Tutorium Grundlagen: Algorithmen und Datenstrukturen

Übungsblatt Woche 13

Aufgabe 13.1 Spannbäume

Aufgabe 13.1

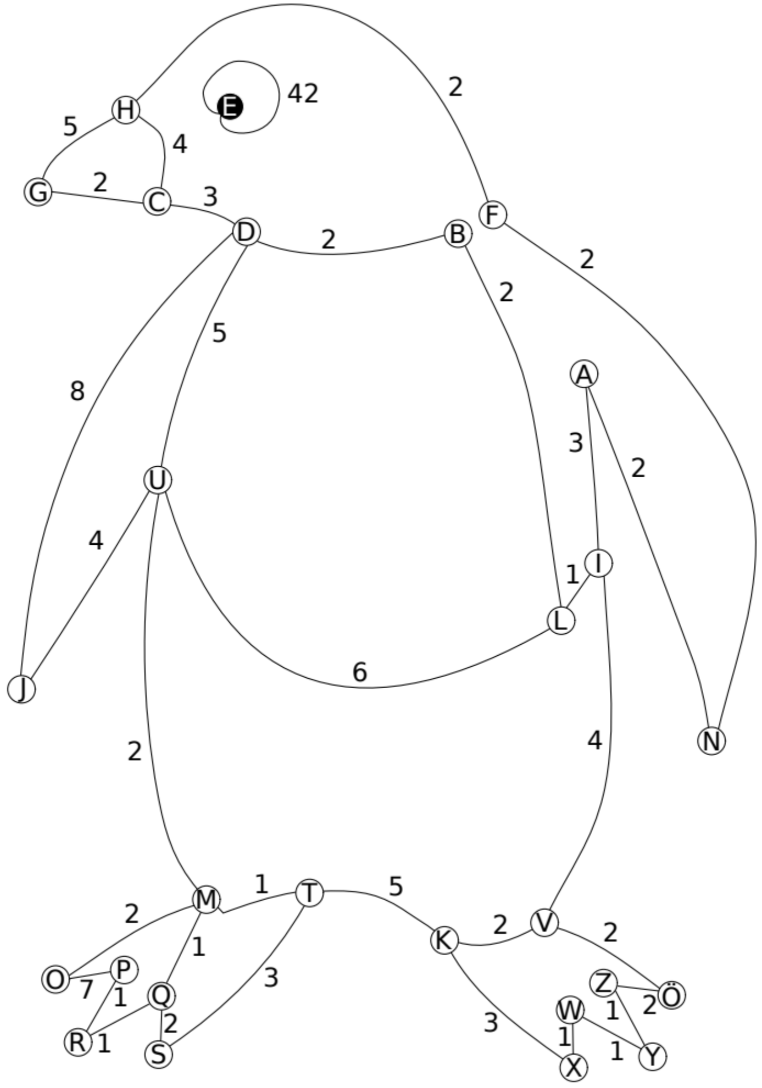
Gegeben sei der folgende Graph. Bearbeiten Sie mit diesem folgende Teilaufgaben:

a) Benutzen Sie den Jarník-Prim-Algorithmus, um im nachfolgenden Graphen aus von Knoten A einen minimalen Spannbaum zu berechnen. Vervollständigen Sie dazu die Tabelle unter dem Graphen, die den Inhalt der Prioritätswarteschlange nach jedem Schritt darstellt.
Die Warteschlange wird nach aufsteigender Priorität geordnet, bei mehreren Elementen mit gleicher Priorität wird alphabetisch nach Knotenbezeichnung sortiert (der Buchstabe ”Ö“ folgt auf ” O“). In ein Kästchen ist jeweils die Knotenbezeichnung zusammen mit der Priorität einzutragen. Markieren Sie Ihren minimalen Spannbaum außerdem im Graphen.

b) Wie kann man den Kruskal-Algorithmus so definieren, dass immer derselbe Spannbaum bei einem zusammenhängenden Graphen generiert wird, auch wenn der Graph z.B. nur Kanten mit demselben Gewicht enthält?

c) Knobelaufgabe: Beweisen Sie, dass auch Jarník-Prim so angepasst werden kann, dass immer derselbe Spannbaum generiert wird, egal bei welchem Knoten man beginnt.

