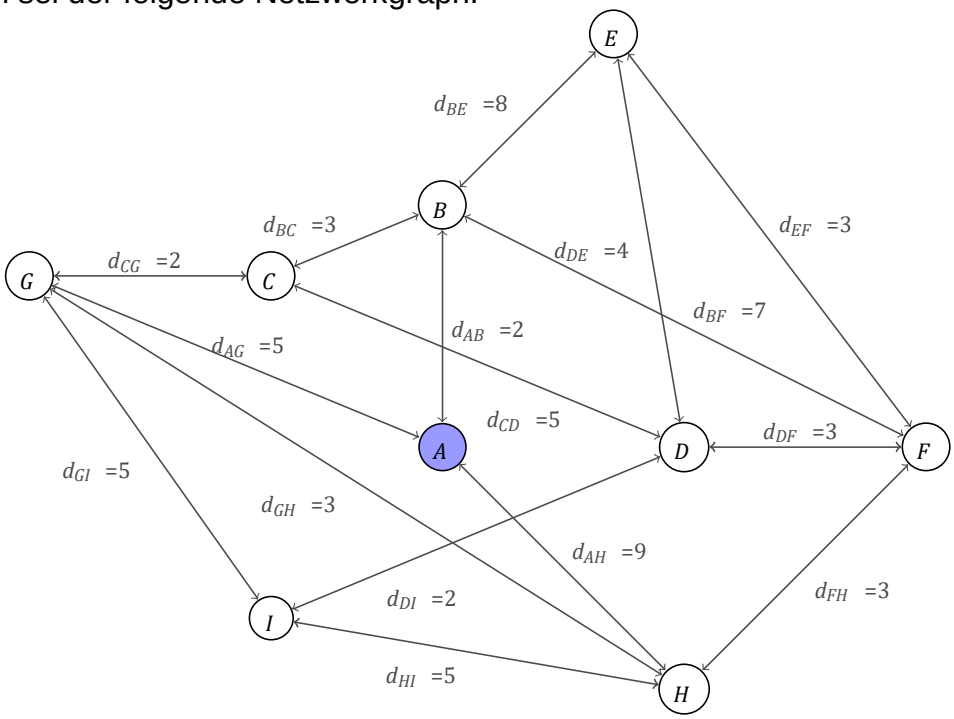
Übungsblatt 11 (Musterlösung)

1. Dijkstra-Algorithmus

Gegeben sei der folgende Netzwerkgraph:



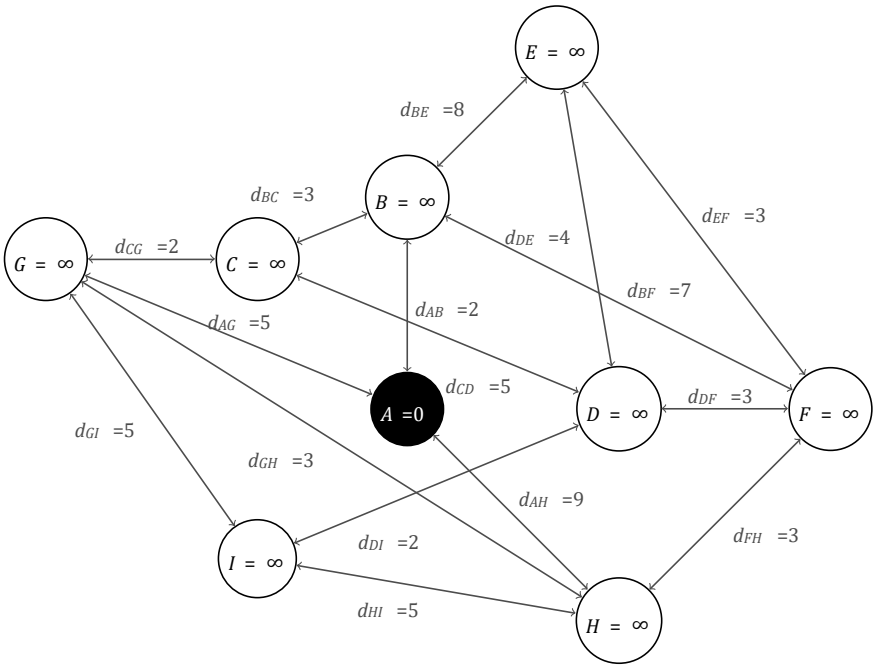
Berechnen Sie den kürzesten Pfad vom Knoten A zu allen anderen Knoten nach dem Dijkstra-Algorithmus. Vervollständigen Sie die unten angegebene Routingtabelle mit den gefundenen Pfaden und Kosten. Die Vorgehensweise muss erkennbar sein. Füllen Sie die folgende Tabelle aus:

Ziel	Pfad	Kosten
A	A	0
B	BA	2
C		
D		
E		
F		
G		
H		
I		

Tabelle 1: Kürzeste Distanzen

Aufgabe 2: Dijkstras Algorithmus (Lösung)

a) Im ersten Schritt (Initialisierung) wird der Startknoten mit Distanz null initialisiert. Alle anderen Knoten haben Distanz ∞. Der Startknoten kann direkt als schwarz markiert werden. Kein Knoten besitzt einen Vorgänger.

