Die Mathematik hinter einem Navigationssystem - Wie Routenplaner ihre Wege berechnen

Theresa Floß
Matrikelnummer: 01624007

Bachelorarbeit

Betreuer: Tim Netzer
Oktober 2023
Innsbruck

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Innsbruck, 12.10.2023

Inhaltsverzeichnis

Abbildungsverzeichnis

Einleitung

Die Geschichte der Navigation reicht weit zurück, Seefahrer nutzen Sterne und natürliche Markierungen um sich auf den Weltmeeren zurecht zu finden. Sie entwickelten Landkarten, Kompasse, Sextanten und andere Werkzeuge, um auch an Land den richtigen Weg zu finden. Dabei war nicht nur die Frage „Wie komme ich zu meinem Ziel?, sondern auch „Wo bin ich? nicht einfach zu beantworten. Jakobsstab und Sextant wurden genutzt um Winkelabstände zu messen um zusammen mit einem Kompass die Position eines Schiffes zu bestimmen. Die Position wurde jedoch nicht nur gemessen, sondern auch berechnet. Die Ausgangsposition und Richtung und Geschwindigkeit der Fortbewegung sind dafür notwenig. Früher nutze man zur Messung der Geschwindigkeit einen sogenannten Logscheit. Ein Holzstück, das an eine langen Leine befestigt wurde, die in regelmäßigen Abständen Knoten hatte. Der Logscheit wurde über Board geworfen und mit Hilfe einer Sanduhr wurde gemessen wie viele Knoten sich in einer vorgegebenen Zeit abwickelten [12]. Allerdings lässt sich dadurch nur die relative Geschwindigkeit gegenüber der Wasseroberfläche messen. Wind und Strömungen sind Ursache für eine Abweichung gegenüber der Geschwindigkeit, die sich auf den Meeresboden bezieht. Um diese Abweichung abschätzen zu können war die Erfahrung der alten Seemänner wichtig. In der Ausbildung und Prüfung des Sportbootführerscheins See, sind auch heute noch Navigationsaufgaben fest verankert. Zwar müssen Geschwindigkeiten nicht mit einem Logscheit gemessen werden, aber die Position muss mit vorgegebenen Werten für Geschwindigkeit, Kurs und Abweichung mitverfolgt werden können [5]. Revolutioniert wurde die Navigation erst mit der Entwicklung des Global Positioning System (GPS), das vom US-Militär entwickelt wurde, ab 1995 voll betriebsfähig und damit auch für die breite Masse nutzbar. Mehr Hintergrund zur Funktionsweise des GPS sowie Ideen zur Behandlung dieses Themas in der Schule finden sich in der Masterarbeit „Das GPS-System, Eine theoretische Annäherung und Ansätze zur Anwendung im Physikunterricht von Carina Homrighausen [9]. Mit der Entwicklung des GPS zogen Navigationsgeräte zunächst in Autos, später auch in Smartphones und Uhren ein. Doch auch wenn das Navigationssystem, dank GPS, weiß wo wir sind, wie berechnet ein Routenplaner eigentlich seinen Weg? Diese Frage ist Kernpunkt der vorliegenden Arbeit. Dazu werden zunächst Vorüberlegungen zur Routenplanung angestellt, anschließend die nötigen Grundlagen der Graphentheorie erläutert und schließlich wird der Algorithmus von Dijkstra, zur Berechnung kürzester Wege, ausführlich erklärt. Dabei ist die Problemstellung so aufgearbeitet, dass ein/e interessierte/r Oberstufen-Schüler*in in der Lage sein sollte, dem Inhalt ohne Probleme zu folgen. Am Ende der Arbeit finden sich darüber hinaus Ideen, wie dieses Thema in der Schule, auch schon in jüngerem Alter, behandelt werden kann. Dabei sollen Schüler*innen gerne dazu motiviert werden ihr Navi einzuschalten aber nicht unbedingt gleichzeitig ihren Kopf aus. Denn dem Navigationssytem blind zu folgen hat schon dazu geführt, dass Menschen an ganz falsche Orte gefahren sind oder ihr Auto in Flüsse gesteuert haben [13].

Vorüberlegungen zur Routenplanung

0.1 Vorüberlegungen

0.1.1 Welche Informationen sind relevant?

Bevor wir die Frage beantworten, wie man am schnellsten von a nach b kommt, sollten wir uns zuerst Gedanken darüber machen, welche Informationen wir benötigen um diese Frage beantworten zu können. Dabei hilft es, an die Zeit vor Routenplanern zurück zu denken. Eine Straßenkarte beinhaltet alle Daten, die wir brauchen. Allerdings nicht nur relevante Daten. Es wird zunächst erläutert, welche der enthaltenen Informationen notwendig sind. Anschließend können wir uns überlegen, wie wir diese darstellen und speichern können.

