

# Kapitel 6

## Graphen

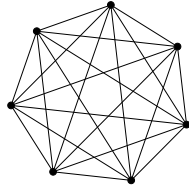
Beziehungen zwischen Objekten werden sehr oft durch binäre Relationen modelliert. Wir beschäftigen uns in diesem Kapitel mit speziellen binären Relationen, die nicht nur besonders anschaulich sind, sondern auch in zahllosen Anwendungen auftreten.

Eine symmetrische und irreflexive binäre Relation  $E \subseteq V \times V$  kann auch als Menge von zweielementigen Teilmengen  $\{u, v\} \subseteq V$  aufgefasst werden, da  $(u, v) \in E \iff (v, u) \in E$  (Symmetrie) und  $E \cap \{(v, v) : v \in V\} = \emptyset$  (Irreflexivität).

### 6.1 Definition (Graph)

*Ein Paar  $G = (V, E)$  aus Mengen  $V$  und  $E \subseteq \binom{V}{2}$  heißt (endlicher, ungerichteter) Graph. Die Elemente von  $V$  heißen Knoten, die von  $E$  Kanten. Wenn keine Missverständnisse zu befürchten sind, schreiben wir kurz  $n = n(G) = |V|$  und  $m = m(G) = |E|$  für deren Anzahl.*

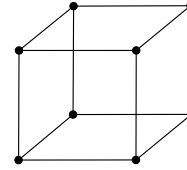
Graphen können sehr anschaulich dargestellt werden, indem man die Knoten durch Punkte und die Kanten durch Kurven, welche die Punkte ihrer beiden Knoten verbinden, repräsentiert. Die folgenden Beispiele sind Darstellungen einiger wichtiger Familien von Graphen.



vollständige Graphen  $K_n$

$$V = \{1, \dots, n\}$$

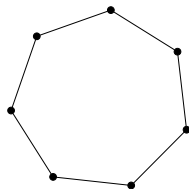
$$E = \binom{\{1, \dots, n\}}{2}$$



Hyperwürfel  $Q_d$

$$V = \{0, 1\}^d$$

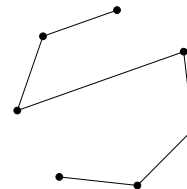
$$E = \left\{ \{(a_1, \dots, a_d), (b_1, \dots, b_d)\} : \sum_{i=1}^d |a_i - b_i| = 1 \right\}$$



Zykel oder Kreise  $C_n$

$$V = \{0, \dots, n-1\}$$

$$E = \{\{i, i+1 \pmod n\} : i = 0, \dots, n-1\}$$



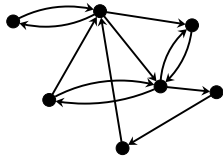
Wege  $P_n$

$$V = \{0, \dots, n\}$$

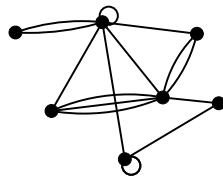
$$E = \{\{i, i+1\} : i = 0, \dots, n-1\}$$

### 6.2 Bemerkung

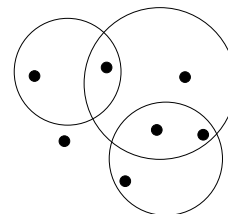
In der Graphentheorie werden auch allgemeinere binäre Relationen behandelt, in denen man auf Symmetrie ( $\rightarrow$  gerichtete Graphen), Irreflexivität ( $\rightarrow$  Graphen mit Schleifen) oder beides verzichtet. Gelegentlich werden auch Multiteilmengen (so genannte Mehrfachkanten) von  $V \times V$  ( $\rightarrow$  Multigraphen) oder Kanten mit beliebiger Knotenzahl ( $\rightarrow$  Hypergraphen) betrachtet.



gerichteter Graph



Multigraph  
(mit Schleifen  
und Mehrfachkanten)



Hypergraph  
(mit drei Hyperkanten)

Wir beschränken uns jedoch auf den Fall ungerichteter Graphen ohne Schleifen oder Mehrfachkanten; diese werden zur besseren Unterscheidung oft als schlichte Graphen bezeichnet.