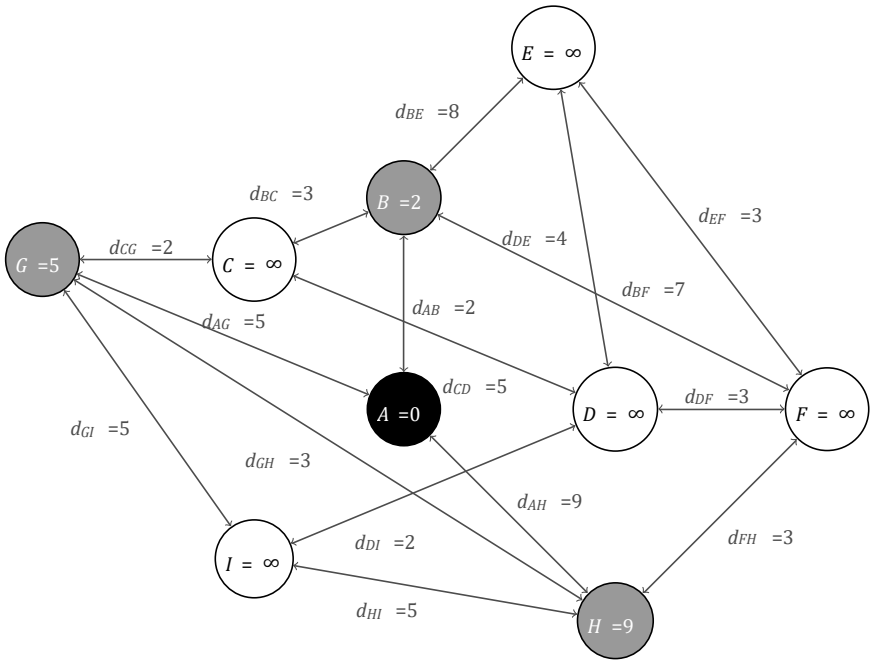


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$

Tabelle 2: Kürzeste Distanzen Lösung

b) Im zweiten Schritt werden die benachbarten Knoten zur grauen Menge hinzugefügt. Damit erhalten B, G und H A als Vorgänger.

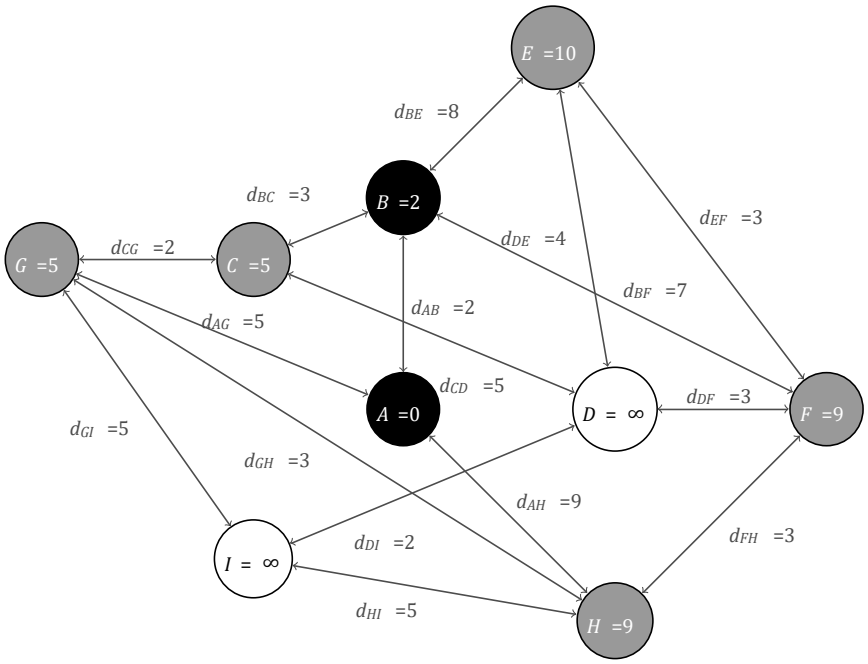


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$

Tabelle 3: Kürzeste Distanzen Lösung

c) Im dritten Schritt wird nun der Knoten mit dem kürzesten Pfad ausgew ahlt. In dem Fall ist das B. Dessen Nachbarn werden wieder grau markiert, wahrend B nun schwarz wird. Das heißt, es gibt keinen kurzeren Weg von A nach B. Nun werden die Distanzen aktualisiert. F, C und E bekommen B als Vorganger.

