Algebraische Graphentheorie und Routenplanung im Schulunterricht

Diplomarbeit
im Lehramtsstudium Mathematik - Physik
zur Erlangung des akademischen Grades
Magistra der Naturwissenschaften

eingereicht an der
Fakultät für Mathematik, Informatik und Physik
der Universität Innsbruck

von

Theresa Anna Müssigang

Betreuer: Univ.-Prof. Dr. Tim Netzer

Innsbruck, Jänner 2019

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt durch meine eigenhändige Unterschrift, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die vorliegende Arbeit wurde bisher in gleicher oder ähnlicher Form noch nicht als Magister-/Master-/Diplomarbeit/Dissertation eingereicht.

Unterschrift: ....

Innsbruck, am ....

Danksagung

An dieser Stelle möchte ich mich bei all jenen bedanken, die mich beim Schreiben meiner Arbeit unterstützt und motiviert haben.

Zuerst möchte ich mich bei meinem Betreuer, Herrn Univ.- Prof. Dr. Tim Netzer, für das Bereitstellen dieses interessanten Themas und der Begutachtung meiner Diplomarbeit bedanken. Vielen Dank für all die hilfreichen und raschen Antworten auf meine Anliegen und die konstruktive Kritik.

Mein besonderer Dank gilt meiner Familie, die mir mein Studium ermöglicht und mich in all den Jahren unterstützt hat.

Inhaltsverzeichnis

Einleitung

Die hier vorliegende Diplomarbeit, welche das Thema algebraische Graphentheorie behandelt, ist in zwei große Teilbereiche gegliedert. Durch einen kurzen geschichtlichen Exkurs wird in den ersten Teil der Arbeit eingeleitet und das Thema historisch eingeordnet. Das besondere liegt hier daran, dass die Graphentheorie- anders als alle anderen Teilgebieten der Mathematik- nicht aufgrund eines fundamentalen Problems der Berechnung entwickelt wurde. Die Entstehungsgeschichte der Graphentheorie beginnt tatsächlich mit einem Rätsel [2].
Der Großteil dieses Abschnittes wird den Definitionen und Begriffen der Graphentheorie gewidmet, um einige ausgewählte Probleme, teils mit Hilfe von Algorithmen, lösen zu können. Dabei habe ich mich auf jene Teilgebiete der algebraischen Graphentheorie konzentriert, welche in meinen Augen schulrelevant sind und sich in der Schule gut umsetzen lassen.

Im zweiten Teil der Arbeit wird eine mögliche Umsetzung der gegenständlichen Theorie für den Schulunterricht vorgestellt. Dabei werde ich zu Beginn den Lehrplan für allgemeinbildende höhere Schulen analysieren und den Bezug zur algebraischen Graphentheorie herstellen.
Abschließend stelle ich mein Unterrichtskonzept inklusive Bedingungs-, Sach- und didaktischer Analyse vor.

Die folgende Diplomarbeit soll LehrerInnen Hilfestellungen und Ideen für die Bearbeitung dieses Themas im Unterricht liefern. Aber auch SchülerInnen sollen von dieser Arbeit profitieren, wenn sie diese als Vorbereitung auf eine mögliche vorwissenschaftliche Arbeit verwenden.

1 Mathematische Grundlagen

1.1 Historische Entwicklung

Begründet wurde die Graphentheorie, welche ein Teilgebiet der Algebra darstellt, bereits 1736 von Leonhard Euler [11]. Der Mathematiker behandelte das berühmte Königsberger Brückenproblem, welches folgende Frage behandelt: Ist es möglich, eine Route durch die Stadt Königsberg zu wählen, sodass jede der sieben vorhandenen Brücken nur einmal überquert werden muss und falls ja, gelingt es auch wieder zu seinem Ausgangspunkt zurückzugelangen?
Königsberg liegt in Ostpreußen und wird heute Kaliningrad genannt. Durch die Stadt fließt die Pregel, ein Fluss, der zwei Inseln umschließt und sich, wie in Abbildung 1.1 zu sehen ist, auf der rechten Seite gabelt.
Leonhard Euler wurde 1736 mit dieser Frage konfroniert und gilt seit seiner Lösung des Problems als Begründer der Graphentheorie [2].

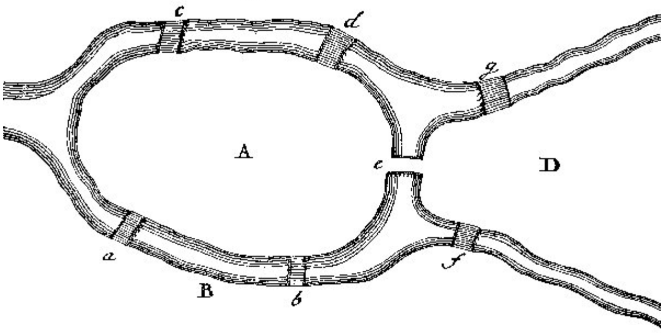


Abbildung 1.1: Königsberger Brückenproblem [2]

Erst nach etwa 100 Jahren wurde der nächste wichtige Beitrag auf diesem Gebiet geleistet [11].
Gustav Robert Kirchhoff, Physiker des 19. Jahrhunderts, beschäftigte sich mit elektrischen Netzwerken und verwendete im Zuge dessen, ohne dies explizit zu wissen, bereits 1847 die Idee eines aufspannenden Baumes [18].
Sylvester und Cayley leisteten in der Erforschung von Bäumen und deren Eigenschaften wesentliche Beiträge [11]. Deshalb dürfen diese zwei Namen in der Geschichte der Entstehung der Graphentheorie nicht fehlen.
Nach dem 2. Weltkrieg erschien weitere Literatur zur Graphentheorie. Zwei der wohl wichtigsten Werke [3] sind zum einen *Spectra of Graphs* [6] von Cvetkovic, Doob und Sachs, erschienen 1980, und zum anderen *Distance-Regular Graphs* [4] von Brouwer, Cohen und Neumaier, erschienen 1989.

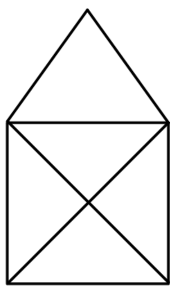
Heute hat die Graphentheorie sehr viele verschiedene Anwendungsgebiete, wie zum Beispiel in der Architektur, dem Bauwesen und Maschinenbau, der Chemie, der Elektrotechnik oder der Betriebsforschung. Um stets über die aktuellen Erkenntnisse und Fortschritte auf diesem Gebiet zu informieren, werden das *Journal of Discrete Mathematics*, das *European Journal of Combiatorics and Graphs and Combinatorics* veröffentlicht [11].

1.2 Grundlegendes Konzept und Motivation

Wie in *Optimal Analysis of Structures by Concepts of Symmetry and Regularity* von Kaveh 2013 sehr deutlich hervorgeht, hängen in vielen Systemen die Leistungs- und Funktionsfähigkeit nicht nur von den Eigenschaften der Komponenten ab, sondern auch von ihrer Position in dem System. Wird von einem Element im System eine Eigenschaft geändert, so kann sich dies auf das Gesamtverhalten auswirken. Daraus lässt sich schließen, dass die Eigenschaften der Komponenten die Leistung beeinflussen können. Allerdings kann dies auch auftreten, wenn die Position eines Elements geändert wird. Dies zeigt, dass auch die Topologie das System beeinflusst, weshalb es wichtig ist, diese klar zu repräsentieren und unmissverständlich darzustellen. Als praktisches Werkzeug hierfür hat sich das Graphenmodell bewiesen [11].

Wo begegnen wir der Graphentheorie im täglichen Leben?

Viele Situationen, welchen wir im Alltag begegnen, finden sich in der Graphentheorie wieder und können dort mathematisch beschrieben werden.
Ein sehr beliebtes Beispiel ist das Haus vom Nikolaus. Jeder hat als Kind schon einmal versucht, das Haus vom Nikolaus in einem Zug zu zeichnen ohne dabei eine Seite zweimal oder öfter zu durchlaufen [19].



Ein weiteres und wohl eines der bedeutendsten Anwendungsgebiete der Graphentheorie ist die Routenplanung. Zum einen kann hier die Frage gestellt werden, wie man die kürzeste Verbindung (zum Beispiel in einem Straßennetz) zwischen Ausgangspunkt und Endpunkt findet. Hierzu stelle man sich zum Beispiel das Bushaltestellennetz der Stadt Innsbruck vor, in dem man den kürzesten Weg vom Innsbrucker Hauptbahnhof zum Flughafen sucht.

Oder man fragt sich, welche Route ein Postbote in seinem Ortsteil zurücklegen muss, damit die Gesamtlänge minimal bleibt, er jedoch alle Häuser erreicht und trotzdem am wenigsten Benzin verbraucht.

Wie der kürzeste und der optimale Weg zu finden ist, werde ich in der folgenden Arbeit genauer betrachten und beantworten.

1.3 Grundlegende Definitionen und Begriffe der algebraischen Graphentheorie

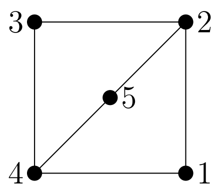
1.3.1 Grundbegriffe und zugehörige Definitionen

Definition Graph

Ein *Graph* ist ein Paar $G = (V, E)$ disjunkter Mengen, wobei die Elemente von E 2-elementige Teilmengen von V sind. Die Menge V (V von engl. vertex) ist die Menge der Ecken (oder Knoten) und ist nicht leer, ihre Elemente v sind die Ecken (oder Knoten) des Graphen G. Die Menge E (E von engl. edge) ist die Menge der Kanten (oder Linien), ihre Elemente e sind die Kanten (oder Linien) des Graphen G [8].

Bildlich kann G dargestellt werden, indem man die Ecken als Punkte zeichnet und diese durch Linien im euklidischen Raum verbindet [11].

Beispiel: $G = (V, E)$ mit $V = \{1, 2, 3, 4, 5\}$ und
$E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}, \{4, 5\}, \{5, 2\}\}$



In der Literatur ist auch häufig $V(G)$ für die Menge der Ecken und $E(G)$ für die Menge der Kanten zu finden. Diese Notation wird verwendet, falls es sich um einen bekannten Graphen handelt.

Ein Graph G heißt *endlich* bzw. *unendlich*, falls die Menge der Ecken $V(G)$ endlich bzw. unendlich ist [8]. In dieser Arbeit werden nur endliche Graphen von Bedeutung sein.

In der Graphentheorie kann man auch auf Graphen stoßen, welche mehrere Kanten zwischen zwei Ecken haben, diese werden dann *Mehrfachkanten* oder auch *parallele Kanten* genannt. Ist eine Ecke mit sich selbst verbunden, wird die entsprechende Kante als *Schlinge* bezeichnet [14]. In dieser Arbeit werde ich mich, falls nicht explizit anders angegeben, ausschließlich auf *einfache Graphen*, also Graphen ohne Schlingen und ohne Mehrfachkanten, beziehen.

Beispiel: Ein Graph mit Mehrfachkanten (parallele Kanten) und einer Schlinge [14]

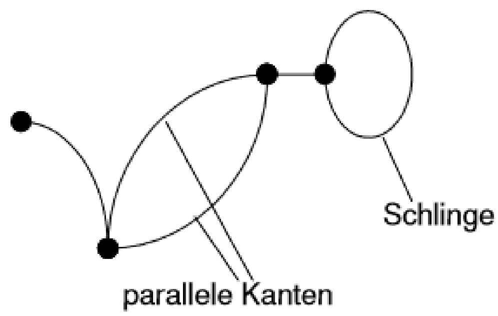


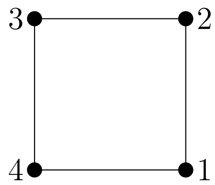
Abbildung 1.2: Graph mit Besonderheiten [14]

Aus diesem Beispiel erkennt man deutlich, dass die Kanten eines Graphen nicht zwingend gerade Linien sein müssen. Sie können genauso gut gebogen sein.

Definition Teilgraph

Es sei $G' = (V', E')$ ein weiterer Graph. Gilt $V' \subseteq V$ und $E' \subseteq E$, so ist G' ein *Teilgraph* von G und G ein *Obergraph* von G', also $G' \subseteq G$. Häufig wird umgangssprachlich einfach gesagt, dass G den Graphen G' *enthält* [8].

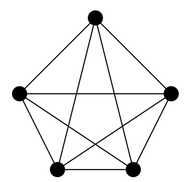
Beispiel: G der Graph wie oben und $G' = (V', E')$ mit $V' = \{1, 2, 3, 4\}$ und $E' = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}, \}$ der Teilgraph G' von G



Definition inzident & adjazent
Eine Ecke $v \in V$ und eine Kante $e \in E$ *inzidieren* miteinander, beziehungsweise eine Ecke ist mit einer Kante *inzident*, falls $v \in e$ gilt [8].
Zwei Ecken $v_1, v_2 \in V$ eines Graphen G heißen *adjazent* oder *benachbart*, falls $\{v_1, v_2\} \in E$ ist, also falls eine Kante zwischen v_1 und v_2 existiert.
Zwei nicht gleiche Kanten sind *benachbart*, falls sie eine gemeinsame Endecke haben [8].

Definition vollständig
Ein Graph heißt *vollständig*, falls je zwei Ecken von G benachbart sind [8].

Beispiel: für einen vollständigen Graphen



Definition Grad einer Ecke
Es sei $G = (V, E)$ ein Graph, welcher nicht leer ist, sprich $V \neq \emptyset$. Der *Grad* (oder auch *Valenz*) einer Ecke v, bezeichnet mit $deg(v)$, ist die Anzahl der angrenzenden Kanten [11]. Besitzt eine Ecke den Grad 0 ($deg(v) = 0$), so nennt man diese *isoliert* [8].

Satz: In jedem (einfachen) Graphen gibt es mindestens zwei Ecken, die denselben Grad besitzen [14].

Beweis:
Die Anzahl der Ecken ist $|V| = n$, so ist der maximale Eckengrad $n - 1$, denn laut Voraussetzung ist der Grad einfach und somit ohne Schlingen oder Mehrfachkanten. Und damit kann jede Ecke nur höchstens so viele Kanten haben, wie es andere Ecken gibt.
Fall 1: Eine Ecke hat den Grad $n - 1$ und ist somit mit allen anderen Ecken verbunden. Also gibt es keine isolierten Ecken, und damit hat auch keine Ecke den Grad 0. Es treten also nur maximal die Gradzahlen $1, 2, ...n-1$ auf. Das sind

höchstens $n-1$ Zahlen, welche auf n Ecken zu verteilen sind, weshalb mindestens eine Gradzahl doppelt oder mehrfach auftritt. Damit ist für diesen Fall der obige Satz bewiesen.
Fall 2: Keine Ecke hat den Grad $n-1$, womit die höchste vorkommende Gradzahl $n-2$ beträgt. Es kommen also nur Grade $0, 1, 2, ...n-2$, oder Teile davon in Frage. Auch hier haben wir wieder $n-1$ Gradzahlen für n Ecken, womit ebenso mindestens eine Gradzahl doppelt oder öfter vorkommen muss. Damit ist der Satz bewiesen und in einfachen Graphen gibt es mindestens zwei Ecken, die dieselbe Gradzahl besitzen [14]. □

Definition Weg
Für paarweise verschiedene $x_i \in V$, für $i \in \{0, ...k\}$, ist ein *Weg* ein nicht leerer Graph $P = (V, E)$, mit $V = \{x_0, x_1, ..., x_k\}$ und $E = \{\{x_0, x_1\}, ..., \{x_{k-1}, x_k\}\}$ [8].

