Greedy-Algorithmen

Wojciech Kawecki
Thomas Sech
IIb07

Gliederung

- Einführung / Erklärung
- Bruchteilrucksack-Problem
- Aufbau von Greedy-Algorithmen
- Beispiel (Geldwechsel)
- Verschiedene Greedy-Algorithmen
  - kürzester Weg in Graphen
  - minimaler Spannbaum

Einführung / Erklärung

- gefräßige Strategie
- rechnerische Entscheidung
  - Entscheidung auf Basis lokaler Informationen
  - kein Rückgriff auf vorher ermittelte Werte
  - endgültige Entscheidung
- Aufwand nie exponentiell

Bruchteilrucksack-Problem

- unterschiedlich zum 0/1 Rucksackproblem
- homogene Verteilung
- der Gegenstand mit den höhsten spezifischen Wert erhält die Nummer 1
- danach folgen die anderen (2, 3, 4, …, n)
- Beispiel: Rohstoffe

Bruchteilrucksack-Problem

- Die Kandidatenliste wird nach dem Quotient aus:
  - w_i → Wert des i-ten Gegenstandes
  - g_i → Gewicht des i-ten Gegenstandes

absteigend sortiert.

$$\frac{w_1}{g_1} \geq \frac{w_2}{g_2} \geq \ldots \geq \frac{w_n}{g_n}$$

Bruchteilrucksack-Problem

- K → Kapazität (Maximalgewicht)
- j → Anzahl der Gegenstände im Rucksack
- j+1 → Gegenstand der nicht vollständig in den Rucksack passt
- g_j → das Gegendstandsgewicht
- t → Teil des Gegendstandsgewichtes ($0 \leq t \leq 1$)

$$K-(g_1+g_2+\ldots+g_j)=t^*\ g_{j+1}$$

Bruchteilrucksack-Problem

- Der Gesamtwert des gefüllten Rucksacks beträgt:

$$W = w_1 + w_2 + \ldots + w_j + t^* \; w_{j+1}$$

Bruchteilrucksack-Problem

- Kann man wirklich sicher sein, dass auf dieser Weise der größtmögliche Gesamtwert erreicht wird?

- Überlegung:
  - Ist es besser von einem vollständig eingepackten Gegenstand e (mit $1 \leq e \leq j$) nur ein Teil y ($0 < y < 1$) einzupacken, so dass man soviel Platz hat, einen bestimmten Teil z eines Gegenstandes r ($e < r \leq n$) mit kleinerem spezifischem Wert als von e einzupacken?

Bruchteilrucksack-Problem

- Kein Wertzuwachs, im Gegenteil es gilt W'<W
- Die Greedy-Methode löst das Bruchteilrucksack-problem effizient
- versagt bei 0/1 Rucksackproblem
- Man erhält aber eine brauchbare Nährungslösung

Aufbau

- Greedy-Algorythmus Bestandteile:
  - Kandidatenliste
  - Resultatliste
  - eine Funktion *loesung?*
  - eine Funktion *okay?*
  - eine Auswahlunktion *naechster?*
  - eine Modifikationsfunktion *modify+*
  - eine Funktion *modify-*
  - eine Funktion *ziel*

```
(define greedy
 (lambda (vorgabewert) ; z.B.: Kapazitaet der Rucksacks
  (let
      ((kandidatenliste '(.....))
       (loesung? (lambda (ls) ... ))
       (okay? (lambda (ls) ... ))
       (naechster (lambda (ls) ... ))
       (modify+ (lambda (x ls) ... ))
       (modify- (lambda (x ls) ... ))
       (ziel (lambda (ls) ... ))
    (letrec
        ((helfer
           (lambda (kandidaten resultat)
            (cond
             ((loesung? resultat)(ziel resultat))
             ((and (null? kandidaten)(not (loesung? resultat)))
              (error 'greedy „Es gibt keine Loesung."))
             (else
              (let* ((neuer-kandidat (naechster kandidaten))
                     (neues-resultat (cons neuer-kandidat resultat)))
                 (if (okay? neues-resultat)
                     (helfer (modify+ neuer-kandidat kandidaten) neues-resultat)
                     (helfer (modify- neuer-kandidat kandidaten) resultat))))))))
        (helfer kandidatenliste '()))))))
```

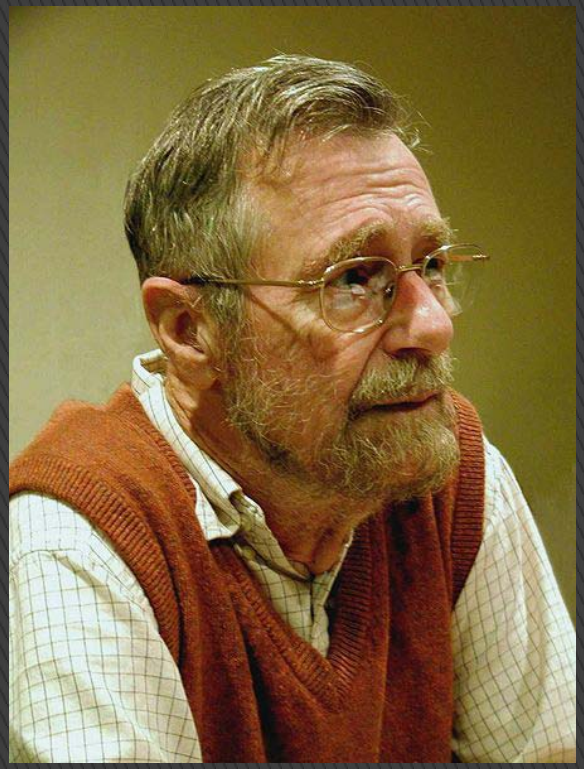
Beispiel: Geldwechsel

- Problem: Einem Kunden wird Wechselgeld herausgegeben, wobei möglichst wenige Münzen verwendet werden sollen.

- Die Kandidatenliste: alle Euro-Münzen.

Dijkstras Algorithmus

- Edsger Wybe Dijkstra
- * 11. Mai 1930; † 6. August 2002
- Ein niederländischer Informatiker
- Im Jahre 1972 erhielt Dijkstra den Turing Award



Dijkstras Algorithmus

- 1959 von Dijkstra gefunden
- kürzeste Wege in Graphen
  - n → Anzahl der Knoten
  - m → Anzahl der Kanten
- die Greedy-Natur ist versteckt
- naive Implementation:
  - $O(n^2)$
- Implementation mit heaps für priority queues:
  - $O((m+n) \log n)$

Dijkstras Algorithmus

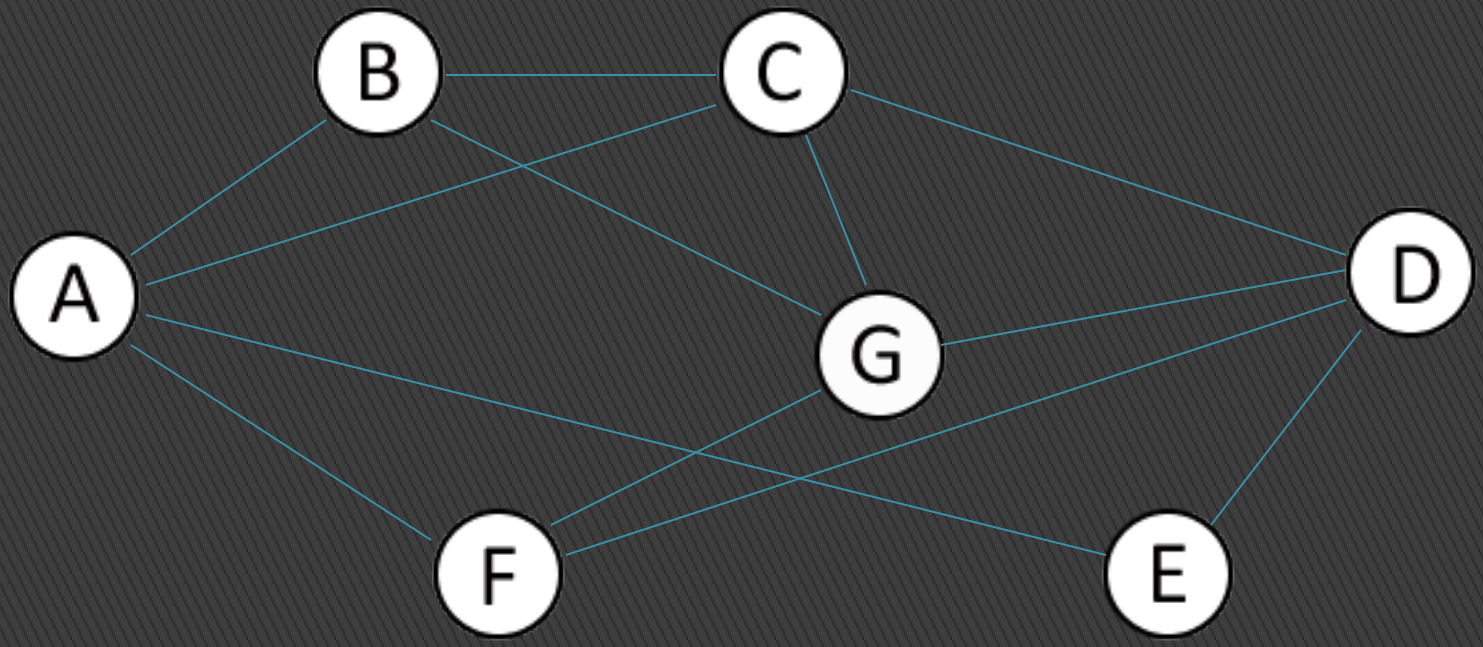
- Adjazenzmatrix Unsetzung in Graph

	A	B	C	D	E	F	G
A	0	4	9	0	10	6	0
B	4	0	3	0	0	0	5
C	9	3	0	8	0	0	6
D	0	0	8	0	3	9	3
E	10	0	0	3	0	0	0
F	0	0	0	9	0	0	7
G	0	5	6	3	0	7	0