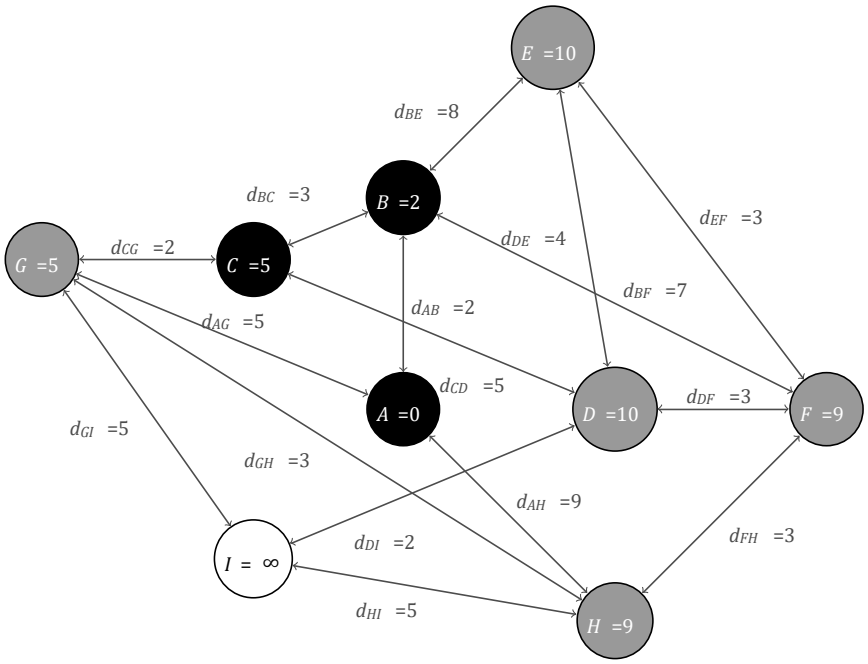


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$

Tabelle 4: Kürzeste Distanzen Lösung

d) Im vierten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Wir nehmen C. C wird damit schwarz. Alle weißen benachbarten Knoten von C werden grau und die Distanzen aktualisiert.
D bekommt C als Vorgänger.

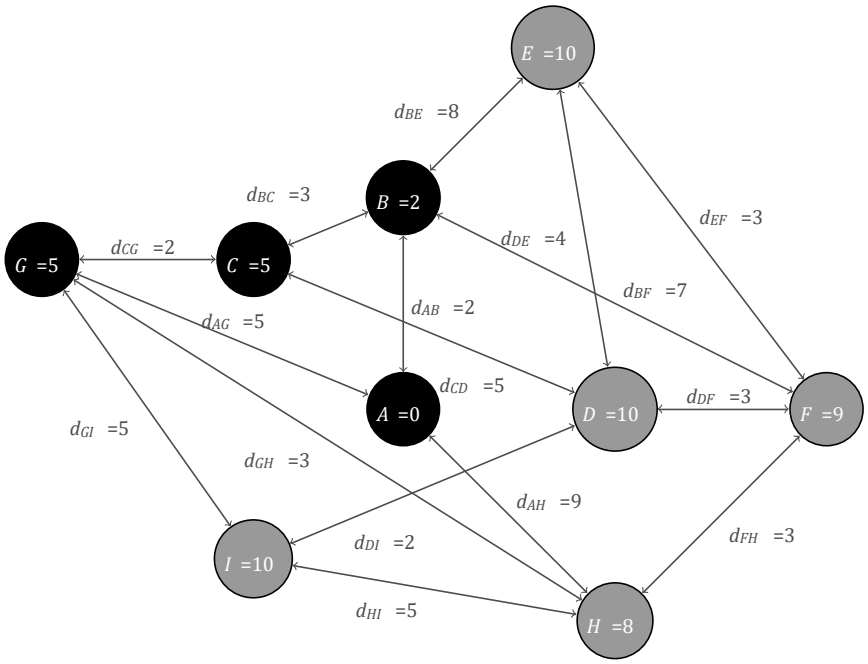


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$

Tabelle 5: Kürzeste Distanzen Lösung

e) Im fünften Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Dies ist G. G wird damit schwarz. Dann werden alle weißen benachbarten Knoten von G grau und die Distanzen aktualisiert. H und I bekommen G als Vorgänger.

