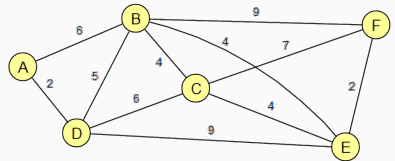
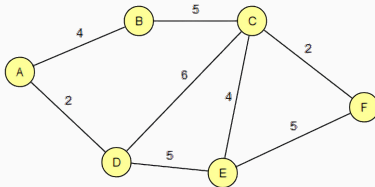


Kürzeste Pfade

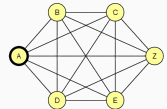
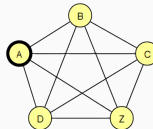
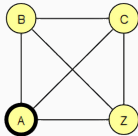
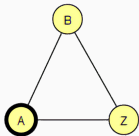
Aufgabe 1

- Finde in jedem der beiden Graphen den kürzesten Weg von A nach F
 - Wie kann man sicher sein, wirklich den kürzesten gefunden zu haben?



Aufgabe 2

- Bestimme jeweils die Anzahl an Wegen von A nach Z, die keinen Knoten doppelt enthalten (aber nicht alle enthalten müssen)



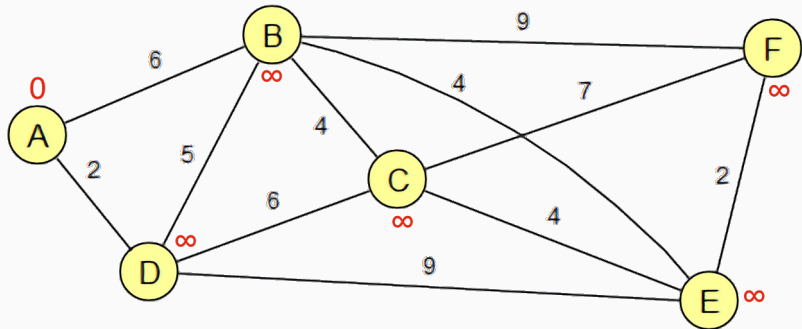
- Bei kleinen Graphen kann man ggf. noch alle möglichen Pfade durchprobieren und die jeweiligen Kosten bestimmen
- Ab einer gewissen Größe des Graphen ist dies nicht mehr sinnvoll möglich
 - Wir brauchen einen besseren Algorithmus

- Berechnet ausgehend von einem wählbaren Startknoten die kürzesten Wege zu allen anderen Knoten
- Funktioniert auf beliebigen, kantengewichteten Graphen, solange die Gewichte positiv sind
- Für jeden Knoten werden zwei Attribute gespeichert
 - Distanz: Kürzeste bisher bekannte Entfernung zum Startknoten
 - Vorgänger: Verweis auf den Knoten, über den der kürzeste bisher bekannte Weg zum Startknoten führt

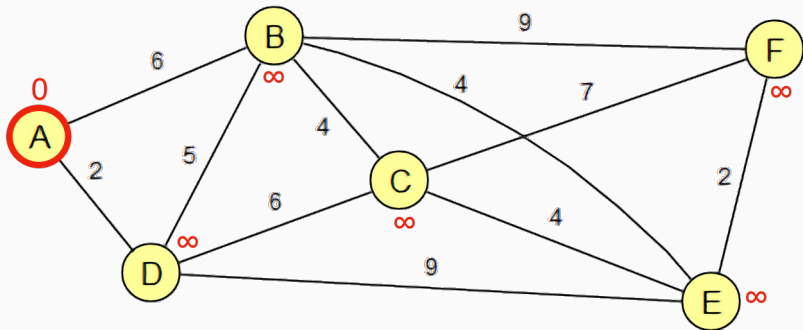
Der Algorithmus von Dijkstra - Ablauf

0. Weise dem Startknoten die Distanz 0 und allen anderen Knoten die Distanz ∞ zu
1. Wähle den Knoten k mit der geringsten Distanz als aktuellen Knoten und markiere ihn als besucht
2. Berechne für alle noch unbesuchten Nachbarknoten n von k die Distanz zum Startknoten beim Weg über k
 - Ist diese für einen Nachbarn kleiner als seine bisher gespeicherte Distanz, so aktualisiere dessen Distanz und setze seinen Vorgänger auf k
3. Solange es noch unmarkierte Knoten gibt, gehe zurück zu Schritt 1