- Gerichteter Graph
  - Kanten sind einseitig gerichtete Pfeile

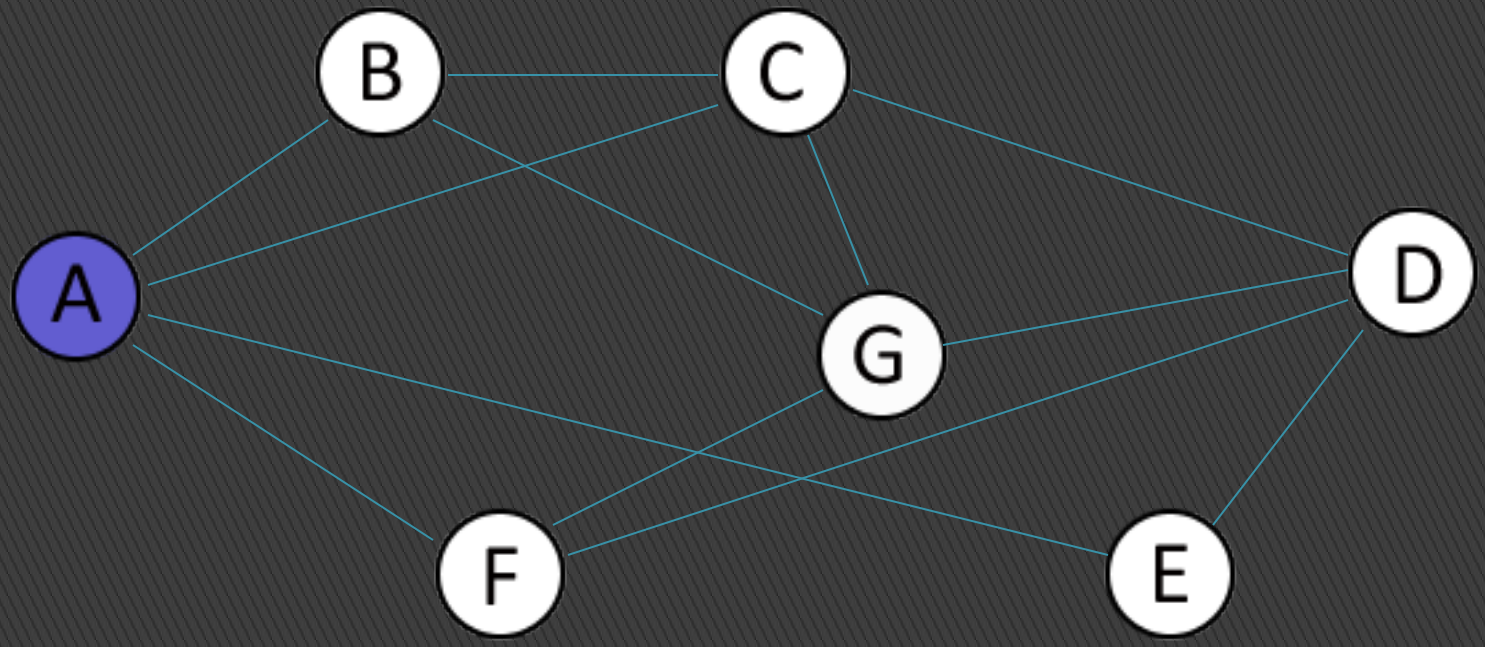
Dijkstras Algorithmus

- kürzeste Wege in Graphen



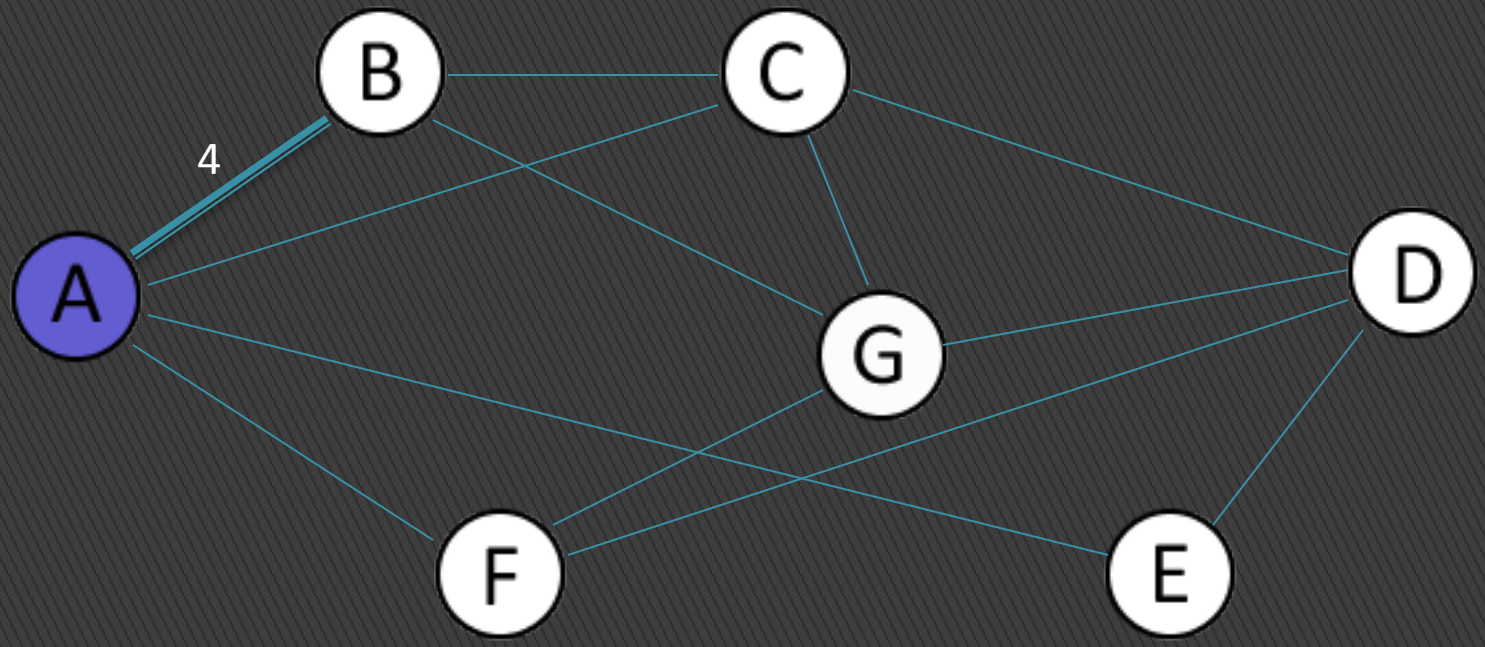
Dijkstras Algorithmus

- kürzeste Wege in Graphen



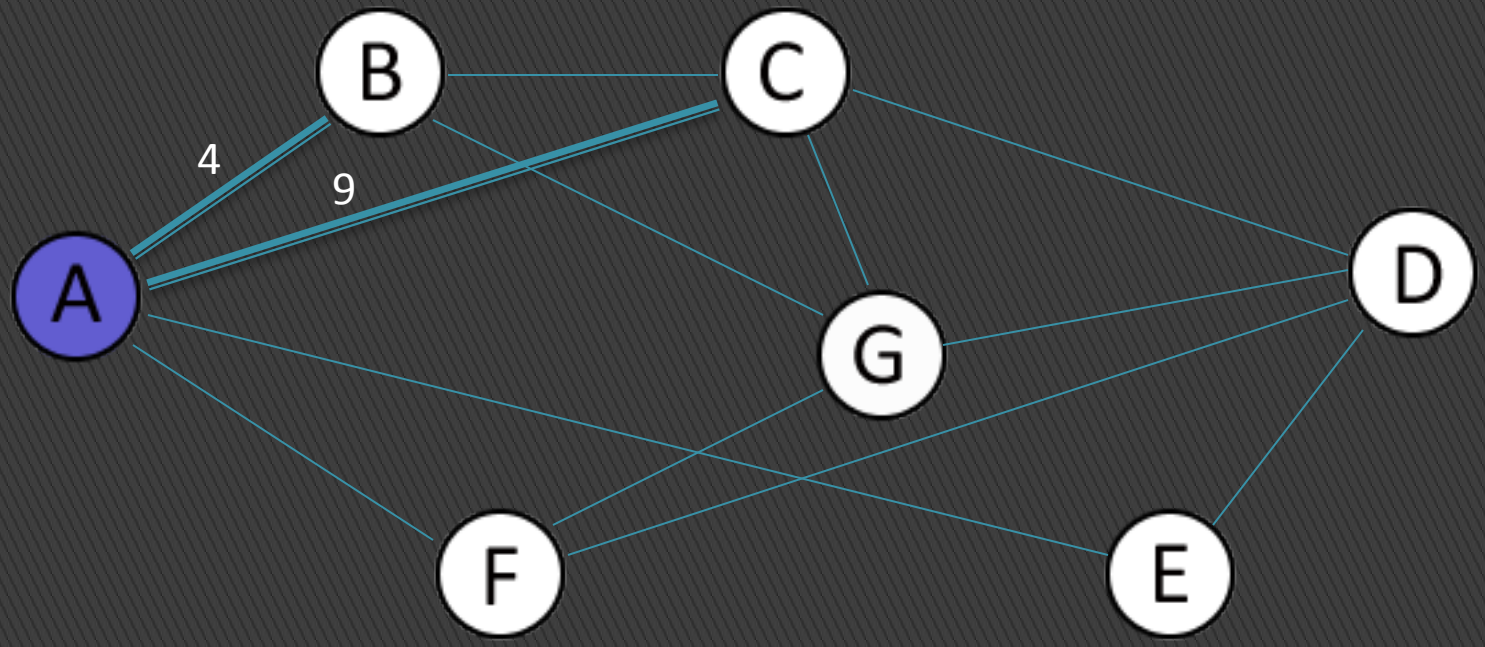
Dijkstras Algorithmus

- kürzeste Wege in Graphen



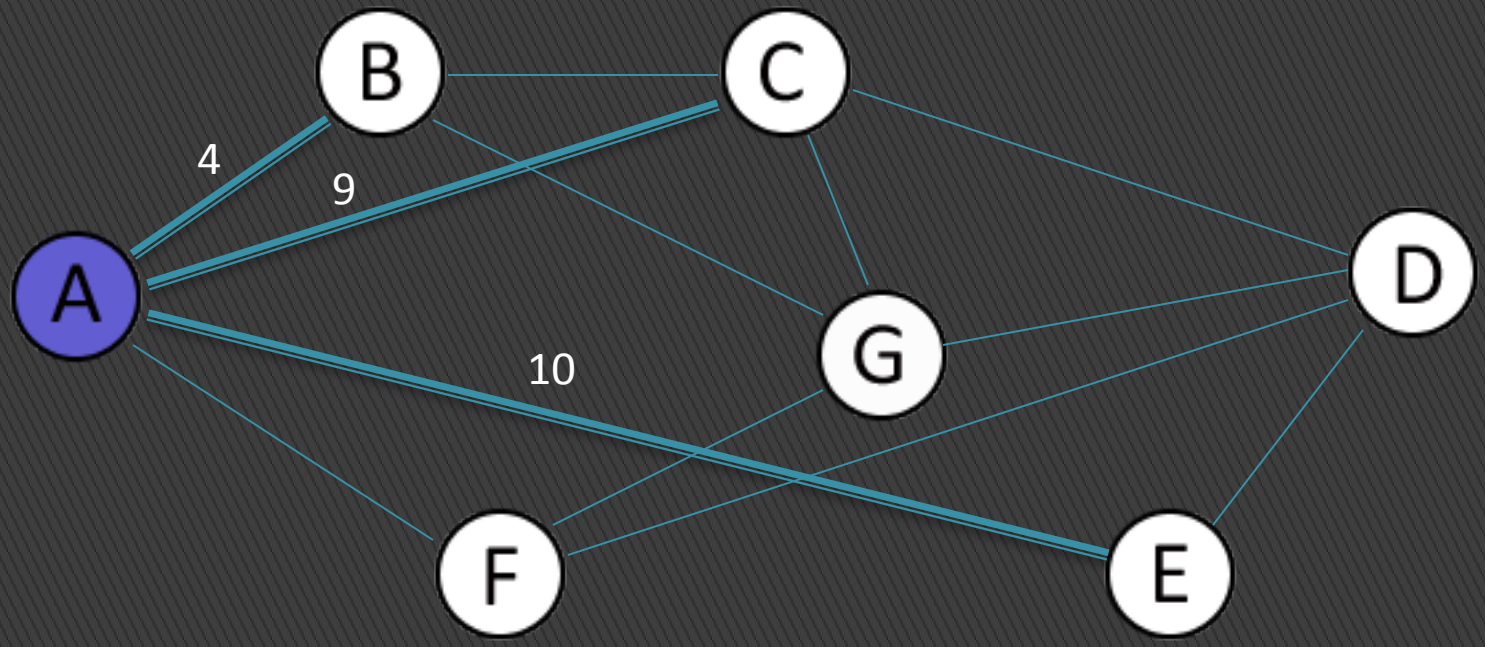
Dijkstras Algorithmus

- kürzeste Wege in Graphen



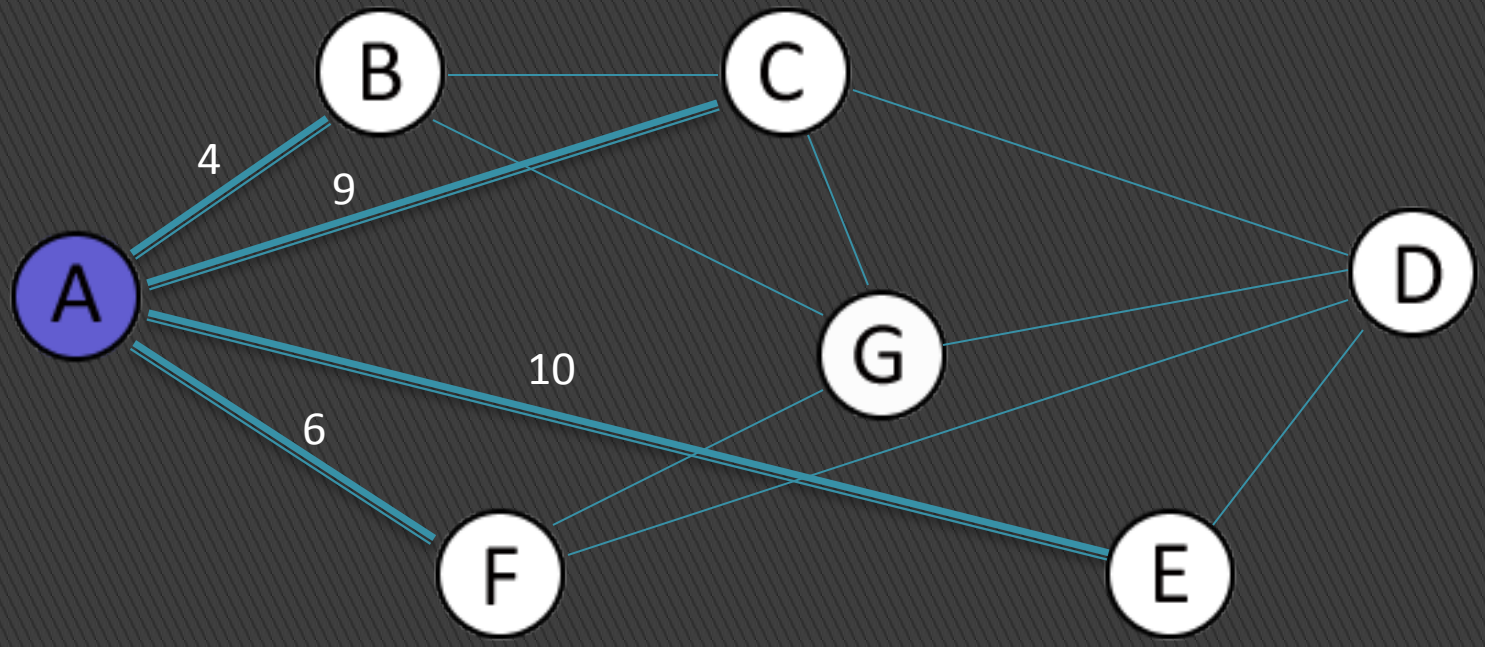
Dijkstras Algorithmus

- kürzeste Wege in Graphen



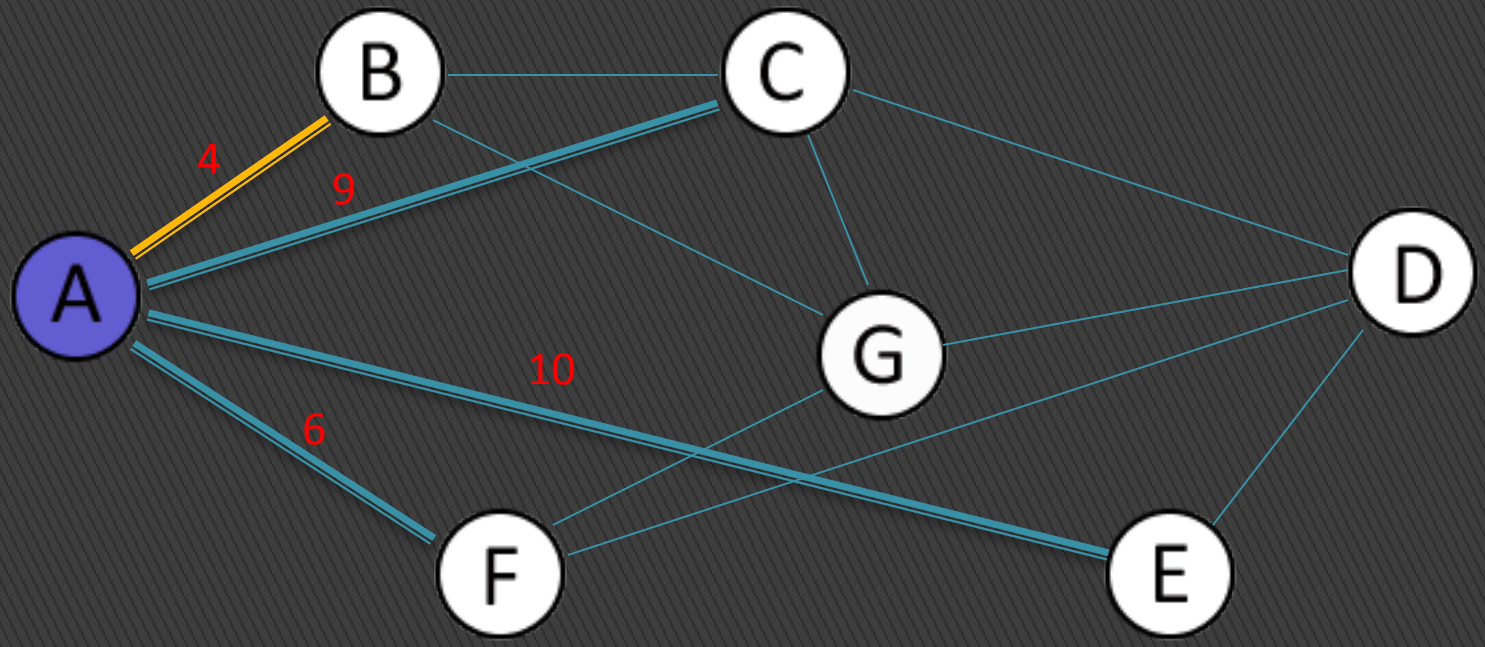
Dijkstras Algorithmus

- kürzeste Wege in Graphen



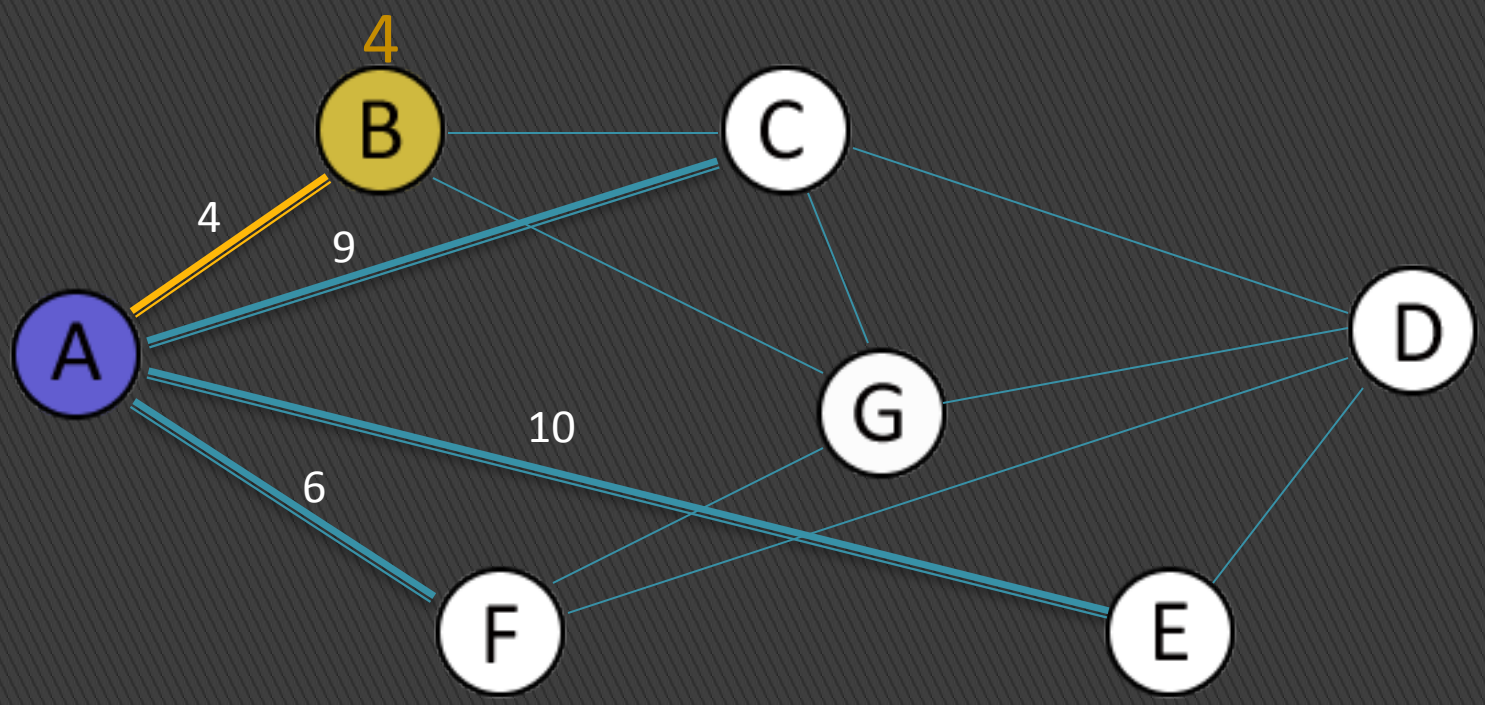
Dijkstras Algorithmus

- kürzeste Wege in Graphen



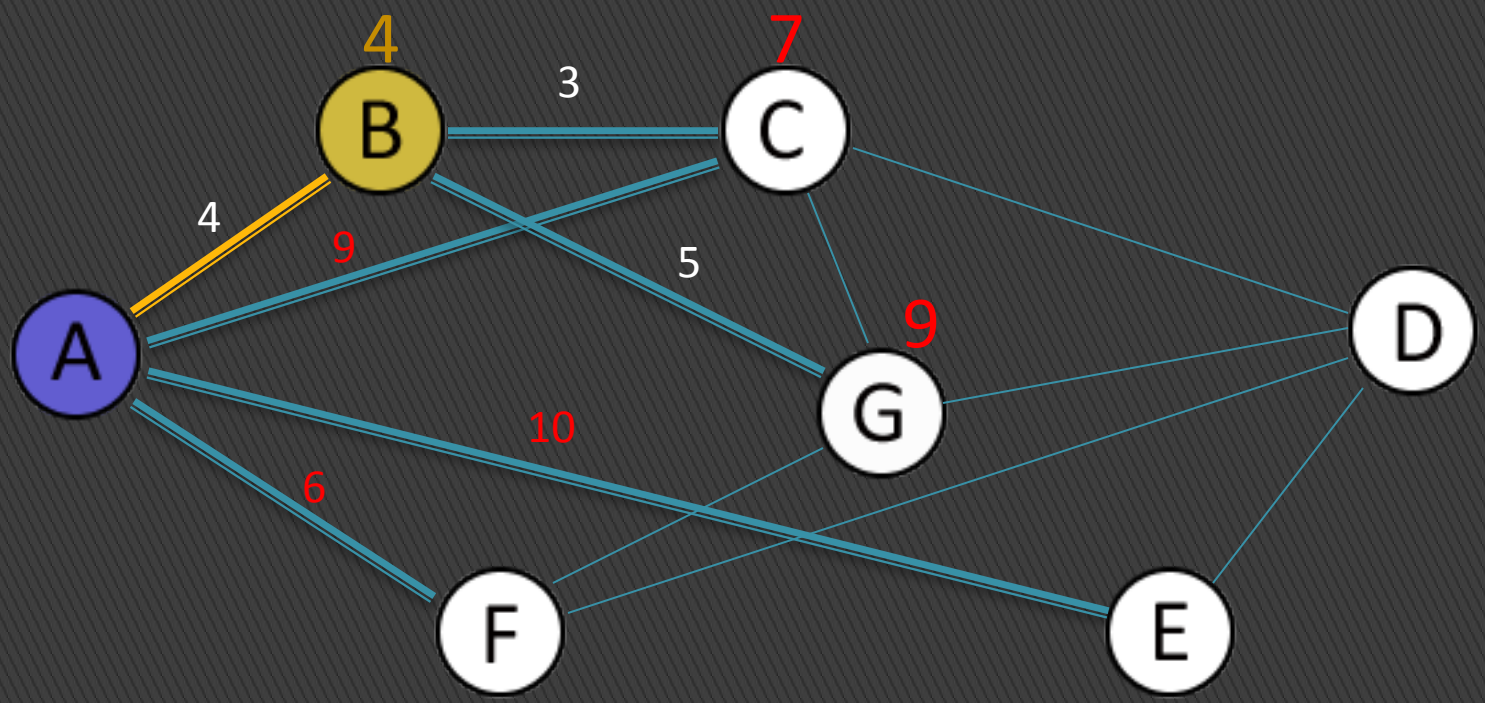
Dijkstras Algorithmus

- kürzeste Wege in Graphen



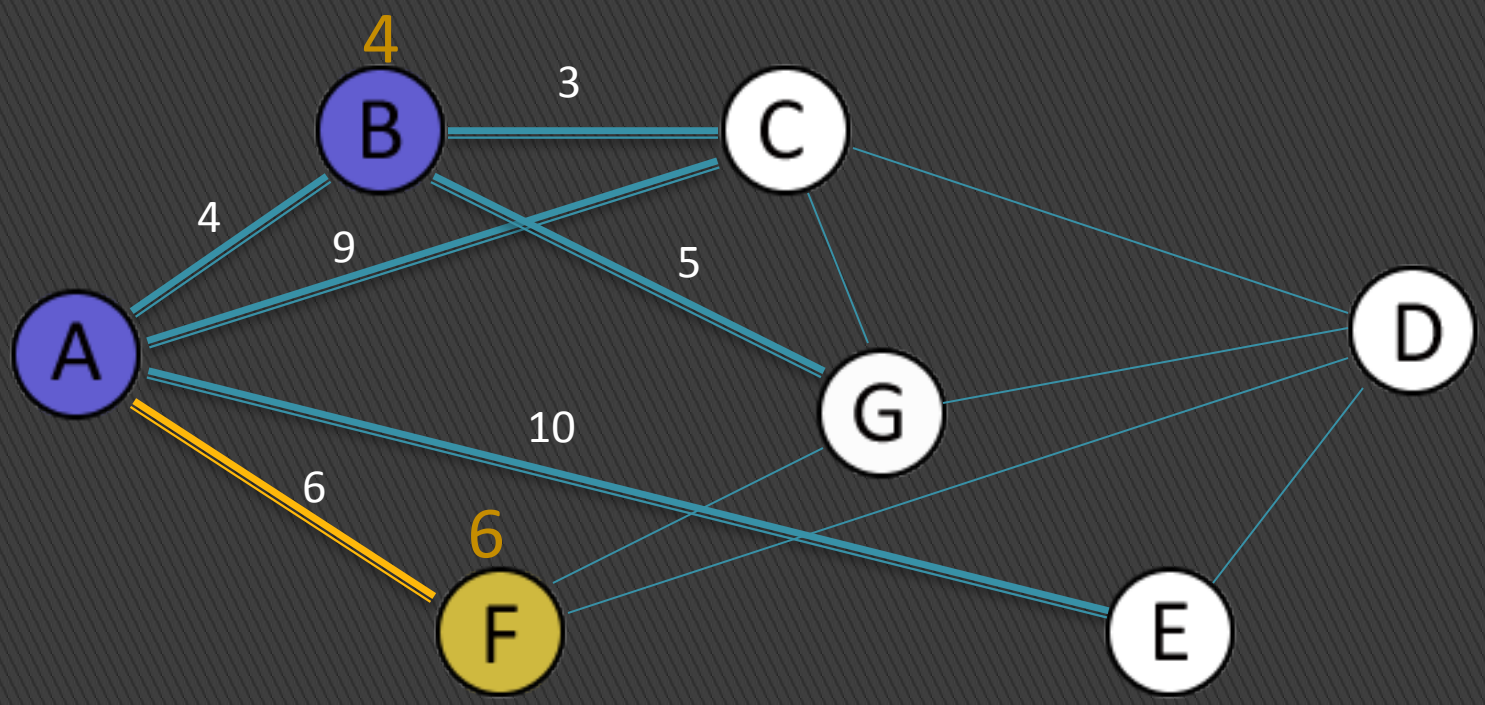
Dijkstras Algorithmus