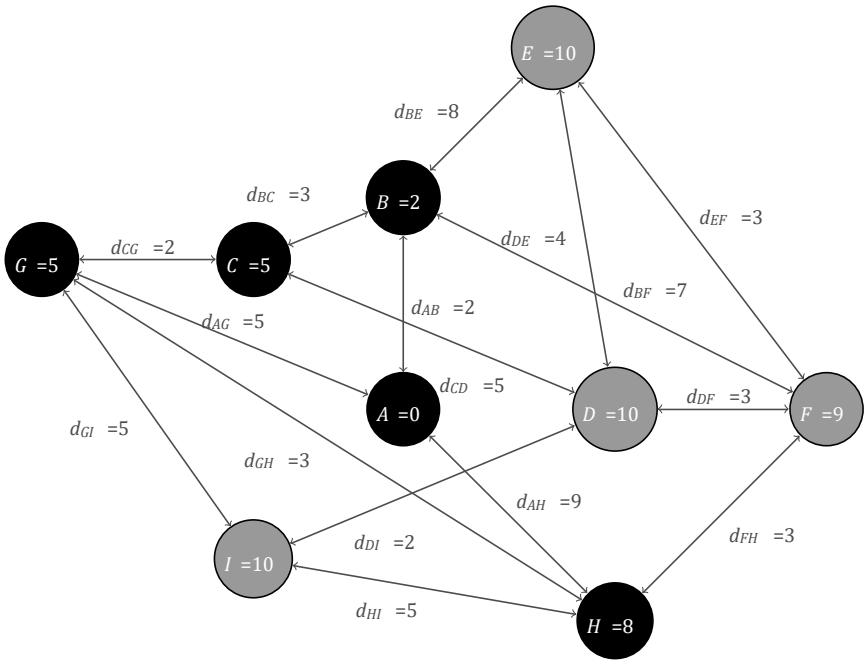


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
5.	$\{A,B,C,G\}$	$\emptyset$	$\{D,E,F,H,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$, $(H,8)$

Tabelle 6: Kürzeste Distanzen Lösung

f) Im sechsten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. H wird damit schwarz. Da es keine benachbarten weißen Knoten mehr gibt, werden nur die Distanzen aktualisiert.

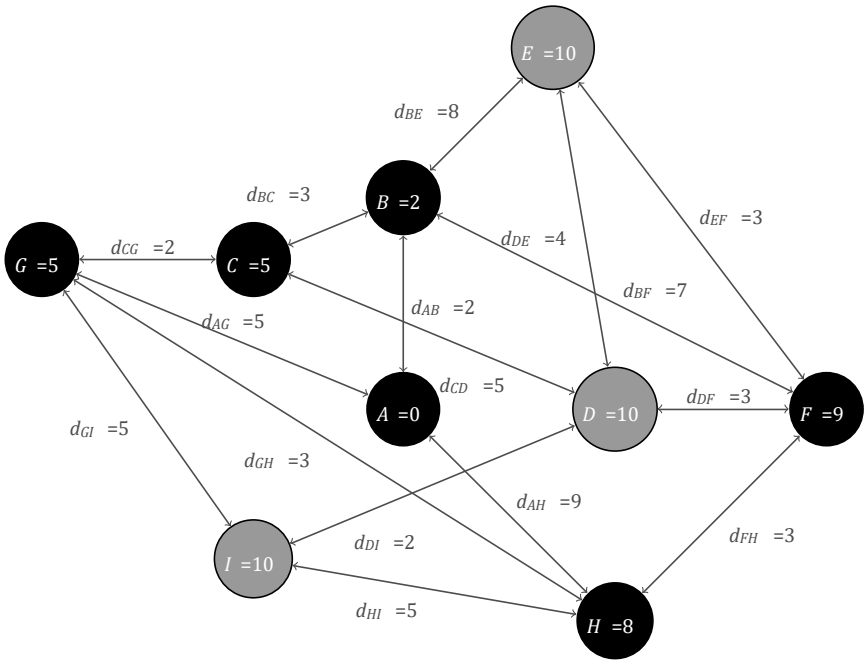


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
5.	$\{A,B,C,G\}$	$\emptyset$	$\{D,E,F,H,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$, $(H,8)$
6.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,F,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$

Tabelle 7: Kürzeste Distanzen Lösung

g) Im siebten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Dies ist F. F wird damit schwarz. Da es keine benachbarten weißen Knoten mehr gibt, werden nur die Distanzen aktualisiert.

