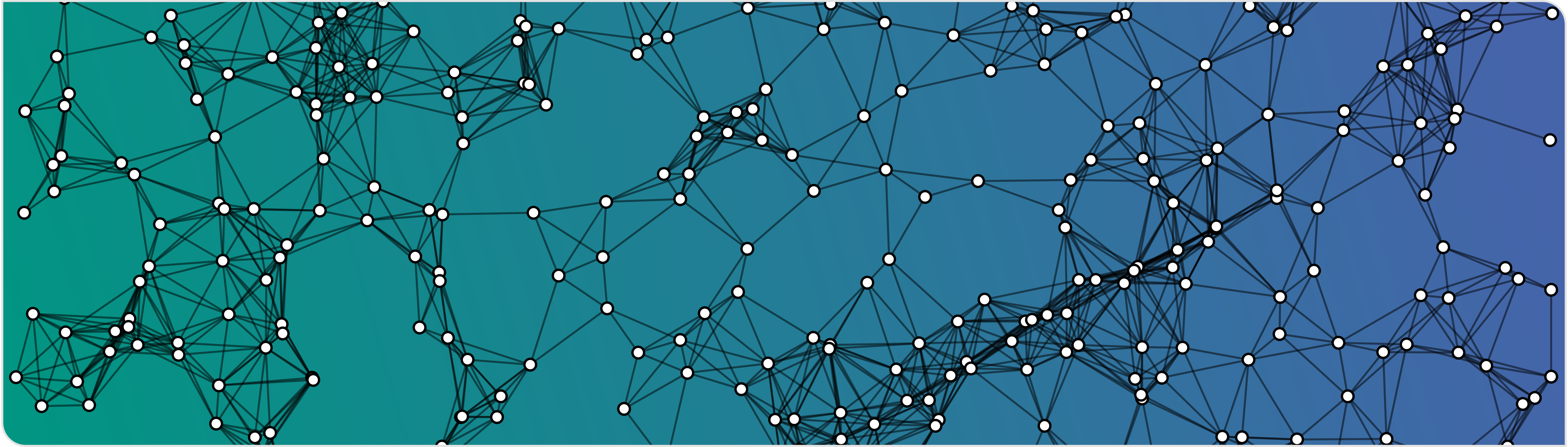
Karlsruher Institut für Technologie

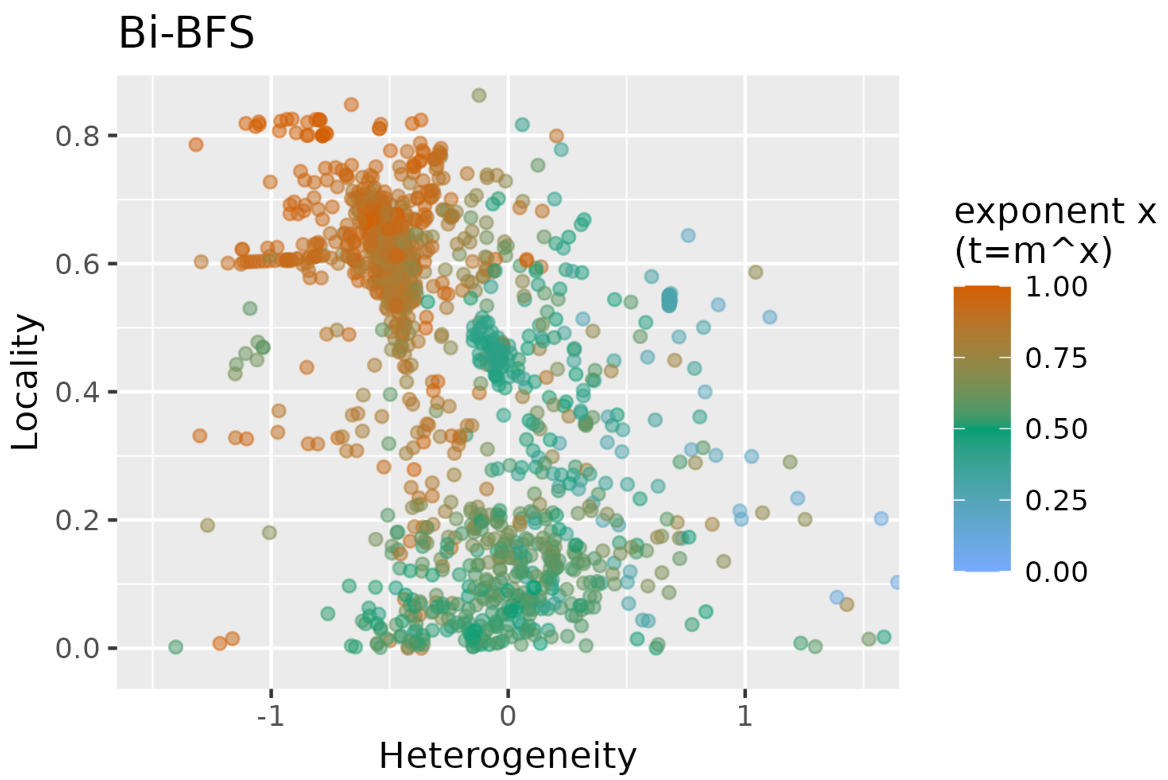
Praktikum – Beating the Worst Case

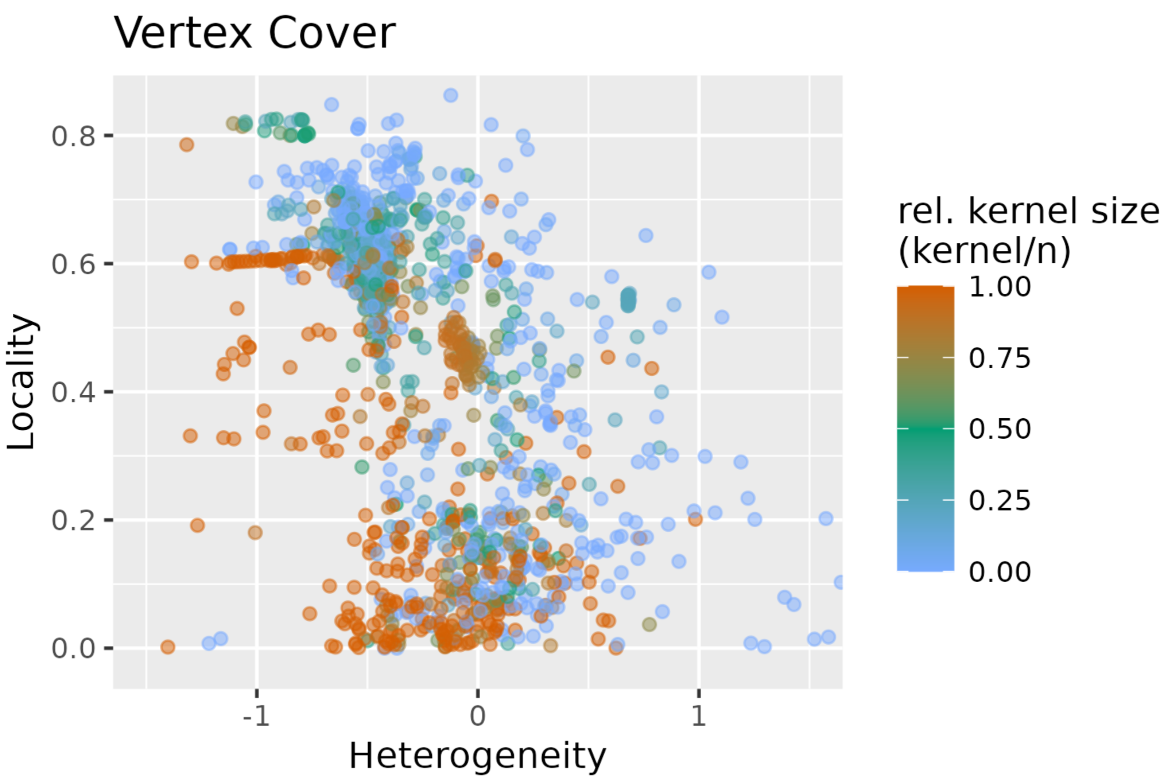
Jean-Pierre von der Heydt und Marcus Wilhelm | 13.11.2023