Geht der Weg von der Ecke x_0 zur Ecke x_k, so werden diese beiden Ecken *Endecken* des Graphen P genannt, und sie sind durch den Graphen P miteinander *verbunden*.
Hingegen werden die Ecken $x_1, ..., x_{k-1}$ als *innere Ecken* bezeichnet [8].

Die *Länge* eines Weges wird durch die Anzahl der Kanten eines Weges bestimmt, und wir schreiben P^k für einen Weg der Länge k [11].

Definition Kantenzug
Salopp gesprochen kann man sich einen *Kantenzug* als eine Folge von aneinander stoßenden Kanten, die man ohne abzusetzen zeichnen kann, vorstellen [10].

Mathematisch korrekt würde man einen *Kantenzug* (der Länge k) in einem Graphen G als eine nicht leere Folge $v_0e_0v_1e_1...e_{k-1}v_k$ von abwechselnd Ecken und Kanten aus G beschreiben. Wobei $e_i = \{v_i, v_{i+1}\}$ für alle $i < k$.
Der Kantenzug wird *geschlossen* genannt, falls $v_0 = v_k$ gilt [8].
Ein Kantenzug mit nur einer Ecke wird als *trivialer Kantenzug* bezeichnet.

Falls die Ecken v_i paarweise verschieden sind, wird durch einen Kantenzug ein Weg in G definiert [8].

Der Unterschied zwischen Wegen und Kantenzügen wird nach Hußmann [10] treffend beschrieben: Wege, im Gegensatz zu Kanten können sich selbst nicht überkreuzen. Kantenzüge jedoch schon und diese können sogar Kanten mehrfach verwenden.

Definition zusammenhängend

Ein nicht leerer Graph heißt *zusammenhängend*, wenn er für je zwei seiner Ecken einen Weg enthält [8].

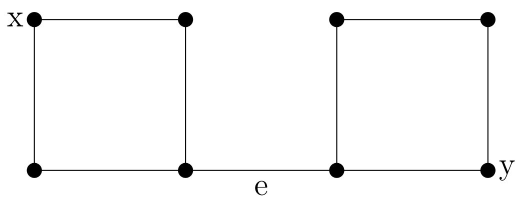
Definition Komponente

Es sei $G = (V, E)$ ein Graph. Die maximalen zusammenhängenden Teilgraphen von G sind seine *Komponenten* [8].

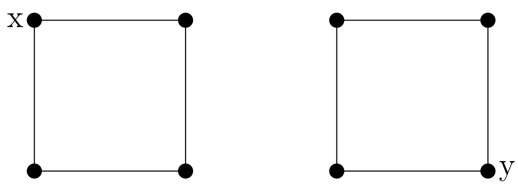
Definition Brücke

Eine *Brücke* ist eine Kante in einem zusammenhängenden Graphen $G = (V, E)$, bei deren Wegnahme der Graph G eine Komponente mehr bekommt, als er zuvor hatte [10].

Beispiel: Ein zusammenhängender Graph mit Brücke e



Beispiel: Ein nicht zusammenhängender Graph, zwischen den Ecken x und y existiert kein Weg. Der Graph zerfällt in seine Komponenten.



Definition Kreis
Ist $P = x_0...x_{k-1}$ ein Weg mit $k \geqslant 3$, so ist der Graph $C := Px_{k-1}x_0$ ein *Kreis*. Sprich, ein Kreis ist ein Weg für welchen gilt, dass $x_0 = x_{k-1}$ ist, also Anfangs- und Endknoten die selben sind und jeder Knoten nur einmal durchlaufen wird. Die Anzahl der Kanten bestimmt die Länge des Kreises [8]. Enthält der Graph keinen Kreis, so wird er *kreisfrei* genannt.

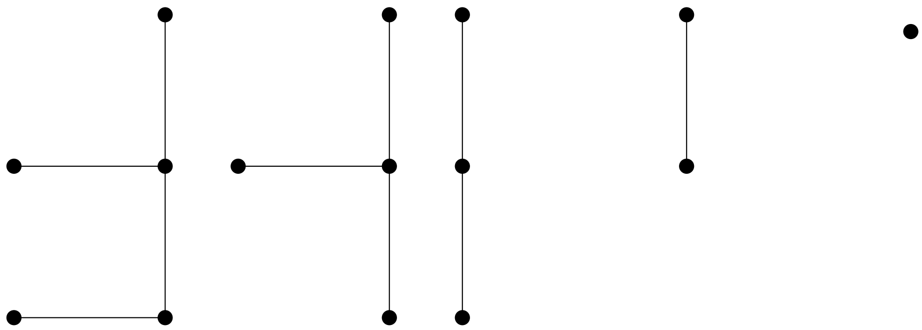
Definition Baum & Wald
Eine spezielle Art von Graphen sind die Bäume. Um dies zu definieren, müssen wir zuerst verstehen, was ein Wald ist.
Ein *Wald* ist ein Graph, der keinen Kreis enthält. Ein zusammenhängender Wald ist ein *Baum*. Also ist ein Baum ein Graph, welcher zusammenhängend und kreisfrei ist. Alle Ecken des Baumes, welche Grad 1 besitzen, werden *Blätter* genannt. Jeder Baum, welcher nicht trivial ist, besitzt ein Blatt [8].

Satz: Bei Bäumen ist die Anzahl der Knoten ($|V|$) stets um 1 größer als die Anzahl der Kanten ($|E|$) [16]:

$$|V| = |E| + 1$$

Beweis:
Wenn man sukzessive ein Blatt und die daran anhängende Kante entfernt, d.h. löscht, verringert man laufend ($|V|$) und ($|E|$) gleichmäßig. Was übrig bleibt, ist am Schluss nur noch ein Knoten [16]. □

Beispiel: Um den Beweisgedanken zu veranschaulichen



Bemerkung: Ein zusammenhängender Graph, welcher k Ecken besitzt, ist ein Baum, wenn er $k-1$ Kanten hat [8].

Definition Spannbaum

Enthält ein Baum $T \subseteq G$ alle Ecken von G, wird er als *Spannbaum* von G bezeichnet. Spannbäume kommen nur in zusammenhängenden Graphen vor [8].

1.3.2 Darstellungsarten von Graphen

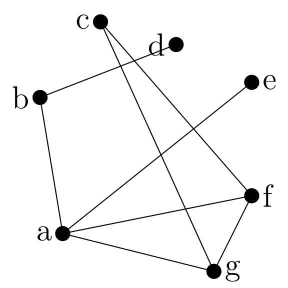
Wie bereits erwähnt, hat die Graphentheorie viele verschiedene Anwendungsgebiete und kommt auch im Alltag häufig vor. Um diese unterschiedlichsten Probleme möglichst effizient lösen zu können, werden oft Algorithmen zur Hilfe genommen. Diese Algorithmen werden normalerweise nicht von Hand, sondern von Computern durchgeführt, weshalb die Frage auftaucht: Wie kann man Graphen "computertauglich" darstellen? [10]
Die Antwort darauf liefern verschiedene Darstellungsformen, wobei ich mich auf die *Adjazenzmatrix* und *Inzidenzmatrix* beschränken werde.

Adjazenzmatrix
Wie der Name schon verrät, handelt es sich um eine $n \times n$-Matrix (n ist die Anzahl der Ecken), diese ist quadratisch und symmetrisch [10].
Die Adjazenzmatrix $A = (a_{ij})_{n \times n}$ des Graphen G ist definiert durch [8]:

$$a_{ij} := \begin{cases} 1, & \text{falls } v_i v_j \in E \\ 0 & \text{sonst.} \end{cases}$$

Dies bedeutet, dass die Einträge der Adjazenzmatrix zeigen, ob es eine Kante zwischen zwei Ecken gibt (falls der Eintrag 1 ist) oder nicht (Eintrag 0).

Beispiel: Graph $G = (V, E)$ mit $V = \{a, b, c, d, e, f, g\}$ und
$E = \{\{a, b\}, \{a, g\}, \{g, f\}, \{a, f\}, \{a, e\}, \{b, d\}, \{g, c\}, \{f, c\}\}$



Die zu dem Graphen G zugehörige Adjazenzmatrix setzt sich, wie in der Definition erläutert, wie folgt zusammen [10]:

	a	b	c	d	e	f	g
a	0	1	0	0	1	1	1
b	1	0	0	1	0	0	0
c	0	0	0	0	0	1	1
d	0	1	0	0	0	0	0
e	1	0	0	0	0	0	0
f	1	0	1	0	0	0	1
g	1	0	1	0	0	1	0

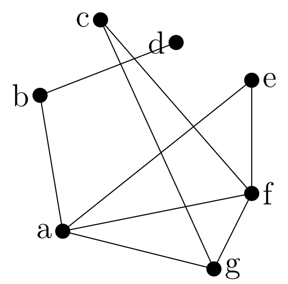
Inzidenzmatrix

Die Inzidenzmatrix $B = (b_{ij})_{n \times m}$ eines Graphen $G = (V, E)$ mit Eckenmenge $V = \{v_1, ..., v_n\}$ und Kantenmenge $E = \{e_1, ..., e_m\}$ ist definiert durch [8]:

$$b_{ij} := \begin{cases} 1, & \text{falls } v_i \in e_j \\ 0 & \text{sonst.} \end{cases}$$

Sie ist eine $n \times m$-Matrix mit n Anzahl der Ecken und m Anzahl der Kanten des Graphen G.

Beispiel: Graph $G = (V, E)$ mit $V = \{a, b, c, d, e, f, g\}$ und
$E = \{\{a, b\}, \{a, g\}, \{g, f\}, \{a, f\}, \{a, e\}, \{b, d\}, \{g, c\}, \{f, c\}, \{e, f\}\}$



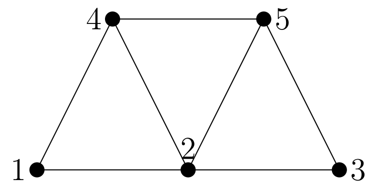
Die zu dem Graphen $G = (V, E)$ zugehörige Inzidenzmatrix, ergibt sich laut obiger Definition [10]:

	a	b	c	d	e	f	g
$\{a,b\}$	1	1	0	0	0	0	0
$\{a,e\}$	1	0	0	0	1	0	0
$\{a,f\}$	1	0	0	0	0	1	0
$\{a,g\}$	1	0	0	0	0	0	1
$\{b,d\}$	0	1	0	1	0	0	0
$\{c,g\}$	0	0	1	0	0	0	1
$\{c,f\}$	0	0	1	0	0	1	0
$\{f,g\}$	0	0	0	0	0	1	1
$\{e,f\}$	0	0	0	0	1	1	0

Die Adjazenz- und Inzidenzmatrix sind zwei sehr wichtige Matrizen, denn diese können zur Darstellung eines Graphen auf dem Computer genutzt werden. Das einzige Problem, welches hierbei auftreten kann, ist die Speicheranforderung. Die beiden Matrizen enthalten sehr viele (unnötige) Nullen und benötigen somit eine hohe Speicherkapazität,weshalb sie in der Praxis oft nicht brauchbar sind. Deshalb gibt es weitere Darstellungsweisen, welche keinen so hohen Speicherbedarf aufweisen.

Als Beispiel sei die *Adjazenzliste* genannt. Dies ist eine Liste, die aus n Zeilen und d Spalten besteht. Die i-te Zeile enthält alle Nachbarn der i-ten Ecke.
Gegenüber der eingangs erwähnten Adjazenz- und Inzidenzmatrix, ist die Speicheranforderung der Adjazenzliste deutlich geringer. Der Speicher, welchen eine Adjazenzliste benötit, beträgt $n \times d$, wogegen die Speicheranforderung der Adjazenzmatrix proportional zu $n \times n$ und die der Inzidenzmatrix proportional zu $m \times (n - 1)$ ist [11].

Beispiel: Graph $G = (V, E)$ mit $V = \{1, 2, 3, 4, 5\}$ und
$E = \{\{1, 2\}, \{2, 3\}, \{1, 4\}, \{2, 4\}, \{4, 5\}, \{2, 5\}, \{3, 5\}\}$



Die dazugehörige Adjazenzliste würde in diesem Fall folgendermaßen lauten [11]:

$$\text{AL} = \begin{bmatrix} 2 & 4 & & \\ 1 & 3 & 4 & 5 \\ 2 & 5 & & \\ 1 & 2 & 5 & \\ 2 & 3 & 4 & \end{bmatrix}$$

1.3.3 Breitensuche

Die Breitensuche, im englischen auch als Breadth-First-Search (BFS) oder Shortest-Route-Tree (SRT) bekannt, beschreibt ein Verfahren, bei dem die Knoten eines Graphen durchlaufen werden. Ziel ist es, einen Weg von einem Startknoten zu einem Endknoten beziehungsweise zu allen anderen Knoten zu finden, welcher eine minimale Anzahl von Kanten aufweist [10].

Um in einem Graphen einen kürzesten Weg zwischen zwei Ecken zu finden, benötigt ein Computer zur Ausführung des Breadth-First-Search-Algorithmus Informationen über den Graphen. Diese bekommt er von der Adjazenzmatrix, welche- bei Betrachtung einer Ecke Auskunft darüber gibt- wie viele und welche Nachbarn diese hat. Aus der Adjazenzmatrix ist also nicht ersichtlich, in welcher Richtung sich eine Endecke befindet [10].
Damit der Algorithmus dennoch zum gewünschten Ziel führt, besteht die Grundidee in der sogenannten Breitensuche. Beginnend bei einem Startknoten betrachten wir alle Knoten, welche eine Kantenlänge entfernt sind, welche zwei Kantenlängen entfernt sind und so weiter [10].

Im folgenden möchte ich eine mögliche Formulierung des Breadth- First- Search-Algorithmus angeben. In der Literatur sind unterschiedlich detaillierte Ausführungen des Alogrithmus zu finden. Der Detaillierungsgrad der Formulierung hängt oft vom Kontext und der Zielgruppe ab, weshalb ich mich für eine gut nachvollziehbare Variante entschieden habe. Hier werden die Knoten beim Besuch mit Nummern markiert, sogenannten *Breitensuchenummern*. Knoten, welche die gleiche Breitensuchenummer aufweisen, bilden ein Niveau. Der Startknoten wird immer mit $niv(e) = 0$ markiert [17].

BFS- Algorithmus : [10]

Eingabe: Graph $G = (V, E)$ mit Knotennamen und ein Startknoten s (Zielknoten nicht zwingend nötig, da der BFS kürzeste Wege zu allen anderen Knoten konstruiert)

Ausgabe: kürzeste Wege (bzgl. Kantenanzahl) vom Startknoten zu allen anderen Knoten

Schritt 1: Markiere den Startknoten mit der Nummer $niv(s) = 0$ und dieser gilt somit als besucht. Der Startknoten hat Niveau 0 und setze $i := 0$.

Schritt 2: Suche alle noch nicht besuchten und markierten Knoten, welche mit einer mit i bewerteten Ecke durch eine Kante verbunden sind. Markiere diese Knoten mit $i + 1$.

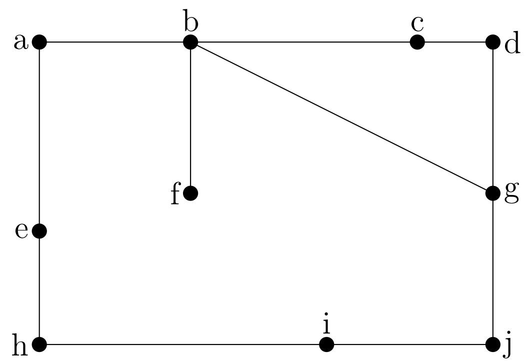
Schritt 3: Führe Schritt 2 solange aus, bis alle vom Startknoten aus erreichbaren Knoten markiert sind. Ist dies der Fall, so endet der Algorithmus.