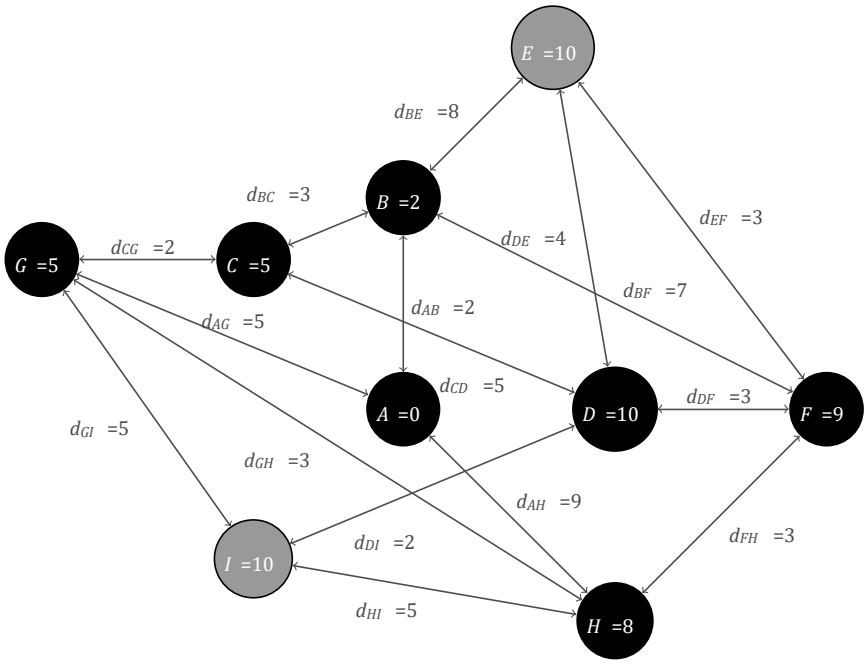


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
5.	$\{A,B,C,G\}$	$\emptyset$	$\{D,E,F,H,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$, $(H,8)$
6.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,F,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$
7.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,I\}$	$(D,10)$, $(E,10)$, $(I,10)$

Tabelle 8: Kürzeste Distanzen Lösung

h) Im achten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Dies sind D, E, I. Wir wählen D. D wird damit schwarz. Da es keine benachbarten weißen Knoten mehr gibt, werden nur die Distanzen aktualisiert.

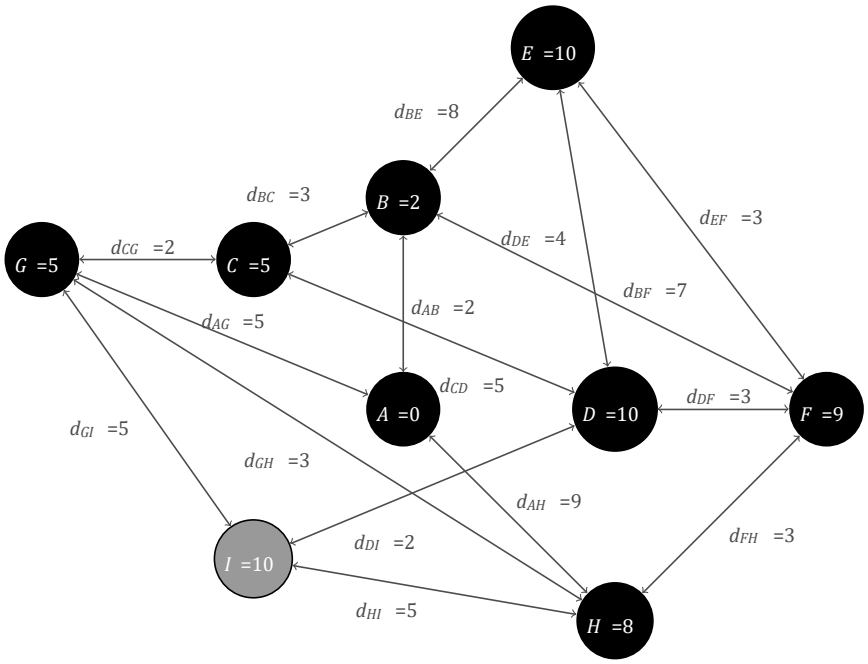


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
5.	$\{A,B,C,G\}$	$\emptyset$	$\{D,E,F,H,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$, $(H,8)$
6.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,F,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$
7.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,I\}$	$(D,10)$, $(E,10)$, $(I,10)$
8.	$\{A,B,C,D,G,F,H\}$	$\emptyset$	$\{E,I\}$	$(E,10)$, $(I,10)$

Tabelle 9: Kürzeste Distanzen Lösung

i) Im neunten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Dies sind E und I. Wir wählen E. E wird damit schwarz. Da es keine benachbarten weißen Knoten mehr gibt, werden nur die Distanzen aktualisiert.

