Shortest path problem MmF

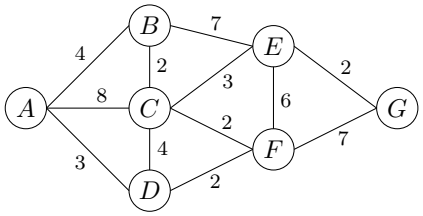
Dein Routenplaner soll den schnellsten Weg von einem Ort A zu einem anderen Ort G ermitteln.
Die Verbindungen benachbarter Orte sind rechts in einem **kantengewichteten Graphen** dargestellt:

In der Graphentheorie nennen wir ...

... die Orte A, B, C, D, E, F, G auch **Knoten**.

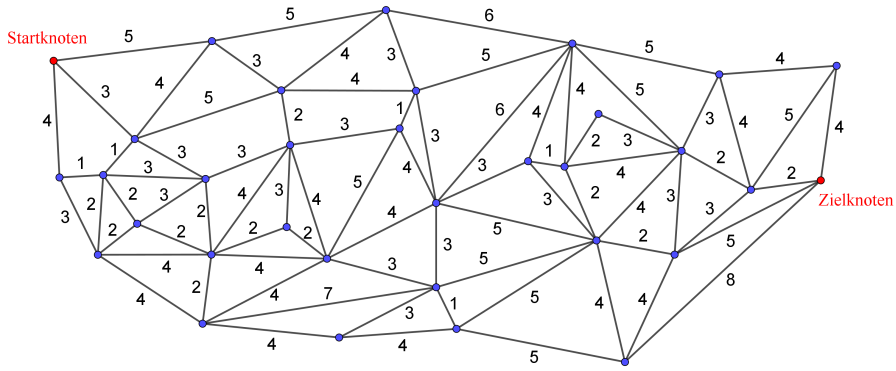
... die Strecken zwischen 2 Knoten auch **Kanten**.

... die Zahlen neben den Kanten auch **Gewichte**.



Die positiven Gewichte geben in diesem Kontext die Fahrzeit zwischen den Orten an.
Ein möglicher Weg von A nach G ist $A \xrightarrow{3} D \xrightarrow{2} F \xrightarrow{7} G$ mit Fahrzeit $3 + 2 + 7 = 12$.

Findest du einen schnelleren Weg? Hast du *sicher* den schnellsten Weg gefunden?



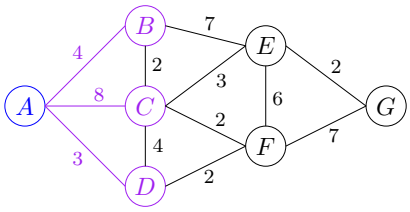
In der Praxis sind die Graphen deutlich größer.

Gesucht ist ein Verfahren, mit dem wir effizient einen kürzesten Weg finden.

Dijkstra-Algorithmus (1959) MmF

Der **Dijkstra-Algorithmus** findet in jedem kantengewichteten Graphen mit positiven Gewichten ausgehend von einem Startknoten den jeweils **schnellsten Weg** zu allen anderen Knoten.
Wenn der Graph n Knoten enthält, dann benötigt der Algorithmus dafür n Durchläufe.
In jedem Durchlauf finden wir den schnellsten Weg vom Startknoten zu einem neuen Knoten:

1) In Durchlauf 1 finden wir den schnellsten Weg von A nach A. Alle Kantengewichte sind positiv.
Die Fahrzeit 0 speichern wir in der folgenden Tabelle ab.
Außerdem tragen wir die jeweiligen Fahrzeiten zu den benachbarten Knoten B, C und D ein.



	A	B	C	D	E	F	G
Durchlauf 1	0	4	8	3	∞	∞	∞

Zu den anderen Knoten haben wir noch keinen Weg gefunden.
Die Fahrzeit ist also noch „unendlich“ (∞) groß.

2) Dann suchen wir in der Tabelle unter den verbleibenden Knoten B, C, ..., G jenen Knoten mit der *kleinsten* Fahrzeit. Nach Durchlauf 1 ist das Knoten D mit Fahrzeit 3.

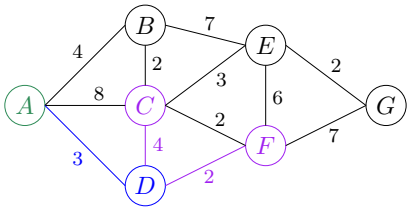
Kürzester Weg MmF

Warum kann kein Weg ausgehend von A über B oder C mit kleinerer Fahrzeit als 3 nach D führen?

Dijkstra-Algorithmus – Durchlauf 2

2) In Durchlauf 2 finden wir also den schnellsten Weg von A nach D. Er hat Fahrzeit 3.
Dann prüfen wir, ob es über D einen schnelleren Weg zu seinen benachbarten Knoten gibt:

$C: 3 + 4 = 7 < 8 \checkmark$ $F: 3 + 2 = 5 < \infty \checkmark$ Zu A haben wir bereits den schnellsten Weg gefunden.