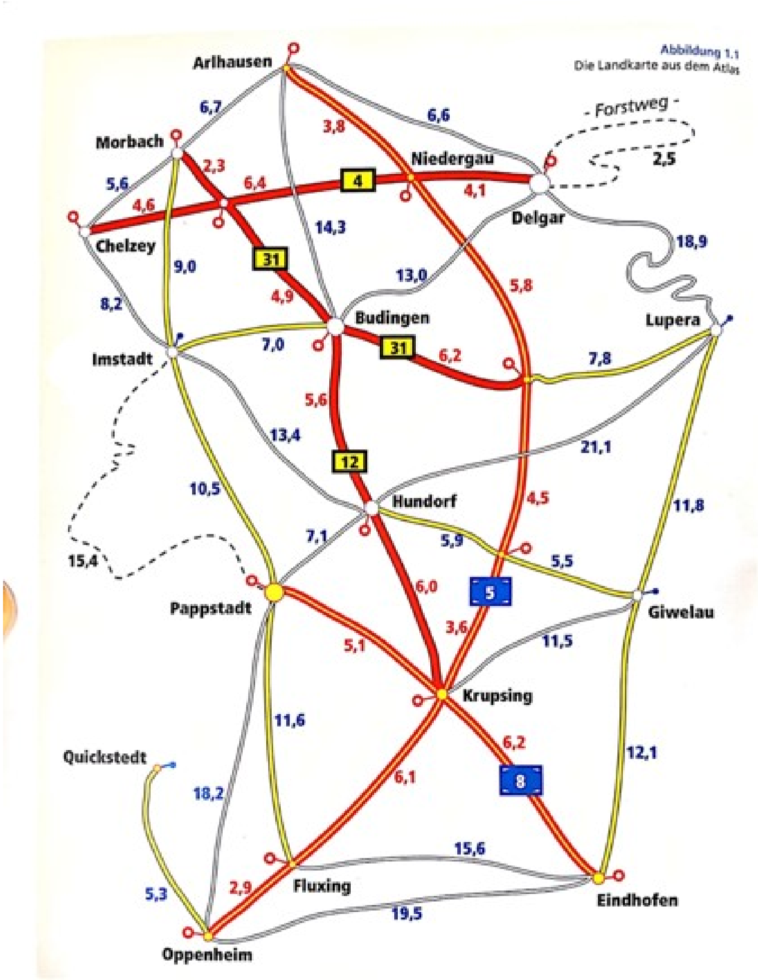


Abbildung 1: Eine Landkarte aus einem Straßenatlas. Quelle: [6]

In Abbildung 1 ist eine Landkarte zu sehen. Aus dieser Karte können wir folgende Informationen herauslesen:

- Posititon oder Lage von Orten
- Ortsnamen
- Straßenarten
- Weglängen
- Kreuzungen
- Größe von Orten
- Straßennamen
- Straßenanfang und -ende
- Straßenverlauf

Häufig können noch deutlich mehr Informationen aus einer Karte gelesen werden. Aber um die kürzeste Strecke zu ermitteln haben wir schon mehr als notwendig. Anhand dieses Beispiels, entscheiden wir welche Informationen zur Berechnung der kürzesten Strecke notwendig sind.

- ✗ Genaue Position eines Ortes oder die Lage zueinander ist für die Berechnung des kürzesten Weges nicht wichtig. Es ist beispielsweise irrelevant, dass Pappstadt 11,6 km nördlich von Fluxing liegt, denn auch wenn Pappstadt 11,6km südlich von Fluxing liegen würde, wäre der direkte Weg über die gelbe Straße der schnellste.
- ✓ Ortsnamen sind tatsächlich wichtig, denn es ist schwierig den kürzesten Weg von Fluxing nach Pappstadt zu finden, wenn weder Fluxing noch Pappstadt irgendwo zu finden sind.
- ✗ Straßenart ist zunächst einmal nicht wichtig. Erst wenn ich beispielsweise wählen kann, ob ich zu Fuß, mit dem Fahrrad oder mit dem Auto unterwegs bin, spielt sie eine Rolle.
- ✓ Weglängen sind zur Berechnung wichtig.
- ✓ Kreuzungen sind von Bedeutung. Dadurch gibt es mehr Möglichkeiten.
- ✗ Größe der Orte spielt keine Rolle.
- ✗ Straßennamen sind unwichtig.
- ✓ Anfangs und Endpunkt einer Straße oder eines Straßenabschnitts sind dagegen schon wichtig.
- ✗ Straßenverlauf wiederum ist egal. Es macht keinen unterschied, ob eine 5km lange Straße kurvig verläuft oder geradeaus.

Damit haben wir entschieden, welche Informationen für die die Berechnung des kürzesten Weges von Bedeutung sind. Im nächsten Kapitel wird überlegt wie man nur die notwendigen Informationen darstellen kann [6].

0.1.2 Modellieren: Von der Karte zum Graph

Nachdem, aus den gegebenen Informationen der Landkarte, die benötigten Informationen herausgefiltert wurden, kann die Karte neu gezeichnet werden. Das Ziel ist dabei nur die nötigen Informationen abzubilden. Dafür werden alle Städtenamen, wie in Abbildung 2 zu sehen ist, abgekürzt und die Orte als Punkte dargestellt. Straßen und Wege sind geradlinige Verbindungen zwischen Anfangs- und Endpunkt, deren Länge angegeben ist. Kreuzungen werden als kleiner Punkt markiert. Es gibt außerdem die Situation, dass sich Straßen auf der Karte kreuzen, eine Auf- oder Abfahrt allerdings nicht möglich ist. In der Realität wäre dies beispielsweise der Fall, wenn eine Straße durch eine Brücke über die Autobahn verläuft. In diesem Fall ist die Möglichkeit wie bei einer Kreuzung die Straße zu wechseln, nicht gegeben. In Abbildung 2 ist dies durch einen kleinen Bogen kenntlich gemacht [6].

