Mathematik I für Studierende der Informatik und Wirtschaftsinformatik (Diskrete Mathematik) im Wintersemester 2017/2018

7. Dezember 2017

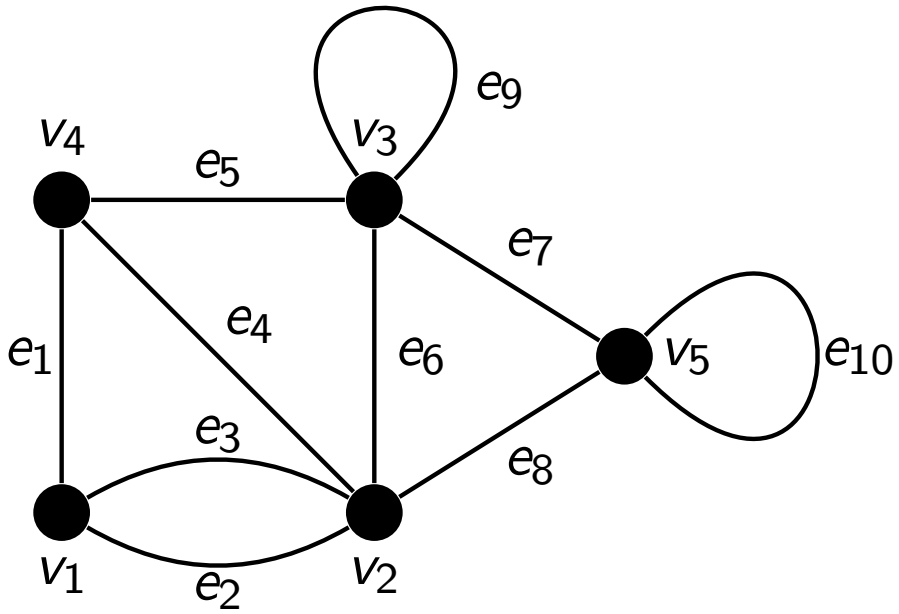
Manchmal ist es nützlich, in Graphen Mehrfachkanten und Schlingen zu erlauben.

Definition 5.28

Ein *Multigraph* ist ein Tripel (V, E, f), wobei V eine Menge von Ecken ist, E eine Menge von Kanten und f eine Abbildung, die jedem Element von E eine ein- oder zweielementige Teilmenge von V zuordnet.
Für eine Kante $e \in E$ ist $f(e)$ die Menge der *Endknoten* von e.
Die Elemente von E, denen durch f eine einelementige Teilmenge von V zugeordnet wird, heißen *Schlingen*.
Wird zwei verschiedenen Kanten e_1 und e_2 dieselbe Menge von Endknoten zugeordnet, gilt also $f(e_1) = f(e_2)$, so spricht man von einer *Mehrfachkante*.

Beispiel 5.29

Ähnlich wie Graphen lassen sich auch Multigraphen durch Punkte, die durch Linien verbunden werden, graphisch darstellen.
Der unten dargestellte Multigraph hat die Eckenmenge $V = \{v_1, \ldots, v_5\}$ und die Kantenmenge $E = \{e_1, \ldots, e_{10}\}$.
Die Funktion f bildet jede Kante auf die Menge ihrer Endpunkte ab. Zum Beispiel gilt $f(e_{10}) = \{v_5\}$ und $f(e_2) = f(e_3) = \{v_1, v_2\}$.