Aufgabe 13.1 (a)



Schritt	Prioritätswarteschlange				
1	A 0				
2	N 2	I 3			
3	F 2	I 3			
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					
22					
23					
23					
24					
25					

Aufgabe 13.1 (b)

Wie kann man den Kruskal-Algorithmus so definieren, dass immer derselbe Spannbaum bei einem zusammenhängenden Graphen generiert wird, auch wenn der Graph z.B. nur Kanten mit demselben Gewicht enthält?

Aufgabe 13.1 (c)

Knobelaufgabe: Beweisen Sie, dass auch Jarník-Prim so angepasst werden kann, dass immer derselbe Spannbaum generiert wird, egal bei welchem Knoten man beginnt.

Aufgabe 13.2
Eure Fragen

Fragestellung 1

a) Was ist die Laufzeit von $f(n)$ in Abhängigkeit von $n \in \mathbb{N}$, wenn gilt $g(n) \in \mathcal{O}(1)$

b) Wie häufig wird $g(n)$ bei dem Aufruf $f(1, \texttt{false})$ insgesamt ausgeführt?

c) Wie häufig wird $g(n)$ bei dem Aufruf $f(2, \texttt{false})$ insgesamt ausgeführt?

d) Wie häufig wird $g(n)$ bei dem Aufruf $f(4, \texttt{false})$ insgesamt ausgeführt?

e) Wie häufig wird $g(n)$ bei dem Aufruf $f(8, \texttt{false})$ insgesamt ausgeführt?

f) Wie häufig wird $f(n)$ bei dem Aufruf $f(n, \texttt{false})$ mit $n \in \{2^k | k = 0, 1, \ldots\}$ insgesamt (inklusive des ersten Aufrufs) ausgeführt? Vervollständigen sie den Ausdruck mit den fehlenden Koeffizienten. Es gilt $\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$.

g) Gleiche Aufgabe wie f) nur für $g(n)$

```java
void f(int n, boolean b) {
    g(n);
    if (n > 1) {
        f(n / 2, true);
        f(n / 2, b);
    }
    if(b) {
        g(n);
    }
}
```

Funktion g(int n) ist gegeben

$$\_\_\_\ 2^n + \_\_\_\ n^2 + \_\_\_\ n - \_\_\_\ \log_2(n) - \_\_\_$$

Fragestellung 1 (a)

Was ist die Laufzeit von $f(n)$ in Abhängigkeit von $n \in \mathbb{N}$, wenn gilt $g(n) \in \mathcal{O}(1)$

```
void f(int n, boolean b) {
    g(n);
    if (n > 1) {
        f(n / 2, true);
        f(n / 2, b);
    }
    if(b) {
        g(n);
    }
}
```

Funktion g(int n) ist gegeben

Fragestellung 1 (b-e)

Wie häufig wird $g(n)$ bei dem Aufruf

b) $f(1, \texttt{false})$
c) $f(2, \texttt{false})$
d) $f(4, \texttt{false})$
e) $f(8, \texttt{false})$

insgesamt ausgeführt?

```
void f(int n, boolean b) {
    g(n);
    if (n > 1) {
        f(n / 2, true);
        f(n / 2, b);
    }
    if(b) {
        g(n);
    }
}
```

Funktion g(int n) ist gegeben

Fragestellung 1 (f-g)

f) Wie häufig wird $f(n)$ bei dem Aufruf $f(n, \texttt{false})$ mit $n \in \{2^k | k = 0, 1, \ldots\}$ insgesamt (inklusive des ersten Aufrufs) ausgeführt? Vervollständigen sie den Ausdruck mit den fehlenden Koeffizienten. Es gilt $\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$.

g) Gleiche Aufgabe wie f) nur für $g(n)$

```java
void f(int n, boolean b) {
    g(n);
    if (n > 1) {
        f(n / 2, true);
        f(n / 2, b);
    }
    if(b) {
        g(n);
    }
}
```