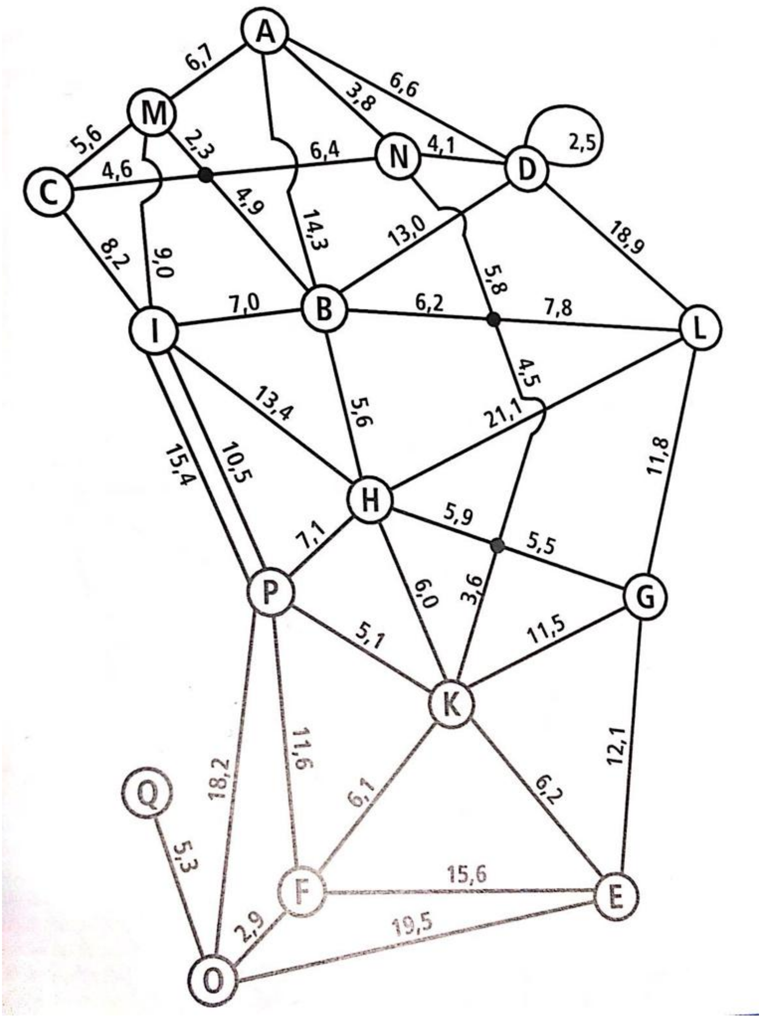


Abbildung 2: Erste Vereinfachung der Landkarte. Quelle: [6]

Abbildung 2 kann noch weiter reduziert werden. Nachdem wir uns für kürzeste Wege interessieren, können wir die längere Straße von I nach P weglassen. Auch der Rundweg bei dem Punkt D ist in diesem Fall uninteressant und kann weggelassen werden. An allen Orten gehen wir davon aus, dass man die Straßen wechseln kann. An Kreuzungen ist dies ebenfalls möglich. Wenn die Kreuzungen also als Orte X,Y,Z aufgefasst werden, können die Punkte vereinheitlicht werden. Nach diesem Schritt sind auch die Bögen, die die "nicht-Kreuzungen" markieren, nicht mehr nötig, denn ein Straßenwechsel kann nur noch bei Punkten stattfinden.

In Abbildung 2 entsprechen die Positionen der Punkte immer noch der Position der Landkarte bzw. der Realität. In dem Abschnitt 0.1.1 wurde diese Information jedoch als nicht relevant befunden. Deshalb kann die Karte ohne Probleme etwas entzerrt werden. Dadurch ist vor allem in Ballungszentren mehr Platz und die relevanten Daten können übersichtlich dargestellt werden [6].

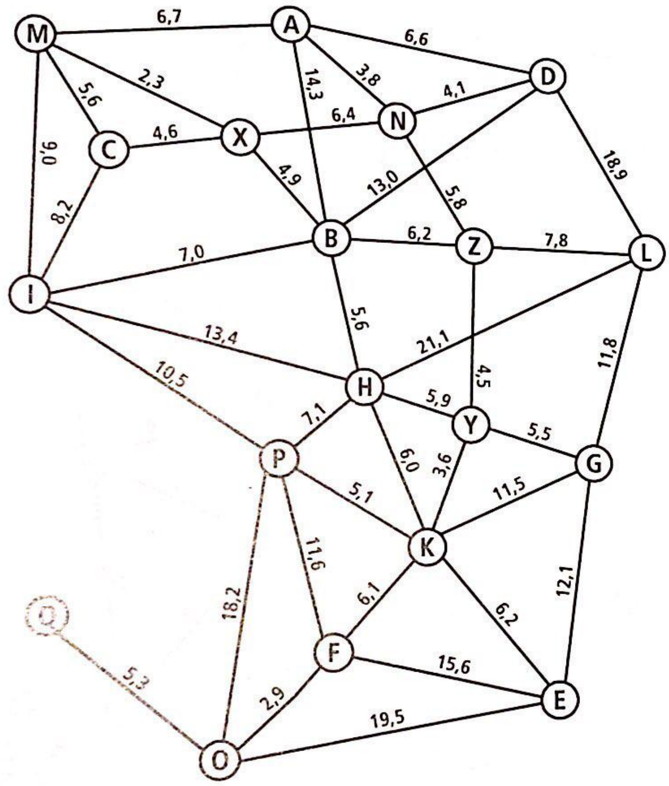


Abbildung 3: Entzerrte und weiter vereinfachte Landkarte. Quelle: [6]

In Abbildung 3 sind nun alle wichtigen Informationen übersichtlich abgebildet. In der Mathematik spricht man hier von einem Graph. Wie genau sich Mathematiker einen Graphen definieren, wird im nächsten Kapitel erläutert.

Fachlicher Einschub: Grundlagen der Graphentheorie

0.2 Definition und Darstellung von Graphen

Im vorigen Kapitel wurde eine Karte systematisch reduziert um einen Graphen zu erhalten. In diesem Kapitel wird behandelt, wie Mathematiker einen Graph definieren, wie sie diesen darstellen und speichern können. Außerdem werden gewichtete und gerichtete Graphen definiert.

Zur Einführung in Bereiche der Graphentheorie werden alle notwendigen Begriffe und Objekte definiert. Diese Grundlagen wurden in enger Anlehnung an das Skript von Professor Pauer, Algebra und diskrete Mathematik, formuliert [14]. Falls Quellen davon abweichen ist dies kenntlich gemacht.

In der Mathematik werden alle Punkte (Orte) als eine Menge betrachtet, die Menge der Ecken. In Abbildung 2 ist die Menge der Ecken folgende:

$$E = \{M, I, Q, C, X, A, B, P, H, O, F, K, E, Y, G, Z, L, N, D\}$$

Ein Element aus der Menge der Ecken, ist eine Ecke des Graphen, bzw. ein Punkt, ein Ort oder eine Kreuzung auf unserer Landkarte. Eine weitere Menge bilden alle Linien (Straßen). Man nennt sie die Menge der Kanten. Die Menge der Kanten ist zweielementig, da eine Kante durch ihren Anfangs- und Endpunkt angegeben wird. In Abbildung 2 ist die Menge der Kanten folgende:

$$\begin{aligned} K = \{&\{M, A\}, \{M, X\}, \{M, C\}, \{M, I\}, \{A, B\}, \{A, N\}, \{A, D\}, \{D, N\}, \{D, B\}, \{D, L\}, \\ &\{C, I\}\{C, X\}, \{X, B\}, \{X, N\}, \{N, Z\}, \{I, B\}, \{I, H\}, \{I, P\}, \{B, Z\}, \{B, H\}, \{Z, Y\}, \\ &\{Z, L\}, \{L, H\}, \{L, G\}, \{H, P\}, \{H, K\}, \{H, Y\}, \{Y, K\}, \{Y, G\}, \{P, O\}, \{P, F\}, \{P, K\}, \\ &\{K, F\}, \{K, E\}, \{O, Q\}, \{O, E\}\} \end{aligned}$$