- kürzeste Wege in Graphen



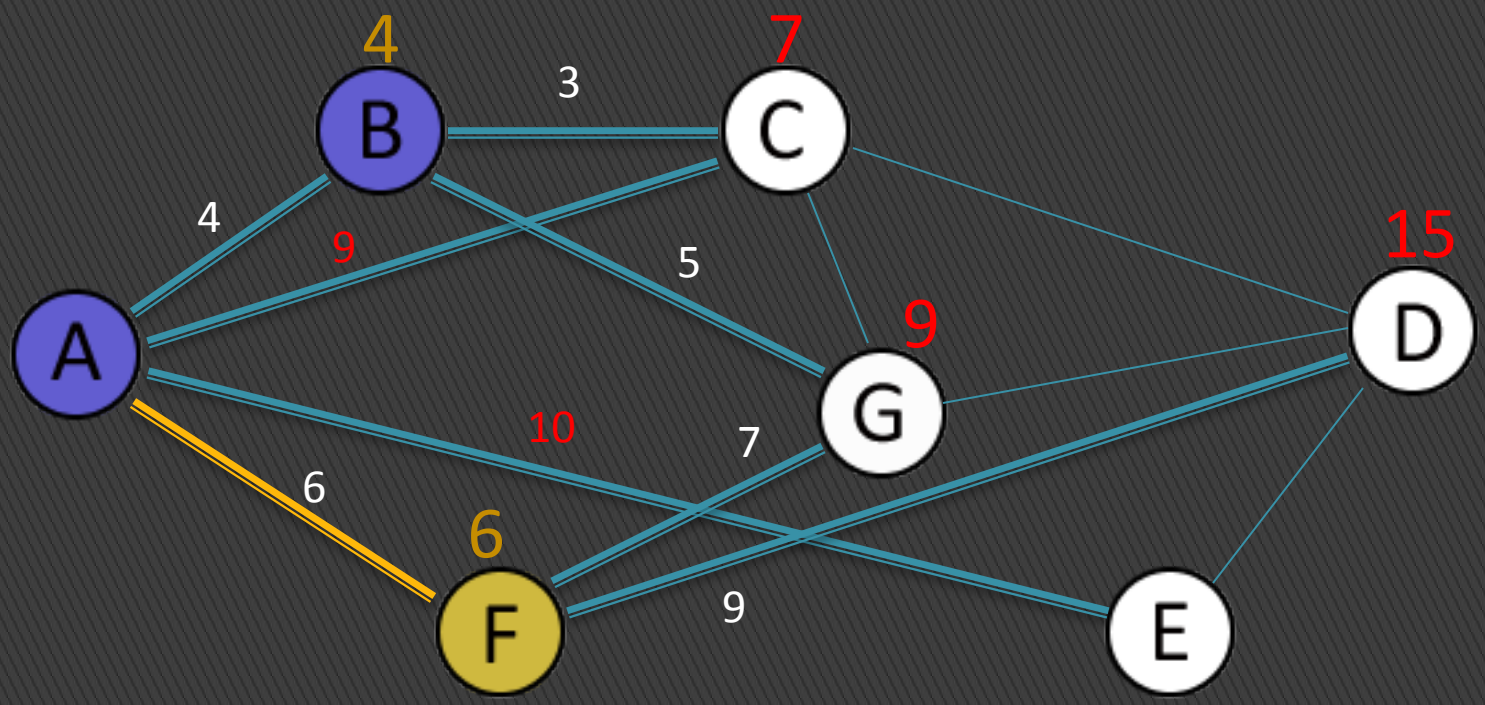
Dijkstras Algorithmus

- kürzeste Wege in Graphen



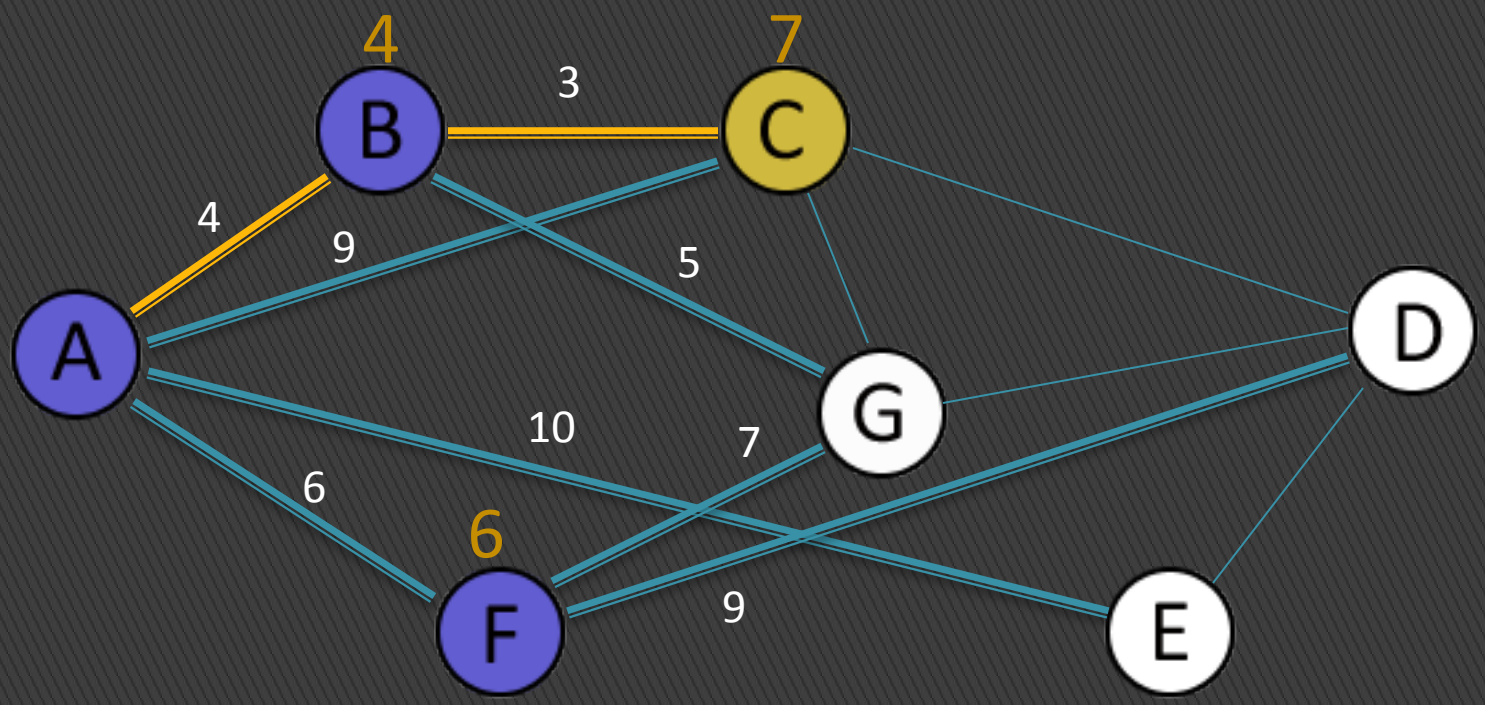
Dijkstras Algorithmus

- kürzeste Wege in Graphen



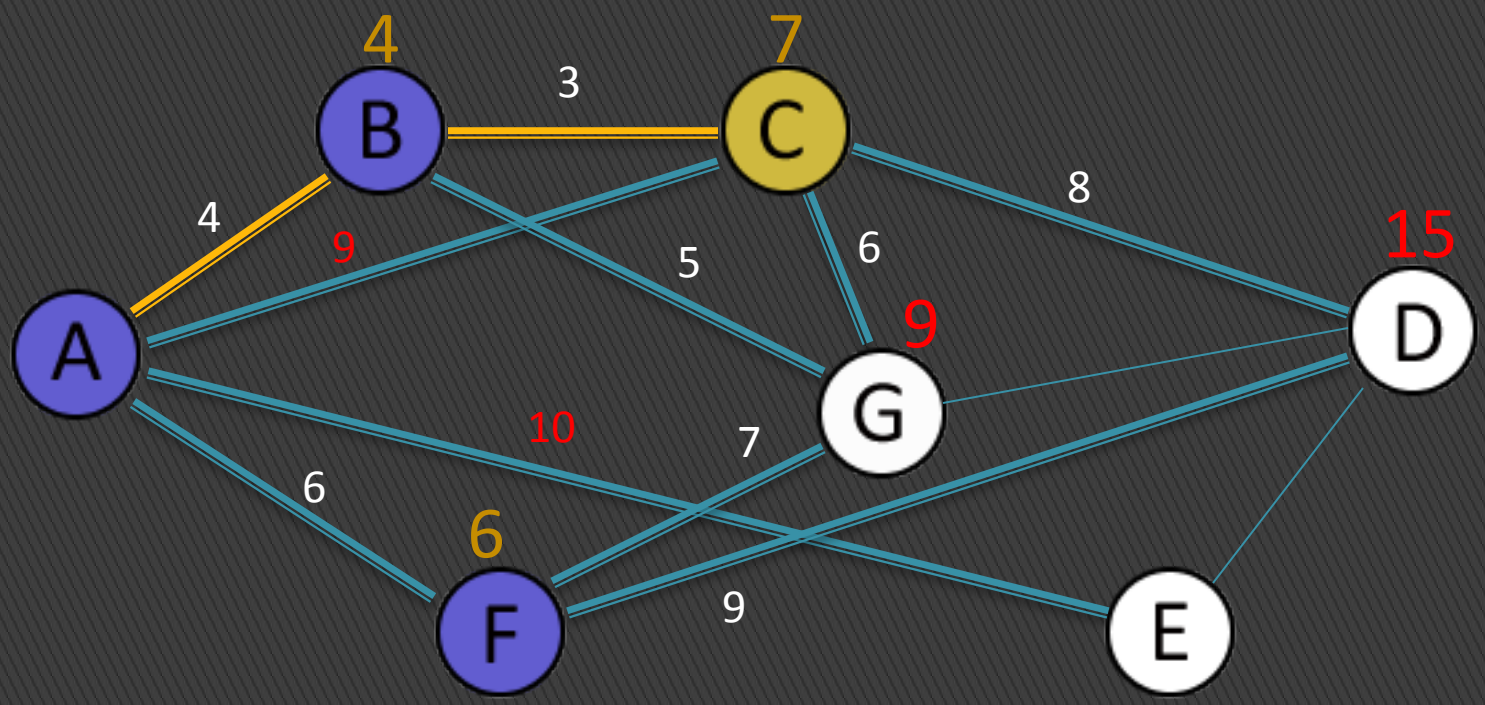
Dijkstras Algorithmus

- kürzeste Wege in Graphen



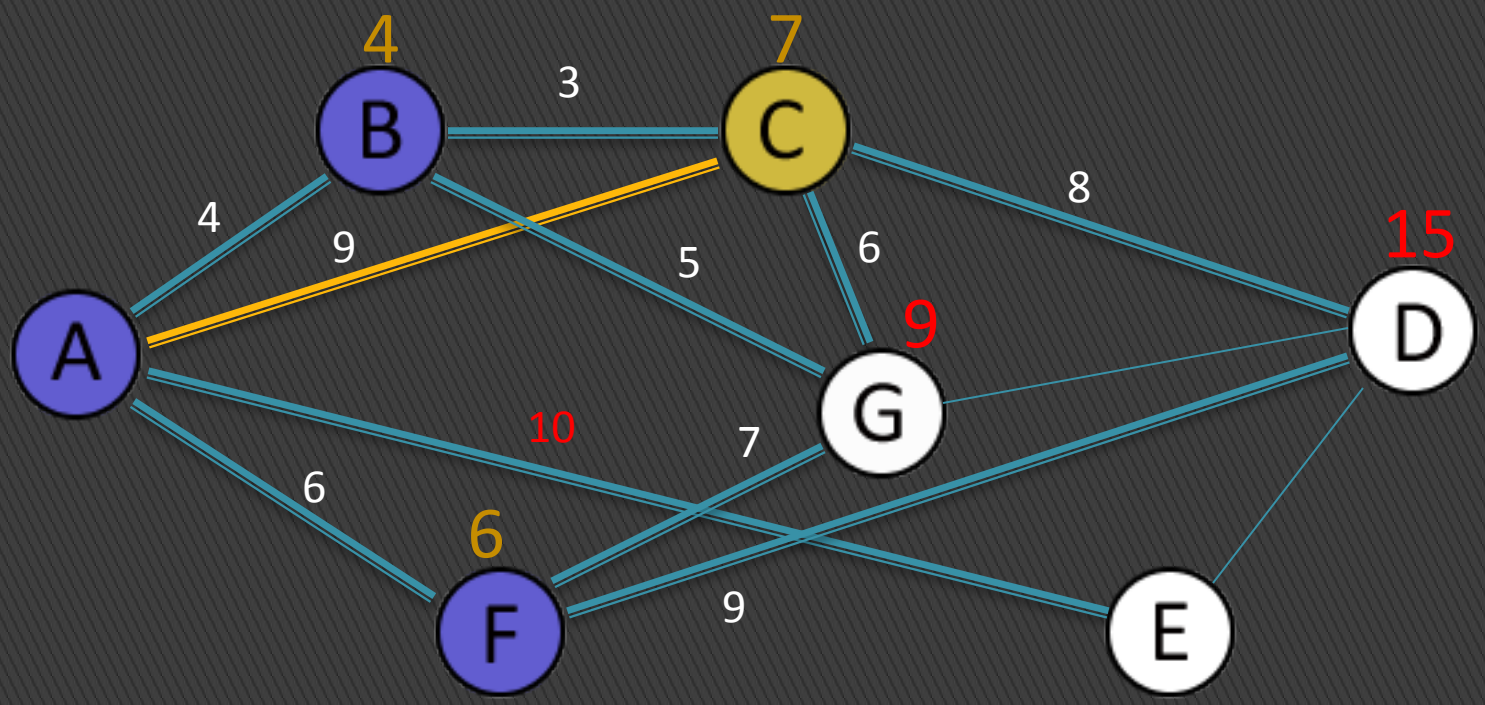
Dijkstras Algorithmus

- kürzeste Wege in Graphen



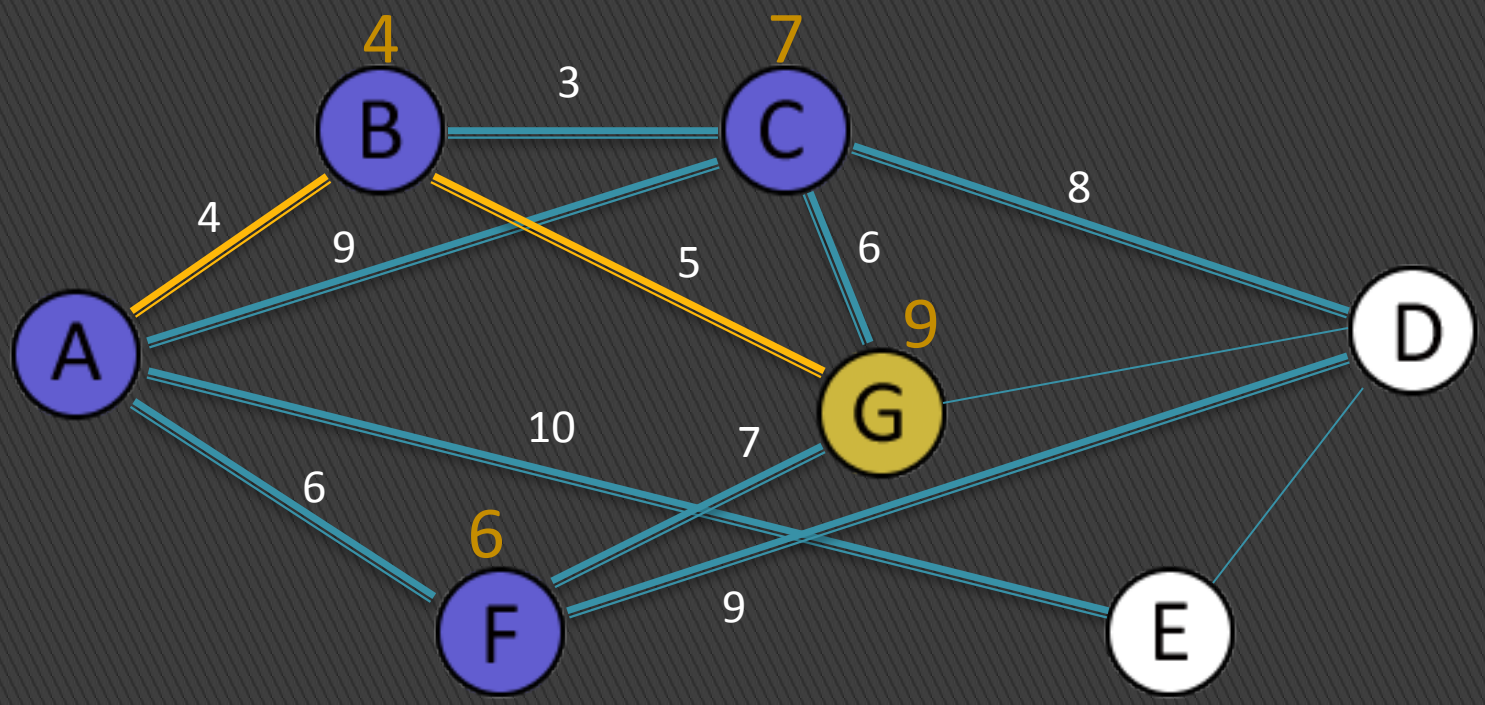
Dijkstras Algorithmus

- kürzeste Wege in Graphen



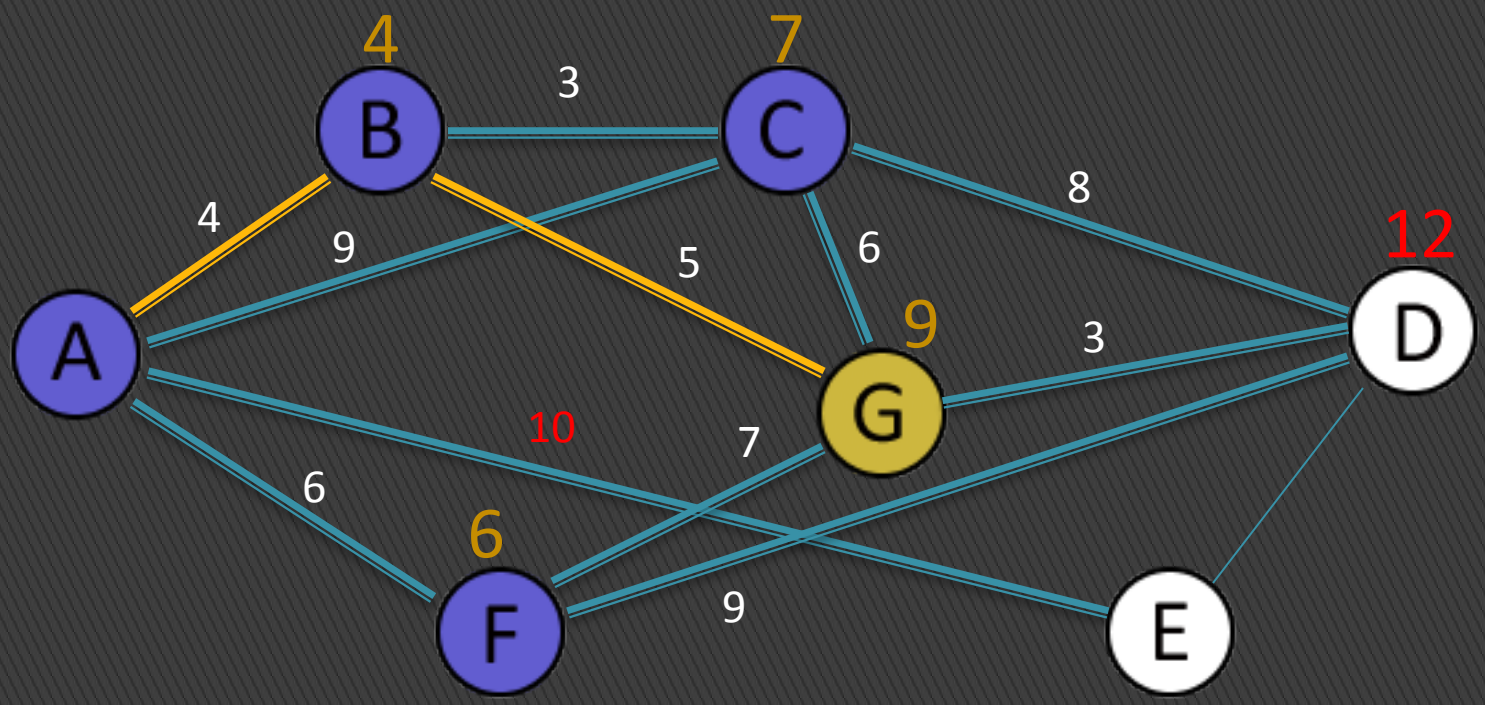
Dijkstras Algorithmus

- kürzeste Wege in Graphen



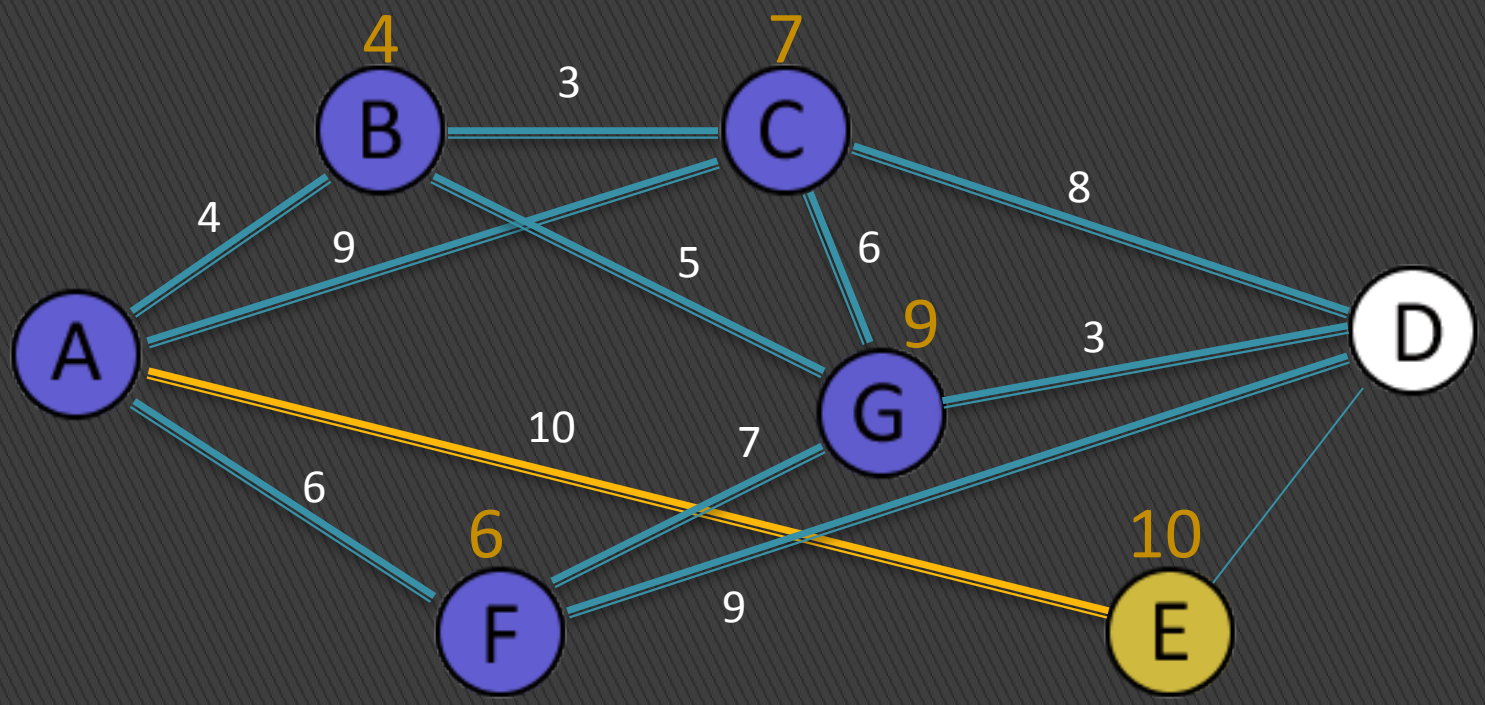
Dijkstras Algorithmus

- kürzeste Wege in Graphen



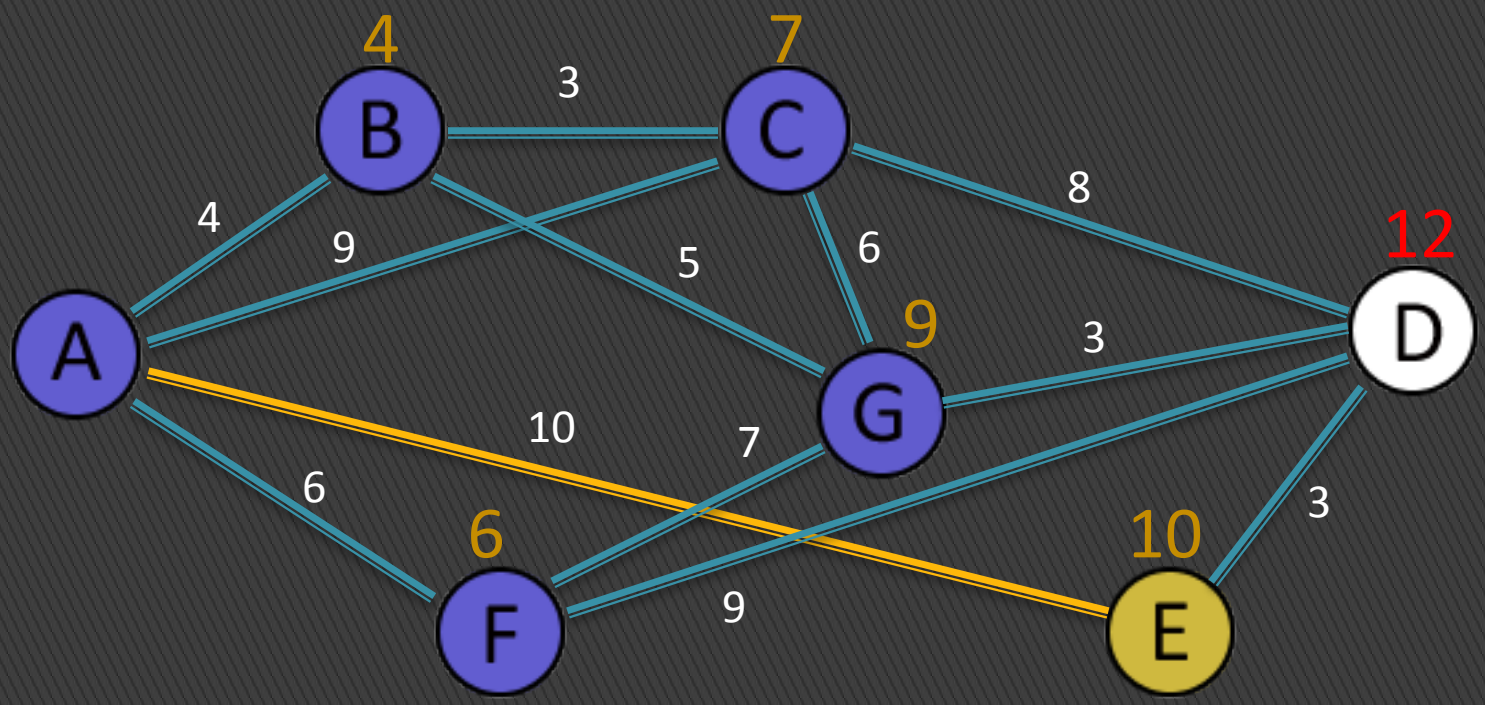
Dijkstras Algorithmus

- kürzeste Wege in Graphen



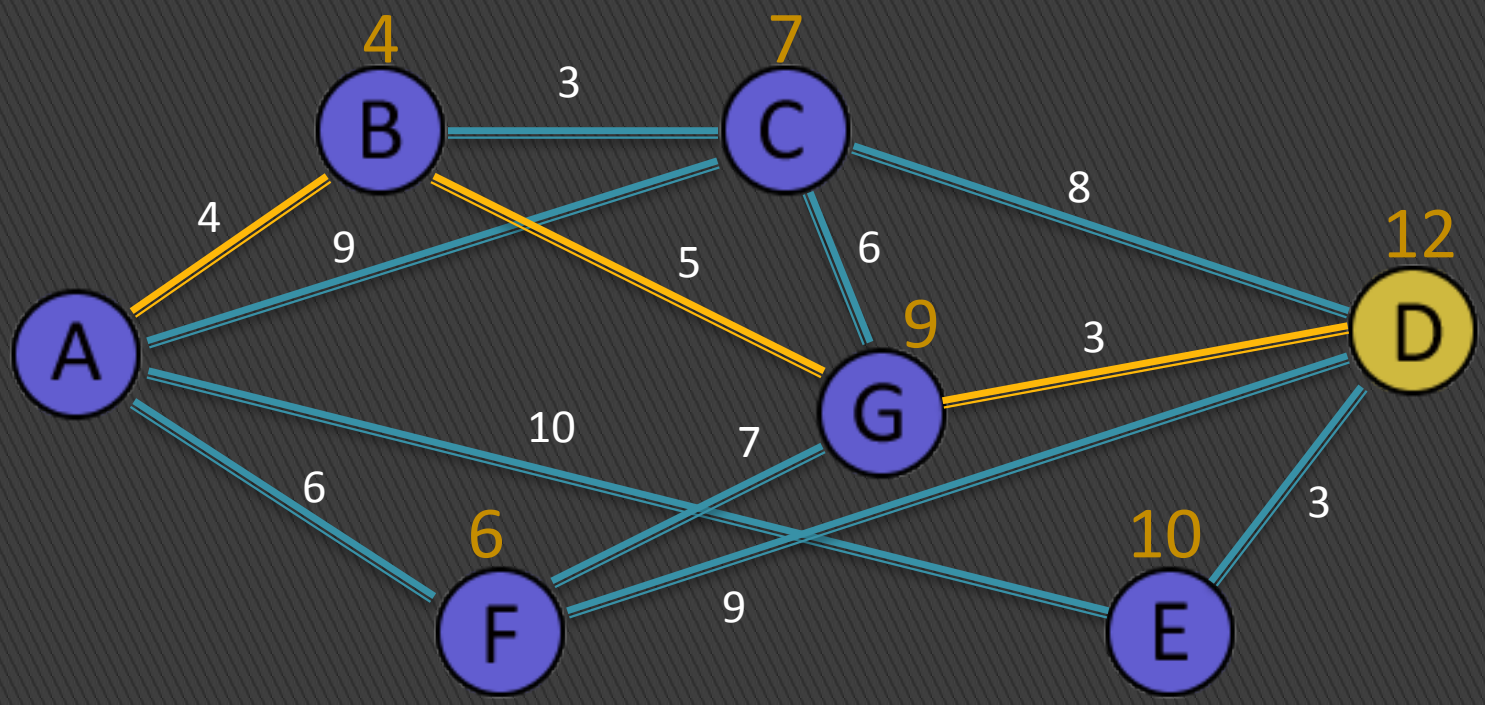
Dijkstras Algorithmus

- kürzeste Wege in Graphen