### 6.3 Definition (Adjazenz, Inzidenz, Grad)

Ist  $G = (V, E)$  ein Graph, dann heißen zwei Knoten  $u, v \in V$  adjazent (auch: benachbart), falls es eine Kante  $\{u, v\} \in E$  gibt, und die Matrix  $A(G) = (a_{v,w})_{v,w \in V}$  mit

$$a_{v,w} = \begin{cases} 1 & \text{falls } \{v, w\} \in E \\ 0 & \text{sonst} \end{cases}$$

Adjazenzmatrix des Graphen. Ein Knoten  $v \in V$  und eine Kante  $e \in E$  heißen inzident, falls  $v \in e$ , und die Matrix  $I(G) = (i_{v,e})_{\substack{v \in V \\ e \in E}}$  mit

$$i_{v,e} = \begin{cases} 1 & \text{falls } v \in e \\ 0 & \text{sonst} \end{cases}$$

Inzidenzmatrix des Graphen. Die Menge  $N_G(v) = \{w \in V : \{v, w\} \in E\}$  der zu  $v \in V$  adjazenten Knoten heißt Nachbarschaft von  $v$  und wir nennen deren Kardinalität

$$d_G(v) = |N_G(v)| = \sum_{w \in V} a_{v,w} = \sum_{e \in E} i_{v,e} ,$$

d.h. die Zeilensumme von  $v$  in  $A(G)$  oder  $I(G)$ , den Grad von  $v$ . Der minimale und maximale Grad werden mit  $\delta(G) = \min_{v \in V} d_G(v)$  bzw.  $\Delta(G) = \max_{v \in V} d_G(v)$  bezeichnet.

### 6.4 Lemma (Handschlaglemma)

Für alle Graphen  $G = (V, E)$  gilt

$$\sum_{v \in V} d_G(v) = 2m .$$

*Beweis.* Prinzip des doppelten Abzählens: auf der linken Seite der Gleichung wird über alle Zeilen der Inzidenzmatrix summiert. Da jede Spalte einer Kante entspricht und jede Kante genau zwei verschiedene Knoten enthält, ergibt die Summe über alle Spalten gerade zweimal die Anzahl der Kanten.  $\square$

### 6.5 Folgerung

Die Anzahl der Knoten ungeraden Grades ist gerade.

*Beweis.* Sei  $V_i = \{v \in V : d_G(v) \equiv i \pmod{2}\}$ ,  $i = 0, 1$ , die Menge der Knoten mit geradem bzw. ungeradem Grad. Mit dem Handschlaglemma 6.4 gilt

$$2m = \sum_{v \in V} d_G(v) = \underbrace{\sum_{v \in V_0} d_G(v)}_{\text{gerade}} + \sum_{v \in V_1} d_G(v),$$

also ist auch die hinterste Summe gerade, sodass, weil jeder ihrer Summanden ungerade ist, deren Anzahl gerade sein muss.  $\square$

### 6.6 Satz

Es gibt immer zwei Knoten mit gleichem Grad.

*Beweis.* Für alle  $v \in V$  gilt  $0 \leq d_G(v) \leq n - 1$ . Gibt es einen Knoten mit Grad 0, dann gibt es keinen mit Grad  $n - 1$ , und umgekehrt (ein Knoten vom Grad 0 ist mit keinem anderen benachbart, einer vom Grad  $n - 1$  mit allen anderen Knoten; beides zusammen geht nicht). Also gilt in einem Graphen sogar  $0 \leq d_G(v) \leq n - 2$  oder  $1 \leq d_G(v) \leq n - 1$  für alle  $v \in V$ . Mit dem Tauberschlagprinzip folgt, dass von den  $n$  Knoten mindestens zwei den gleichen der  $n - 1$  möglichen Grade haben.  $\square$

### 6.7 Definition (Teilgraphen)

Sind  $G = (V, E)$  ein Graph sowie  $V' \subseteq V$  und  $E' \subseteq E$  Teilmengen seiner Knoten und Kanten, dann ist  $G' = (V', E')$  ein Teilgraph von  $G$ . falls  $E' \subseteq \binom{V'}{2}$ . Wir sagen dann  $G$  enthält  $G'$  und schreiben  $G' \subseteq G$ . Die Teilgraphen

$$G[V'] = (V', E \cap \binom{V'}{2}) \quad \text{und} \quad G[E'] = (\bigcup_{e \in E'} e, E')$$

heißen der von  $V'$  knoten- bzw. von  $E'$  kanteninduzierte Teilgraph. Ein Teilgraph heißt aufspannend, falls er alle Knoten enthält.

Wir werden außerdem Schreibweisen wie z.B.  $G - v$  für  $G[V \setminus \{v\}]$  oder  $G + e$  für den Graphen  $G' = (V, E \cup e)$  verwenden. Um die Knoten- und Kantenmenge eines Graphen  $G$  von anderen zu unterscheiden, schreiben wir auch  $V(G)$  und  $E(G)$ .