Die zweielementigen Elemente der Menge der Kanten werden in geschweiften Klammern geschrieben, da die Reihenfolge der Eckpunkte keine Rolle spielt. Das Paar dieser beiden Mengen bildet den Graphen:

$$G = (E, K)$$

Eine mathematische Definition sieht dementsprechend wie folgt aus:

Definition 1. *(Graph)*
Ein Graph ist ein Paar (E, K) von endlichen Mengen, wobei E nicht leer und K eine Menge von zweielementigen Teilmengen von E ist. Die Elemente von E heißen Ecken, Knoten oder Punkte (engl.: vertices), die Elemente von K heißen Kanten (engl.: edges) des Graphen (E, K). Wenn $k := \{a, b\}$ eine Kante des Graphen (E, K) ist, dann heißen die Ecken a und b die Eckpunkte (oder Ecken) von k. In diesem Fall sind die Ecken a und b benachbart.

Beispiel 1. *Ein Graph G=(E,K) besteht aus den beiden Mengen*
$E := \{1, 2, 3, 4\}$ *und*
$K := \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}\}$.

Beispiel 2. *Ein weiteres Beispiel ist der Graph H=(F,L), der aus den beiden Mengen besteht.*
$F := \{1, 2, 3, 4, 5, 6, 7, 8\}$ *und*
$L := \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{5, 6\}, \{5, 7\}, \{6, 7\}, \{6, 8\}\}$.

Zeichnerische Darstellung von Graphen:
Die Darstellung eines Graphen kennen wir schon aus dem Kapitel von der Karte zum Graph. Dabei werden die Ecken als Punkte der Ebene und die Kanten als Strecke zwischen ihren Eckpunkten gezeichnet.

Beispiel 3. *Sehen wir uns an dieser Stelle an, wie der Graph G aus Beispiel 1 aussehen kann:*
$E := \{1, 2, 3, 4\}$ *und*
$K := \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}\}$.

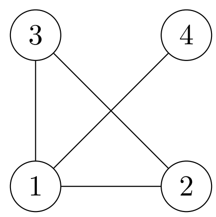


Abbildung 4: Darstellung des Graphen G

Beispiel 4. *Der Graph H aus Beispiel 2 kann wie folgt dargestellt werden:*
$F := \{1, 2, 3, 4, 5, 6, 7, 8\}$
$L := \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{5, 6\}, \{5, 7\}, \{6, 7\}, \{6, 8\}\}$.

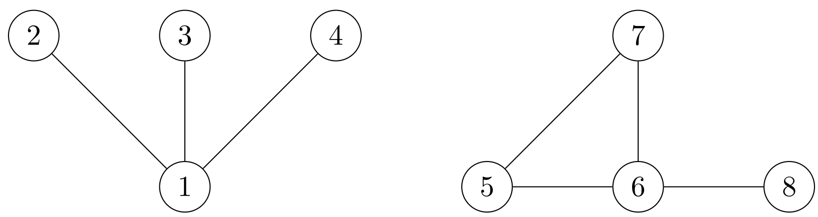


Abbildung 5: Darstellung des Graphen H

Ein Graph muss nicht unbedingt zusammenhängend sein. Beispiel 4 zeigt einen nicht zusammenhängenden Graphen. Wollen wir Karten bzw. die Realität als Graph darstellen, so können nicht zusammenhängende Graphen beispielsweise Inseln darstellen, die keine Straße als Verbindung zu einer anderen Insel oder zum Festland haben.

Beispiel 5. *Der Graph*
({1,2,3,4,5,6,7,8},{{1,2},{2,3},{3,4},{4,1},{5,6},{6,7},{7,8},{8,5},{1,5},{2,6},{3,7},{4,8}}) kann

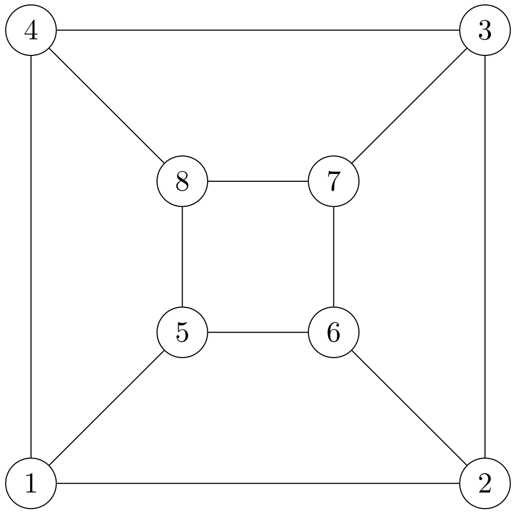


Abbildung 6: Eine mögliche Darstellung des Graphen.

(in der Zeichenenebene) wie folgt dargestellt werden.
Alternativ ist auch folgende Darstellung möglich:

Die Darstellung eines Graphen ist also nicht eindeutig.

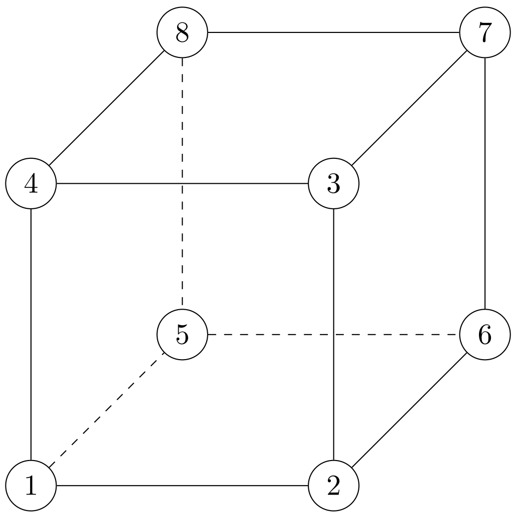


Abbildung 7: Eine andere mögliche Darstellung des Graphen.

