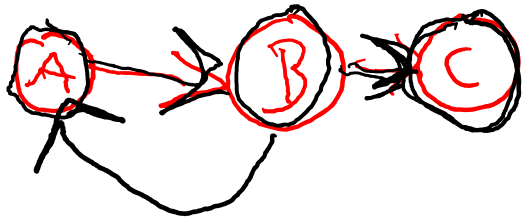
Algo-Tutorium 7

08.12.2021, Lukas Weber

Starke Zusammen-hangs-komponente



Definition:

- A ist die maximale Knotenteilmenge, deren Knoten untereinander alle erreichbar sind.
- $A \subseteq V: \forall u, v, \in A: \exists\, path(u, v)$

(und damit auch: $path(v, u)$, da Graph ungerichtet)

Auffinden von CC?

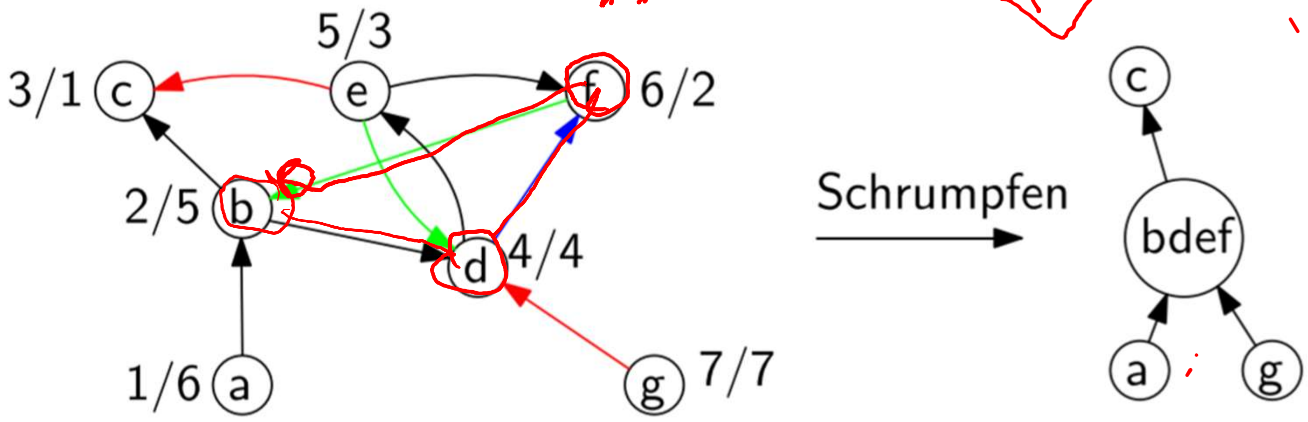
→ Graph explorieren!

Beispiel

- Was ist der Unterschied zu einer Zusammenhangskomponente?

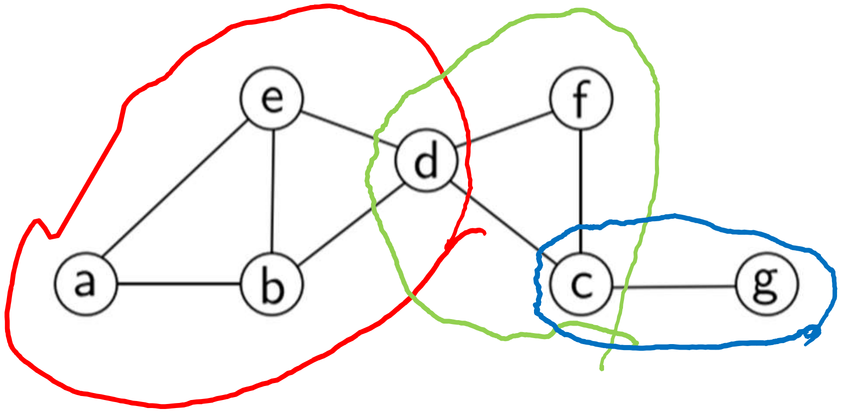
a → b → c

- Beispiel einer starken SZK



2ZK

1. Ein ungerichteter Graph G = (V, E) heißt 2-fach zusammenhängend, falls für alle v ∈ V auch G\{v} zusammenhängend.
2. Eine 2ZK eines ungerichteten Graphen ist ein maximaler 2-fach zusammenhängender Teilgraph.