Die Ergebnisse speichern wir in der Tabelle ab:

	A	B	C	D	E	F	G
Durchlauf 1	0	4	8	3	∞	∞	∞
Durchlauf 2	0	4	7	3	∞	5	∞

3) Dann suchen wir in der Tabelle unter den verbleibenden Knoten B, C, E, F und G jenen Knoten mit der *kleinsten* Fahrzeit. Nach Durchlauf 2 ist das Knoten B mit Fahrzeit 4.

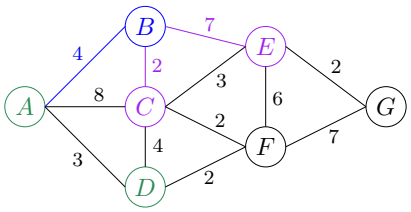
Kürzester Weg – Durchlauf 2

Warum kann kein Weg ausgehend von A mit kleinerer Fahrzeit als 4 nach B führen?

Dijkstra-Algorithmus – Durchlauf 3

3) In Durchlauf 3 finden wir also den schnellsten Weg von A nach B. Er hat Fahrzeit 4.
Dann prüfen wir, ob es über B einen schnelleren Weg zu seinen benachbarten Knoten gibt:

$C: 4 + 2 = 6 < 7 \checkmark$ $E: 4 + 7 = 11 < \infty \checkmark$ Zu A haben wir bereits den schnellsten Weg gefunden.



Die Ergebnisse speichern wir in der Tabelle ab:

	A	B	C	D	E	F	G
Durchlauf 2	0	4	7	3	∞	5	∞
Durchlauf 3	0	4	6	3	11	5	∞

4) Dann suchen wir in der Tabelle unter den verbleibenden Knoten C, E, F und G jenen Knoten mit der *kleinsten* Fahrzeit. Nach Durchlauf 3 ist das Knoten F mit Fahrzeit 5.

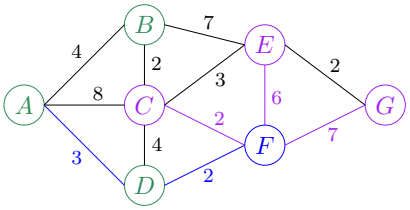
Kürzester Weg – Durchlauf 3

Warum kann kein Weg ausgehend von A mit kleinerer Fahrzeit als 5 nach F führen?

4) In Durchlauf 4 finden wir also den schnellsten Weg von A nach F. Er hat Fahrzeit 5.
Dann prüfen wir, ob es über F einen schnelleren Weg zu seinen benachbarten Knoten gibt:

$C:\ 5+2=7\geq 6$ ✗ $\quad E:\ 5+6=11\geq 11$ ✗ $\quad G:\ 5+7=12<\infty$ ✓

Zu D haben wir bereits den schnellsten Weg gefunden.



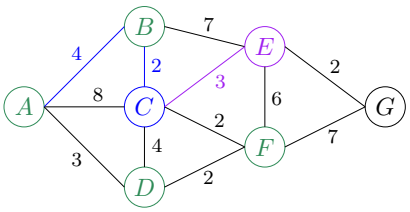
Die Ergebnisse speichern wir in der Tabelle ab:

	A	B	C	D	E	F	G
Durchlauf 3	0	4	6	3	11	5	∞
Durchlauf 4	0	4	6	3	11	5	12

5) Dann suchen wir in der Tabelle unter den verbleibenden Knoten C, E und G jenen Knoten mit der *kleinsten* Fahrzeit. Nach Durchlauf 4 ist das Knoten C mit Fahrzeit 6.
In Durchlauf 5 finden wir also den schnellsten Weg von A nach C. Er hat Fahrzeit 6.
Dann prüfen wir, ob es über C einen schnelleren Weg zu seinen benachbarten Knoten gibt:

$E:\ 6+3=9<11$ ✓

Zu A, B, D und F haben wir bereits den schnellsten Weg gefunden.



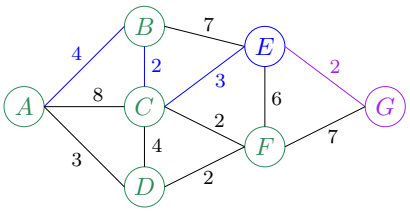
Die Ergebnisse speichern wir in der Tabelle ab:

	A	B	C	D	E	F	G
Durchlauf 4	0	4	6	3	11	5	12
Durchlauf 5	0	4	6	3	9	5	12

6) Dann suchen wir in der Tabelle unter den verbleibenden Knoten E und G jenen Knoten mit der *kleinsten* Fahrzeit. Nach Durchlauf 5 ist das Knoten E mit Fahrzeit 9.
In Durchlauf 6 finden wir also den schnellsten Weg von A nach E. Er hat Fahrzeit 9.
Dann prüfen wir, ob es über E einen schnelleren Weg zu seinen benachbarten Knoten gibt:

$G:\ 9+2=11<12$ ✓

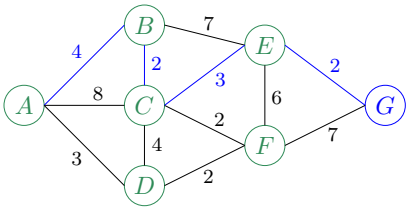
Zu B, C und F haben wir bereits den schnellsten Weg gefunden.