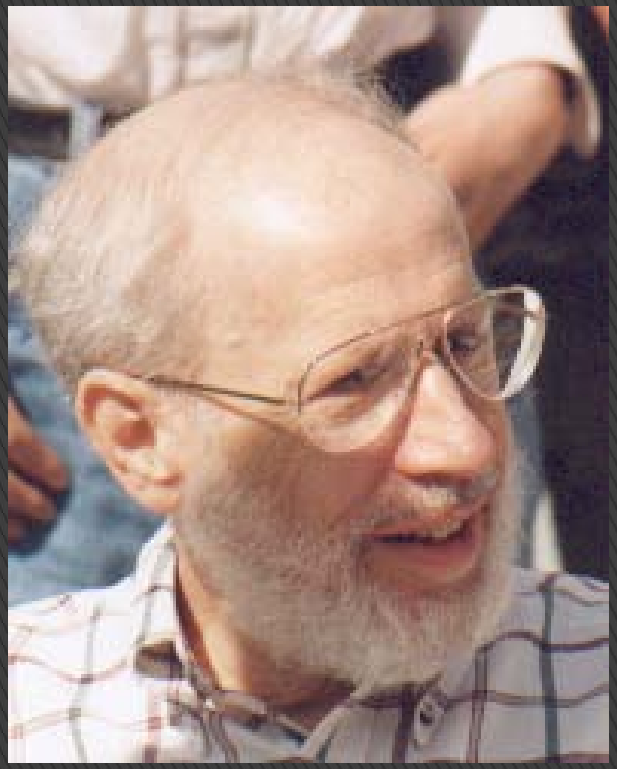
Dijkstras Algorithmus

- kürzeste Wege in Graphen



Algorithmus von Kruskal

- Joseph Kruskal
- * 1929 in New York City
- US-amerikanischer Mathematiker und Statistiker



Algorithmus von Kruskal

- minimaler Spannbaum
  - Zusammenhängender Teilgraph
  - Gesamtgewicht: minimal
  - Knotenmenge gleich
  - Knoten genau einmal angeschlossen

Algorithmus von Kruskal

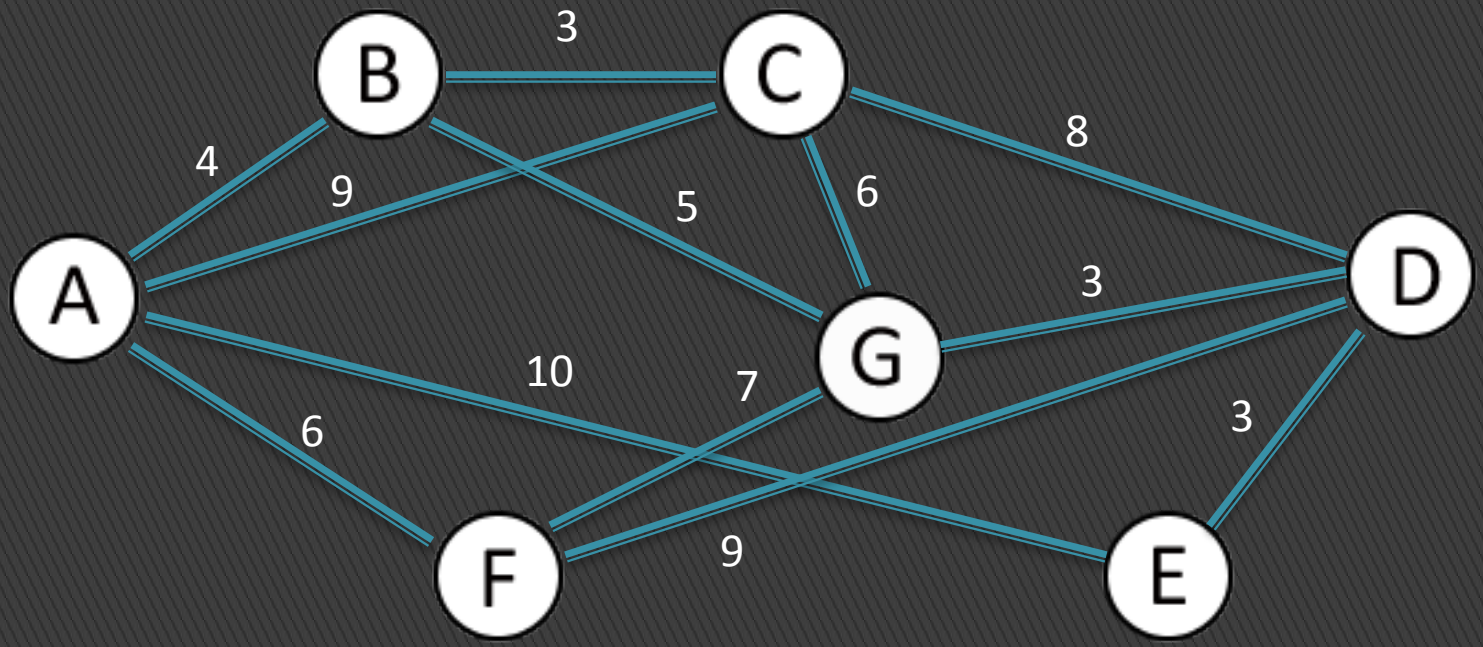
- Minimaler Spannbaum
  - n → Anzahl der Knoten
  - a → Anzahl der Kanten
- Aufwand:
  - $O(a \log n)$
  - $O(n \log n)$ $a \leq n$ wenige Kanten
  - $O(n^2 \log n)$ $a \leq 2/n(n-1)$ höherer Dichte

Algorithmus von Kruskal

- Grundidee
  - Kanten nach ihren Gewichten geordnet (aufsteigend)
  - Die Kante mit dem kleinsten gewicht wird gewählt
  - egal ob angeschlossen an der ersten Kante
  - wenn Kreis lässt man die Kante weg