Wir haben damit die kürzesten Wege vom Startknoten zu allen anderen Knoten gefunden und markiert. Durch die Nummerierung sind diese auch leicht zu erkennen.

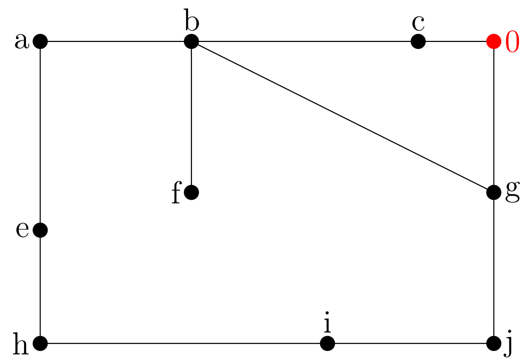
Um den Breadth- First- Search- Algoritmus zu veranschaulichen, möchte ich nun ein Beispiel ausführlich besprechen und mit Hilfe des BFS- Algorithmus den kürzesten Weg in einem Graphen finden.

Beispiel: BFS- Algorithmus

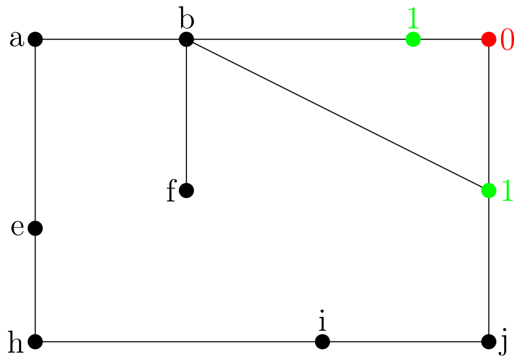
Gegeben sei der Graph mit Knotennamen: $V = \{a, b, c, d, e, f, g, h, i, j\}$ und Startknoten $s := d$



1. Als Startknoten wählen wir d; und dieser hat Niveau $niv(d) = 0$.

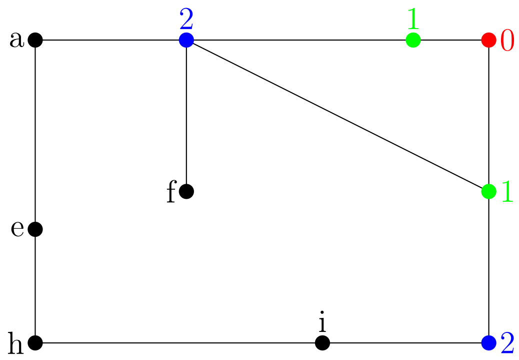


2. Von d aus, führen Kanten nach c und g, weshalb wir diese mit Breitensuchenummer 1 markieren.



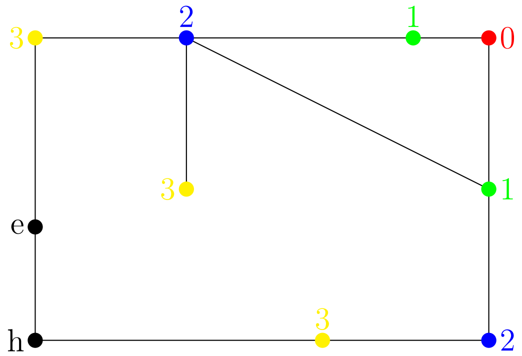
3. Da noch nicht alle vom Startknoten aus erreichbaren Knoten markiert sind, führen wir den Algorithmus fort.

4. Die Knoten c & g sind mit b durch eine Kante verbunden und g zusätzlich noch mit Knoten j, weshalb b & j mit 2 bewertet werden.



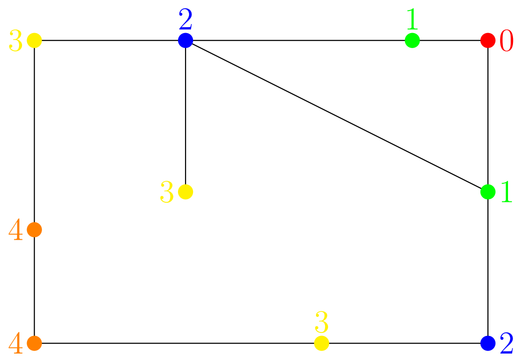
5. Da noch nicht alle vom Startknoten aus erreichbaren Knoten markiert sind, führen wir den Algorithmus fort.

6. Wir suchen alle noch nicht markierten Knoten, die mit b oder j durch eine Kante verbunden sind und markieren diese mit 3.



7. Da noch nicht alle vom Startknoten aus erreichbaren Knoten markiert sind, führen wir den Algorithmus fort.

8. Von Knoten a aus führt eine Kante nach e und von Knoten i eine nach h.



9. Alle vom Startknoten aus erreichbaren Knoten sind markiert, und der Algorithmus endet. Damit haben wir nun die kürzesten Wege vom Startknoten zu allen anderen Knoten erhalten.

Es bleibt noch zu zeigen, dass wir mit Hilfe des Breadth- First- Search- Algorithmus wirklich die kürzesten Wege von einer Startecke zu allen anderen Ecken finden. Dafür dient der nachfolgende Satz inklusive Beweis.

Satz: Es sei G ein Graph. Für jede Ecke gibt niv(e) die Anzahl der Kanten eines kürzesten Weges von der Startecke zur Ecke e an [17].

Beweis:
Diese Aussage lässt sich durch vollständige Induktion nach dem Niveau i zeigen.
Alle Knoten, die mit 1 markiert wurden, sind mit der Startecke direkt verbunden.
Es sei nun e eine Ecke mit niv(e) $= i + 1$ und e' eine Ecke aus Niveau i, welche direkt mit e verbunden ist.
Nach Induktion existiert ein Weg, welcher sich aus i Kanten von der Startecke zur Ecke e' zusammensetzt. Und damit gibt es ebenso einen Weg von der Startecke zur Ecke e, welcher aus $i + 1$ Kanten besteht.
Würde es umgekehrt einen Weg von der Startecke zur Ecke e mit weniger als $i + 1$ Kanten geben, so würde daraus folgen, dass sich e in einem Niveau k mit $k < i + 1$ befände.
Dieser Widerspruch zeigt die Behauptung [17]. □

Die folgende Abbildung 1.3 zeigt eine mögliche Art den BFS- Algorithmus zu implementieren, und zwar mit Hilfe von Python [20].

```python
import queue as q


def breitensuche(adj, start, suche):
    # adj ist die Adjazenzliste {knoten: [kanten]}
    # start ist der Index des Knoten, in dem die Suche beginnt
    # suche ist der gesuchte Knoten
    queue = q.Queue()
    queue.put(start)
    besucht = []
    while queue.qsize() > 0:
        aktiverKnoten = queue.get()
        besucht.append(aktiverKnoten)
        for andererKnoten in adj[aktiverKnoten]:
            if andererKnoten in besucht:
                continue
            if andererKnoten == suche:
                # Knoten gefunden
                return True
            queue.put(andererKnoten)
    return False
```

Abbildung 1.3: Implementierung BFS- Algorithmus [20]

Der Breadth-First-Search Algorithmus findet vor allem seine Nützlichkeit, wenn man U-Bahn-Verbindungen genauer betrachten möchte. Gibt es zum Beispiel einen Kurzstreckentarif für S- und U-Bahnen, dann beinhaltet dieser Tarif, dass man zum Beispiel nur die Hälfte des ursprünglichen Preises zahlt, wenn man höchstens 4 Stationen mit der S- oder U-Bahn fährt.

Dann kann der Breitensuche-Algorithmus zur Anwendung kommen, denn dieser kann Fragen beantworten, wie zum Beispiel: Welches ist der kürzeste Weg zwischen dem Richard-Wagner-Platz und der Stadtmitte im Berliner Liniennetzplan, welcher in Abbildung 1.4 zu sehen ist.

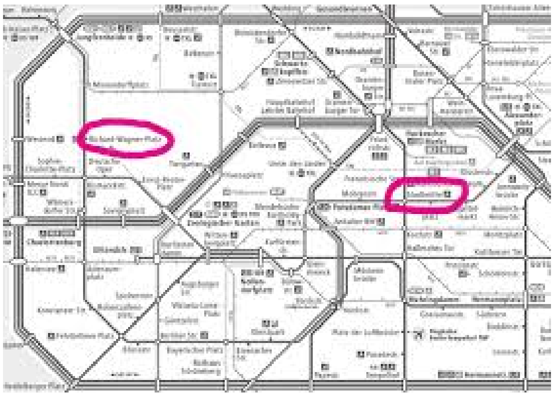


Abbildung 1.4: Ausschnitt aus dem Berliner S- und U-Bahnnetz [10]

Dies ist natürlich ein beliebiges Beispiel, kann jedoch für den Schulunterricht sehr interessant werden. Oft bekommen SchülerInnen nur ein begrenztes Taschengeld und das normale Bahn-Ticket ist für die Ferienzeit nicht gültig. So können sie ganz einfach kürzeste Wege zwischen zwei beliebigen Stationen finden und mögliche Kurzstreckentarif Angebote verwenden.

1.3.4 Der Algorithmus von Dijkstra

Bisher haben wir nur eine Strategie gefunden, mit der man Wege findet, welche eine minimale Anzahl an Kanten haben. Allerdings reicht dies nicht aus, um alle Probleme in der Routenplanung zu lösen. Möchten wir etwa wissen, welches die schnellste oder günstigste Strecke von einem Startort zu einem Zielort ist, so müssen wir noch zusätzliche Informationen über die Graphen und deren Kanten erhalten. Die Graphen, welche wir bisher betrachtet haben, reichen also nicht aus. Deshalb benötigen wir noch einige Definitionen, bevor ich schlussendlich den Dijkstra Algorithmus näher erläutern kann [10].

Definition gerichteter Graph
Bei einem *gerichteten Graphen G* oder auch kurz *Digraphen* (di von engl. directed), muss zusätzlich zur Angabe, welche Knoten durch Kanten verbunden sind, noch die Kantenrichtung angegeben werden. Dies geschieht am einfachsten durch Angabe des Anfangs- und Endknotens als Paar von geordneten Knoten[16].

Beispiel: Ein gerichteter Graph und der zugrundeliegende ungerichtete Graph

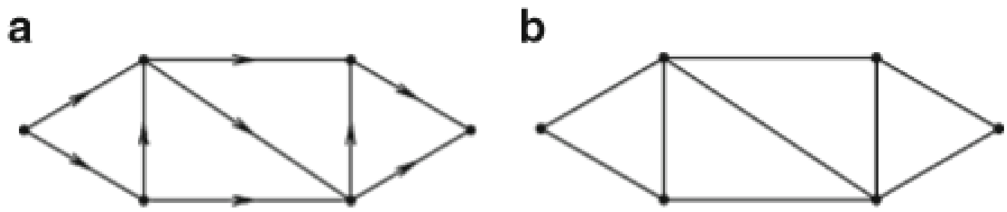


Abbildung 1.5: a. ein gerichteter Graph, b. ein ungerichteter Graph [11]

In ungerichteten Graphen haben wir bisher den Graphen $G = (V, E)$ mit seiner Knoten- und Kantenmenge meist folgendermaßen angegeben: $V = \{v_0, ..., v_k\}$ und die Elemente der Kantenmenge E beispielsweise gemäß $e_i = \{v_j, v_l\}$ mit $j, l \in \{0, ..., k\}$.
Für gerichtete Graphen werden wir die Elemente der Kantenmenge E in Zukunft wie folgt angeben: $e_i = (v_j, v_l)$. Durch die Angabe als Paar ist somit eine Richtung vorgegeben und der Start- und Endknoten eindeutig festgelegt.

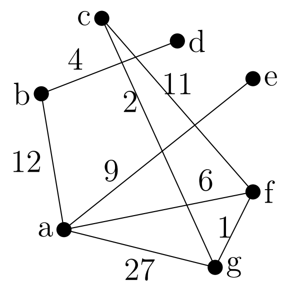
Durch die Einführung von gerichteten Graphen können wir uns nun realen Situationen wie zum Beispiel Einbahnstraßen widmen. Um aber auch andere Faktoren, wie etwa Geschwindigkeitsbegrenzungen oder Ähnliches berücksichtigen zu können, müssen wir noch eine Definition einführen.

Definition gewichteter Graph
Ein *gewichteter Graph* ist ein Graph mit einer zusätzlichen Gewichtsfunktion, die jeder Kante eine Zahl zuordnet [10]. Die Gewichtsfunktion wird oft auch Kostenfunktion genannt: $w : E \longrightarrow \mathbb{R}_+$ und die Zahl $w(e)$ für $e \in E$ als Kantengewicht, oder einfach nur Gewicht, bezeichnet.

Besitzt ein Graph keine Gewichtsfunktion, so heißt dieser ungewichteter Graph.

Die *Länge* eines Weges in einem gewichteten Graphen G ist definiert als die Summe der einzelnen Kantengewichte $w(e)$ des Weges.

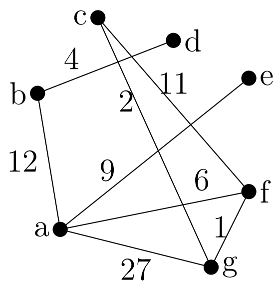
Beispiel: ein gewichteter Graph $G = (V, E)$ mit $V = \{a, b, c, d, e, f, g\}$ und $E = \{\{a, b\}, \{a, g\}, \{g, f\}, \{a, f\}, \{a, e\}, \{b, d\}, \{g, c\}, \{f, c\}\}$.
Die Kantengewichte der Kanten $e_1 = \{a, b\}$, $e_2 = \{a, g\}$, $e_3 = \{g, f\}$, $e_4 = \{a, f\}$, $e_5 = \{a, e\}$, $e_6 = \{b, d\}$, $e_7 = \{g, c\}$ und $e_8 = \{f, c\}$ lauten:
$w(e_1) = 12$, $w(e_2) = 27$, $w(e_3) = 1$, $w(e_4) = 6$, $w(e_5) = 9$, $w(e_6) = 4$, $w(e_7) = 2$, $w(e_8) = 11$



Bemerkung: In diesem soeben angeführten Beispiel führt ein Weg von a nach c über g, dieser hat dann die Länge $27 + 2 = 29$. Ein anderer Weg führt über die Ecke f und hat die Länge $6 + 11 = 17$.

Für gewichtete Graphen können wir die oben eingeführte Darstellungsart der Adjazenzmatrix einfach erweitern. Die Kantengewichte tauchen als Einträge in der Adjazenzmatrix auf.
Wird die Darstellung mittels Adjazenzliste geführt, so lässt sich die Liste der Nachbarn erweitern, indem zusätzlich noch die Gewichte der entsprechenden Kante abgespeichert werden [17].

Beispiel: Der Graph G wie im Beispiel oben und seine zugehörige Adjazenzmatrix [10]



	a	b	c	d	e	f	g
a	0	12	0	0	9	6	27
b	12	0	0	4	0	0	0
c	0	0	0	0	0	11	2
d	0	4	0	0	0	0	0
e	9	0	0	0	0	0	0
f	6	0	11	0	0	0	1
g	27	0	2	0	0	1	0

Der Algorithmus von Dijkstra:

Mit Hilfe der Dijkstra-Algorithmus können wir nun tatsächlich in gewichteten Graphen den kürzesten Weg zwischen zwei Ecken finden. Edsger Wybe Dijkstra, ein niederländischer Mathematiker und Informatiker, hat diesen Algorithmus 1960 erfunden und bewiesen, dass er in jedem Fall den kürzesten Weg zwischen zwei Ecken in gewichteten Graphen findet [14].
Die Aufgabe besteht darin, ausgehend von einer Startecke, einen Weg mit der kleinsten Gewichtung zu einer Endecke zu finden. Wir erhalten damit auch automatisch einen Weg mit minimaler Gewichtung zu allen anderen Ecken [14].
Problemstellungen dieser Art finden sich häufig im Alltag, wie zum Beispiel die Suche nach einer Strecke von einem Ort zu einem anderen, mit der minimalsten Kilometeranzahl. Oder man interessiert sich für die kürzeste Fahrzeit zwischen den beiden Orten, hier kann jedoch ein ganz anderer Weg die Lösung sein.

