Kapitel 2

Starke Zusammenhangskomponenten in Graphen

Effiziente Algorithmen, SS 2018

Professor Dr. Petra Mutzel

VO 5 vom 24. April 2018

Einführung

Grundlegende Definitionen: siehe ppt-Folien

- Graph (gerichtet und ungerichtet)
- einfache Graphen
- Knotengrade und Nachbarmengen
- Kantenzüge, Pfade und Wege
- Kreise
- Datenstrukturen für Graphen: Adjazenzlisten, Matrizen

Tiefensuche in Graphen

zum Aufwärmen und Einsteigen etwas Wiederholung von DAP 2

Tiefensuche (ungerichtet) (auch: Depth First Search, DFS)

Erinnerung
- zentraler Algorithmus
- ziemlich einfach
- extrem nützlich
- sehr effizient (Linearzeit)

jetzt: gerichtete Tiefensuche

Tiefensuche in gerichteten Graphen

Kantenklassifikation

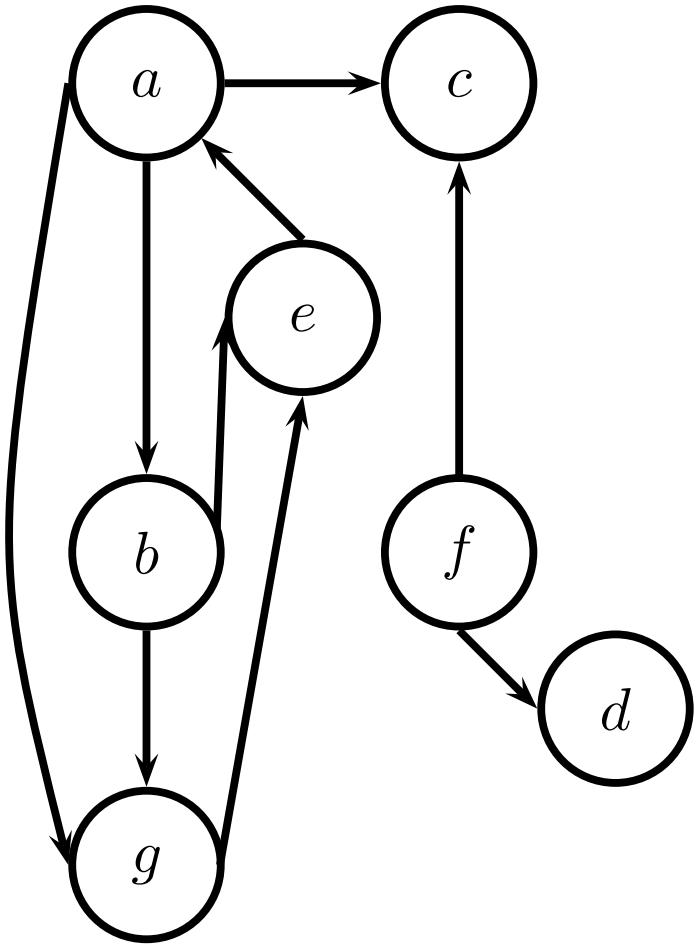
- Baum-Kanten (T): Kanten, denen die Tiefensuche folgt
- Rückwärtskanten (B): Kanten von einem DFS-Knoten v zu einem Vorgänger von v im DFS-Baum
- Vorwärtskanten (F): Kanten von v zu einem bereits durch T-Kanten erreichten Nachfolgerknoten w
- Querkanten (C): alle anderen Kanten