Eulersche Linien und Hamiltonsche Kreise

Definition 5.30

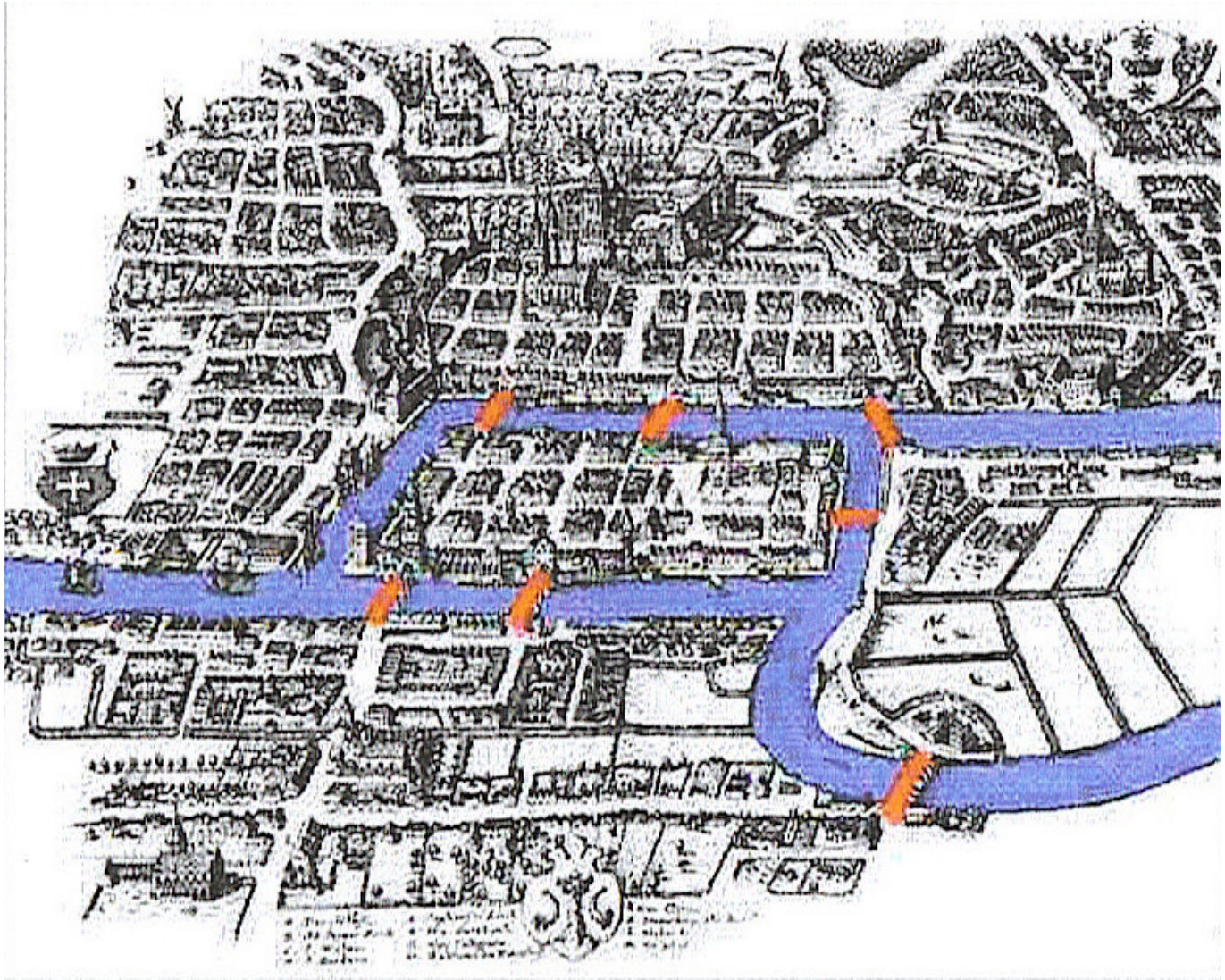
Gegeben sei ein Multigraph G mit der Knotenmenge V, der Kantenmenge E und einer Folge

$$v_0, e_1, v_1, \ldots, v_{\ell-1}, e_\ell, v_\ell$$

mit $v_i \in V$ $(i = 0, \ldots, \ell)$ und $e_i \in E$ $(i = 1, \ldots, \ell)$.

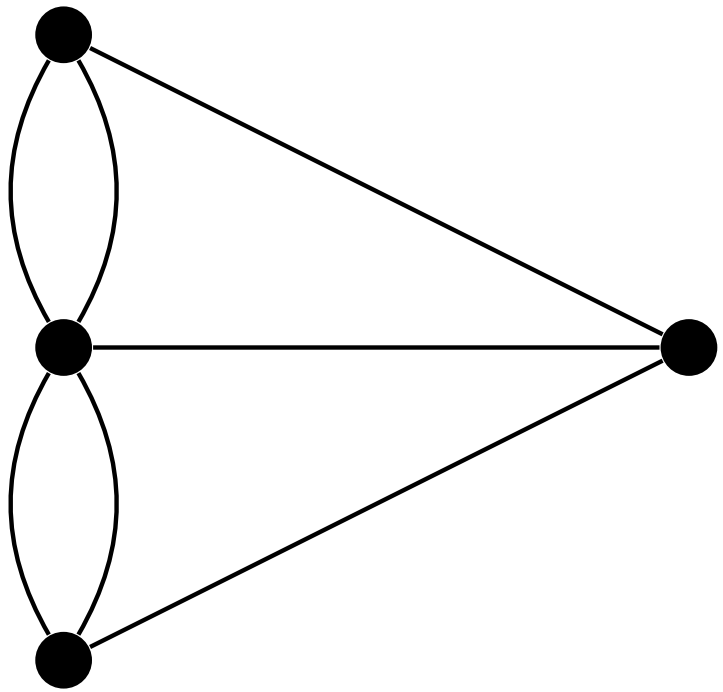
1. Die Folge heißt *Kantenfolge*, falls jedes e_i eine Kante ist, deren Endpunkte die Ecken v_{i-1} und v_i sind.
2. Ist die Folge eine Kantenfolge, in der alle Kanten verschieden sind, so spricht man von einem *Kantenzug*.
3. Ist die Folge ein Kantenzug, in dem alle Ecken verschieden sind, so handelt es sich um einen *Weg* von v_0 nach v_ℓ.
4. Die Zahl ℓ ist die *Länge* der Kantenfolge.
5. Die Kantenfolge ist *geschlossen*, falls $v_0 = v_\ell$ gilt.

