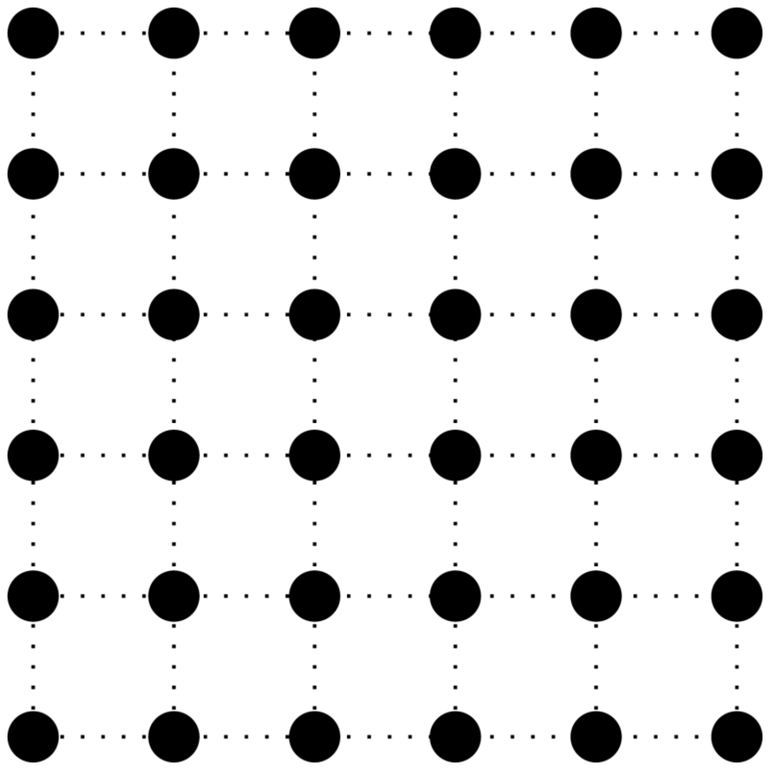
Aufgabe 6 a)

Wie viele Knoten/Kanten im Gittergraph?



Aufgabe 6 b)c)

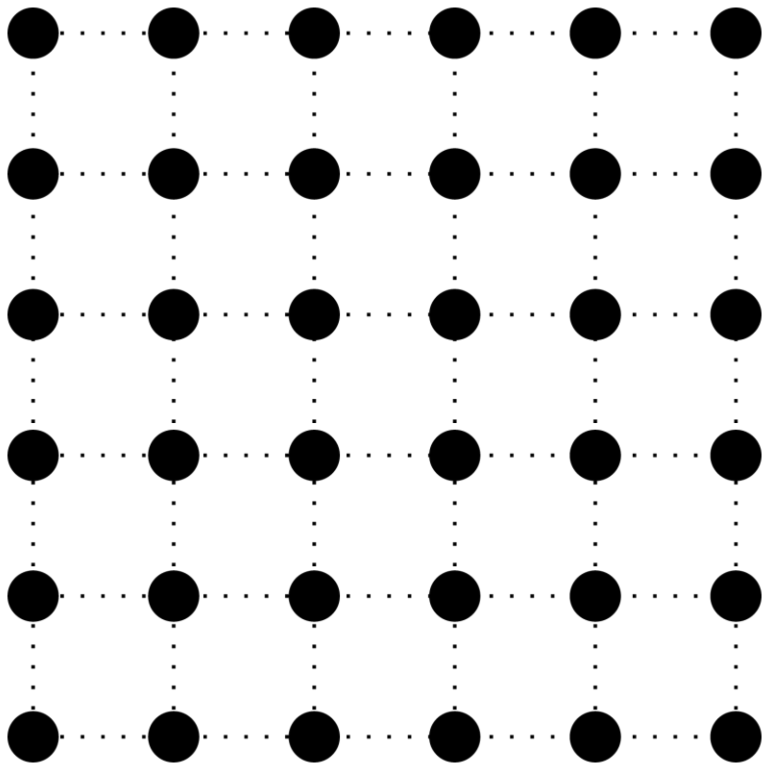
Labyrinth als Gittergraph modellieren

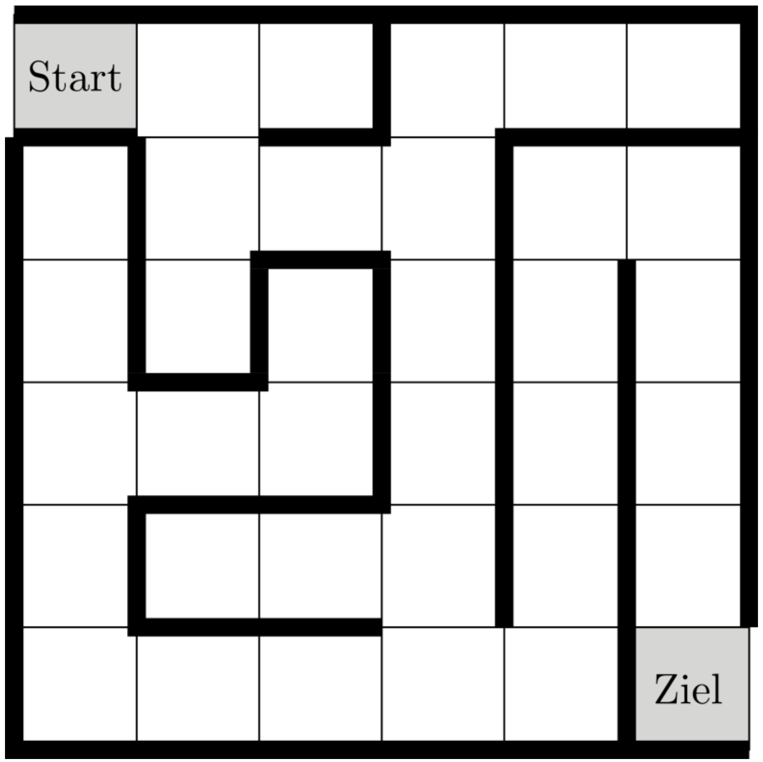
Im Labyrinth kann man nur in ein Nachbarfeld gehen
Im Gittergraph gibt es Kanten nur zwischen Nachbarknoten