Tiefensuche in gerichteten Graphen: Beispiel

v	num(v)	status(v)
a		
b		
c		
d		
e		
f		
g		

num(v): Reihenfolge im DFS

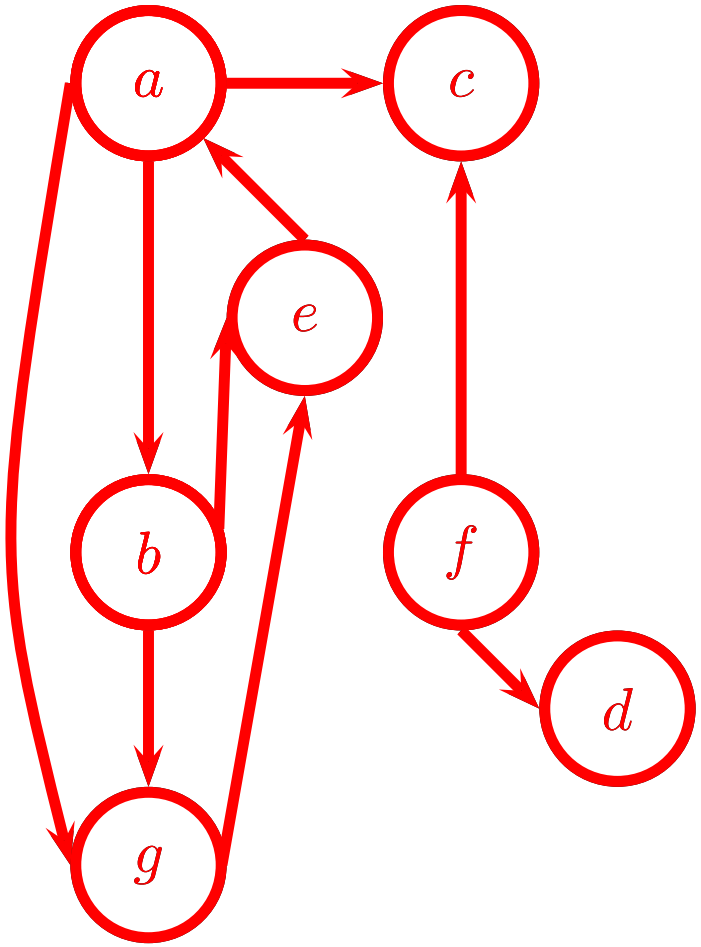
status(v): 0 (noch nicht besucht)
1 (besucht, nicht fertig)
2 (fertig)



Tiefensuche in gerichteten Graphen: Beispiel

v	num(v)	status(v)
a	1	2
b	2	2
c	5	2
d	6	2
e	3	2
f	7	2
g	4	2

$$
\begin{aligned}
T &= (a,b),\ (b,e),\ (b,g),\ (a,c) \\
F &= (a,g) \\
B &= (e,a) \\
C &= (g,e),\ (f,c),\ (f,d)
\end{aligned}
$$



Algorithmus 2.1 (Gerichtete Tiefensuche (DFS))

1. **Initialisierung**
   $i := 0$; $T := \emptyset$; $B := \emptyset$; $C := \emptyset$; $F := \emptyset$
   Für alle $v \in V$: dfsnum$(v) := 0$; status$(v) := 0$
2. Für alle $v \in V$: If dfsnum$(v) = 0$ Then DFS-visit(v)

DFS-visit(v)

1. status$(v) := 1$; $i := i + 1$; dfsnum$(v) := i$
2. Für alle $w \in$ Adj(v)
   If dfsnum$(w) = 0$
   Then $T := T \cup \{(v, w)\}$; DFS-visit(w)
   Else If dfsnum$(w) >$ dfsnum(v)
   Then $F := F \cup \{(v, w)\}$
   Else If status$(w) = 1$
   Then $B := B \cup \{(v, w)\}$
   Else $C := C \cup \{(v, w)\}$
3. status$(v) := 2$

Analyse der Tiefensuche

Lemma

Der Algorithmus 2.1 zur gerichteten Tiefensuche klassifiziert die Kanten korrekt in Baumkanten, Rückwärtskanten, Vorwärtskanten und Querkanten. Er läuft in Zeit $O(|V| + |E|)$.

Tiefensuche — Na, und?

Wen interessiert denn das?

Jemanden, der

- wissen will, ob ein Graph zusammenhängend ist.
- wissen will, ob ein Graph kreisfrei ist.
- wissen will, ob ein Graph stark zusammenhängend ist.