Wir nennen einen Multigraphen wieder *zusammenhängend*, wenn es zwischen je zwei Ecken des Graphen einen Weg gibt, der die beiden Ecken verbindet.
Das *Königsberger Brückenproblem* ist die aus dem 18. Jahrhundert stammende Frage, ob es in der Stadt Königsberg einen Spaziergang machen kann, bei dem man jede der 7 Brücken der Stadt genau einmal überquert und am Schluss wieder auf demselben der vier Landstücke ankommt, auf dem man gestartet ist.



Eine Karte von Königsberg

Graphentheoretisch kann man dieses Problem wie folgt formulieren: Gibt es in dem folgenden Multigraphen einen geschlossenen Kantenzug, der alle Kanten durchläuft? Dabei entsprechen die Kanten den Brücken und die Ecken den Landstücken.



Der Mathematiker Leonhard Euler konnte diese Frage negativ beantworten.

Definition 5.31

Sei G ein zusammenhängender Multigraph.
Einen Kantenzug in G nennt man einen *Eulerschen Kreis*, falls er geschlossen ist und sämtliche Kanten von G durchläuft.

In Multigraphen definieren wir den Grad einer Ecke als die Anzahl der Kanten, die an der Ecke anstoßen.
Schlingen werden dabei doppelt gezählt, da sie mit zwei Enden an demselben Knoten anstoßen.

Hat ein Multigraph einen Eulerschen Kreis, so muss jeder Knoten des Graphen einen geraden Grad haben.
Das zeigt, dass der Spaziergang über die Königsberger Brücken unmöglich ist. In dem zum Brückenproblem gehörendem Multigraphen gibt es nämlich Ecken von ungeradem Grad.

Eine notwendige Bedingung für die Existenz eines Eulerschen Kreises in einem Multigraphen ist also, dass jede Ecke einen geraden Grad hat.
Im nächsten Satz stellen wir fest, dass Zusammenhang und gerade Grade sogar hinreichende Bedingungen für die Existenz eines Eulerschen Kreises sind.

Satz 5.32

Ein zusammenhängender Multigraph G besitzt genau dann einen Eulerschen Kreis, wenn alle Ecken einen geraden Grad haben.

Definition 5.33

Sei G ein Graph und C ein Kreis in G.
Dann heißt C ein *Hamiltonscher Kreis*, wenn C alle Knoten von G enthält.

Sei $c(G)$ die Anzahl der Zusammenhangskomponenten eines Graphen G.

Satz 5.34

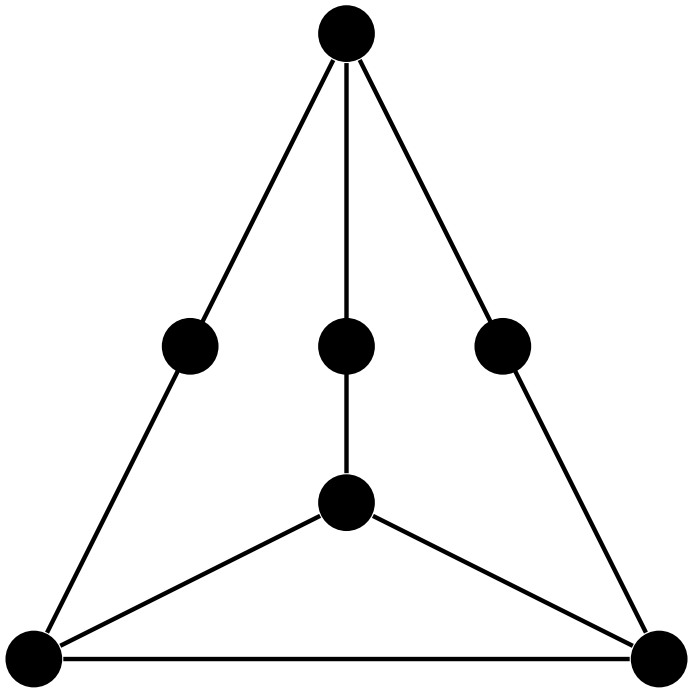
Hat ein Graph G einen Hamiltonschen Kreis, so gilt für jede nicht leere Teilmenge A von $V(G)$ die Ungleichung

