Algorithmen und Datenstrukturen

Blatt 6

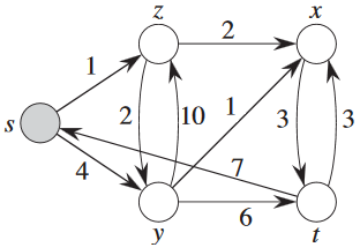
Uni Hamburg, Wintersemester 2019/20

Präsentation am 22.01. bis 24.01.2020

Jede Teilaufgabe zählt als ein einzelnes Kreuzchen.

Übung 1.

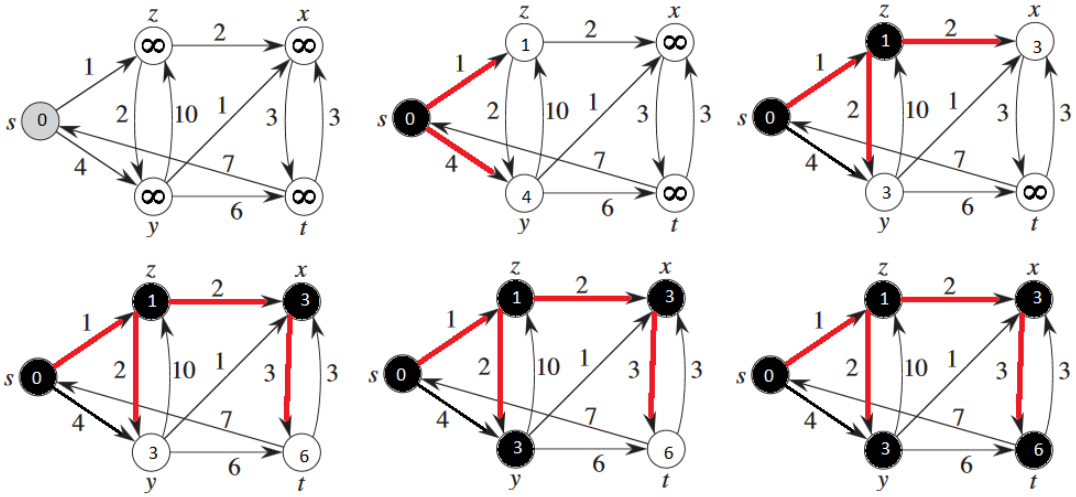
a) Finden Sie mittels Dijkstra den kürzesten Pfad zwischen s und t in dem folgenden Graphen. Skizzieren Sie dabei den Status des Algorithmus nach jedem Schleifendurchlauf.



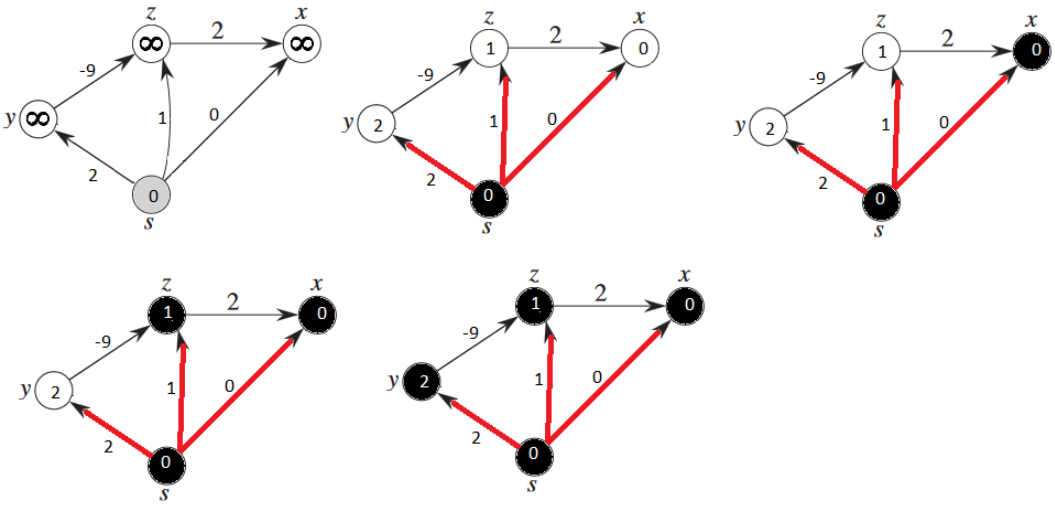
b) Finden Sie ein einfaches Beispiel eines Graphen mit einer Kante mit negativer Kantenlänge, in welchem Dijkstra den falschen kürzesten Weg berechnet. Stellen Sie dabei sicher, dass es noch einen kürzesten Weg gibt. Skizzieren Sie wie in (a) die Anwendung von Dijkstra.