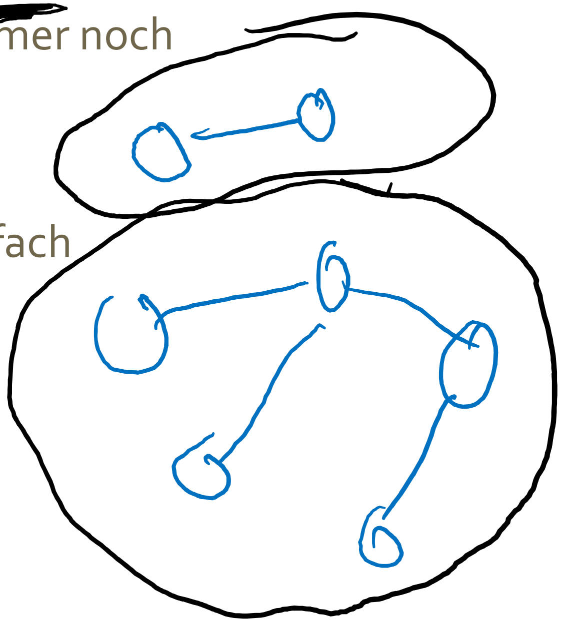
2ZKs:
{a,b,e,d}, {d,c,f}, {c,g}

Artikulationspunkte:
c, d

k-facher Zusammenhang

Definition:

- Ein ungerichteter Graph $G = (V, E)$ ist k-fach zusammenhängend, wenn es keinen Trenner $T = (A, B)$ in G mit einer maximal $(k-1)$-elementigen Knotenmenge A und leerer Kantenmenge $B = \emptyset$ gibt.
- Sprich: Wir dürfen max. Menge A, $|A| = k - 1$ und keine Kanten aus G entfernen und dann soll der Graph immer noch zusammenhängen.



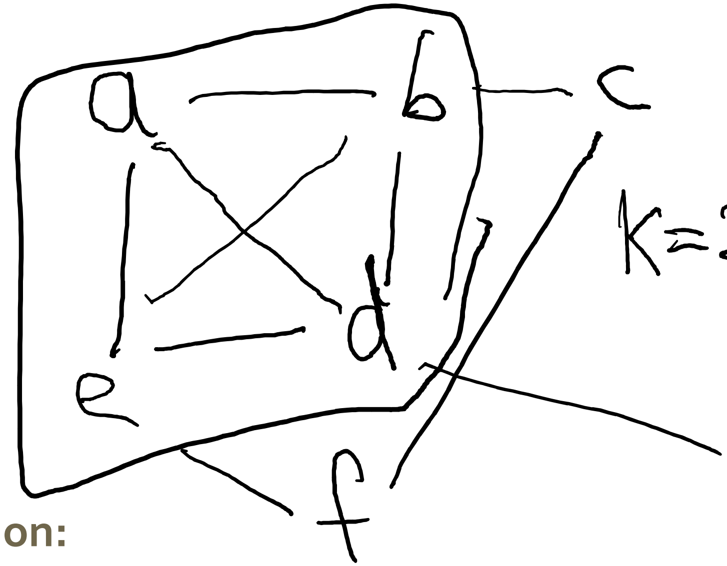
- Welche Graphen mit $k = |V|$ sind immer k-fach zusammenhängend?

Vollständige Graphen!

Und alle anderen ☺

$|V| - (|V| - 1) = 1$

k-fache CC



$|V| = 6$

$k=3 \Rightarrow |A| = k-1 = 2$

$3 \geq k$

Definition:

- Maximale Knotenmenge $A \subseteq V$, sodass A k-fach zusammenhängend ist.

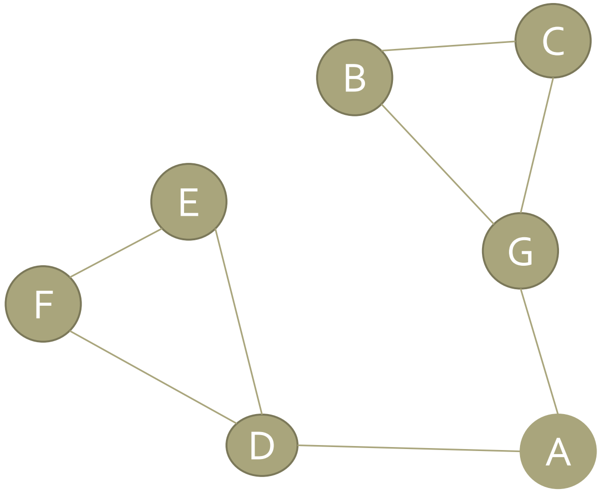
Artikulation, Brücke

Artikulationspunkt v:

- $G\backslash\{v\}$ ist nicht mehr zusammenhängend.

Brücke e:

- $G\backslash\{e\}$ ist nicht mehr zusammenhängend.