$$c(G - A) \leq |A|.$$

Dabei bezeichnet $G - A$ den Graphen, den man erhält, wenn die Ecken in A und die mit diesen Ecken inzidenten Kanten aus G entfernt.

Beispiel 5.35

Der folgende Graph erfüllt die Bedingung aus Satz 5.34, hat aber keinen Hamiltonschen Kreis.



Während wir mit Satz 5.32 ein einfaches Werkzeug in der Hand haben, um zu entscheiden, ob ein gegebener Graph oder Multigraph einen Eulerschen Kreis besitzt, ist kein entsprechendes Kriterium für die Existenz eines Hamiltonschen Kreises in einem Graphen bekannt.

Es gibt auch effiziente Algorithmen, mit denen man Eulersche Kreise in Multigraphen finden kann. Zum Finden von Hamiltonschen Kreisen in beliebigen Graphen sind keine effizienten Algorithmen bekannt.

Gerichtete Graphen

Definition 5.36

Ein *gerichteter Graph* (oder *Digraph*) G ist ein Paar (V, E), wobei V eine beliebige Menge ist und E eine zweistellige Relation auf V, also $E \subseteq V^2$.
Wieder bezeichnen wir die Elemente von V als *Ecken* oder *Knoten* und die Elemente von E als (*gerichtete*) *Kanten*.
Eine Kante der Form (v, v) nennen wir *Schlinge*.
Ist G ein gerichteter Graph, so schreiben wir $V(G)$ für die Menge der Ecken von G und $E(G)$ für die Menge der Kanten.

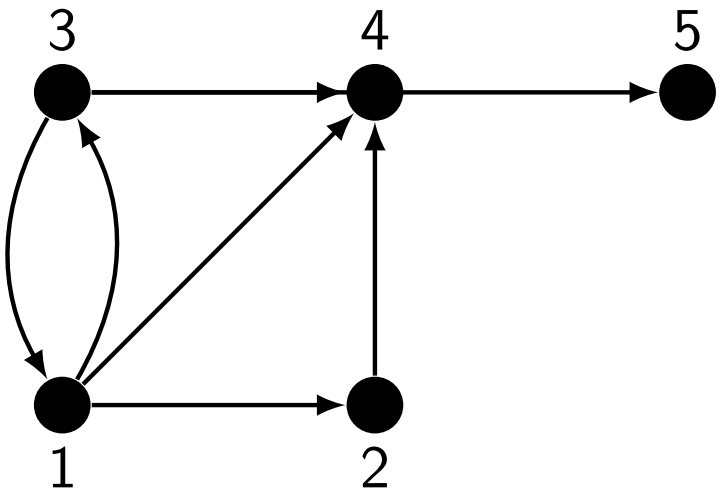
Viele Begriffe lassen sich leicht von Graphen auf gerichtete Graphen übertragen.
Zum Beispiel ist klar, was ein (gerichteter) Teilgraph eines gerichteten Graphen ist, oder wann zwei gerichtete Graphen isomorph sind.

Einen gerichteten Graphen G kann man in Form einer *Adjazenzmatrix* darstellen.
Sei $V(G) = \{v_1, \ldots, v_n\}$. Die Adjazenzmatrix von G ist dann ein quadratisches Zahlenschema mit n Zeilen und n Spalten.
Der Eintrag in der i-ten Zeile und der j-ten Spalte ist genau dann 1, wenn das Paar (v_i, v_j) in $E(G)$ ist, und sonst 0.

Man beachte, dass die Adjazenzmatrix von G von der gewählten Aufzählung $v_1, \ldots, v_n$ von $V(G)$ abhängt.

Beispiel 5.37

Sei G der folgende gerichtete Graph:



Natürlich kann man auch Adjazenzmatrizen für ungerichtete Graphen angeben, wobei jede Kante zweimal auftaucht, nämlich je einmal für jede mögliche Richtung.
Adjanzenzmatrizen ungerichteter Graphen sind symmetrisch: Spiegelung an der Diagonalen von links oben nach rechts unten führt die Matrix wieder in sich selbst über.