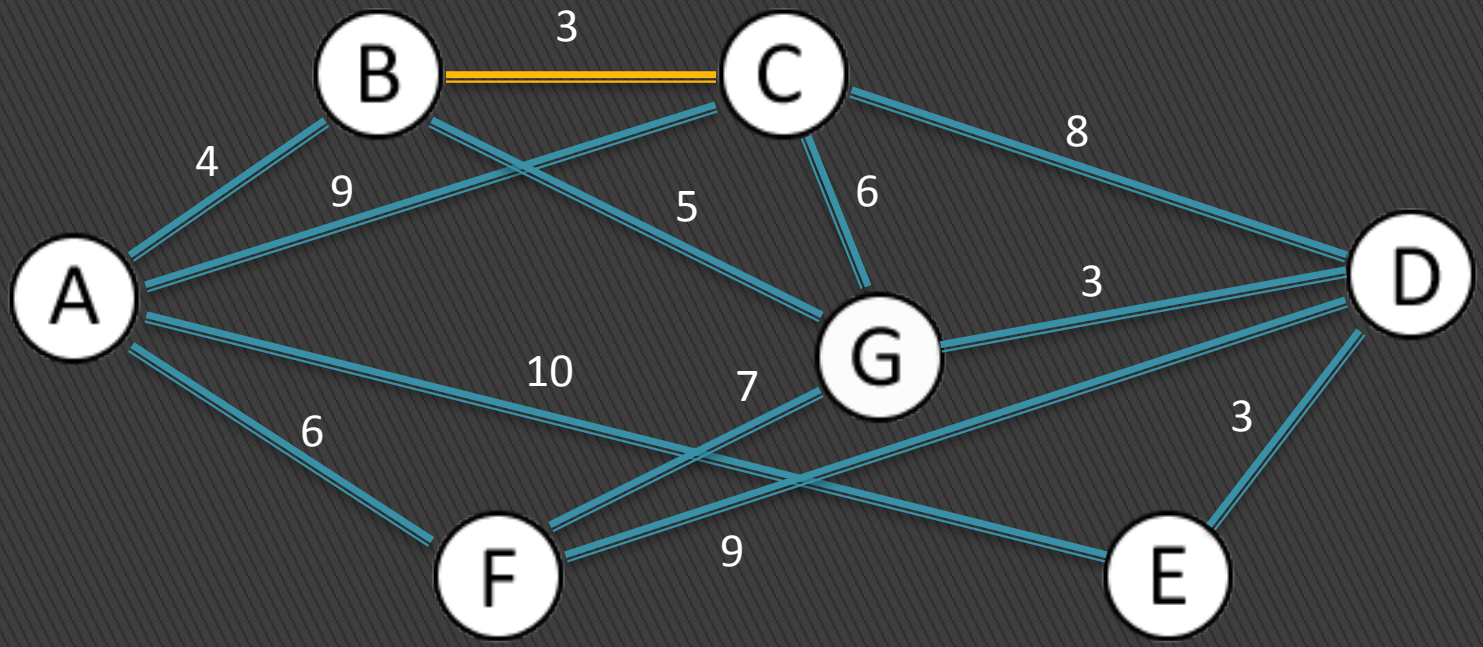
Algorithmus von Kruskal

- minimaler Spannbaum



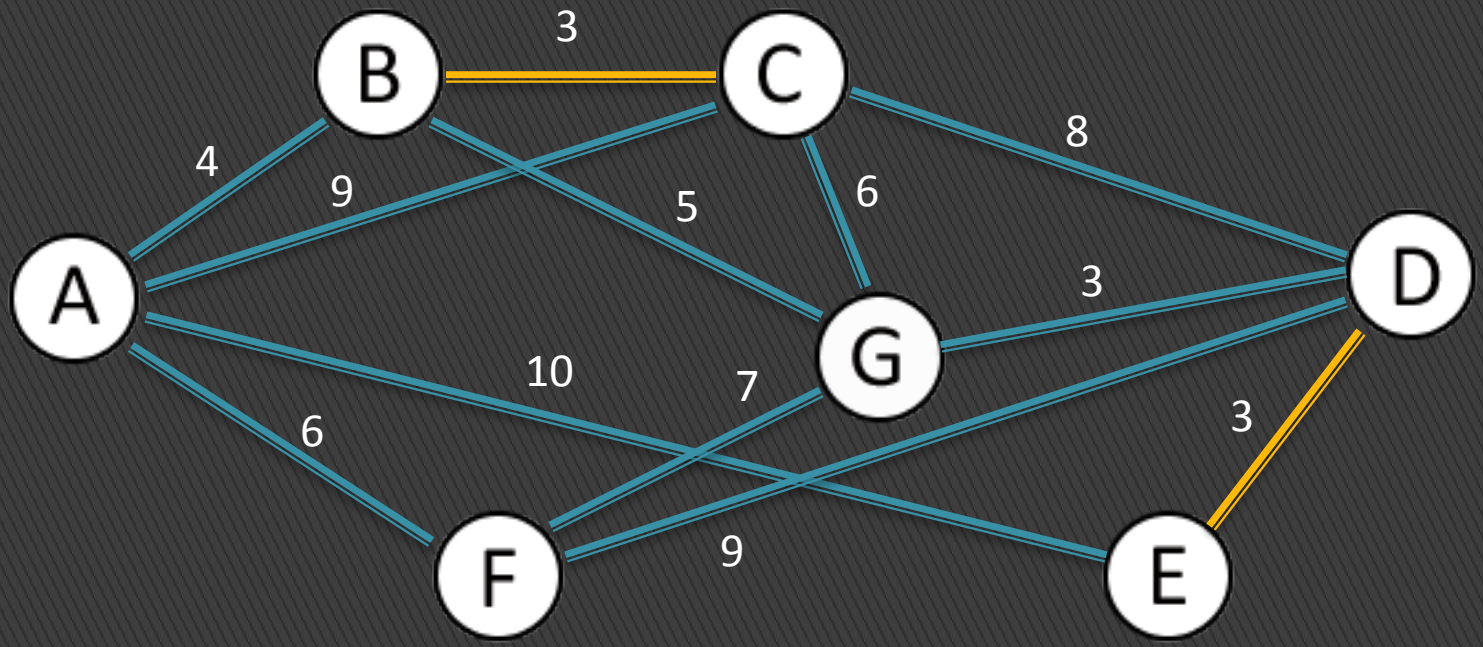
Algorithmus von Kruskal

- minimaler Spannbaum



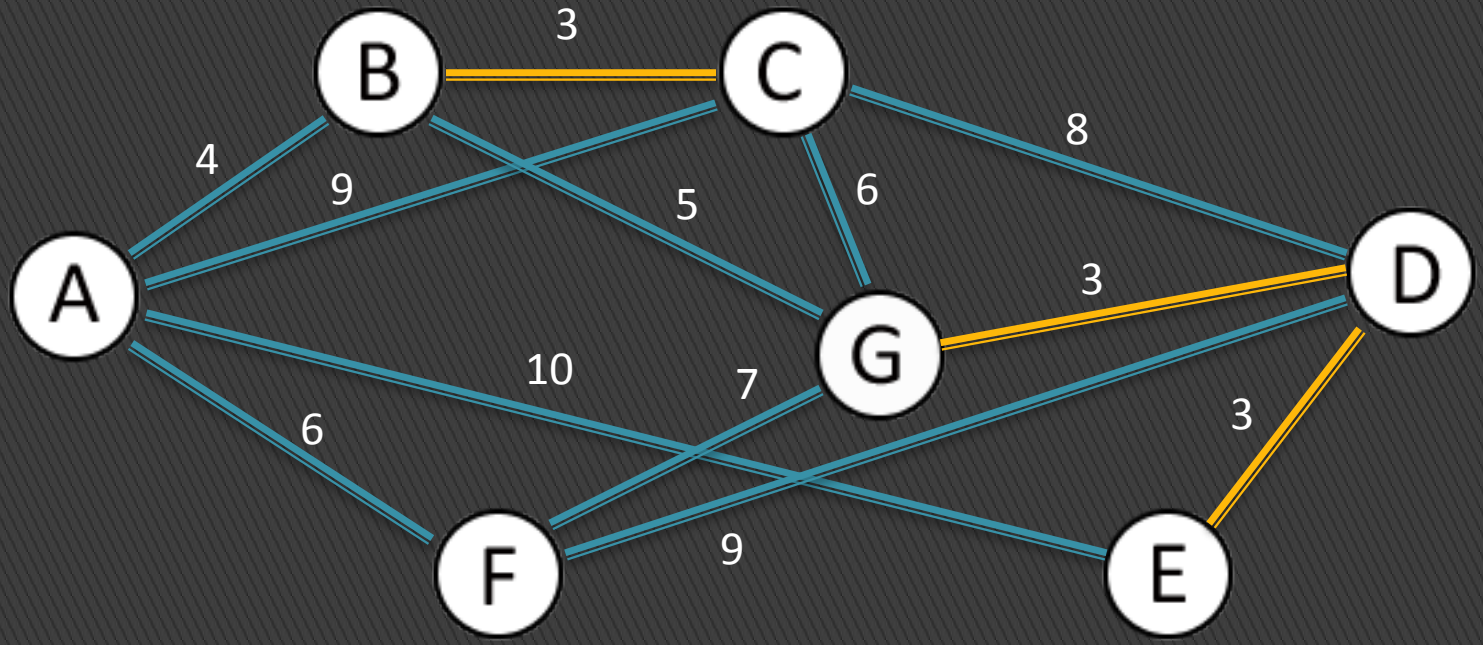
Algorithmus von Kruskal

- minimaler Spannbaum



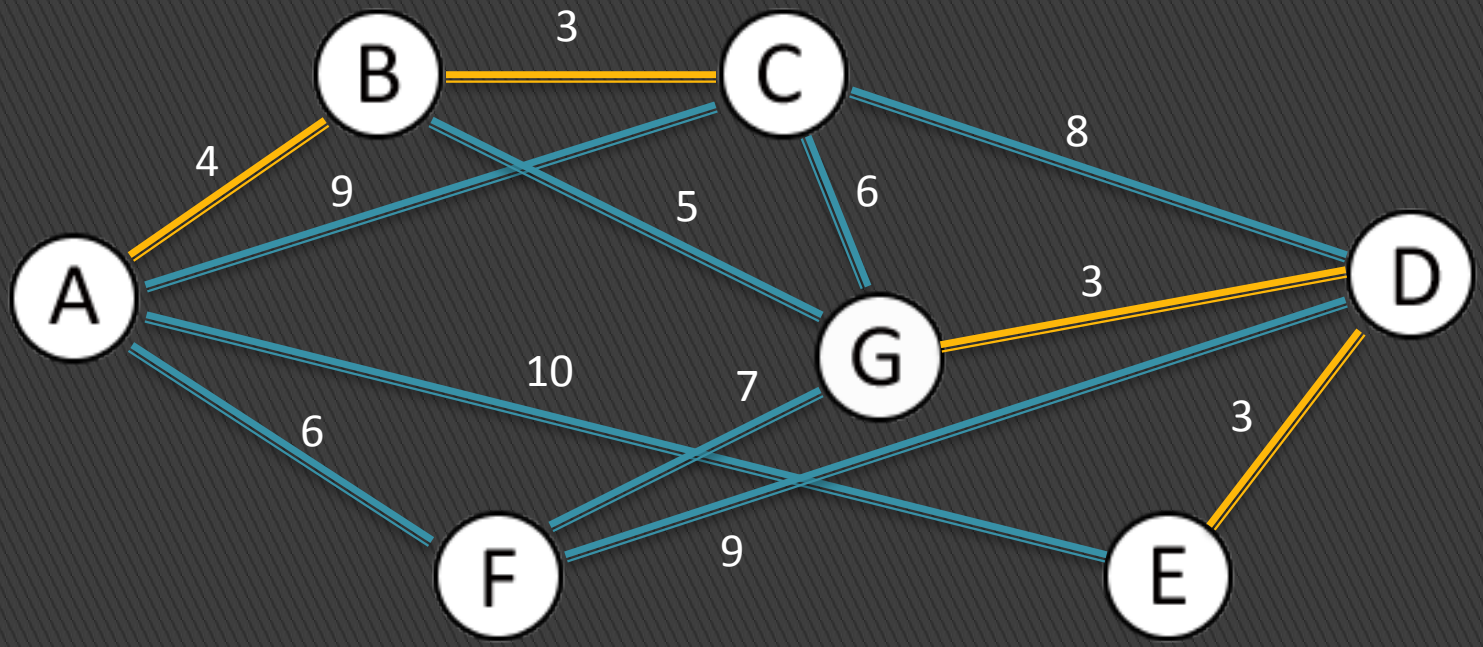
Algorithmus von Kruskal

- minimaler Spannbaum



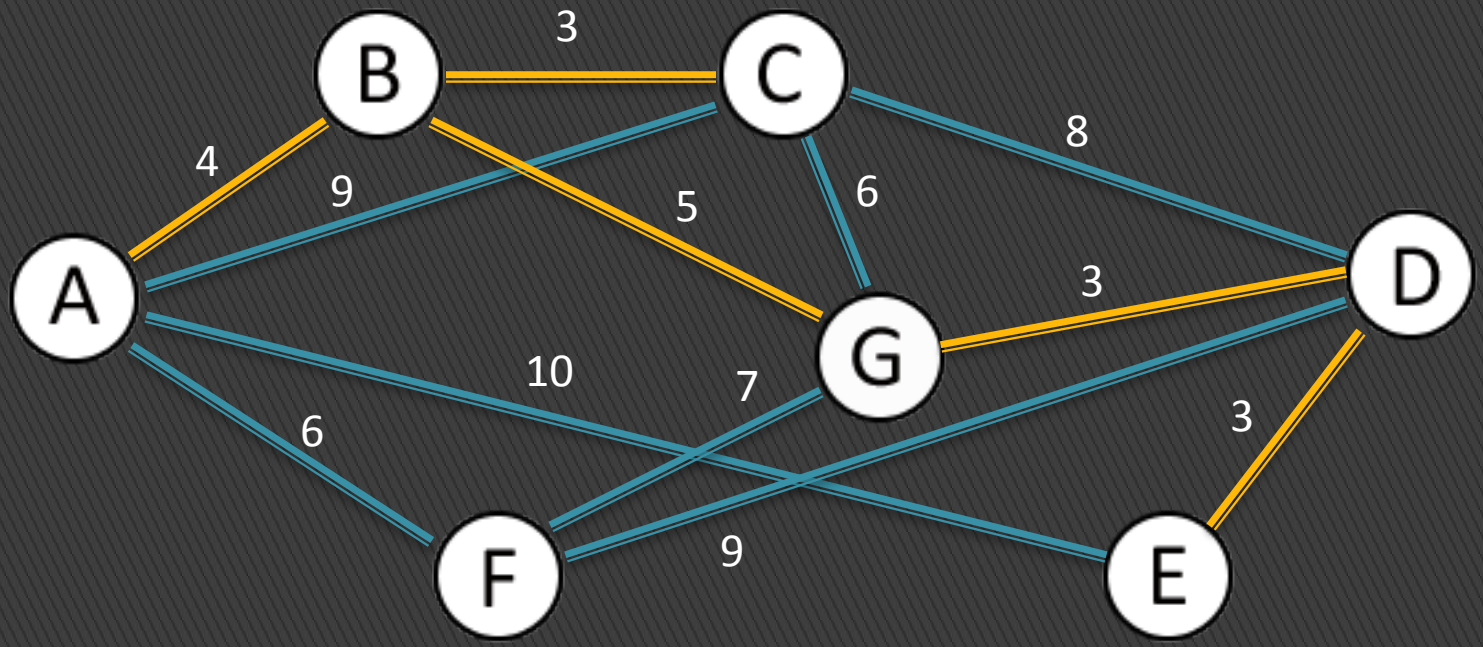
Algorithmus von Kruskal

- minimaler Spannbaum



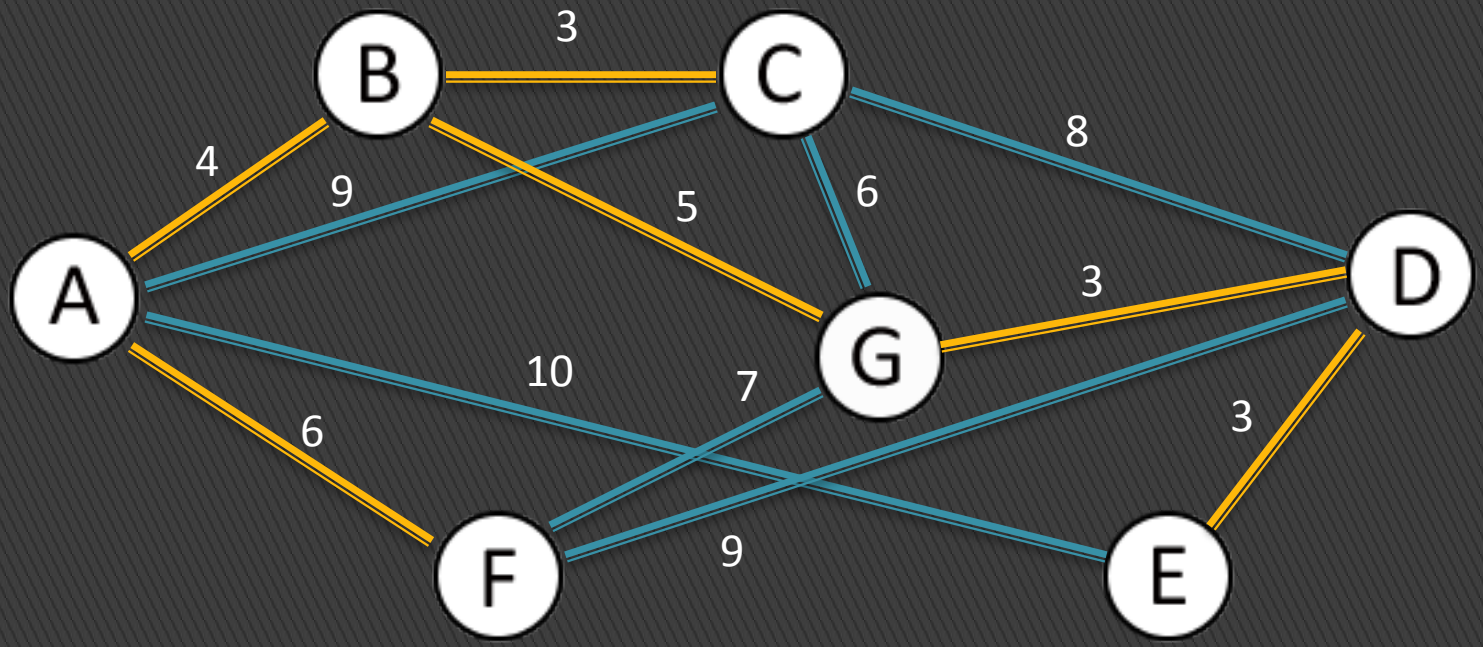
Algorithmus von Kruskal

- minimaler Spannbaum



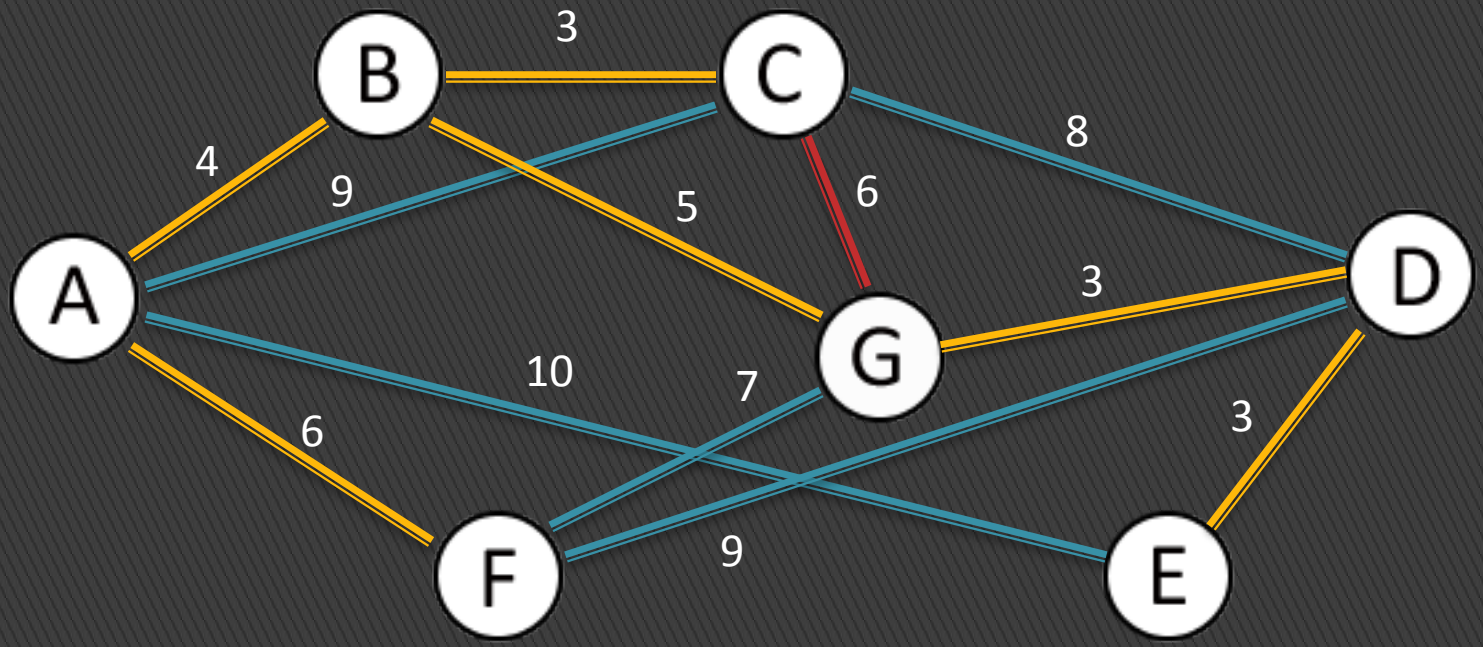
Algorithmus von Kruskal

- minimaler Spannbaum



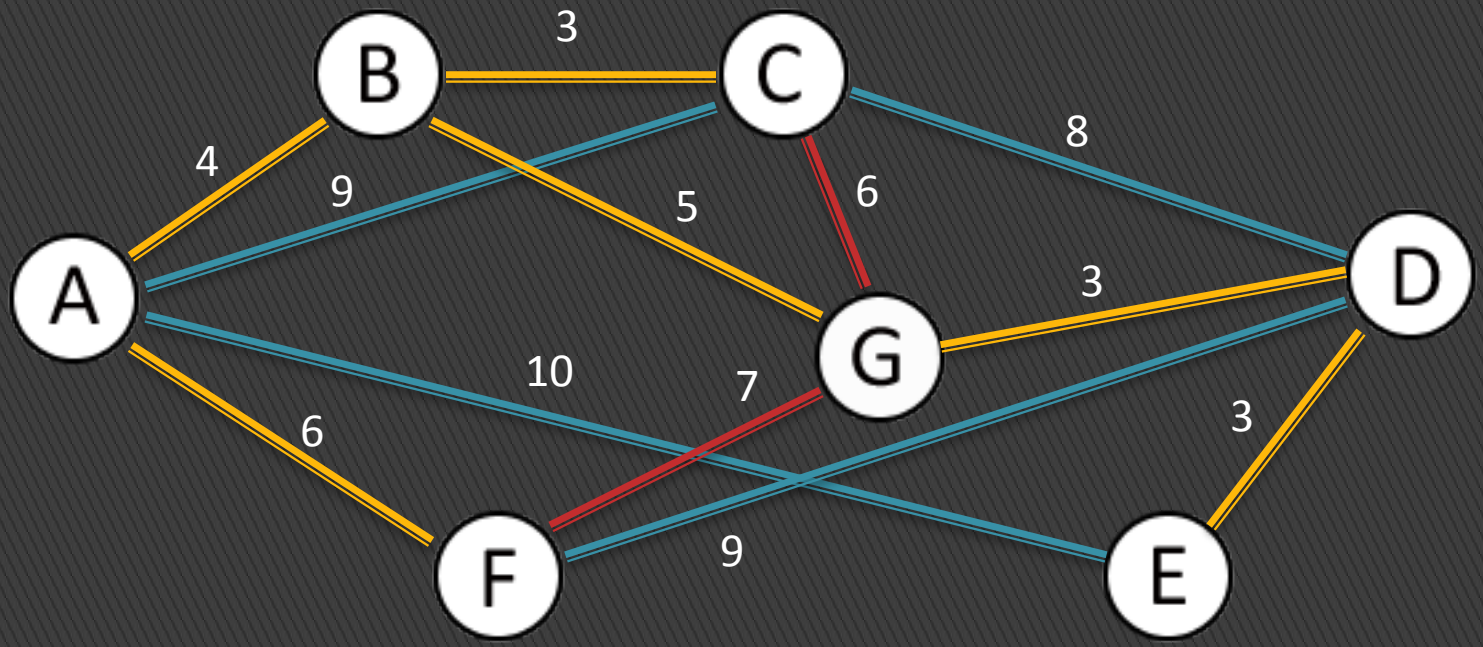
Algorithmus von Kruskal

- minimaler Spannbaum



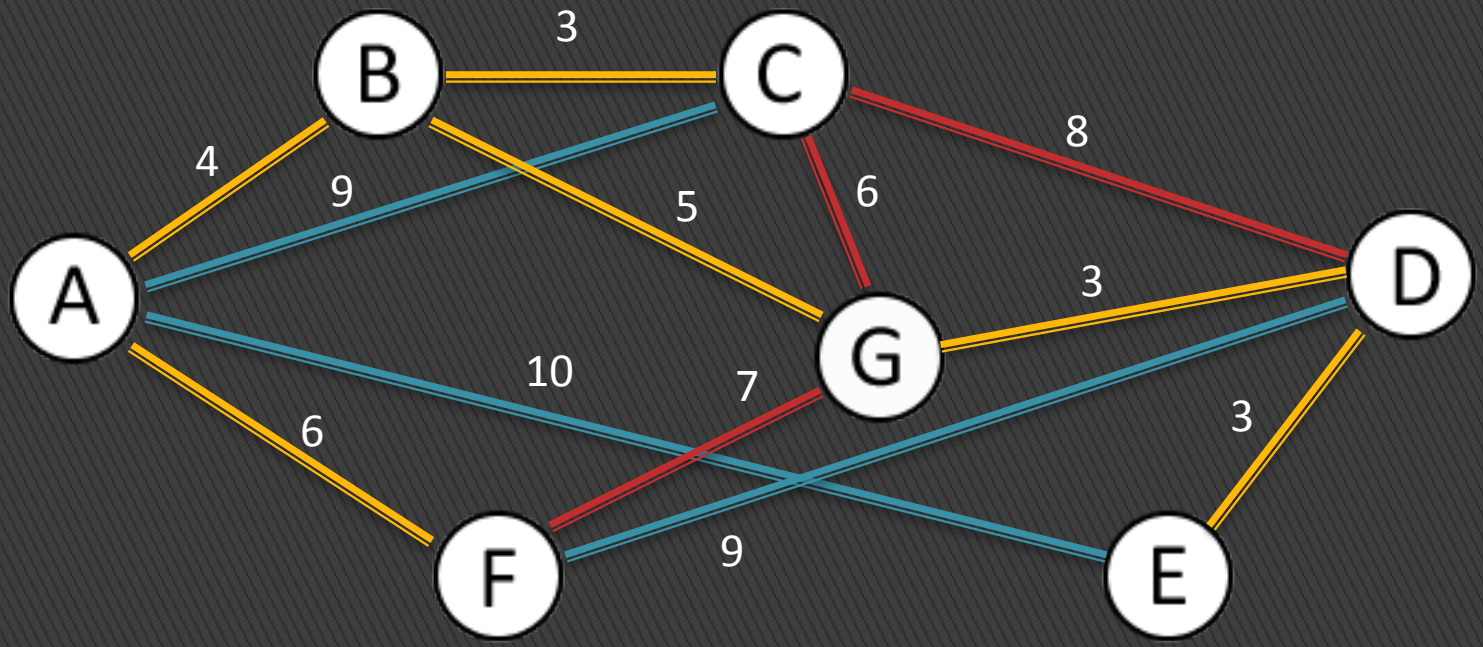
