7. Übung zur Vorlesung Algorithmen auf Graphen

Musterlösungen

Aufgabe 1: Aufgrund der vorgegebenen Knotenreihenfolge für die erste Phase führt der Algorithmus Starke Zusammenhangskomponenten zu den nachfolgenden Ergebnissen.

a) Die starken Zusammenhangskomponenten sind $\{a\}$, $\{b\}$, $\{c, d, g, h\}$ und $\{e, f\}$.

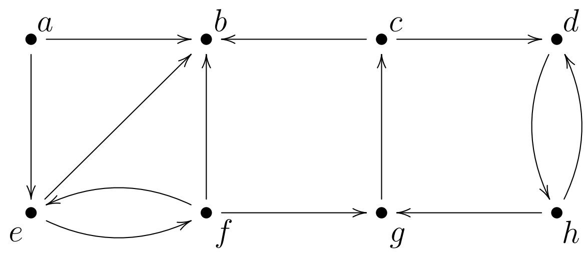
b) Die ermittelten *Push*– und *Pop*–Zeiten können der nachfolgenden Tabelle entnommen werden:

Knoten	a	b	c	d	e	f	g	h
Push–Zeit $d[v]$	1	15	5	4	8	7	6	3
Pop–Zeit $f[v]$	2	16	12	13	9	10	11	14

Der Keller L enthält nach der ersten Phase die Knotensequenz $[a, e, f, g, c, d, h, b]$. Dabei ist b der oberste Knoten und wird in der nächsten Phase als erstes wieder entfernt.

Das Ergebnis dieser ersten Phase ist eindeutig.

c) Der umorientierte Graph G' sieht wie folgt aus:



d) Die Verarbeitung der Knoten aus L hat die folgenden Auswirkungen:

Knoten	Reaktion
b	DFS wird mit Knoten b gestartet, Komponente $\{b\}$ wird ausgegeben
h	DFS wird mit Knoten h gestartet, Komponente $\{c, d, g, h\}$ wird ausgegeben
d	Knoten ist bereits markiert, keine Reaktion
c	Knoten ist bereits markiert, keine Reaktion
g	Knoten ist bereits markiert, keine Reaktion
f	DFS wird mit Knoten f gestartet, Komponente $\{e, f\}$ wird ausgegeben
e	Knoten ist bereits markiert, keine Reaktion
a	DFS wird mit Knoten a gestartet, Komponente $\{a\}$ wird ausgegeben

Aufgabe 2: Aufgrund der vorgegebenen Knotenreihenfolge für die erste Phase führt der Algorithmus Starke Zusammenhangskomponenten zu den nachfolgenden Ergebnissen.

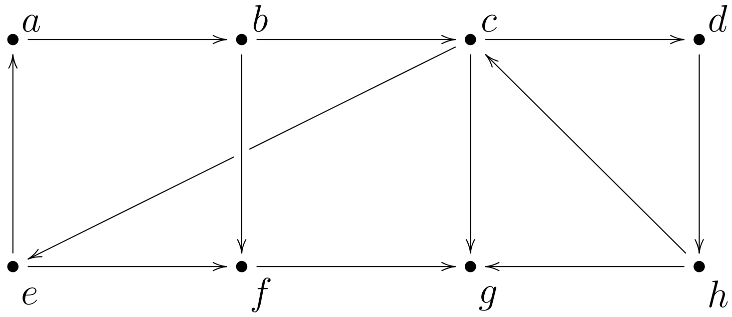
a) Die starken Zusammenhangskomponenten sind $\{a, b, c, d, e, h\}$, $\{f\}$ und $\{g\}$.

b) Die *Push*– und *Pop*–Zeiten sind bei der vorgegebenen Knotenreihenfolge eindeutig und können der folgenden Tabelle entnommen werden:

Knoten	a	b	c	d	e	f	g	h
Push–Zeit $d[v]$	5	4	3	2	6	13	15	1
Pop–Zeit $f[v]$	8	9	10	11	7	14	16	12

Der Keller L enthält nach der ersten Phase die Knotensequenz $[e, a, b, c, d, h, f, g]$. Dabei ist g der oberste Knoten und wird in der nächsten Phase als erstes wieder entfernt.