Kante zwischen Feldern, zwischen denen keine Wand steht

Aufgabe 6 b)c)

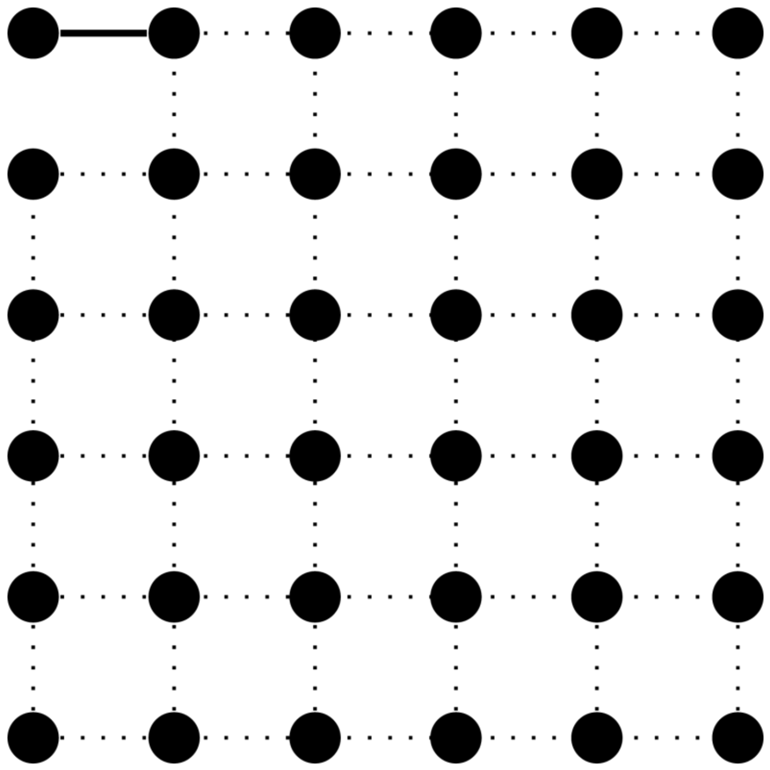
Labyrinth als Gittergraph modellieren

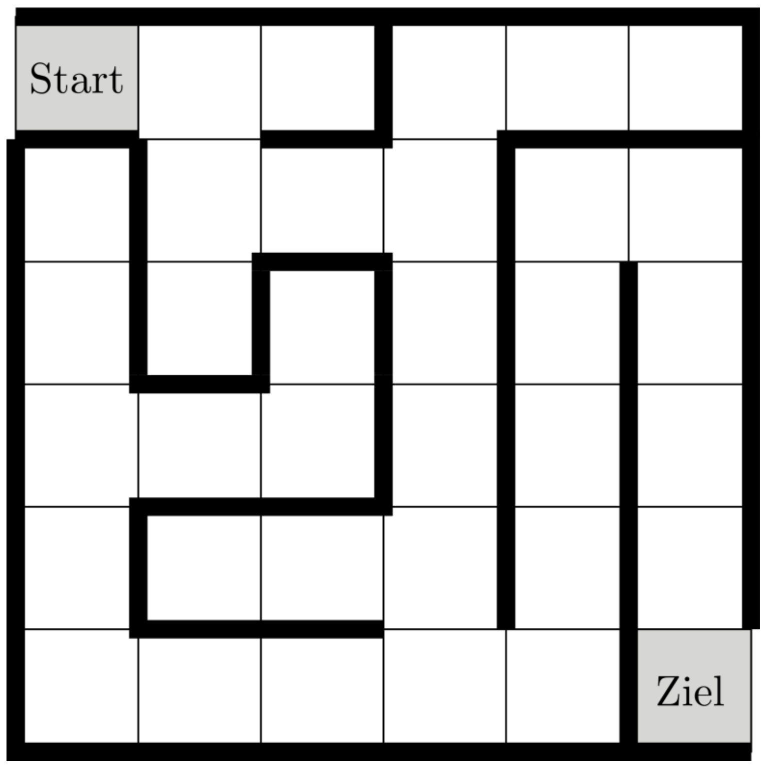




Aufgabe 6 b)c)