Stark zusammenhängend?

neuer Begriff starker Zusammenhang

Starker Zusammenhang

Definition 2.2 (starker Zusammenhang)

Sei $G = (V, E)$ ein gerichteter Graph. Gibt es einen gerichteten Weg von v nach w (mit $v, w \in V$), so schreiben wir $v \to w$. Gilt $v \to w$ und $w \to v$, so schreiben wir $v \leftrightarrow w$.
Zwei Knoten $v, w \in V$ heißen stark zusammenhängend, wenn $v \leftrightarrow w$ gilt.
Die Äquivalenzklassen von V bezüglich $\leftrightarrow$ heißen *starke Zusammenhangskomponenten*.

Ist das wohldefiniert? Ist $\leftrightarrow$ eine Äquivalenzrelation?

Äquivalenzrelation

Starker Zusammenhang

Definition 2.2 (starker Zusammenhang)

Sei $G = (V, E)$ ein gerichteter Graph. Gibt es einen gerichteten Weg von v nach w (mit $v, w \in V$), so schreiben wir $v \to w$. Gilt $v \to w$ und $w \to v$, so schreiben wir $v \leftrightarrow w$.
Zwei Knoten $v, w \in V$ heißen stark zusammenhängend, wenn $v \leftrightarrow w$ gilt.
Die Äquivalenzklassen von V bezüglich $\leftrightarrow$ heißen *starke Zusammenhangskomponenten*.

Ist das wohldefiniert? Ist $\leftrightarrow$ eine Äquivalenzrelation?

Äquivalenzrelation

- reflexiv $v \leftrightarrow v$ (Weglänge 0)
- symmetrisch $v \leftrightarrow w \Leftrightarrow w \leftrightarrow v$ (nach Definition)
- transitiv $v \leftrightarrow w$ und $w \leftrightarrow x \Rightarrow v \leftrightarrow x$ (Konkatenation)

Idee des Algorithmus von Kosaraju

- Führe DFS-Traversierung von $G = (V, E)$ durch; vergib dabei in absteigender Reihenfolge sogenannte f-Nummern n, $n-1$, $\ldots$, 1 an die Knoten.
- Knoten $v \in V$ erhält seine f-Nummer beim Abschluss des Aufrufs von DFS-visit(v).
- Hierzu wird DFS-visit(v) in der letzten Zeile abgeändert: dort wird zusätzlich die f-Nummer von v vergeben.
- Ein zweiter DFS-Durchlauf, der den Graphen mit umgekehrten Kanten traversiert und dabei die Knoten $v \in V$ in Reihenfolge aufsteigender f-Nummern durchläuft, legt dann die SHKs fest.

Berechnung der starken Zusammenhangskomponenten

Algorithmus von Kosaraju

1. Führe DFS-Traversierung von $G = (V, E)$ durch, vergib dabei in absteigender Reihenfolge die f-Nummern n, $n-1$, ..., 1 an die Knoten.
2. Berechne $G^* := (V, E^*)$ mit $E^* := \{(w, v) \mid (v, w) \in E\}$.
3. Führe DFS-Traversierung von G^* durch, durchlaufe dabei im Rahmenalgorithmus die Knoten $v \in V$ in Reihenfolge aufsteigender f-Nummern, Start in Knoten v mit $f[v] = 1$.
4. Gib die T-Bäume der zweiten DFS-Traversierung als starke Zusammenhangskomponenten aus.

