

Übungen zur Vorlesung
Effiziente Algorithmen
SS 2002
Blatt 4

Aufgabe 4.1 (4 Punkte)

Der DFS-Algorithmus zerlegt die Kantenmenge E eines *gerichteten* Graphen $G = (V, E)$ in disjunkte Mengen von Tree-, Forward-, Back- und Cross-Kanten (Skript S. 7/8).

Betrachten Sie nun den BFS-Algorithmus (vgl. Aufg. 3.2 a) für G . In welche disjunkten Kantenmengen kann E durch diesen Algorithmus zerlegt werden? Ändern Sie den Algorithmus zu Aufg. 3.2 a) derart, dass zusätzlich zu den BFS-Nummern die Kantenmengen berechnet werden.

Aufgabe 4.2 (4 Punkte)

Einige in den Algorithmus von Dijkstra eingehende Ideen lassen sich oft auf eingeschränkte Graphklassen übertragen. Finden Sie für kantenbewertete, kreisfreie, gerichtete Graphen $G = (V, E, c)$ mit $c: E \rightarrow \mathbb{R}$ einen Algorithmus mit Rechenzeit $O(|V| + |E|)$, der von einem Knoten $s \in V$ kürzeste (d. h. kostenminimale) Wege zu allen Knoten $v \in V$ bestimmt.

Hinweis: Es ist hilfreich, sich zu überlegen, in welcher Reihenfolge die in der Beschreibung des Dijkstra-Algorithmus mit A bezeichnete Menge vergrößert werden soll.

Aufgabe 4.3 (4 Punkte)

Eine Punktmenge $M \subseteq \mathbb{R}^2$ der euklidischen Ebene heißt *konvex*, wenn zu je zwei Punkten $p, q \in M$ auch jeder Punkt der Verbindungslinie von p nach q in M enthalten ist. Die *konvexe Hülle* einer Punktmenge $M \subseteq \mathbb{R}^2$ ist die kleinste konvexe Punktmenge, die M umfasst. (Anschaulich kann man sich die Konstruktion der konvexen Hülle so vorstellen: Man legt eine Schnur kreisförmig um die Punktmenge M und zieht dann diese Schnur stramm.)

Gegeben seien nun n Punkte $p_1, \dots, p_n \in \mathbb{R}^2$ der euklidischen Ebene. Es sei vorausgesetzt, dass diese Punkte hinsichtlich ihrer x -Koordinate sortiert sind. Die konvexe Hülle der Punktmenge $\{p_1, \dots, p_n\}$ kann dann dadurch konstruiert werden, dass man sukzessive für jeweils die ersten i ($1 \leq i \leq n$) Punkte (bez. der gegebenen Sortierung) die konvexe Hülle bestimmt.

Geben Sie einen Algorithmus an, der die konvexe Hülle der Punktmenge $\{p_1, \dots, p_n\}$ in Zeit $O(n)$ berechnet!

Hinweis: Verwenden Sie für die Rechenzeitabschätzung eine amortisierte Analyse.

Aufgabe 4.4 (4 Punkte)

Gegeben ist eine Stackimplementierung mit den folgenden Operationen:

- $CreateStack()$ erzeugt einen leeren Stack.
- $Push(S, i)$ fügt Objekt i zum Stack hinzu.
- $Pop(S)$ entfernt das zuletzt eingefügte Objekt vom Stack S und liefert es als Ergebnis.
- $Top(S)$ liefert das zuletzt eingefügte Objekt vom Stack S als Ergebnis.
- $Empty(S)$ gibt an, ob der Stack S leer ist. ↪

Jede dieser fünf Operationen wird in Zeit $O(1)$ durchgeführt. Beschreiben Sie die Implementierung einer Queue, welche die Operationen

- *CreateQueue()*, erzeuge eine leere Queue,
- *Enqueue(Q, i)*, füge das Objekt i zum Queueinhalt von Q hinzu,
- *Dequeue(Q)*, entferne unter allen Queueelementen in Q das zuerst eingefügte und liefere es als Ergebnis,
- *Empty(Q)*, gib an, ob die Queue leer ist,

unterstützt, wobei als Datenstruktur nur zwei Stacks verwendet werden dürfen und die amortisierte Rechenzeit von n Operationen bei $O(n)$ liegen soll. Führen Sie die Analyse der amortisierten Rechenzeit mit der Buchhaltermethode durch.