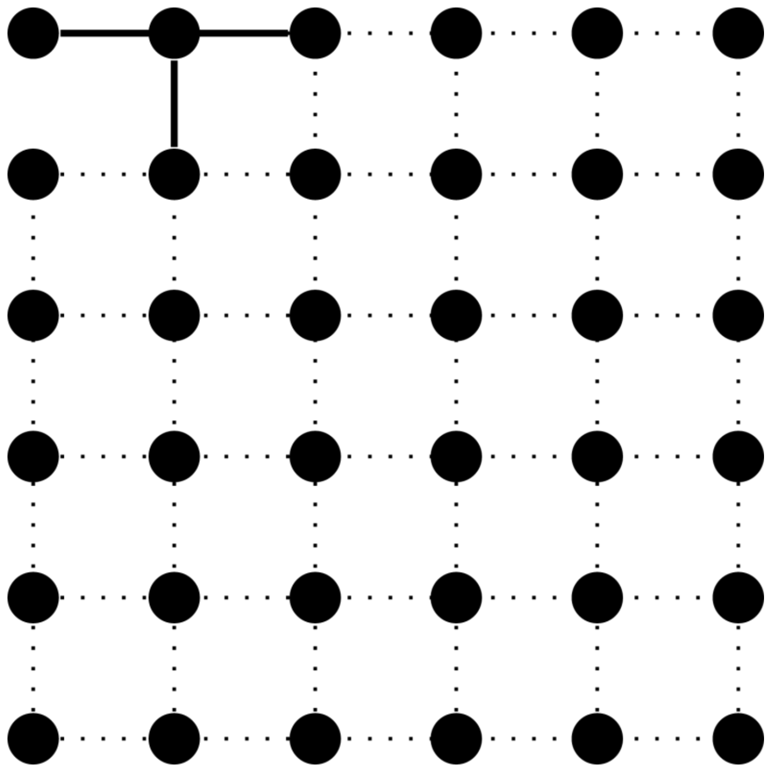
Labyrinth als Gittergraph modellieren

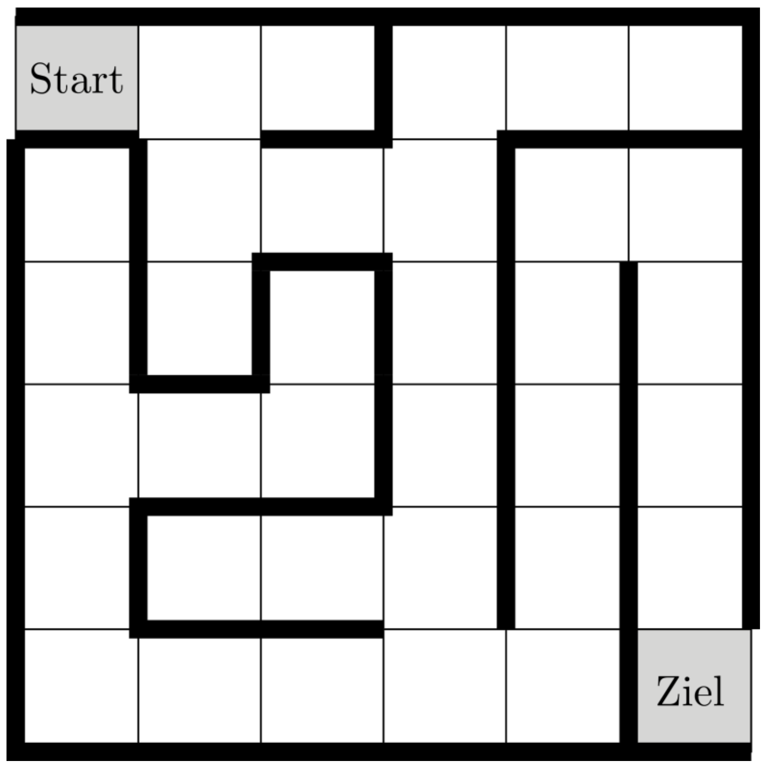




Aufgabe 6 b)c)

