Algorithmen und
Berechnungskomplexität I
Wintersemester 2022/23

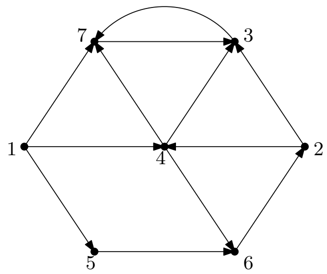
Abgabe: 14.12.2022 bis 10:00 Uhr

Prof. Dr. Anne Driemel
Frederik Brüning, Jan Eube
Institut für Informatik

Übungsblatt 9

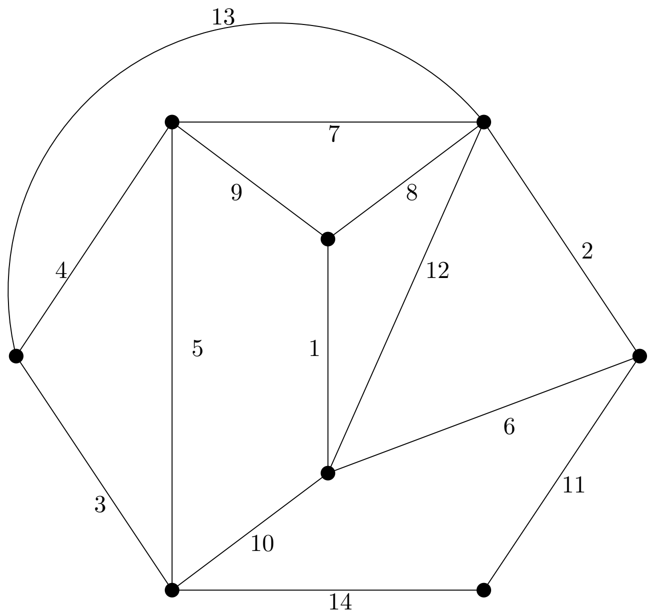
Aufgabe 9.1: Tiefensuche (4 Punkte)

Führen Sie eine Tiefensuche auf dem unten abgebildeten Graphen G durch. Geben Sie dafür den ***D**epth-**F**irst-**S**earch*-Baum an, der entsteht, wenn Sie bei Knoten 1 beginnen.Nehmen Sie hierbei an, dass Nachbarn mit kleinerer Nummer stets zuerst besucht werden. Ordnen Sie jede Kante ihrer Klasse (T-, F-, C- oder B-Kante) zu und geben Sie die *discover* und *finish* Zeiten der Knoten an.



Aufgabe 9.2: Divide-and-Conquer für Spannbäume (4+ 3 = 7 Punkte)

(a) Bestimmen Sie mithilfe des Algorithmus von Kruskal einen minimalen Spannbaum des folgenden Graphen. Geben Sie dazu die Reihenfolge an, in der Kruskal die Kanten betrachtet. Außerdem soll für jede Kante, die nicht in den Spannbaum aufgenommen wurde, einen Kreis angegeben werden, der durch die Kante entstanden wäre. Zeichnen sie schließlich den final resultierenden Spannbaum.



(b) Einige Probleme lassen sich mithilfe einer Divide-and-Conquer-Strategie lösen. Wir wollen diesen Ansatz für die Berechnung von minimalen Spannbäumen untersuchen.

Gegeben seien ein zusammenhängender ungerichteter Graph $G = (V, E)$ mit einer Kantengewichtung $w\colon E \to \mathbb{R}_{>0}$ und eine Aufteilung $(V_1, V_2 := V \setminus V_1)$ der Knotenmenge V. Es seien E_1 und E_2 die Mengen der Kanten $\{u, v\}$ mit $u, v \in V_1$ bzw. $u, v \in V_2$. Wir nehmen an, dass die Graphen $G_1 = (V_1, E_1)$ und $G_2 = (V_2, E_2)$ zusammenhängend sind. Ein Divide-and-Conquer-Ansatz könnte wie folgt aussehen.

1. Bestimme minimale Spannbäume von G_1 und G_2.

2. Bestimme eine kostenminimale Kante zwischen Knotenmenge V_1 und Knotenmenge V_2.
3. Vereinige die beiden Spannbäume und die Kante zu einem Spannbaum von G.

Beweisen oder widerlegen Sie, dass dieser Ansatz einen minimalen Spannbaum liefert.

Aufgabe 9.3: Maximale Spannbäume (3 Punkte)

Gegeben sei ein zusammenhängender ungerichteter Graph $G = (V, E)$ mit einer Kantengewichtung $w\colon E \to \mathbb{R}_{>0}$. Wir wollen in Zeit $O(|E| \log |E|)$ einen **maximalen** Spannbaum von G bestimmen, d. h. einen Spannbaum T mit maximalem Gewicht $w(T)$. Dazu konstruieren wir aus w eine Kantengewichtung w', sodass für alle Kanten e gilt, dass $w'(e) = W - w(e)$, wobei $W = \max_{e \in E} w(e) + 1$. Daraufhin wird Kruskal ausgeführt um einen minimalen Spannbaum T bezüglich w' zu berechnen. Zeigen Sie, dass T gleichzeitig ein maximaler Spannbaum bezüglich w ist.

Aufgabe 9.4: Alternativen zu Kruskal (6 Punkte)

Wir betrachten den folgenden Algorithmus, der als Eingabe einen nichtleeren ungerichteten Graphen $G = (V, E)$ und eine Kantengewichtung $w\colon E \to \mathbb{R}_{>0}$ erhält.

AltMST(G, w)

1. **if** (G ist nicht zusammenhängend) **then return** „G ist nicht zusammenhängend.“
2. Wähle einen Knoten $v_0 \in V$.
3. Setze $V_T = \{v_0\}$ und $T = \emptyset$.
4. **for** $i = 1, \ldots, |V| - 1$ **do**
5. Bestimme eine gewichtsminimale Kante $e_i = \{u_i, v_i\}$ mit $u_i \in V_T$ und $v_i \notin V_T$.
6. Setze $V_T = V_T \cup \{v_i\}$ und $T = T \cup \{e_i\}$.
7. **end for**
8. **return** T

Zeigen Sie, dass der Algorithmus AltMST einen minimalen Spannbaum von G bezüglich der Gewichtsfunktion w berechnet, sofern G zusammenhängend ist.