Optimierung für Studierende der Informatik

Wintersemester 2019/20
Blatt 11

A: Präsenzaufgaben am 15./16. Januar 2018

1. Gegeben sei eine Menge $S = \{s_1, \ldots, s_n\}$ und eine Kollektion $T_1, \ldots, T_m$ von k-elementigen Teilmengen von S. Außerdem besitze jedes Element s_i ein Gewicht $w_i \geq 0$ mit $w_i \in \mathbb{Q}$ ($i = 1, \ldots, n$).

Zur Erinnerung: Eine Teilmenge $H \subseteq S$ wird ein *Hitting Set* genannt, falls $H \cap T_i \neq \emptyset$ für alle $i = 1, \ldots, m$ gilt. Gesucht ist ein Hitting Set H, dessen Gewicht so klein wie möglich ist. Anders gesagt: Die Summe

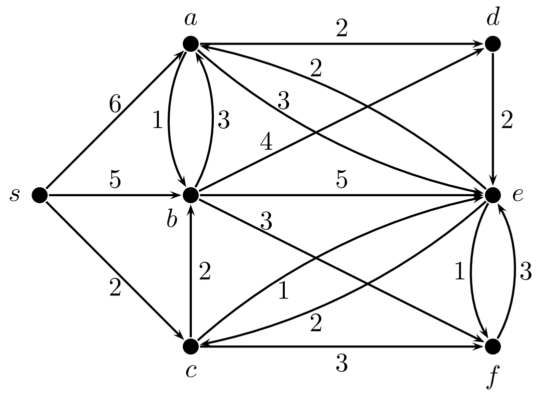
$$\sum_{s_i \in H} w_i$$

soll so klein wie möglich sein. Wir wollen das beschriebene Problem WEIGHTED k-HITTING SET nennen.

a) Formulieren Sie dieses Problem als ein ganzzahliges Programmierungsproblem, dass Sie (ILP) nennen.

b) Wie lautet die LP-Relaxation (LP) dieses Problems?

2. Der Graph $G = (V, E)$ mit Längenfunktion ℓ sei durch die folgende Zeichnung gegeben:



a) Verwenden Sie den Algorithmus von Dijkstra in der Version auf Seite 181 des Skripts, um für alle $v \in V$ die Länge $d(v)$ eines kürzesten s, v-Pfades zu berechnen. Legen Sie eine Tabelle wie auf Seite 183 des Skripts an, d.h., notieren Sie auch immer einen „Vorgängerknoten“.

b) Bestimmen Sie einen kürzeste-Pfade-Baum anhand der Einträge in der letzten Zeile Ihrer Tabelle.

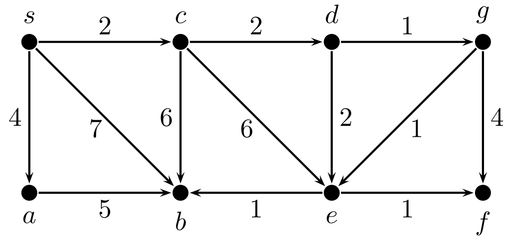
B: Hausaufgaben zum 22./23. Januar 2018

1. Wir knüpfen an Präsenzaufgabe 1 an und betrachten das dort formulierte Problem WEIGHTED k-HITTING SET. Die Bezeichnungen (ILP) und (LP) verwenden wir wie in dieser Präsenzaufgabe.

a) Geben Sie basierend auf (LP) einen (polynomiellen) Approximationsalgorithmus für WEIGHTED k-HITTING SET an, bei dem es sich um einen k-Approximationsalgorithmus handelt.

b) Weisen Sie nach, dass es sich bei dem von Ihnen angegebenen Algorithmus tatsächlich um einen k-Approximationsalgorithmus handelt.

2. a) Der Graph $G = (V, E)$ mit Längenfunktion ℓ sei durch die folgende Zeichnung gegeben:



Verwenden Sie den Algorithmus von Dijkstra (Skript, Seite 181) um für alle $v \in V$ die Länge $d(v)$ eines kürzesten s, v-Pfades zu berechnen. Legen Sie eine Tabelle an, an der man zusätzlich kürzeste s, v-Pfade ablesen kann. Bestimmen Sie auch einen kürzeste-Pfade-Baum.

b) Für den folgenden Graphen bestimme man einen minimalen aufspannenden Baum auf drei Arten:

(i) mit dem Algorithmus von Prim (mit Startknoten a);

(ii) mit dem Algorithmus von Kruskal;

(iii) mit dem Reverse-Delete-Algorithmus.

Geben Sie jeweils die Kanten in der Reihenfolge an, in der sie hinzugefügt bzw. weggelassen wurden. (Kommen mehrere Kanten infrage, so wähle man willkürlich eine aus.)