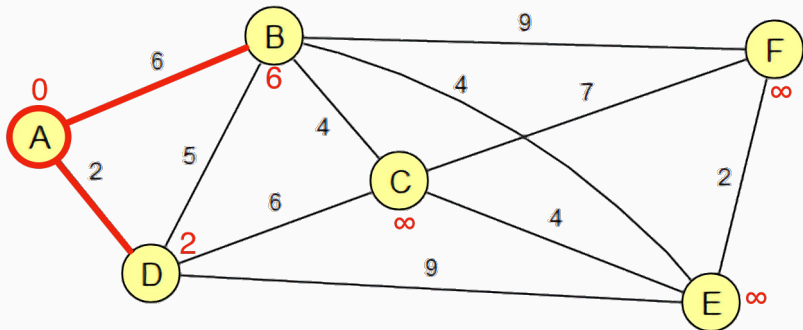
Der Algorithmus von Dijkstra - Beispiel



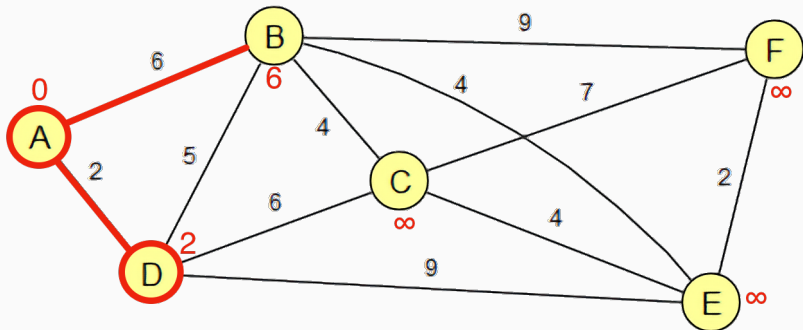
Der Algorithmus von Dijkstra - Beispiel



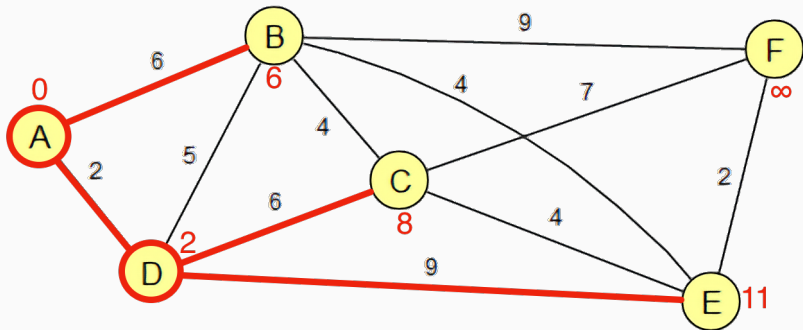
Der Algorithmus von Dijkstra - Beispiel



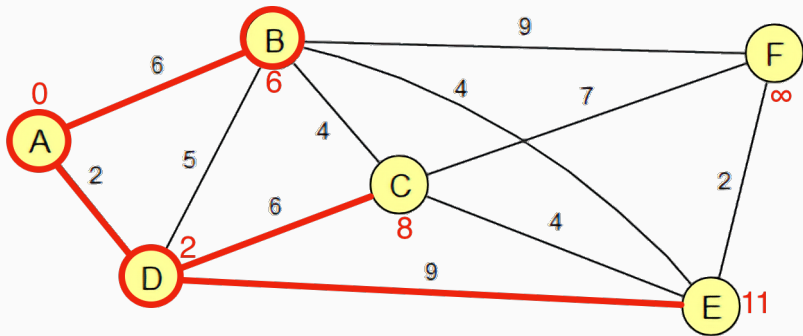
Der Algorithmus von Dijkstra - Beispiel



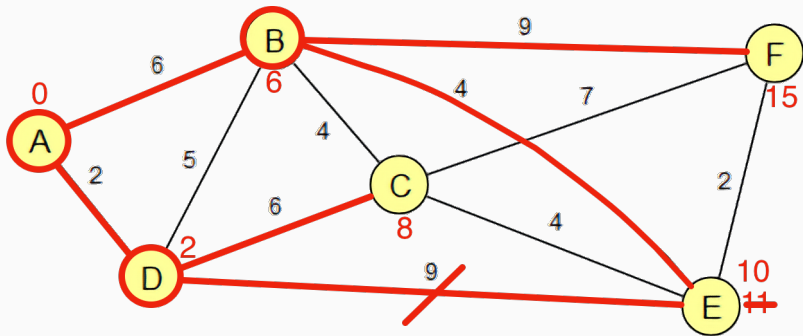
Der Algorithmus von Dijkstra - Beispiel



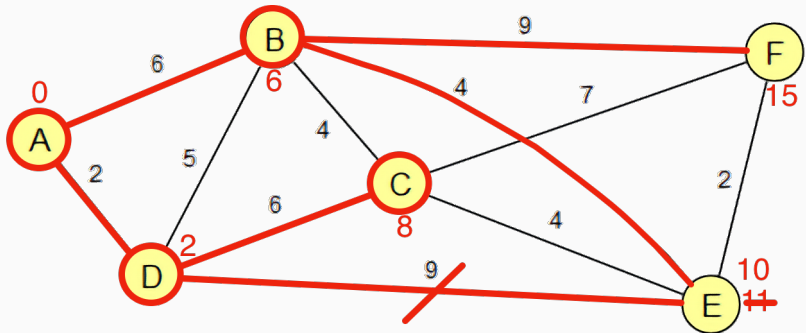
Der Algorithmus von Dijkstra - Beispiel



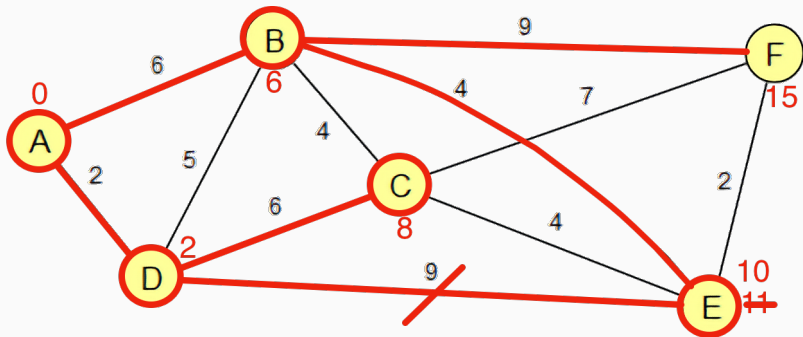
Der Algorithmus von Dijkstra - Beispiel



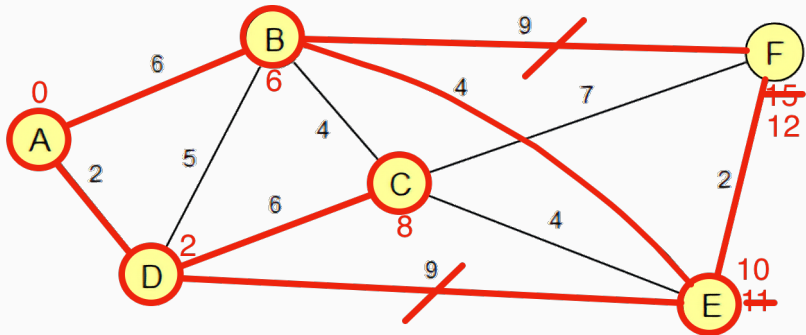
Der Algorithmus von Dijkstra - Beispiel



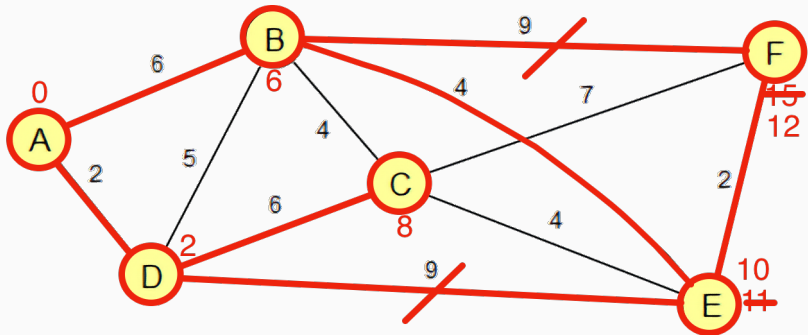
Der Algorithmus von Dijkstra - Beispiel