Algorithmus von Kruskal

- minimaler Spannbaum



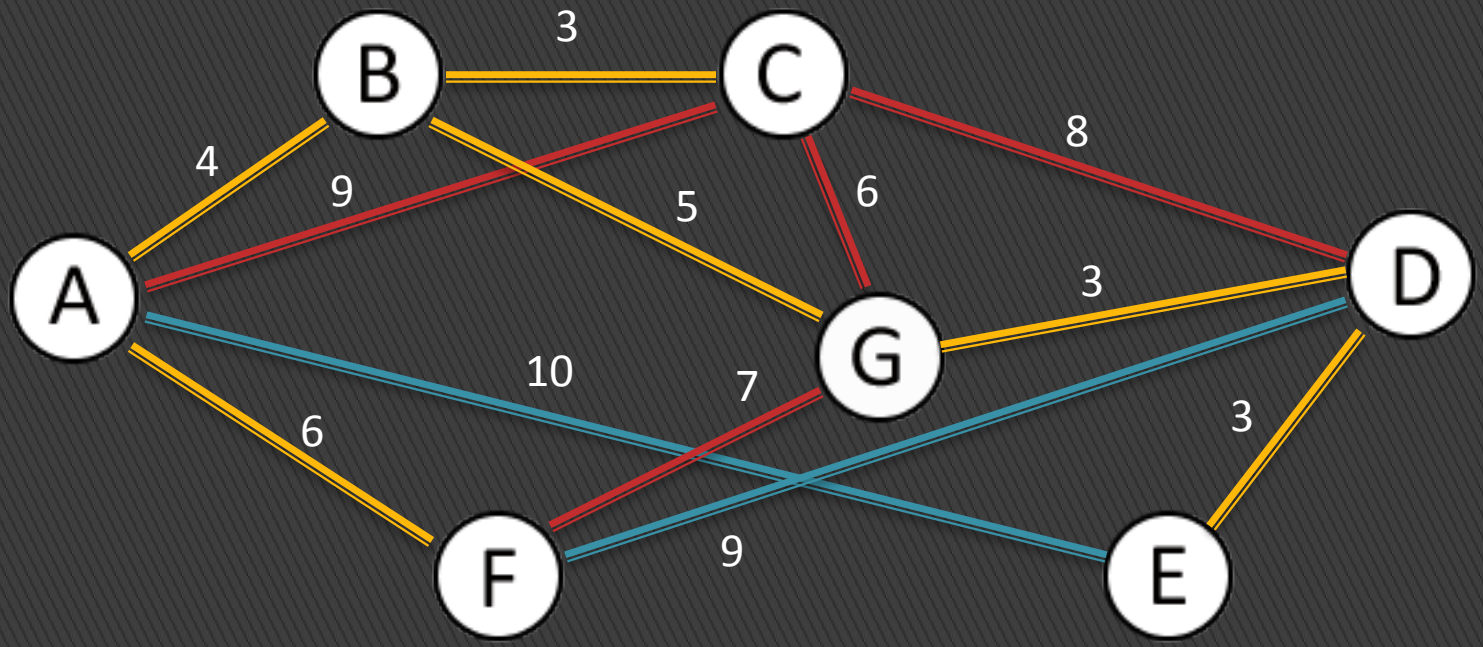
Algorithmus von Kruskal

- minimaler Spannbaum



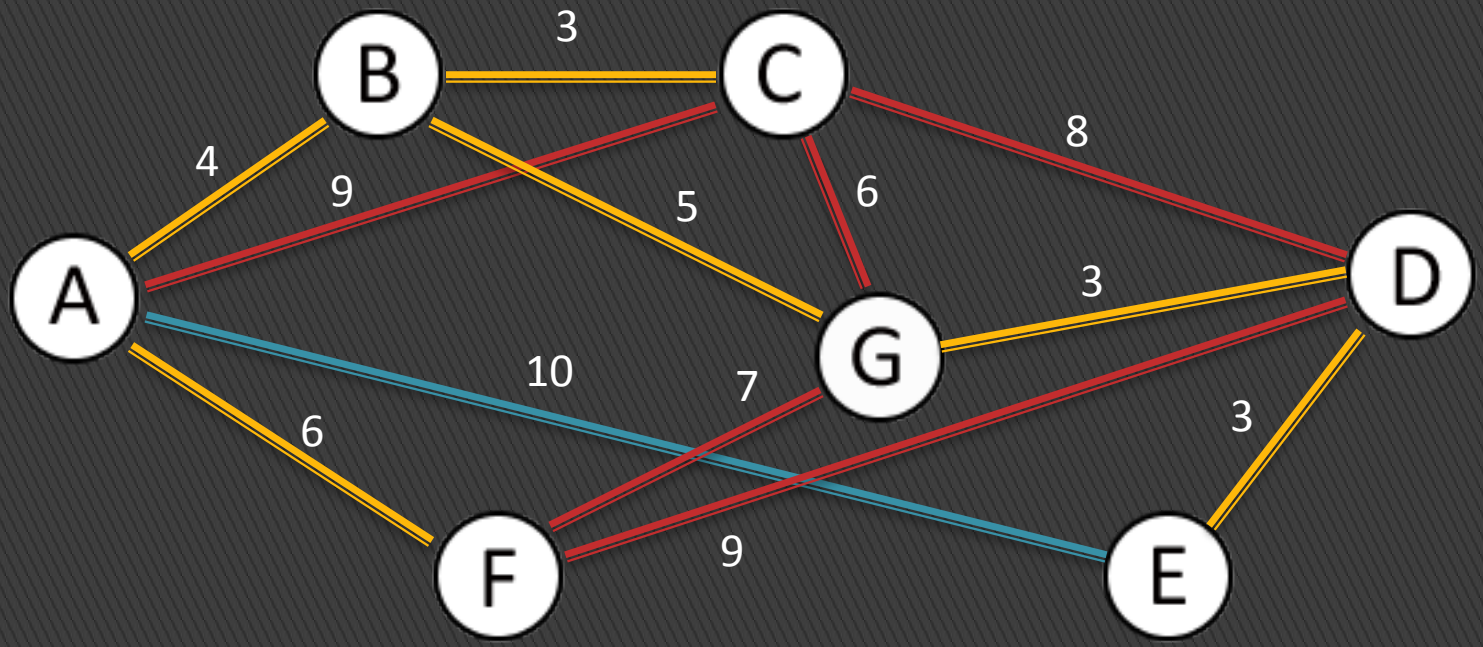
Algorithmus von Kruskal

- minimaler Spannbaum



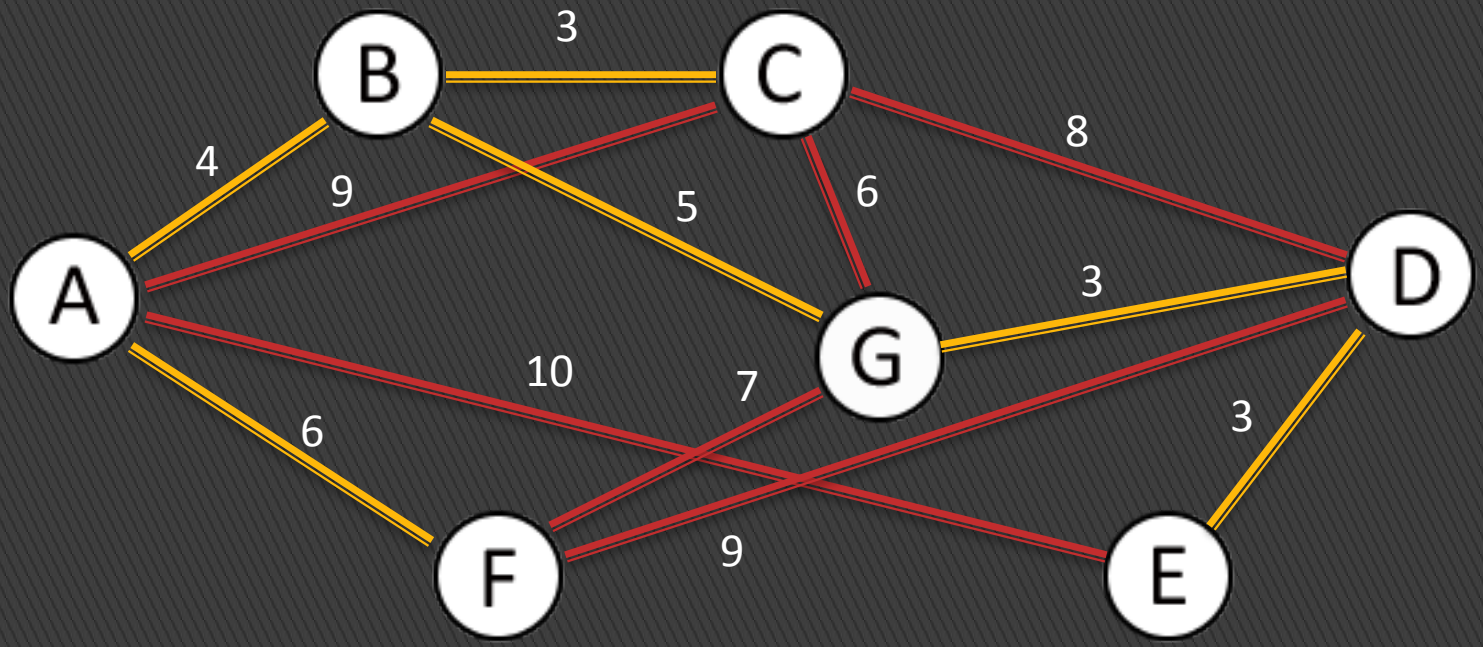
Algorithmus von Kruskal

- minimaler Spannbaum



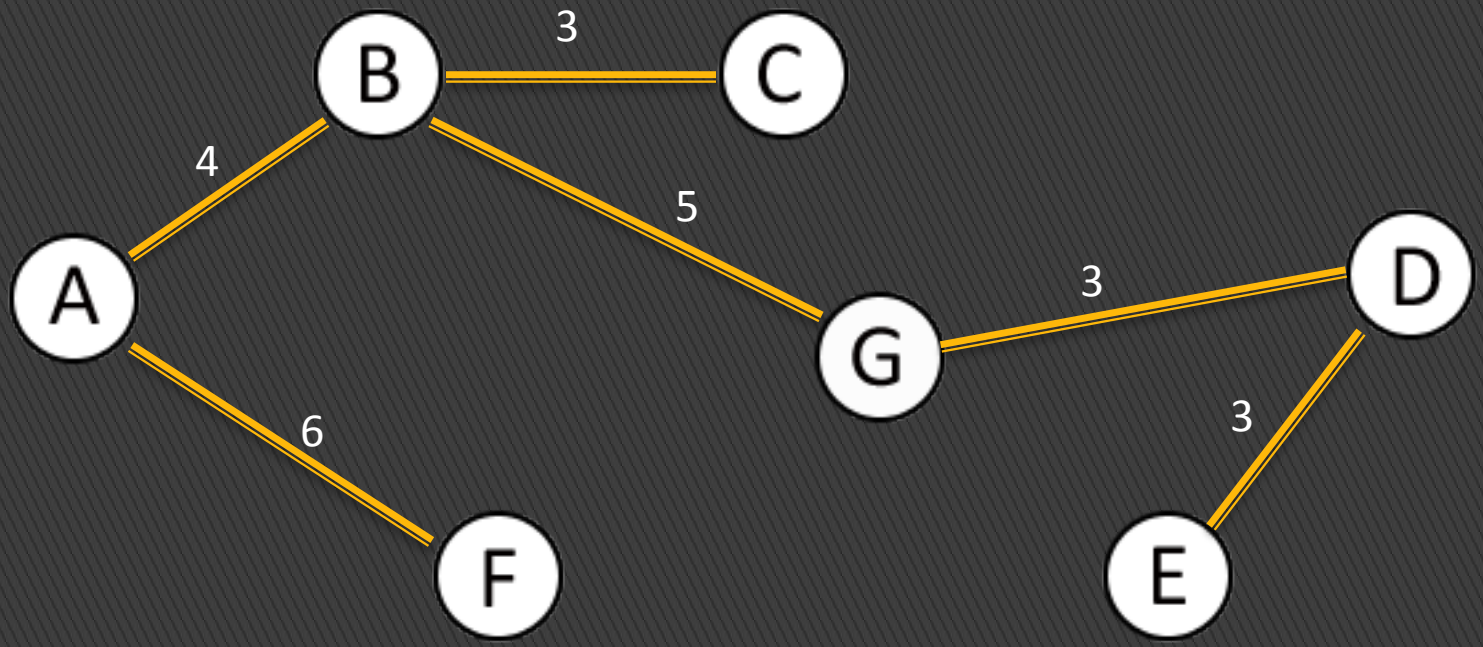
Algorithmus von Kruskal

- minimaler Spannbaum



Algorithmus von Kruskal

- minimaler Spannbaum

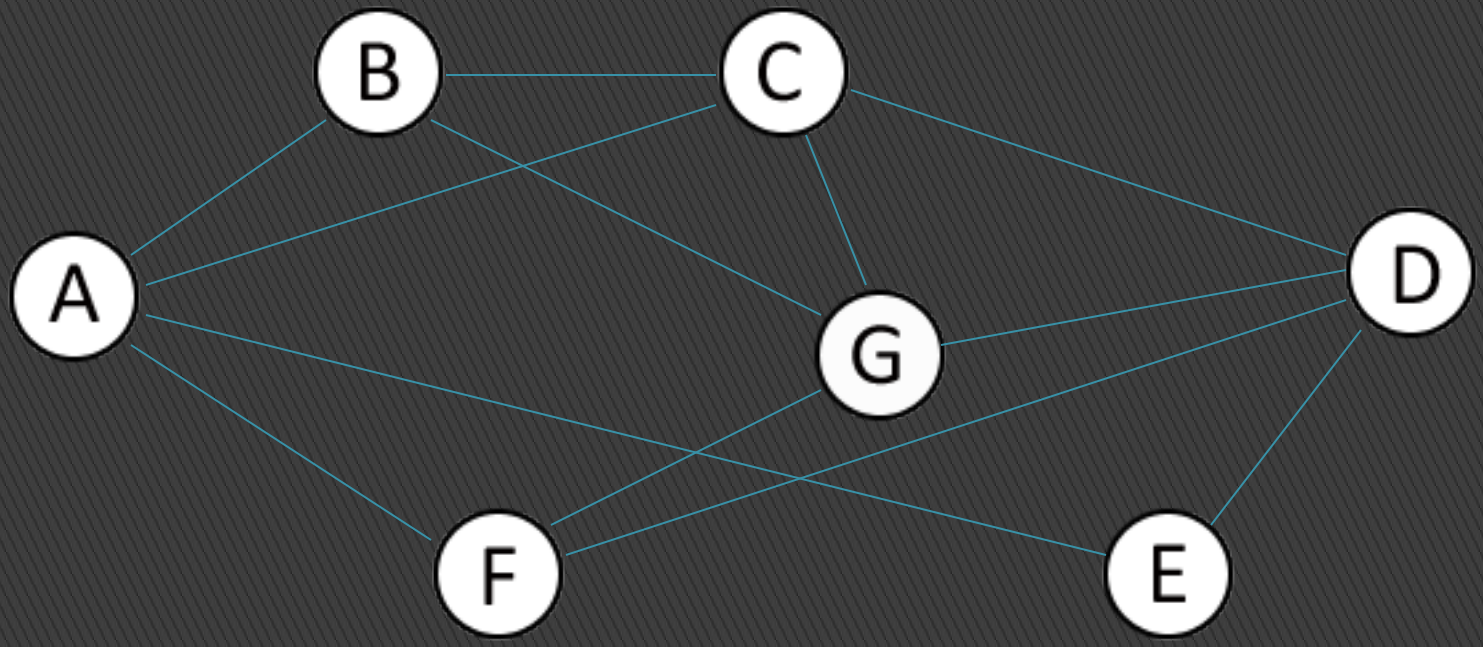


Algorithmus von Prim

- minimaler Spannbaum
  - Alternative zu Kruskal-Algorithmus
  - Modifikation aus DIJKSTRA-Algorithmus
    - änderungen an nur 3 Stellen
  - Effizienter bei Graphen mit vielen Kanten als KRUSKAL-Algorithmus

