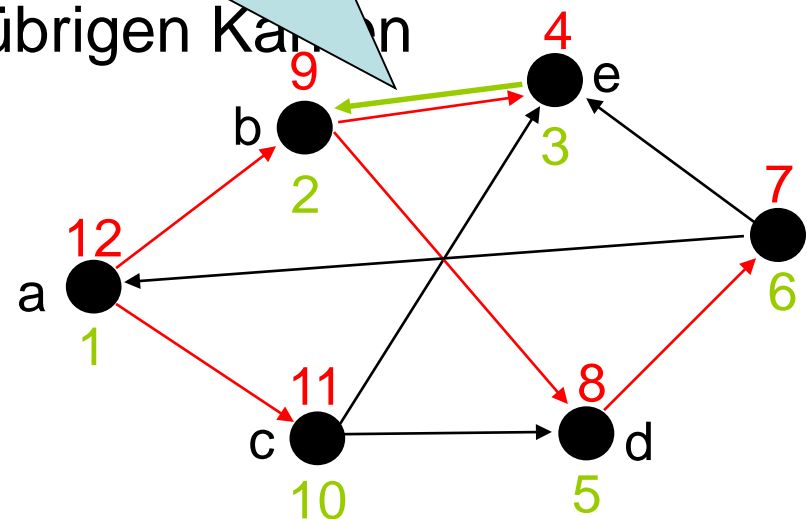
Satz 14.11: Bei Eingabe von Graph $G=(V,E)$ besitzt Algorithmus DFS Laufzeit $O(|V| + |E|)$.

Analyse: Wie bei Breitensuche. Nutzen aus dass Gesamtgröße aller Adjazenzlisten $O(|E|)$

Tiefensuche

Klassifikation von Kanten:

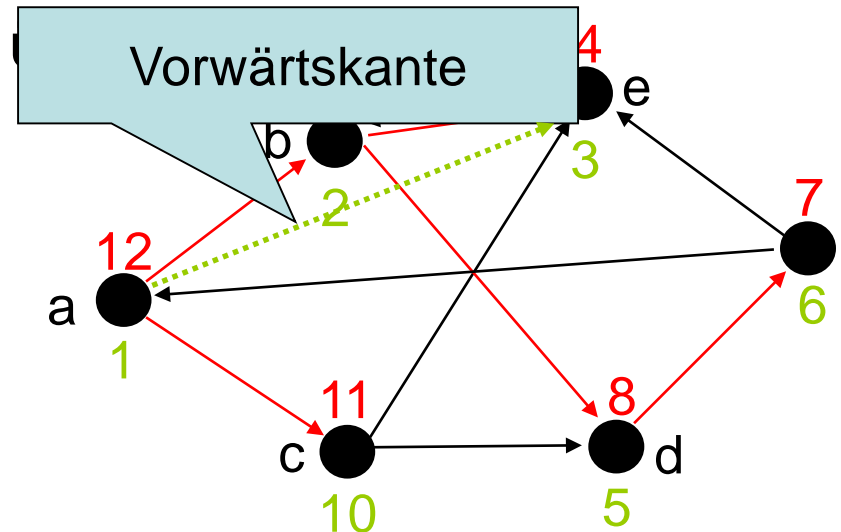
- **Baumkanten** sind Kanten des DFS-Walds G
- **Rückwärtskanten** sind Kanten (u,v) , die Knoten u mit Ahnen von u im DFS-Baum verbinden
- **Vorwärtskanten** sind die nicht-Baum Kanten (u,v) die u mit einem Nachfolger v in einem DFS-Baum verbinden
- **Kreuzungskanten** sind alle übrigen Kanten



Tiefensuche

Klassifikation von Kanten:

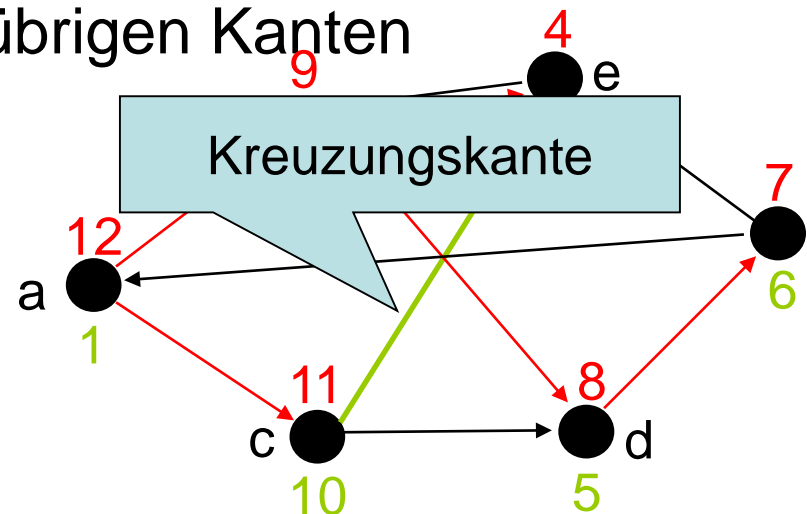
- **Baumkanten** sind Kanten des DFS-Walds G
- **Rückwärtskanten** sind Kanten (u,v) , die Knoten u mit Ahnen von u im DFS-Baum verbinden
- **Vorwärtskanten** sind die nicht-Baum Kanten (u,v) , die u mit einem Nachfolger v in einem DFS-Baum verbinden
- **Kreuzungskanten** sind alle