Wie sieht der Algorithmus nun aber im Detail aus?

Eingabe: ein gewichteter Graph mit Gewichten $w(e) \in \mathbb{R}_+$ und ein Startknoten

Ausgabe: kürzeste Wege vom Startknoten zu allen anderen Knoten

Schritt 1: Zu Beginn bekommen alle Knoten die Distanz unendlich. (Distanz ist hier immer Weglänge zum Startknoten)

Schritt 2: Gib dem Startknoten die permanente Distanz 0. Er ist nun ein aktiver Knoten.

Schritt 3: Berechne die Distanzen (temporäre Distanzen) aller noch nicht mit permanenten Distanzen versehenen Nachbarknoten der aktiven Knoten: Kantengewicht der verbindenden Kante + Distanz der aktiven Knoten

Schritt 4: Ist diese neu berechnete Distanz für einen Knoten kleiner als die bereits vorhandene, so wird die vorherige Distanz gelöscht, die kleinere Distanz notiert, und dieser aktive Knoten wird als Vorgänger dieses Knotens gespeichert. Ein vorher gespeicherter Vorgänger dieses Knotens wird gelöscht (Mit dieser Vorgängerliste lässt sich am Ende der Weg rekonstruieren). Ist die neu berechnete Distanz größer als die bereits vorhandene, ändert sich nichts an Distanz und Vorgänger.

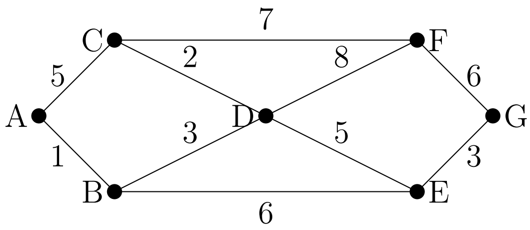
Schritt 5: Wähle einen Knoten mit minimaler temporärer Distanz. Dieser Knoten wird nun zu einem neuen aktiven Knoten. Seine Distanz wird unveränderlich festgeschrieben (permanente Distanz).

Schritt 6: Wiederhole Schritt 3 bis 6 so lange, bis es keinen Knoten mit permanenter Distanz mehr gibt, dessen Nachbarn noch temporäre Distanzen haben [10].

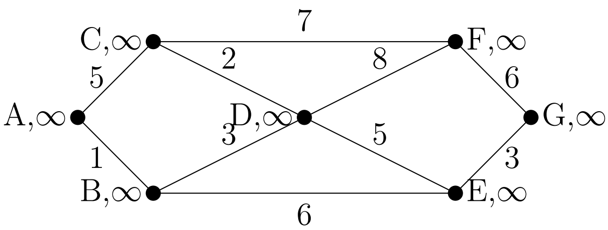
Um den Dijkstra-Algorithmus noch verständlicher zu machen, möchte ich diesen noch an einem Beispiel durchführen.

Beispiel: Dijkstra-Algorithmus

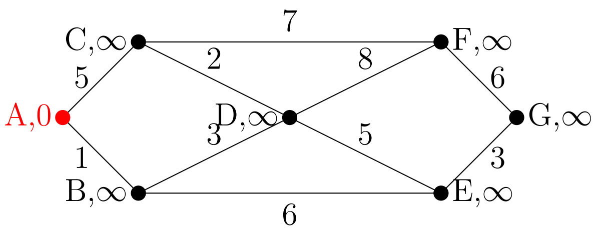
Gegeben sei der gewichtete Graph G



1. Alle Knoten bekommen die Distanz ∞



2. Wähle Startknoten A, dieser bekommt die permanente Distanz 0, er ist der aktive Knoten.

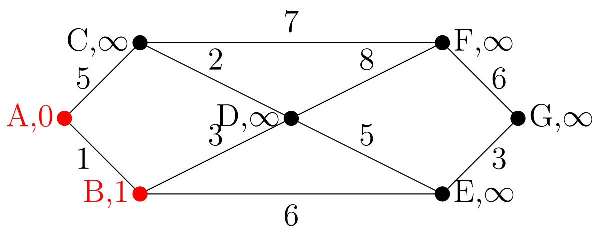


3. Berechne die Distanzen aller noch nicht mit permanenten Distanzen versehenen Nachbarknoten des aktiven Knotens A. Die Nachbarknoten sind C und B.

Weg	AB	AC
Distanz	1	5

4. Die neu berechneten Distanzen sind größer als die bereits vorhandene, also ändert sich nichts an Distanz und Vorgänger.

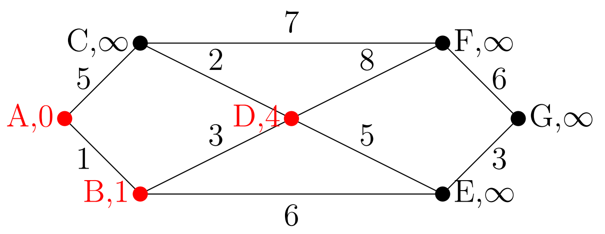
5. Wähle Knoten mit minimaler temporärer Distanz, Knoten B. Er wird nun zum aktiven Knoten, und seine Distanz wird festgelegt (permanente Distanz).



6. Berechne Distanzen zu Nachbarknoten der aktiven Knoten:

Weg	AC	ABD	ABE
Distanz	5	4	7

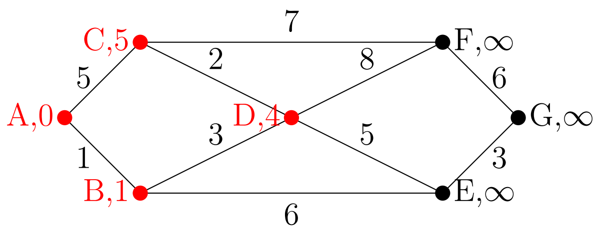
7. Der Knoten D hat die minimale temporäre Distanz, deshalb wird er zum aktiven Knoten und seine Distanz permanent.



8. Berechne die Distanzen aller noch nicht mit permanenten Distanzen versehenen Nachbarknoten der aktiven Knoten:

Weg	AC	ABDC	ABE	ABDE	ABDF
Distanz	5	6	7	9	12

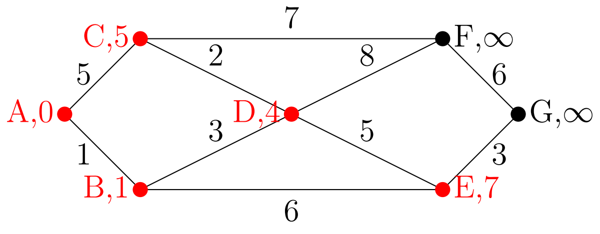
9. AC hat die minimale temporäre Distanz und seine Distanz wird nun permanent.



10. Wir berechnen wieder die Distanzen aller noch nicht mit permanenten Distanzen versehenen Nachbarknoten, dies sind die Ecken E und F.

Weg	ACF	ABDF	ABE	ABDE
Distanz	12	12	7	9

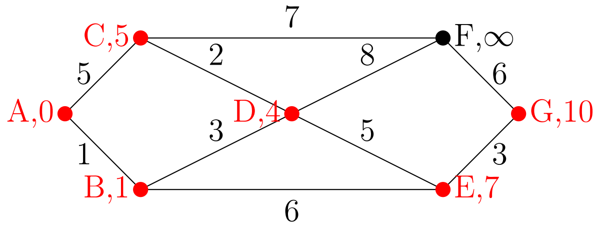
11. Die Ecke E bekommt die permanente Distanz 7.



12. Nun ergibt die Berechnung der Distanzen folgendes:

Weg	ACF	ABDF	ABEG
Distanz	12	12	10

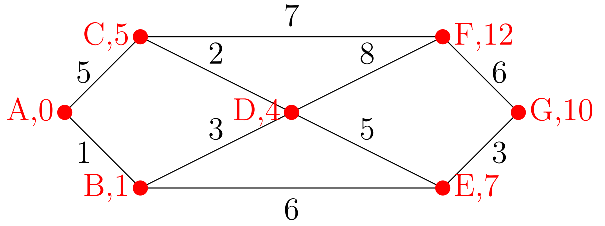
13. Die Ecke G gekommt die permanente Distanz 10.



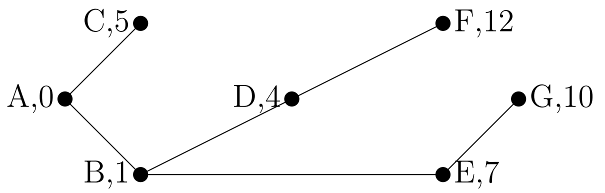
14. Die letzte Berechnung ergibt:

Weg	ACF	ABDF	ABEGF
Distanz	12	12	16

15. Da wir zwei gleiche Distanzen haben, können wir uns die Kante aussuchen.



In unserem System haben wir jede Ecke nur einmal erreicht. Dies bedeutet, dass alle Wege zusammen einen Baum bilden. In unserem Beispiel haben wir folgenden Baum erhalten:



Uns gelingt es also mit Hilfe des Dijkstra-Algorithmus, einen kürzesten Weg in einem gewichteten Graphen, ausgehend von einer Startecke zu allen anderen Ecken zu finden. Zum Beispiel führt der kürzeste Weg von A nach G über die Ecken B und E und hat die Länge 10.

Der soeben eingeführte Algoritmus von Dijkstra, sowie auch der zuvor vorgestelle Breitensuche-Algorithmus können ganz analog auf gerichtete Graphen angewandt werden [10].

1.3.5 Eulersche Graphen

Wir wollen nun eine Antwort auf das eingangs erwähnte Königsberger Brückenproblem finden und auch verstehen, wie Leonhard Euler 1736 dieses Problem löste.
Seine Lösung veröffentlichte er in einem Artikel unter dem Titel *Solutio problematis ad geometriam situs pertinentis*, woraus schon ersichtlich ist, dass nur die Art der Anordnung der Flussarme und Brücken eine Rolle beim Lösen des Problems spielt [9]. Er konzentrierte sich nur auf die wesentlichen Details, welche er von der Stadt Königsberg kannte.
Das Königsberger Brückenproblem kann durch den zugehörigen Graphen modelliert werden. Hierbei sind die Stadtteile die Ecken und die Brücken werden als Kanten repräsentiert, wie in Abbildung 1.6 zu erkennen ist.

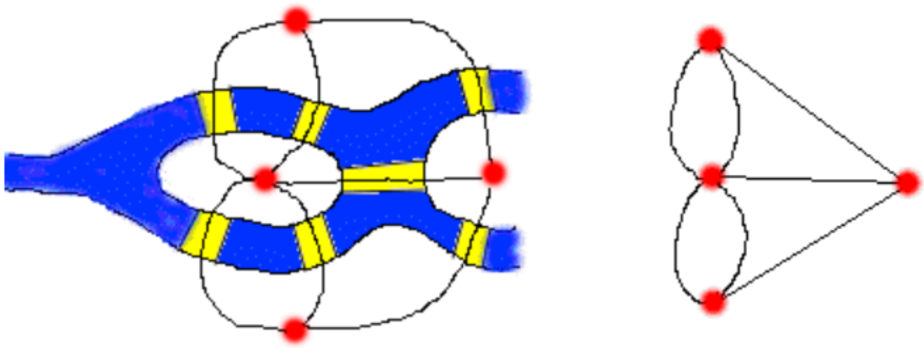


Abbildung 1.6: Köngisberger Brückenproblem als Graph modelliert [9]

Studieren wir den Graphen in Abbildung 1.6, so erkennen wir, dass die Grade der Knoten 3 beziehungsweise 5 sind. Das bedeutet, dass 3 beziehungsweise 5 Brücken aus den jeweiligen Stadtteilen führen. Diese Modellierung und die Berücksichtigung der Grade der einzelnen Knoten führten zum Erfolg von Euler. Er zeigte folgende Aussagen und löste somit das Problem [5]:

1. Gibt es mehr als zwei Stadtteile, aus denen eine ungerade Anzahl von Brücken tritt, ist das Problem unlösbar.

2. Treffen sich in genau zwei Stadtteilen eine ungerade Anzahl an Brücken, ist der Spaziergang nur dann möglich, wenn man ihn in einer der zwei Stadtteile beginnt und im anderen beendet.

3. Treffen sich in allen Stadtteilen gerade Anzahlen von Brücken, dann kann man den Spaziergang in einem beliebigen Stadtteil beginnen und auch dort wieder beenden.

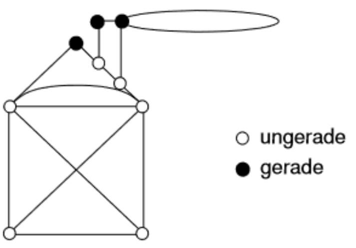
Wir erkennen also, dass die Knotengrade zur Lösung des Problems entscheidend sind. Deshalb folgen hier zwei wichtige Sätze.

Handshaking Lemma: In jedem Graph ist die Summe aller Eckengrade doppelt so groß wie die Anzahl der Kanten [14].

Beweis:
Jede Kante in einem Graphen hat zwei Enden, deshalb ist die Anzahl der Enden eines Graphen mit n Kanten gleich $2 * n$. Dies ist zugleich die Summe der Grade aller Ecken, denn beim Aufsummieren der Eckengrade wird jede Kante doppelt gezählt. Außerdem ist die Zahl gerade. □

Satz: In jedem Graphen ist die Anzahl der Ecken mit ungeradem Grad eine gerade Zahl [14].

Beweis:
Die Aussage dieses Satzes wollen wir mit Hilfe von folgendem Graphen, dem erweiterten Haus vom Nikolaus, zeigen.



Ecken mit ungeradem und geradem Grad

Dazu addieren wir alle Ecken mit ungeradem Eckengrad und nennen die Summe u. Das gleiche machen wir mit den Ecken mit geradem Eckengrad, diese Summe nennen wir g. Die Summe aller Eckengrade ist dann $a = u + g$. Nach dem

Handshaking Lemma ist a eine gerade Zahl. Ebenso ist g gerade, da es die Summe von geraden Zahlen ist. Wir können u auch wie folgt ausdrücken: $u = a - g$. Als Differenz von zwei geraden Zahlen ist u somit auch eine gerade Zahl. Bedenken wir, dass u aus der Addition von lauter ungeraden Zahlen zustande gekommen ist, dann kann die Anzahl dieser Summanden nur eine gerade Zahl sein. Womit wir den Satz gezeigt haben. □

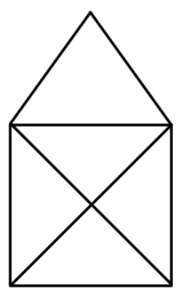
Um die Aussagen von Euler besser nachvollziehen zu können, folgen noch einige Definitionen.

Definition Eulerweg
Ein Weg, der durch jede Kante eines zusammenhängenden Graphen genau einmal führt, heißt *Eulerweg* [10].

Definition Eulertour
Eine *Eulertour*, oder auch *Eulerkreis* genannt, ist ein Eulerweg, der wieder im Ausgangknoten endet [9].