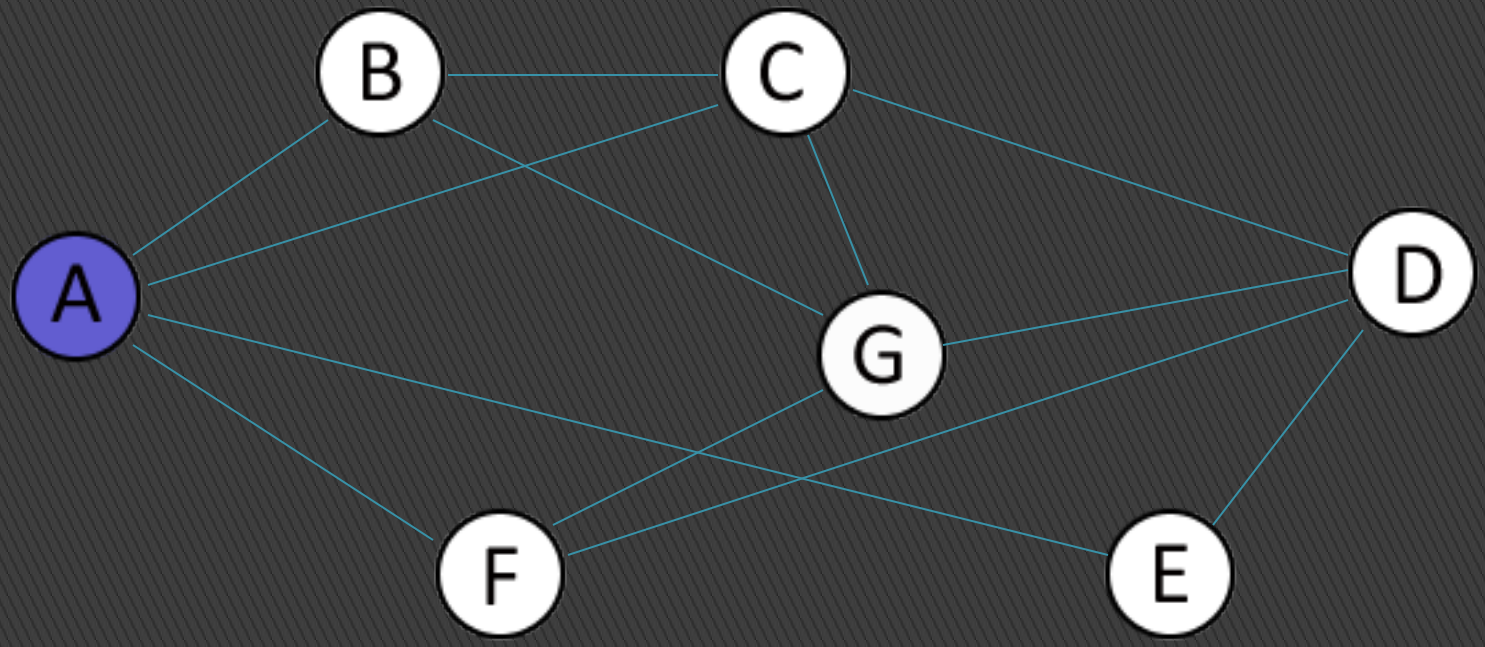
Algorithmus von Prim

- minimaler Spannbaum



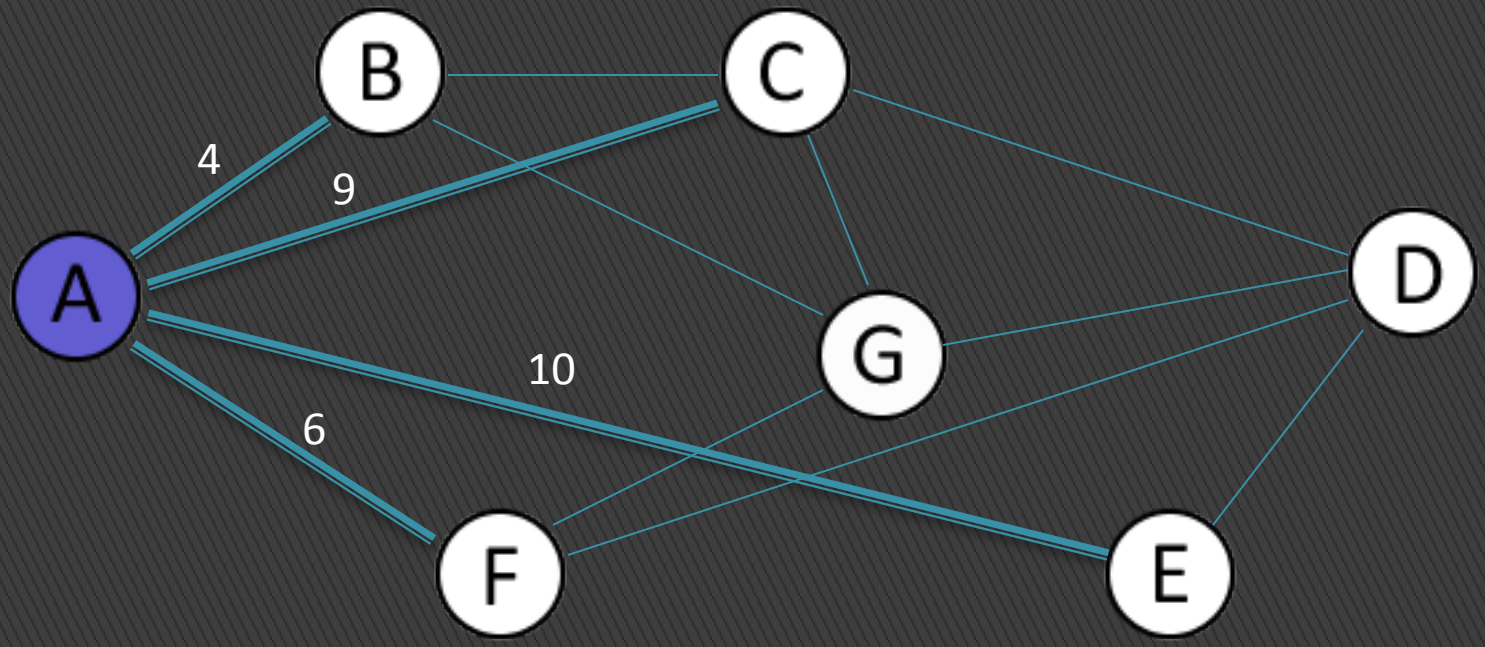
Algorithmus von Prim

- minimaler Spannbaum



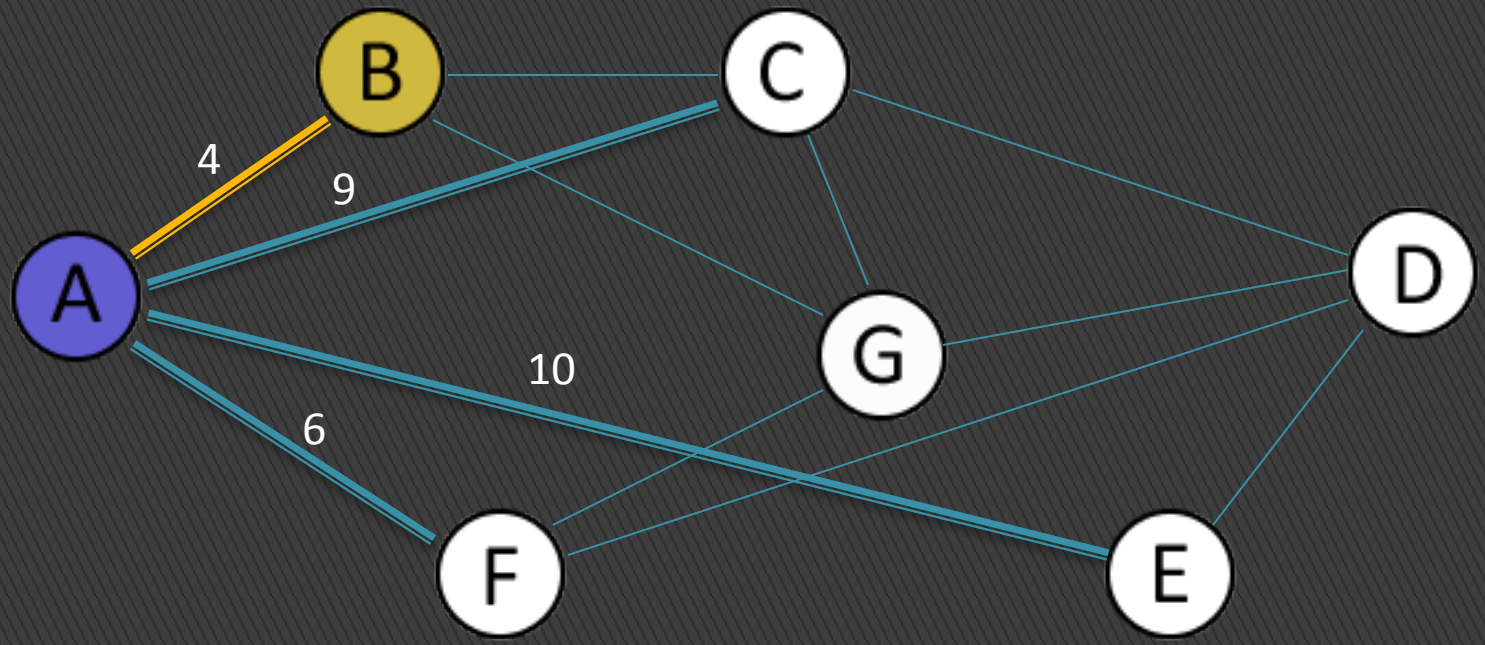
Algorithmus von Prim

- minimaler Spannbaum



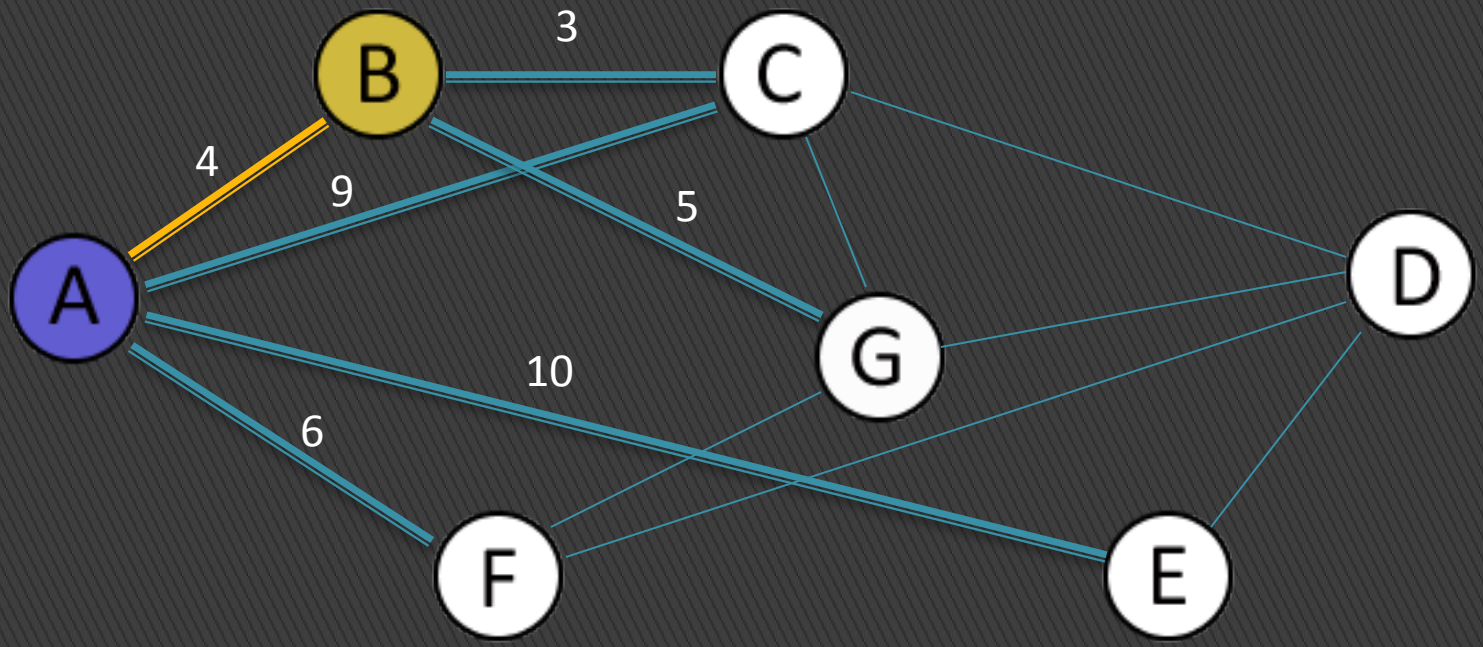
Algorithmus von Prim

- minimaler Spannbaum



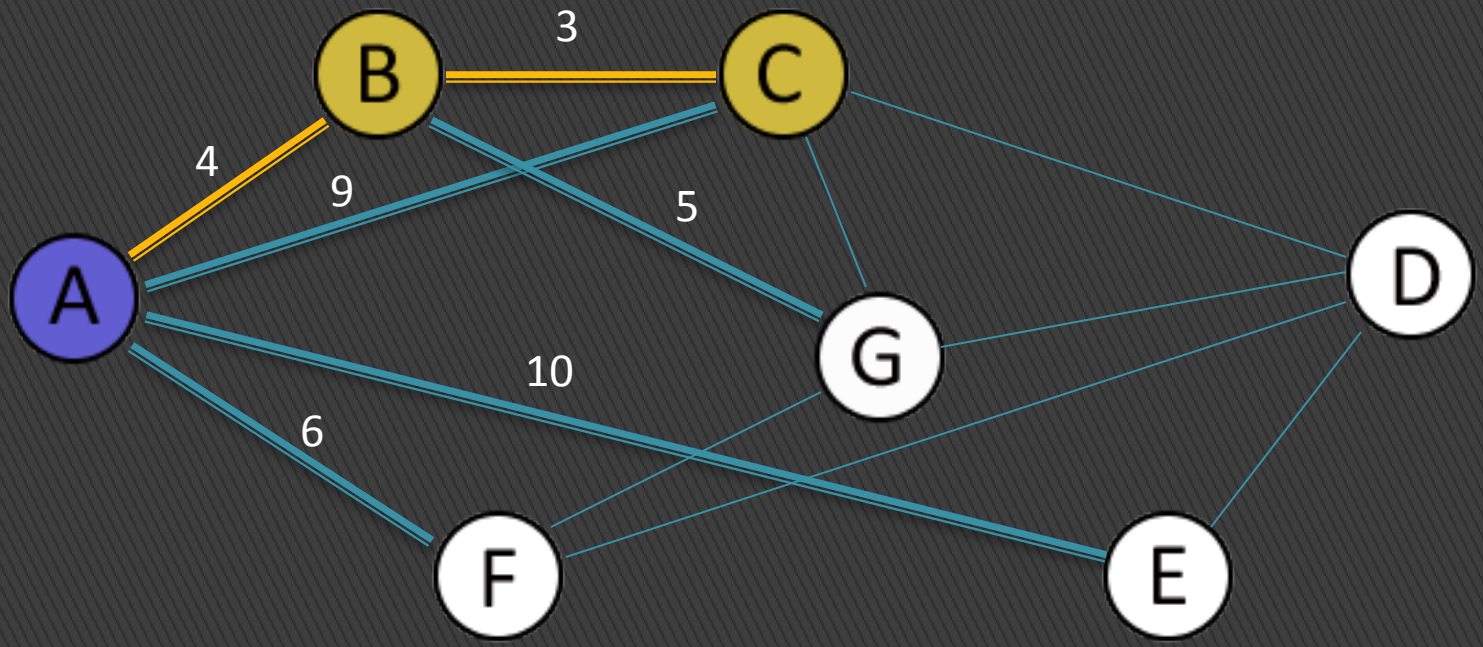
Algorithmus von Prim

- minimaler Spannbaum



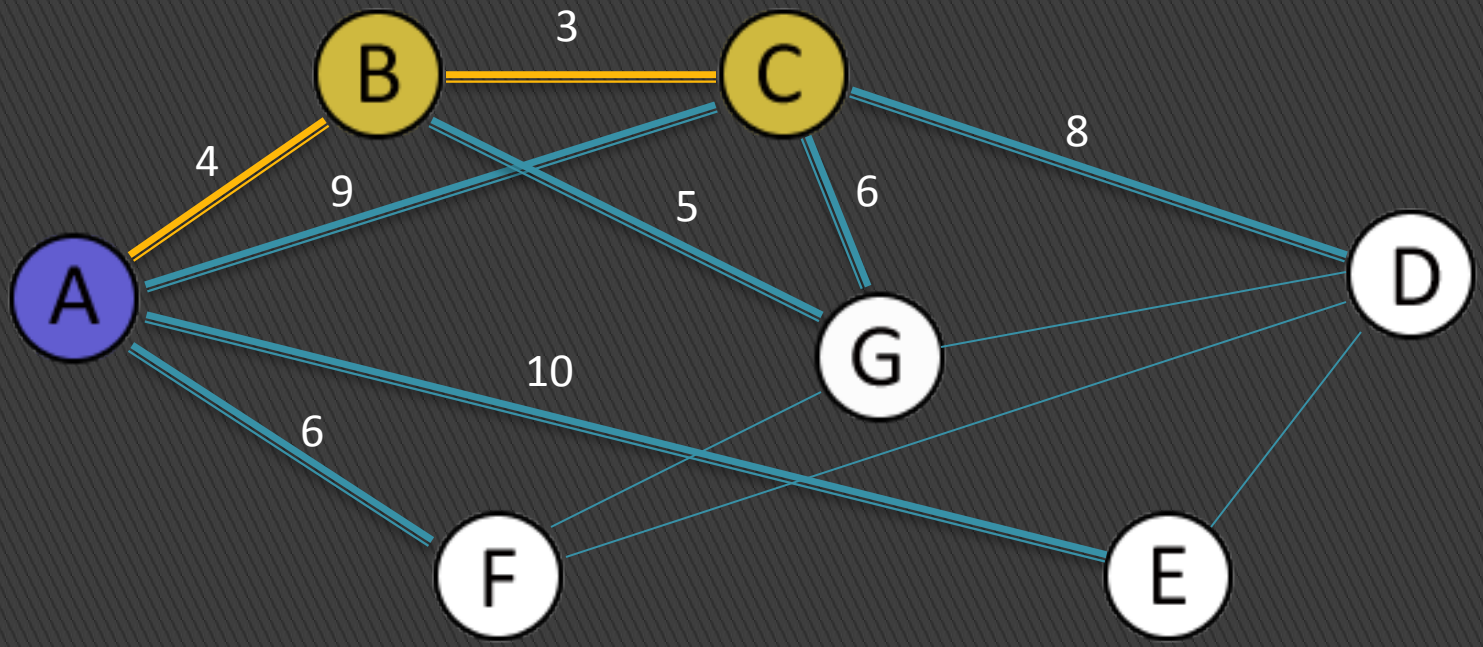
Algorithmus von Prim

- minimaler Spannbaum



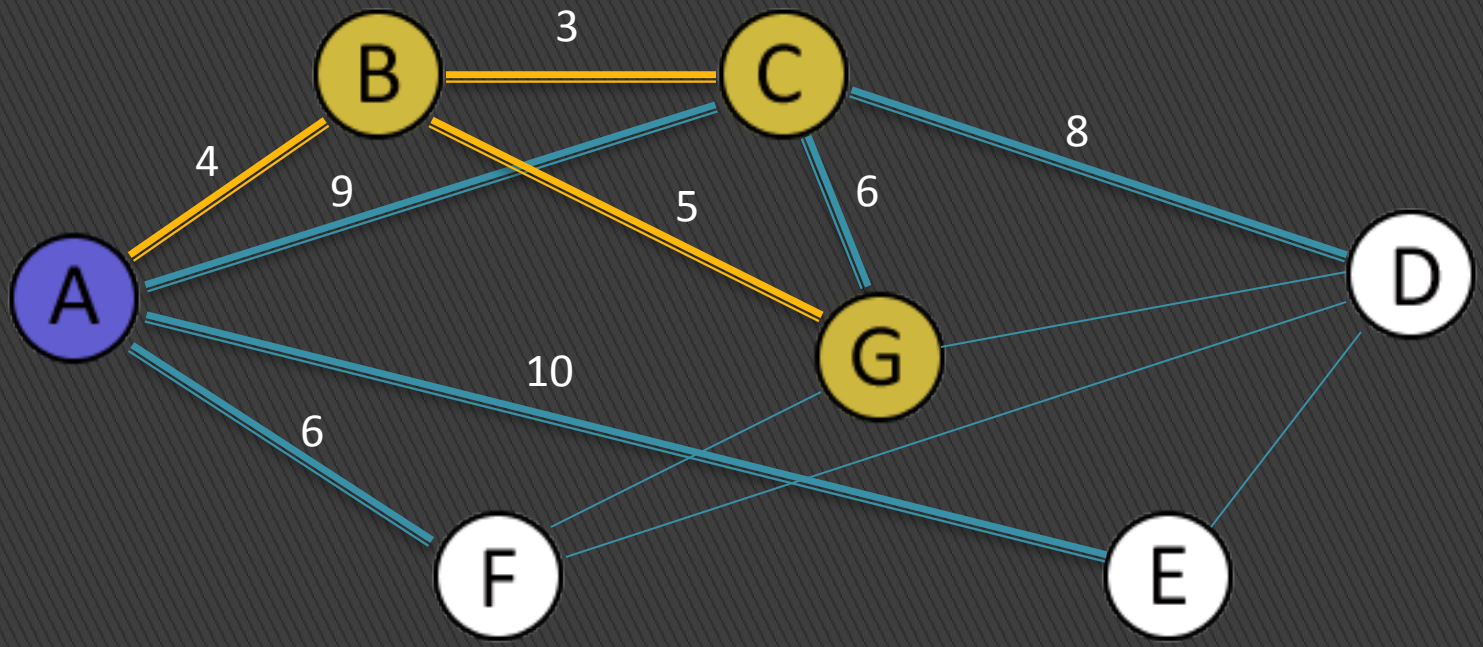
Algorithmus von Prim

- minimaler Spannbaum