Der Algorithmus von Dijkstra - Beispiel

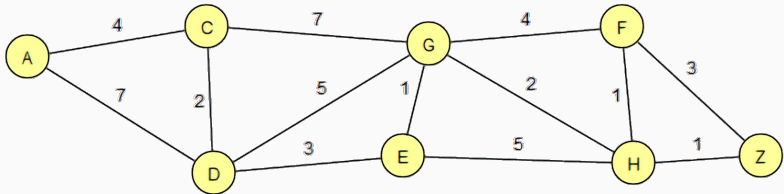


Der Algorithmus von Dijkstra - Beispiel



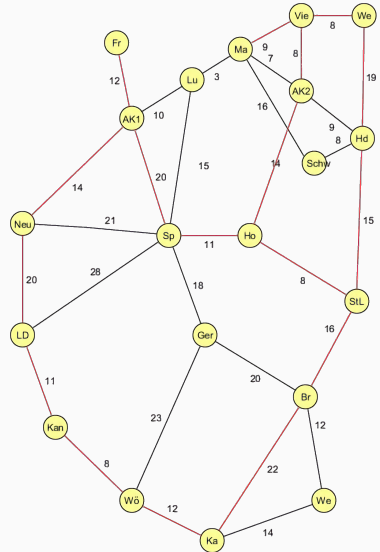
Aufgabe 3

- Wende den Algorithmus von Dijkstra auf dem folgenden Graphen an, um den kürzesten Weg von A nach Z zu bestimmen



Aufgabe 4

- Wende den Algorithmus von Dijkstra auf dem folgenden Graphen an, um den kürzesten Weg von Karlsruhe (Ka) nach Mannheim (Ma) zu bestimmen



- Der Dijkstra-Algorithmus findet, ausgehend von einem Startknoten, die kürzesten Wege zu allen anderen Knoten im Graphen
 - Ist man nur am Weg zu einem bestimmten Zielknoten interessiert, so kann man den Algorithmus abbrechen, sobald dieser Zielknoten der aktive Knoten ist
- Der Dijkstra-Algorithmus sucht vom Startknoten ausgehend “kreisförmig” in alle Richtungen
 - Ist man nur am Weg zu einem bestimmten Zielknoten interessiert, so werden viele “unnötige” Entfernungen berechnet (die Entfernung von Karlsruhe nach Rastatt ist uninteressant, wenn man von Karlsruhe nach Mannheim fahren will)
- Mögliche Optimierung:
 - Der A*-Algorithmus (gesprochen: “A Stern”)

- Informiere Dich im Internet über den A*-Algorithmus
 - Wie sorgt A* dafür, dass Knoten “in die richtige Richtung” bevorzugt werden?
 - Linktipp: https://www-m9.ma.tum.de/graph-algorithms/spp-a-star/index_de.html
- Vergleiche die Arbeitsweise von Dijkstra und A* auf folgender Seite
 - <https://qiao.github.io/PathFinding.js/visual/>