Man kann einen ungerichteten Graphen auch als einen gerichteten Graphen auffassen, indem man für jede ungerichtete Kante $\{v, w\}$ die beiden gerichteten Kanten (v, w) und (w, v) betrachtet.

Bemerkung 5.38

Für manche Anwendungen, insbesondere algorithmischer Art, ist es nützlich, für einen gerichteten Graphen zwei Nachbarschaftslisten zu führen: eine mit den Nachbarn, die sich von jedem Knoten aus erreichen lassen, und eine mit den Nachbarn, von denen aus man einen Knoten erreichen kann.

Definition 5.39

Ist G ein gerichteter Graph und v eine Ecke von G, so definiertman den *Außengrad* $d^+(v)$ von v als die Anzahl der Kanten, die von v wegführen, und den *Innengrad* $d^-(v)$ als die Anzahl der Kanten, die zu v hinführen.

Definition 5.40

Gegeben sei ein gerichteter Graph $G = (V, E)$ sowie eine Folge

$$v_0, e_1, v_1, \ldots, v_{\ell-1}, e_\ell, v_\ell$$

mit $v_i \in V$ für alle $i \in \{0, \ldots, \ell\}$ und $e_i \in E$ für alle $i \in \{1, \ldots, \ell\}$.

1. Diese Folge heißt *gerichtete Kantenfolge* von v_0 nach v_ℓ, falls für alle $i \in \{1, \ldots, \ell\}$ die Kante e_i eine Kante von v_{i-1} nach v_i ist.
2. Sind die Kanten in dieser Kantenfolge paarweise verschieden, so spricht man von einem *gerichteten Kantenzug*.
3. Sind außerdem die Knoten paarweise verschieden, so spricht man von einem *gerichteten Weg*.
4. Eine gerichtete Kantenfolge von v_0 nach v_ℓ heißt geschlossen, falls $v_0 = v_\ell$ gilt.

Es gibt mindestens zwei sinnvolle Arten Zusammenhang in gerichteten Graphen zu definieren.

Definition 5.41

Sei $G = (V, E)$ ein gerichteter Graph. Unter dem G *zugrunde liegendem ungerichteten Graphen* verstehen wir den Graphen G_u mit der Eckenmenge V, dessen Kantenmenge die Menge

$$E(G_u) = \{\{v, w\} : (v, w) \in E \wedge v \neq w\}$$

ist.

Definition 5.42

Sei G eingerichteter Graph.

1. G heißt *schwach zusammenhängend*, falls G_u zusammenhängend ist.
2. G heißt *stark zusammenhängend*, falls für je zwei verschiedene Ecken v und w von G ein gerichteter Weg von v nach w existiert.
3. Ein gerichteter Teilgraph $G' \subseteq G$ ist eine *schwache Zusammenhangskomponente* von G, falls G' schwach zusammenhängend ist und kein Teilgraph, der G' umfasst und echt größer ist, schwach zusammenhängend ist.
4. Ein gerichteter Teilgraph $G' \subseteq G$ ist eine *starke Zusammenhangskomponente* von G, falls G' stark zusammenhängend ist und kein Teilgraph, der G' umfasst und echt größer ist, stark zusammenhängend ist.

Bäume

Wir erinnern uns daran, dass Bäume zusammenhängende Graphen ohne Kreise sind.

Sei B ein Baum. Nach Wahl einer Wurzel w von B können wir B als gerichteten Graphen auffassen, wobei jede Kante von der Wurzel weg gerichtet ist.
Geht bei dieser Orientierung eine Kante von einem Knoten v zu einem Knoten w, so bezeichnen wir v als den *Vater* von w und w als das *Kind* von v.
Ein Knoten, der keine Kinder hat, heißt *Blatt*.
Ein Knoten, der kein Blatt ist, heißt *innerer Knoten* des Baumes.
Die *Höhe* von B ist die maximale Länge eines Weges von der Wurzel von B zu einem Blatt.

Unter dem *Grad* von B verstehen wir die maximale Zahl von Kindern eines Knotens in B.
B ist ein *binärer Baum*, falls B den Grad 2 hat.
Hat B den Grad 3, so heißt B *ternär*.
B heißt *regulär*, falls jeder innere Knoten von B dieselbe Anzahl von Kindern hat.