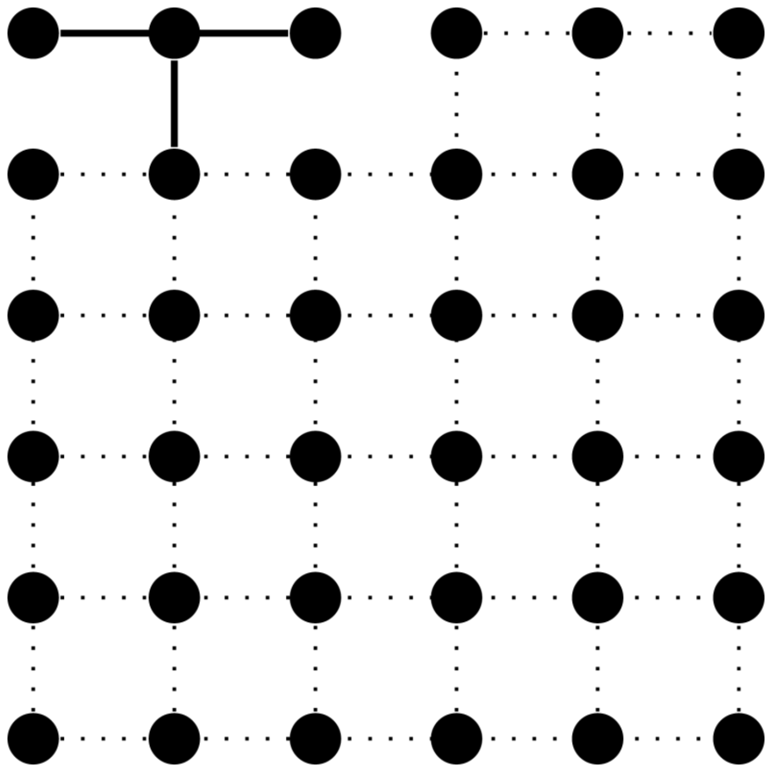
Labyrinth als Gittergraph modellieren

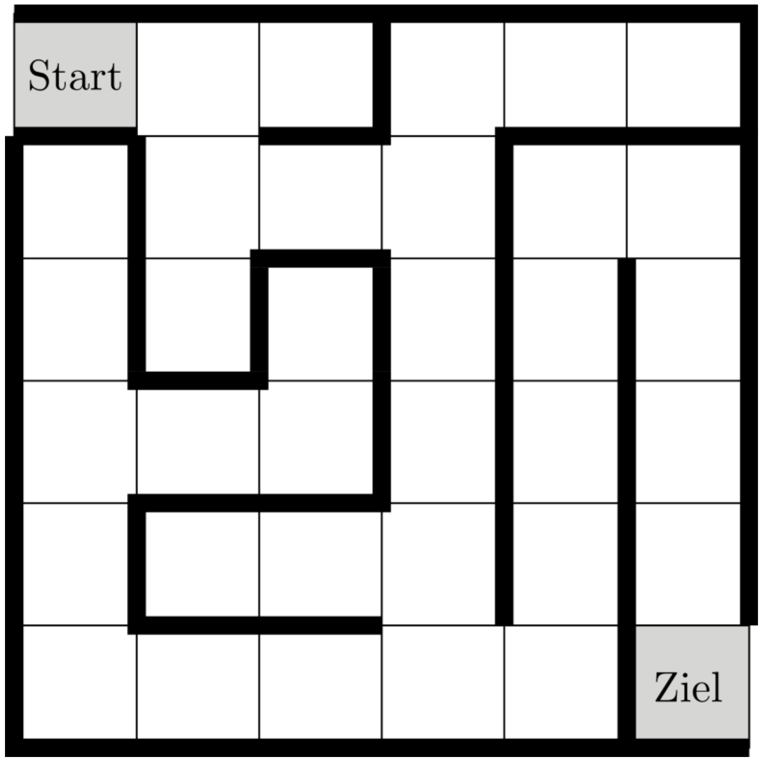




Aufgabe 6 b)c)

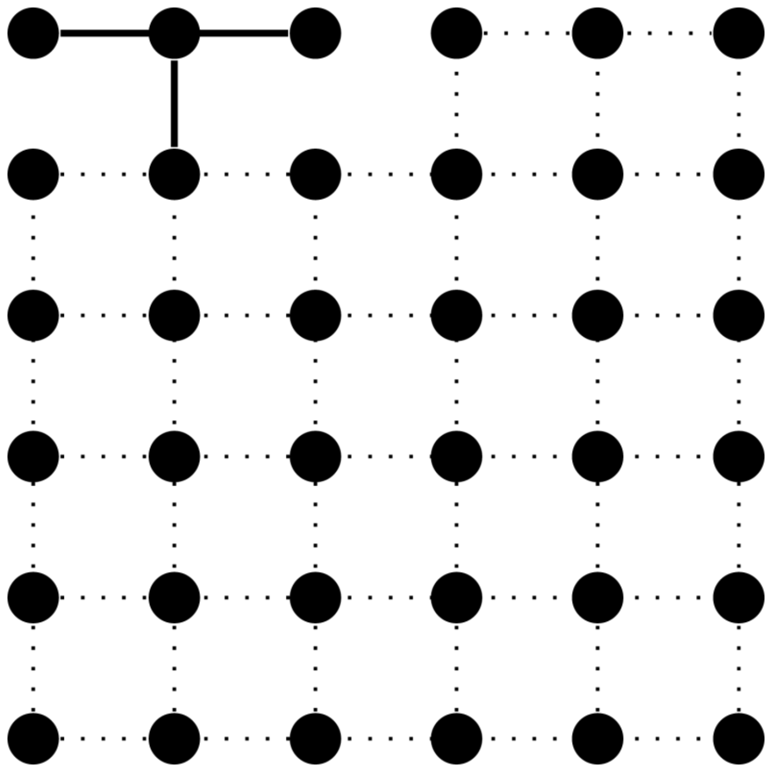
Labyrinth als Gittergraph modellieren

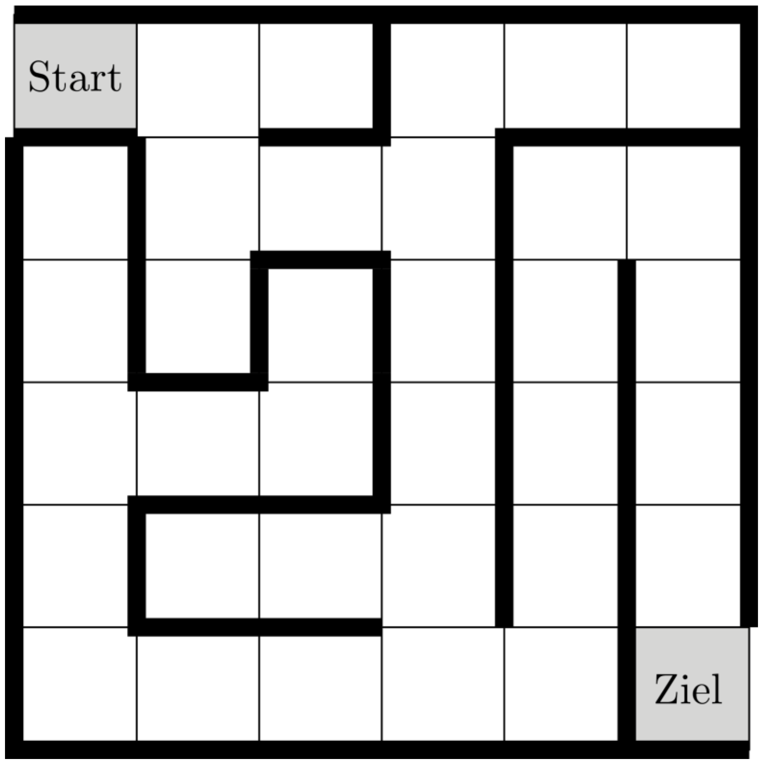




Aufgabe 6 b)c)

Labyrinth als Gittergraph modellieren





Aufgabe 6 d)

ist ein Labyrinth schön?

Schönheitskriterien:

- Es gibt genau einen Weg vom Start zum Ziel.
- Es gibt einen Weg vom Start zu jeder anderen Zelle des Labyrinths.
- Es gibt keinen Weg, der im Kreis führt.

Aufgabe 6 d)

ist ein Labyrinth schön?

Überprüfe im Gittergraph G:

- G ist kreisfrei
- G ist zusammenhängend

Das kann Breitensuche!