Die Ergebnisse speichern wir in der Tabelle ab:

	A	B	C	D	E	F	G
Durchlauf 5	0	4	6	3	9	5	12
Durchlauf 6	0	4	6	3	9	5	11

7) Schließlich finden wir in Durchlauf 7 den schnellsten Weg von A nach G mit Fahrzeit 11:



	A	B	C	D	E	F	G
Durchlauf 6	0	4	6	3	9	5	11
Durchlauf 7	0	4	6	3	9	5	11

Der schnellste Weg von A nach G ist also $A \xrightarrow{4} B \xrightarrow{2} C \xrightarrow{3} E \xrightarrow{2} G$ mit Fahrzeit 11.

Der Dijkstra-Algorithmus findet vom Startknoten den schnellsten Weg zu *jedem* anderen Knoten.
Wenn nur der schnellste Weg zu einem bestimmten Knoten gesucht ist, kann man auch nach dem entsprechenden Durchlauf abbrechen.

Rückverfolgung

In der folgenden Tabelle ist das Ergebnis des Dijkstra-Algorithmus vollständig dargestellt.
In jeder Zeile ist in blauer Farbe markiert, zu welchem Knoten der kürzeste Weg gefunden werde.
Wir können nur mithilfe der Tabelle den schnellsten Weg von A nach G rückverfolgen:

	A	B	C	D	E	F	G
Durchlauf 1	0	4	8	3	∞	∞	∞
Durchlauf 2	0	4	7	3	∞	5	∞
Durchlauf 3	0	4	6	3	11	5	∞
Durchlauf 4	0	4	6	3	11	5	12
Durchlauf 5	0	4	6	3	9	5	12
Durchlauf 6	0	4	6	3	9	5	11
Durchlauf 7	0	4	6	3	9	5	11

In Spalte G suchen wir das Update auf 11.
Nach G sind wir also von ____ gekommen.

In Spalte E suchen wir das Update auf 9.
Nach E sind wir also von ____ gekommen.

In Spalte C suchen wir das auf Update 6.
Nach C sind wir also von ____ gekommen.

In Spalte B suchen wir das auf Update 4.
Nach B sind wir also von ____ gekommen.

Der schnellste Weg von A nach G ist also $A \xrightarrow{4} B \xrightarrow{2} C \xrightarrow{3} E \xrightarrow{2} G$ mit Fahrzeit 11.

Vorgänger merken

Bei großen Graphen können wir Speicherplatz sparen, indem wir nicht die komplette Tabelle speichern.
Stattdessen merken wir uns bei jedem Update, von welchem Knoten der neue schnellere Weg kommt:

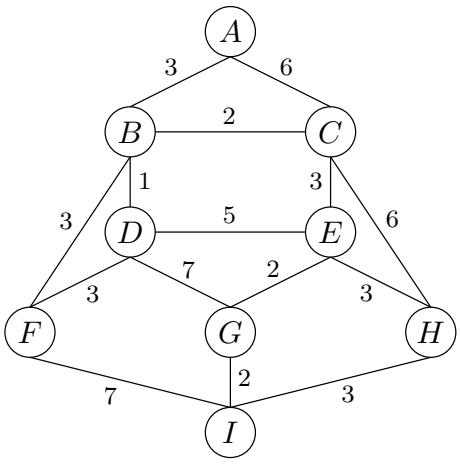
	A	B	C	D	E	F	G
Durchlauf 1	0	4	8	3	∞	∞	∞
Durchlauf 2	0	4	7	3	∞	5	∞
Durchlauf 3	0	4	6	3	11	5	∞
Durchlauf 4	0	4	6	3	11	5	12
Durchlauf 5	0	4	6	3	9	5	12
Durchlauf 6	0	4	6	3	9	5	11
Durchlauf 7	0	4	6	3	9	5	11

Vorgänger	A	B	C	D	E	F	G
Durchlauf 1	–	A	A	A	–	–	–
Durchlauf 2	–	A	D	A	–	D	–
Durchlauf 3	–	A	B	A	B	D	–
Durchlauf 4	–	A	B	A	B	D	F
Durchlauf 5	–	A	B	A	C	D	F
Durchlauf 6	–	A	B	A	C	D	E
Durchlauf 7	–	A	B	A	C	D	E

Erkläre, wie du nur mithilfe der letzten Zeile den schnellsten Weg von A nach G rückverfolgen kannst:

Dijkstra-Algorithmus

Ermittle mit dem Dijkstra-Algorithmus den schnellsten Weg von A nach I.



	A	B	C	D	E	F	G	H	I
Durchlauf 1									
Durchlauf 2									
Durchlauf 3									
Durchlauf 4									
Durchlauf 5									
Durchlauf 6									
Durchlauf 7									
Durchlauf 8									
Durchlauf 9									

Der schnellste Weg von A nach I ist also ____ mit Fahrzeit ____.