Vor allem in Städten kann es sein, dass eine Straße nur in eine Richtung befahrbar ist. Um Einbahnstraßen berücksichtigen zu können ist ein gerichteter Graph von Vorteil.

Definition 2. *(gerichteter Graph)*
Ein gerichteter Graph oder Digraph ist ein Paar (E, K) von endlichen Mengen, wobei E nicht leer und K eine Teilmenge von
$(E \times E) \setminus \{(a, a) \mid a \in K\}$
ist. Die Elemente von E heißen Ecken, die Elemente von K heißen gerichtete Kanten oder $Pfeile$ des Digraphen (E, K). Wenn $k := (a, b)$ eine gerichtete Kante des Digraphen (E, K) ist,

dann heißt a Anfangsecke und b Endecke von k. Die Ecke a ist dann ein Vorgänger von b und die Ecke b ist ein Nachfolger von a.

Zeichnerische Darstellung von Digraphen

Zeichne die Ecken als Punkte der Ebene und die gerichteten Kanten als Pfeile. Der Digraph $(\{1,2,3,4\},\{(1,2),(2,3),(3,4),(4,1),(1,4),(1,3),(4,2)\})$
kann beispielsweise auf folgende zwei Arten dargestellt werden.

Abbildung 8: Zwei mögliche Darstellungen des Digraphen.

Um kürzeste Wege berechnen zu können ist nicht nur die Existenz von Straßen und Kreuzungen wichtig sondern auch die Länge von Strecken.Um diese darzustellen wird eine Bewertungsfunktion genutzt. Diese ordnet jeder Kante ihre Länge zu. Die Leitfrage dieser Arbeit, handelt davon die kürzesten Wege zu berechnen. Viele Routenplaner bieten auch an die schnellsten Wege zu berechnen. In diesem Fall ordnet die Bewertungsfunktion jeder Kante die Zeit zu, die benötigt wird, sie entlang zu fahren. Die Bewertungsfunktion kann auch mit aktuellen Daten gespeist werden, sodass Baustellen, Unfälle, etc. berücksichtigt werden können.

Definition 3. *(Bewerteter Graph)*
$G := (E, K)$ sei ein Graph oder Digraph. Eine Abbildung

$$\omega : K \to \mathbb{R}$$

heißt Bewertungsfunktion von G. Für $k \in K$ heißt die Zahl $\omega(k)$ die Bewertung der Kante k. Das Paar (G, ω) heißt dann bewerteter Graph bzw. Digraph.

Zeichnerische Darstellung von bewerteten Graphen
Ein bewerteter Graph lässt sich darstellen wie ein normaler Graph oder Digraph. Die Bewertung einer Kante wird über, unter oder neben die Kante geschrieben.

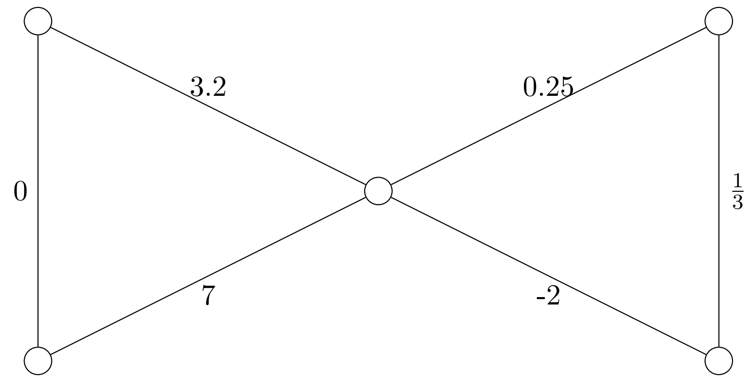


Abbildung 9: Darstellung eines bewerteten Graphen.

Beispiel 6.

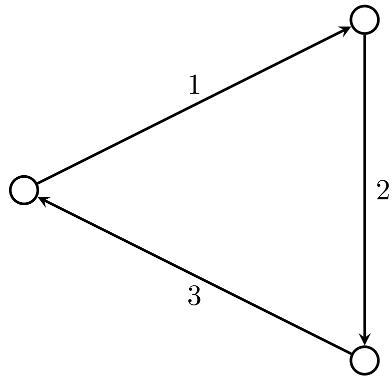


Abbildung 10: Darstellung eines bewerteten Digraphen.

Beispiel 7.

Unter dem Begriff Weg kann man sich intuitiv schon etwas vorstellen. Folgende Definition präzisiert diesen Ausdruck für die Graphentheorie.

Definition 4. *(Kantenfolge, Weg, Kreis) Seien G:=(E,K) ein Graph und n eine positive ganze Zahl. Eine endliche Folge* $a_0, a_1, ..., a_n$ *in E heißt Kantenfolge der Länge n in G, wenn für* $1 \leq i \leq n$ *gilt:* $\{a_{i-1}, a_i\} \in K$*. Sprich wenn alle, in der Folge aufgereihten Kanten, in der Menge der Kanten enthalten sind. Schreibweise:* $[a_0, a_1, ..., a_n]$*.*
Ist $a_0 = a_n$*, ist also der Startpunkt der Kantenfolge gleich dem Endpunkt der Kantenfolge, dann ist die Kantenfolge geschlossen.*
Eine Kantenfolge $[a_0, a_1, ..., a_n]$ *heißt Weg von* a_0 *nach* a_n*, wenn* $a_0, a_1, ..., a_n$ *paarweise verschieden sind. Bei einem Weg wird also kein Punkt zweimal besucht. Eine geschlossene Kantenfolge* $[a_0, ..., a_n]$ *der Länge* ≥ 3 *heißt Kreis, wenn die Ecken, bis auf* $a_0 = a_n$*, paarweise verschieden sind.*