**6.8 Definition (Graphenisomorphismus)**

Gibt es zu zwei Graphen  $G_1 = (V_1, E_1)$  und  $G_2 = (V_2, E_2)$  eine bijektive Abbildung  $\alpha : V_1 \rightarrow V_2$  mit

$$\{u, v\} \in E_1 \iff \{\alpha(u), \alpha(v)\} \in E_2 ,$$

dann heißen die Graphen  $G_1$  und  $G_2$  isomorph,  $G_1 \cong G_2$ , und  $\alpha$  Graphen)isomorphismus.

Wir interessieren uns vor allem für strukturelle Eigenschaften und werden isomorphe Graphen daher nicht unterscheiden. Dadurch können wir z.B. von *dem* vollständigen Graphen mit 7 Knoten sprechen, auch wenn seine Knotenmenge nicht  $\{1, \dots, 7\}$  ist.

**6.9 Definition (Wege und Kreise in einem Graphen)**

Ein Graph  $G = (V, E)$  mit  $s, t \in V$  enthält einen  $(s, t)$ -Weg der Länge  $k$ , falls er einen zu  $P_k$  isomorphen Teilgraphen mit Endknoten  $s$  und  $t$  enthält, d.h. es gibt einen Teilgraphen  $P = (V(P), E(P)) \subseteq G$  mit  $s, t \in V(P)$  und einen Isomorphismus  $\alpha : \{0, \dots, k\} \rightarrow V(P)$  von  $P_k$  so, dass  $\alpha(0) = s$  und  $\alpha(k) = t$ .

Die Länge eines kürzesten  $(s, t)$ -Weges heißt Abstand (auch: Distanz) von  $s$  und  $t$  und wird mit  $d_G(s, t)$  bezeichnet.

Der Graph enthält einen Kreis der Länge  $k$ , falls er einen zu  $C_k$  isomomorphen Teilgraphen enthält.

**6.10 Definition (Zusammenhang)**

Ein Graph  $G = (V, E)$  heißt zusammenhängend, wenn er zu je zwei Knoten  $s, t \in V$  einen  $(s, t)$ -Weg enthält. Die inklusionsmaximalen Teilgraphen eines Graphen sind seine (Zusammenhangs)komponenten und ihre Anzahl wird mit  $\kappa(G)$  bezeichnet.

## 6.1 Bäume und Wälder

Die folgende spezielle Klasse von Graphen ist uns schon begegnet und eine zentrale Struktur der Informatik.

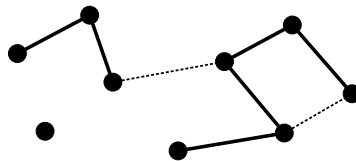
### 6.11 Definition (Baum)

Ein zusammenhängender Graph ohne Kreise heißt Baum. Ein Graph, dessen sämtliche Zusammenhangskomponenten Bäume sind, heißt Wald.

### 6.12 Satz

Für jeden Graphen  $G = (V, E)$  gilt  $m \geq n - \kappa(G)$ . Gleichheit gilt genau dann, wenn  $G$  ein Wald ist.

*Beweis.* Induktion über die Anzahl  $m$  der Kanten bei fester Anzahl  $n$  von Knoten. Enthält ein Graph keine Kante, sind keine zwei Knoten durch einen Weg verbunden, jeder bildet also seine eigene Komponente (und damit einen trivialen Baum) und es gilt  $m = 0 = n - \kappa(G)$ . Fügt man zu einem Graphen eine Kante zwischen bestehenden Knoten hinzu, verbindet sie zwei Knoten, die entweder in derselben oder in zwei verschiedenen Komponenten (die dann vereinigt werden) liegen.



Die Zahl der Komponenten verringert sich also um maximal eins, sodass die Ungleichung weiterhin gilt. Wenn durch die neue Kante zwei Komponenten vereinigt werden, entsteht kein neuer Kreis, denn sonst müsste es zwischen den beiden neu verbundenen Knoten vorher schon einen Weg gegeben haben.  $\square$

### 6.13 Satz

Für einen Graphen  $G = (V, E)$  sind folgende Aussagen äquivalent:

- (i)  $G$  ist ein Baum.
- (ii) Zwischen je zwei Knoten in  $G$  existiert genau ein Weg.

