ALP III: Datenstrukturen und Datenabstraktion
11. Aufgabenblatt
Übungsgruppe 1.8: Marcel Erhardt

Tobias Lohse/ Marvin Kleinert/ Anton Drewing

16.01.2015

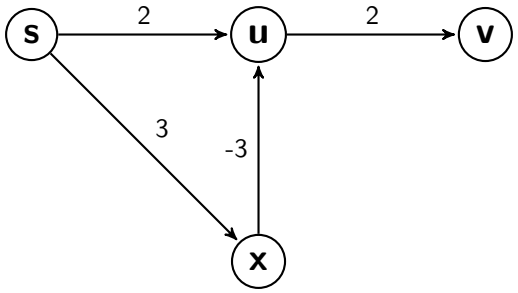
Aufgabe 1

a) Dijkstra

	s	u	v	x	y	z	Q	S
1	0	∞	∞	∞	∞	∞	s,u,v,x,y,z	$\emptyset$
2	0	3	∞	6	∞	∞	u,v,x,y,z	s
3	0	3	9	4	∞	11	v,x,y,z	s,u
4	0	3	9	4	5	11	v,y,z	s,u,x
5	0	3	9	4	5	11	v,z	s,u,x,y
6	0	3	9	4	5	10	z	s,u,x,y,v
7	0	3	9	4	5	10		s,u,x,y,v,z

b) negative Kantengewichte

Beispiel:



Bereits die Behauptung des Korrektheitsbeweises, dass nach jeder Iteration der while-Schleife gilt

$$\forall v \in S : D[v] = d(v)$$

trifft im Falle von negativen Kantengewichten nicht mehr zu. In unserem Beispiel befindet sich nach der zweiten Iteration u in S mit $D[u] = 2$, was ungleich $d(u) = 0$ ist.

Aufgabe 2

Um die schnellste Zugverbindung bei der Bahn zu finden, lässt sich folgendes graphentheoretisches Model konstruieren:

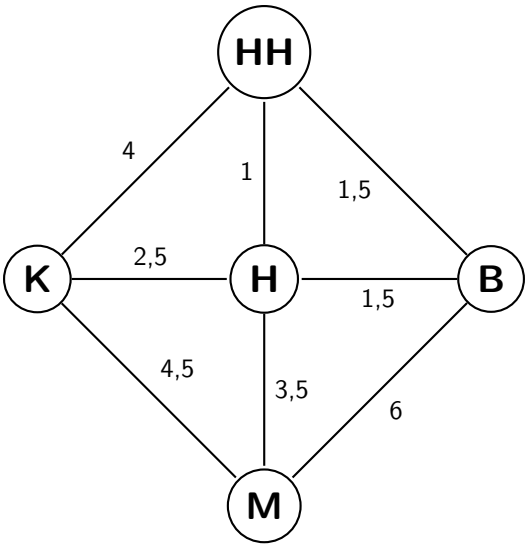
1. Zunächst wird ein Liniennetz als Graph erstellt:
   - Kanten $\widehat{=}$ Strecken
   - Kantengewichte $\widehat{=}$ Zeit für die Befahrung dieser Strecke

- Knoten $\widehat{=}$ Bahnhöfe mit abgespeicherten Abfahrtstabellen

2. Für jede Anfrage wird dann ein auf die entsprechende Anfrage angepasster Graph erzeugt:
   - Aus der Abfahrtsliste des Startbahnhofs werden Züge gemäß der gewünschten Abfahrtszeit ausgewählt und für jede Zugabfahrt ein Knoten mit Abfahrtszeit und Bahnhof erzeugt
   - Als Kanten werden die bereits existierenden Kanten des Liniennetzes verwendet.
   - An die Kanten schließen sich nun Knoten mit den Ankunftszeiten der Züge sowie dem jeweiligen Bahnhofsnamen an.
   - Unter Berücksichtigung von gewünschten Umsteigezeiten können jetzt wieder Züge aus der Abfahrtsliste ausgewählt und deren Abfahrtszeiten durch Knoten im Graphen dargestellt werden.
   - Die Kanten zwischen den Knoten des gleichen Bahnhofs einmal mit Ankunfts-, einmal mit Abfahrtszeit erhalten als Gewicht die Umsteigezeiten.
   - Die Erzeugung des Graphen wird abgebrochen, wenn eine bestimmte Ankunftszeit überschritten wird oder die Bahnhöfe aus einem bestimmten Radius vom Startbahnhof austreten.
3. Abschließend werden mit dem Algorithmus von Dijkstra alle kürzesten Wege vom Startknoten berechnet und der kürzeste Weg zum Zielbahnhof zurückgegeben.

Korrektheit:
Das Modell ist offensichtlich korrekt und löst das gestellte Problem, da mit jeder Anfrage ein Graph erzeugt wird, der beginnend bei der gewünschten Startzeit alle möglichen Strecken zum Zielbahnhof und anderen Bahnhöfen im Radius enthält, Umsteigezeiten berücksichtigt und mit Hilfe des Algorithmus von Dijkstra den kürzesten Weg mit der kürzesten Fahrtzeit für die gewünschte Strecke findet.