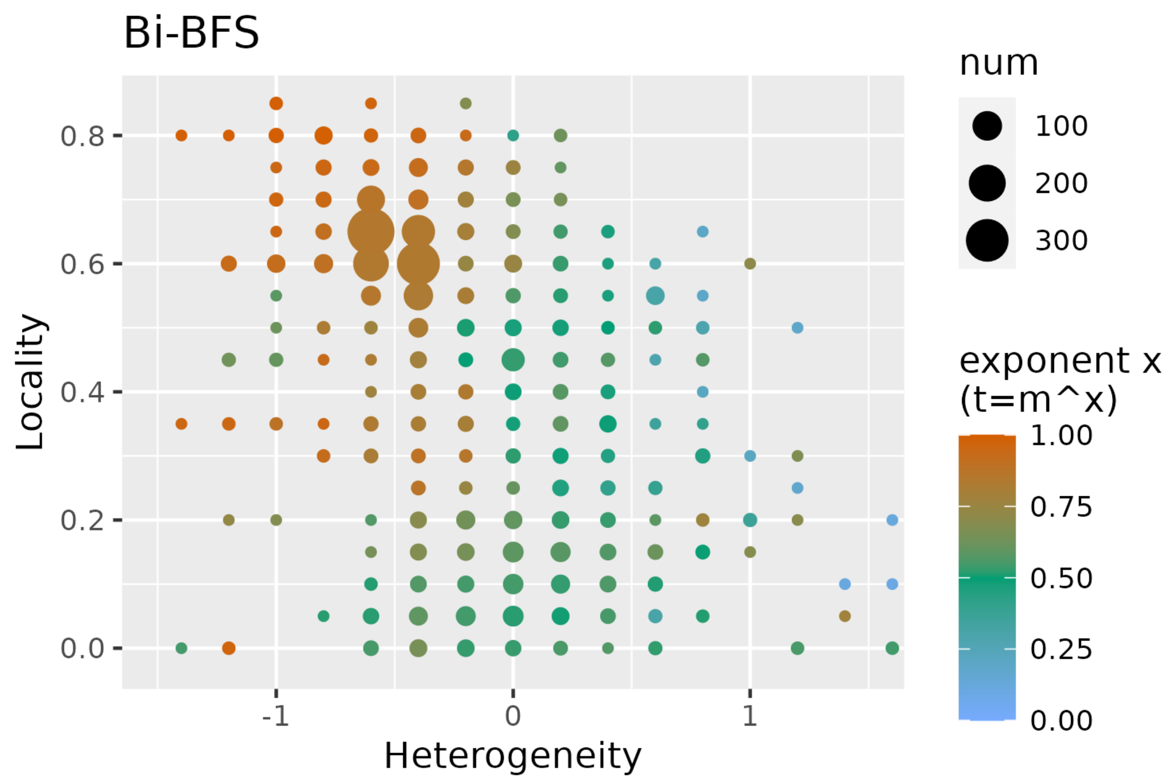
Vorstellung Übungsblatt 2

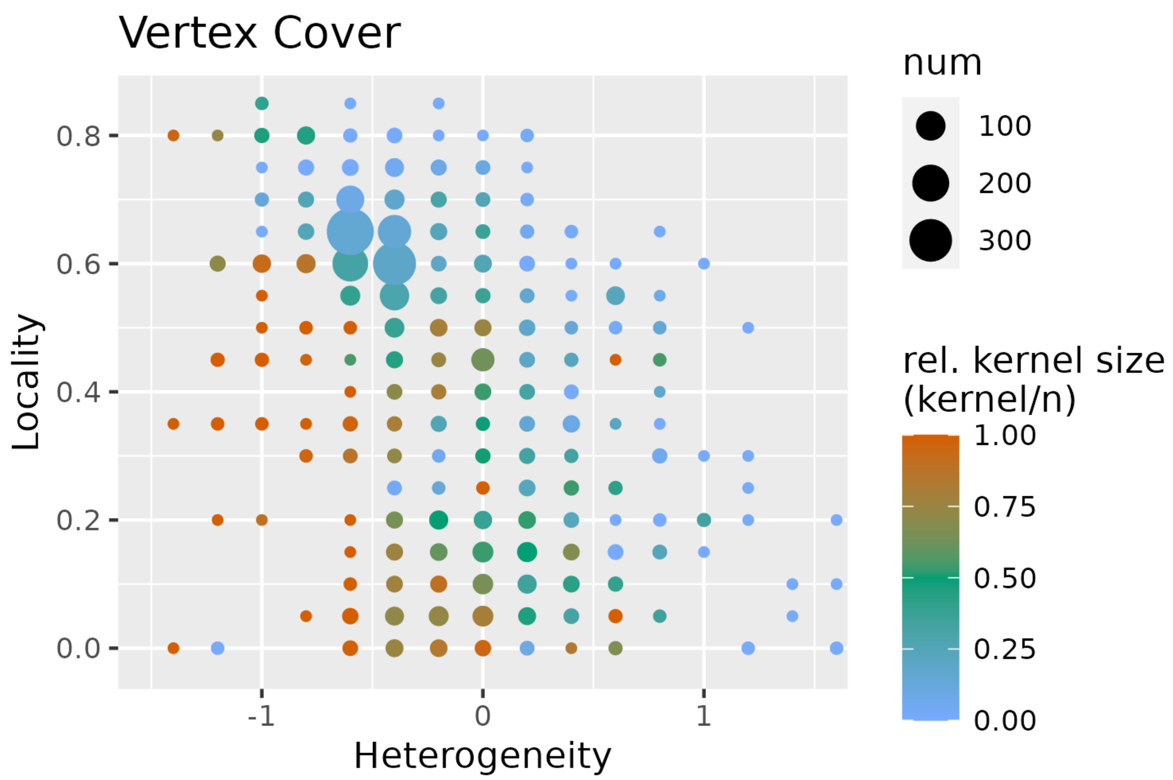
Lösung Übungsblatt 2



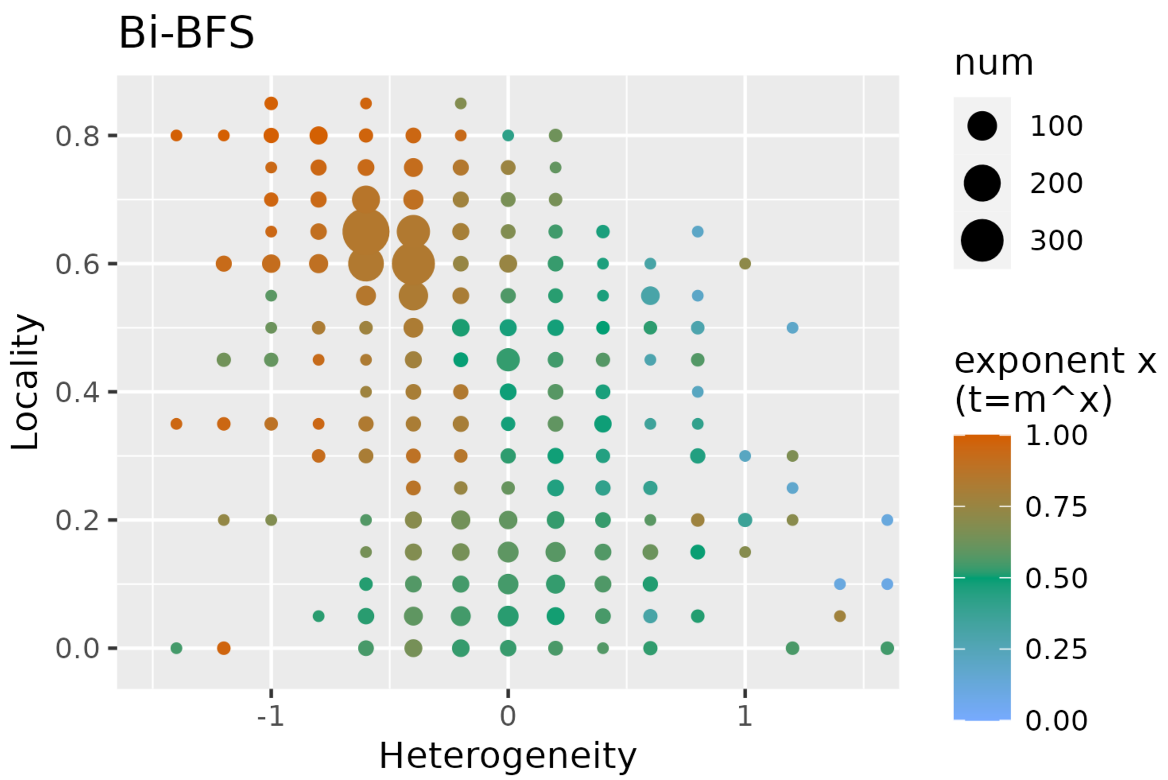


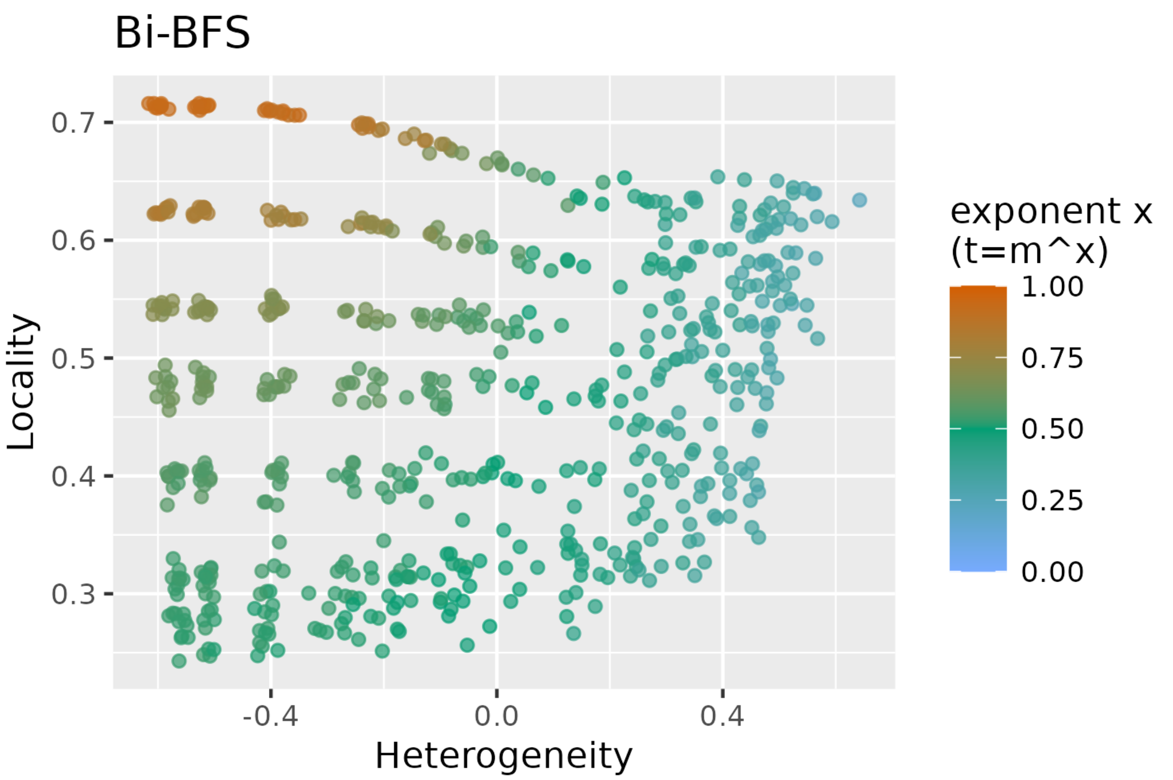
Lösung Übungsblatt 2



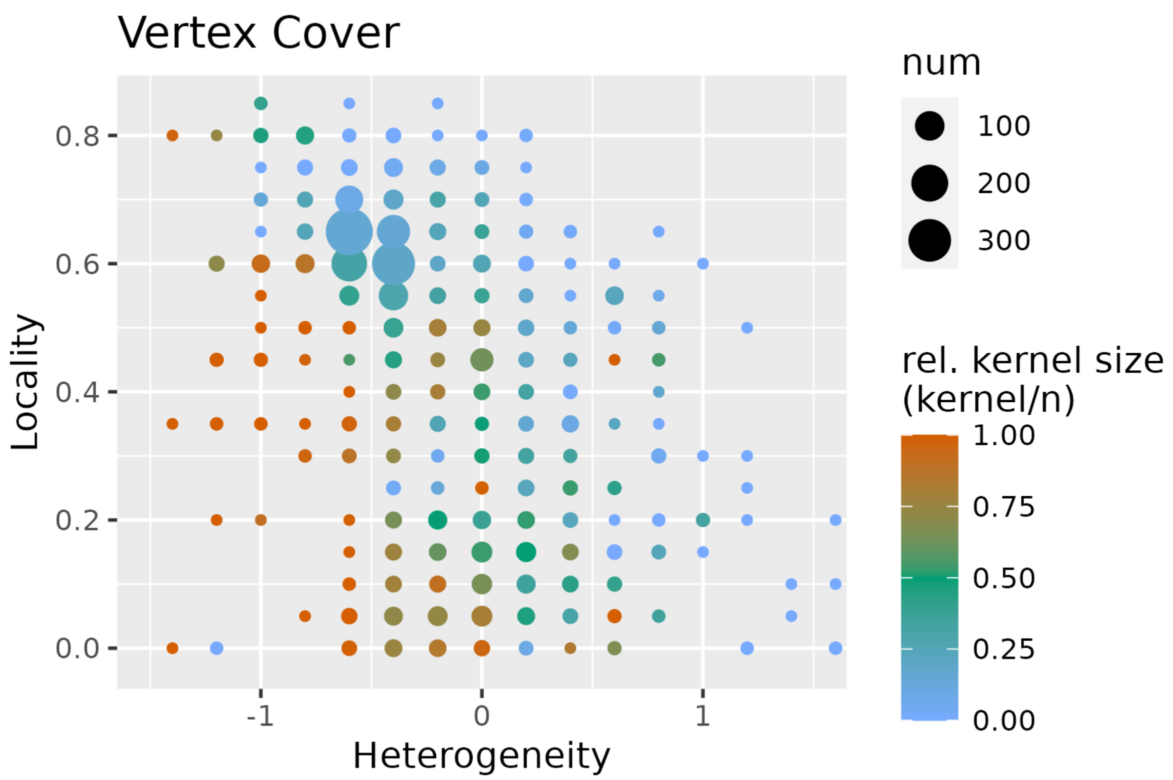


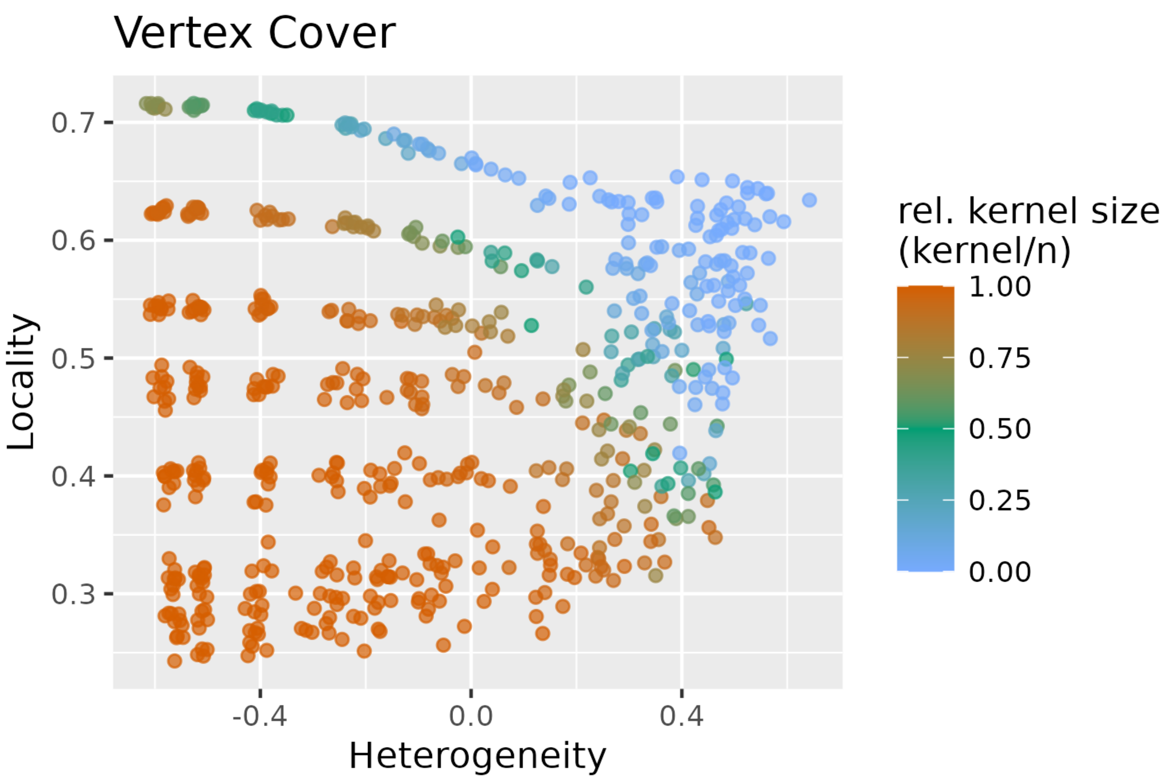
Lösung Übungsblatt 2



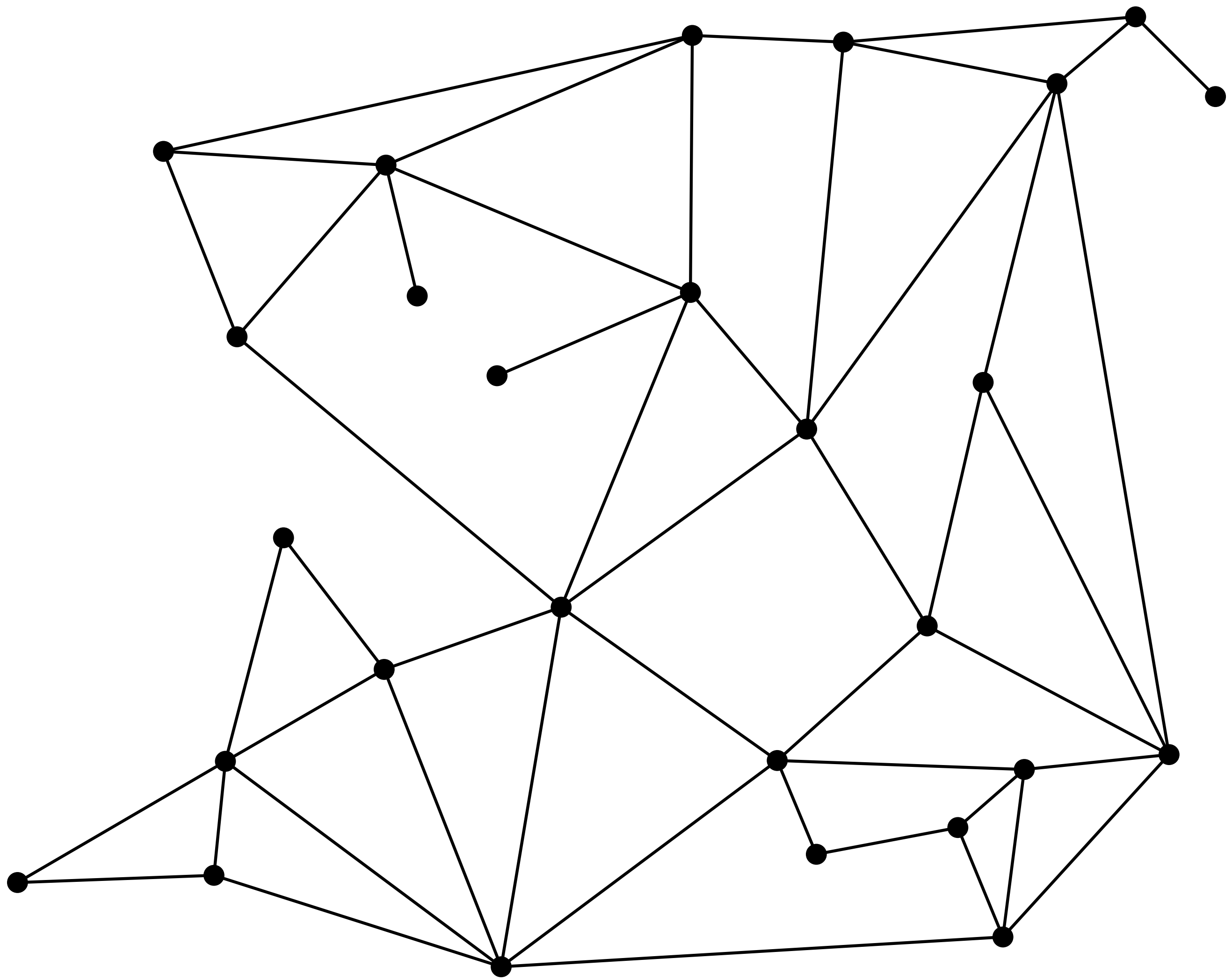


Lösung Übungsblatt 2

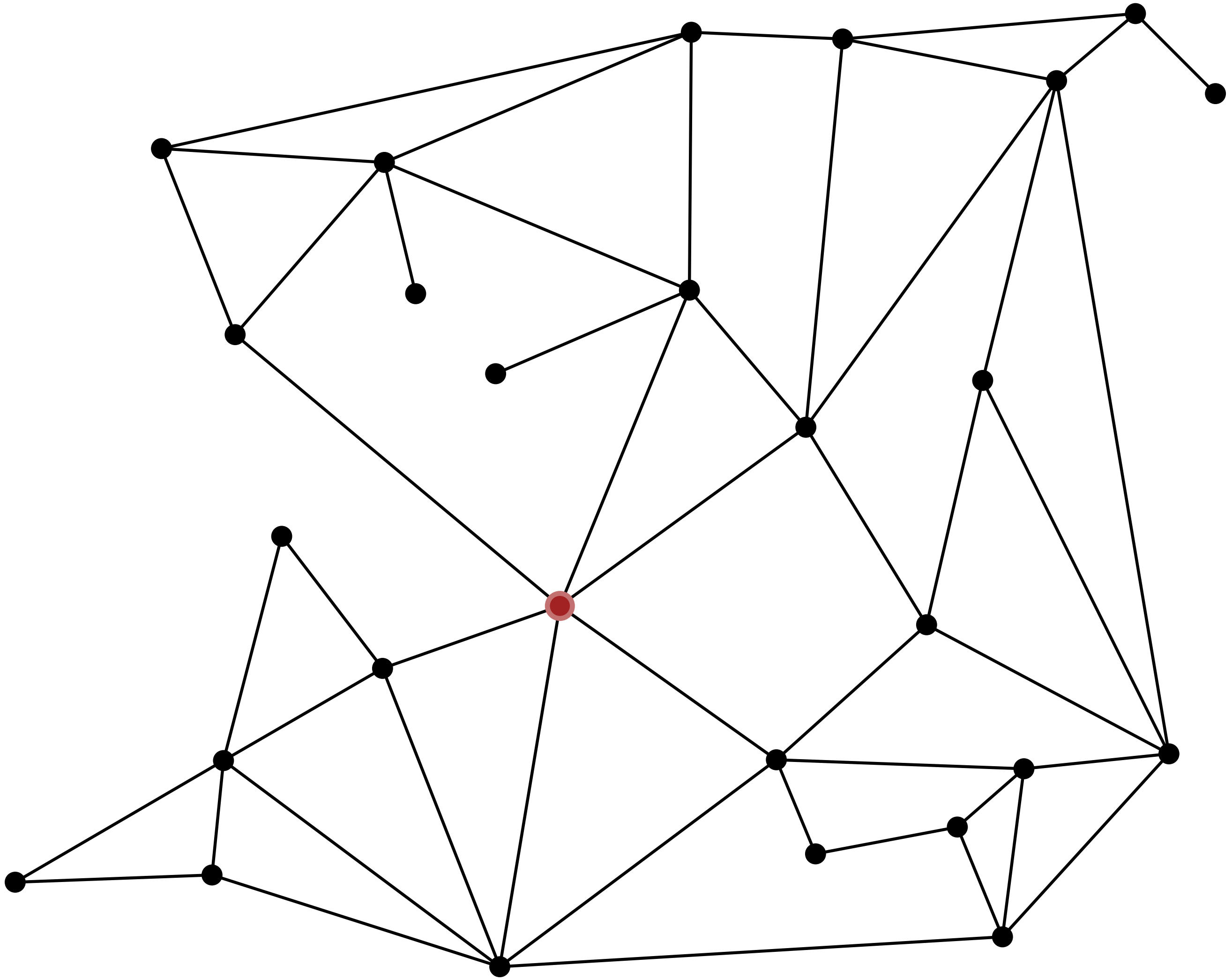




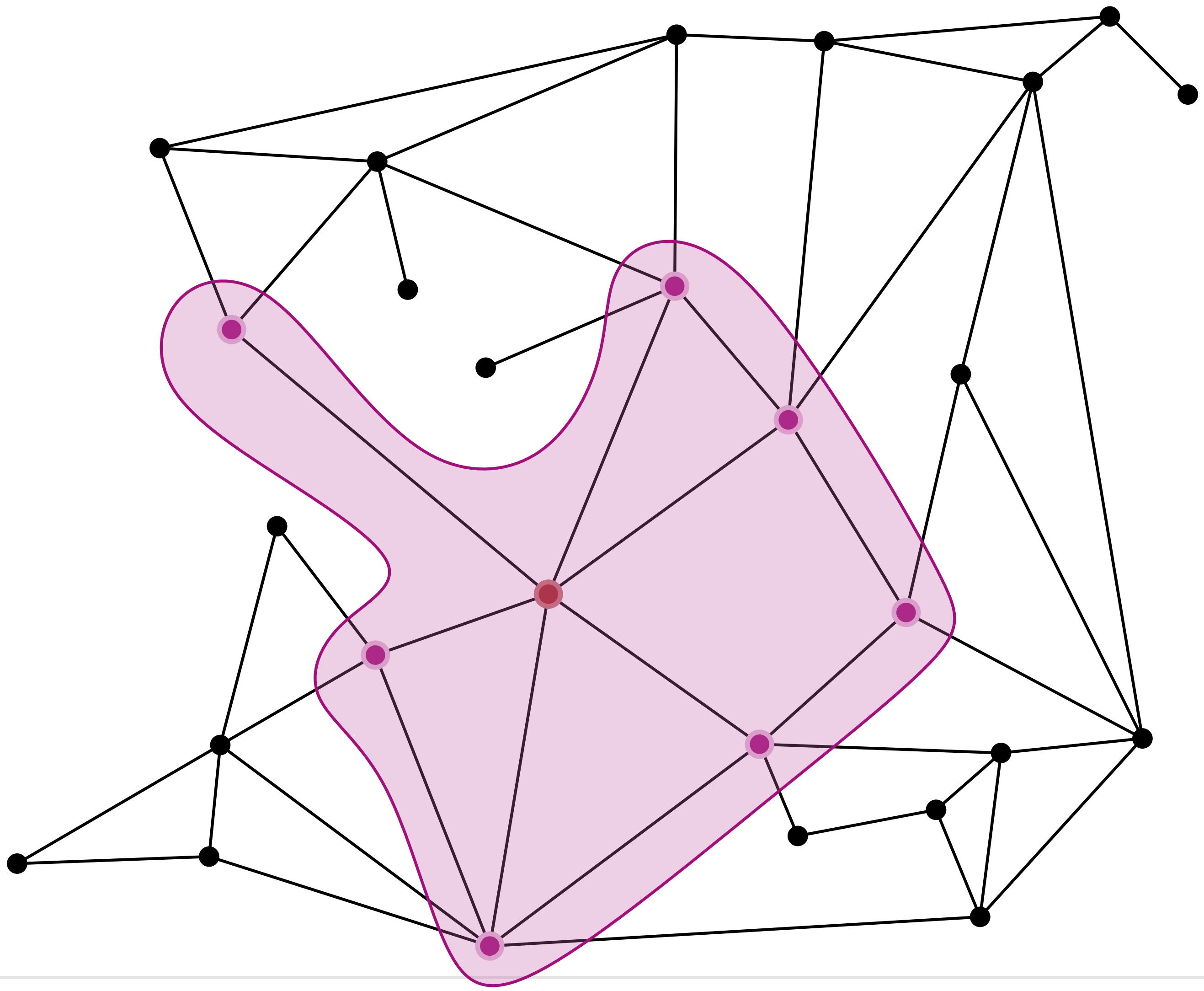
Exzentrizität und Durchmesser



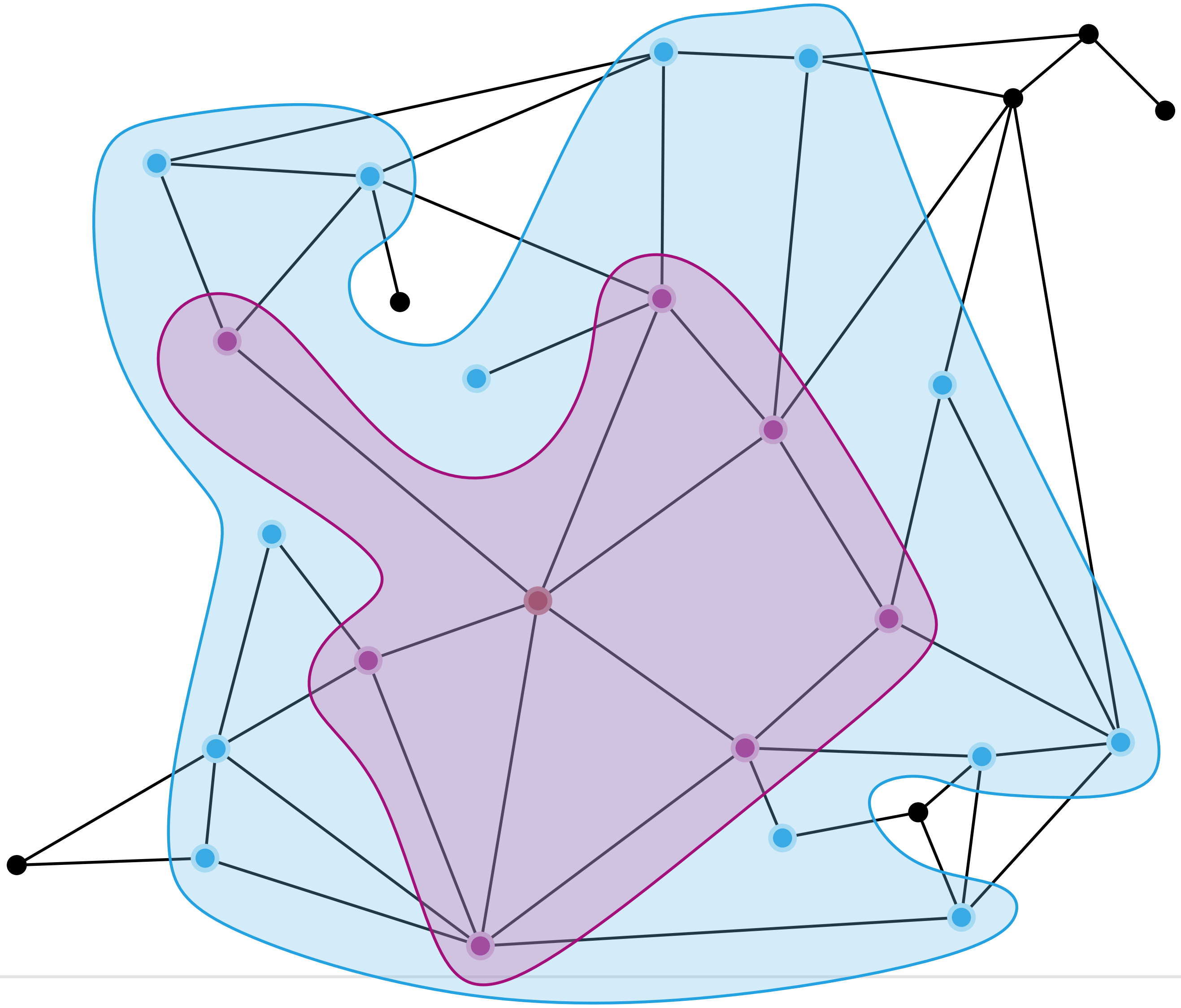
Exzentrizität und Durchmesser



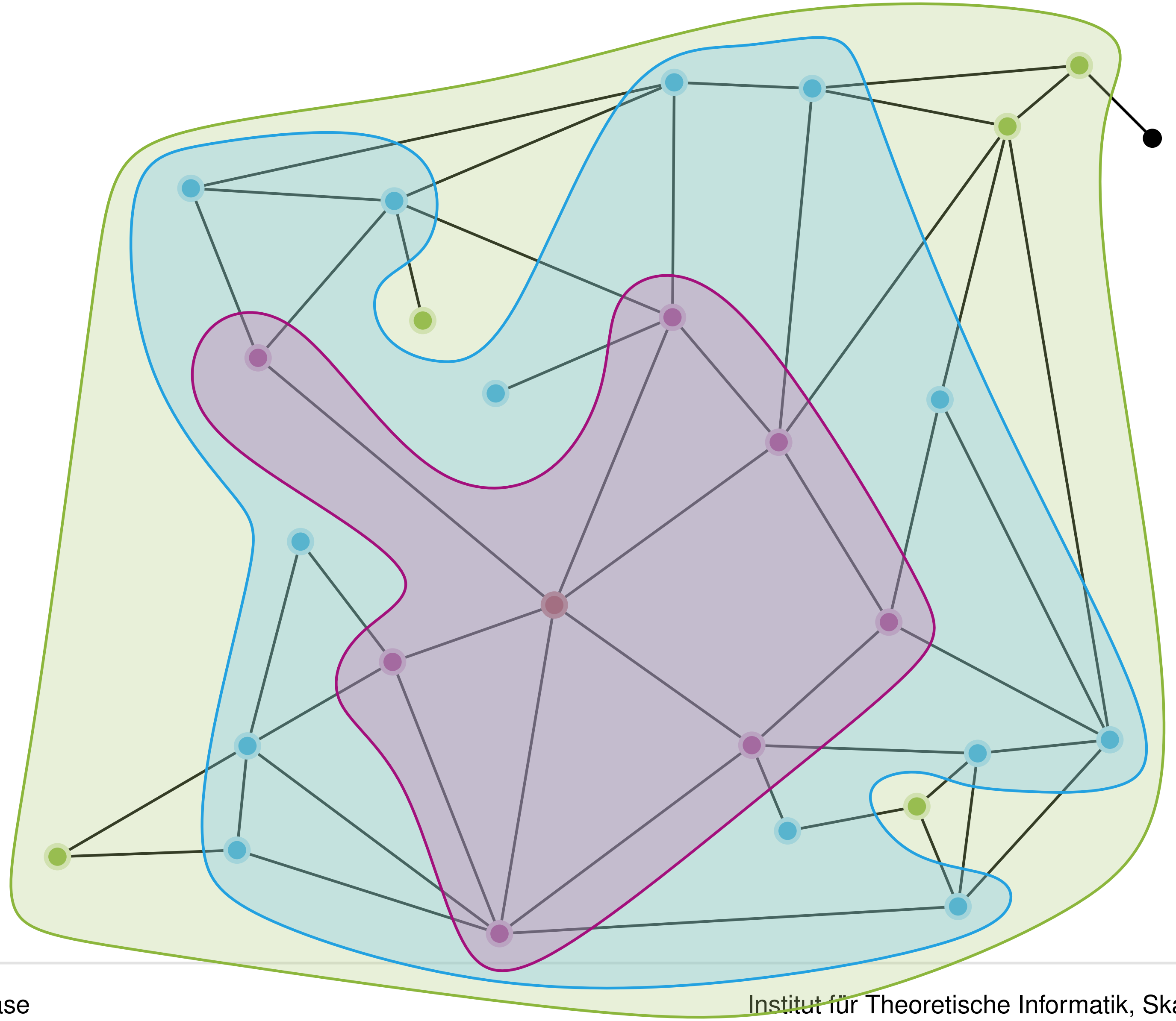
Exzentrizität und Durchmesser



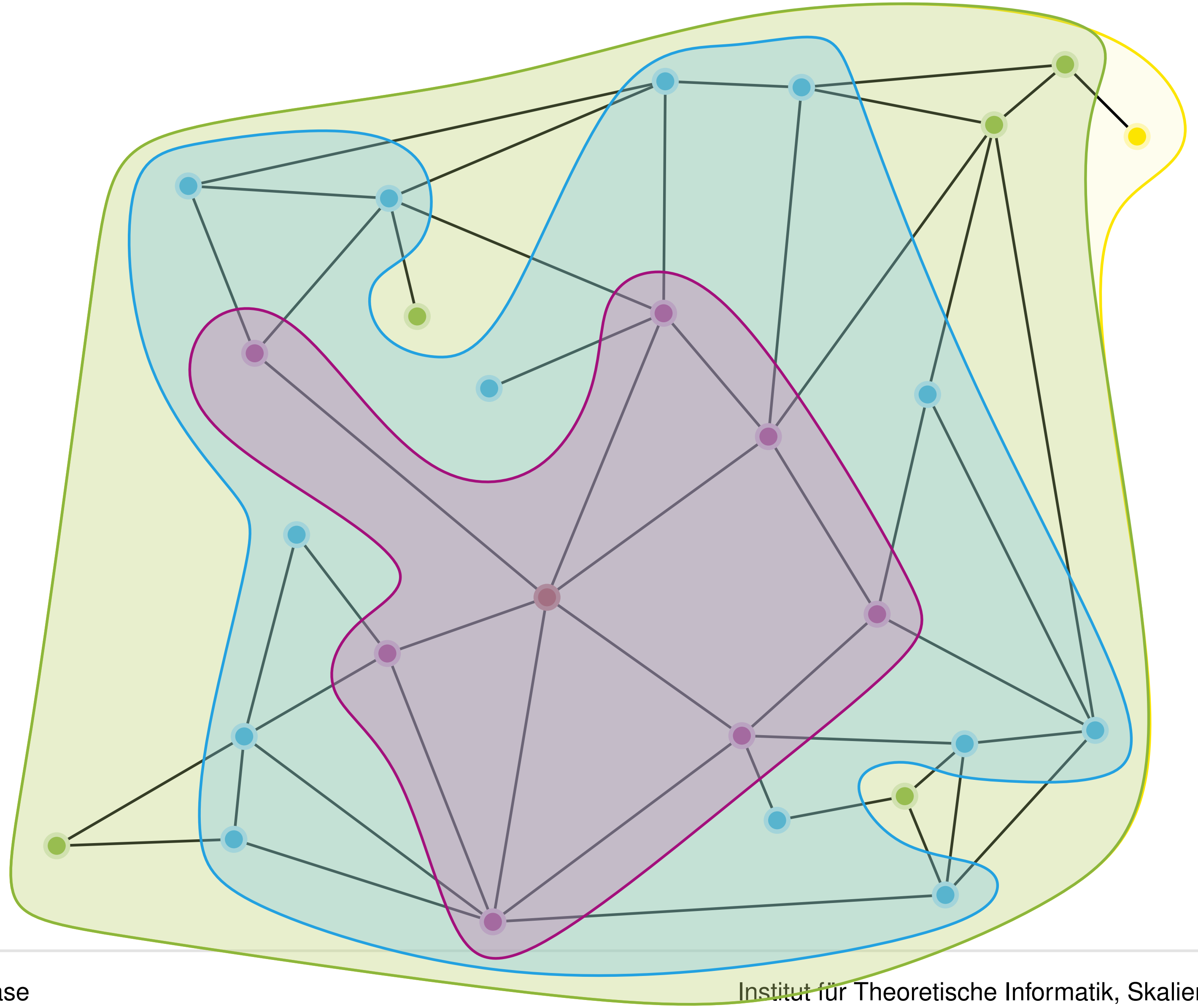
Exzentrizität und Durchmesser



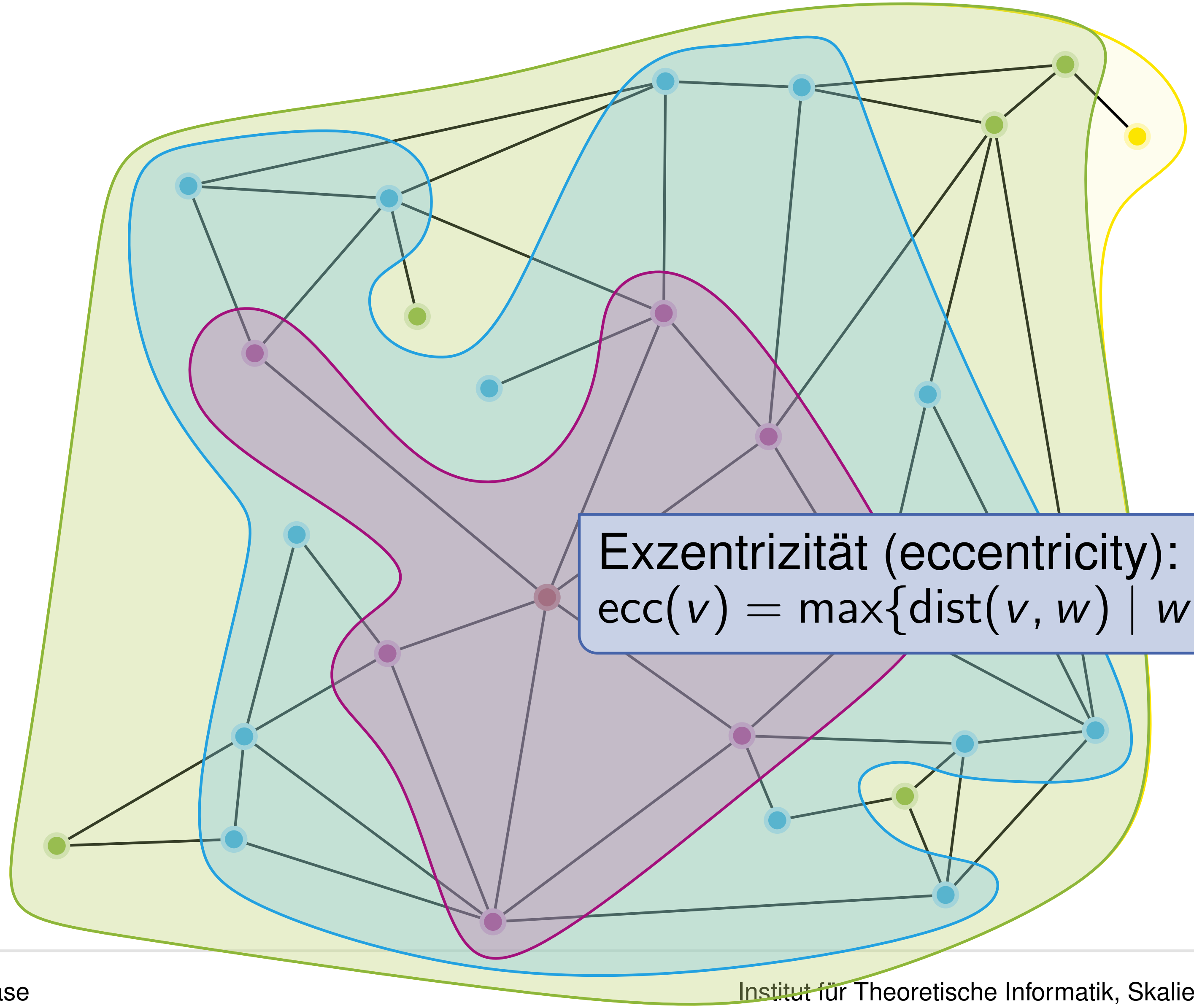
Exzentrizität und Durchmesser



Exzentrizität und Durchmesser

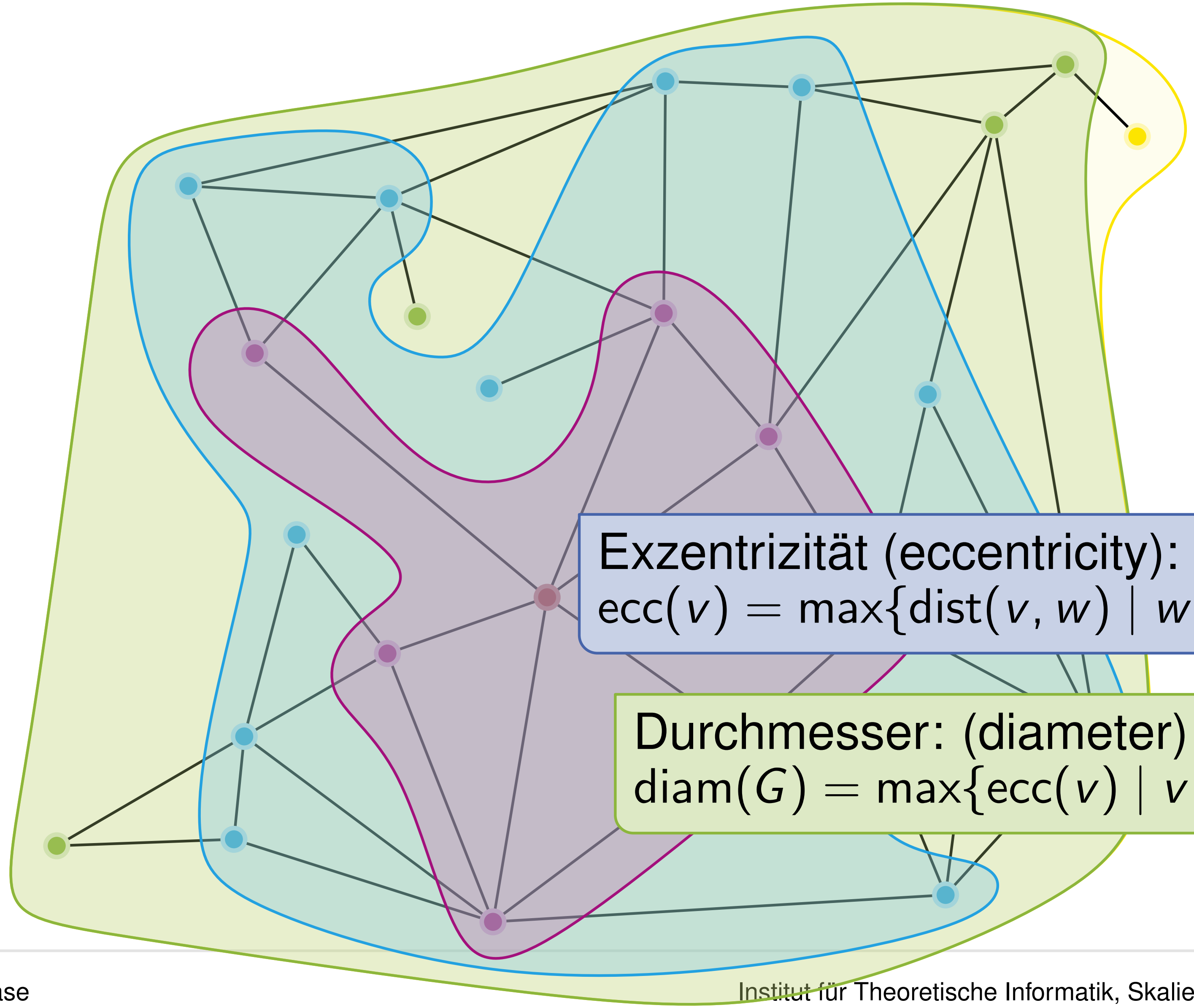


Exzentrizität und Durchmesser



Exzentrizität (eccentricity):