- (iii)  $G$  ist zusammenhängend und hat  $n - 1$  Kanten.
- (iv)  $G$  ist minimal zusammenhängend, d.h.  $G$  ist zusammenhängend und für alle  $e \in E$  ist  $G - e$  unzusammenhängend.
- (v)  $G$  ist maximal kreisfrei (azyklisch), d.h.  $G$  ist kreisfrei und für alle  $e \in \binom{V}{2} \setminus E$  enthält  $G + e$  einen Kreis.

*Beweis.* (i)  $\iff$  (ii): Da Bäume zusammenhängend sind, gibt es zwischen je zwei Knoten mindestens einen Weg. Gäbe es zwei verschiedene, so enthielte deren Vereinigung einen Kreis. Gibt es umgekehrt immer genau einen Weg, dann ist der Graph kreisfrei, weil es andernfalls zwei Wege zwischen je zwei Knoten des Kreises gäbe.

(i)  $\iff$  (iii): Folgt unmittelbar aus Satz 6.12.

(iii)  $\implies$  (iv): Wegen Satz 6.12 kann ein Graph mit  $n - 2$  Kanten nicht zusammenhängend sein.

(iv)  $\implies$  (v): Enthielte  $G$  einen Kreis, so könnte eine beliebige Kante des Kreises entfernt werden und  $G$  wäre immer noch zusammenhängend. Könnte man eine Kante hinzufügen, ohne einen Kreis zu erzeugen, kann es vorher keinen Weg zwischen den beiden Knoten der neuen Kante gegeben haben, der Graph wäre also nicht zusammenhängend gewesen.

(v)  $\implies$  (i): Da  $G$  kreisfrei ist, müssen wir nur zeigen, dass  $G$  auch zusammenhängend ist. Wäre  $G$  nicht zusammenhängend, so könnte zwischen zwei Knoten in verschiedenen Komponenten eine Kante eingefügt werden, ohne einen Kreis zu erzeugen.  $\square$

#### 6.14 Folgerung

*Jeder zusammenhängende Graph enthält einen aufspannenden Baum.*

Die Anzahl

$$\mu(G) = m - n + \kappa(G)$$

der Kanten, die man aus einem Graphen entfernen muss/kann, um einen aufspannenden Wald mit gleicher Anzahl von Komponenten zu erhalten, heißt auch zyklomatische Zahl des Graphen.

## 6.2 Durchläufe

In diesem Abschnitt werden wir Graphen *durchlaufen*, um Eigenschaften zu testen oder Teilgraphen zu identifizieren. Durchlaufen bedeutet dabei, an einem Knoten zu starten und jeweils von einem bereits besuchten Knoten über eine Kante den benachbarten Knoten aufzusuchen. Dazu werden zunächst die Definitionen von Wegen und Kreisen verallgemeinert.

### 6.15 Definition (Graphenhomomorphismus)

Gibt es zu zwei Graphen  $G_1 = (V_1, E_1)$  und  $G_2 = (V_2, E_2)$  eine Abbildung  $\alpha : V_1 \rightarrow V_2$  mit

$$\{u, v\} \in E_1 \implies \{\alpha(u), \alpha(v)\} \in E_2,$$

dann heißt  $\alpha$  (Graphen)homomorphismus und wir nennen den Teilgraphen

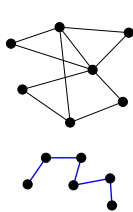
$$\alpha(G_1) = (\alpha(V_1), \{\{\alpha(u), \alpha(v)\} \in E_2 : \{u, v\} \in E_1\}) \subseteq G_2$$

homomorphes Bild von  $G_1$  in  $G_2$ .

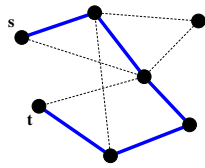
### 6.16 Definition (Kantenzug und Tour)

Ist  $G = (V, E)$  ein Graph und  $\alpha : V(P_k) \rightarrow V(G)$  ein Homomorphismus, dann heißt  $\alpha(P_k) \subseteq G$  Kantenzug der Länge  $k$ . Ein Kantenzug heißt geschlossen oder Tour, falls  $\alpha(0) = \alpha(k)$  (in diesem Fall ist  $\alpha(P_k)$  auch homomorphes Bild des Kreises  $C_k$ ).

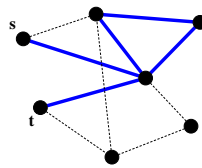
Eine Kante  $\{v, w\} \in E$  ist  $|\{\{i, i + 1\} \in E(P_k) : \{\alpha(i), \alpha(i + 1)\} = \{v, w\}\}|$  mal im Kantenzug  $\alpha(P_k)$  enthalten.