Beispiel 8. *In dem Graphen: ist*

- $[4, 1, 2, 3]$ *ist eine Kantenfolge, und ein Weg von 4 nach 3.*
- $[5, 6, 3]$ *ist eine Kantenfolge, und ein Weg von 5 nach 3.*
- $[1, 2, 5, 6, 3, 2, 5, 4]$ *ist eine Kantenfolge, aber kein Weg*

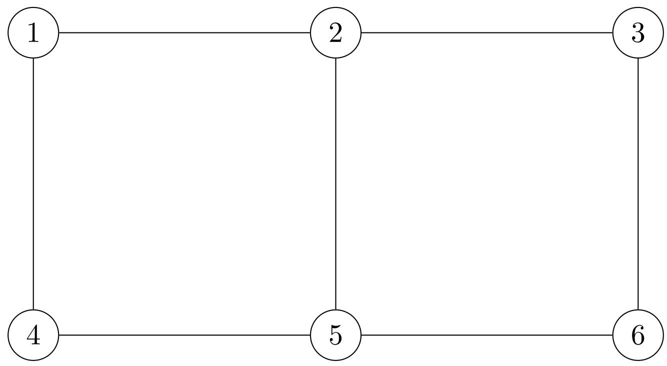


Abbildung 11: Graph

- $[1,2,5,3]$ *ist keine Kantenfolge,*
- $[1,3,2]$ *ist keine Kantenfolge,*
- $[1,2,5,6,3,2,1]$ *eine geschlossene Kantenfolge aber kein Kreis, und kein Weg.*
- $[2,5,6,3,2]$ *ist eine Kantenfolge, kein Weg aber ein Kreis*
- $[1,2,5,6]$ *ist eine Kantenfolge und ein Weg.*

Die Definition für gerichtete Kantenfolgen und gerichtete Wege in einem Digraphen sind fast identisch:

Definition 5. *(gerichtete Kantenfolge und Wege)*
Seien G:=(E,K) ein Digraph und n eine positive ganze Zahl. Eine endliche Folge $a_0, a_1, ..., a_n$*in E heißt gerichtete Kantenfolge der Länge n in G, wenn für* $1 \leq i \leq n$ *gilt:* $(a_{i-1}, a_i) \in K$*. Schreibweise:* $[a_0, a_1, ..., a_n]$*. Ist* $a_0 = a_n$*,dann ist die gerichtetet Kantenfolge geschlossen. Eine gerichtete Kantenfolge* $[a_0, a_1, ..., a_n]$ *heißt gerichteter Weg von* a_0 *nach* a_n*, wenn* $a_0, a_1, ..., a_n$ *paarweise verschieden sind. Eine Ecke* $b \in E$ *ist von* $a \in E$ *aus erreichbar, wenn es einen gerichteten Weg von a nach b gibt.*

Betrachtet man nur einen Teil eines Graphen, so ist die Definition eines Untergraphen hilfreich. Bei der Berechnung der kürzesten Wege nutzen wir Untergraphen, die wir schrittweise erweitern. Diese enthalten nur noch kürzeste Wege. Alle anderen Kanten werden weggelassen.

Definition 6. *(Untergraph)*
G:=(E,K) und G´:=(E´,K´) seien Graphen (oder Digraphen). G ist genau dann ein Untergraph von G´(Schreibweise $G \subseteq G´$*), wenn* $E \subseteq E´$*und* $K \subseteq K´$ *ist. Ein Untergraph G:=(E,K) von G´ ist der von E induzierte Untergraph, wenn K alle (gerichteten) Kanten in K´, deren Ecken in E liegen enthält. Ein Untergraph G:=(E,K) von G´heißt aufspannend, wenn E=E´ ist. Seien* $a \in E´$ *und* $k \in K´$*. Der Untergraph (E´,K´*$\setminus\{k\}$*) heißt der Untergraph von G´, der durch Weglassen der Kanten k entsteht. Der Untergraph von G´, der von E*$\setminus\{a\}$ *induziert wird, heißt der Untergraph von G´, der durch Weglassen einer Ecke entsteht.*

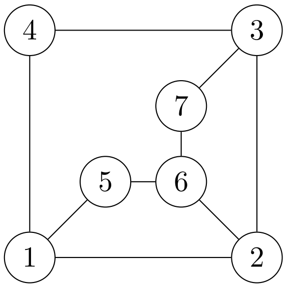


Abbildung 12: Untergraph 1

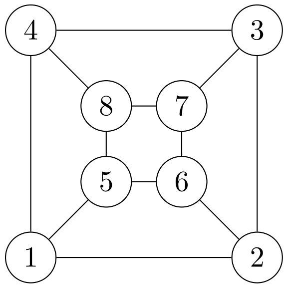


Abbildung 13: Zweiter Graph

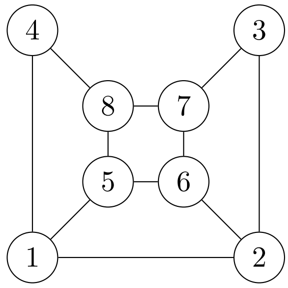


Abbildung 14: Dritter Graph

Beispiel 9. *Der linke Graph, ist ein Untergraph des Graphen in der Mitte, der durch Weglassen einer Ecke, und den zugehörigen Kanten, entstanden ist. Der rechte Untergraph ist durch das Weglassen einer Kante aus dem mittleren Graphen entstanden.*

0.3 Speicherung von Graphen

Zur Speicherung von Graphen gibt es verschiedene Möglichkeiten. Eine Möglichkeit ist die Adjazenzliste. Dabei werden die alle Punkte und alle Kanten in einer Liste wie folgt angegeben.

Definition 7. *(Adjazenzliste) Das n-Tupel*

$$\Big(\big(a_i, \{a_j | \{a_i, a_j\} \in K bzw. (a_i, a_j) \in K\}\big)\Big)_{1 \leq i \leq n}$$

heißt Adjazenzliste von G.