Definition eulersch
Ein Graph, der einen Eulerkreis enthält, nennt man *eulersch* [9].

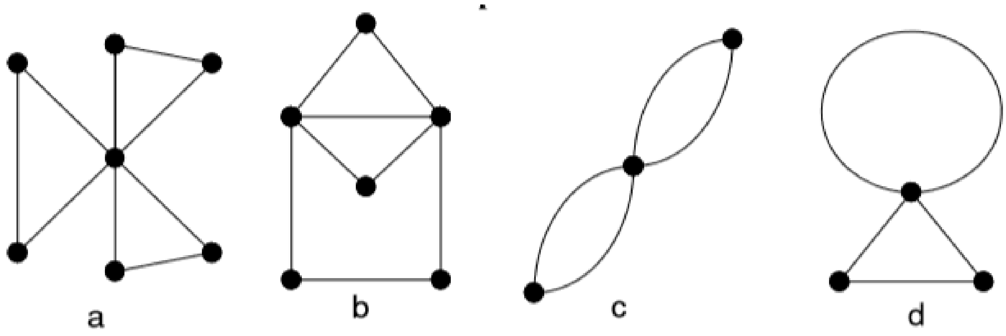
Beispiel: Das wohl bekannteste Beispiel für einen Graphen, der einen Eulerweg enthält, ist das Haus vom Nikolaus.



Ein Graph, welcher genau zwei Ecken mit ungeradem Grad besitzt und die restlichen Ecken geraden Grad haben, kann in einem Zug gezeichnet werden, ohne eine Linie doppelt zu durchlaufen, es existiert also ein Eulerweg [14].

Im Haus vom Nikolaus gibt es zwar einen Eulerweg, jedoch keine Eulertour. Wenn wir es zeichnen, sind Start- und Endknoten verschieden.

Beispiel: In der nachfolgenden Abbildung sind eulersche Touren zu erkennen [14].



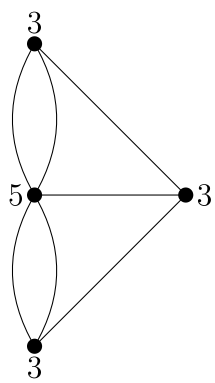
Vier eulersche Graphen

Wann wird jetzt aber ein Graph als *eulersch* bezeichnet?

Um diese Frage zu beantworten, dient folgender Satz:

Satz von Euler: Ist ein Graph zusammenhängend und der Grad jeder Ecke gerade, so ist der Graph eulersch [14].Insbesondere ist ein Graph G eulersch, wenn jede Ecke von G geraden Grad hat. Und umgekehrt gilt: Gibt es eine Eulertour, so haben alle Knoten geraden Grad [10].

Um dies nun direkt für das Königsberger Brückenproblem anwenden zu können, müssen wir uns nur noch einmal überlegen, was die Problemstellung mathematisch beudetet: Ist der Graph des Königsberger Brückenproblems eulersch?
Da wir nun wissen, dass der Satz von Euler darauf eine Antwort liefert, betrachtetn wir noch einmal genau den Graphen und die Grade der Ecken.



Die Zahlen an den Knoten zeigen die Grade an. Da alle Grade des Graphen ungerade sind, existiert keine Eulertour, und die Frage nach dem Spaziergang in Königsberg kann beantwortet werden. Es ist nicht möglich einen Spaziergang so zu wählen, dass man in einem Stadtteil startet, jede Brücke genau einmal überquert und wieder zum Ausgangspunkt zurückkehrt.

Wir wollen nun den *Satz von Euler* beweisen.

Beweis:
Wir betrachten eine beliebige Ecke $e \in V$. Zu zeigen ist, dass die Anzahl der Kanten, die an e angrenzen, gerade ist. Überlegen wir dazu, dass die Eulertour die Ecke e öfters durchläuft. Wie oft genau wissen wir nicht, nehmen wir an, es sei a mal.
Bei jedem Durchgang durch e verbraucht die Eulertour 2 Kanten, dies sind dann in a Durchgängen durch e insgesamt $2 * a$ Kanten. Da keine Kante zweimal durchlaufen werden darf, ist der Grad der Ecke e also mindestens $2 * a$.
Der Grad der Ecke e kann aber auch nicht größer sein, da jede Kante, und somit auch alle Kanten, die an e grenzen, in der Eulertour einmal vorkommen muss.
Womit der Satz von Euler bewiesen ist [1]. □

Wie findet man nun in einem Graphen eine Eulertour, von dem man bereits weiß, dass es ein Eulergraph ist?
Dies gelingt zum Beispiel mit Hilfe des *Zwiebelschalen-Algorithmus*, welcher auch als *Hierholzer-Algorithmus* bekannt ist. Der Algorithmus zeigt, dass in jedem zusammenhängenden Graphen mit geraden Graden der Ecken eine Eulertour gefunden werden kann [10].

Zwiebelschalen- Algorithmus [10]

<u>Eingabe:</u> ein Eulergraph

<u>Ausgabe:</u> eine Eulertour

Schritt 1: Wähle einen Startknoten.

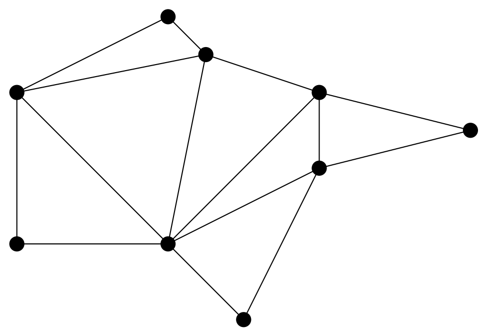
Schritt 2: Laufe von diesem Knoten aus entlang noch unmarkierter Kanten und

markiere die verwendeten Kanten solange, bis ein Knoten erreicht wird, von dem keine unmarkierte Kante mehr ausgeht. Prüfe, ob alle Kanten des Graphen bereits markiert wurden.
Falls ja: Gehe zu Schritt 3.
Falls nein: Suche einen Knoten, der noch unmarkierte Kanten besitzt und wiederhole Schritt 2.

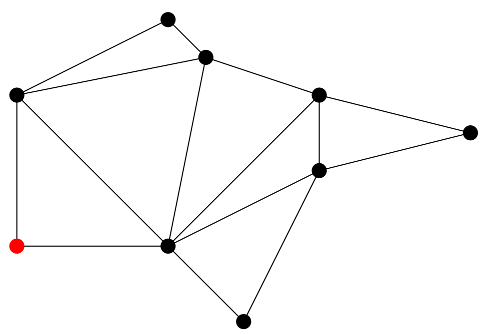
Schritt 3: Die Eulertour wird nun aus den Kreisen zusammengesetzt:
Gehe entlang des ersten Kreises, bis er einen weiteren Kreis berührt. Folge dem neuen Kreis, bis dieser wiederum an einen nächsten Kreis stößt, und so weiter. Findest du keinen neuen beginnenden Kreis, so gehe den zuletzt begonnenen Kreis zu Ende und dann wieder in den vorherigen hinein. Und so weiter, bis alle Kanten besucht wurden.

Beispiel: Hierholzer-Algorithmus

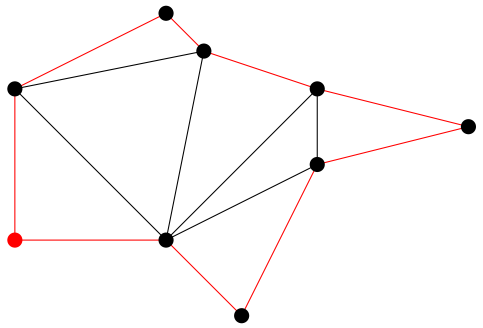
Gegeben sei ein Eulergraph G



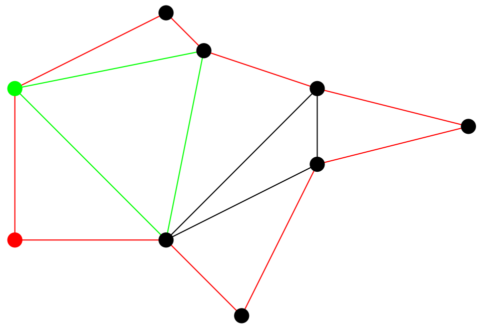
1. Wähle einen Startknoten



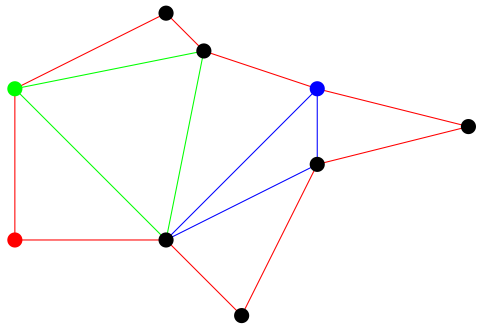
2. Von dieser Ecke aus laufen wir entlang noch unmarkierter Kanten und markieren die verwendeten Kanten solange, bis ein Knoten erreicht wird, von dem keine unmarkierte Kante mehr ausgeht.



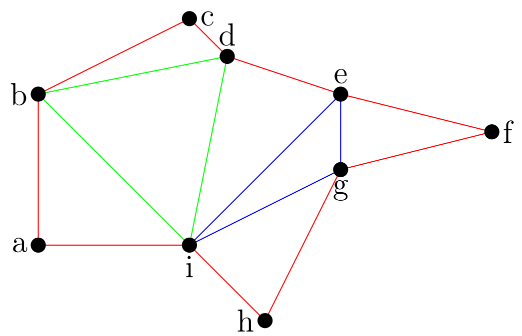
3. Da noch nicht alle Kanten des Graphen markiert sind, suchen wir einen Knoten, der noch unmarkierte Kanten besitzt und wiederholen Schritt 2.



4. Es sind immer noch nicht alle Kanten des Graphen markiert, weshalb wir wieder einen Knoten suchen, der noch unmarkierte Kanten besitzt und wiederholen Schritt 2.



5. Nun sind alle Kanten des Graphen markiert, weshalb wir zu Schritt 3 übergehen und die Eulertour nun aus den Kreisen zusammensetzen. Eine mit Hilfe des Zwiebelschalen-Algorithmus ermittelte Eulertour lautet:
$(a, b, i, e, d, c, b, d, i, g, e, f, g, h, i, a)$



Für den soeben ausgeführten Algorithmus waren zwei Arbeitsschritte notwendig, um eine Eulertour zu erzeugen.
Noch schöner wäre jedoch ein Algorithmus, welcher eine Eulertour in nur einem Arbeitsschritt konstruiert. Dazu dient der sogenannte *Fleury-Algorithmus*, welchen ich im Folgenden kurz vorstellen möchte.

Der Algorithmus von Fleury

Eingabe: ein Eulergraph

Ausgabe: eine Eulertour

Schritt 1: Wähle einen Knoten als Startknoten aus, dieser ist nun der aktuelle Knoten.

Schritt 2: Wähle die nächste noch unmarkierte Kante so, dass sie mit dem aktuellen Knoten inzidiert und unter den noch unmarkierten Kanten keine Brückenkante ist.

Schritt 3: Markiere diese soeben gewählte Kante. Sie ist nun in der Kantenfolge der Eulertour. Der zweite Knoten der Kante ist nun der aktuelle Knoten.

Schritt 4: Sind noch unmarkierte Kanten im Graphen übrig, gehe zu Schritt 2. Ansonsten endet der Algorithmus hier.

Bei diesem Algorithmus kann mit Hilfe des Begriffs der Brücke ein sehr kurzer Algorithmus angegeben werden um Eulertouren erzeugen zu können. Das Problem bei diesem Algorithmus ist es herauszufinden, ob eine Kante eine Brückenkante ist. Dies macht den Arbeitsaufwand nicht wirklich geringer als für den Hierholzer-Algorithmus [10].

Das chinesische Postbotenproblem

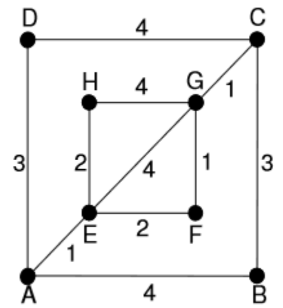
Wir wollen nun noch zum Schluss Touren von Postautos beziehungsweise Briefträgern finden. Da ein Briefträger meist mehrere Straßen in seiner Schicht befahren muss und seine Arbeitszeit gut einteilen möchte, soll sein Weg möglichst kurz sein. Zusätzlich möchte der Briefträger auch wieder bei seinem Ausgangspunkt ankommen, um etwaige Pakete oder Briefe, welche er nicht zustellen konnte, wieder im Depot zu hinterlegen. Um dieses Problem zu modellieren fassen wir die Straßen als Kanten und die Kreuzungen als Knoten eines Graphen auf. Da die Länge der Straßen eine Rolle spielt, betrachten wir gewichtete Graphen. Der chinesische Mathematiker Mei Go Guan hat sich im Jahre 1962 als Erster damit beschäftigt, deshalb ist dieses Problem nach ihm benannt und heute als das chinesische Postbotenproblem bekannt.
Graphentheoretisch betrachtet suchen wir einen geschlossenen Kantenzug durch einen zusammenhängenden gewichteten Graphen, welcher alle Kanten mindestens einmal enthält und dessen Gesamtgewicht minimal ist [14]. Haben wir eine solche Tour gefunden, bezeichnen wir diese als optimale Tour.

Um dieses Problem lösen zu können, müssen wir zwei verschiedene Fälle betrachten. Wir unterscheiden dazu zwei Arten von Graphen, die die Eigenschaft eulersch zu sein aufweisen und jene die dies nicht sind.
Handelt es sich um einen eulerschen Graphen, dann ist die Lösung ganz einfach, denn der Briefträger kann jede Eulertour als seinen Rundgang wählen und hat damit eine optimale Tour gefunden [14].

Ist der Graph aber nicht eulersch, wird die Lösung schon ein wenig komplizierter. Beschränken wir uns auf den Fall, dass der Graph genau zwei Knoten mit ungeradem Knotengrad besitzt. Dies wäre auch leicht zu lösen, wenn der Briefträger in einem Eck mit ungeradem Grad beginnen darf und im anderen Eck, mit ungeradem Grad, seine Tour beenden würde. Allerdings möchten wir, dass der Kantenzug geschlossen ist. Deshalb muss der Briefträger einige Kanten doppelt ablaufen, nämlich genau jene zwischen den beiden Knoten mit ungeradem Grad. In unserem Problem muss dieser Rückweg möglichst kurz sein, womit wir aber diese Aufgabe lösen können. Wir suchen nämlich den kürzesten Weg zwischen zwei Knoten, und diesen Weg muss der Postbote doppelt laufen, alle anderen nur einmal. Wir gehen also wie folgt vor:

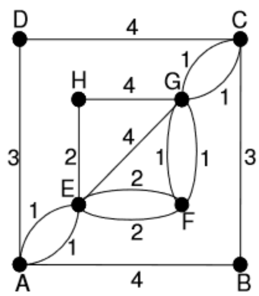
1. Wir suchen die beiden Knoten mit ungeradem Grad und einen kürzesten Weg zwischen ihnen.
2. Anschließend verdoppeln wir die Kanten dieses kürzesten Weges, welchen wir soeben ermittelt haben.
3. Was wir nun erhalten haben, ist ein eulerscher Graph und wir können somit eine Eulertour suchen [14].