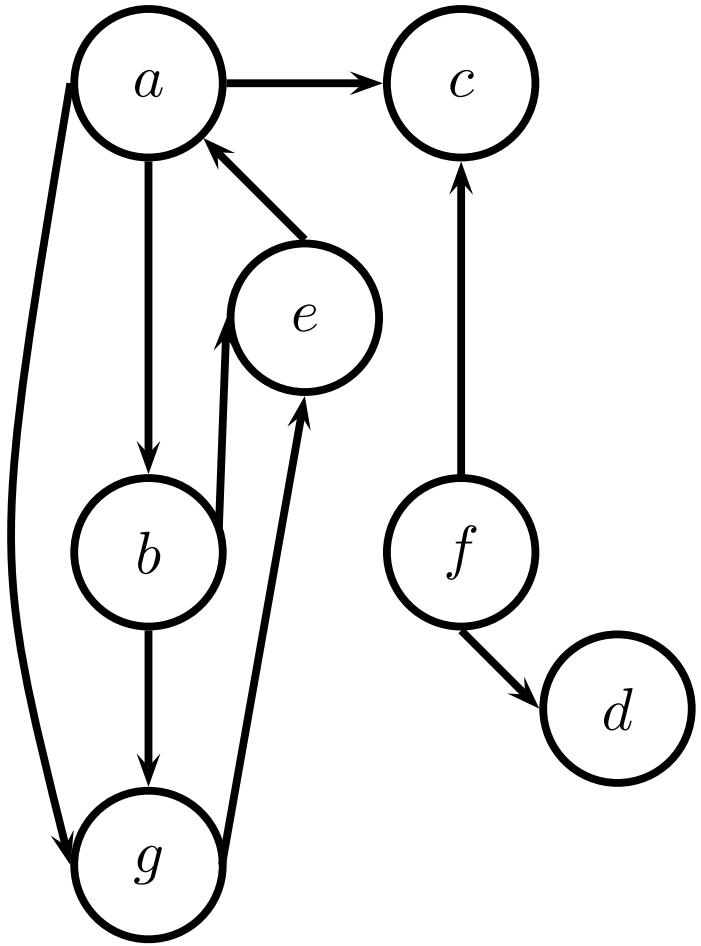
einfach zu implementieren ✓ **Laufzeit** $O(|V| + |E|)$ ✓

Korrektheit? Sind das wirklich die SZHK?

Starke Zusammenhangskomponenten — Beispiel

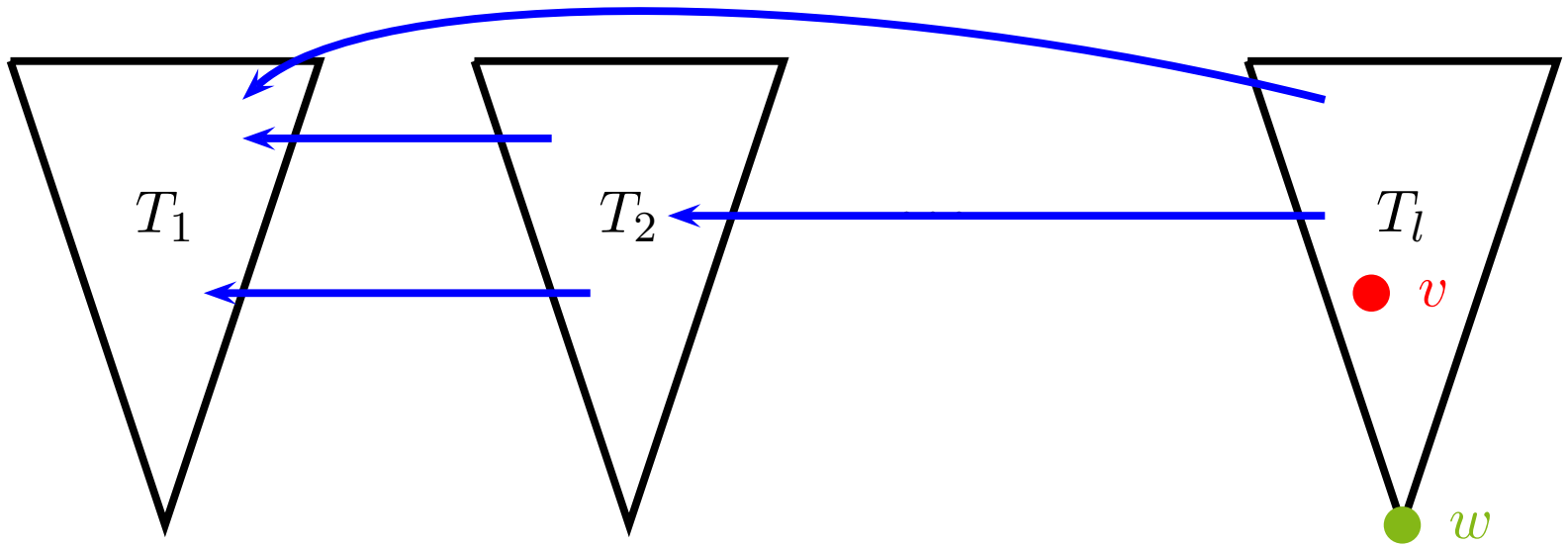
v	num(v)	$f(v)$
a	1	3
b	2	5
c	5	4
d	6	2
e	3	7
f	7	1
g	4	6