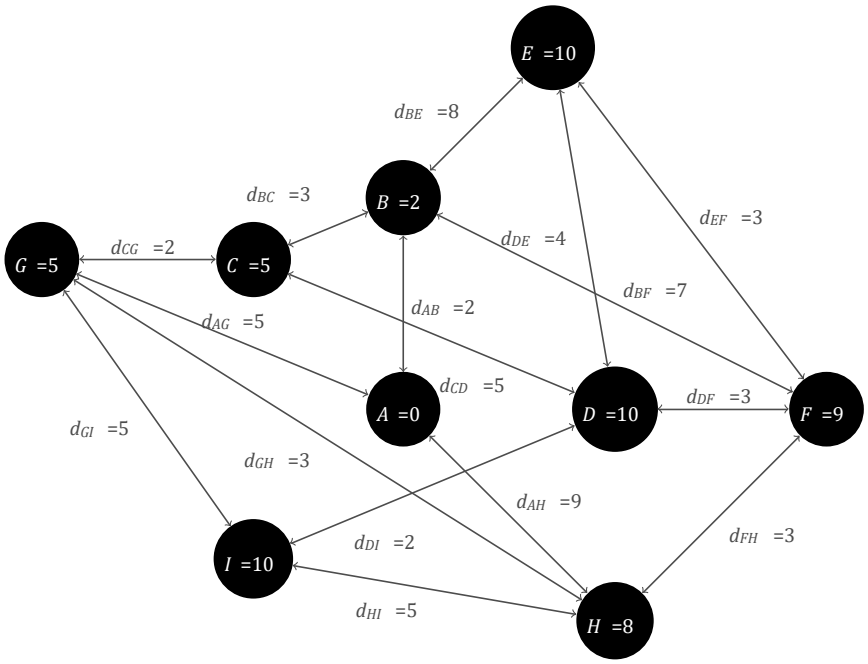


In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	$\{A\}$	$\{B,C,D,E,F,G,H,I\}$	$\emptyset$	$\emptyset$
2.	$\{A\}$	$\{C,D,E,F,I\}$	$\{B,H,G\}$	$(B,2)$, $(G,5)$, $(H,9)$
3.	$\{A,B\}$	$\{D,I\}$	$\{C,E,F,G,H\}$	$(C,5)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
4.	$\{A,B,C\}$	$\{I\}$	$\{D,E,F,G,H\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(G,5)$, $(H,9)$
5.	$\{A,B,C,G\}$	$\emptyset$	$\{D,E,F,H,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$, $(H,8)$
6.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,F,I\}$	$(D,10)$, $(E,10)$, $(F,9)$, $(I,10)$
7.	$\{A,B,C,G,F,H\}$	$\emptyset$	$\{D,E,I\}$	$(D,10)$, $(E,10)$, $(I,10)$
8.	$\{A,B,C,D,G,F,H\}$	$\emptyset$	$\{E,I\}$	$(E,10)$, $(I,10)$
9.	$\{A,B,C,D,E,G,F,H\}$	$\emptyset$	$\{I\}$	$(I,10)$

Tabelle 10: Kürzeste Distanzen Lösung

j) Im zehnten Schritt wird wieder aus der grauen Menge der Knoten mit der niedrigsten Gesamtdistanz ausgewählt. Dies ist I. I wird damit schwarz. Da es keine benachbarten weißen Knoten mehr gibt, werden nur die Distanzen aktualisiert.



In Tabellenform gilt somit:

Schritt	Schwarz	Weiß	Grau	Distanz(en)
1.	{A}	{B,C,D,E,F,G,H,I}	∅	∅
2.	{A}	{C,D,E,F,I}	{B,H,G}	(B,2), (G,5), (H,9)
3.	{A,B}	{D,I}	{C,E,F,G,H}	(C,5), (E,10), (F,9), (G,5), (H,9)
4.	{A,B,C}	{I}	{D,E,F,G,H}	(D,10), (E,10), (F,9), (G,5), (H,9)
5.	{A,B,C,G}	∅	{D,E,F,H,I}	(D,10), (E,10), (F,9), (I,10), (H,8)
6.	{A,B,C,G,F,H}	∅	{D,E,F,I}	(D,10), (E,10), (F,9), (I,10)
7.	{A,B,C,G,F,H}	∅	{D,E,I}	(D,10), (E,10), (I,10)
8.	{A,B,C,D,G,F,H}	∅	{E,I}	(E,10), (I,10)
9.	{A,B,C,D,E,G,F,H}	∅	{I}	(I,10)
10.	{A,B,C,D,E,G,F,H,I}	∅	∅	∅

Tabelle 11: Kürzeste Distanzen Lösung

Die Lösungen kann man nun einfach von den Vorgängern ablesen. Alternativ kann man dies natürlich schon beim Algorithmus selber machen, wenn man den Weg wissen will.