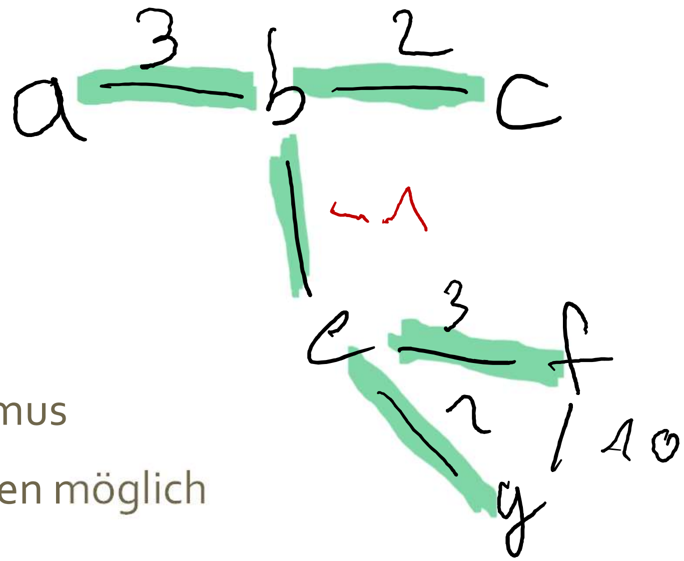
Artikulationspunkt?

Brücke?

MST

- Minimal aufspannender Baum
- Erreichen jedes anderen Knotens über minimale Weglänge
- Algorithmen:
  1. Kruskal
  2. Prim‘s Algorithmus

Kruskal



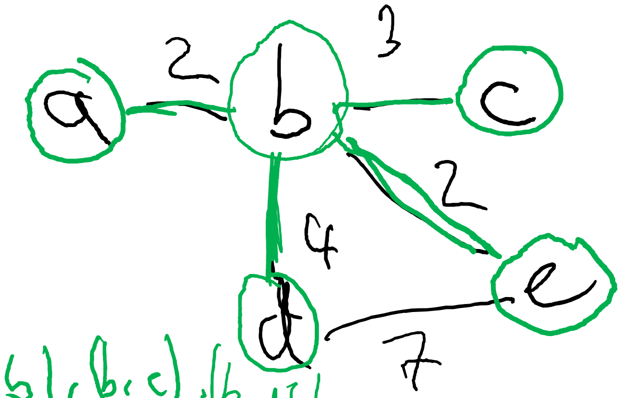
- Greedy-Algorithmus
- Neg. Kantenkosten möglich
- Naiv:

  Ordne Kanten $e_1, \ldots, e_m$ so dass $c(e_1) \leq c(e_2) \leq, , , \leq c(e_m)$

  Baumkanten $E_T \leftarrow \emptyset$;
  for all $(i = 1, \ldots, m)$ {
      if $(E_T \cup \{e_i\}$ azyklisch$)$ $\{E_T \leftarrow E_T \cup \{e_i\}\}$
  }

Prim

- Nur pos. Kantenkosten



$E_T \leftarrow$ {(b,e),(a,b),(b,c),(b,d)}

$T \leftarrow$ {b,e,a,c,d}

while $(T \neq V)$ {

  finde $e = (u, v)$ mit $u \in T$, $v \notin T$ und $c(e)$ minimal;

  $E_T \leftarrow E_T \cup \{e\}$;

  $T \leftarrow T \cup \{v\}$;

}

Cluster

- Aufteilung von V in disjunkte Teilmengen
- Formal:

$$C \in P(V) \text{ mit } \forall C_1, C_2 \in C: C_1 \cap C_2 = \emptyset\ ,$$

$$\text{bzw. } \bigcup_{C_i \in C} C_i = V$$

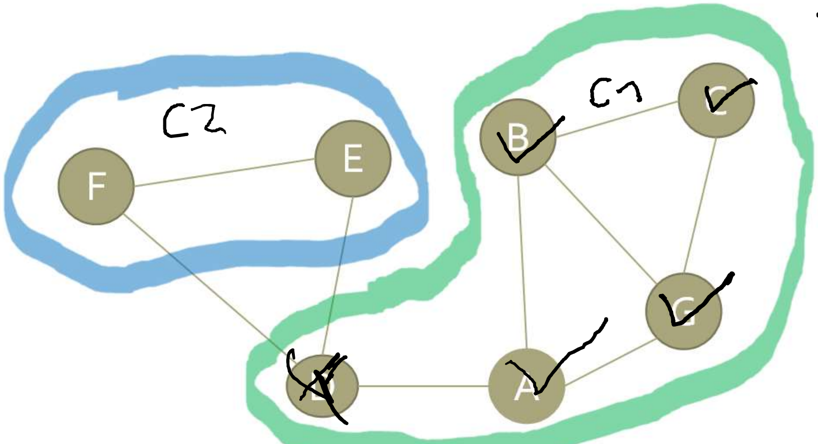
- wollen oft „Zusammenhänge“ clustern (also bspw. CC als Cluster)
- Was aber, wenn der Graph selber schon CC? → abwägen (PÜ)