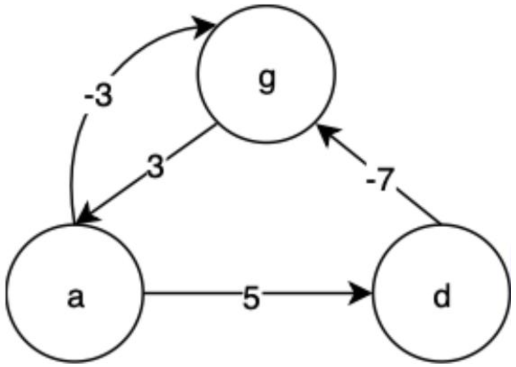
Funktion g(int n) ist gegeben

$$\_\_\_\ 2^n + \_\_\_\ n^2 + \_\_\_\ n - \_\_\_\ \log_2(n) - \_\_\_$$

$$\_\_\_\ 2^n + \_\_\_\ n^2 + \_\_\_\ n - \_\_\_\ \log_2(n) - \_\_\_$$

Fragestellung 2

Gegeben ist der folgende Graph G_3

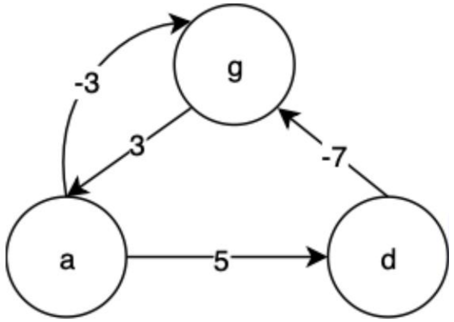


a) Warum lässt sich das APSP-Problem auf diesem Graphen G_3 nicht mittels des *Dijkstra-Algorithmus* lösen?

b) Als erster Schritt des Johnson-Algorithmus zur Lösung des APSP-Problems muss der Graph G_3 zuerst in einen Graphen mit nicht-negativen Kantengewichten transformiert werden. Berechnen Sie die dazu nötigen Potenziale $h(v)$ für alle Knoten $v \in G_3$ unter Verwendung des Hilfsknotens *s.* Geben Sie ausreichend Zwischenschritte an. Zeichnen Sie anschließend den transformierten Graphen mit den nicht-negativen Kantengewichten.

Fragestellung 2 (b)

Als erster Schritt des Johnson-Algorithmus zur Lösung des APSP-Problems muss der Graph G_3 zuerst in einen Graphen mit nicht-negativen Kantengewichten transformiert werden. Berechnen Sie die dazu nötigen Potenziale $h(v)$ für alle Knoten $v \in G_3$ unter Verwendung des Hilfsknotens *s.* Geben Sie ausreichend Zwischenschritte an. Zeichnen Sie anschließend den transformierten Graphen mit den nicht-negativen Kantengewichten.



Fragestellung 3 (a)

Gegeben sei die Familie von Hashfunktionen

$$h_a(x) \left\lfloor \frac{(a \cdot x) \bmod 16}{2^{4-k}} \right\rfloor$$

wobei $a \in \mathbb{N}, k \in \{1, 2, 3, 4\}$ und die floor-Operation $\lfloor\,.\,\rfloor$ auf ganze Zahlen abrundet.

Angenommen für ein Paar von beliebigen Werten $(x, y)_{x \neq y}$ gilt $h_a(x) = h_a(y)$.
Wir betrachten den Ausdruck $z = \big(a \cdot (x - y)\big) \bmod 16$ in Binärdarstellung.
Welche Aussage können Sie über die höchsten k Bits von z treffen, wenn mindestens ein anders Bit von z gesetzt ist? (Nur eine Aussage richtig)

- ☐ Identische zu den niedrigsten k Bits
- ☐ Abwechselnd 0 und 1
- ☐ Identisch zu den höchsten k Bits von $(a \cdot x) \bmod 16$
- ☐ Identisch zu den höchsten k Bits von $(a \cdot y) \bmod 16$
- ☐ Alle 0
- ☐ Alle gleich
- ☐ Alle 1
- ☐ Gleichverteilt

Fragestellung 3 (b)

Gegeben sei die Familie von Hashfunktionen

$$h_a(x) \left\lfloor \frac{(a \cdot x) \bmod 16}{2^{4-k}} \right\rfloor$$

Für c-universelle Hashfamilien $\mathcal{H}$ gilt

$$\forall (x, y)_{x \neq y}\colon \Pr\{h(x) = h(y)\} \leq \frac{c}{m}$$

wenn h zufällig aus $\mathcal{H}$ gewählt wird. Sei $m = 2^k$.

Wir untersuchen nun die Universalität der Hashfamilie $\widetilde{\mathcal{H}} = \{h_a \mid 0 < a < 16 \text{ und } a \text{ ungerade}\}$.

Benutzen Sie die Lösung aus dem vorigen Aufgabenteil um das kleinste c zu bestimmen, für das die obige Bedingung für $\widetilde{\mathcal{H}}$ im Allgemeinen erfüllt ist.