Ziel	Pfad	Kosten
A	A	0
B	$B \rightarrow A$	2
C	$C \rightarrow B \rightarrow A$	5
D	$D \rightarrow C \rightarrow B \rightarrow A$	10
E	$E \rightarrow B \rightarrow A$	10
F	$F \rightarrow B \rightarrow A$	9
G	$G \rightarrow A$	5
H	$H \rightarrow G \rightarrow A$	8
I	$I \rightarrow G \rightarrow A$	10

Tabelle 12: Kürzeste Distanzen

2. Distance-Vector-Algorithmus

Gegeben sei folgende Topologie von Routern:

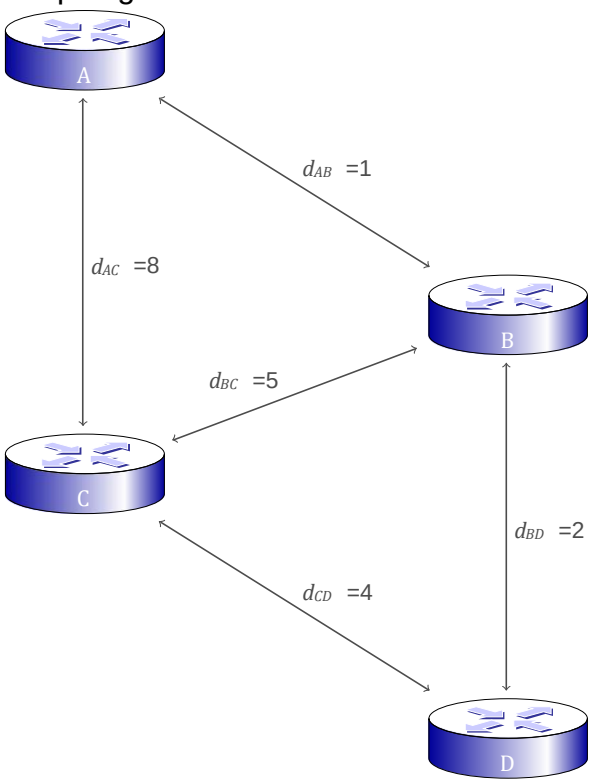


Abbildung 1: Routingtopologie

Die Kantengewichte geben hierbei die jeweiligen Linkkosten an. Beantworten Sie nun folgende Aufgabenstellungen:

a) Ermitteln Sie mittels Distance-Vector-Algorithmus die kürzesten Pfade zwischen allen Routern. Berücksichtigen Sie dabei jeden Zwischenschritt.

b) Beschreiben Sie das “Count-to-Infinity” Problem. Wann tritt dieses Problem auf?

c) Nehmen Sie nun in Abb. 1 an, dass die Kante AB unterbrochen wird und damit nicht mehr zur Verfügung steht. Wie wirkt sich das “Count-to-Infinity” Problem in diesem Beispiel aus? Nach wie vielen Schritten hat sich die Situation erneut stabilisiert?

d) Wie wurde sich die Situation ändern, wenn Poisoned-Reverse verwendet wurde?

Aufgabe 3: Distance Vector Algorithmen (Lösung)

a) Wir gehen hier wieder Schritt für Schritt vor:

- Im ersten Schritt initialisieren wir alles, indem wir die Distanzen alle auf ∞ setzen. Des Weiteren können die Distanzen zu den direkten Nachbarn direkt eingetragen werden. Diese Zeile wird danach an alle direkten Nachbarn gesendet. (A sendet also seine Sicht an B und C, während C seine Sicht an A, B und D übermittelt). Die Zeilen, in denen “von” und “nach” identisch sind, sind als lokale Werte zu verstehen.

	Zeit	1	2	3
bei	nach / von	$A\ B\ C\ D$	$A\ B\ C\ D$	$A\ B\ C\ D$
A	A	0 1 8 ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	B	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
B	A	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	B	1 0 5 2	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	D	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
C	A	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	B	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	C	8 5 0 4	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	D	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
D	B	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞
	D	∞ 2 4 0	∞ ∞ ∞ ∞	∞ ∞ ∞ ∞

- Im zweiten Schritt sind die Informationen der Nachbarn angekommen. Nun schaut jeder Knoten, ob es über den Nachbarn schneller geht. Offensichtlich wird nur die kürzeste mögliche Route übernommen. Diese Information wird dann wieder an alle Nachbarn übermittelt.

	Zeit	1	2	3
bei	nach / von	$A\ B\ C\ D$	$A\ B\ C\ D$	$A\ B\ C\ D$
A	A	0 1 8 ∞	0 1 6 3	∞ ∞ ∞ ∞
	B	∞ ∞ ∞ ∞	1 0 5 2	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	8 5 0 4	∞ ∞ ∞ ∞
B	A	∞ ∞ ∞ ∞	0 1 8 ∞	∞ ∞ ∞ ∞
	B	1 0 5 2	1 0 5 2	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	8 5 0 4	∞ ∞ ∞ ∞
	D	∞ ∞ ∞ ∞	∞ 2 4 0	∞ ∞ ∞ ∞
C	A	∞ ∞ ∞ ∞	0 1 8 ∞	∞ ∞ ∞ ∞
	B	∞ ∞ ∞ ∞	1 0 5 2	∞ ∞ ∞ ∞
	C	8 5 0 4	6 5 0 4	∞ ∞ ∞ ∞
	D	∞ ∞ ∞ ∞	∞ 2 4 0	∞ ∞ ∞ ∞
D	B	∞ ∞ ∞ ∞	1 0 5 2	∞ ∞ ∞ ∞
	C	∞ ∞ ∞ ∞	8 5 0 4	∞ ∞ ∞ ∞
	D	∞ 2 4 0	3 2 4 0	∞ ∞ ∞ ∞

- Im dritten Schritt sind die neuen Informationen der Nachbarn angekommen. Nun schaut jeder Knoten, ob es über den Nachbarn schneller geht. Offensichtlich wird wieder nur die kürzeste mögliche Route übernommen. Diese Information wird dann wieder an alle Nachbarn übermittelt.

