ETH Zürich
Institut für Theoretische Informatik
Prof. Dr. Angelika Steger

WS 2004/05
Klausur

Klausur: Algorithmen und Komplexität

Aufgabe 1

Im folgenden betrachten wir Algorithmen, die zu je zwei Folgen $a_1, \ldots, a_n$ und $b_1, \ldots, b_m$ von natürlichen Zahlen entscheiden, ob es Indizes i, j mit $a_i = b_j$ gibt oder nicht.

Algorithmus 1
Input: $a_1, \ldots, a_n$, $b_1, \ldots, b_m$ $(\in \mathbb{N})$
Output: TRUE, wenn i, j existieren mit $a_i = b_j$, sonst FALSE

```
1: for j = 1 to m do
2:   for i = 1 to n do
3:     if a_i = b_j then
4:       return TRUE;
5:     end if
6:   end for
7: end for
8: return FALSE;
```

(a) Bestimmen Sie die Anzahl der im worst-case benötigten Vergleiche von Algorithmus 1 in Abhängigkeit von (n, m).

(b) Wir betrachten folgende modifizierte Version des obigen Algorithmus. Die Prozedur BinarySearch$(x, a_1, \ldots, a_n)$ testet mittels binärer Suche, ob $x = a_i$ für ein $i \in \{1, \ldots, n\}$ gilt, und gibt entsprechend den Wert TRUE oder FALSE zurück.

Algorithmus 2
Input: $a_1, \ldots, a_n$, $b_1, \ldots, b_m$ $(\in \mathbb{N})$
Output: TRUE, wenn i, j existieren mit $a_i = b_j$, sonst FALSE

```
1: MergeSort(a_1, ..., a_n)
2: for j = 1 to m do
3:   if BinarySearch(b_j, a_1, ..., a_n)=TRUE then
4:     return TRUE;
5:   end if
6: end for
7: return FALSE;
```

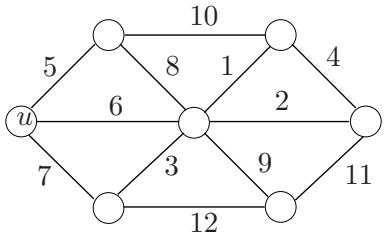
Wie groß ist für Algorithmus 2 im worst-case die Anzahl der benötigten Vergleiche? (in Abhängigkeit von (n, m))

(c) Geben Sie (in PseudoCode) einen Algorithmus TEST an, der zu je zwei **sortierten** Folgen $a_1 < \ldots < a_n$ und $b_1 < \ldots < b_m$ höchstens $O(n+m)$ Vergleiche benötigt, um festzustellen, ob es Indizes i, j mit $a_i = b_j$ gibt oder nicht (Hinweis: Denken Sie an MergeSort).

(d) Entscheiden Sie, ob man mit folgender Strategie einen noch besseren Algorithmus erhalten kann: Sortiere zunächst beide Folgen $a_1, \ldots, a_n$ und $b_1, \ldots, b_m$ und benutze dann TEST, um zu testen, ob es Indizes i, j mit $a_i = b_j$ gibt oder nicht.

Aufgabe 2

Gegeben sei der folgende, gewichtete Graph $G = (V, E)$.



a) Bestimmen Sie einen minimalen Spannbaum von G. Erläutern Sie Ihre Vorgehensweise. Gibt es in G noch weitere minimale Spannbäume? (Begründung!)

b) Bestimmen Sie einen Spannbaum T von G, so dass für jeden Knoten v von G gilt: Der eindeutige Weg von Knoten u nach v in T ist ein kürzester Weg von u nach v in G. Erläutern Sie wiederum Ihre Vorgehensweise!

Aufgabe 3

Gegeben sei ein Graph $G(V, E)$ mit einem positiven rationalen Gewicht an jeder Kante. Das Gewicht eines Pfades sei das *Maximum* der Gewichte aller Kanten des Pfades (und nicht wie üblich die Summe der Gewichte aller Kanten). Beschreiben Sie eine modifizierten Dijkstra Algorithmus, der einen kürzesten Pfad findet. Geben Sie den Algorithmus als kommentierten Pseudo-Code an, und begründen Sie die Korrektheit des Algorithmusses.

Aufgabe 4

Ein Hamiltonkreis eines Graphens ist ein Kreis, der jeden Knoten des Graphens genau einmal besucht. Das Problem zu bestimmen, ob ein gerichteter Graph einen Hamiltonkreis besitzt, ist NP-volständig. Benutzen Sie dieses Resultat, um zu zeigen, dass es

a) NP-schwer ist, in einem ungerichteten Graphen einen Hamiltonkreis zu finden,
(Tip: Ersetzen Sie jeden Knoten durch einen Pfad)

b) NP-schwer ist, in einem mit Kantengewichten versehenen gerichteten Graphen einen kürzesten Kreis zu finden, der alle Knoten besucht. (Dies ist das sogenannte MINIMUM TRAVELLING SALESMAN Problem).

Aufgabe 5
Welche der folgenden Aussagen ist wahr. Begründen Sie Ihre Antworten.

a) Es gibt keine auf Vergleichen basierende priority-queue, die in $O(1)$ amortisierter Zeit, die INSERT und die DELETE-MIN Operationen ausführt.

b) Gegeben sei ein ungewichteter Graph $G = (V, E)$ und ein Knoten $v \in V$. Man kann in $O(|V| + |E|)$ Zeit die Länge der kürzesten Wege von u zu allen anderen Knoten in V bestimmen.

c) Damit der Algorithmus DEPTH-FIRST-SEARCH $O(|V| + |E|)$ Zeit benötigt, muss der Graph $G = (V, E)$ als Adjazenzmatrix gegeben sein.

d) Für jeden Matroid $(\mathcal{S},\mathcal{U})$ und jede nichtnegative Gewichtsfunktion $w : \mathcal{S} \to \mathbb{N}_0$ bestimmt der Greedy-Algorithmus eine optimale Lösung.

e) Sind die Kantengewichte eines Graphens alle unterschiedlich, so gibt es einen eindeutigen minimalen Spannbaum.