Um dieses Prozedere kurz zu veranschaulichen, möchte ich ein Beispiel aus [14] vorstellen:



Die Straßen, in denen Herr Fleißig die Briefe zu verteilen hat (mit Entfernungsangaben)

Die Kantengewichte beschreiben die Entfernungen von einer Ecke zur nächsten. Betrachten wir die Eckengrade genauer, erkennen wir, dass die Ecken D,B,H,G,E und F geraden Grad und die beiden Ecken A und C ungeraden Grad haben. Deshalb suchen wir den kürzesten Weg zwischen den Knoten A und C. Es ist leicht zu erkennen, dass der kürzeste Weg AEFGC ist. Das bedeutet, im nächsten Schritt müssen wir diese Kanten verdoppeln:



Ein minimaler eulerscher Graph zu dem vorigen Graphen

Was wir erhalten haben ist ein gewichteter eulerscher Graph, und wir können mit Hilfe der vorgestellten Algorithmen eine Eulertour finden.
Das Problem kann natürlich auch auf die Müllabfuhr, den Schneeräumungsdienst, der Straßenreinigung oder auf Stadtrundgänge analog angewandt werden [14].

Besitzt ein Graph aber mehrere Ecken ungeraden Grades, so ist die Anzahl der Ecken ungeraden Grades immer gerade. Daher kann man diese Ecken (ungeraden Grades) in Paare zusammenfassen und nennt dieses Prozedere Matching. Da wir immer noch den kürzesten Weg zwischen zwei Ecken ungeraden Grades möchten, fassen wir jene zwei Ecken zusammen, sodass die Summe der Entfernungen zwischen den Ecken minimal wird. Auch hier verdoppeln wir wieder für jedes der Paare die Kanten des kürzesten Weges [15].

Falls, der Graph ein gerichteter Graph ist, kann das chinesische Postbotenproblem ganz analog zum ungerichteten Fall behandelt werden. Die einzig nötige Änderung wäre die Betrachtung des Verhältnisses von Ein- und Ausgangsgrad des Graphen statt der Knotengrade [10].

In der realen Arbeitswelt wird ein Graph, welcher ein Straßennetz repräsentieren soll, weder gerichtet noch ungerichtet sein. Denn in den meisten Städten gibt es neben den gewöhnlichen Straßen auch Einbahnstraßen, womit wir einen sogenannten gemischten Graphen erhalten. Dieses Problem, das gemischte chinesische Postbotenproblem, gehört zu der Klasse der NP- schweren Probleme. Das bedeutet, dass es vermutlich keinen effizienten Algorithmus gibt, um dieses Problem zu lösen. Es zählt zu den 7 Milleniumsproblemen, für deren Lösung im Jahr 2000, ein Preisgeld von je 1.000.000 US$ ausgesetzt wurde [10].

2 Algebraische Graphentheorie im Schulunterricht

2.1 Lehrplanbezug

Ziel dieses Abschnittes wird es sein, zu diskutieren, ob und inwieweit das Thema algebraische Graphentheorie an allgemeinbildenden höheren Schulen (AHS) bearbeitet werden kann/soll. Ausgangspunkt dafür ist die aktuelle Fassung des AHS-Lehrplans, über den das Rechtsinfomationssystem des Bundes (RIS) Auskunft gibt [7].

Es verwundert keineswegs, dass die Graphentheorie in den ersten vier - allgemeinen - Teilen des AHS-Lehrplans (Allgemeines Bildungsziel, Allgemeine Didaktische Grundsätze, Schul- und Unterrichtsplanung, Stundentafeln) nicht genannt wird. Dass sie jedoch auch an keiner Stelle der AHS-Fachlehrpläne für den Pflichtgegenstand Mathematik - sowohl für die Unterstufe wie für die Oberstufe - genannt ist, verwundert doch. Obwohl der konkrete Hinweis auf die algebraische Graphentheorie in den detaillierten Lehrstoffangaben der Mathematik-Fachlehrpläne fehlt, ist daraus nicht ableitbar, dass die Behandlung der algebraischen Graphentheorie im Mathematikunterricht an AHS keinen Platz hat. Warum es also gerechtfertigt ist, dieses Thema mit SchülerInnen zu behandeln, zeigt ein genauer Blick auf unterschiedliche Stellen des AHS-Lehrplans.

Im Fachlehrplan des Pflichtgegenstands Mathematik für die AHS-Unterstufe ist als **Bildungs- und Lehrauftrag** unter anderem formuliert:

- *Das Bilden mathematischer Modelle und das Erkennen ihrer Grenzen soll zu einem verantwortungsvollen Umgang mit Aussagen führen, die mittels mathematischer Methoden entstanden sind [7]*

- *Die SchülerInnen sollen durch das Benutzen entsprechender Arbeitstechniken, Lernstrategien und heuristischer Methoden Lösungswege und -schritte bei Aufgaben und Problemstellungen planen und in der Durchführung erproben [7]*

In sehr vereinfachter Form lässt sich schon in höheren Klassen der Unterstufe ein Stadtplan als Graph darstellen und somit ein mathematisches Modell bilden. Auch einfache Algorithmen sind bestimmt schon in diesen Schulstufen durchführbar und setzen die eben dargestellten Forderungen aus dem Lehrplan um.

Der zitierte Fachlehrplan fordert vom Mathematikunterricht die Entwicklung konkret genannter **mathematischer Grundtätigkeiten**, unter anderem:

- *Darstellen und Interpretieren, insbesondere: verbales, formales oder graphisches Darstellen von Sachverhalten [7]*

Im Bereich der Didaktischen Grundsätze des zitierten Fachlehrplanes wird zur **Motivierung von SchülerInnen** ausgeführt:

- *Mit Hilfe von Problemstellungen aus Themenkreisen, die den Erfahrungen und Interessen der Schülerinnen und Schüler entsprechen, sollen mathematisches Wissen und Können entwickelt und gefestigt werden. Dabei soll die Nützlichkeit der Mathematik in verschiedenen Lebens- und Wissensbereichen erfahren werden. Wünschenswert für diese Phase ist eine Mitverantwortung durch die Schülerinnen und Schüler. Hilfen oder Informationen sollen dann erfolgen, wenn sie verlangt oder benötigt werden. Selbstständiges Entdecken und Erfolgserlebnisse sind ein wesentlicher Beitrag zur Motivation. [7]*

Die algebraische Graphentheorie beinhaltet einige Themen, welche bereits 12- bis 14- jährige SchülerInnen interessieren, sei es der kürzeste Weg von zu Hause bis zur Schule oder Ähnliches.

Auch im Fachlehrplan Mathematik für die Oberstufe finden sich an mehreren Stellen Hinweise, die eine Behandlung der algebraischen Graphentheorie im Unterricht sinnvoll erscheinen lassen. So wird im Bereich der **Bildungs- und Lehraufgabe** unter anderem gefordert:

- *Die mathematische Beschreibung von Strukturen und Prozessen der uns umgebenden Welt, die daraus resultierende vertiefte Einsicht in Zusammenhänge und das Lösen von Problemen durch mathematische Verfahren und Techniken sind zentrale Anliegen des Mathematikunterrichts. [7]*

Auch die im Folgenden angeführten Hinweise im Lehrplan lassen sich als Argumente für die Behandlung der Graphentheorie im Unterricht heranziehen:

- *Mathematik ist eine spezielle Form der Erfassung unserer Erfahrungswelt. Sie ist eine spezifische Art, die Erscheinungen der Welt wahrzunehmen und durch Abstraktion zu verstehen. Mathematisierung eines realen Phänomens kann die Alltagserfahrung wesentlich vertiefen. [7]*

- *Der Unterricht soll aufzeigen, dass Mathematik in vielen Bereichen des Lebens (Finanzwirtschaft, Soziologie, Medizin, ...) eine wichtige Rolle spielt. [7]*

- *Viele Naturphänomene lassen sich mit Hilfe der Mathematik adäquat beschreiben und damit auch verstehen. Die Mathematik stellt eine Fülle von Methoden zur Verfügung, mit denen Probleme bearbeitbar werden. [7]*

- *Mathematik besitzt neben der deduktiven auch eine induktive Seite. Vor allem das Experimentieren im Rahmen der Bearbeitung neuer Aufgaben und Probleme macht diese Seite sichtbar, bei der Kreativität und Einfallsreichtum gefördert werden. [7]*

Auch der folgende Satz aus den **Didaktischen Grundsätzen** des Fachlehrplans für die Oberstufe mag als Beleg für die Sinnhaftigkeit der Berücksichtigung der Graphentheorie im Oberstufenunterricht dienen:

- *Anwendungsorientierte Kontexte verdeutlichen die Nützlichkeit der Mathematik in verschiedenen Lebensbereichen und motivieren so dazu, neues Wissen und neue Fähigkeiten zu erwerben. [7]*

Im neuen Lehrplan für die AHS-Oberstufe sind die **mathematischen Kompetenzen** komplett überarbeitet worden und an das Kompetenzmodell angepasst. Es unterscheidet die Inhalts-, Handlungs- und Komplexitätsdimension. Welche Dimension inwieweit bei der Erarbeitung der algebraischen Graphentheorie gerade im Einsatz ist, kommt individuell auf die Durchführung an und kann nicht pauschal gesagt werden. Weshalb ich hier auf das RIS [7] verweisen möchte.

Zusammenfassend sei bemerkt, dass die Fachlehrpläne Mathematik sowohl für die Unterstufe als auch für die Oberstufe gute Argumente für die alters- und situationsangepasste Behandlung von Elementen der Graphentheorie im Pflichtgegenstand Mathematik liefern. Zudem bietet der AHS-Lehrplan auch außerhalb des Pflichtgegenstandes Mathematik Möglichkeiten, die Graphentheorie zu thematisieren. Insbesondere der vertiefende Wahlpflichtgegenstand Mathematik (ab der 6. Klasse/10. Schulstufe), aber auch Freigegenstände für besonders motivierte/begabte SchülerInnen (z.B. Kurse zur Mathematikolympiade) seien genannt. Darüber hinaus eignet sich die algebraische Graphentheorie auch sehr gut für eine vorwissenschaftliche Arbeit (VWA) eines Maturanten oder einer Maturantin. Ein weiterer idealer Zeitpunkt, um einen Unterrichtsblock diesem Thema zu widmen, sind die Unterrichtsstunden vor etwaigen Ferien, sofern der reguläre Lehrstoff schon bearbeitet worden ist, oder Projekttage. Je nach Schulstufe lassen sich der Schwierigkeitsgrad und Komplexität adaptieren, indem man zum Beispiel auf Beweise verzichtet oder diese nur skizziert. Des weiteren steht es auch jeder Lehrperson offen, welche Themen der Graphentheorie behandelt und welche Algorithmen besprochen werden. Somit kann die Lehrperson diese Stunden sinnvoll nutzen, indem sie nicht nur ihren Lehrauftrag erfüllt und den Lehrplan befolgt, sondern noch zusätzlich den SchülerInnen spielerisch ein sehr interessantes und faszinierendes Fachgebiet der Mathematik näher bringt.

2.2 Graphentheorie unterrichten

Die Frage nach dem Wie (Wie unterrichtet man Graphentheorie am besten?) ist nicht verallgemeinernd zu beantworten. In der Didaktik wurden diverse didaktische Prinzipien formuliert, welche die verschiedenen Forschungsbereiche, wie etwa die Erziehungswissenschaft, der Pädagogik, der Psychologie und viele weiter, beinhalten und den Lehrpersonen eine Stütze sein sollen.
Die Anwendung ist von der Klassensituation und auch von der Persönlichkeit und den Lehrerfahrungen der Lehrperson abhängig.
Es soll helfen, den Unterricht zu planen, reflektieren und diesen zu rechtfertigen. Welche Prinzipien für welche Klasse, Schulstufe oder Unterrichtsstoff am Besten geeignet ist, kann die Lehrperson selbst entscheiden. [12].

Beispielhaft möchte ich hier das **operative Prinzip** genauer erläutern und die Lernkultur **handlungsorientierter Unterricht** vorstellen. Beide eignen sich meiner Meinung nach gut für die Bearbeitung des Themas algebraische Graphentheorie.

Bevor ich jedoch diese Theorien vorstelle, möchte ich noch ein Zitat des chinesischen Philosophen Konfuzius (551 - 479 v. Chr.) aufgreifen.

„Sage es mir, und ich werde es vergessen. Zeige es mir, und ich werde es vielleicht behalten. Lass es mich tun, und ich werde es können.“

Ich habe dieses Zitat im Laufe meines Studiums kennengelernt und es hat mich von Beginn an beeindruckt und bis jetzt begleitet. Nicht nur weil ich der Meinung bin, dass SchülerInnen etwas erst richtig können, wenn sie es anwenden, sondern auch weil ich selbst schon diese Erfahrung bei meinen NachhilfeschülerInnen und bei mir selbst gemacht habe. Erst wenn ich mit *Herz*, *Hirn* und *Hand* bei der Sache bin, kann ich persönlich sagen, dass ich es wirklich kann. Und genau dies ist der Grundgedanke eines handlungsorientierten Unterrichts. Der Unterricht soll das Augenmerk auf die aktive Konstruktion der SchülerInnen legen und nimmt dafür sogar den Verzicht auf inhaltliche Vollständigkeit in Kauf. Wichtig ist, dass die SchülerInnen Hirn und Hand in einem ausgewogenem Verhältnis verwenden

und mit Herz bei der Sache sind. Das Lernen mit Herz, Hirn und Hand geht auf Johann Heinrich Pestalozzi (1746-1827) zurück und soll verdeutlichen, dass Lernen ganzheitlich geschieht [13].
Ebenso sagt das sogenannte operative Prinzip oder auch aktives Lernen, dass die Anweisungen der LehrerInnen von aktiven Konsturkionen der SchülerInnen begleitet werden sollen, damit der Unterricht zum gewünschten Ziel führt [12].
Um diese Theorie jetzt aber auch umsetzten zu können, sollen folgende Maßnahmen aus [12] helfen:

- angreifbares Unterrichtsmaterial
- inhaltlich interessanter und an die Lebenserfahrung der SchülerInnen anknüpfender Stoff
- Problemaufgaben und spielerische Momente

Natürlich ist all dies keine Garantie dafür, dass der Unterricht damit gelingt und perfekt abläuft. Vielmehr soll es, wie bereits eingangs erwähnt, eine Stütze für Lehrpersonen sein. Das hier vorgestellte Unterrichtskonzept ist ein Vorschlag für Lehrpersonen und bedarf der Anpassung an die jeweils aktuelle Lernsituation.

2.3 Unterrichtskonzept

Am Ende meiner Arbeit möchte ich nun ein von mir entwickeltes Unterrichtskonzept für die Behandlung des Themas "Routenplanungïn einer 6. Klasse (10. Schulstufe) vorstellen. Es ist von den schon erläuterten unterrichtstheoretischen Prinzipien (operatives Lernen; Handlungsorientierung) geprägt.

Kontextüberlegung

Thema: Routenplanung Fach: Mathematik Schultyp: Allgemeinbildende höhere Schule Schulstufe: 2. Klasse Oberstufe, 10. Schulstufe zur Verfügung stehende Unterrichtsstunden: 3 Doppelstunden zu je 100 Minuten

Bedingungsanalyse
Ziel der Bedingungsanalyse ist es, sich genau vor Augen zu führen, **wem** ich etwas vermitteln möchte. Dazu helfen einige Leitfragen:

Was wissen und können die SchülerInnen bereits?
Voraussetzungen an ein fachliches Vorwissen gibt es außer dem Rechnen mit natürlichen Zahlen keine. Es muss also kein bestimmtes mathematisches Thema vor dieser Unterrichtseinheit abgehalten worden sein. Allerdings ist es von Vorteil, wenn sich die Klasse bereits mit anderen Algorithmen auseinandergesetzt hat, wie zum Beispiel mit dem Algorithmus zur Bestimmung des größten gemeinsamen Teilers. Ich gehe bei meiner Planung davon aus, dass die SchülerInnen wissen, was ein Algorithmus ist und wie Algorithmen im Allgemeinen ablaufen.