$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$

Exzentrizität und Durchmesser

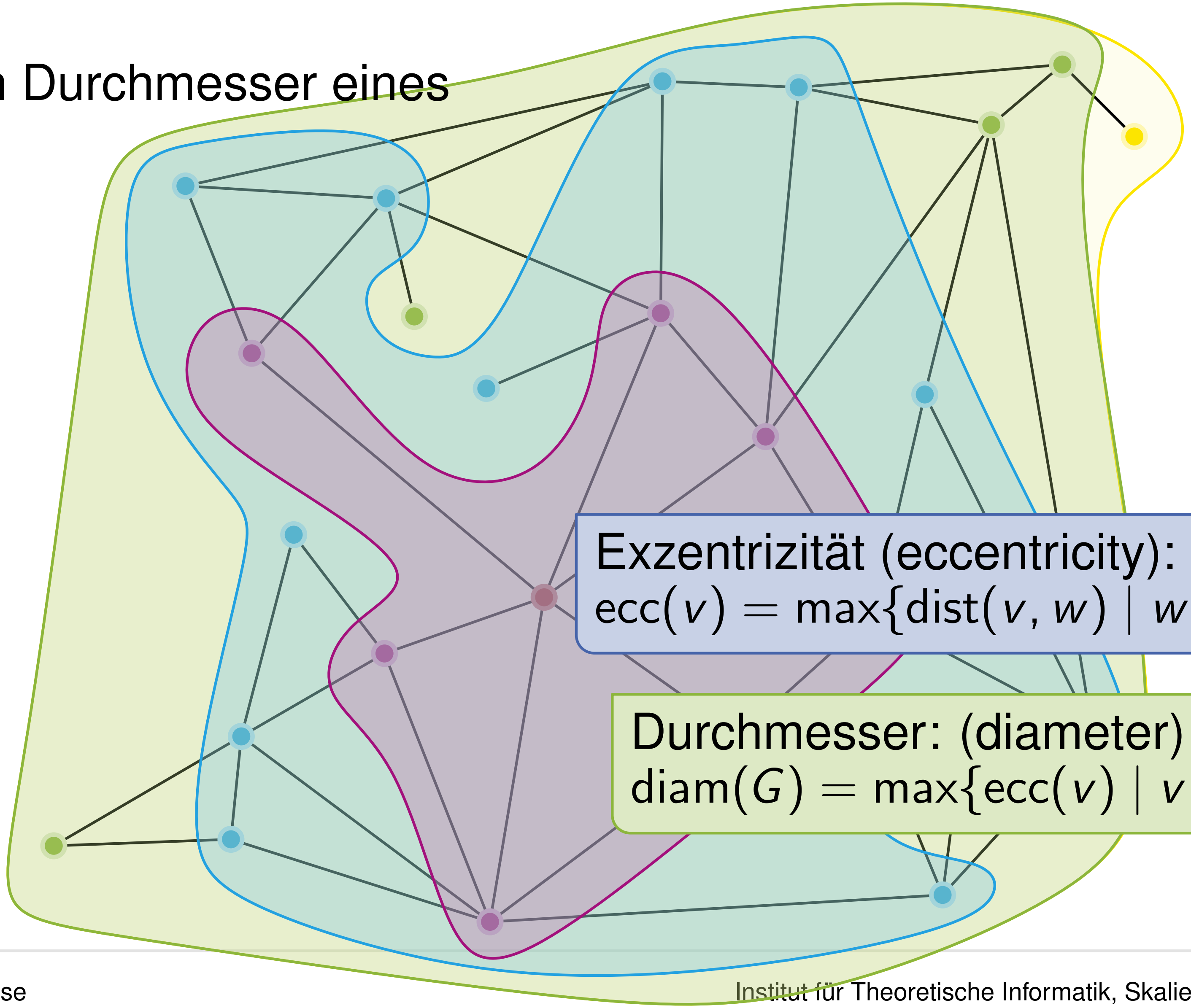


Exzentrizität (eccentricity):
$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$

Durchmesser: (diameter)
$\mathrm{diam}(G) = \max\{\mathrm{ecc}(v) \mid v \in V\}$

Exzentrizität und Durchmesser

Wie berechnet man den Durchmesser eines Graphen?



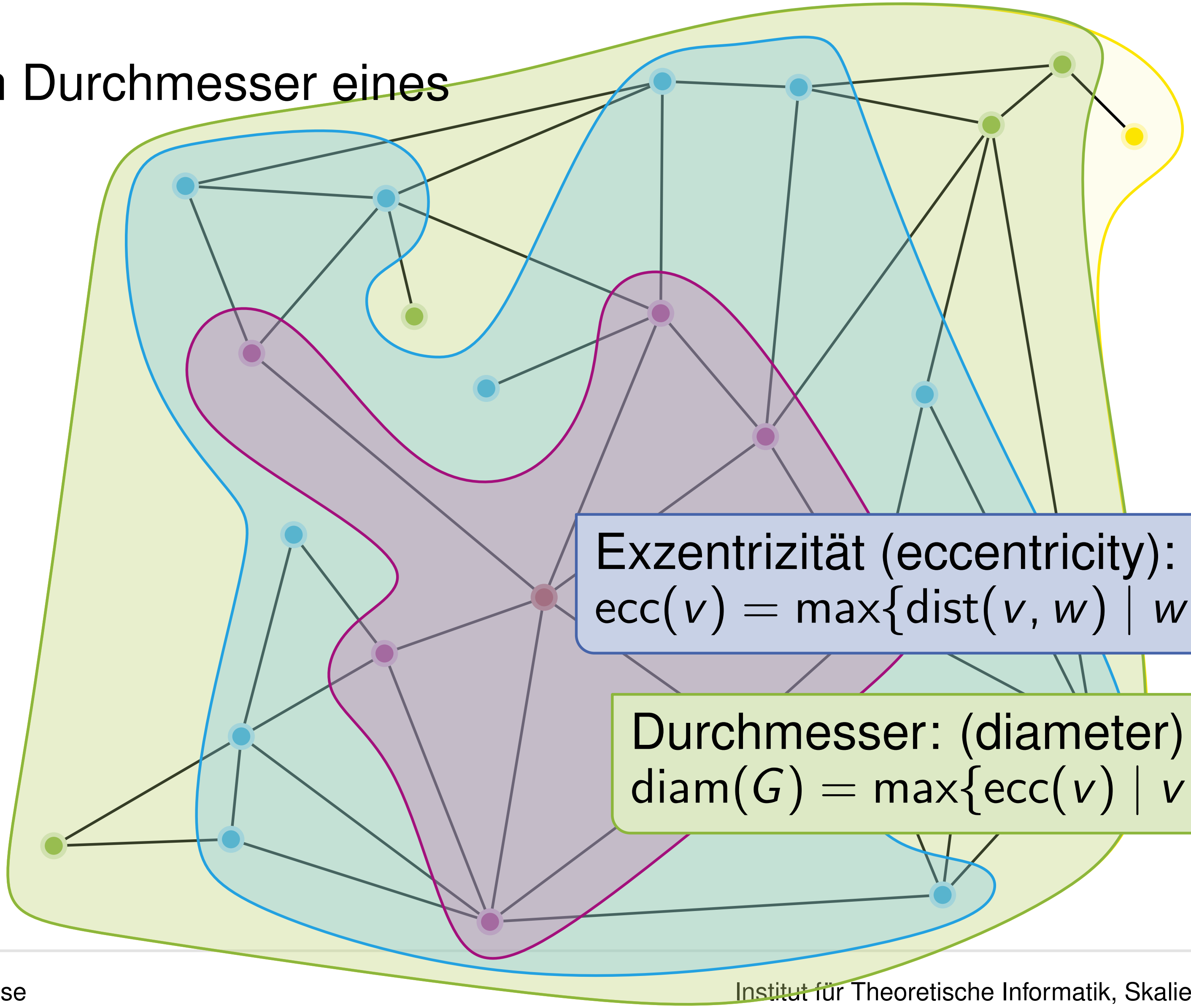
Exzentrizität (eccentricity):
$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$

Durchmesser: (diameter)
$\mathrm{diam}(G) = \max\{\mathrm{ecc}(v) \mid v \in V\}$

Exzentrizität und Durchmesser

Wie berechnet man den Durchmesser eines Graphen?

- naiv: n mal BFS



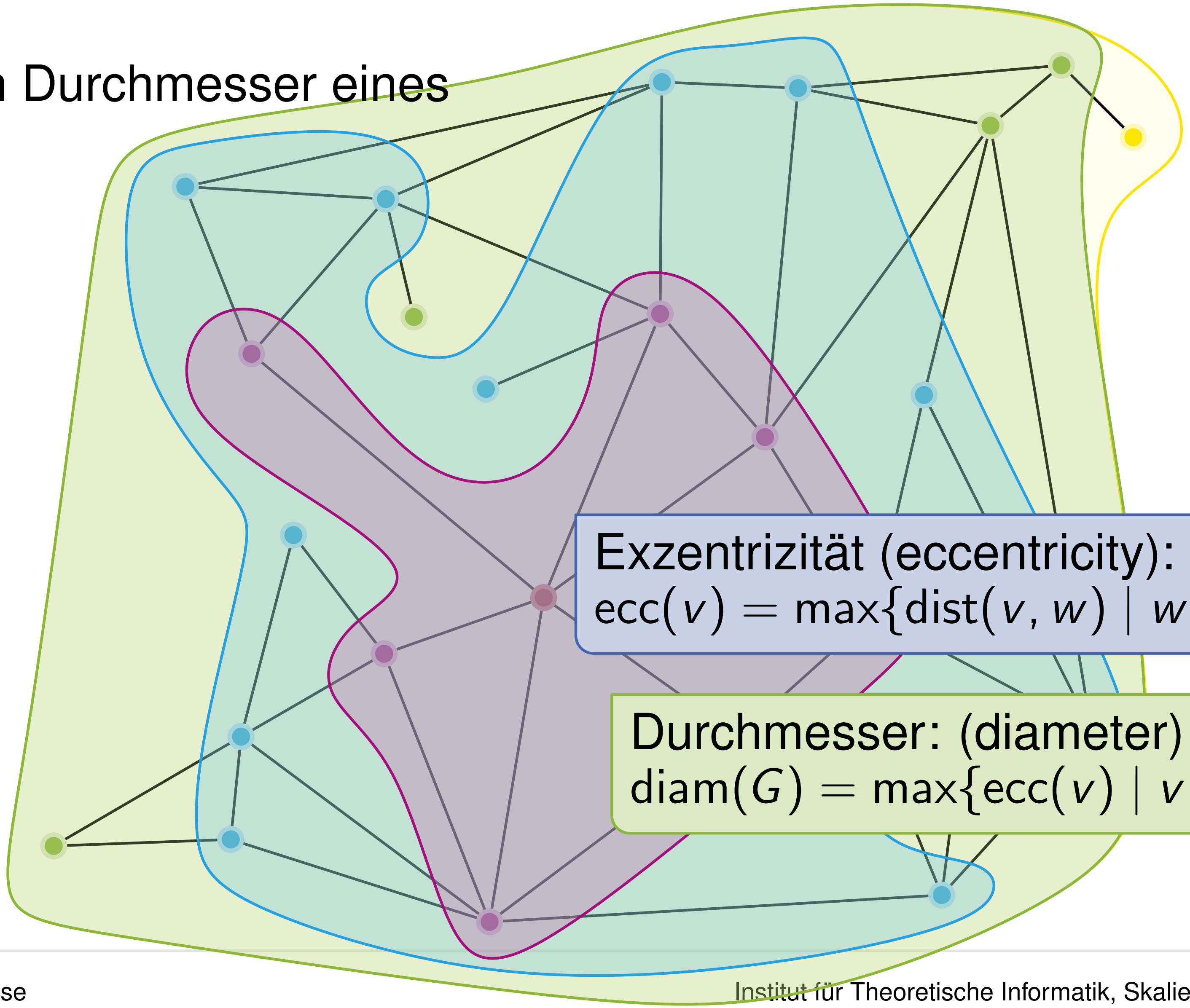
Exzentrizität (eccentricity):
$$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$$

Durchmesser: (diameter)
$$\mathrm{diam}(G) = \max\{\mathrm{ecc}(v) \mid v \in V\}$$

Exzentrizität und Durchmesser

Wie berechnet man den Durchmesser eines Graphen?