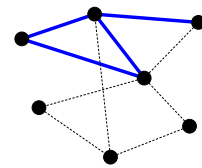
$G$  und  $P_5$



Kantenzug  
der Länge 5;  
gleichzeitig  $(s, t)$ -Weg



Kantenzug  
der Länge 5;  
aber kein Weg



geschlossener  
Kantenzug (eine Kante  
zweimal enthalten)

## 6.2.1 Eulertouren

Manche Graphenprobleme unterscheiden sich nur geringfügig in der Definition, aber erheblich im Schwierigkeitsgrad. Wir beginnen mit einem „leichten“, dass gleichzeitig das erste in der Graphentheorie betrachtete war.

Euler,  
1736

### 6.17 Definition (Eulertour)

Eine Tour heißt Eulertour, falls sie jede Kante genau einmal enthält.

### 6.18 Satz

Ein zusammenhängender Graph enthält genau dann eine Eulertour, wenn alle Knoten geraden Grad haben.

*Beweis.* Der Graph  $G = (V, E)$  enthalte eine Eulertour, d.h. es gebe eine homomorphe Abbildung von  $C_m$ , dem Kreis mit  $m$  Kanten, auf  $G$ . Da jede Kante nur einmal in der Tour vorkommt, sorgt jeder Knoten des Kreises dafür, dass zwei zum selben Knoten von  $G$  inzidente Kanten in der Tour enthalten sind. Da aber alle Kanten in der Tour vorkommen, muss die Anzahl der zu einem Knoten von  $G$  inzidente Kanten gerade sein.

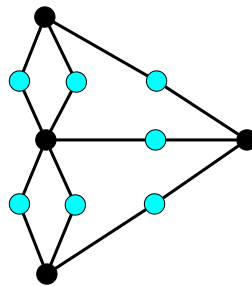
Betrachte nun umgekehrt einen zusammenhängenden Graphen, in dem jeder Knoten geraden Grad hat. Beginne bei irgendeinem Knoten und wähle eine Kante. Am benachbarten Knoten wähle eine noch nicht gewählte Kante und fahre so fort. Dies geht solange, bis man wieder den ersten Knoten erreicht, denn an jedem anderen gibt es wegen des geraden Grades immer eine noch nicht gewählte Kante. Die gewählten Kanten sind homomorphes Bild eines Kreises und an jedem Knoten ist die Anzahl der gewählten und ungewählten Kanten gerade. Falls es noch einen Knoten mit ungewählten Kanten gibt, dann wegen des Zusammenhangs auch einen, der zu schon gewählten Kanten inzident ist. Wiederhole die Konstruktion von diesem Knoten aus und füge die beiden homomorphen Bilder von Kreisen zu einem zusammen (Anfangsstück des ersten bis zum gemeinsamen Knoten, dann der zweite Kantenzug und schließlich das Endstück des ersten), bis alle Kanten einmal gewählt wurden.

□

Algorithmus  
von Hier-  
holzer

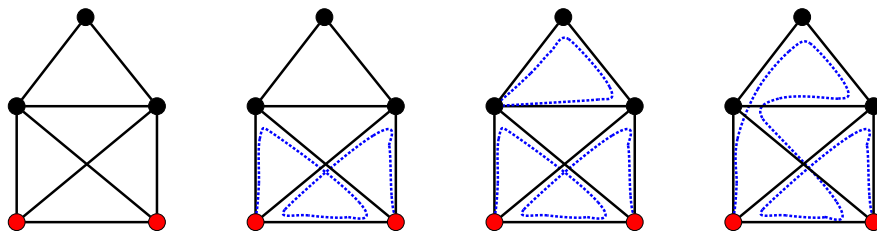
### 6.19 Beispiel (Königsberger Brückenproblem)

Der unten stehende Graph stellt sieben Brücken (helle Knoten) über den durch Königsberg fließenden Fluss Pregel dar. Da nicht alle Knoten geraden Grad haben, gibt es keine Eulertour (und damit keinen Rundgang, der jede der Brücken genau einmal überquert).



### 6.20 Beispiel (Haus vom Nikolaus)

Wenn man sich mit einem (nicht notwendig geschlossenen) Kantenzug, der alle Kanten genau einmal enthält, begnügt, dann funktioniert die Konstruktion aus dem Beweis von Satz 6.18 auch dann noch, wenn es zwei Knoten ungeraden Grades gibt. Man muss dann allerdings bei einem davon anfangen und wird beim anderen aufhören.



