9 Minimale spannende Bäume

Wir betrachten folgendes Problem.

MINIMUM SPANNING TREE

Instanz: Ein zusammenhängender Graph G und eine Kostenfunktion $c : E(G) \to \mathbb{R}$.

Aufgabe: Bestimme einen Baum T mit $T \subseteq G$ und $V(T) = V(G)$, für den

$$c(T) := c(E(T)) := \sum_{e \in E(T)} c(e)$$

minimal ist. Solch ein T nennt man Minimum Spanning Tree.

Satz 9.1. *Es sei (G, c) eine Instanz des* MINIMUM SPANNING TREE *und T ein Baum mit $V(T) = V(G)$ und $E(T) \subseteq E(G)$. Folgende Aussagen sind äquivalent.*

(a) T ist eine optimale Lösung von MINIMUM SPANNING TREE.

(b) Für jede Kante $xy \in E(G) \setminus E(T)$ gilt $c(e) \leq c(xy)$ für alle Kanten e des Weges in T zwischen x und y.

(c) Für eine Kante $e \in E(T)$ sei $E_G(e)$ die Menge der Kanten von G zwischen den beiden Komponenten von $T - e$. Es gilt für jede Kante $e \in E(T)$

$$c(e) = \min_{f \in E_G(e)} c(f).$$

□

Algorithmus 9.2. KRUSKAL'S ALGORITHM

Input: *Ein zusammenhängender Graph G und eine Kostenfunktion $c : E \to \mathbb{R}$.*
Output: *Ein* MINIMUM SPANNING TREE *T für (G, c).*

```
begin
    Sortiere die Kanten von G so, dass c(e_1) ≤ c(e_2) ≤ ... ≤ c(e_m) gilt;
    T ← (V(G), ∅);
    for i = 1 to m do
        if T + e_i ist ein Wald then
            T ← (V(G), E(T) ∪ {e_i});
        end
    end
    return T;
end
```

Satz 9.3. KRUSKAL'S ALGORITHM *arbeitet korrekt und kann so implementiert werden, dass seine Laufzeit $O(m(G) \log n(G))$ beträgt.*

□

Algorithmus 9.4. PRIM'S ALGORITHM

Input: *Ein zusammenhängender Graph* G *und eine Kostenfunktion* $c : E(G) \to \mathbb{R}$.
Output: *Ein* MINIMUM SPANNING TREE T.

begin
- *Wähle* $v \in V(G)$;
- $T \leftarrow (\{v\}, \emptyset)$;
- **while** $V(T) \neq V(G)$ **do**
  - *Wähle eine Kante* xy *minimaler Kosten* $c(xy)$ *mit* $x \in V(T)$ *und* $y \in V(G) \setminus V(T)$;
  - $T \leftarrow (V(T) \cup \{y\}, E(T) \cup \{e\})$;
- **end**
- **return** T;

end