Tiefensuche

Klassifikation von Kanten:

- **Baumkanten** sind Kanten des DFS-Walds G
- **Rückwärtskanten** sind Kanten (u,v) , die Knoten u mit Ahnen von u im DFS-Baum verbinden
- **Vorwärtskanten** sind die nicht-Baum Kanten (u,v) , die u mit einem Nachfolger v in einem DFS-Baum verbinden
- **Kreuzungskanten** sind alle übrigen Kanten



Tiefensuche

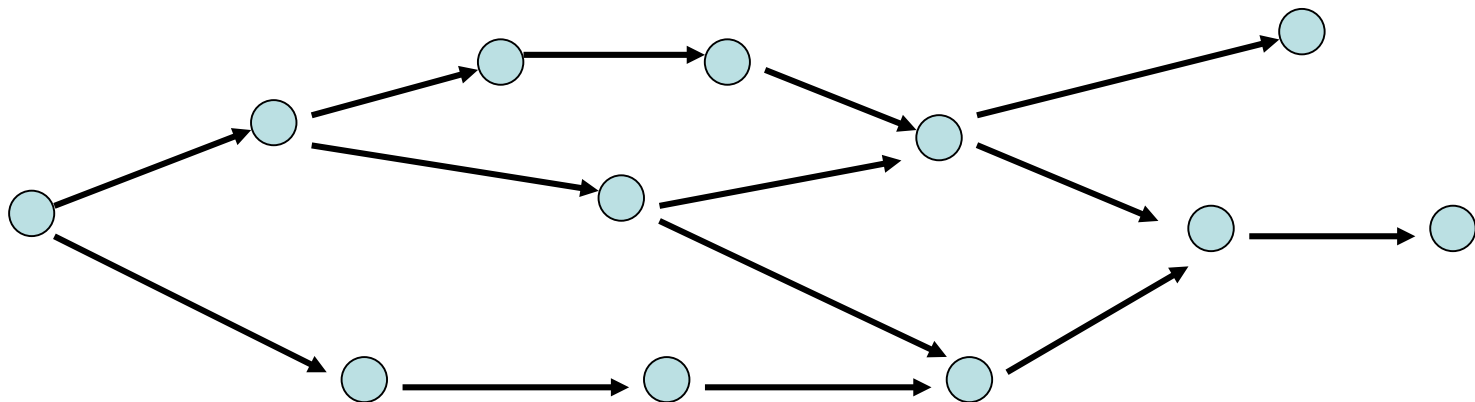
Beobachtung für Kante (v,w) :

Kantentyp	$d[v] < d[w]$	$f[v] > f[w]$
Baum & Vorwärts	Ja	Ja
Rückwärts	Nein	Nein
Kreuz	Nein	Ja

Tiefensuche

Anwendung:

- Erkennung eines azyklischen gerichteten Graphen (engl. DAG)



Merkmale: keine gerichtete Kreise

Tiefensuche

Lemma 14.12: Das Folgende ist äquivalent:

1. G ist ein DAG
2. DFS enthält keine Rückwärtskante
3. $\forall (v,w) \in E: f[v] > f[w]$

Beweis:

2 \Rightarrow 3: folgt aus Tabelle

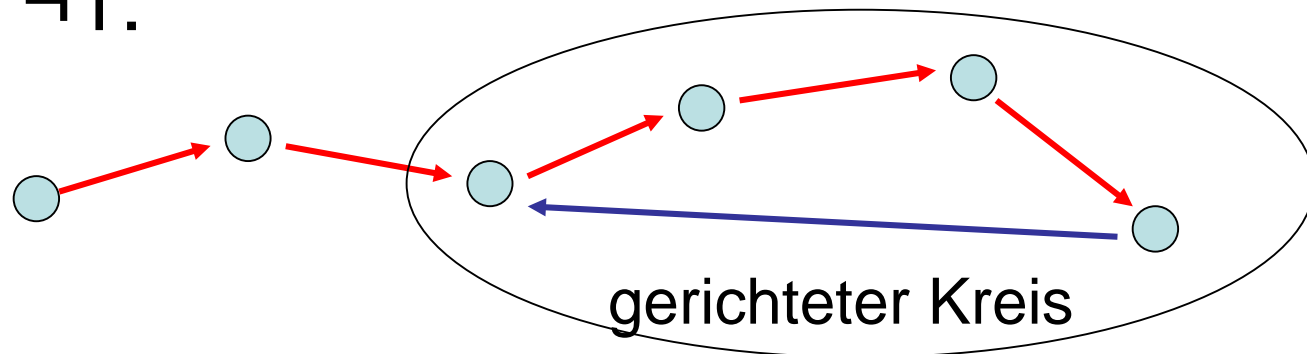
Tiefensuche

Lemma 14.12: Das Folgende ist äquivalent:

1. G ist ein DAG
2. DFS enthält keine Rückwärtskante
3. $\forall (v,w) \in E: f[v] > f[w]$

Beweis:

$\neg 2 \Rightarrow \neg 1$:



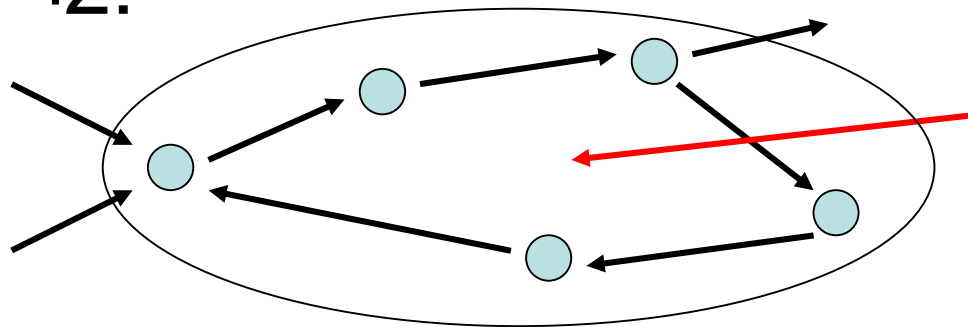
Tiefensuche

Lemma 14.12: Das Folgende ist äquivalent:

1. G ist ein DAG
2. DFS enthält keine Rückwärtskante
3. $\forall (v,w) \in E: f[v] > f[w]$

Beweis:

$\neg 1 \Rightarrow \neg 2$:



Eine davon
Rückwärtskante

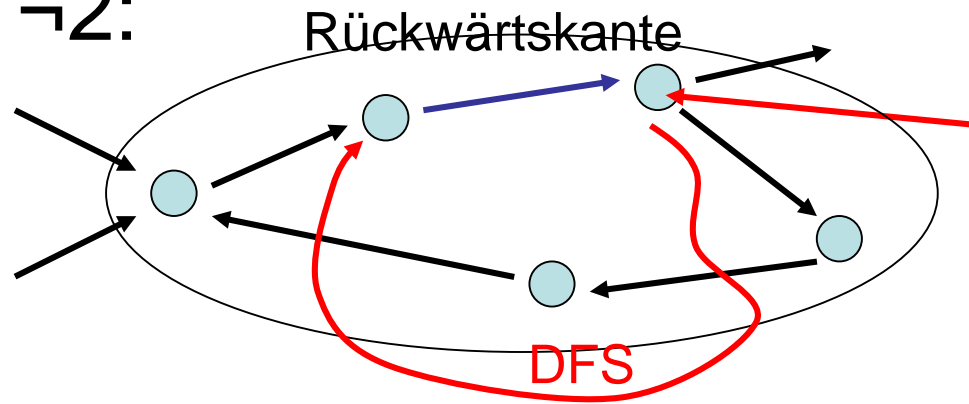
Tiefensuche

Lemma 14.12: Das Folgende ist äquivalent:

1. G ist ein DAG
2. DFS enthält keine Rückwärtskante
3. $\forall (v,w) \in E: f[v] > f[w]$

Beweis:

$\neg 1 \Rightarrow \neg 2$:



Annahme: Erster von DFS besuchter Knoten im Kreis

Zusammenfassung

- **Breitensuche:** Zunächst alle Nachbarn abarbeiten; liefert kürzeste Wege in (ungewichteten) Graphen. Laufzeit $O(|V|+|E|)$.
- **Tiefensuche:** Zunächst in die „Tiefe“ gehen; Durchläuft alle Knoten eines Graphen in Zeit $O(|V|+|E|)$.