Def. von $f(C)$ und $g(C)$ (hier)

- $f(C) = \sum_{C_i \in C} \max\{|E(C_i)| - 2 \cdot ||C_i| - 5|, 0\}$
- Dichte ≈ Kanten im Cluster zählen (solange Cluster von richtiger Größe)
- 5 weil: In jedem Cluster sollen möglichst genau 5 Elemente sein
- $g(C) = \sum_{C_i \in C} |\{v \in C_i | \nexists C' \in C: |N(v) \cap C_i| < |N(v) \cap C'|\}|$
- „Spärlichkeit" ≈ Maß für Kanten zwischen den Clustern
- Viele Kanten zwischen Clustern deuten darauf hin, dass einige Knotensuboptimal zugewiesen wurden

$$\text{influence}(\mathcal{C}) = \frac{f(\mathcal{C}) + g(\mathcal{C})}{|V| + |E|}.$$

Bsp. zu $f(C)$ und $g(C)$



$D \to N(D) = \{A, E, F\}$

$N(D) \cap C1 = \{A\}$

$N(D) \cap C2 = \{E, F\}$

$\nexists C2 : |N(D) \cap C1| < N(D) \cap C2$

- $C = \{C_1, C_2\}, C_1 = \{D, B, C, G, A\}, C_2 = \{E, F\}$
- $f(C) = \max\{|E(C_1)| - 2 \cdot ||C_1| - 5|, 0\} + \max\{|E(C_2)| - 2 \cdot ||C_2| - 5|, 0\} = \max\{6 - 0{,}0\} + \max\{1 - 2 \cdot |(2 - 5)|, 0\} = 6 + \max\{1 - 6{,}0\} = 6 + 0 = 6$
- $g(C) = |\{v \in C_1 | \nexists C' \in C : |N(v) \cap C_1| < |N(v) \cap C'|\} + |\{v \in C_2 | \nexists C' \in C : |N(v) \cap C_2| < |N(v) \cap C'|\} = 6$
- da bspw. $N(D) = \{E, F, A\}, N(D) \cap \mathrm{C}_1 = \{A\}, N(D) \cap \mathrm{C}_2 = \{E, F\}$
- $merriness(C) = \frac{6+6}{7+9} = 3/4$

PÜ

A1

Bei den sogenannten *Linkage*-Verfahren werden die Cluster iterativ miteinander vereinigt, wobei die Verlinkung in der aktuellen Situation vorteilhaft wirkt. Das initiale Clustering ist dann $\mathcal{C}_0 = \{\{v\} | v \in V\}$, das heißt, jeder Knoten ist sein eigenes Cluster. Durch das Verlinken werden immer zwei Cluster des aktuellen Clusterings $\mathcal{C}_i$ zu einem neuen Cluster vereinigt, es entsteht ein neues Clustering $\mathcal{C}_{i+1}$.

Bei einer solchen Methode wird zunächst eine Menge von Zentren $Z \subseteq V$ gewählt. Das initiale Clustering ist dann $\mathcal{C}_0 = \{\{v\} | v \in V\}$, das heißt, jeder Knoten ist sein eigenes Cluster. Im Anschluss daran wird iterativ ein Knoten v, der nicht in einem Cluster mit einem Knoten von Z ist, ausgewählt und dem Cluster, das ein Zentrum aus Z enthält, hinzugefügt, zu dem er den kürzesten Abstand hat. Gibt es mehrere solche Cluster, so soll zunächst das kleinste gewählt werden. Ergibt das immer noch keine eindeutige Entscheidung, wird das Cluster gewählt, dessen Zentrum nach einer zuvor festgelegten Sortierung den kleinsten Index hat. Nachdem nun die beiden Cluster vereinigt wurden, erhält man ein neues Clustering. Dieses kann durch ein neues Verlinken weiter verfeinert werden.

a) Geben Sie das oben beschriebene Verfahren in Pseudocode an.

b) Bestimmen Sie die Laufzeit des Algorithmus.

c) Ein Vorteil der Zentrumswahl ist es auch, bestimmte Knoten getrennt halten zu können. In unserem Beispiel kann man nun so bspw. sicherstellen, dass Nutzer_innen, die schwierige Persönlichkeiten sind, nicht in derselben Zielgruppe landen. Nutzen Sie den Algorithmus, um eine Zielgruppeneinteilung für unser Beispiel zu bestimmen, wobei $Z = \{\text{Wario}, \text{Waluigi}, \text{Bowser Jr.}\}$. Geben Sie alle Zwischenschritte an. Betrachten Sie die Knoten in der Reihenfolge ihrer ID-Nummern.