## 6.2.2 Tiefensuche

Der Algorithmus zur Konstruktion einer Eulertour durchläuft solange Kanten vom jeweils zuletzt besuchten Knoten, bis dieser nicht mehr zu unbesuchten Kanten inzident ist. In diesem Fall wird an irgendeinem zu unbesuchten Kanten inzidenten Knoten weitergemacht.

In der Variante in Algorithmus 24 wird immer vom zuletzt *gefundenen* (statt besuchten) Knoten aus weitergesucht. Auf diese Weise können z.B. ein aufspannender Baum, ein Kreis oder ein Weg zwischen zwei Knoten konstruiert werden.

---

**Algorithmus 24:** Tiefensuche (*depth-first search, DFS*)

---

**Eingabe** : Graph  $G = (V, E)$ , Wurzel  $s \in V$   
**Daten** : Knotenstack  $S$ , Zähler  $d$   
**Ausgabe** : Tiefensuchnummern DFS (initialisiert mit  $\infty$ )  
 Vorgänger parent (initialisiert mit nil)

```

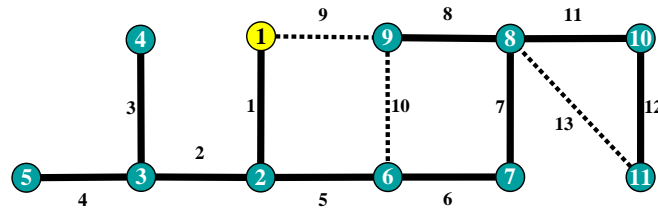
DFS[s] ← 1;  d ← 2
markiere s;  push s → S
while S nicht leer do
  v ← top(S)
  if es ex. unmarkierte Kante {v, w} ∈ E then
    markiere {v, w}
    if w nicht markiert then
      DFS[w] ← d;  d ← d + 1
      markiere w;  push w → S
      parent[w] ← v
    else pop v ← S
  
```

---

Offensichtlich erhalten alle Knoten eines Graphen, die im Verlauf des Algorithmus' markiert werden, eine endliche Tiefensuchnummer  $\text{DFS}[v]$ . Falls nach Ablauf des Algorithmus'  $\text{parent}[w] = v$  gilt, heißt die Kante  $\{v, w\}$  Baumkante, alle anderen markierten Kanten heißen Nichtbaumkanten (auch: Rückwärtskanten).

### 6.21 Beispiel

*Tiefensuche vom hellen Knoten aus: die Knoten sind mit ihren Tiefensuchnummern und die Kanten in Reihenfolge ihres Durchlaufs beschriftet.*



*Beobachtung:* Die Tiefensuche liefert einen aufspannenden Baum (Baumkanten durchgezogen). Die Knoten einer Nichtbaumkante (gestrichelt) sind durch einen monoton nummerierten Weg im Baum verbunden.

### 6.22 Lemma

Wenn  $v \in V$  auf  $S$  abgelegt wird, sei  $M \subseteq (V \setminus \{v\})$  die Menge aller anderen bereits markierten Knoten. Der Knoten  $v$  wird erst von  $S$  entfernt, nachdem alle Knoten  $t \in V \setminus M$ , für die es einen  $(v, t)$ -Weg in  $G[V \setminus M]$  gibt, ebenfalls markiert wurden.

*Beweis.* Angenommen, es gibt einen Knoten, für den die Aussage falsch ist. Wähle unter allen diesen dasjenige  $v \in V$  mit maximaler  $\text{DFS}[v]$  und einen Knoten  $t \in V$ , der bei Entfernen von  $v$  aus  $S$  noch nicht markiert ist.

Gibt es einen  $(v, t)$ -Weg über unmarkierte Knoten, dann beginnt er mit einem unmarkierten Nachbarn von  $v$ . In der **while**-Schleife werden alle Nachbarn von  $v$  markiert, bevor  $v$  aus  $S$  entfernt wird, und wegen der Maximalität von  $\text{DFS}[v]$  wird  $t$  markiert, bevor dieser Nachbar aus  $S$  entfernt wird. Das ist ein Widerspruch.  $\square$

### 6.23 Satz

Ist  $B \subseteq E$  die Menge der Baumkanten nach einer Tiefensuche auf  $G = (V, E)$  mit Wurzel  $s \in V$ , dann ist  $(G[B] + s) \subseteq G$  ein aufspannender Baum der Zusammenhangskomponente, die  $s$  enthält.