- naiv: n mal BFS
- geht es besser?



Exzentrizität (eccentricity):
$$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$$

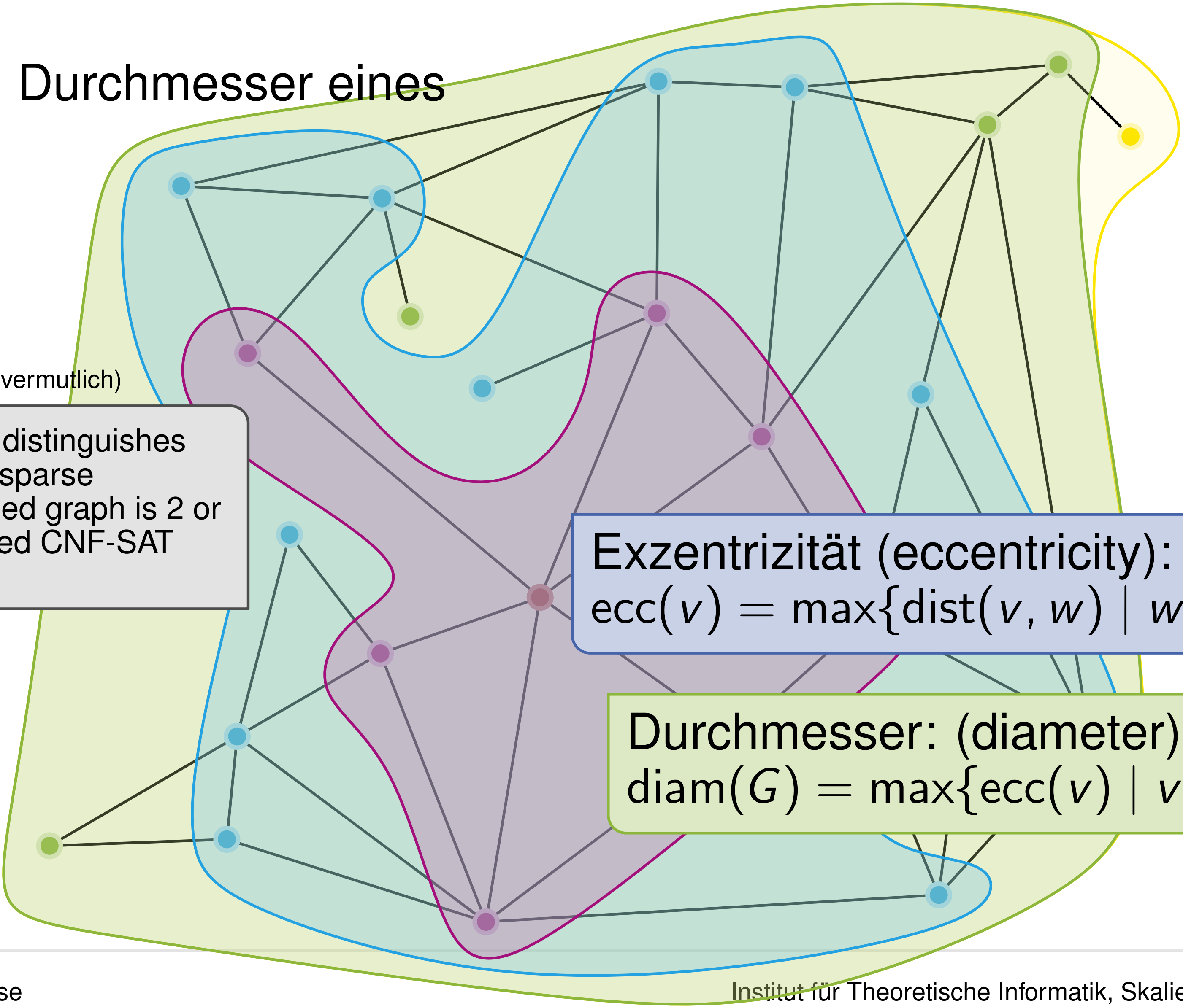
Durchmesser: (diameter)
$$\mathrm{diam}(G) = \max\{\mathrm{ecc}(v) \mid v \in V\}$$

Exzentrizität und Durchmesser

Wie berechnet man den Durchmesser eines Graphen?

- naiv: n mal BFS
- geht es besser?
  - nein: (vermutlich)

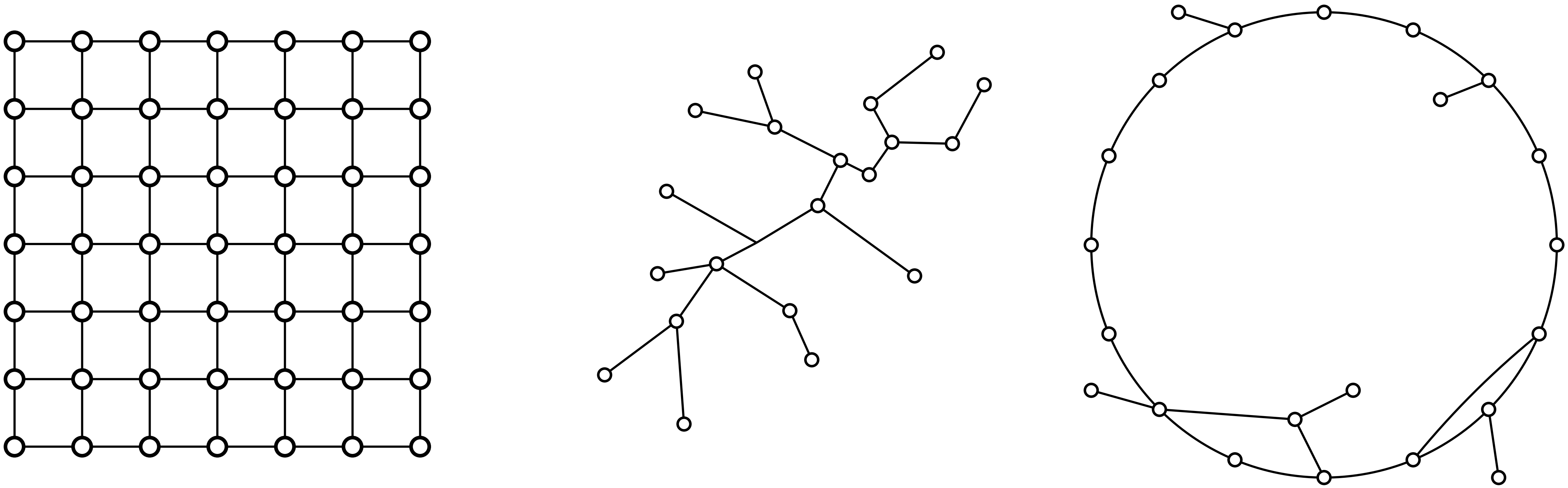
> " any $O(n^{2-\varepsilon})$ time algorithm that distinguishes whether the diameter of a given sparse ($m = O(n)$) undirected unweighted graph is 2 or at least 3 would imply an improved CNF-SAT algorithm [Liam, Williams – STOC'13]



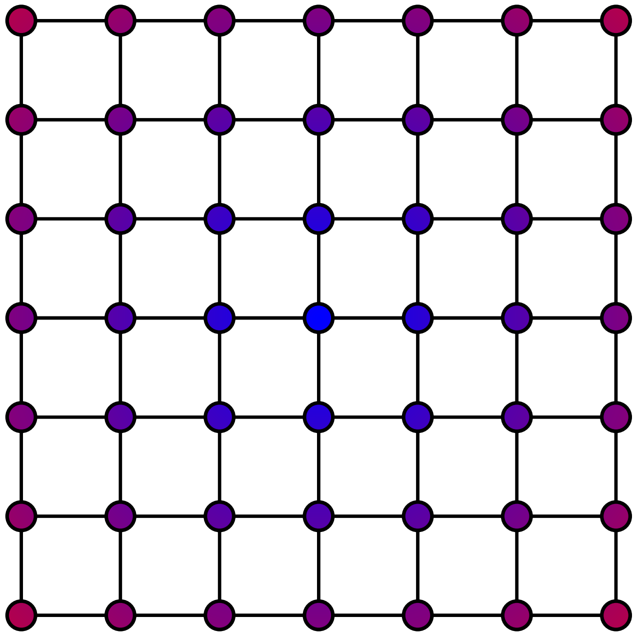
Exzentrizität (eccentricity):
$\mathrm{ecc}(v) = \max\{\mathrm{dist}(v, w) \mid w \in V\}$

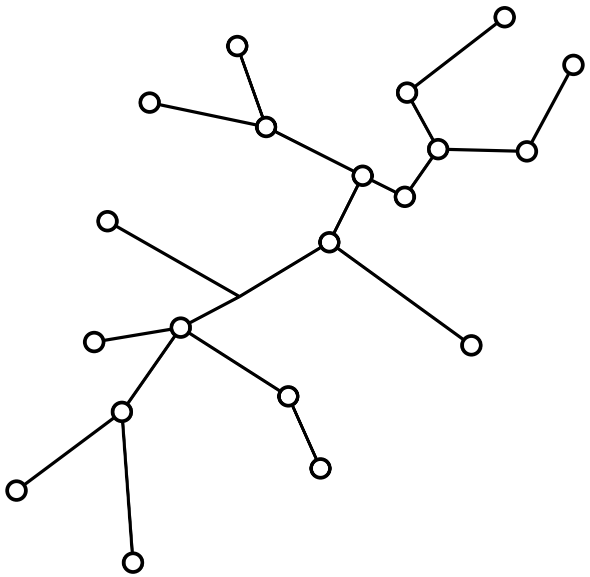
Durchmesser: (diameter)
$\mathrm{diam}(G) = \max\{\mathrm{ecc}(v) \mid v \in V\}$

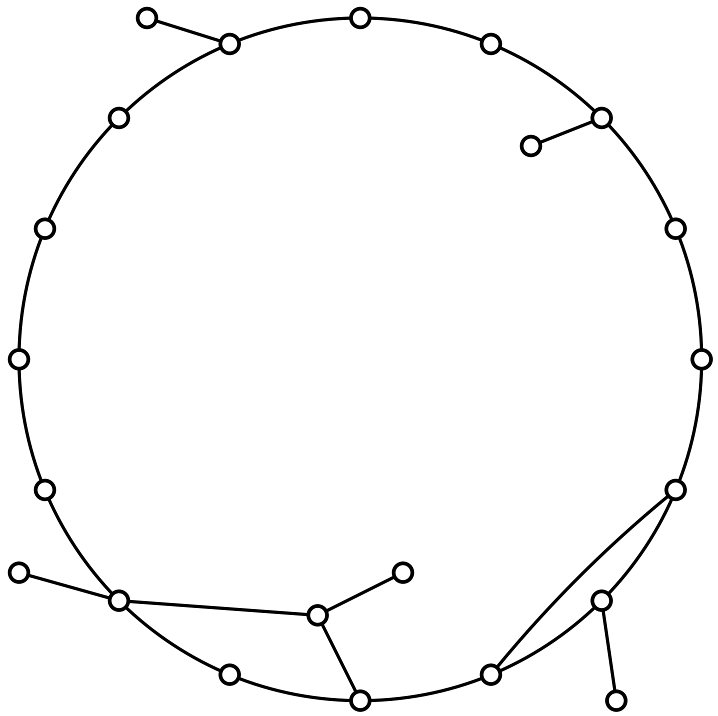
Algorithmen für nette Eingaben



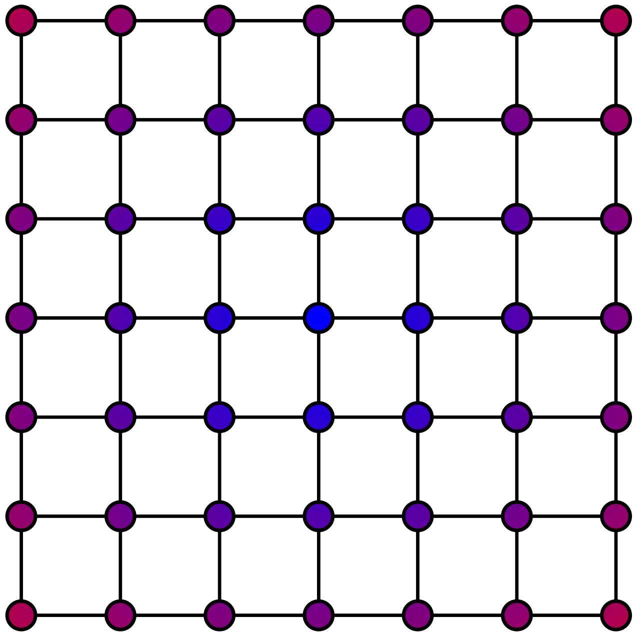
Algorithmen für nette Eingaben

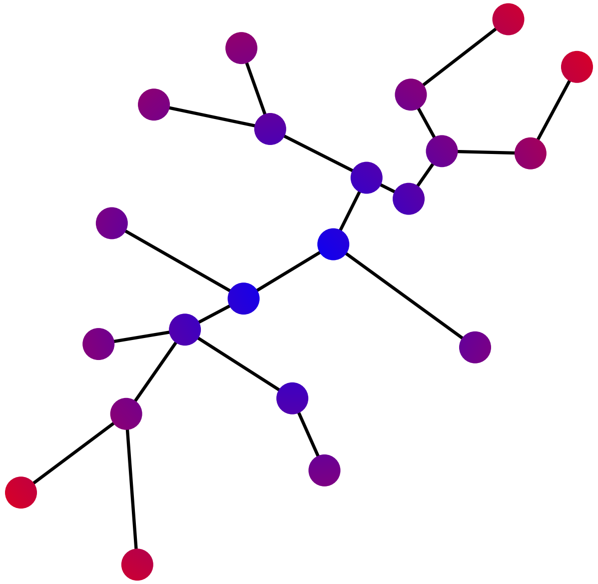


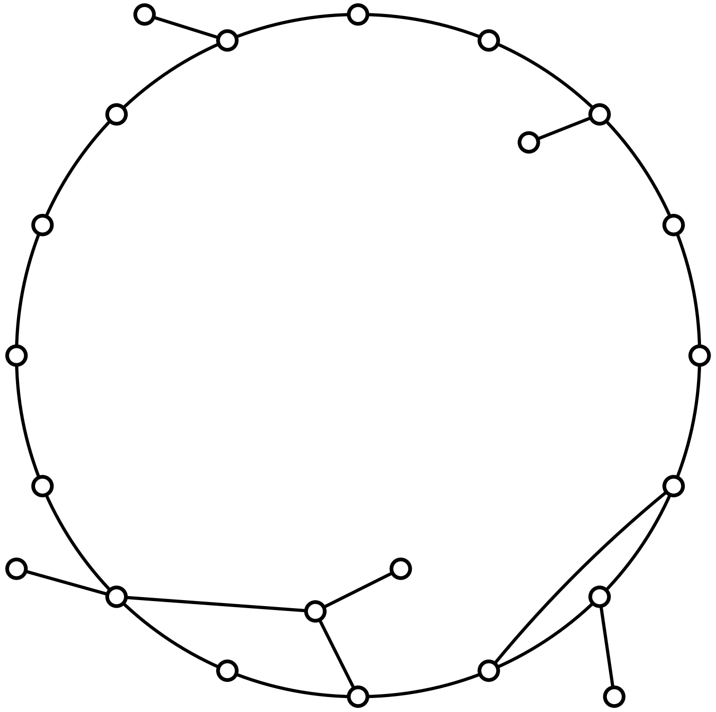




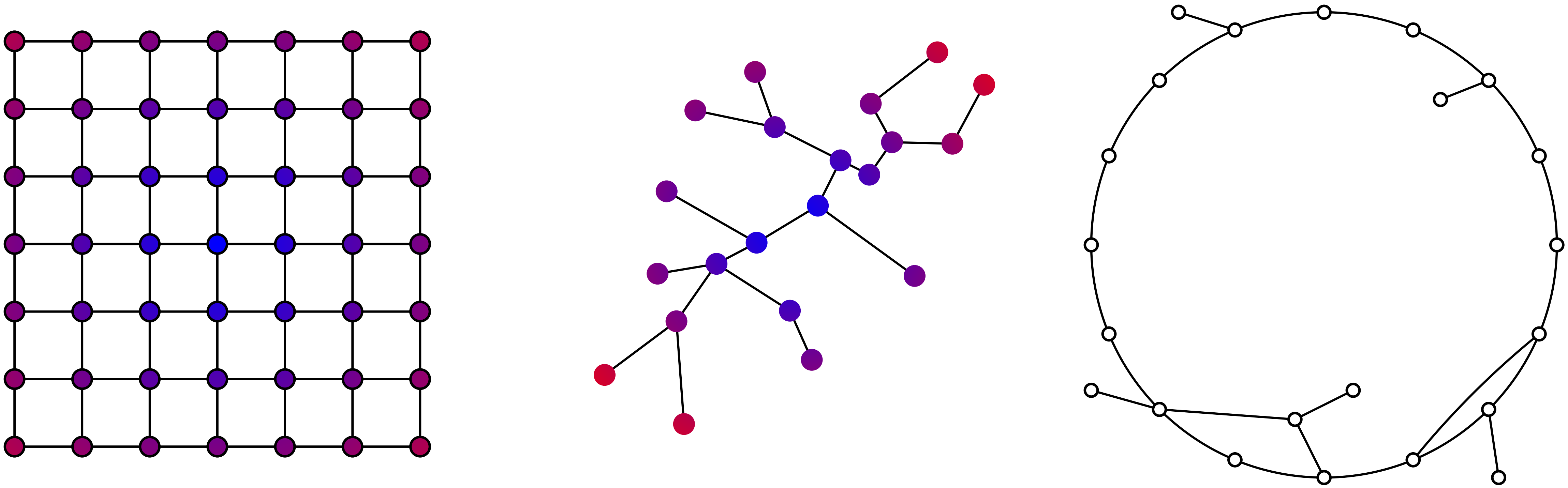
Algorithmen für nette Eingaben





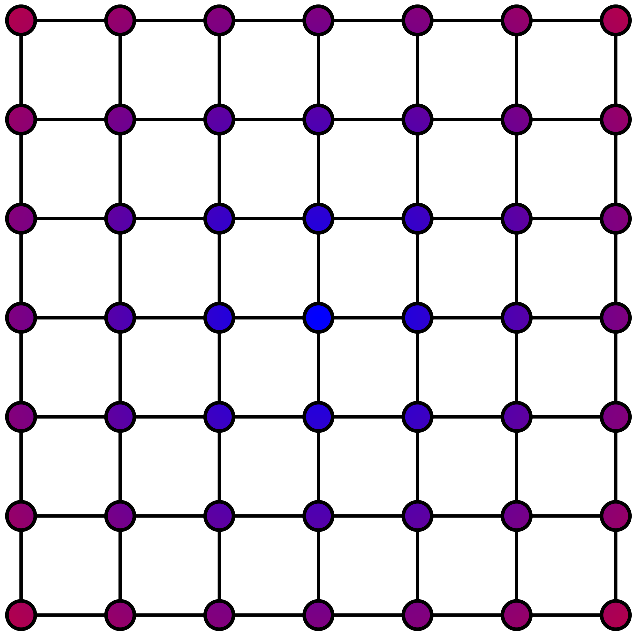
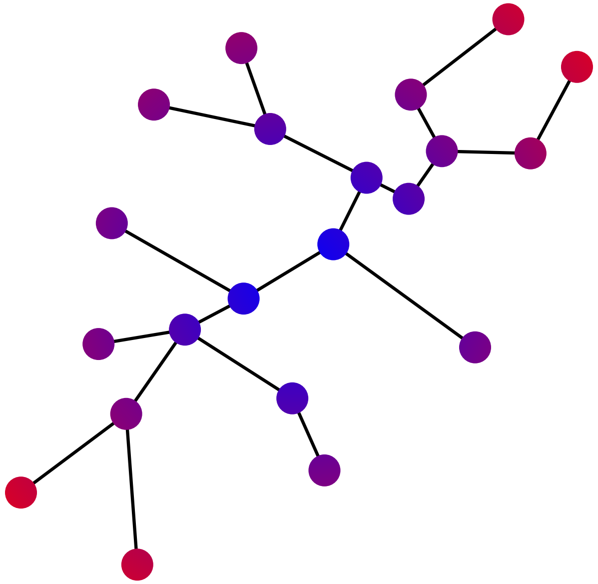
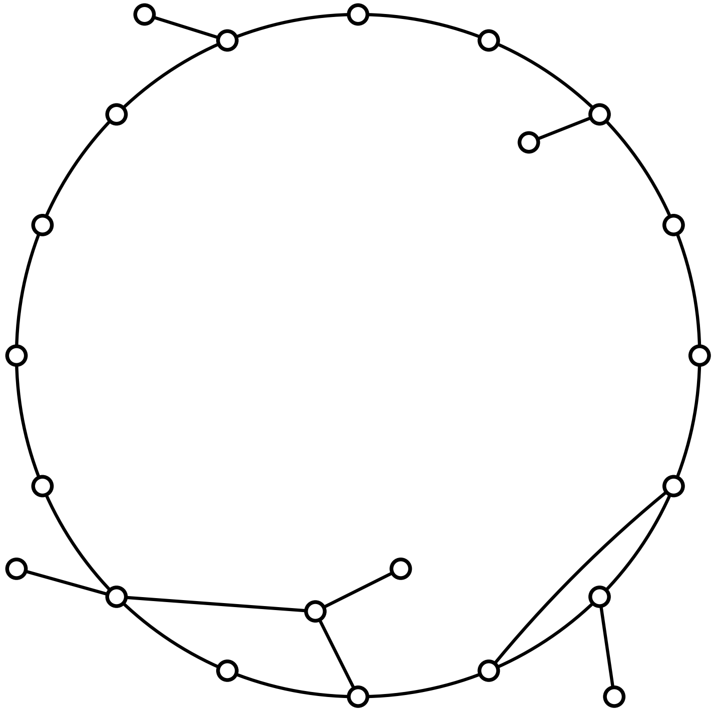


