Lösung 10: Breiten- und Tiefensuche

Aufgabe 1: Breitensuche

a) Reihenfolge der Schwarzfärbung: 3 – 5 – 6 – 4 – 2 (die 1 wird nicht besucht)

Knoten	1	2	3	4	5	6
d	-	3	0	2	1	1
π	-	4	-	5	3	3

b) Reihenfolge der Schwarzfärbung: u – t – x – y- w – s – r- v

Knoten	r	S	t	u	v	w	X	y
d	4	3	1	0	5	2	1	1
π	s	W	u	-	r	T	U	u

c) Die Entfernung zwischen u und v kann höchstens u.d + v.d sein.

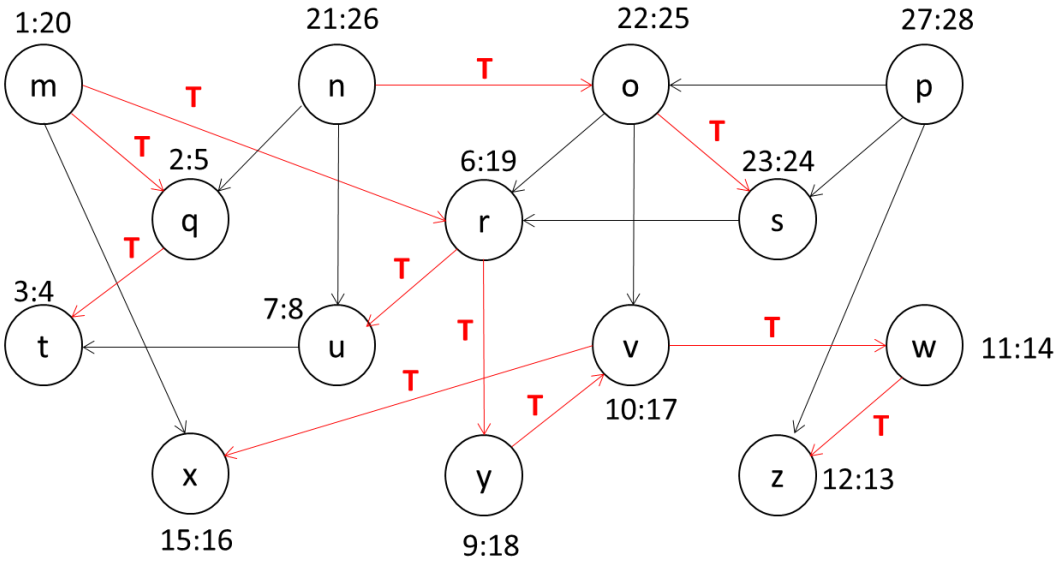
Aufgabe 2: Tiefensuche

a) Die Discovery und Finish Times sind in der unteren Zeichnung nach ähnlichem Schema wie in der Vorlesung eingetragen. Achtung: Der Graph ist nicht zusammenhängend.
Notation: <a>:<b> mit <a> als Discovery Time und <b> als Finish Time.

b) Die Knoten werden in der folgenden Reihenfolge schwarz gefärbt:
t – q – u – z – w – x - v – y – r – m – s – o – n - p

c) Die entsprechenden Kanten sind in der Abbildung rot markiert und mit „T“ beschriftet.

d) Für eine topologische Sortierung betrachtet man einfach die Finish Times. Die größte Finish Time muss am Anfang stehen. Somit ergibt sich genau die umgekehrte Sortierung wie in Aufgabe b), also: p – n – o – s – m – r – y – v – x - w – z – u – q – t
Hinweis: Für die topologische Sortierung gibt es übrigens meist mehrere mögliche Ergebnis, ebenso wie es je nach Sortierung der Adjazenzliste bei einer Tiefensuche unterschiedliche Besuchsreihenfolgen geben kann. Eine Anwendung der topologischen Sortierung: In welcher Reihenfolge muss man Aufgaben erledigen, damit es keine Konflikte gibt?



Aufgabe 3: Iterative Tiefensuche in Java

Lösung siehe Quelltext:

https://inf-git.fh-rosenheim.de/muwo522/ad_wise_2019/tree/master/src/de/th_rosenheim/ad/uebung10