Wo gibt es Anknüpfungsmöglichkeiten zur Lebenswelt der SchülerInnen?
Routenplanung kommt nicht nur dann vor, wenn man sein Navigationsgerät nach der optimalen Route fragt, sondern auch dann, wenn man zum Beispiel die kürzeste U-Bahn-Verbindung zwischen zwei Orten sucht. Im Alter von 15 oder 16 Jahren machen einige SchülerInnen in den Sommerferien schon erste Arbeitserfah-

rungen in einem Sommerjob, so zum Beispiel mit dem Austragen von Zeitungen oder Briefen. Aus diesem Grunde liegt es auf der Hand, dass sie an einem optimalen Weg interessiert sind. Da die Graphentheorie eine große Anwendungsspanne aufweist, ist das Interesse der SchülerInnen als hoch einzuschätzen.

Wie setzt sich die Schülergruppe zusammen und welche SchülerInnen sind besonders auffällig?
Es erscheint sinnvoll, wenn sich die Lehrperson im Vorhinein mit diesen Fragen befasst, um z.B. durch ein situationsangepasstes Setting (Wahl der Unterrichtsform) Unterrichtsstörungen möglichst auszuschließen.

Sachanalyse
Der Kern der Sachanalyse besteht darin, sich zu überlegen, **was** der fachliche Hintergrund des Themas ist. Dies wird im ersten Teil dieser Arbeit dargestellt.

Didaktische Analyse
Die Bearbeitung der folgenden vier Fragen im Vorfeld des Unterrichts erscheint für eine erfolgreiche Gestaltung der Unterrichtssequenz sinnvoll:

Warum ist das Thema für die SchülerInnen wichtig?
Wie schon so oft erwähnt, ist die Graphentheorie ein Wissenschaftsgebiet der Mathematik, welches auch im SchülerInnen-Alltag Einzug findet. Die Tatsache, dass die algebraische Graphentheorie noch nicht vollständig erforscht ist, und sich ständig weiterentwickelt, kann für SchülerInnen reizvoll und motivierend sein.

Wohin soll der Unterricht führen?
Der Unterricht soll die SchülerInnen befähigen, Graphen zu modellieren, den historischen Bezug und den Ursprung der Graphentheorie zu kennen. Sie werden den Breitensuche-Algorithmus kennenlernen und Eulertouren in Eulergraphen mit Hilfe des Hierholzer-Algorithmus finden können.

Wie gehe ich vor und warum wähle ich diese Schritte?
Eine gut reflektierte Unterrichtsplanung zieht sich wie ein roter Faden durch die dem Thema gewidmeten Unterrichtseinheiten.

1. Doppelstunde
Nach dem Begrüßungsritual und der Klärung diverser organisatorischer Dinge, kündigt die Lehrperson an, was in den nächsten drei Doppelstunden passieren wird. In der ersten Doppelstunde wird durch einen geschichtlichen Exkurs in das Thema eingeführt und der Begriff **Graph** erläutert. Die nächste Unterrichtseinheit steht zur Verfügung um kürzeste Wege zu finden. Dabei erarbeiten sich die SchülerInnen ihr Wissen selbstständig mit Hilfe eines Arbeitsblattes und am Ende der Doppelstunde wird gemeinsam der BFS-Algorithmus festgehalten. In der letzten Doppelstunde geht es dann um Eulergraphen und Eulertouren. Auch hier wird vieles durch selbstständiges Handeln der SchülerInnen erarbeitet.

> *Den SchülerInnen wird dadurch eine inhaltliche Klarheit geboten. Sie wissen was sie zu erwarten haben und was in den nächsten Stunden geschehen wird. Es wird ihnen dadurch ein Überblick gegeben.*

Anschließend an diese Verlaufserörterung folgt die geschichtliche Einführung in das Thema der algebraische Graphentheorie durch das Königsberger Brückenproblem. Dieses wird von der Lehrperson erzählt, und zusätzlich wird der Stadtplan an die Wand oder Tafel projiziert.

> *Die geschichtliche Einführung wird deshalb gewählt, damit die SchülerInnen das Thema historisch einordnen können und ihnen bewusst wird, dass die algebraische Graphentheorie, im Gegensatz zu vielen anderen mathematischen Gebieten, recht neu ist. Zur Veranschaulichung dient der Stadtplan.*

Nun sollen sich die SchülerInnen selbst mit dem Problem auseinandersetzen und überlegen, welche Wege es gibt und versuchen, den Stadtplan etwas vereinfachter zu zeichnen und eine Skizze davon anzufertigen. Im Anschluss können freiwillige SchülerInnen ihre Erkenntnisse in kleinen Präsentationen vorstellen.

> *Hier wird das aktive Lernen umgesetzt. Die SchülerInnen setzen sich spielerisch mit der Materie auseinander und versuchen so das Problem zu lösen.*

Nach der Pause geht es etwas frontal weiter, indem die Lehrperson die Begriffe Graph, Grad einer Ecke und zusammenhängend, definiert. Dieses neu erlernte Wissen soll nun mit ihrer vorherigen Skizze in Verbindung gebracht werden. Danach wird noch gemeinsam der Stadtplan von Königsberg als Graph modelliert.

> *Durch die Vernetzung der neuen Begriffe und der Skizze bekommen die SchülerInnen einen Sinn hinter ihrer Tätigkeit.*

Die Methode stille Post, welche ich im Laufe meines Studiums kennenlernen durfte, kommt danach zum Einsatz, um die Darstellung der Graphen noch einmal zu üben. Durch das Auffalten des Blattes können die SchülerInnen selbst die Korrektheit überprüfen.

> *Um noch vertrauter mit dem Begriffen zu werden, können die SchülerInnen wieder spielerisch selbst handeln.*

Am Ende dieser Doppelstunde leitet die Lehrperson zum BFS-Algorithmus über und projiziert das Bild eines U-Bahn-Plans an die Wand. Es wird besprochen, dass es Ziel der nächsten Stunde sein wird, den kürzesten Weg von A nach B zu finden. Abschließend sollen die SchülerInnen den Plan als Graph modellieren.

> *Durch den Ausblick, was nächste Stunde geschehen wird, bekommen die SchülerInnen einen Überblick.*

2.Doppelstunde
Zu Beginn fasst die Lehrperson die wichtigsten Erkenntnisse der ersten Doppelstunde zusammen, bevor die SchülerInnen ihren Graphen, welchen sie am Ende der letzten Einheit erstellt haben, vorstellen.

> *Die Lehrperson holt damit das neu erlernte Wissen der SchülerInnen wieder ins Gedächtnis und die SchülerInnen können leichter anknüpfen.*

Der Großteil dieser Stunde wird zur Bearbeitung des Arbeitsblattes verwendet. Die Lehrperson teilt jedem Schüler/jeder Schülerin ein Exemplar aus und bespricht Verständnisfraen mit den SchülerInnen. Während die SchülerInnen aktiv sind und das Arbeitsblatt bearbeiten, steht die Lehrperson als Coach zur Verfügung.

> *Die SchülerInnen sind auch hier wieder selbst aktiv und es ist das operative*

Prinzip zu erkennen. Durch die eigene Tätigkeit werden Hand, Herz und Hirn angesprochen und dies soll den SchülerInnen eine Hilfe sein, die Dinge leichter zu lernen.

Nach der kurzen Pause leitet die Lehrperson zu den Präsentationen über. Die SchülerInnen sollen in Zweierteams ihre Ergebnisse vorstellen. Dabei fängt eine Gruppe an und ein Team mit ähnlichen Erkenntnissen ergänzt. Dies geschieht solange, bis alle Gruppen an der Reihe waren, oder niemand mehr etwas ergänzen möchte. Die Lehrperson notiert währenddessen die Ergebnisse stichwortartig an der Tafel.

Zum einen geht es natürlich darum, dass die SchülerInnen erkennen sollen, dass sie das Arbeitsblatt nicht umsonst ausgefüllt haben und ihre Ergebnisse den Unterrichtsverlauf später beeinflussen werden. Aber ein weiterer, ganz wichtiger Aspekt dieser kleinen Präsentationen soll das Sprechen vor einer Gruppe darstellen. Die SchülerInnen sollen damit auf spätere Berufssituationen vorbereitet werden.

Unter Berücksichtigung der Notizen an der Tafel wird nun der Breitensuche-Algorithmus von der Lehrperson an der Tafel vollständig formuliert und aufgeschrieben.

Dies ist nötig, damit alle SchülerInnen am Ende der zweiten Dopplestunde den selben - und richtigen - Algorithmus in ihren Heften haben. Natürlich wird zu den Notizen eine Verbindung hergestellt, damit die Arbeit der SchülerInnen auch gewürdigt wird.

3. Doppelstunde

Auch zu Beginn dieser dritten Doppelstunde wird die vergangene Einheit in groben Zügen wiederholt. Die Lehrperson kommt hier zum Schluss, dass das Königsberger Brückenproblem immer noch nicht beantwortbar ist.

Die Wiederholung dient ebenfalls wieder dazu, dass das neu gelernte Wissen wieder ins Gedächtnis gerufen wird. Durch die Anknüpfung an das Königsberger Brückenproblem wird ein großer Bogen zum Beginn der Unterrichtsreihe gespannt.

Die restliche Stunde wird für die Bearbeitung des Arbeitsblattes zu Eulergraphen verwendet. Die Lehrperson leitet wieder an; den Großteil der Stunde sind dann jedoch die SchülerInnen aktiv und die Lehrperson hält sich im Hintergrund auf, beziehungsweise gibt jenen Hilfestellungen, die diese benötigen.

> *Durch die selbständige Bearbeitung des Blattes sind die SchülerInnen wieder aktiv und handeln selbstständig.*

Der zweite Teil der Doppelstunde wird dafür verwendet, um das vorherige Arbeitsblatt im Plenum zu besprechen. Das Gespräch im Plenum soll im Idealfall zu einem sehr wichtigen Satz über Eulertouren führen. Dieser wird anschließend gemeinsam formuliert.

> *Die SchülerInnen übernehmen Verantwortung für ihre eigenen Arbeitsschritte; durch das Aufgreifen ihrer Erkenntnisse erfahren sie Wertschätzung.*

Den Abschluss der gesamten Unterrichtseinheit bildet wieder ein Arbeitsblatt. Die SchülerInnen und die Lehrperson nehmen die bekannten Rollen ein. Dieses Arbeitsblatt zu Eulertouren muss allerdings nicht mehr gemeinsam besprochen werden, da sich die SchülerInnen am Ende gegenseitig kontrollieren. Sollten dennoch Fragen auftauchen, steht die Lehrperson natürlich zur Verfügung.

In den letzten fünf Minuten beendet die Lehrperson die Einheit und fasst noch einmal das Wichtigste zusammen. Hier ist auch noch Platz für Verweise auf aktuelle Anwendungsgebiete der Graphentheorie, um den SchülerInnen die Allatgsrelevanz dieser Methode neuerlich vor Augen zu führen.

> *Den SchülerInnen wird dadurch noch einmal vor Augen geführt, was sie in den letzten Stunden alles geleistet und gelernt haben.*

Welche Materialien gehören dazu?

Zum einen benötigen die SchülerInnen ein Stück Karton und eine Schere, um sich eine Lochblenden zu basteln. Ansonsten wird außer ein paar Zetteln, der Tafel, Kreide und eventuell einem Beamer oder Overheadprojektor nicht viel zusätzliches Material benötigt. Die SchülerInnen sollen allerdings einige Gedanken in ihren sogenannten Laborheften festhalten. Sie sind mit dieser Arbeitsweise vertraut und wissen wie sie ihr Laborheft zu verwenden haben. Falls die Klasse nicht mit Laborheften arbeitet, kann stattdessen auch einfach das Schulübungsheft zur Hand genommen werden.

Unterrichtsentwurf

Der Unterrichtsentwurf besteht zum einen aus den ausformulierten Lernzielen und zum anderen aus dem geplanten Verlauf. Letzteres wird meist in Form einer Tabelle übersichtlich dargestellt. Die Lernziele, welche ich in der ILS-Lehrveranstaltung so kennenlernen durfte, setzen sich aus den fachlichen Kompetenzen, welche die SchülerInnen erlangen sollen, und den überfachlichen Kompetenzen, wie den sozialen und personalen Kompetenzen, zusammen. Generell sollten Lernziele möglichst konkret benannt werden und eine Messbarkeit aufweisen, dies bedeutet, es muss überprüft werden können, ob das gewünschte Ziel erreicht worden ist. Des Weiteren sollte es akzeptabel und realistisch sein und in einer bestimmten Zeit erreicht werden können.In aller Kürze: Sie sollten SMART (spezifisch, messbar, akzeptabel, realistisch und terminisiert) sein.

Lernziele:

Die SchülerInnen können...

- ... diverse Stadtpläne und Straßennetze als Graph modellieren.
- ... aus gegebener Ecken- und Kantenmenge einen Graphen zeichnen.
- ... Ecken und Kanten unterscheiden und Eigenschaften der beiden Mengen aufzählen.

- ... in einem Graphen den kürzesten Weg mit Hilfe des Breitensuche-Algorithmus bestimmen.
- ... eine Eulertour von einem Eulerweg unterscheiden und beide in eigenen Worten definieren.
- ... entscheiden, ob es in einem Graphen eine Eulertour gibt.
- ... mit Hilfe des Hierholzer-Algorithmus eine Eulertour in einem eulerschen Graphen bestimmen.

Geplanter Verlauf:

1. Doppelstunde:

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:05	Begrüßung	Lehrperson kündigt den Verlauf der nächsten Unterrichtseinheiten an		
0:05 - 0:15	Einführung (Königsberger Brückenproblem)	LehrerIn erzählt Geschichte von Königsberg, projiziert Stadtplan von Königsberg an Tafel	Frontal	Beamer
0:15 - 0:20	Arbeitsauftrag anweisen	LehrerIn stellt der Klasse die Frage, welche Wege möglich sind und ob der Plan auch leichter bzw. einfacher darstellbar ist. SchülerInnen sollen sich dies mit Sitznachbarn überlegen und Gedanken im Laborheft festhalten.	Frontal	
0:20 - 0:35	Arbeitsauftrag	SchülerInnen besprechen mit Banknachbar die Fragen und schreiben Überlegungen ins Laborheft.	PA	Laborheft
0:35 - 0:50	Ergebnissicherung	SchülerInnen präsentieren ihre Erkenntnisse und stellen ihre Möglichkeiten vor (freiwillige). Lehrperson greift (falls notwendig) ein und stellt richtig.	kleine Präsentationen	evt. Tafel und Kreide

Anschließend folgt eine kurze Pause (meistens 5 Minuten).