Ist B ein regulärer binärer Baum mit mehr als einem Knoten, so hat die Wurzel von B den Grad 2, jeder innere Knoten außer der Wurzel den Grad 3 und jedes Blatt den Grad 1.
Wir wissen bereits, dass ein Baum mit n Knoten genau $n-1$ Kanten hat und dass die Summe der Grade in einem Graphen genau die zweifache Kantenzahl ist.
Ist p die Zahl der Blätter von B, so gilt

$$2 + (n - 1 - p) \cdot 3 + p = 2(n - 1).$$

Es folgt $p = \frac{n+1}{2}$.
Die Zahl der inneren Knoten von B ist damit $n - p = \frac{n-1}{2}$.
Das zeigt den folgenden Satz:

Satz 5.43

Ein regulärer binärer Baum mit n Knoten hat $\frac{n+1}{2}$ Blätter und $\frac{n-1}{2}$ innere Knoten.

Abschließend beweisen wir noch einen Satz über die Anzahl der Knoten in einem Baum in Abhängigkeit von Höhe und Grad.

Satz 5.44

Ein Baum der Höhe h vom Grad s hat höchstens $\frac{s^{h+1}-1}{s-1}$ Knoten.

Breiten- und Tiefensuche

Wir betrachten zwei Algorithmen mit denen man in einem Graphen die Menge der Knoten berechnen lässt, die man von einem gegebenen Startknoten aus erreichen kann. Es wird also die Zusammenhangskomponente eines Knotens berechnet.

Wir stellen die Algorithmen für gerichtete Graphen vor. Im Falle von ungerichteten Graphen kann man die Algorithmen anwenden, indem man jede ungerichtete Kante $\{v, w\}$ die zwei gerichteten Kanten (v, w) und (w, v) einführt. Man beachte, dass im Falle eines gerichteten Graphen die Menge der von einem Knoten v aus mit gerichteten Wegen erreichbaren Knoten weder die starke noch die schwache Zusammenhangskomponente von v sein muss.

Tiefensuche

Sei $G = (V, E)$ ein gerichteter Graph und sei $v \in V$.

Wir konstruieren schrittweise einen gerichteten Baum B mit der Wurzel v. Dabei ist ein gerichteter Baum mit einer Wurzel v ein gerichteter Graph, dessen zugrunde liegender ungerichtete Graph ein Baum ist und bei dem alle Kanten von der Wurzel weg zeigen. Dieser gerichtete Baum B ist ein gerichteter Teilgraph von G.

Im Laufe des Algorithmus markieren wir mehr und mehr Knoten von G und versuchen unmarkierte Nachbarn eines aktuellen Knoten a zu finden.

1. Markiere den Knoten v und setze $a := v$. In diesem Schritt sei B der Baum, dessen einziger Knoten die Wurzel v ist.
2. Falls es einen unmarkierten Knoten $u \in V$ gibt, so dass $(a, u) \in E$ gilt, so wählte ein solches u, füge u und die Kante (a, u) zu dem Baum B hinzu, markiere u und setze $a := u$. Diesen Schritt bezeichnet man als den Vorwärtsschritt (advance step).
3. Falls es keinen unmarkierten Knoten $u \in V$ gibt, so dass $(a, u) \in E$ gilt, und falls a nicht die Wurzel von T ist, so geht man zurück zum Vater w von a in B und setzt $a := w$. Diesen Schritt bezeichnet man als den Rückwärtsschritt (back-tracking step). Nun fährt man mit Schritt (2) fort.
4. Falls es keinen unmarkierten Knoten $u \in V$ gibt, so dass $(a, u) \in E$ gilt, und falls a die Wurzel von T ist, so endet der Algorithmus.

Die Tiefensuche wird auf Englisch *depth first search* (*DFS*) genannt. Dementsprechend heißt der Baum, der bei der Tiefensuche gewählt wird, *DFS-Baum*.

Satz 5.45

Sei G ein gerichteter Graph und $v \in V(G)$.
Weiter sei B der Baum der markierten Knoten, der entsteht, wenn man die Tiefensuche in G ausgehend von v durchführt.
Dann ist ein Knoten $w \in V(G)$ genau dann in B, wenn es einen gerichteten Weg $v_0, v_1, \ldots, v_\ell$ von v nach w in G gibt.

Breitensuche

Sei $G = (V, E)$ ein gerichteter Graph und sei $v \in V$.

Wieder konstruieren wir einen gerichteten Baum B mit der Wurzel v.
Wenn der Algorithmus endet, so enthält B wieder alle Knoten, die von v aus erreichbar sind.
Der Unterschied zur Tiefensuche liegt darin, dass wir länger beim aktuellen Knoten bleiben und die Suche entsprechend anders organisieren.