*Beweis.* Wir stellen zunächst fest, dass alle zu Baumkanten inzidenten Knoten markiert werden. Der von den Baumkanten induzierte Graph ist zusammenhängend, da jede neue Baumkante immer inzident zu einem bereits markierten Knoten ist. Er ist außerdem kreisfrei, denn jede neue Baumkante enthält genau einen Knoten, der zuvor nicht markiert war (ein Kreis kann also niemals geschlossen werden).

Weil Lemma 6.22 insbesondere für die Wurzel selbst gilt, werden alle Knoten der Zusammenhangskomponente von  $s$  markiert.  $\square$

Durch mehrfache Tiefensuche können daher alle Zusammenhangskomponenten bestimmt werden.

Der folgende Satz besagt, dass Nichtbaumkanten immer nur zwischen Knoten verlaufen, die mit der Wurzel auf einem gemeinsamen Weg liegen. Daher kann man die Tiefensuche z.B. auch benutzen, um in Komponenten, die keine Bäume sind, als *Zertifikat* einen Kreis zu konstruieren.

### 6.24 Satz

Ist  $B \subseteq E$  die Menge der Baumkanten nach einer Tiefensuche auf  $G = (V, E)$ ,  $\{v, w\} \in E$  eine Nichtbaumkante mit  $DFS[w] < DFS[v]$  und  $v = v_1, v_2, \dots, v_k = w$  der eindeutige  $(v, w)$ -Weg in  $G[B]$ , dann gilt  $parent[v_i] = v_{i+1}$  für alle  $i = 1, \dots, k - 1$ .

*Beweis.* Die Existenz der Nichtbaumkante zeigt, dass  $v$  und  $w$  in der selben Zusammenhangskomponente liegen, und  $DFS[w] < DFS[v]$  bedeutet, dass  $v$  später als  $w$  gefunden wird. Die Nichtbaumkante wird daher markiert während  $v$  (und nicht  $w$ ) oben auf dem Stack liegt (andernfalls wäre sie eine Baumkante). Das bedeutet aber, dass  $w$  zu diesem Zeitpunkt noch in  $S$  ist, weil ein Knoten erst entfernt wird, nachdem alle inzidenten Kanten markiert wurden. Die Behauptung folgt damit aus der Beobachtung, dass für einen Knoten  $u$ , der auf dem Stack unmittelbar auf einen Knoten  $u'$  gelegt wird,  $parent[u] = u'$  gilt.  $\square$

### 6.2.3 Breitensuche

Wir ändern die Durchlaufreihenfolge nun so, dass immer vom *zuerst* gefundenen Knoten aus weitergesucht wird. Dadurch kann nicht nur wieder ein aufspannender Wald konstruiert werden, sondern auch kürzeste Wege von einem Anfangsknoten zu allen anderen in seiner Komponente.

Offensichtlich erhalten alle Knoten eines Graphen, die im Verlauf des Algorithmus' markiert werden, eine endliche Breitensuchnummer  $BFS[v]$ . Falls nach Ablauf des Algorithmus'  $parent[w] = v$  gilt, heißt die Kante  $\{v, w\}$  Baumkante, alle anderen markierten Kanten heißen Nichtbaumkanten.

Man beachte, dass für Baumkanten  $BFS[w] = BFS[v] + 1$  gilt. Gilt dies für eine

**Algorithmus 25:** Breitensuche (*breadth-first search, BFS*)

**Eingabe** : Graph  $G = (V, E)$ , Wurzel  $s \in V$   
**Daten** : Knotenwarteschlange  $Q$   
**Ausgabe** : Breitensuchnummern  $BFS$  (initialisiert mit  $\infty$ )  
 Vorgänger  $parent$  (initialisiert mit  $nil$ )

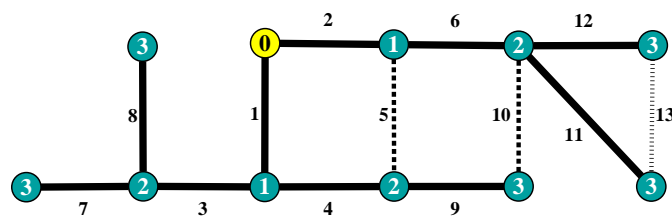
```

BFS[s] ← 0
markiere s; enqueue Q ← s
while Q nicht leer do
  dequeue v ← Q
  for unmarkierte Kanten {v, w} ∈ E do
    markiere {v, w}
    if w nicht markiert then
      BFS[w] ← BFS[v] + 1
      markiere w; enqueue Q ← w
      parent[w] ← v
  
```

Nichtbaumkante, nennen wir sie auch Vorwärtskante, andernfalls Querkante.