Fortsetzung 1. Doppelstunde:

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:15	Def. Graph und Co	Lehrperson definiert die Begriffe: Graph, zusammenhängend und Grad einer Ecke und stellt Bezug zu den Präsentationen der SchülerInnen her	Frontal	Tafel und Kreide
0:15 - 0:20	Bsp Graph	gemeinsame Bearbeitung eines Beispiels: ges: Graph, geg: $V = \{1, 2, 3, 4, 5\}$ und $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 1\}, \{4, 5\}, \{5, 2\}\}$	Frontal	Tafel
0:20 - 0:35	Graphen und Stille Post	SchülerInnen sind in 3er Gruppen eingeteilt, alle schreiben Knoten- und Kantenmenge auf Blatt, nun geht Blatt die Runde und zweiter zeichnet den durch die Mengen beschriebenen Graphen, falten die erste Zeile um und geben Blatt wieder weiter. Nun wird von Graphen die Knoten und Kantenmenge abgelesen.	Stille Post, 3er Teams	Zettel und Stift
0:35 - 0:40	Kontrolle	Zettel wird aufgefaltet und in 3er Teams besprochen für Fragen ist Lehrperson da.		
0:40- 0:45	Einführung Breitensuchalgorithmus	Lehrperson projiziert U-Bahn- Plan, Ziel der nächsten Stunde einen kürzesten Weg zu finden.	Plenum	Beamer
0:45 - 0:50	Modellieren	SchülerInnen sollen den U-Bahn Plan als Graph modellieren.	PA	

2. Doppelstunde

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:05	Begrüßung + kurze WH	Lehrperson fasst die wichtigsten Erkenntnisse der letzten Stunde zusammen	Frontal	evt. Tafel und Kreide
0:05 - 0:10	Graph modellieren	U-Bahn-Plan als Graph von SchülerInnen vorgestellt (von letzter Stunde)	Präsentationen	Tafel und Kreide
0:10 - 0:15	Anweisung Arbeitsblatt BFS	Lehrperson teilt Arbeitsblatt zu BFS aus und bespricht diesen mit den SchülerInnen kurz, Fragen werden geklärt und SchülerInnen in zweier Gruppen eingeteilt		Arbeitsblatt BFS
0:15 - 0:45	Arbeitsblatt BFS	SchülerInnen sind aktiv und Lehrperson für Fragen da	PA	Arbeitsblatt, Karton und Schere
0:45 - 0:50	Abschluss	Lehrperson achtet darauf, dass SchülerInnen zu Ende kommen und ein Ergebnis haben		

Anschließend folgt eine kurze Pause (meistens 5 Minuten).

Fortsetzung 2. Doppelstunde:

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:05	Überleitung zu Präsentationen	alle Zweierteams sollen ihre Ergebnisse vorstellen, ein Team fängt an und das Team mit ähnlichen Ergebnissen macht weiter und ergänzt	Frontal	
0:05 - 0:40	Präsentationen	SchülerInnen präsentieren und Lehrperson schreibt Stichwörter an Tafel mit	Präsentation	Tafel und Kreide
0:40 - 0:50	BFS-Algorithmus formulieren	Lehrperson schreibt, unter Berücksichtigung der SchülerInnen-Ergebnisse, den Algorithmus an die Tafel	Frontal	Tafel

Bemerkung: An dieser Stelle kann nach Bedarf auch eine Hausübung gegeben werden. Zum Beispiel sollen die SchülerInnen an einem Graphen den Algorithmus üben.

3. Doppelstunde:

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:05	Begrüßung			
0:05 - 0:15	WH + Überleitung zum Königsberger Brückenproblem	Lehrperson wiederholt wichtigsten Punkte der letzten Stunde und kommt zum Schluss, dass Königsberger Brückenproblem nocht nicht beantwortbar ist –> Arbeitsblatt	Frontal	
0:15 - 0:45	Arbeitsblatt Eulersche Graphen	SchülerInnen sind aktiv und arbeiten am Arbeitsblatt, Lehrperson steht für Fragen zur Verfügung	EA	Arbeitsblatt Eulersche Graphen
0:45 - 0:50	Abschluss	Lehrperson achtet darauf, dass SchülerInnen zum Ende gelangen		

Anschließend folgt eine kurze Pause (meistens 5 Minuten).

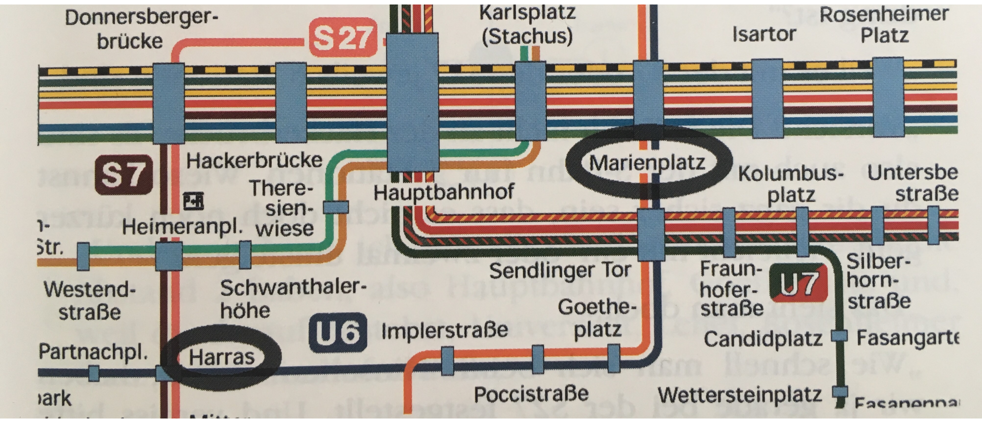
Fortsetzung 3. Doppelstunde:

Zeit	Grobgliederung	Lernaktivität	Sozialform	Material
0:00 - 0:15	Besprechung Arbeitsblatt Eulersche Graphen	Lehrperson bespricht im Plenum die Lösungen der SchülerInnen	Plenum	Arbeitsblatt Eulersche Graphen
0:15 - 0:20	Satz Eulertour	Ein zusammenhängender Graph, mit geraden Eckengraden existiert eine Eulertour. (im Idealfall ist dies bereits aus Gespräch im Plenum herausgekommen)	Frontal	Tafel
0:20 - 0:45	Arbeitsblatt Eulertouren	SchülerInnen aktiv, Lehrperson für Fragen bereit	PA	Arbeitsblatt Eulertouren
0:45 - 0:50	Abschluss	Lehrperson fasst zusammen was in den letzten 3 Doppelstunden passiert ist und verweist noch einmal auf die Anwendungsgebiete der Graphentheorie	Frontal	

2.3.1 Arbeitsblätter

1. Arbeitsblatt: Breitensuche

1. Modelliere den folgenden U-Bahn-Plan [9] als Graph in dein Laborheft. Achte darauf, dass du deinen Graphen groß genug zeichnest.



2. Suche den Weg von Harras zum Marienplatz, welcher die kleinste Anzahl an Kanten benötigt.
Notiere dabei genau, wie du vorgegangen bist.

3. Bastle aus Karton und Schere (bekommst du von deiner Lehrperson am Pult) eine Lochblende, so groß, damit du sie für deinen Graphen verwenden kannst.

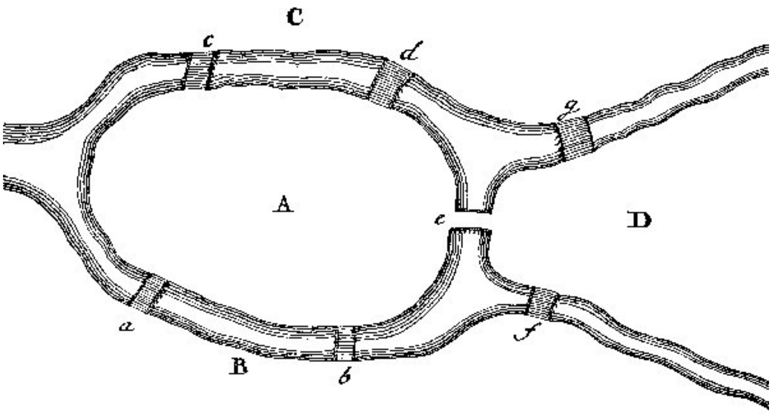
In der Realität sind U-Bahn-Netze viel größer und die kürzesten Wege von A nach B werden mit Hilfe eines Algorithmus von Computern gefunden. Der Computer kennt zu Beginn nur den Ausgangspunkt.
Du darfst jetzt auch einmal Computer spielen, und damit du dich besser in die Lage versetzten kannst, kommt nun die Lochblende zum Einsatz.

4. Lege diese über den Ausgangspunkt, und dein Partner oder deine Partnerin gibt dir nun genaue Anweisungen.
Der oder die Partner(in) muss nun versuchen, durch exakte Anweisungen, den Computer zum gewünschten Zielpunkt zu führen, wobei nur möglichst wenig Kanten gefahren werden sollen. Wir suchen ja immer noch den kürzesten Weg.
Notiere alle Vorgehensweisen möglichst genau in dein Laborheft.

5. Versuche nun mit deinem Partner oder deiner Partnerin einen Algorithmus zu formulieren, sodass jeder, der den Algorithmus dann ausführen würde, den kürzesten Weg von einem Ausgangspunkt zum Zielpunkt findet. Vergiss dabei nicht auf Ein- und Ausgabe des Algorithmus.

2. Arbeitsblatt: Eulersche Graphen

1. Modelliere den Stadtplan von Königsberg als Graph in dein Laborheft.



2. Ziel der Bewohner von Königsberg war es, einen Spaziergang so zu finden, dass jede Brücke genau einmal überquert wird und man anschließend wieder beim Ausgangspunkt endet.
Versuche durch Hinzufügen oder Wegnehmen von Kanten einen solchen Rundweg zu finden. Probiere solange, bist du einen gefunden hast und zeichnen diesen Graphen neben den ursprünglichen Graphen.

3. Vergleiche nun deinen geänderten Graphen mit dem Graphen der Stadt Königsberg.
Welche Unterschiede fallen dir auf.
Beachte hierbei besonders die Grade der Ecken.

Da Leonhard Euler das Königsberger Brückenproblem löste, ehrte man ihn und nannte Wege, Touren und sogar ganze Graphen nach ihm.

Ein **Eulerweg** ist ein Weg, der duch jede Kante eines zusammenhängenden Graphen genau einmal führt.
Eine **Eulertour** ist ein Eulerweg, der wieder im Ausgangknoten endet.
Ein Graph heißt **eulersch** oder **Eulergraph**, wenn er eine Eulertour enthält.

4. Einen Eulerweg kennt ihr alle! Es ist das Haus vom Nikolaus.
Versuche das Haus vom Nikolaus selbst als Eulerweg zu zeichnen.

5. Wie ihr bestimmt bemerkt habt, funktioniert dies nur, wenn man in einer bestimmten Ecke anfängt. Welche sind das, und wie unterscheiden sie sich von den anderen Ecken?

6. Versuche ein Kriterium anzugeben, wann ein Graph ein Eulergraph ist, also wann er eine Eulertour enthält und wann nicht.

3. Arbeitsblatt: Eulertouren

1. Vervollständige folgende Sätze:

Ich erkenne nun, wann ein Graph eulersch ist, nämlich genau dann, wenn eine .... zu finden ist.
Gibt es eine Eulertour, so haben alle Knoten .... Grad. Auch die Umkehrung gilt: Ist ein Graph .... und haben alle ...... geraden Grad, so ist der Graph ....

2. Betrachte nun noch einmal den Graphen der Stadt Königsberg und begründe, warum es keinen Spaziergang, so wie ihn die Bewohner gerne gehabt hätten, gibt.

Die Frage, welche du dir jetzt wahrscheinlich stellst ist: Wenn ich jetzt aber weiß, dass der vorliegende Graph ein Eulergraph ist, wie kann ich dann eine Eulertour darin finden?

Antwort darauf bietet der sogenannte **Hierholzer-Algorithmus**, er ist auch als Zwiebelschalen-Algorithmus bekannt. Dieser kommt in nur zwei Arbeitsschritten zu einer Eulertour und sieht wie folgt aus:

Zwiebelschalen-Algorithmus

<u>Eingabe:</u> ein Eulergraph

<u>Ausgabe:</u> eine Eulertour

Schritt 1: Wähle einen Startknoten.

Schritt 2: Laufe von diesem Knoten aus entlang noch unmarkierter Kanten und markiere die verwendeten Kanten, solange bis ein Knoten erreicht wird, von dem keine unmarkierte Kante mehr ausgeht. Prüfe, ob alle Kanten des Graphen bereits markiert wurden.
Falls ja: Gehe zu Schritt 3.

Falls nein: Suche einen Knoten, der noch unmarkierte Kanten besitzt und wiederhole Schritt 2.

Schritt 3: Die Eulertour wird nun aus den Kreisen zusammengesetzt:
Gehe entlang des ersten Kreises, bis er einen weiteren Kreis berührt. Folge dem neuen Kreis, bis dieser wiederum an einen nächsten Kreis stößt, und so weiter. Findest du keinen neuen beginnenden Kreis, so gehe den zuletzt begonnenen Kreis zu Ende und dann wieder in den vorherigen hinein. Und so weiter, bis alle Kanten besucht wurden.

3. Zeichnet euch nun gegenseitig einen Eulergraphen auf ein Blatt Papier und findet mit Hilfe des oben beschriebenen Zwiebelschalen-Algorithmus eine Eulertour.

Literaturverzeichnis

[1] Albrecht Beutelspacher and Marc-Alexander Zschiegner. *Diskrete Mathematik für Einsteiger*. Springer Spektrum, 2014.

[2] Noman L. Biggs, E. Keith Lloyd, and Robin J. Wilson. *Graph Theory*. Clarendon Press, Oxford, 1986.

[3] Norman Biggs. *Algebraic Graph Theory*. Cambridge Mathematical Library, Cambridge, 1993.

[4] A.E. Brouwer, A.M. Cohen, and A. Neumaier. *Distance- Regular Graphs*. Springer-Verlag, Berling, 1989.

[5] Franka Miriam Brückler. *Geschichte der Mathematik kompakt*. Springer Spektrum, 2018.

[6] D.M. Cvekovic, M. Doob, and H. Sachs. *Spectra of Graphs*. Academic Press, New York, 1980.

[7] Rechtsinformationssystem des Bundes. `https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10008568&FassungVom=2017-09-01`. zuletzt zugegriffen, am 24.10.2018.

[8] Reinhard Diestel. *Graphentheorie*. Springer Spektrum, Berlin, 2017.

[9] Peter Gritzmann and René Brandenberg. *Das Geheimnis des kürzesten Weges*. Spinger, 2002.

[10] Stephan Hußmann and Brigitte Lutz-Westphal. *Diskrete Mathematik erle-

ben; Anwendungsbasierte und verstehensorientierte Zugänge. Springer Spektrum, 2015.

[11] Ali Kaveh. *Optimal Analysis of Structures by Concepts of Symmetry and Regularity*. Springer, Wien, 2013.

[12] Manfred Kronfellner. Fachdidaktik mathematik. `https://homepage.univie.ac.at/christian.schmeiser/Aspekte-Fachdidaktik.pdf`. zuletzt zugegriffen, am 24.10.2018.

[13] Erich Mayr. Handlungsorientierter unterricht. `https://www.uibk.ac.at/ils/downloads/lernkulturen/handlungsorientierter-unterricht.pdf`. zuletzt zugegriffen, am 24.10.2018.

[14] Manfred Nitzsche. *Graphen für Einsteiger Rund um das Haus vom Nikolaus*. Vieweg+Teubner, 2009.

[15] Kristina Reiss and Gernot Stroth. *Endliche Strukturen*. Springer- Verlag, 2011.

[16] Winfrid Schneeweiss. *Grundbegriffe der Graphentheorie für praktische Anwendungen*. Hüthig, Heidelberg, 1985.

[17] Volker Turau and Christoph Weyer. *Algorithmische Graphentheorie*. De Gruyter Studium, 2015.

[18] Klaus Wagner. *Graphentheorie III*. Wissenschaftsverlag, Mannheim, 1992.

[19] Hansjoachim Walther. *Anwendungen der Graphentheorie*. Vieweg+Teubner Verlag, 1978.

[20] Wikibooks. Algorithmensammlung. `https://de.wikibooks.org/wiki/Algorithmensammlung:_Graphentheorie:_Breitensuche`. zuletzt zugegriffen, am 09.10.2018.