Laufzeit $\mathcal{O}(n + m) = \mathcal{O}(k^2) + \mathcal{O}(k^2) = \mathcal{O}(k^2)$

Aufgabe 7 a)

G hat einen Eulerkreis $\Leftrightarrow$ alle Knoten haben geraden Grad

- $\Rightarrow$
  Laufe den Eulerkreis entlang und zähle so die Grade
  nur gerade Grade („rein & raus gehen“ $\rightarrow +2$)

- $\Leftarrow$
  Suche einen Eulerkreis, Start in v_1, gehe immer zu beliebigem Nachbarn
  keine „Sackgassen“, Weg endet in v_1: $(v_1, v_2, \ldots, v_1)$
  Solange noch Kanten übrig: Starte neue Suche und füge Kreise zusammen
  $(v_1, v_2, \ldots, v_i, v_j, \ldots, v_1) + (v_i, v_k, \ldots, v_l, v_i) =$
  $(v_1, v_2, \ldots, v_i, v_k, \ldots, v_l, v_i, v_j, \ldots, v_1)$

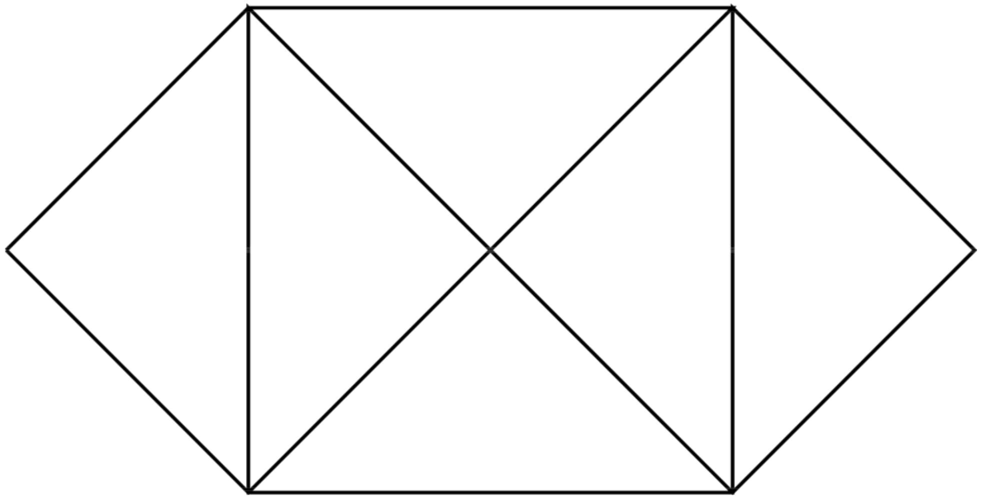
Aufgabe 7 a)

G hat einen Eulerpfad $\Leftrightarrow$ 0 oder 2 Knoten mit ungeradem Grad

- 0 Knoten mit ungeradem Grad
  G hat einen Eulerkreis
- 2 Knoten mit ungeradem Grad
  Füge einen Knoten v ein der mit beiden „ungeraden Knoten“ verbunden wird
  G' hat einen Eulerkreis

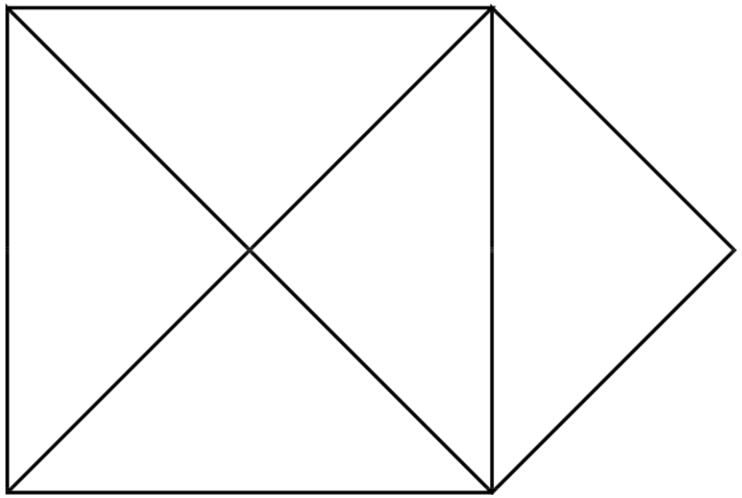
Aufgabe 7 c)

Müssen wir den Stift absetzen?



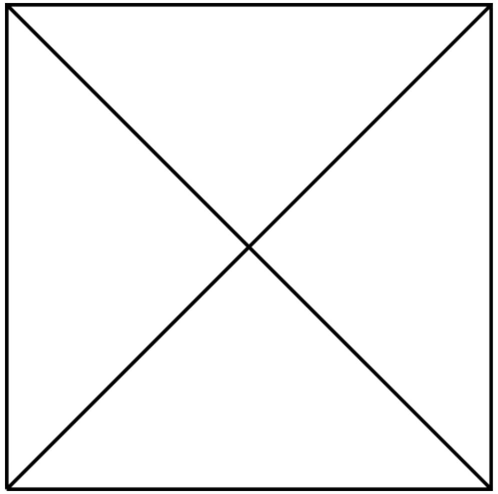
Aufgabe 7 c)

Müssen wir den Stift absetzen?



Aufgabe 7 c)

Müssen wir den Stift absetzen?



Aufgabe 7 d)

Algorithmus: Hat G einen Eulerkreis?