SZHK
1: f
2: d
3: a, e, b, g
4: c



Korrektheit — Vorüberlegungen

DFS auf G liefert T-Bäume $T_1, T_2, \ldots, T_l$.



Kanten zwischen den Bäumen verlaufen nur von rechts nach links.

Also ist jede SZHK vollständig in einem T-Baum.

Betrachte Wurzel w von T_l: w hat minimale f-Nummer: $f[w] = 1$

In G^* kann DFS(w) keinen Knoten $\notin T_l$ erreichen.

Betrachte Knoten v mit $w \to v$ in G^*: $\boldsymbol{v \leftrightarrow w}$

Das gilt für alle Knoten v im T^*-Baum mit Wurzel w.

Also wird die SZHK von w korrekt berechnet.

Korrektheitsbeweis

per Induktion über die Anzahl k der SZHK:

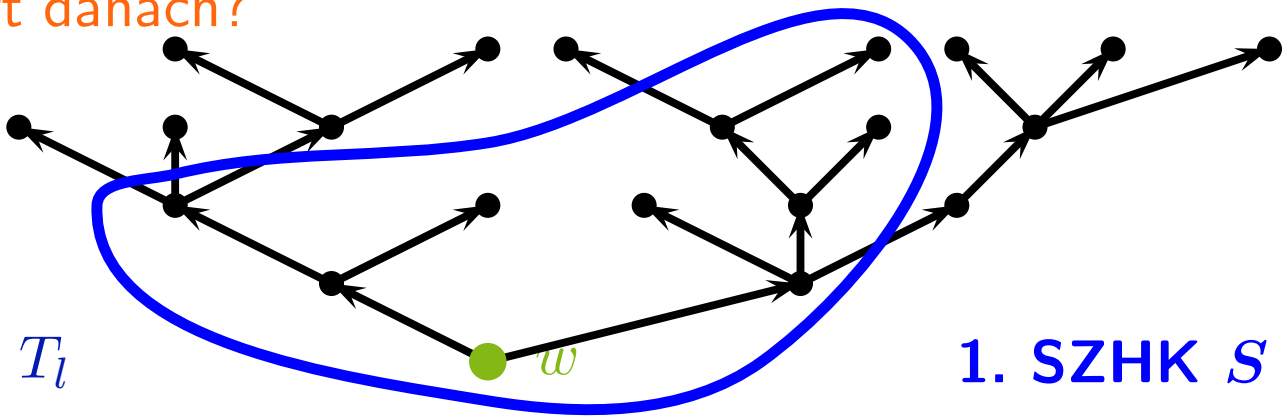
Induktionsanfang $k = 1$

wie gesehen korrekt berechnet ✓

Induktionsschritt:

wie gesehen erste SZHK korrekt berechnet

Was passiert danach?



Beobachtung: $T_1, T_2, \ldots, T_{l-1}, T_l \setminus S$ entspricht einer DFS-Traversierung von $G \setminus S$ in passender Reihenfolge

$G \setminus S$ ist ein Graph mit $k - 1$ SZHK.

Induktionsvoraussetzung ✓ $\rightsquigarrow$ □

Wir haben gezeigt:

Lemma

Der Algorithmus von Kosaraju berechnet die starken Zusammenhangskomponenten in einem Graphen $G = (V, E)$ korrekt und besitzt eine Laufzeit von $O(|V| + |E|)$.