1. Markiere den Knoten v und setze $a := v$. In diesem Schritt sei B der Baum, dessen einziger Knoten die Wurzel v ist.
2. Falls es einen unmarkierten Knoten $u \in V$ gibt, so dass $(a, u) \in E$ gilt, so wähle ein solches u, füge u und die Kante (a, u) zu dem Baum B hinzu und markiere u. Im Unterschied zur Tiefensuche bleibt in diesem Schritt der ursprüngliche Knoten a der aktuelle Knoten.
3. Falls es keinen unmarkierten Knoten $u \in V$ gibt, so dass $(a, u) \in E$ gilt, und falls es einen Knoten b in B gibt, von dem aus es eine Kante (b, u) zu einem unmarkierten Knoten u gibt, so wähle aus allen solchen Knoten b denjenigen aus, der schon am längsten in dem Baum B ist und setze $a := b$. Fahre mit Schritt (2) fort.
4. Falls es keine Kante (a, u) vom aktuellen Knoten zu einem unmarkierten Knoten gibt und auch kein Knoten b in B existiert, der zu einem unmarkierten Knoten benachbart ist, so stoppt der Algorithmus.

Die markierten Knoten verwaltet man bei der Breitensuche am besten mit Hilfe einer Warteschlange (queue).
In den Schritten (1) und (2) tut man jeweils den neuen markierten Knoten hinten in die Warteschlange.
Im Schritt (3) betrachtet man den vordersten Knoten in der Warteschlange und testet, ob dieser Knoten noch unmarkierte Nachbarn hat.
Falls nicht, so wird dieser Knoten aus der Warteschlange entfernt und der nächste Knoten in der Warteschlange getestet.

Die Breitensuche wird auf Englisch *breadth first search* (*BFS*) genannt. Dementsprechend heißt der Baum, der bei der Breitensuche gewählt wird, *BFS-Baum*.

Satz 5.46

Sei G ein gerichteter Graph und $v \in V(G)$. Weiter sei B der Baum der markierten Knoten, der entsteht, wenn man die Breitensuche in G ausgehend von v durchführt. Dann ist ein Knoten $w \in V(G)$ genau dann in B, wenn es einen gerichteten Weg $v_0, v_1, \ldots, v_\ell$ von v nach w in G gibt.

Restklassenringe und das RSA-Verschlüsselungsverfahren

Restklassenringe

Sei $m \in \mathbb{N}$.
Wir erinnern uns an die Definition der Kongruenz modulo m.
Zwei Zahlen $a, b \in \mathbb{Z}$ sind *kongruent modulo* m,

$$a \equiv b \,(\mathrm{mod}\, m),$$

falls a und b bei Division durch m denselben Rest haben.
Die Kongruenz $a \equiv b \,(\mathrm{mod}\, m)$ gilt genau dann, wenn $a - b$ durch m teilbar ist.

Die folgenden drei Eigenschaften aus Satz 3.37 zeigen, dass die Kongruenz modulo m eine Äquivalenzrelation ist:

1. $a \equiv a \pmod{m}$ (Reflexivität)
2. $a \equiv b \pmod{m} \Rightarrow b \equiv a \pmod{m}$ (Symmetrie)
3. $a \equiv b \pmod{m} \wedge b \equiv c \pmod{m} \Rightarrow a \equiv c \pmod{m}$ (Transitivität)

Die Äquivalenzklassen dieser Äquivalenzrelation haben wir *Restklassen* genannt und die Restklasse einer Zahl a mit $[a]_m$ bezeichnet.
Es ist also

$$[a]_m = \{b \in \mathbb{Z} : a \equiv b \pmod{m}\} = \{\ldots, a-m, a, a+m, a+2m, \ldots\}.$$

Es gibt genau m verschiedene Restklassen modulo m, nämlich

$$[0]_m, [1]_m, \ldots, [m-1]_m.$$

Definition 6.1

Es sei

$$\mathbb{Z}_m := \{[0]_m, [1]_m, \ldots, [m-1]_m\}$$

die Menge der Restklassen modulo m.

Für eine gegebene Restklasse K modulo m nennen wir ein Element $a \in K$ einen *Repräsentanten* oder *Vertreter* der Restklasse K.
Ist a ein Repräsentant von K, so gilt $K = [a]_m$.
Wählen wir aus jeder Restklasse genau einen Repräsentanten, so spricht man von einem *Repräsentanten-* oder *Vertretersystem*.
Das *Standardrepräsentantensystem* für die Restklassen in $\mathbb{Z}_m$ sind die Zahlen $0, 1, \ldots, m-1$.