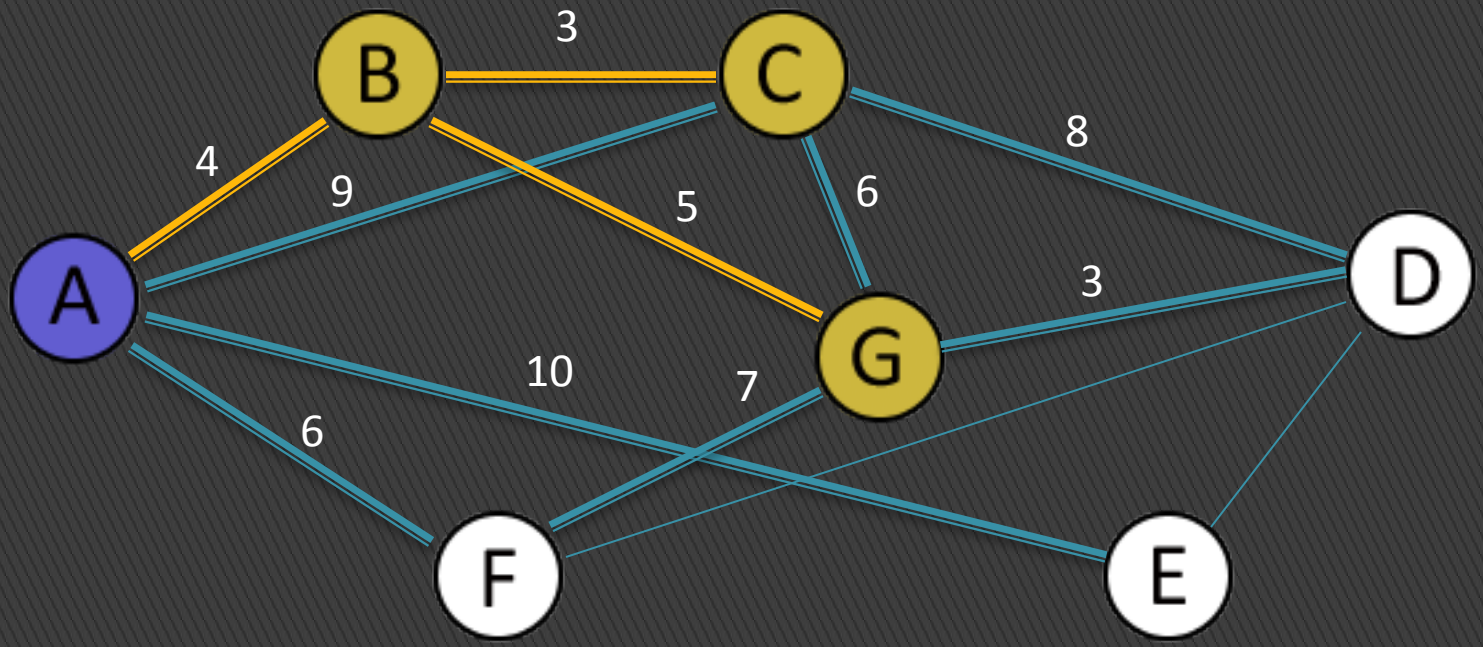
Algorithmus von Prim

- minimaler Spannbaum



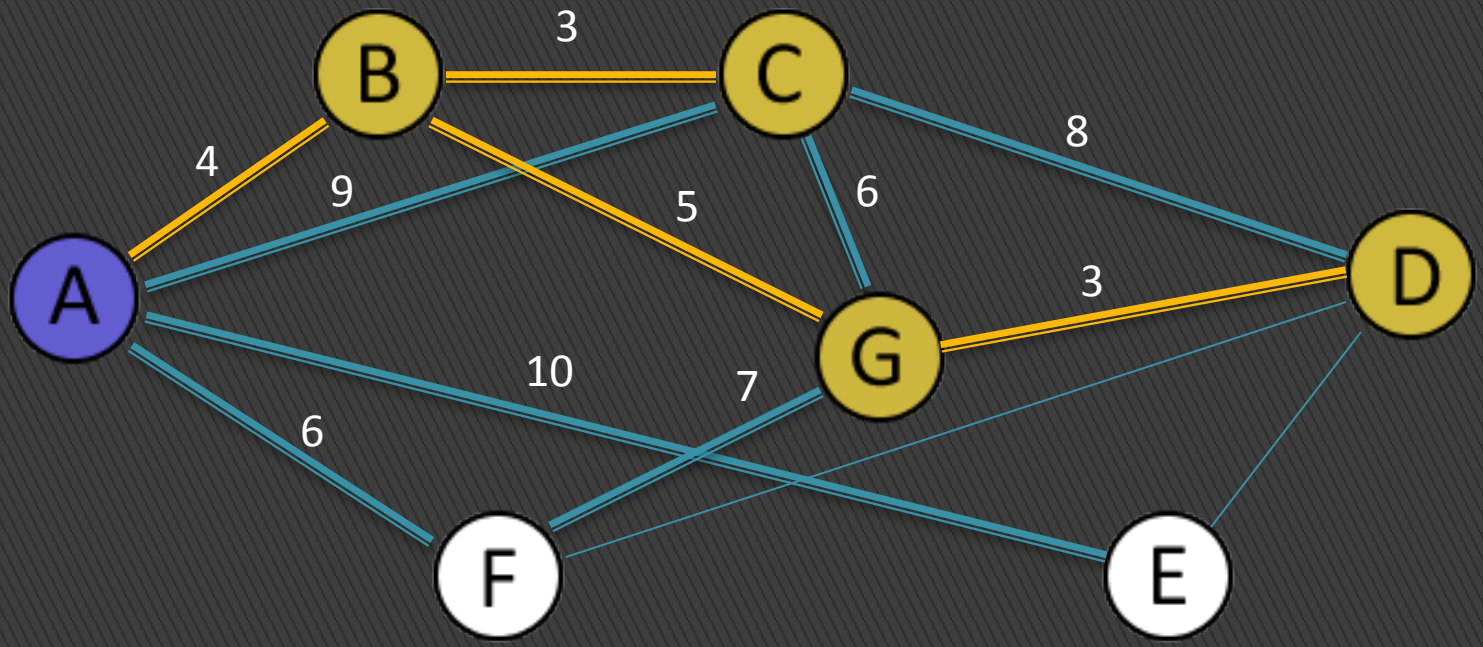
Algorithmus von Prim

- minimaler Spannbaum



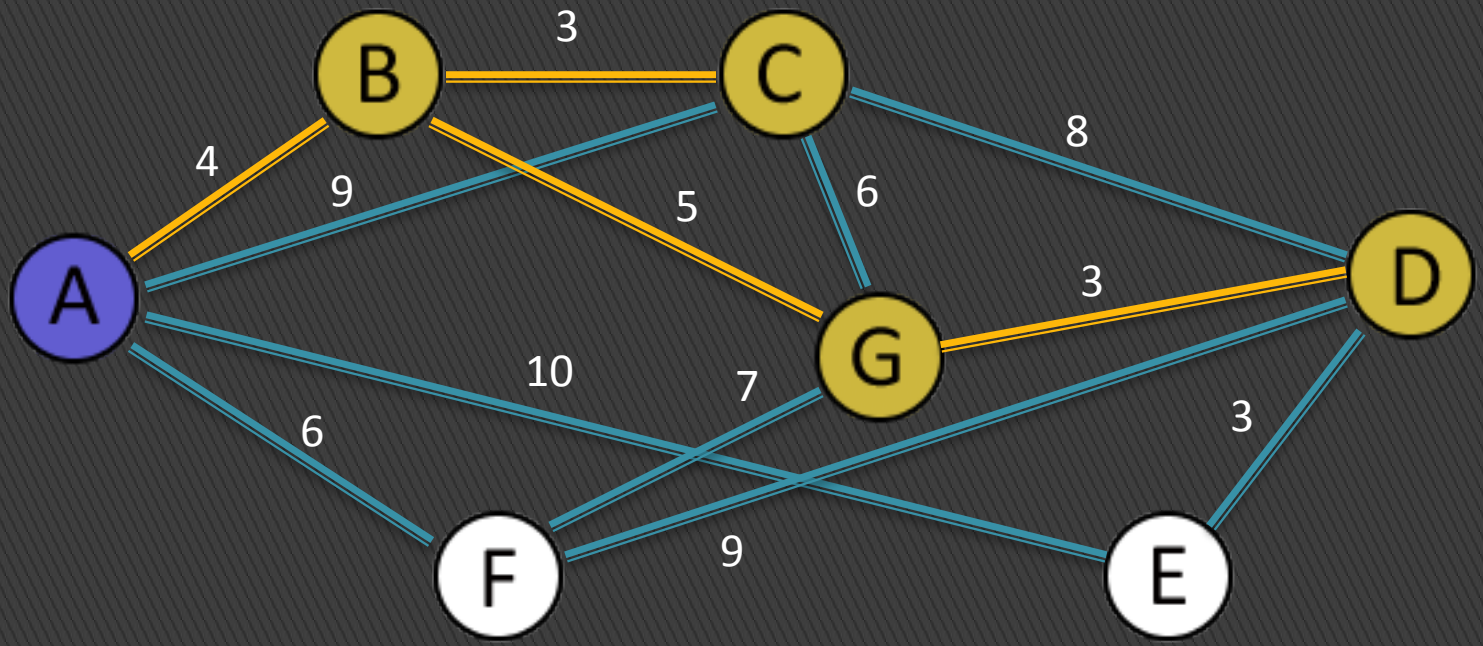
Algorithmus von Prim

- minimaler Spannbaum



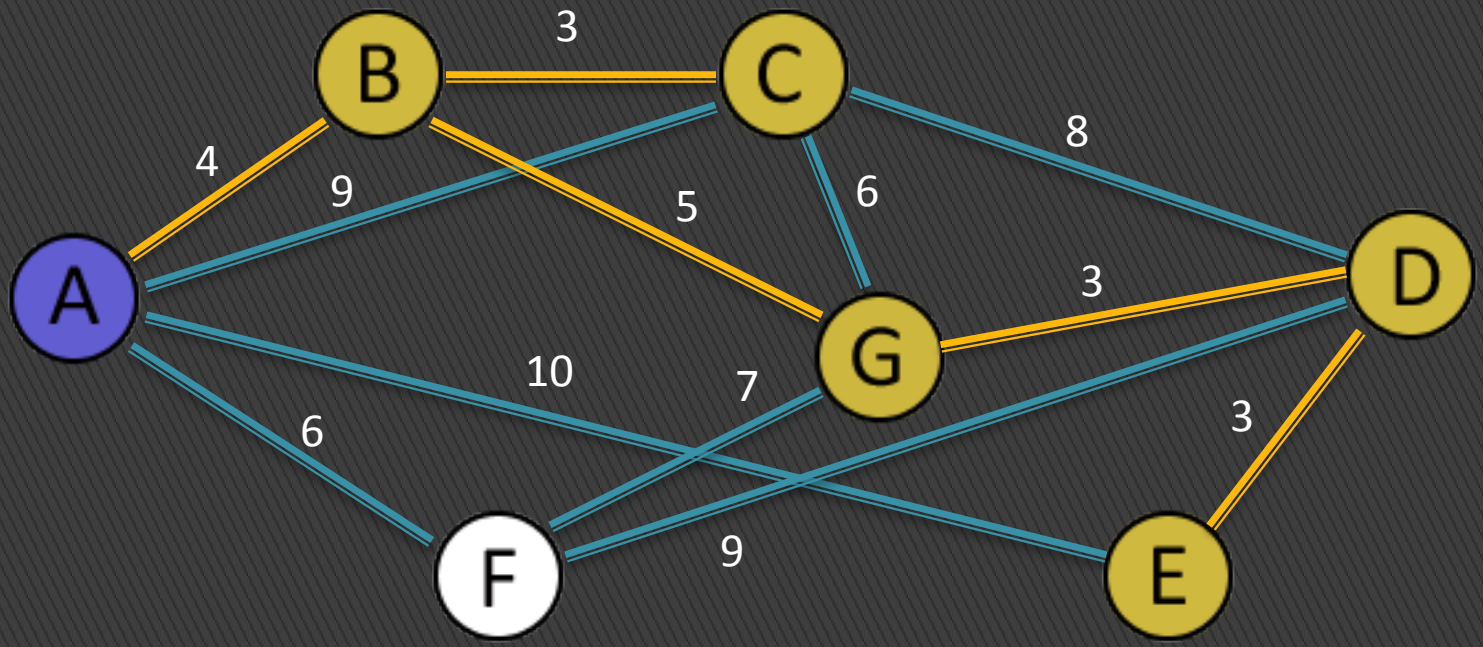
Algorithmus von Prim

- minimaler Spannbaum



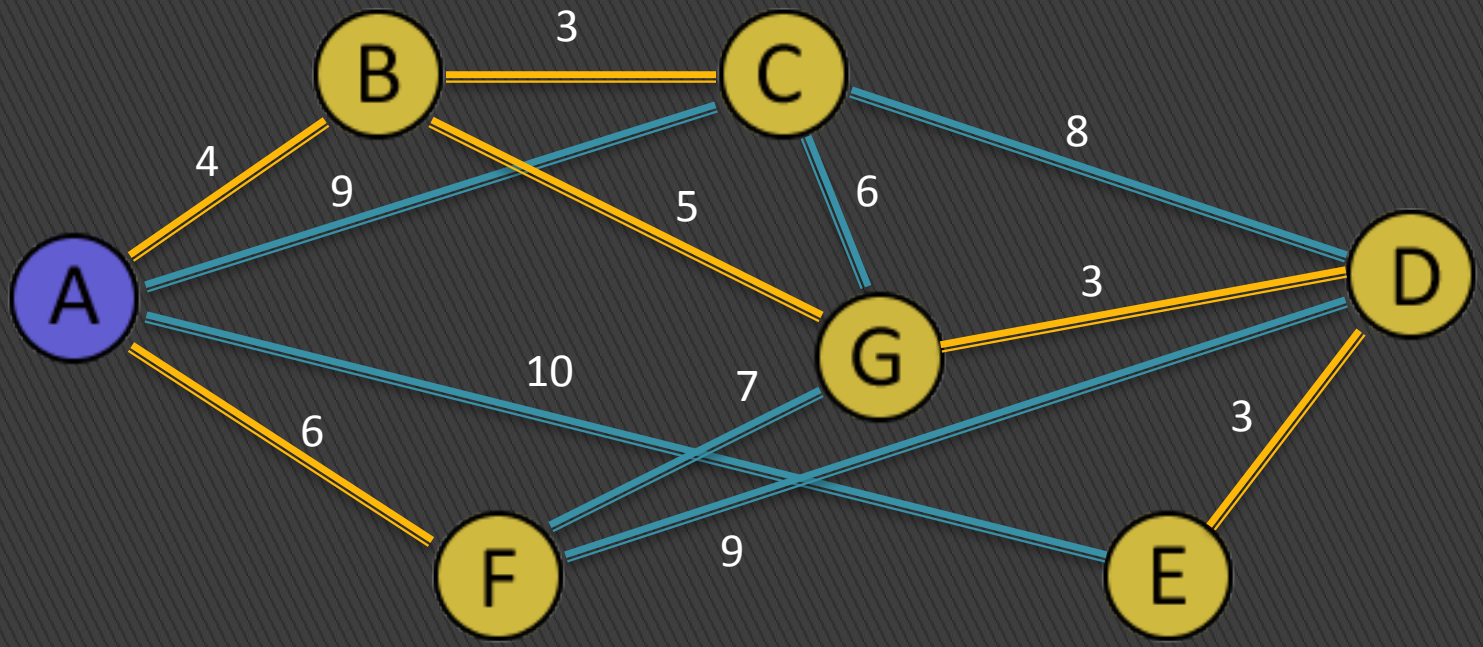
Algorithmus von Prim

- minimaler Spannbaum



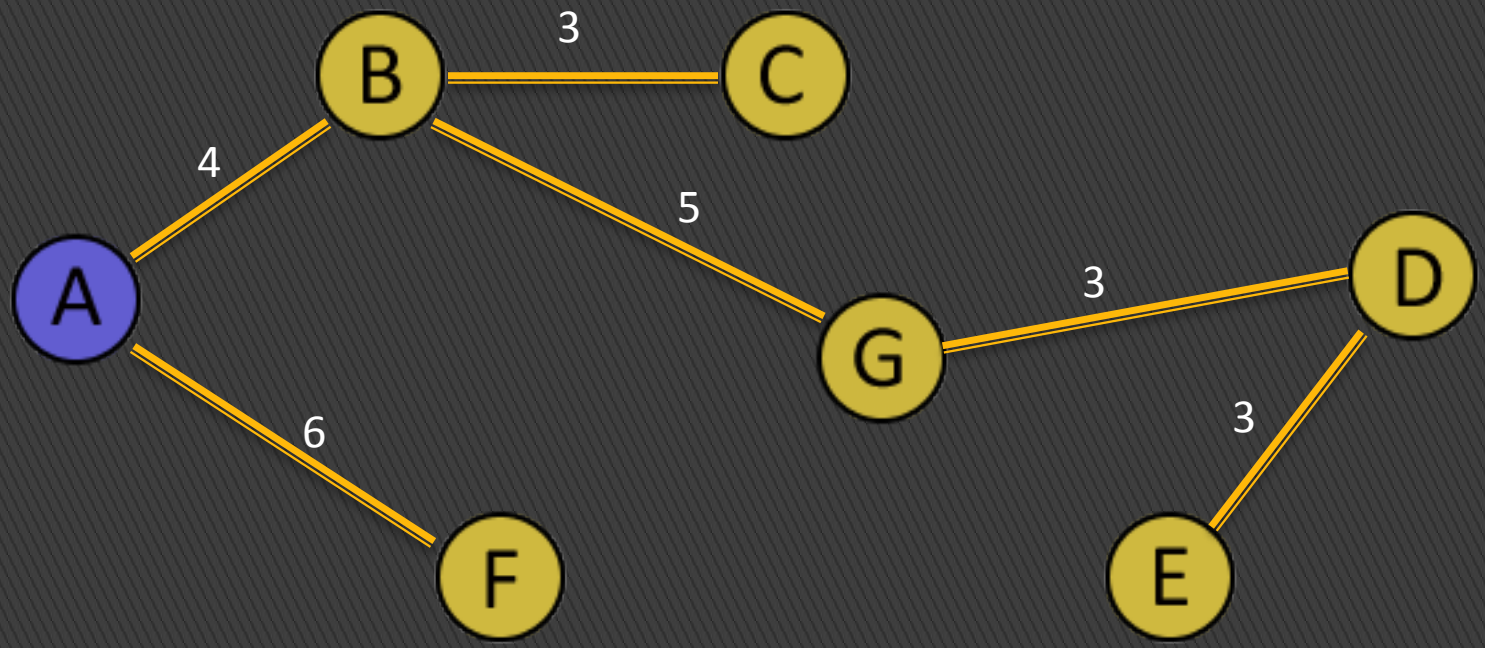
Algorithmus von Prim

- minimaler Spannbaum



Algorithmus von Prim

- minimaler Spannbaum



Danke für Ihre Aufmerksamkeit

Übungen sind im Wiki zu finden.