**6.25 Beispiel**

Breitensuche vom hellen Knoten aus: die Knoten sind mit ihren Breitensuchnummern und die Kanten in Reihenfolge ihres Durchlaufs beschriftet.



*Beobachtung:* Die Breitensuche liefert einen aufspannenden Baum (Baumkanten sind durchgezogen, Vorwärtskanten gestrichelt, Querkanten gepunktet). Die Breitensuchnummern sind gerade die Längen kürzester Wege von der Wurzel aus.

**6.26 Lemma**

Alle Knoten und Kanten der Zusammenhangskomponente der Wurzel werden markiert. Für jede markierte Kante  $\{v, w\}$  gilt  $|BFS[v] - BFS[w]| \leq 1$ .

*Beweis.* Wird ein Knoten markiert, dann auch alle seine inzidenten Kanten und alle seine Nachbarn. Eine nicht markierte Kante kann also nicht zu einem markierten Knoten inzident sein.

Angenommen, es gibt einen nicht markierten Knoten  $v$  in der Zusammenhangskomponente der Wurzel  $s$ . Nach Definition des Zusammenhangs gibt es einen  $(v, s)$ -Weg. Da  $s$  markiert wird, muss es auf diesem Weg zwei adjazente Knoten geben, von denen genau einer markiert ist. Das ist ein Widerspruch.

Die Breitensuchnummern der Knoten in der Warteschlange unterscheiden sich um höchstens 1 und die Knoten mit kleinerer Nummer werden zuerst entnommen, denn wenn ein Knoten  $w$  markiert und eingefügt wird, wurde zuvor ein Knoten  $v$  mit um 1 kleinerer Nummer entnommen. Wird eine weitere zu  $w$  inzidente Kante von einem Knoten  $v'$  aus markiert, dann ist  $w$  selbst noch in der Warteschlange (sonst wären alle inzidenten Kanten bereits markiert), sodass der Nachbar  $v'$  nach  $v$  und vor  $w$  eingefügt wurde und seine Breitensuchnummer zwischen denen von  $v$  und  $w$  liegt.  $\square$

**6.27 Satz**

Ist  $B \subseteq E$  die Menge der Baumkanten nach einer Breitensuche auf  $G = (V, E)$  mit Wurzel  $s \in V$ , dann ist  $(G[B] + s) \subseteq G$  ein aufspannender Baum der Zusammenhangskomponente, die  $s$  enthält.

*Beweis.* Ist  $n_s$  die Anzahl der Knoten in der Zusammenhangskomponente von  $s$ , dann ist  $|B| = n_s - 1$ , denn nach Lemma 6.26 werden alle  $n_s$  Knoten markiert, und eine Baumkante kommt genau dann hinzu, wenn ein anderer Knoten als  $s$  markiert wird.

Wegen Satz 6.13 brauchen wir also nur noch zu zeigen, dass  $G[B]$  für  $B \neq \emptyset$  zusammenhängend ist. Ein anderer Knoten als  $s$  wird aber nur dann markiert, wenn er einen bereits markierten Nachbarn hat.  $\square$

**6.28 Satz**

Der eindeutige Weg von der Wurzel  $s$  zu einem Knoten  $v \in V$  in  $G[B] + s$  ist ein kürzester  $(s, v)$ -Weg in  $G$  und  $BFS[v] = d_G(s, v)$ .

*Beweis.* Wenn in der Breitensuche eine Baumkante  $\{\text{parent}[w], w\}$  durchlaufen wird, erhält  $w$  die Breitensuchnummer  $BFS[\text{parent}[w]] + 1$ . Der ein-

deutige  $(s, v)$ -Weg im Baum hat daher die Länge  $\text{BFS}[v]$ .

Sind  $s = v_0, v_1, \dots, v_k = v$  die Knoten auf einem  $(s, v)$ -Weg in  $G$ , dann gilt  $|\text{BFS}[v_i] - \text{BFS}[v_{i-1}]| \leq 1$  für alle  $i = 1, \dots, k$  wegen Lemma 6.26. Weil  $\text{BFS}[s] = 0$  hat jeder solche Weg mindestens  $\text{BFS}[v]$  Kanten.  $\square$