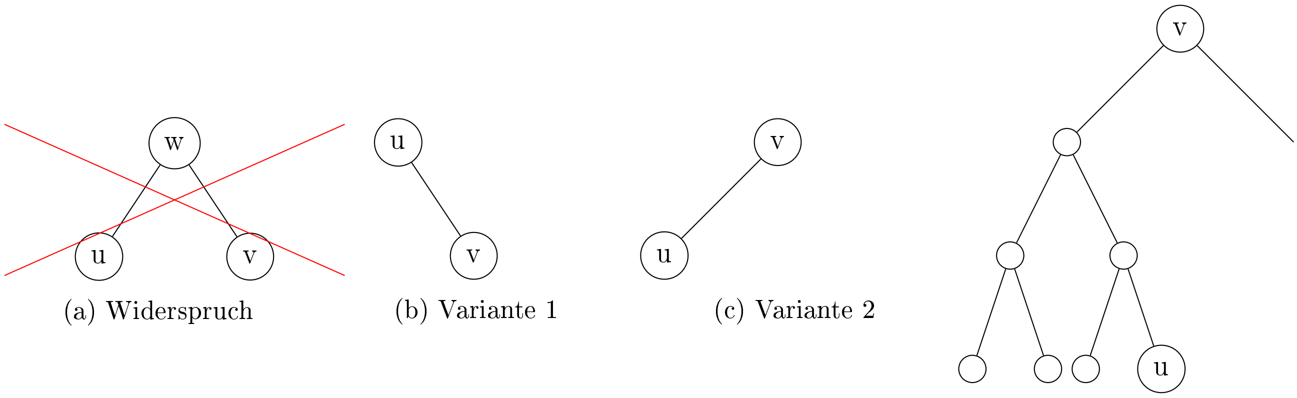
Graphentheorie
Blatt 4

Markus Vieth Christian Stricker

30. November 2016

1 Aufgabe 1

1.1 a)



(a) Widerspruch (b) Variante 1 (c) Variante 2 (d) Variante 3

Angenommen es gäbe einen LCA(u,v) w ungleich u oder v, so ist w „zwischen“ u und v im Widerspruch zur Annahme, dass v unmittelbarer Nachfolger von u ist. Somit gilt, dass w gleich u oder v ist.

q.e.d.

1.2 b)

Fall 1 Offensichtlich gilt $\Delta(T) = \delta(v_{2^h-2}, v_{2^h-1}) = h$, wobei v_{2^h-2} das letzte Element der Traversierung des linken Teilbaumes ist (vergleiche u aus Abb d)) und v_{2^h-1} die Wurzel ist (vergleiche v aus Abb d)).

Fall 2 $\Delta(T) = 1$, da $\delta(v_i, v_{i+1}) = 1 \ \forall i \in (0, \ldots, h-1)$

2 Aufgabe 2

2.1 a)

Angenommen es gäbe einen MST $\tilde{T}$ von G und eine lokal minimale Kante (u, v) in G aber nicht in $\tilde{T}$. Das $\tilde{T}$ MST gilt es existiert ein Pfad von u nach v in $\tilde{T}$. Da (u, v) lokal minimal ist gilt auch, $\delta((u, v)) < \delta((u, w))$ mit (u, w) in $\tilde{T}$ und $w \neq v$. Nun gilt offensichtlich:

1. Durch das entfernen von (u, w) und hinzunehmen von (u, v) sinkt das gesamt Gewicht
2. Durch das entfernen von (u, w) und hinzunehmen von (u, v) entsteht wieder ein Baum

Somit gilt der neue Baum T ist kleiner als $\tilde{T}$ im Widerspruch zur Annahme $\tilde{T}$ sei MST.
$\Rightarrow$ Jede lokal minimale Kante gehört zu allen MST in G.

q.e.d.

2.2 b)

zu zeigen Die Menge aller lokal minimalen Kanten eines Graphen ist ein Wald $\Rightarrow$ Es reicht zu zeigen: Die Menge aller minimalen Kanten eines Graphen ist zykelfrei

Beweis Angenommen es existiert ein Zyklus. Dank der injektiven Gewichtsfunktion existiert genau eine Kante (u, v) mit maximalem Gewicht in diesem Zyklus. Da es sich um einen Zyklus handelt, existieren die Kanten (u, t) und (v, w). Da (u, v) maximales Gewicht im Zyklus hat gilt:

$$\delta((u,t)) < \delta((u,v)) > \delta((v,w))$$

$\Rightarrow$ (u, v) ist nicht minimal im Widerspruch zur Annahme.

q.e.d.