Algorithmen für nette Eingaben



Idee: mache BFS nur von besonders dezentralen Knoten

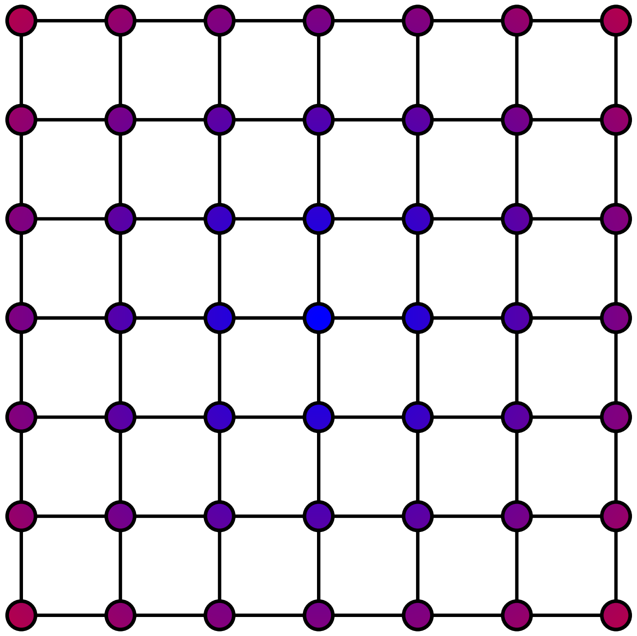
Algorithmen für nette Eingaben

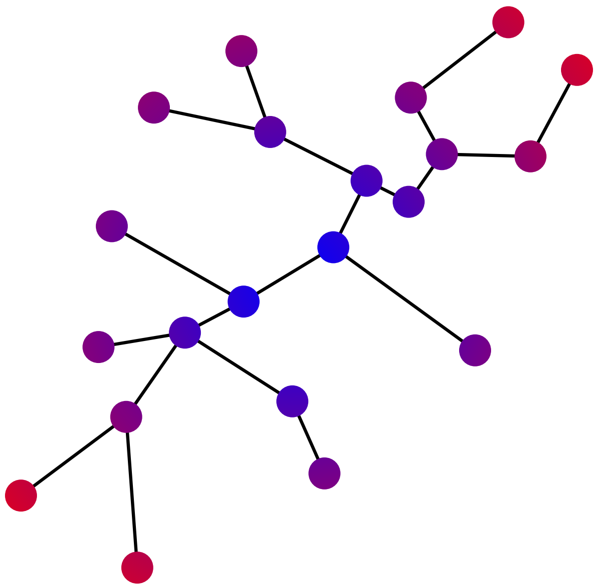
  

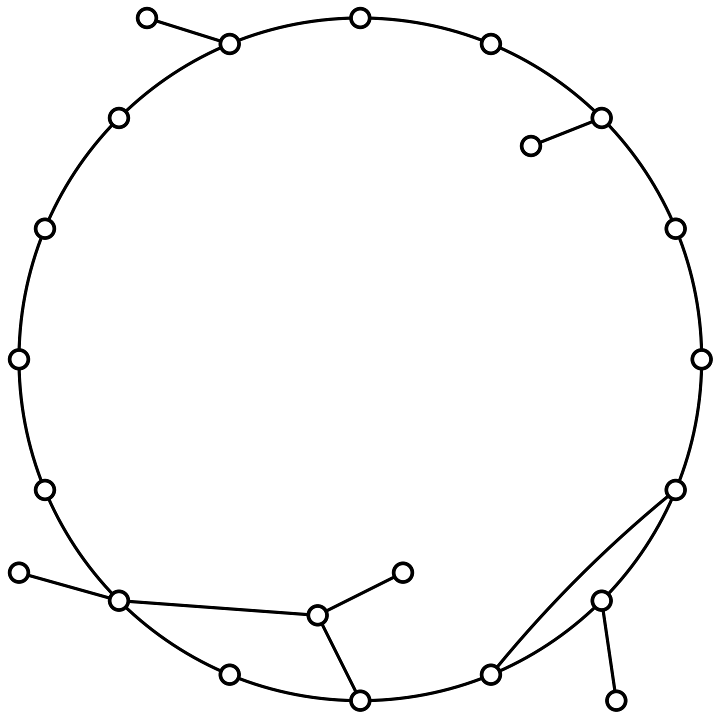
Idee: mache BFS nur von besonders dezentralen Knoten

iFUB Algorithmus

Algorithmen für nette Eingaben



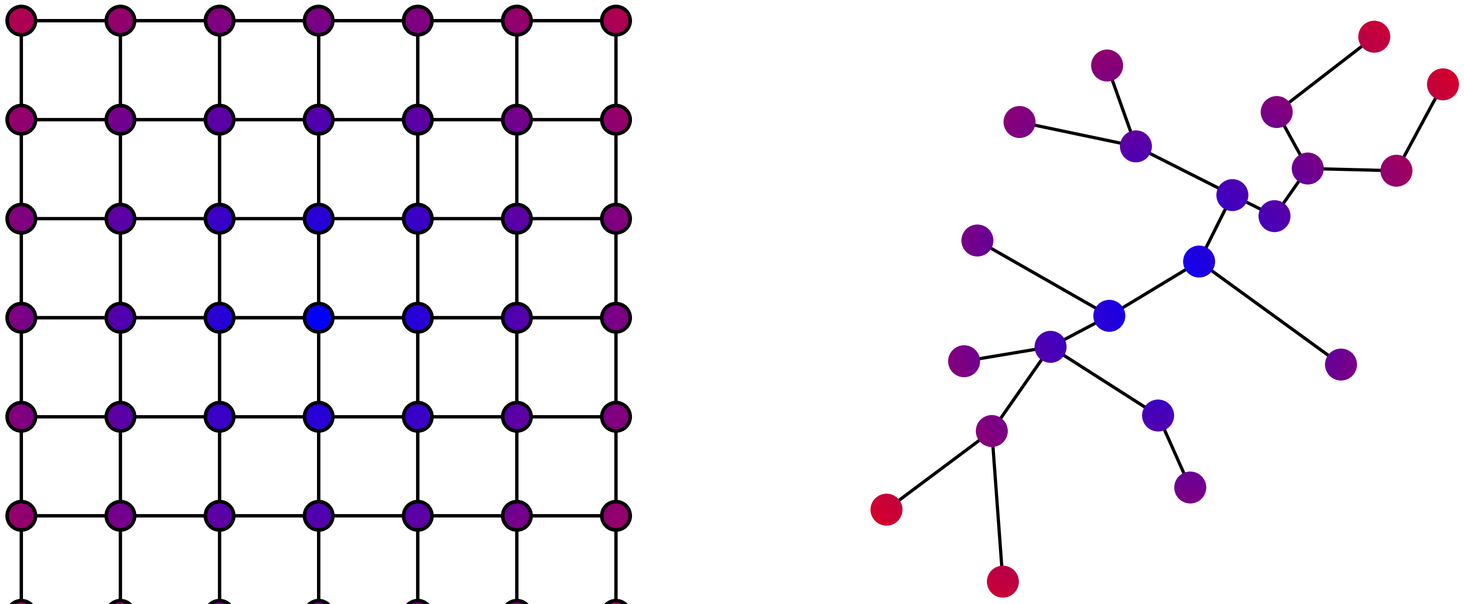


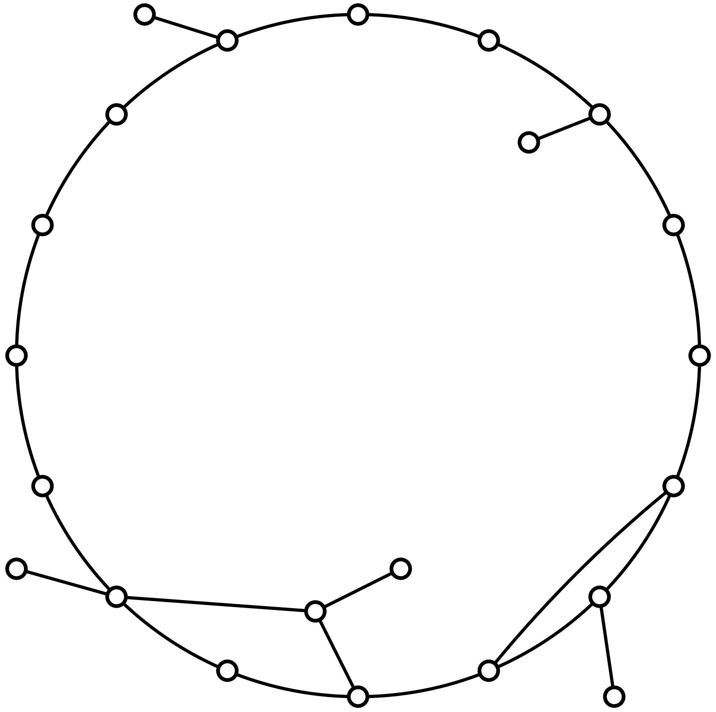


Idee: mache BFS nur von besonders dezentralen Knoten

iFUB Algorithmus

Algorithmen für nette Eingaben



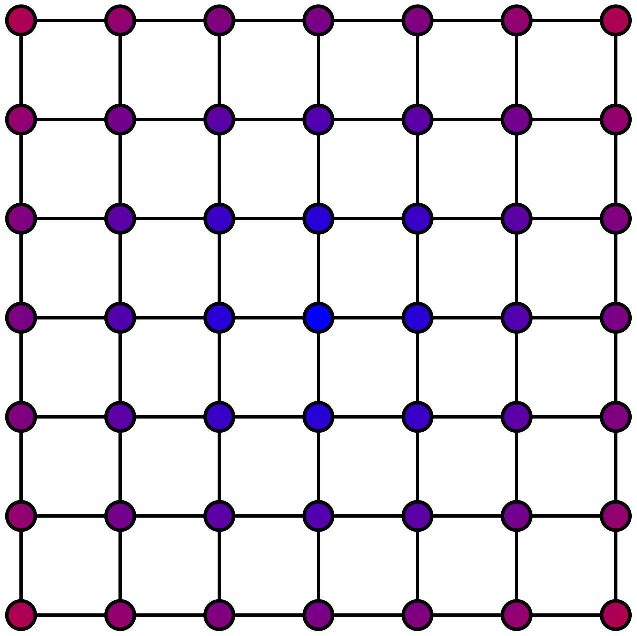


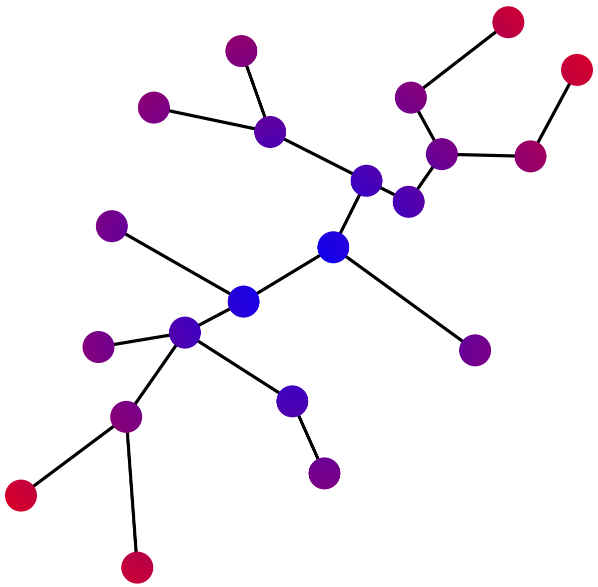
Idee: mache BFS nur von besonders dezentralen Knoten

iFUB Algorithmus

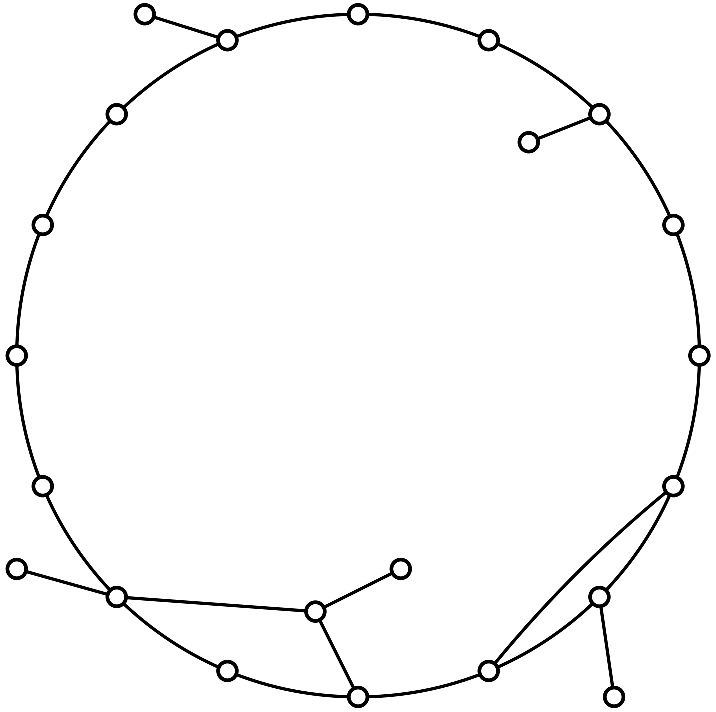
(iterative fringe upper bound, [Crescenzi et al. 2013])

Algorithmen für nette Eingaben





Idee: mache BFS nur von besonders dezentralen Knoten

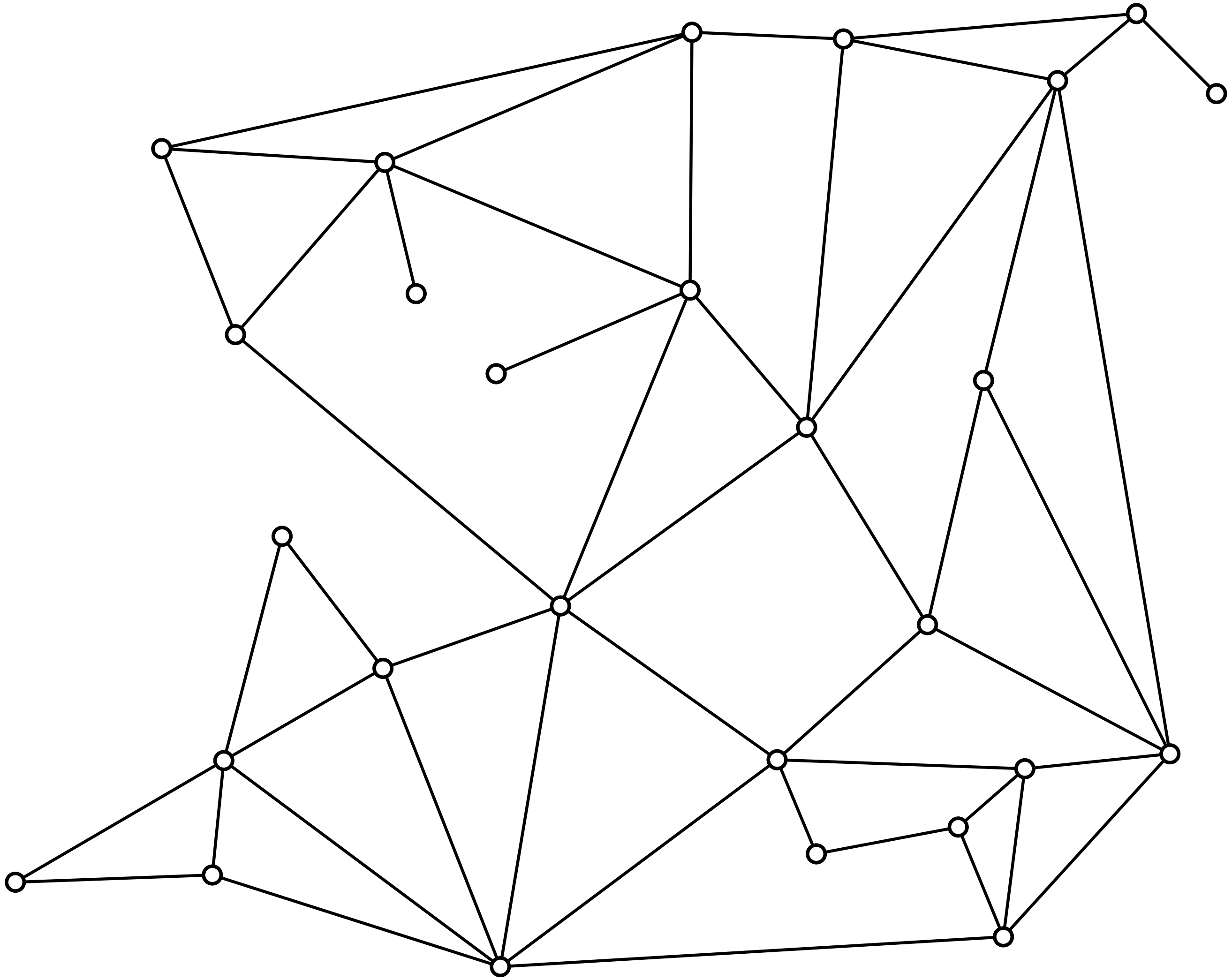


Graphen mit Torus Geometrie:
zu schwer ;-)

iFUB Algorithmus

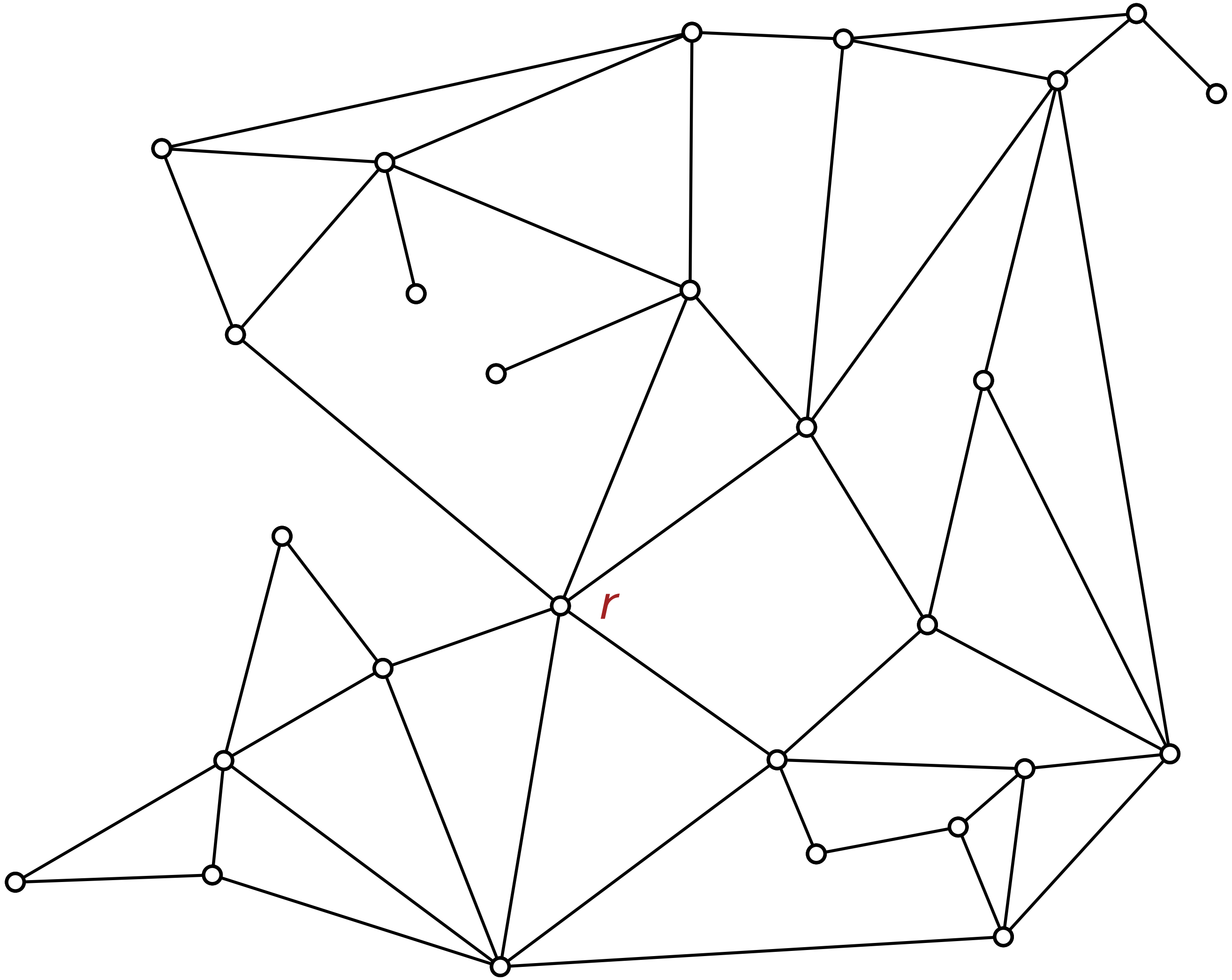
(iterative fringe upper bound, [Crescenzi et al. 2013])