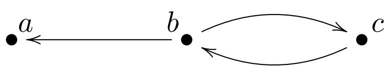
c) Der umorientierte Graph G' sieht wie folgt aus:



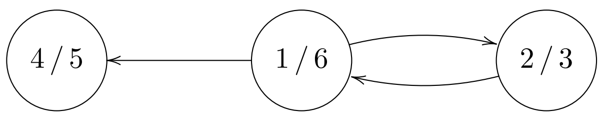
d) Die Verarbeitung der Knoten aus L hat die folgenden Auswirkungen:

Knoten	Reaktion
g	DFS wird mit g gestartet, Komponente $\{g\}$ wird ausgegeben
f	DFS wird mit f gestartet, Komponente $\{f\}$ wird ausgegeben
h	DFS wird mit h gestartet, Komponente $\{a, b, c, d, e, h\}$ wird ausgegeben
d	Knoten ist bereits markiert, keine Reaktion
c	Knoten ist bereits markiert, keine Reaktion
b	Knoten ist bereits markiert, keine Reaktion
a	Knoten ist bereits markiert, keine Reaktion
e	Knoten ist bereits markiert, keine Reaktion

Aufgabe 3: Man kann durch ein einfaches Gegenbeispiel zeigen, dass der geänderte Algorithmus nicht korrekt ist. Ein dazu geeigneter Graph war vorgegeben:



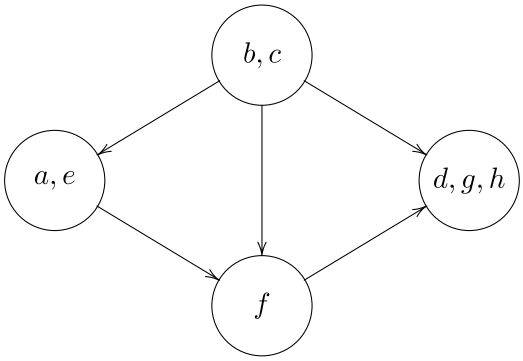
Die — unveränderte — erste Phase könnte mit b als Startknoten für den ersten DFS–Aufruf beginnen. Hierdurch würden dann alle Knoten markiert, und zwar möglicherweise mit diesem Ergebnis:



In L würden sich dann die Knoten $[c, a, b]$ befinden, wobei — da es sich jetzt um eine Queue und nicht mehr um einen Keller handelt — die Entnahme der Knoten mit c beginnt. Bereits der erste DFS–Aufruf würde wiederum — auf dem nicht umorientierten Graph — alle Knoten markieren, so dass die falsche Zusammenhangskomponente $\{a, b, c\}$ ermittelt werden würde. Richtig sind aber $\{a\}$ und $\{b, c\}$. $\square$

Aufgabe 4: Hier die Lösung zu beiden Teilaufgaben:

a) Für den Beispielgraph G sieht G' wie folgt aus (die Mengenklammern wurden dabei aus Platzgründen weggelassen, d.h. a, e bezeichnet z.B. die Menge $\{a, e\}$):



b) Angenommen, G' wäre nicht immer azyklisch. Dann existiert ein Graph G, so dass der zugehörige Graph G' einen Kreis $C_1 \rightarrow C_2 \rightarrow \ldots \rightarrow C_k$ mit $k \geq 2$ und $C_k = C_1$ besitzt. Nun korrespondiert die Kante (C_1, C_2) in E' zu einer Kante $(v, v') \in E$ mit $v \in C_1$ und $v' \in C_2$. Da alle Knoten aus C_1 den Knoten v erreichen können (denn C_1 ist eine starke Zusammenhangskomponente), kann man von allen Knoten aus C_1 auch den Knoten v' erreichen. Genauso kann man von $v' \in C_2$ alle Knoten aus C_2 erreichen, d.h. man kann insgesamt gesehen von jedem beliebigen Knoten aus C_1 jeden beliebigen Knoten aus C_2 erreichen. Analog kann man für C_2 und C_3, für C_3 und C_4, usw. bis hin zu C_{k-1} und C_k (also C_{k-1} und C_1) argumentieren. Für beliebige Knoten v und w aus zwei beliebigen starken Zusammenhangskomponenten des Kreises lassen sich also problemlos verbindende Pfade in beide Richtungen konstruieren. Also müssten alle Knoten aus allen Komponenten eigentlich in einer einzigen starken Zusammenhangskomponente von G liegen. Dies ist ein Widerspruch. $\square$