	Zeit	1	2	3
bei	nach / von	$A\ B\ C\ D$	$A\ B\ C\ D$	$A\ B\ C\ D$
A	A	0 1 8 ∞	0 1 6 3	0 1 6 3
	B	∞ ∞ ∞ ∞	1 0 5 2	1 0 5 2
	C	∞ ∞ ∞ ∞	8 5 0 4	6 5 0 4
B	A	∞ ∞ ∞ ∞	0 1 8 ∞	0 1 6 3
	B	1 0 5 2	1 0 5 2	1 0 5 2
	C	∞ ∞ ∞ ∞	8 5 0 4	6 5 0 4
	D	∞ ∞ ∞ ∞	∞ 2 4 0	3 2 4 0
C	A	∞ ∞ ∞ ∞	0 1 8 ∞	0 1 6 3
	B	∞ ∞ ∞ ∞	1 0 5 2	1 0 5 2
	C	8 5 0 4	6 5 0 4	6 5 0 4
	D	∞ ∞ ∞ ∞	∞ 2 4 0	3 2 4 0
D	B	∞ ∞ ∞ ∞	1 0 5 2	1 0 5 2
	C	∞ ∞ ∞ ∞	8 5 0 4	6 5 0 4
	D	∞ 2 4 0	3 2 4 0	3 2 4 0

b) Das “Count-to-Infinity” Problem tritt auf, wenn ein Link im Netzwerk wegfallt oder sich seine Kosten deutlich verschlechtern. In diesem Fall kann es passieren, dass eine

Gruppe von Routern sukzessive untereinander Routen anpreist, ohne dabei zu erkennen dass die günstige Route bereits nicht mehr existiert.

c) Am Beispiel der Router B und D:

   a) B weiß, dass er A nicht mehr direkt erreichen kann, da der Link AB weggefallen ist. Er weiß aber auch, dass D eine Route mit Lange 3 kennt. Die Distanz zu D ist 2, also setzt B seine eigene Route zu A im nächsten Schritt auf 3 + 2 = 5 (beachte, dass die Route über C nach A mit Kosten von 13 deutlich teurer ist).

   b) D erhalt nun die neue Route von B und muss seine eigene Distanz zu A anpassen. Die vorherige Route führte ja über B, welcher aber nun statt einer Route mit Kosten 1 die Route mit Kosten 5 anpreist. D setzt also seine eigene Route auf 5 + 2 = 7, was immer noch günstiger ist als die Route via C (4 + 8 = 12).

   c) Nun muss wiederum B seine Route anpassen. Er addiert erneut die Kosten des Links BD auf die Route via D: 7 + 2 = 9.

   d) Auch D passt anschließend noch einmal seine Route an: 9 + 2 = 11.

   e) Im nächsten Schritt passt B erneut seine Route an und kennt nun zwei Routen nach A, welche beide Kosten von 13 aufweisen (via C und - fälschlicherweise - via D).

   f) D passt seine Route ebenfalls erneut an, setzt aber nun die korrekte Route via C, welche mit 12 günstiger ist als die vermeintliche Route über B mit aktuellen Kosten von 15.

   g) B erkennt nun, dass die Route über C die günstigere ist (Kosten von 13 anstelle vermeintlicher Kosten von 17) und nimmt zukünftig diese Route. Damit hat sich die Situation wieder stabilisiert (C und A terminieren beide schneller).

d) Poisoned-Reverse wurde das oben geschilderte Problem zwar nicht vollständig losen, aber die Routenfindung beschleunigen. In diesem Fall hatte D die Route nach A fur B mit Kosten von ∞ und für C mit Kosten von 3 ausgezeichnet. Nach Ausfall des Links kennt B zunächst nur noch Routen mit Kosten ∞ zu A (sowohl C als auch D hatten über B geroutet). C wurde im nächsten Schritt über D senden (Kosten: 4 + 2 + 1 = 7, wohingegen D nur noch über C senden kann (und daher seine Route an C mit ∞ auszeichnet). Einen Schritt weiter wählt nun C korrekterweise die direkte Route nach A. Im darauffolgenden Schritt passen auch B und D ihre Routen an und die Situation hat sich wieder stabilisiert.

Viel Erfolg !!!