iFUB Algorithmus



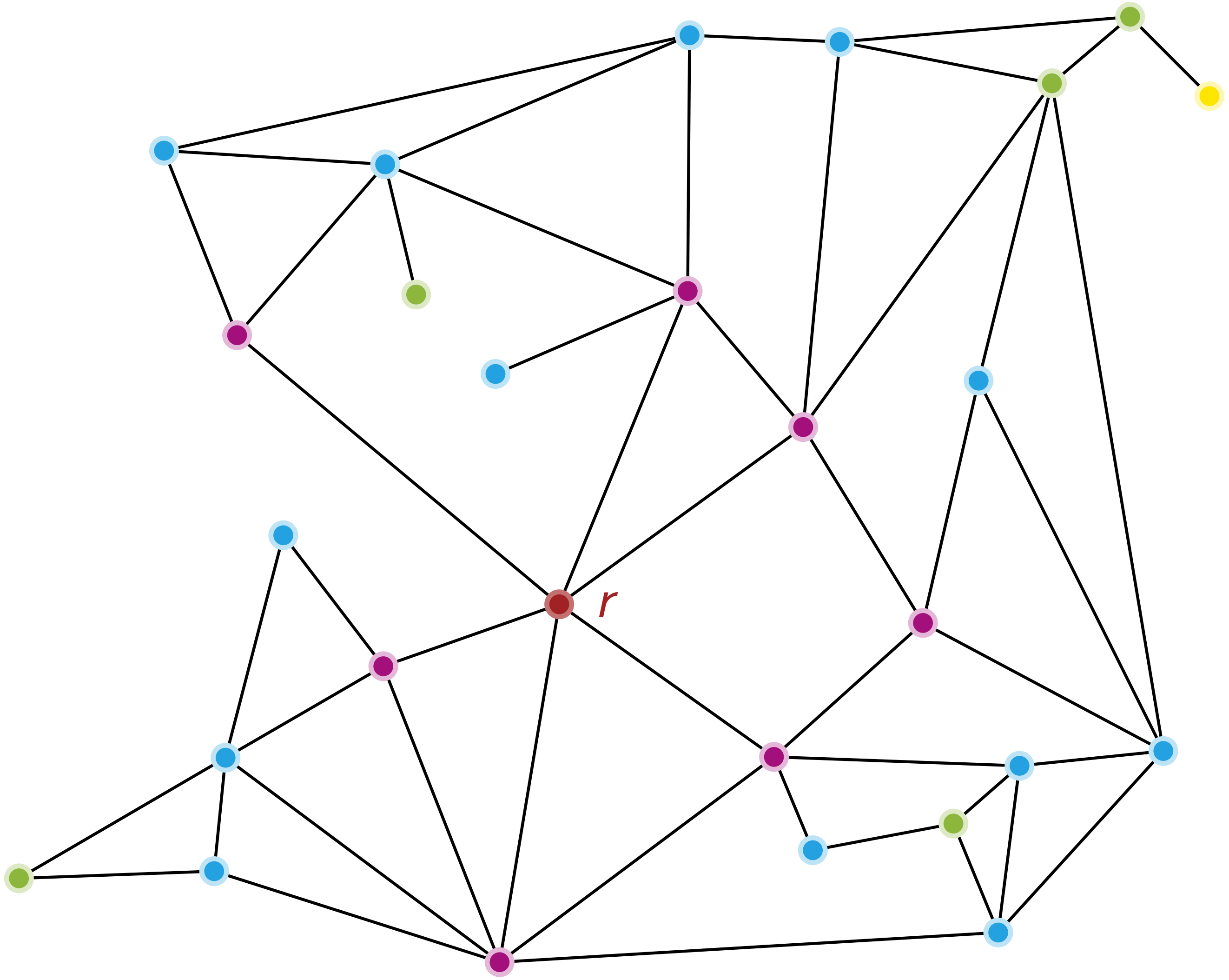
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r



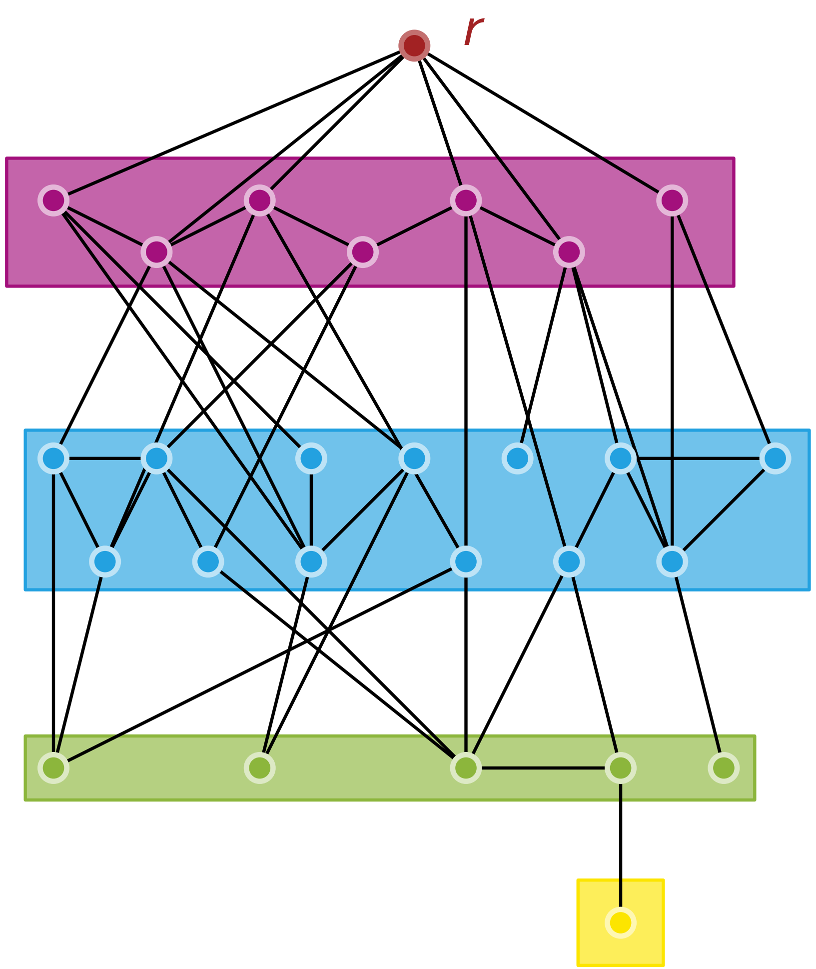
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r



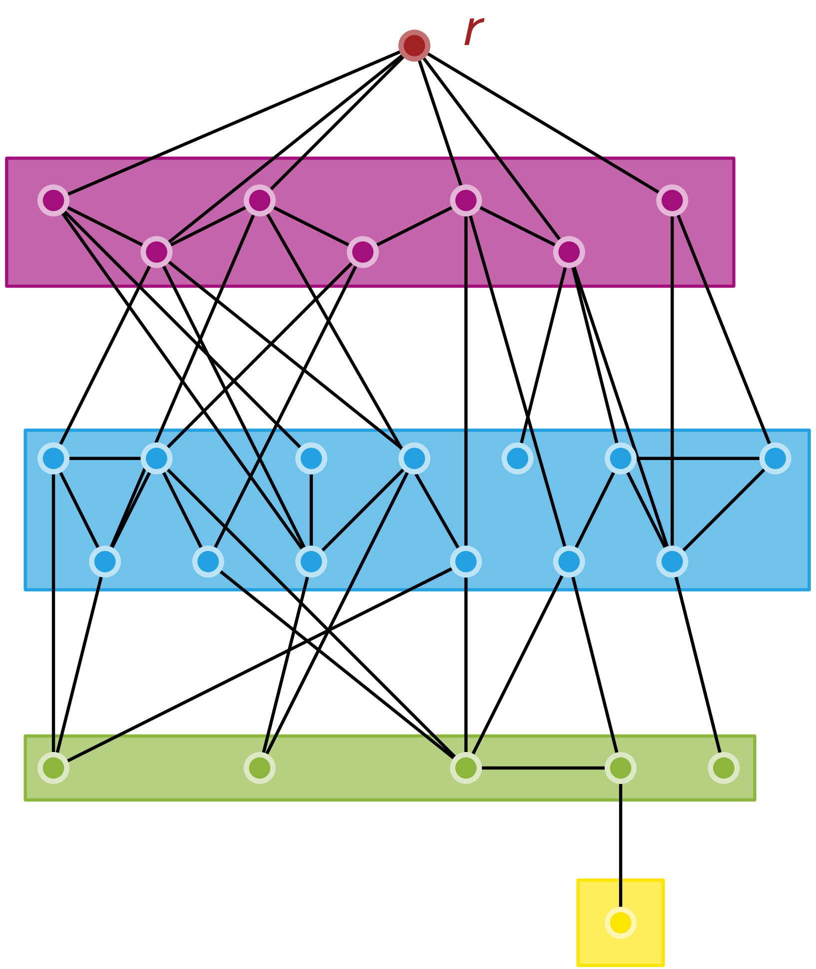
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r



iFUB Algorithmus

- Wähle „*zentralen*“ Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{ecc(r)}$



L_0

L_1

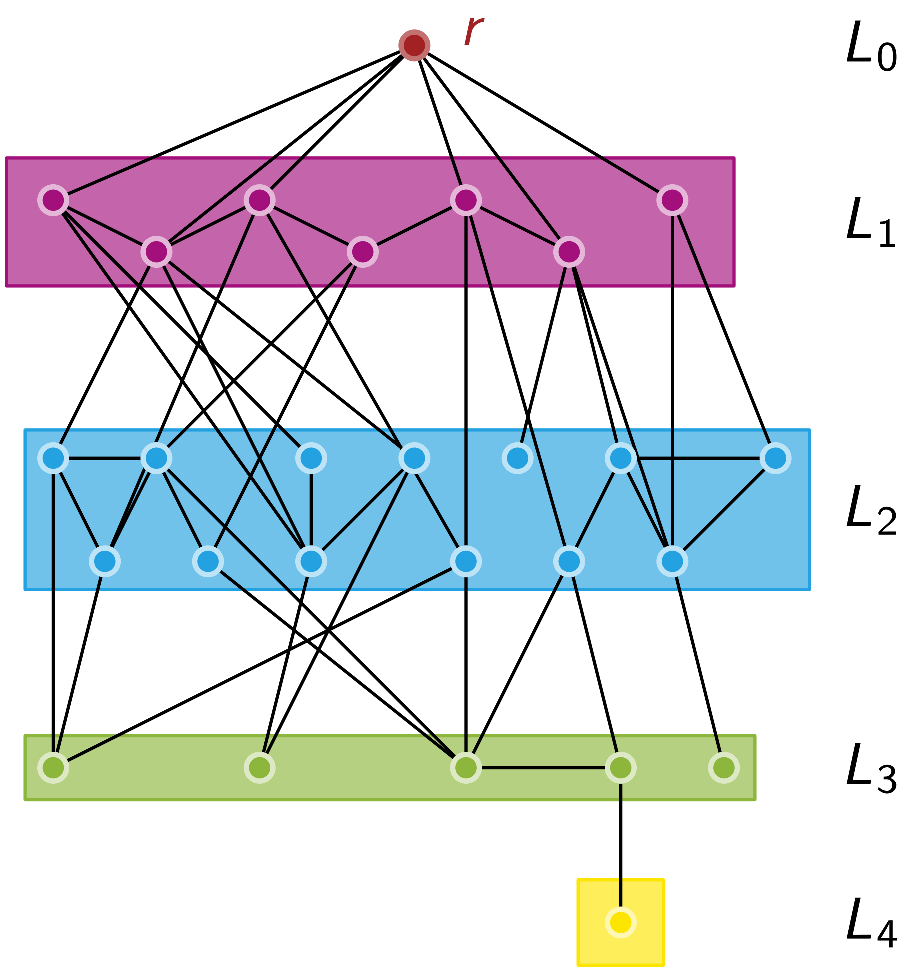
L_2

L_3

L_4

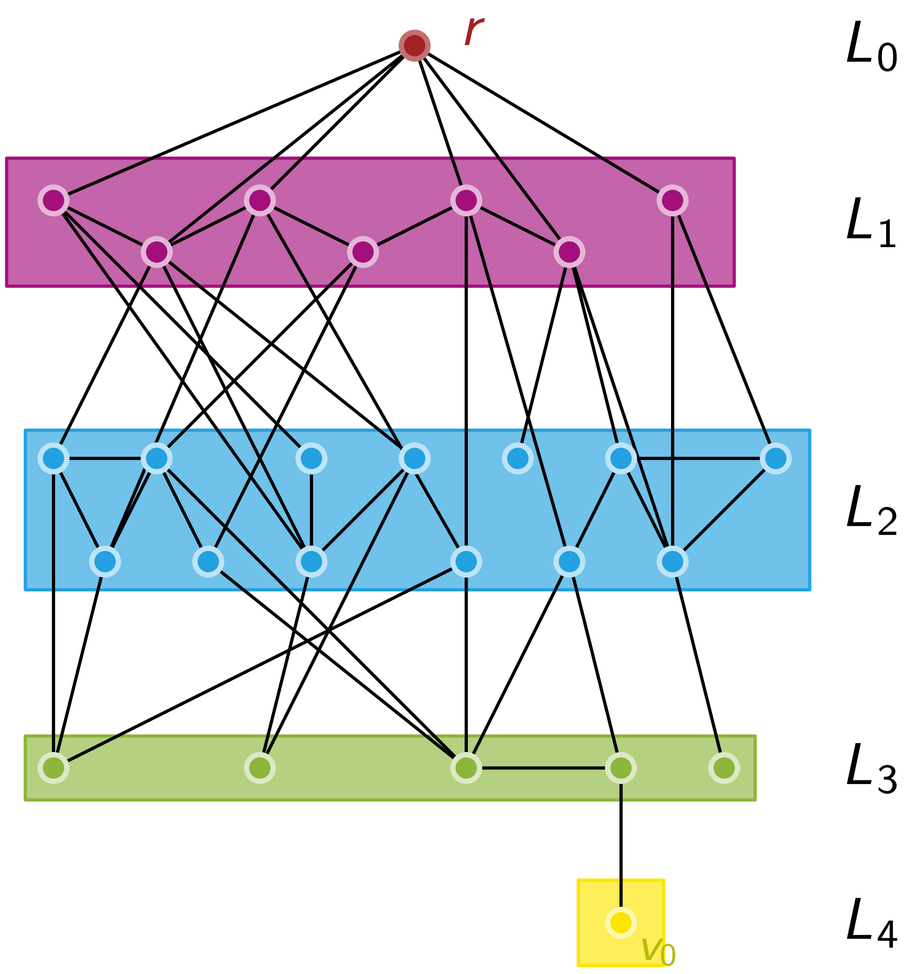
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



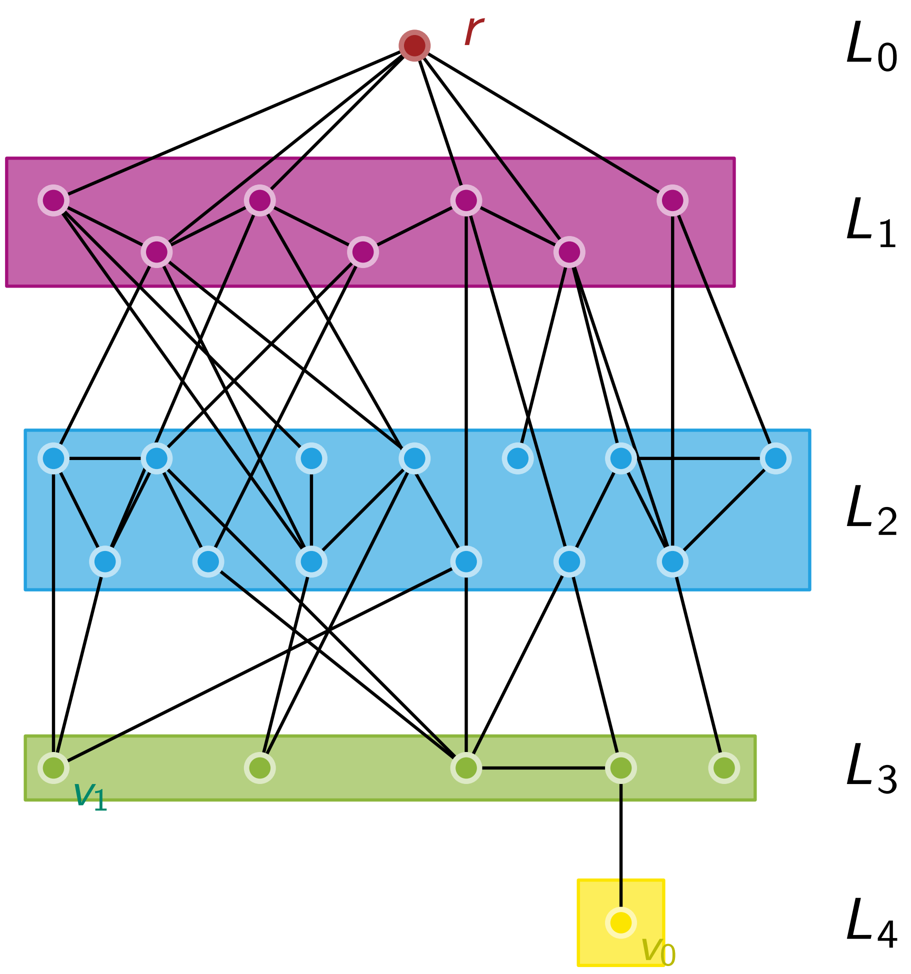
iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



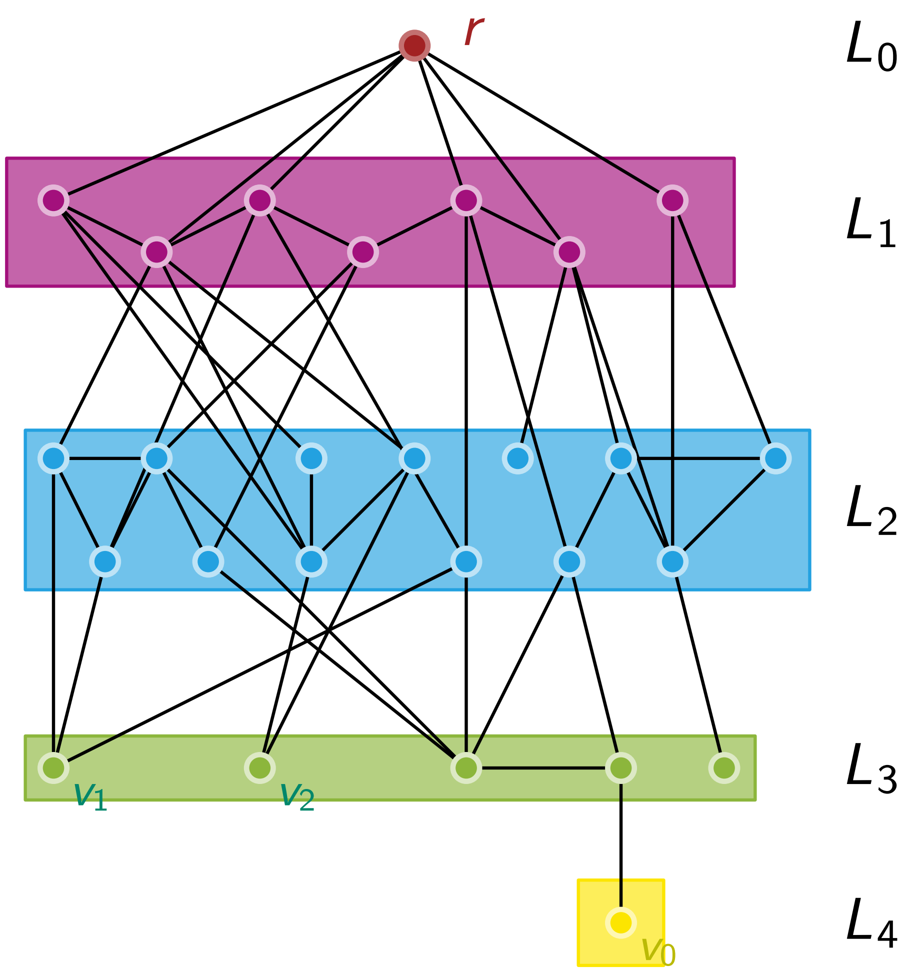
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



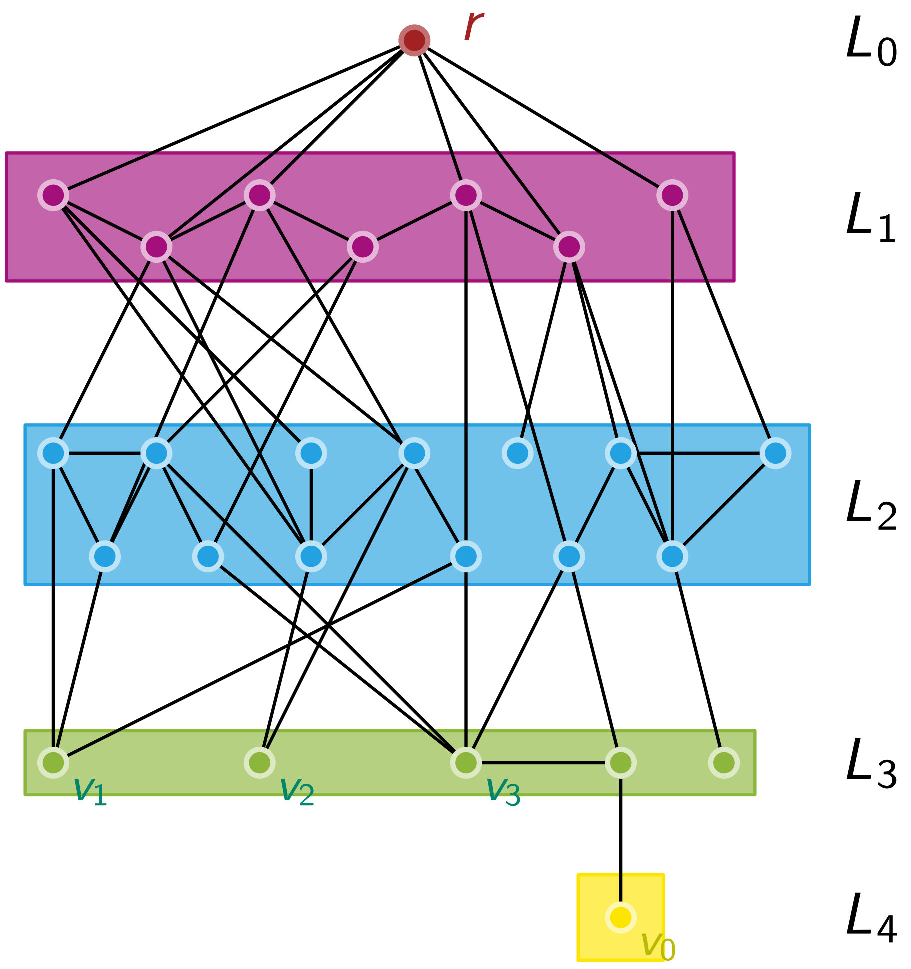
iFUB Algorithmus

- Wähle „*zentralen*“ Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{ecc(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