Wir definieren Rechenoperationen $\oplus$ und $\odot$ zwischen Restklassen modulo m.

Definition 6.2

Für $a, b \in \mathbb{Z}$ sei

$$[a]_m \oplus [b]_m := [a + b]_m$$

und

$$[a]_m \odot [b]_m := [a \cdot b]_m.$$

Man beachte, dass diese Definition nur dann sinnvoll ist, wenn die Definition unabhängig von der Wahl der Repräsentanten a und b der Restklassen $[a]_m$ und $[b]_m$ ist, wenn also für alle $c, d \in \mathbb{Z}$ mit $[a]_m = [c]_m$ und $[b]_m = [d]_m$ gilt:

$$[a + b]_m = [c + d]_m \text{ und } [a \cdot b]_m = [c \cdot d]_m$$

An dieser Stelle erinnern wir uns wieder an Satz 3.37.
Es gilt:

(5) $a \equiv b \,(\mathrm{mod}\, m) \wedge c \equiv d \,(\mathrm{mod}\, m) \Rightarrow a + c \equiv b + d \,(\mathrm{mod}\, m)$

Mit anderen Worten, wenn $[a]_m = [c]_m$ und $[b]_m = [d]_m$ gilt, dann gilt auch

$$[a + c]_m = [b + d]_m.$$

Das heißt, dass unsere Definition von $[a]_m \oplus [b]_m$ tatsächlich nur von den Restklassen $[a]_m$ und $[b]_m$ abhängt, und nicht von der Wahl der Repräsentanten a und b.
Man sagt, dass $\oplus$ *wohldefiniert* ist.

Beispiel 6.3

Sei $m = 7$, $a = 5$ und $b = 8$. Dann ist

$$[a]_m \oplus [b]_m = [5]_7 \oplus [8]_7 = [5 + 8]_7 = [13]_7 = [6]_7.$$

Wählt man nun $c = -2$ und $d = 1$, so gilt $a - c = 7$ und $b - d = 7$. Es gilt also $a \equiv c \pmod{m}$ und $c \equiv d \pmod{m}$ und damit $[a]_m = [c]_m$ und $[b]_m = [d]_m$. Nun ist

$$[c]_m \oplus [d]_m = [-2]_7 \oplus [1]_7 = [-2 + 1]_7 = [-1]_7 = [6]_7.$$

Also ist $[a + b]_m = [c + d]_m$, wie erwartet.

Satz 6.4

Für alle $a, b, c \in \mathbb{Z}$ gilt:

1. *Kommutativgesetz:*
   - $[a]_m \oplus [b]_m = [b]_m \oplus [a]_m$
   - $[a]_m \odot [b]_m = [b]_m \odot [a]_m$
2. *Assoziativgesetz:*
   - $([a]_m \oplus [b]_m) \oplus [c]_m = [b]_m \oplus ([a]_m \oplus [c]_m)$
   - $([a]_m \odot [b]_m) \odot [c]_m = [b]_m \odot ([a]_m \odot [c]_m)$
3. *Existenz neutraler Elemente:*
   - $[a]_m \oplus [0]_m = [a]_m$
   - $[a]_m \odot [1]_m = [a]_m$
4. *Distributivgesetz:*
   - $[a]_m \odot ([b]_m \oplus [c]_m) = ([a]_m \odot [b]_m) \oplus ([a]_m \odot [c]_m)$
5. *Existenz additiver Inverser.*
   - $[a]_m \oplus [-a]_m = [0]_m$

Wir schreiben von nun an einfach $+$ und $\cdot$ für $\oplus$ und $\odot$ und stellen fest, dass sich nicht jede Rechenregel von $\mathbb{Z}$ auf $\mathbb{Z}_m$ überträgt.

Die Kürzungsregel, dass also für $a \neq 0$ aus $ab = ac$ immer $b = c$ folgt, gilt zum Beispiel im Allgemeinen nicht in $\mathbb{Z}_m$.

Zum Beispiel gilt $[2]_4 \cdot [1]_4 = [2]_4 = [6]_4 = [2]_4 \cdot [3]_4$ und $[2]_4 \neq [0]_4$, aber $[1]_4 \neq [3]_4$.

Dieses Beispiel hängt damit zusammen, dass $[2]_4 \cdot [2]_4 = [4]_4 = [0]_4$ gilt, dass es also in $\mathbb{Z}_4$ von 0 verschiedene Elemente gibt, deren Produkt 0 ist.