Lösung 1.

(a) .



(b) Der kürzeste Weg zwischen s und x wäre in diesem Beispiel $s \to y \to z \to x \hat{=} -5$. Dijkstra findet allerdings den Pfad $s \to x \hat{=} 0$.



Übung 2.

Sei $G(V, E)$ ein einfacher Graph. Eine Knotenüberdeckung $K \subset V$ ist eine Menge an Knoten, sodass $N(K) := \{(u, v) \in E | u \in K \vee v \in K\}$, also die Menge aller Kanten die einen Knoten aus K enthalten, $N(K) = E$ erfüllt. Entsprechend ist eine minimale Knotenüberdeckung ein solches K, dessen Anzahl an Knoten minimal ist unter allen Knotenüberdeckungen.

(a) Sei $W \subseteq E$ ein maximales Matching und $v(W)$ die dazugehörigen Knoten. Beweisen Sie, dass $v(W)$ eine Knotenüberdeckung ist und folgere, dass eine minimale Knotenüberdeckung K daher $|K| \leq 2|W|$ erfüllt.

(b) Zeigen Sie weiter: Sei G ein bipartiter Graph und K und W wie in (a), dann gilt $|K| \geq |W|$.

Lösung 2.

(a) Wir nehmen an $v(W)$ ist keine Knotenüberdeckung und folgern dass W kein maximales Matching sein kann. In diesem Fall gibt es eine Kante $(u, v) \in E$ für die gilt, dass $v \notin v(W)$ und $u \notin v(W)$ gilt. Demnach sind weder v noch u beteiligt am Matching W, wodurch $W' = \{(u, v)\} \cup W$ ein gültiges Matching ergibt. Da $|W'| > |W|$, ist W somit kein maximales Matching. Eine minimale Knotenüberdeckung muss somit notwendigerweise $|K| \leq |v(W)|$ erfüllen, und da nach Definition eines Matchings $|v(W)| = 2|W|$ gilt, folgt hieraus direkt $|K| \leq 2|W|$.

(b) Nach der Definition des maximalen Matchings gilt für jeden Knoten $v \in V$, dass er Teil von höchstens einer Kante $e \in E$ ist. Somit benötigt man mindestens $|W|$ Knoten um alle Kanten aus W und ensprechend auch E zu überdecken. (Dass der Graph bipartit ist, spielt keine Rolle.)

Übung 3.
Wir modellieren ein Kommunikationsnetz durch einen gerichteten, gewichteten Graphen $G = (V, E)$. Eine Kante repräsentiert dabei den Kommunikationskanal zwischen zwei Knoten. Die Wichtung einer Kante $(u, v) \in E$ repräsentiert die Ausfallsicherheit $0 < r(u, v) \leq 1$ des Kanals. Dies ist die Wahrscheinlichkeit, dass der Kanal von u nach v nicht ausfallen wird und es wird angenommen, dass diese Wahrscheinlichkeiten unabhängig sind.

Modifizieren Sie den Graphen, so dass der Algorithmus von Dijkstra den zuverlässigsten Pfad zwischen zwei Knoten ermittelt.

Lösung 3.
Sei $\mathcal{P}(s, t)$ die Menge aller Pfade zwischen zwei Knoten s und t und $x \in \mathcal{P}(s, t)$ ein Pfad zwischen diesen Knoten. Die Wahrscheinlichkeit, dass der Pfad nicht ausfällt, $P(x)$, ist dann das Produkt der Wahrscheinlichkeiten, dass ein einzelner Kanal ausfällt, also $P(x) = \prod_{(u,v)\in x} r(u, v)$. Der sicherste Kanal ist offensichtlich der Kanal mit der höchsten Ausfallsicherheit, also der Kanal für den $P(x)$ maximal ist. Sei x_{max} dieser sicherste Pfad, dann gilt $P(x_{max}) \geq P(x)$, bzw. $\prod_{(u,v)\in x_{max}} r(u, v) \geq \prod_{(u,v)\in x} r(u, v)$, für alle $x \in \mathcal{P}(s, t)$. Da Dijkstra den Pfad mit der minimalen Summe ermittelt, muss die Eigenschaft des "maximalen Produktes" in die einer "minimalen Summe" überführt werden. Dafür nutzen wir den Logarithmus:

$$
\begin{aligned}
&\prod_{(u,v)\in x_{max}} r(u, v) \geq \prod_{(u,v)\in x} r(u, v) \\
\Leftrightarrow &\sum_{(u,v)\in x_{max}} \log(r(u, v)) \geq \sum_{(u,v)\in x} \log(r(u, v)) \\
\Leftrightarrow &- \sum_{(u,v)\in x_{max}} \log(r(u, v)) \leq - \sum_{(u,v)\in x} \log(r(u, v)) \\
\Leftrightarrow &\sum_{(u,v)\in x_{max}} \log(\frac{1}{r(u, v)}) \leq \sum_{(u,v)\in x} \log(\frac{1}{r(u, v)})
\end{aligned}
$$

Durch Redefinition aller Kantengewichte zu $r'(u, v) := \log(\frac{1}{r(u,v)})$ findet Dijkstra somit den sichersten Pfad zwischen zwei Knoten. Beachte hierbei, dass dies nur gegeben ist, weil $0 < r(u, v) \leq 1$ gilt, und somit der Logarithmus des Kehrwertes immer positiv ist, alle Kantengewichte somit nicht-negativ sind.

Übung 4.
Eine Kante $e \in E$ eines zusammenhängenden, ungerichteten Graphen $G = (V, E)$ heißt Brückenkante, falls der Graph $G = (V, E \setminus e)$, der durch das Entfernen der Kante e aus G entsteht, nicht mehr zusammenhängend ist.

(a) Geben Sie einen Algorithmus in Pseudocode an, der in Laufzeit $\mathcal{O}(|V| \cdot |E| + |E|^2)$ entscheidet, ob ein ungerichteter Graph G (gegeben als Adjazenzliste) eine Brückenkante besitzt oder nicht.

(b) Zeigen Sie, dass der Algorithmus korrekt arbeitet.

(c) Zeigen Sie, dass der Algorithmus die Laufzeitschranke einhält.

Lösung 4.

(a)

```
func(G):
    s=SomeNode(G)
    for e ∈ G.E do
        G' = G(V, E \ e))
        BREITENSUCHE(G',s)
        for n ∈ G'.V do
            if n.COLOR!=SCHWARZ then
                return TRUE
            end
        end
    end
return FALSE
```

(b) Im obigen Algorithmus wird zuerst ein willkürlicher Knoten s ausgewählt. Es wird dann für jede Kante $e \in G$ überprüft ob es eine Brückenkante ist. Dafür wird ein neuer Graph G' erstellt welcher identisch zu G ist, außer dass er e nicht beinhaltet. Es wird dann eine Breitensuche von Knoten s gestartet und anschließend überprüft, ob ein Knoten in G' existiert welcher nicht Schwarz gefärbt wurde. Da die Breitensuche jeden Knoten welcher von s erreichbar ist, Schwarz einfärbt, lässt sich aus der Existenz eines solchen Knotens schließen, dass G' nicht zusammenhängend ist, also e eine Brückenkante war. In diesem Fall gibt der Algorithmus TRUE zurück. Nachdem alle Kanten so überprüft wurden, und nicht TRUE zurückgegeben wurde, folgt entsprechend, dass keine Brückenkante existiert und es wird entsprechend FALSE zurückgegeben.

(c) Die Auswahl eines Knotens ist in $\mathcal{O}(1)$ durchführbar. Die erste Schleife wird für jede Kante ausgeführt, also $|E|$ mal. Für das innere der Schleife gilt, dass die Breitensuche laut Vorlesung in $\mathcal{O}(|E| + |V|)$ durchführbar ist, die zweite Schleife kann in $\mathcal{O}(|V|)$ ausgeführt werden und das Erstellen von G' ist ebenfalls in $\mathcal{O}(|E| + |V|)$ machbar. Insgesamt ergiebt sich somit eine Laufzeit von $|E|\mathcal{O}(|E| + |V|) = \mathcal{O}(|V| \cdot |E| + |E|^2)$.