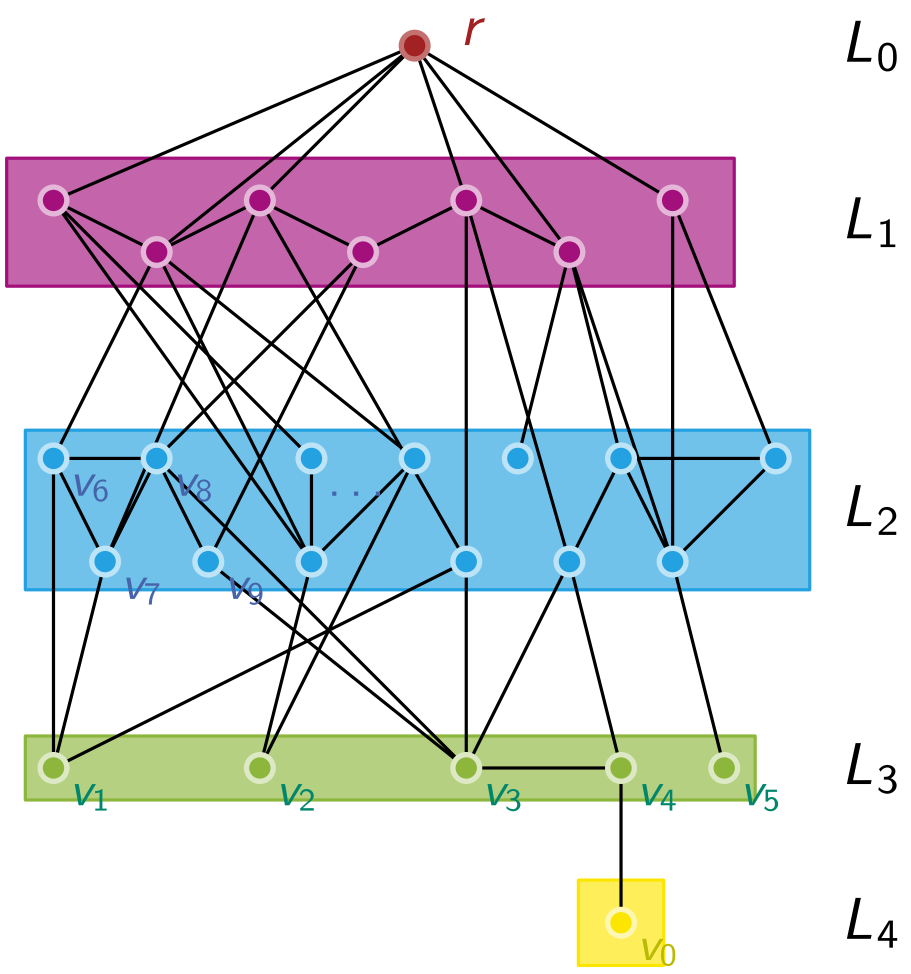
iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{ecc(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



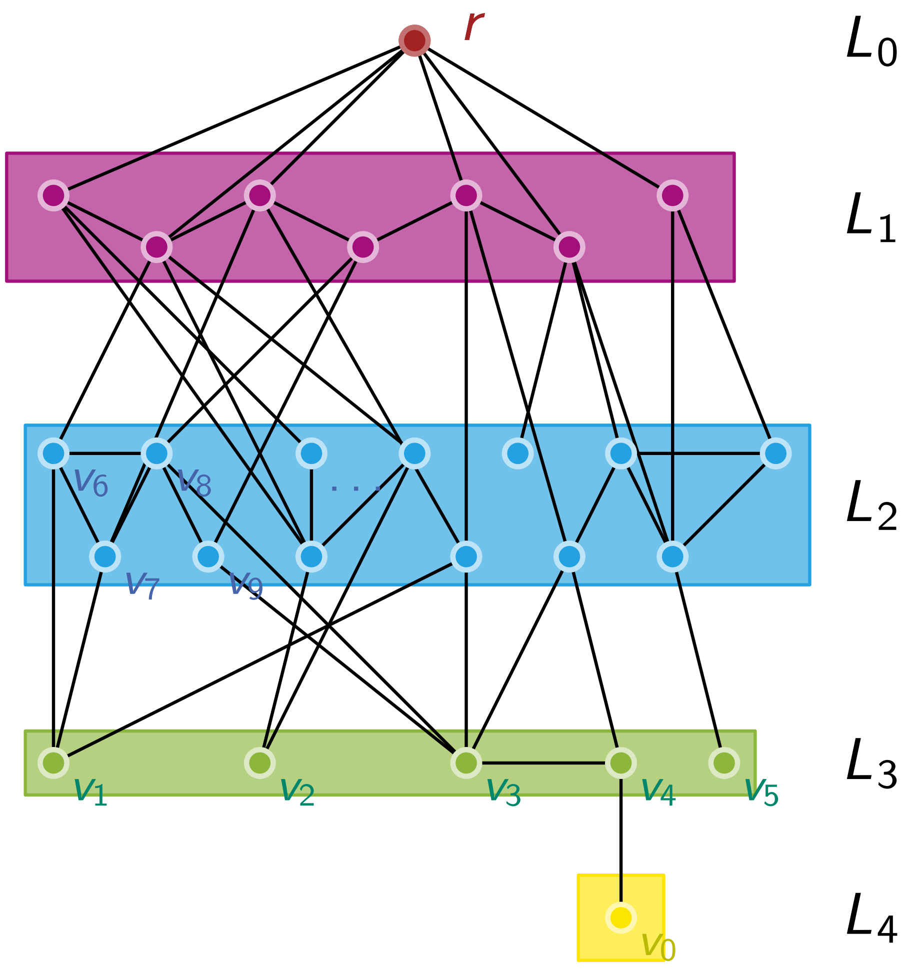
iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{ecc(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$



iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS

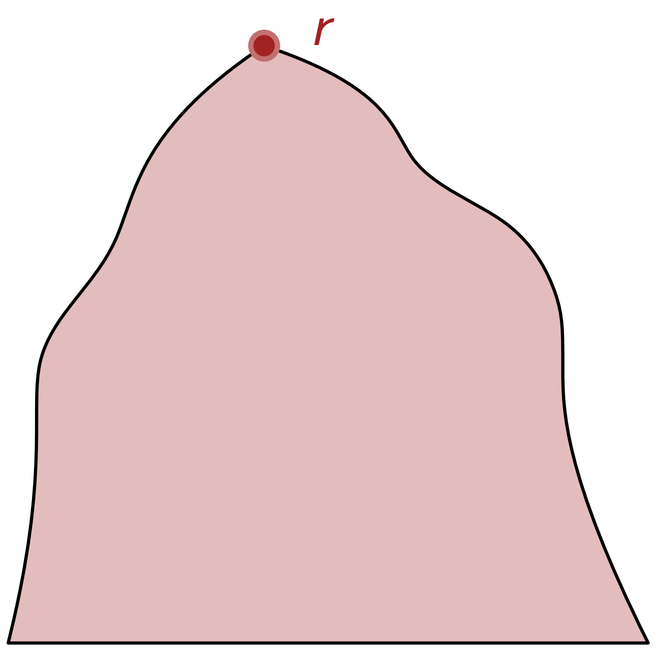


iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS

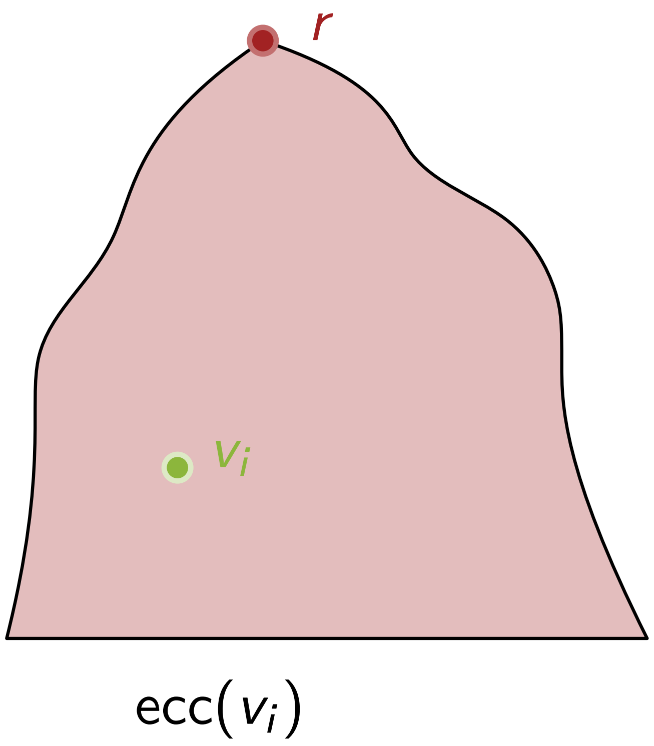
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS



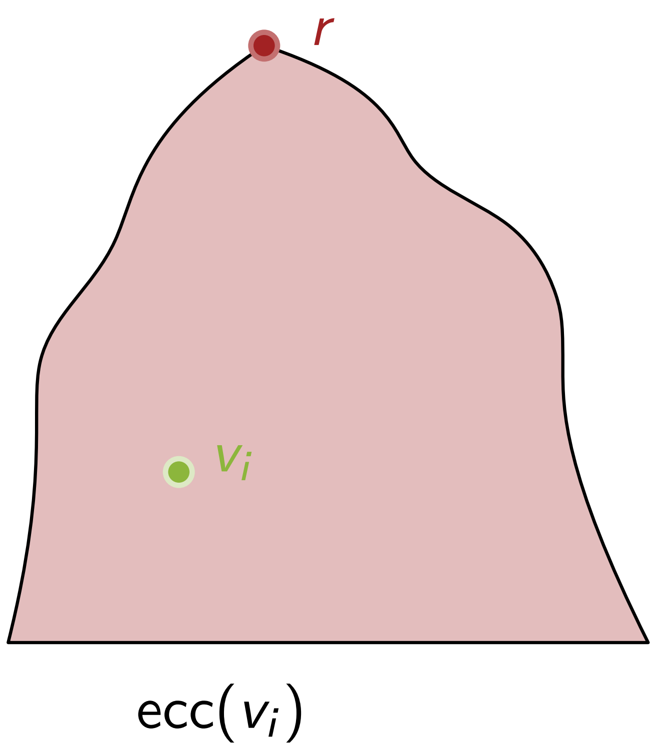
iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS



iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
  - Obere Schranke

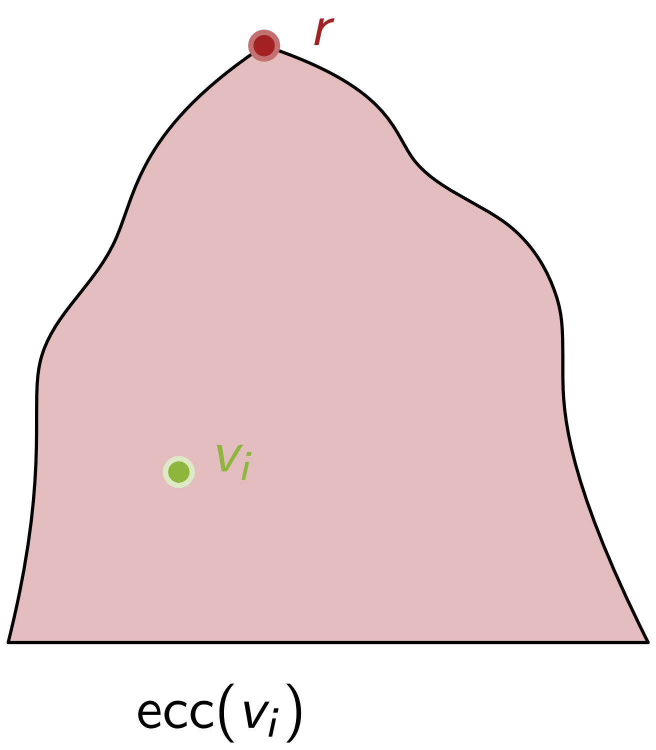


iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \dots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \dots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \dots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

$$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke

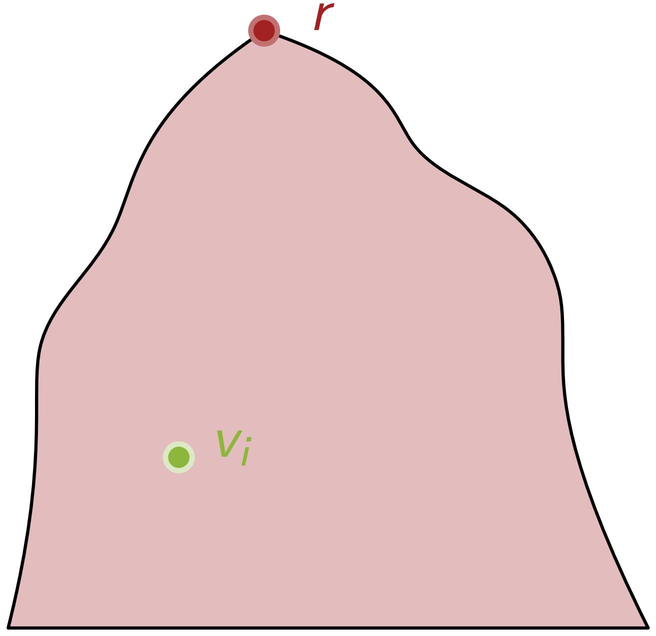


iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

$$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke

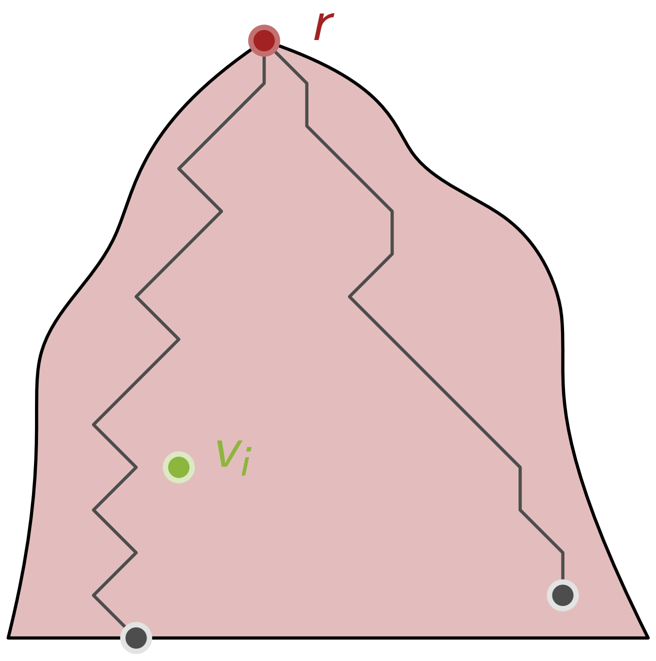


$\mathrm{ecc}(v_i)$

Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke

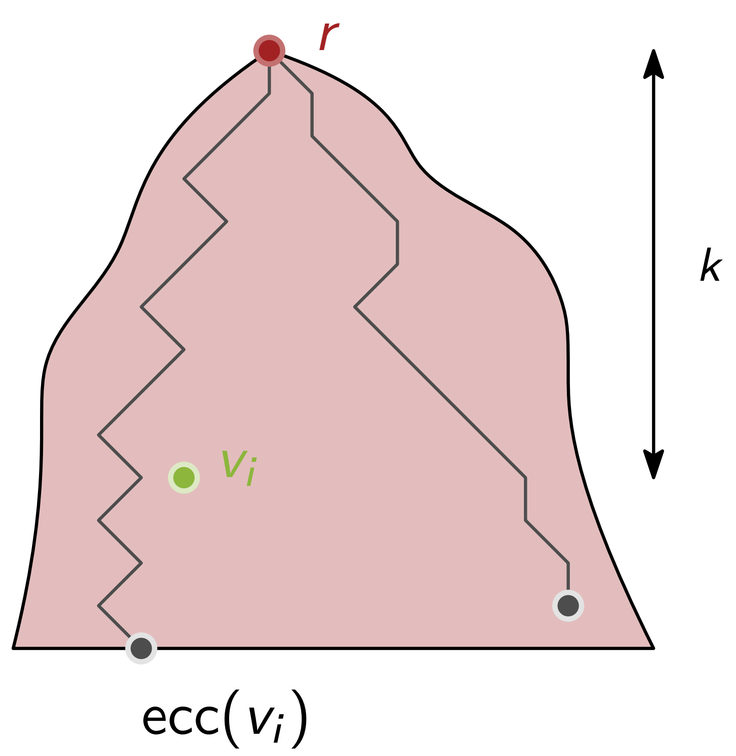


$\mathrm{ecc}(v_i)$

Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke



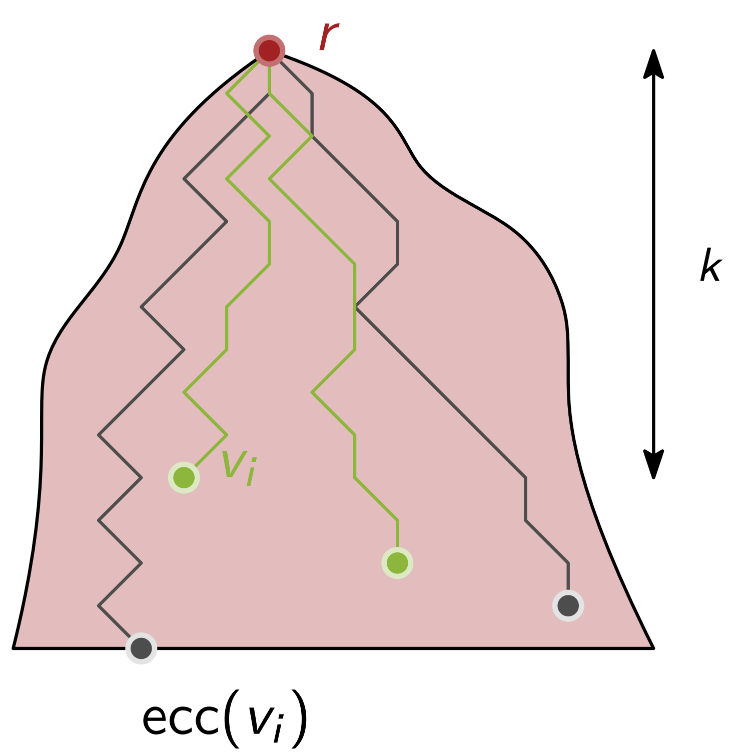
Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke



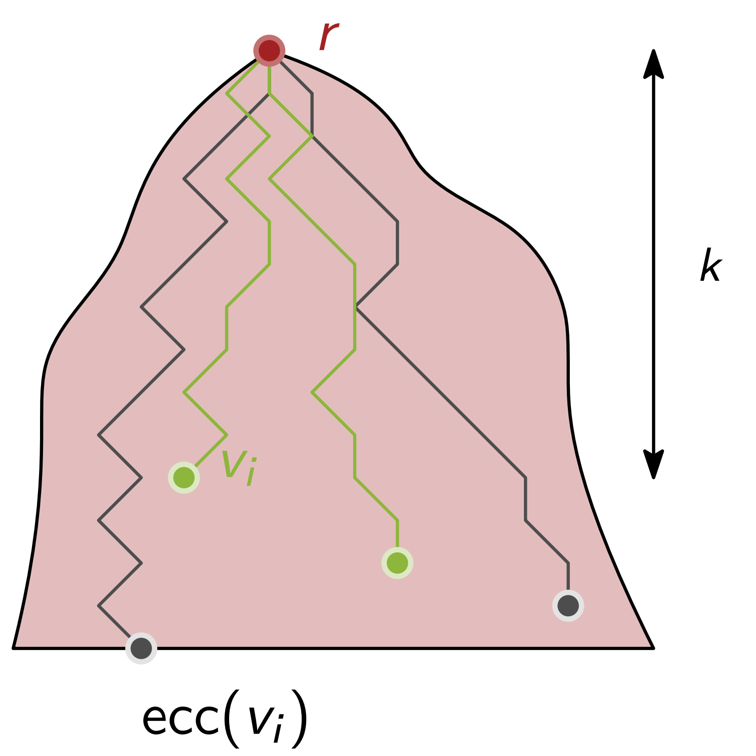
Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$