Zähle alle Knotengrade

Jede Adjazenzliste einmal durchgehen in $\mathcal{O}(n + m)$

Aufgabe 7 e)

Algorithmus: Finde einen Eulerkreis in G

Prüfe ob G einen Eulerkreis hat

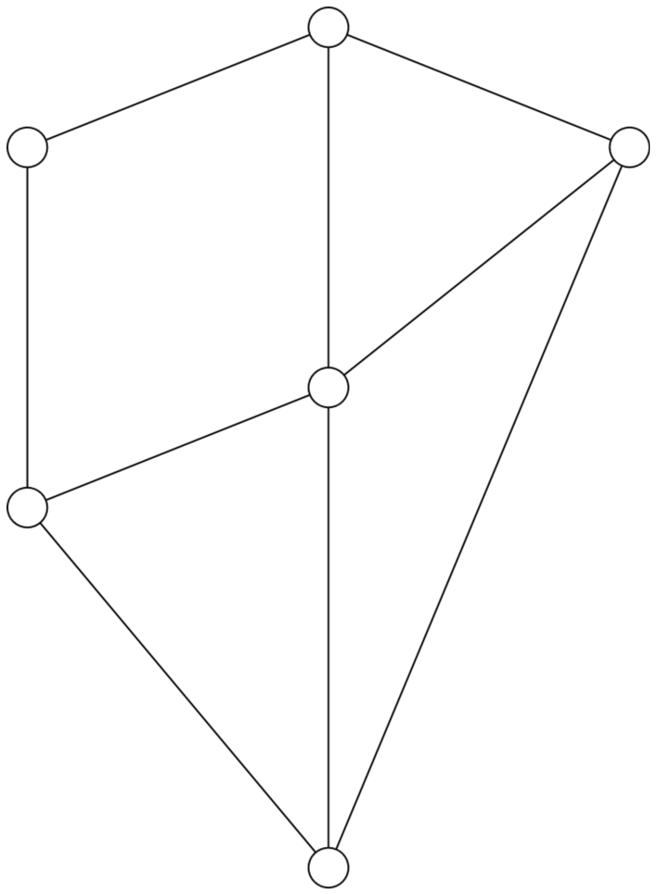
Baue wie im Beweis zu a) den Weg Schritt für Schritt auf
Lösche jeweils benutzte Kanten und verrrigngere den Grad
Pointer um Kreise in konstanter Zeit zusammenfügen zu können

Aufgabe 8 a)

Algorithmus ($\mathcal{O}(n^2)$): Durchmesser eines Baumes

Durchmesser ist die Länge des
<u>längsten kürzesten Weges</u>

Idee: Vergleiche alle kürzesten Wege



<u>kein</u> Baum!

Aufgabe 8 a)

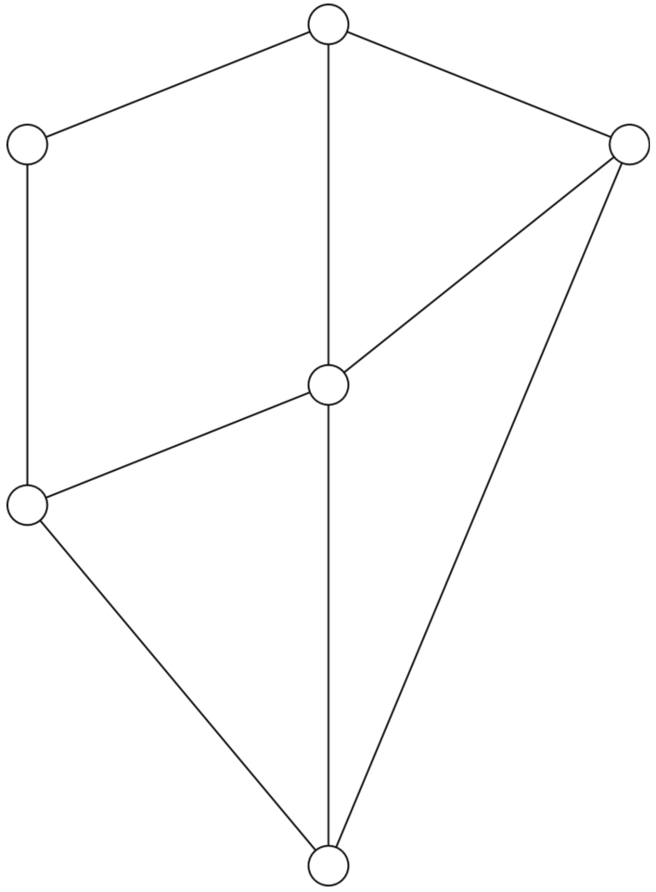
Algorithmus ($\mathcal{O}(n^2)$): Durchmesser eines Baumes

Breitensuche erzeugt Baum kürzester Wege für einen Knoten

Weglänge kann während der BFS „mitgeschrieben“ werden (zB statt Markierung)

Starte BFS in jedem Knoten, merke maximale Weglänge

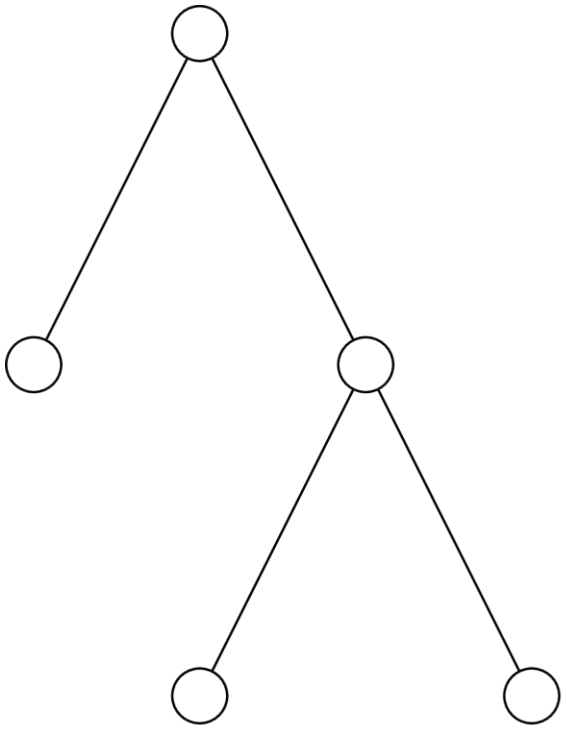
Laufzeit $n \cdot \mathcal{O}(n+m) = n \cdot \mathcal{O}(n+(n-1)) = \mathcal{O}(n^2)$



<u>kein</u> Baum!

Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes



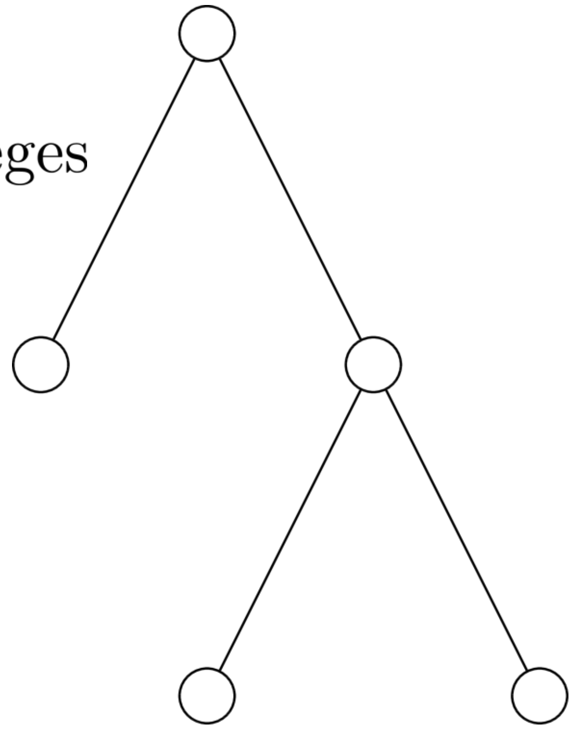
Bäume sind kreisfrei

„Abkürzung“ nur möglich wenn die kürzesten Wege zum Teil gleich sind

$\ell(v, w) = t(v) + t(w) - 2t(p_{v,w})$

Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes



tiefster Knoten ist Startpunkt eines längsten kürzesten Weges

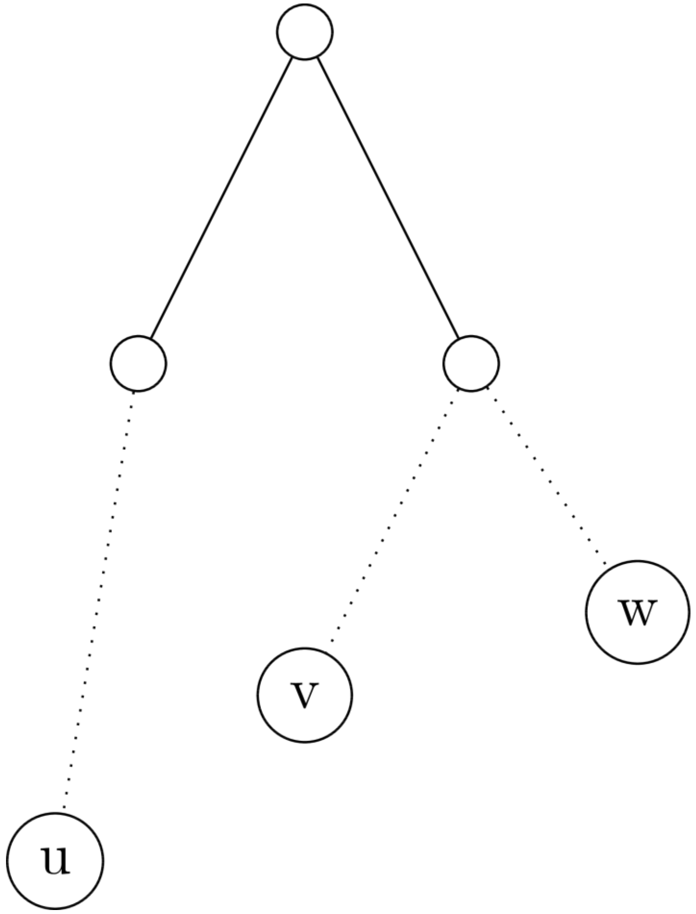
Sei u ein Knoten mit maximaler Tiefe
Finde für alle v, w einen längeren Weg der in u startet

Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes

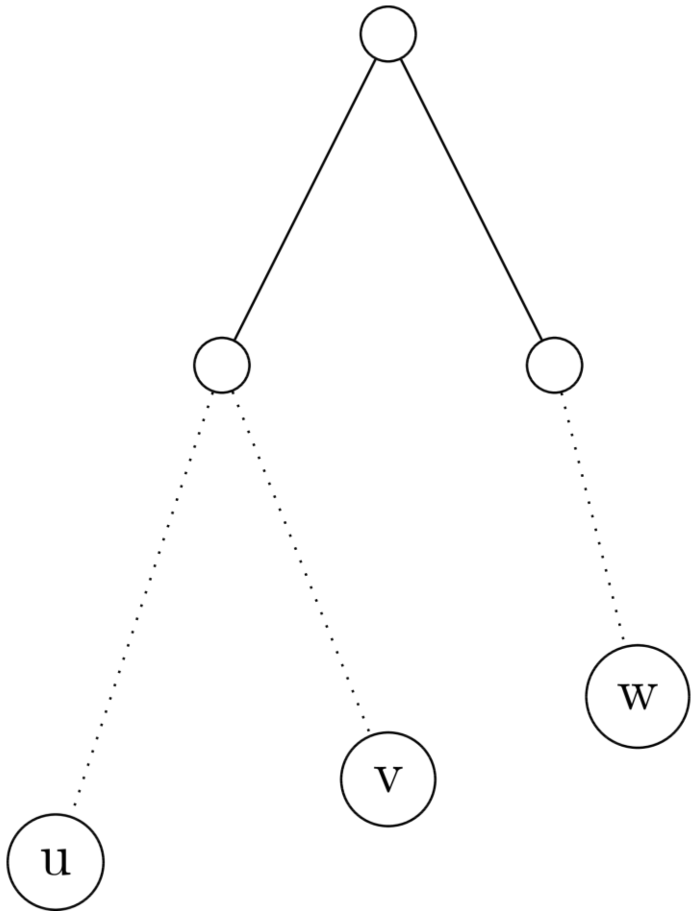
Sei $t(v) \geq t(w)$

$\ell(v, w) = t(v) + t(w) - 2t(p_{v,w})$
$\ell(u, v) = t(v) + t(u)$



Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes



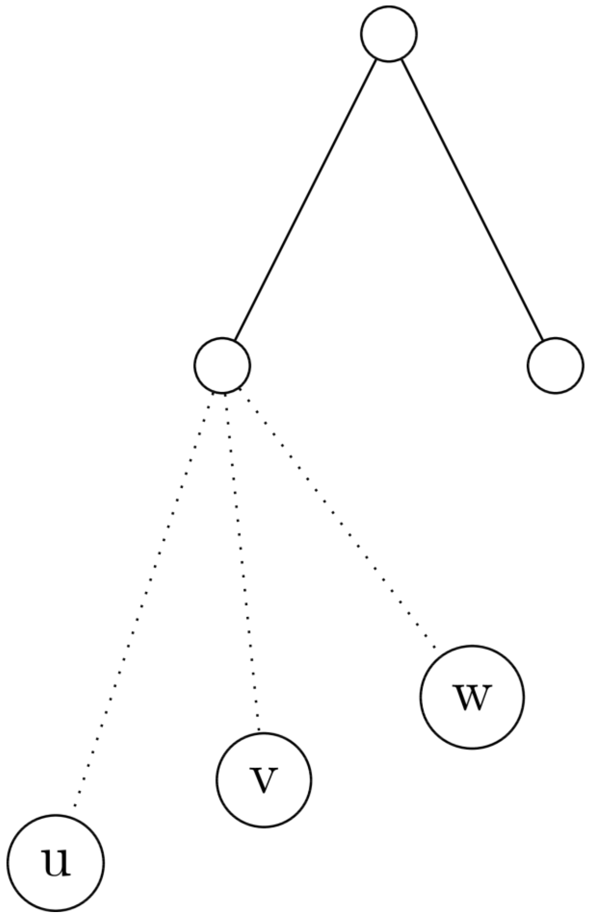
Es gilt $t(u) \geq t(v)$

$$\ell(v, w) = t(v) + t(w)$$
$$\ell(u, w) = t(u) + t(w)$$

Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes

Nur den linken Teilbaum betrachten bis einer der anderen beiden Fälle vorliegt



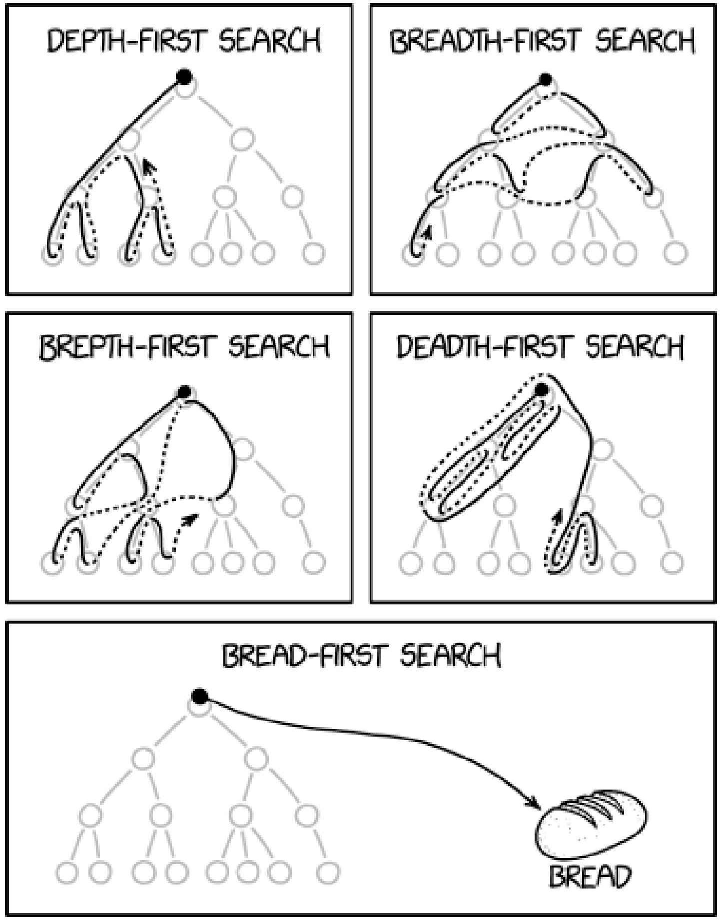
Aufgabe 8 b)

Algorithmus ($\mathcal{O}(n)$): Durchmesser eines Baumes

Es reichen also zwei Breitensuchen

eine um einen tiefsten Knoten u zu finden
eine um von u aus alle kürzesten Wege zu vergleichen

xkcd


DEPTH-FIRST SEARCH
BREADTH-FIRST SEARCH
BREPTH-FIRST SEARCH
DEADTH-FIRST SEARCH
BREAD-FIRST SEARCH
BREAD