3 Aufgabe 3

3.1 Borůvka in $O(|E| \log |V|)$

Wir initialisieren unseren Wald F mit $(V, \emptyset)$. In jeder Iteration, sucht jeder Baum T in F die leichteste Kante in G, die T verlässt und markiert diese. Anschließend werden alle markierten Kanten F hinzugefügt und wieder demarkiert. Dies wird wiederholt, bis F komplett verbunden ist. Wie im letzten Blatt gezeigt, folgt eine Laufzeit in $O(|E| \log |V|)$.

3.2 Borůvka in $O(|V|^2)$

Wir starten mit unserem Graphen $G_0 = (V_0, E_0)$. In jeder Iteration i initialisieren wir einen Wald F_i mit $(V_i, \emptyset)$. Jeder Knoten $v \in V_i$ in F_i sucht die leichteste Kante in G_i, die v verlässt un markiert diese. Anschließend werden alle markierten Kanten F_i hinzugefügt. Nun wird G_i mit den in F_i gewählten Kanten reduziert, also alle in F_i verbunden Knoten werden zu einem Überknoten zusammengefasst. G_{i+1} ist nun der Graph, der aus der so zusammengefassten Knotenmenge V_{i+1} besteht und allen angepassten Kanten, ohne 1- und 2-Zykel-Kanten (z.B. (u, u) oder $(u, v), (v, u)$). Mit der gleichen Argumentation wie im letzten Blatt kann man sagen, dass $|V_{i+1}| \in O\left(\frac{|V_i|}{2}\right)$ gilt. Der Algorithmus terminiert, wenn $|V_i| = 1$. Die Lösung sind dann alle bislang hinzugefügt Kanten. Wenn wir $|E_i| \leq |V_i|^2$ abschätzen folgt für eine Iteration eine Laufzeit von $O(|V_i|^2)$. Somit gilt, dass die Laufzeit in $O\left(|V|^2 + \left(\frac{|V|}{2}\right)^2 + \left(\frac{|V|}{4}\right)^2 + \ldots\right) = O(|V|^2)$

3.3 Borůvka in $O\left(|E| \log\left(\frac{|V|^2}{|E|}\right)\right)$

Wir nehmen beide Algorithmen und nutzen in jeder Iteration den gerade schnelleren, damit erhalten wir pro Iteration eine Laufzeit von $O(|E_i|)$ wobei $|E_i|$ durch $|E|$ und $|V_i|^2$ begrenzt ist. Wir wählen also den 1. Algorithmus solange gilt $|E| < |V_i|^2$. Dies passiert in der $j = \frac{1}{2} \log\left(\frac{|V|^2}{|E|}\right)$ Iteration, da gilt:

$$\begin{aligned}
&|E| = |V_i|^2 \\
\Leftrightarrow &|E| = \left(\frac{|V|}{2^i}\right)^2 = \frac{|V|^2}{2^{2i}} \\
\Leftrightarrow &2^{2i} = \frac{|V|^2}{|E|} \\
\Leftrightarrow &2i = \log_2\left(\frac{|V|^2}{|E|}\right) \\
\Leftrightarrow &i = \frac{1}{2}\log_2\left(\frac{|V|^2}{|E|}\right)
\end{aligned}$$

Die ersten j Iterationen je $O(|E|)$ lange und für die restlichen insgesamt $O\left(\sum_{i \geq j}^{|V_i|^2}\right)$ $|V_j|^2$ war gerade $|E|$ und somit gilt

$$O\left(|V_j|^2 + \left(\frac{|V_j|}{2}\right)^2 + \left(\frac{|V_j|}{4}\right)^2 + \dots\right) = O\left(|E| + \frac{|E|}{4} + \frac{|E|}{16} + \dots\right) = O(|E|)$$

Daraus folgt eine gesamt Laufzeit für den Algorithmus von $O(|E|(1+j)) = O\left(|E| + |E|\frac{1}{2}\log\left(\frac{|V|^2}{|E|}\right)\right) = O\left(|E|\frac{1}{2}\log\left(\frac{|V|^2}{|E|}\right)\right)$

Der Algorithmus arbeitet Korrekt, da er das Ergebnis der beiden anderen Algorithmen nicht verändert.