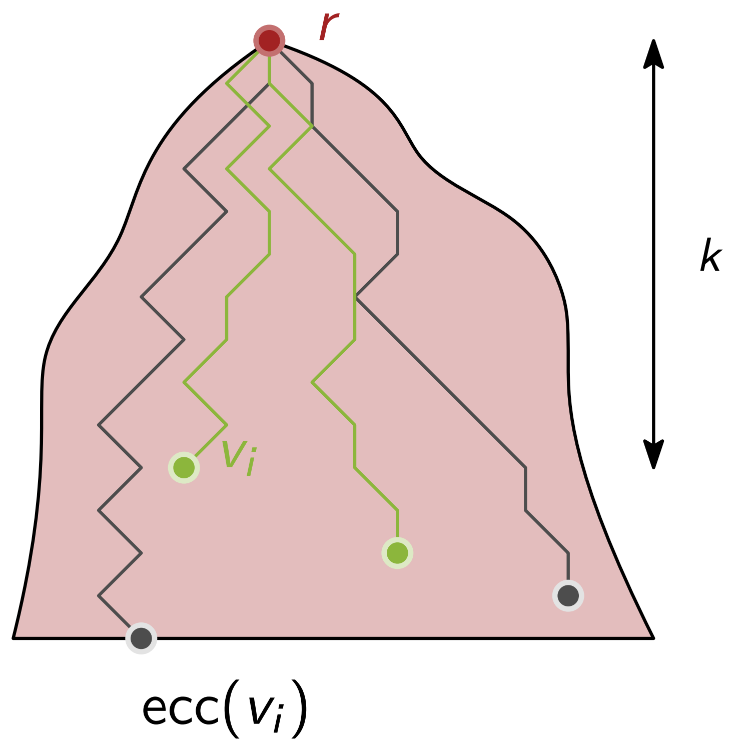
Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:



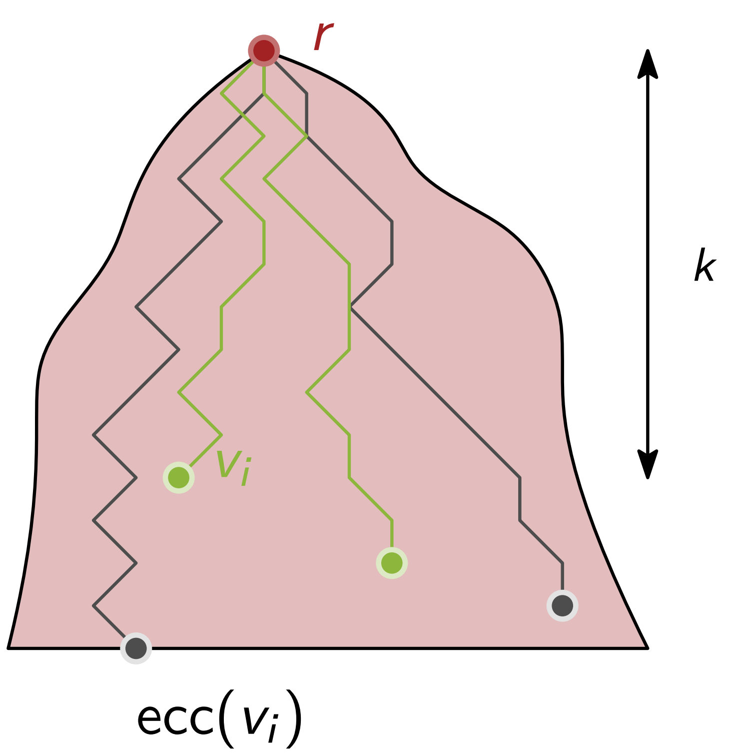
Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

Karlsruher Institut für Technologie

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:
    $$\mathrm{ecc}(v_i) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}\right)$$



Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

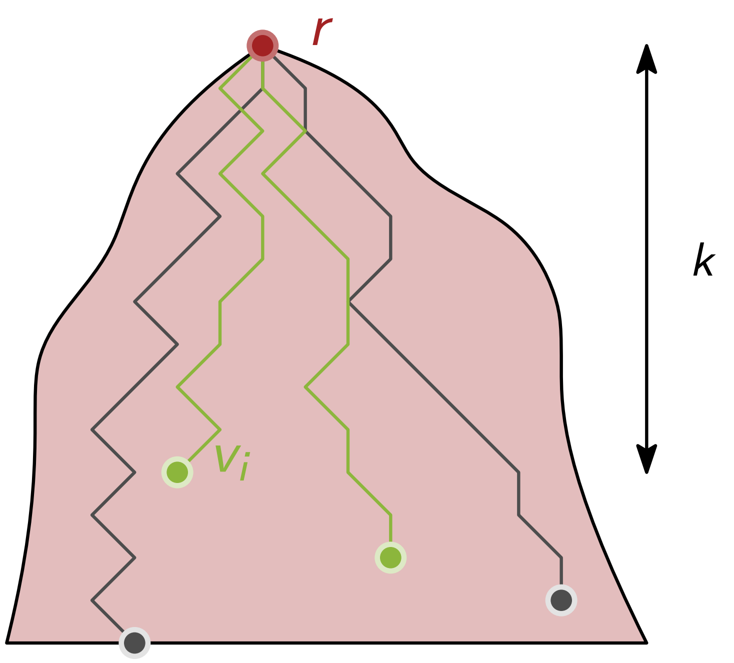
Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

    $$\mathrm{ecc}(v_i) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}\right)$$



$\mathrm{ecc}(v_i)$

Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

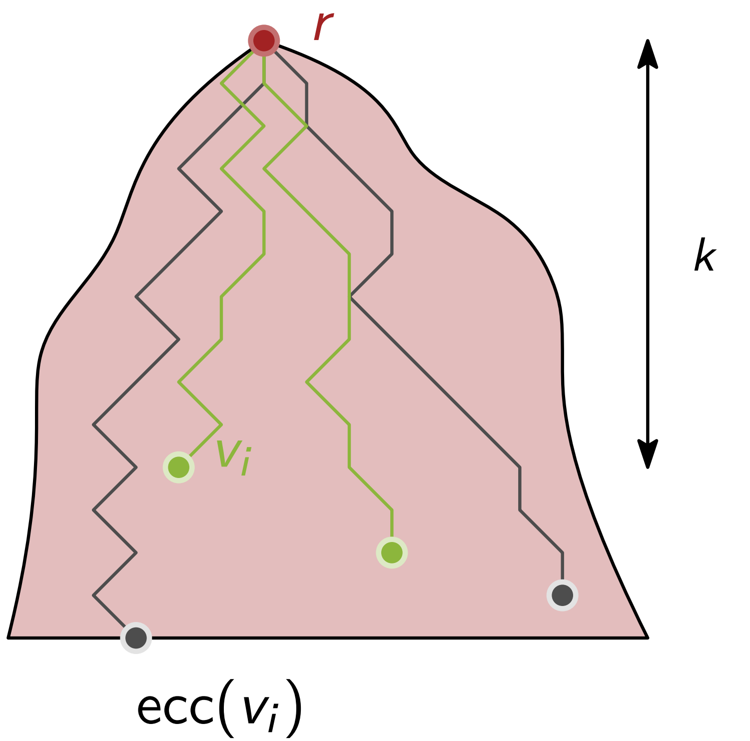
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

$$\mathrm{ecc}(v_i) \leq \max\left(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\right)$$



Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

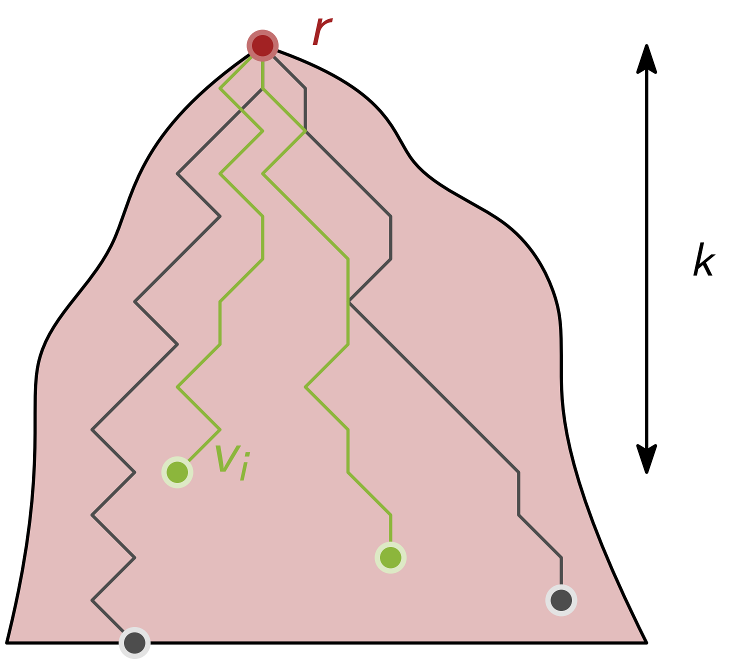
- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

$$\mathrm{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$

$$\mathrm{diam}(G) \leq \max\big(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\big)$$



$\mathrm{ecc}(v_i)$

Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

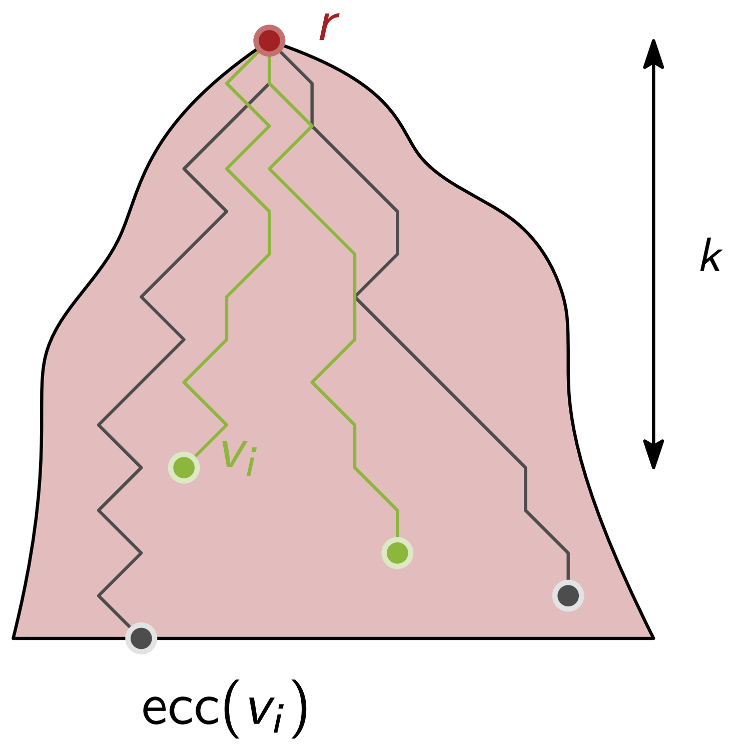
iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

$$\mathrm{ecc}(v_i) \leq \max\left(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\right)$$

$$\mathrm{diam}(G) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\right)$$

- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$



Kann $\mathrm{diam}(G) > 2 \cdot \mathrm{ecc}(r)$ *sein?*

Kann $\mathrm{ecc}(v_i) > 2 \cdot k$ *sein?*

iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:
    $$\mathrm{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$
    $$\mathrm{diam}(G) \leq \max\big(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\big)$$
- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\text{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\text{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\text{diam}(G) \geq \max\{\text{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\text{ecc}(v_i)$, $\text{diam}(G)$
    - sei v_i in Layer L_k, dann:
    $$\text{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\text{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$
    $$\text{diam}(G) \leq \max\left(2k, \max\{\text{ecc}(v) \mid v \in V_i\}\right)$$
- Abbruch sobald $2k \leq \max\{\text{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

iFUB Algorithmus

Karlsruher Institut für Technologie

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

    $$\mathrm{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$

    $$\mathrm{diam}(G) \leq \max\big(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\big)$$
- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)

iFUB Algorithmus

- Wähle *„zentralen"* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \dots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \dots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \dots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

    $$\mathrm{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$

    $$\mathrm{diam}(G) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\right)$$

- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

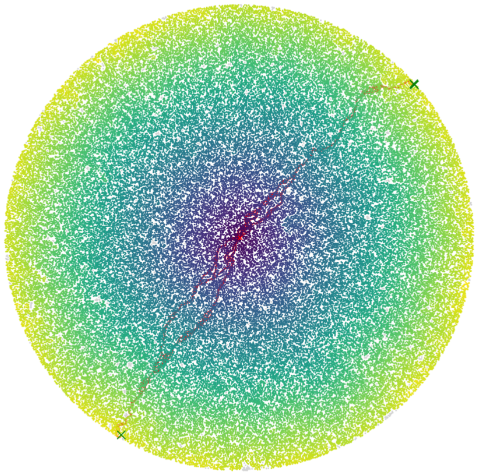
- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)

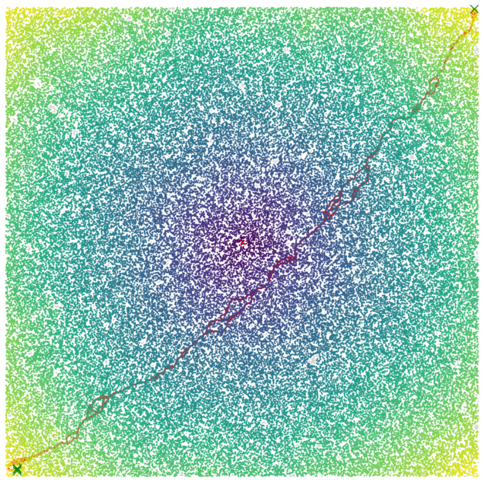
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:
    $$\mathrm{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$
    $$\mathrm{diam}(G) \leq \max\big(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\big)$$
- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)





iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$

  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

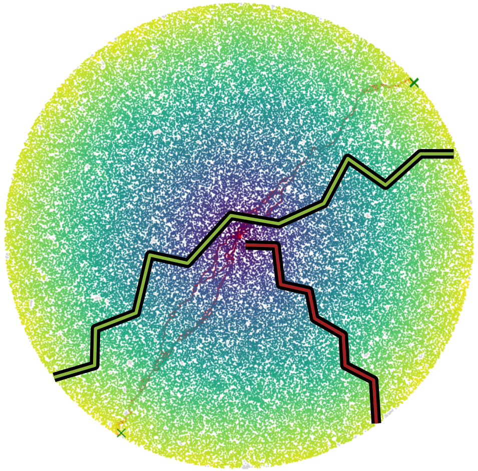
    $$\mathrm{ecc}(v_i) \leq \max\left(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\right)$$

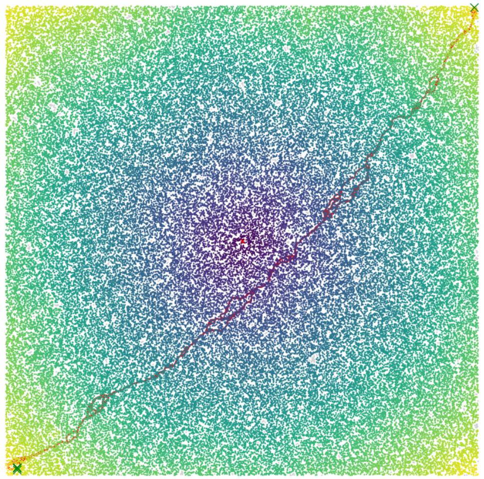
    $$\mathrm{diam}(G) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\right)$$

- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)





$\mathrm{ecc}(r)$ ist klein

$\mathrm{ecc}(v_0)$ ist groß

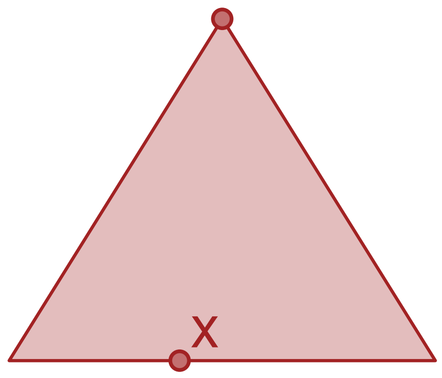
iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke
    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:
    $$\mathrm{ecc}(v_i) \leq \max\left(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\right)$$
    $$\mathrm{diam}(G) \leq \max\left(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\}\right)$$
- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)

- BFS: Finde Knoten x am *Rand*



iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\mathrm{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\mathrm{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\mathrm{diam}(G) \geq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\mathrm{ecc}(v_i)$, $\mathrm{diam}(G)$
    - sei v_i in Layer L_k, dann:

$$\mathrm{ecc}(v_i) \leq \max(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\mathrm{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}})$$

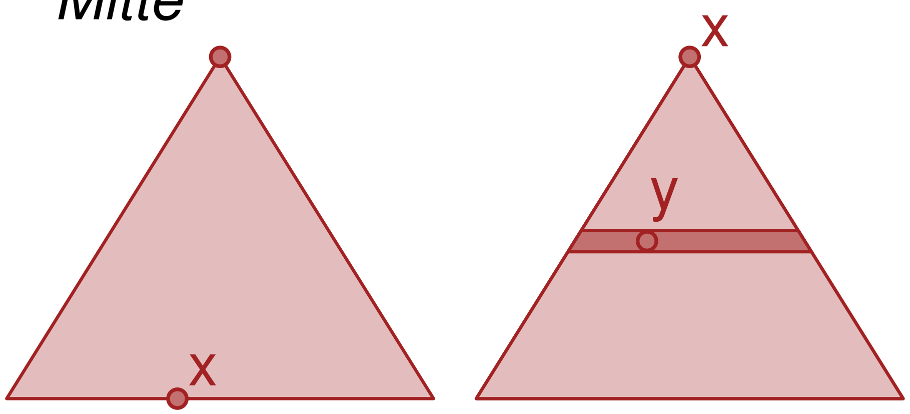
$$\mathrm{diam}(G) \leq \max(2k, \max\{\mathrm{ecc}(v) \mid v \in V_i\})$$

- Abbruch sobald $2k \leq \max\{\mathrm{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)

- BFS: Finde Knoten x am *Rand*
- BFS von x: Finde Knoten in der *Mitte*



iFUB Algorithmus

- Wähle *„zentralen“* Knoten r
- Breitensuche von r, Layer $L_0, L_1, \ldots, L_{\text{ecc}(r)}$
- Sei $\langle v_0, \ldots, v_{n-1} \rangle$ Reihenfolge der Knoten im Baum von unten nach oben, $V_i = \{v_0, \ldots, v_i\}$
- Berechne $\text{ecc}(v_i)$ mittels BFS
  - Untere Schranke

    $$\text{diam}(G) \geq \max\{\text{ecc}(v) \mid v \in V_i\}$$
  - Obere Schranke an: $\text{ecc}(v_i)$, $\text{diam}(G)$
    - sei v_i in Layer L_k, dann:

    $$\text{ecc}(v_i) \leq \max\big(\underbrace{2k}_{\text{(schrumpft)}}, \underbrace{\max\{\text{ecc}(v) \mid v \in V_{i-1}\}}_{\text{wächst}}\big)$$

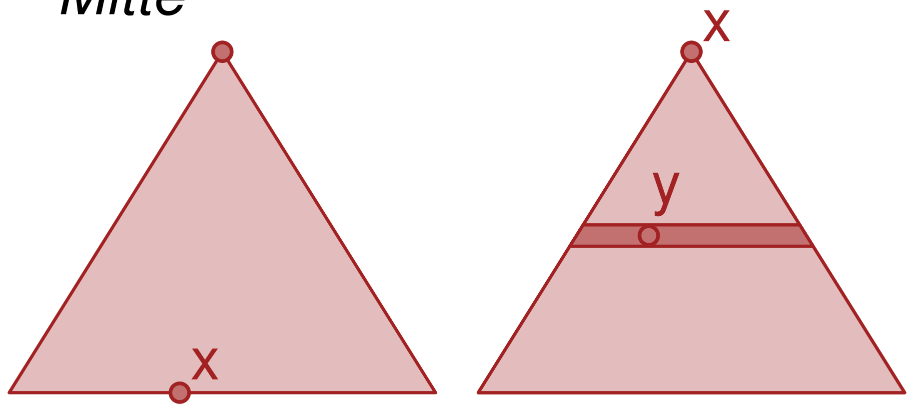
    $$\text{diam}(G) \leq \max\big(2k, \max\{\text{ecc}(v) \mid v \in V_i\}\big)$$
- Abbruch sobald $2k \leq \max\{\text{ecc}(v) \mid v \in V_i\}$

Frage: wie wählen wir r?

- höchster Grad (Heterogenität)
- mittlerer Knoten (Lokalität)

ggf. iterieren
2-sweep
4-sweep

- BFS: Finde Knoten x am *Rand*
- BFS von x: Finde Knoten in der *Mitte*



Übungsblatt 3 und Projekt

Übungsblatt 3

- Code und Workflow optimieren / aufräumen
- Performance von iFUB auf realistischen Eingaben untersuchen
- Zeitrahmen: nur *eine* Woche

Übungsblatt 3 und Projekt

Übungsblatt 3

- Code und Workflow optimieren / aufräumen
- Performance von iFUB auf realistischen Eingaben untersuchen
- Zeitrahmen: nur *eine* Woche

Anschließend: Projekt (6 Wochen)

- Ziel: Forschungsfrage untersuchen und beantworten

Übungsblatt 3 und Projekt

KIT
Karlsruher Institut für Technologie

Übungsblatt 3

- Code und Workflow optimieren / aufräumen
- Performance von iFUB auf realistischen Eingaben untersuchen
- Zeitrahmen: nur *eine* Woche

Anschließend: Projekt (6 Wochen)

- Ziel: Forschungsfrage untersuchen und beantworten
- Inspiration:
  - On the External Validity of Average-Case Analyses of Graph Algorithms [B., F. 2022]

Übungsblatt 3 und Projekt

Übungsblatt 3

- Code und Workflow optimieren / aufräumen
- Performance von iFUB auf realistischen Eingaben untersuchen
- Zeitrahmen: nur *eine* Woche

Anschließend: Projekt (6 Wochen)

- Ziel: Forschungsfrage untersuchen und beantworten
- Inspiration:
  - On the External Validity of Average-Case Analyses of Graph Algorithms [B., F. 2022]
  - Deterministic Performance Guarantees for Bidirectional BFS on Real-World Networks [B., W. 2022]

Übungsblatt 3 und Projekt

Übungsblatt 3

- Code und Workflow optimieren / aufräumen
- Performance von iFUB auf realistischen Eingaben untersuchen
- Zeitrahmen: nur *eine* Woche

Anschließend: Projekt (6 Wochen)

- Ziel: Forschungsfrage untersuchen und beantworten
- Inspiration:
  - On the External Validity of Average-Case Analyses of Graph Algorithms [B., F. 2022]
  - Deterministic Performance Guarantees for Bidirectional BFS on Real-World Networks [B., W. 2022]
  - Algorithmen verstehen / verbessern (z.B. diameter, schwere Probleme, . . .)