Beispiel 10. *Die Adjazenzliste des abgebildeten Graphen ist:*

$$\big((1, \{2, 3, 4\}), (2, \{1, 4\}), (3, \{1\}), (4, \{1, 2\})\big)$$

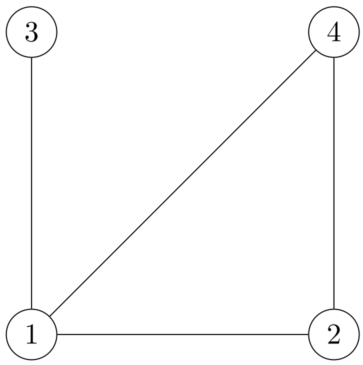


Abbildung 15: Graph

Es wird also zuerst eine Ecke notiert und dahinter alle Knoten zu denen eine Kante geht. So werden alle Ecken und Kanten aufgelistet. Die Kanten sogar doppelt.

Beispiel 11. *Den Graphen, der die Landkarte darstellt*

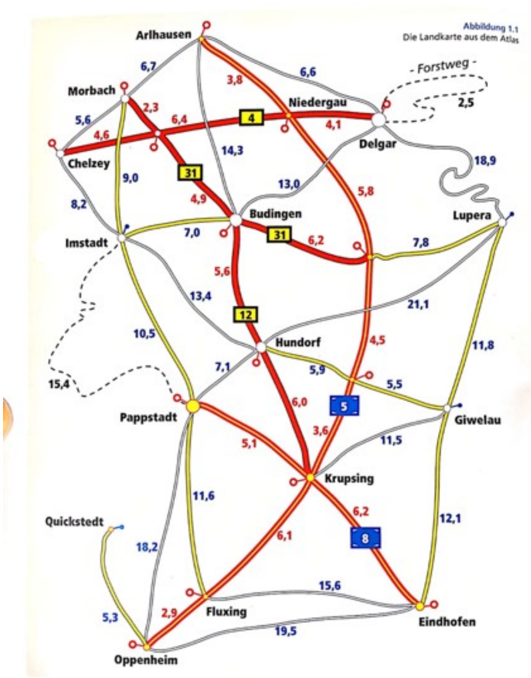


Abbildung 16: Landkarte. Quelle: [6]

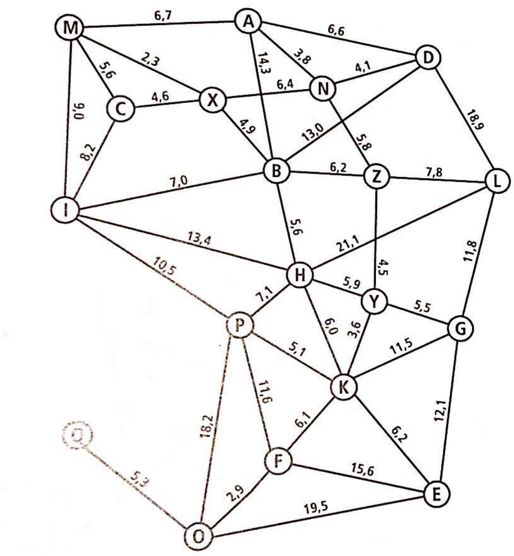


Abbildung 17: Graph der Karte. Quelle: [6]

kann man mit folgender Adjazenzliste speichern:

$$\Big((M,\{I,C,A,X\}),(A,\{M,B,N,D\}),(D,\{A,N,B,L\}),$$
$$(C,\{I,M,X\}),(X,\{C,M,N,B\}),(N,\{X,A,D,Z\}),$$
$$(I,\{M,C,B,H,P\}),(B,\{I,X,A,D,Z,H\}),(Z,\{B,N,L,Y\}),$$
$$(L,\{D,B,H,G\}),(H,\{I,B,L,Y,K,P\}),(Y,\{H,Z,G,K\}),$$
$$(G,\{L,Y,K,E\}),(P,\{I,H,K,F,O\}),(K,\{P,H,Y,G,E,F\}),$$
$$(F,\{P,K,E,O\}),(E,\{G,K,F,O\}),(O,\{Q,P,F,E\}),(Q,\{O\})\Big)$$

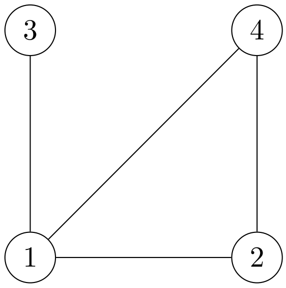
Eine Adjazenzliste ist einfach zu erstellen und einfach zu lesen. Allerdings wird sie bei größeren Graphen schnell unübersichtlich und sehr lang. Außerdem können Informationen wie die Bewertung von Kanten nicht gespeichert werden. Diese brauchen wir aber zur Berechnung kürzester Wege unbedingt.

Eine weitere Möglichkeit die Informationen eines Graphen zu speichern stellt die Adjazenzmatrix oder Nachbarmatrix dar. Dazu brauchen wir geordnete Ecken: $E = \{a_1, ..., a_n\}$.

Definition 8. *(Adjazenzmatrix) Die ganzzahlige* $n \times n - Matrix\ A := A(G) := (A_{ij})_{1 \leq i,j \leq n}$ *mit* $A_{ij} := \begin{cases} 1 & \textit{wenn } \{a_i, a_j\} \in K \textit{ bzw. } (a_i, a_j) \in K \\ 0 & \textit{sonst} \end{cases}$ *heißt Adjazenzmatrix oder Nachbarmatrix von G.*

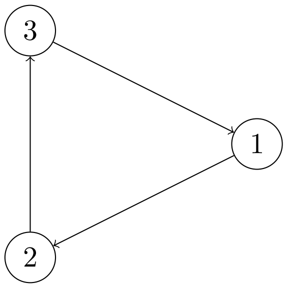
In der Diagonale der Adjazenzmatrix stehen nur Nullen. Die Nachbarmatrix eines Graphen ist symmetrisch, das heißt für $1 \leq i, j \leq n$ gilt $A_{ij} = A_{j,i}$, die eines Digraphen nicht unbedingt.

Beispiel 12. *Adjazenzmatrizen*



$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Abbildung 18: Ein Graph und seine Nachbarmatrix.