Beispiele:
Als Liniennetz verwenden wir ein vereinfachtes Netz der Deutschen Bahn:

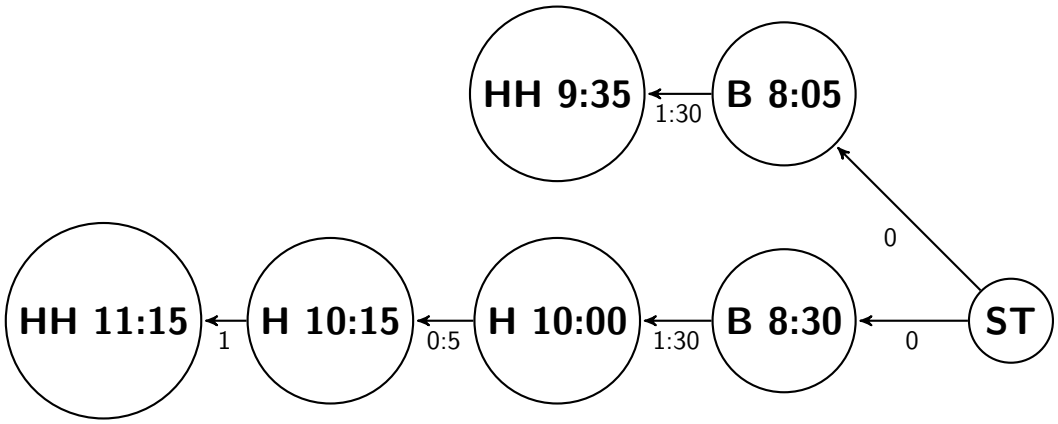


Für jden Bahnhof gibt es eine Tabelle der Abfahrtszeiten:

Berlin				Hannover	
Zugname	Abfahrt	Richtung	Zugname	Abfahrt	Richtung
ICE 1	7:50	M	ICE 3	10:05	K
ICE 2	8:05	HH	ICE 6	10:05	M
ICE 3	8:30	H	ICE 7	10:15	HH
ICE 4	8:50	M	ICE 8	10:35	M
ICE 5	10:00	H	ICE 5	11:35	HH

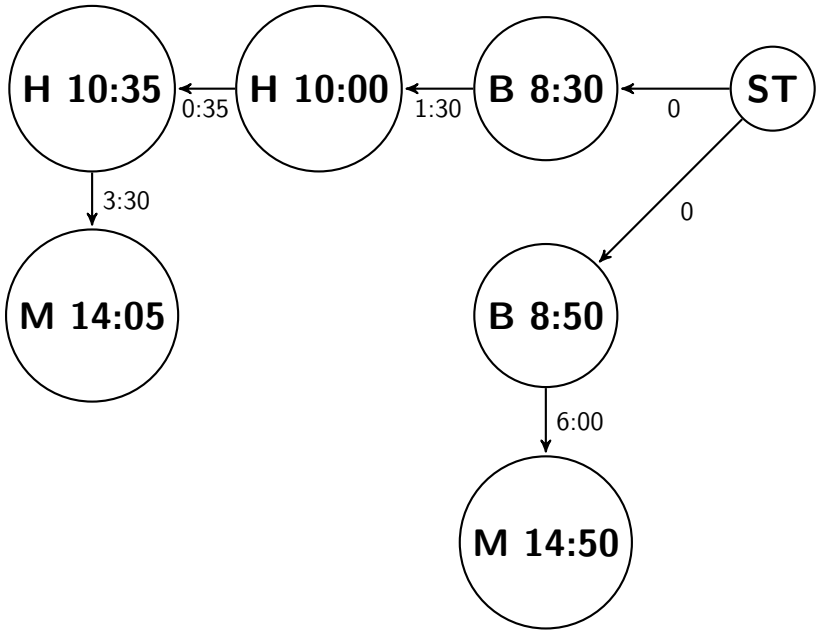
Als Beispiel dienen folgende Anfragen:

1. Strecke: B → HH Abfahrt: 8:00



   kürzestmögliche Reisedauer: 1:30

2. Strecke: B → M Abfahrt: 8:00 Umsteigezeit >= 30min



   kürzestmögliche Reisedauer: 5:35

Aufgabe 3

a) Varianten des Floyd-Warshall-Algorithmus

1. min $\rightarrow$ min und + $\rightarrow$ max
   Anwendungen:

   - Abhängig von der Tankgröße eines Autos soll eine minimale Route gewählt werden, bei der die Entfernung = Kantengewicht zwischen zwei Tankstellen = Knoten nicht die maximale Reichweite des Autos überschreitet.
   - Abhängig von den Fahr- und Ruhezeiten eines LKW-Fahrers soll eine zeitlich möglichst kurze Route gefunden werden, bei der die benötigten Fahrzeiten = Kantengewichten zwischen zwei Raststätten = Knoten berücksichtigt werden.

   Initialisierung: wie bei der ursprünglichen Variante; ∞ wird duch NaN ersetzt

2. min $\rightarrow$ max und + $\rightarrow$ min
   Anwendung: Abhängig vom Gewicht eines LKWs soll eine Route gefunden werden, bei der die maximalen Traglasten von Brücken = Kantengewichte maximal sind.
   Initialisierung: wie bei der ursprünglichen Variante; ∞ wird duch NaN ersetzt