$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

Abbildung 19: Ein Digraph und seine Adjazenzmatrix.

s

Um die Adjazenzmatrix des (unbewerten) Graphen der Landkarte zu erstellen müssen die Ecken zuerst geordnet werden. Eine Möglichkeit ist zum Beispiel nach dem Alphabet: $a_1 = A, a_2 = B, a_3 = C, a_4 = D, a_5 = E, a_6 = F, a_7 = G, a_8 = H, a_9 = I, a_{10} = K, a_{11} = L, a_{12} = M, a_{13} = N, a_{14} = O, a_{15} = P, a_{16} = Q, a_{17} = X, a_{18} = Y, a_{19} = Z$

Dann erhält man die 19×19*-Matrix:*

$$\begin{pmatrix}
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}$$

Abbildung 20: Adjazenzmatrix des unbewerteten Graphen der Landkarte.

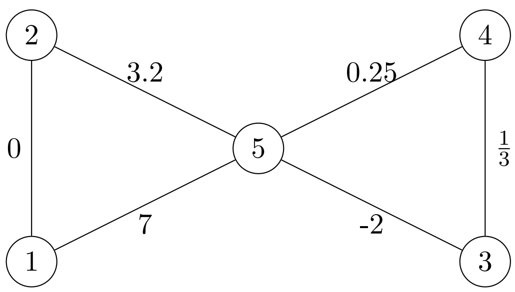
Die Bewertung eines Graphen, kann auch in der Matrix gespeichert werden. Dazu wird die 1 durch die Bewertung der jeweiligen Kante ersetzt, die Nullen in der Diagonale bleiben, die anderen ersetze ich durch ∞. Diese Definition ist in der Literatur nicht einheitlich. Franz Pauer definiert die Matrix eines bewerteten Graphen beispielsweise auch in der Diagonalen mit unendlich. Für den Kontext, für den unsere Matrix genutzt wird finde ich jedoch folgende Definition von [7] passender. Damit sind die Matrixeinträge die Bewertung der jeweiligen Kanten, falls diese existieren und unendlich falls keine existieren. Die Diagonale fülle Ich mit Nullen, es sei denn es existiert ein Rundweg von einem Ort zu sich selbst zurück. Dann kann auch dessen Bewertung hier stehen. Für den kürzesten Weg sind solche Wege jedoch sowieso nicht zielführend.

Definition 9. *(Matrix eines bewerteten Graphen)*
(G, ω) sei ein bewerteter Graph bzw. Digraph. Sei $\overline{\mathbb{R}} := \mathbb{R} \cup \{\infty\}$. Die $n \times n$ Matrix mit Koeffizienten in $\overline{\mathbb{R}}$.

$$A(G,\omega) := (A_{i,j})_1 \leq i, j \leq n \quad mit \quad A_{ij} := \begin{cases} 0 & wenn\ i = j \\ \infty & wenn\ \{a_i, a_j\}\ bzw.\ (a_i, a_j) \notin K \\ \omega(\{a_i, a_j\})\ bzw.\ \omega(a_i, a_j) & wenn\ \{a_i, a_j\}\ bzw.\ (a_i, a_j) \in K \end{cases}$$

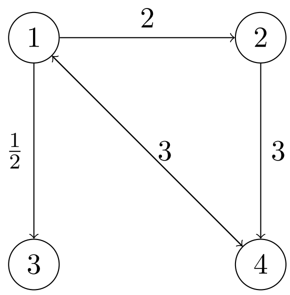
wird Matrix des bewerteten Graphen bzw. Digraphen oder Gewichtsmatrix genannt.

Beispiel 13. *Gewichtsmatrizen*



$$\begin{pmatrix} 0 & 0 & \infty & \infty & 7 \\ 0 & 0 & \infty & \infty & 3.2 \\ \infty & \infty & 0 & \frac{1}{3} & -2 \\ \infty & \infty & \frac{1}{3} & 0 & 0.25 \\ 7 & 3.2 & -2 & 0.25 & 0 \end{pmatrix}$$

Abbildung 21: Ein bewerteter Graph und seine Matrix.



$$\begin{pmatrix} 0 & 2 & \frac{1}{2} & 3 \\ \infty & 0 & \infty & 3 \\ \infty & \infty & 0 & \infty \\ 3 & \infty & \infty & 0 \end{pmatrix}$$

Abbildung 22: Ein Digraph und seine Matrix.

Beispiel 14. *Die bewertete Matrix des Graphen unserer Landkarte sieht folgendermaßen aus:*

$$\begin{pmatrix}
0 & 14,3 & \infty & 6,6 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 3,8 & \infty & \infty & \infty & \infty & \infty & \infty \\
14,3 & 0 & \infty & 13,0 & \infty & \infty & \infty & 5,6 & 7,0 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 4,9 & \infty & 6,2 \\
\infty & \infty & 0 & \infty & \infty & \infty & \infty & \infty & 8,2 & \infty & \infty & 5,6 & \infty & \infty & \infty & \infty & 4,6 & \infty & \infty \\
6,6 & 13,0 & \infty & 0 & \infty & \infty & \infty & \infty & \infty & \infty & 18,9 & \infty & 4,1 & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 0 & 15,6 & 12,1 & \infty & \infty & 6,2 & \infty & \infty & \infty & 19,5 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 15,6 & 0 & \infty & \infty & \infty & 6,1 & \infty & \infty & \infty & 2,9 & 11,6 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 12,1 & \infty & 0 & \infty & \infty & 11,5 & 11,8 & \infty & \infty & \infty & \infty & \infty & \infty & 5,5 & \infty \\
\infty & 5,6 & \infty & \infty & 12,1 & \infty & \infty & 0 & 13,4 & 6,0 & 21,1 & \infty & \infty & \infty & 7,1 & \infty & \infty & 5,9 & \infty \\
\infty & 7,0 & 8,2 & \infty & \infty & \infty & \infty & 13,4 & 0 & \infty & \infty & 9,0 & \infty & \infty & 10,5 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & 6,2 & 6,1 & 11,5 & 6,0 & \infty & 0 & \infty & \infty & \infty & \infty & 5,1 & \infty & \infty & 3,6 & \infty \\
\infty & \infty & \infty & 18,9 & \infty & \infty & 11,8 & 21,1 & \infty & \infty & 0 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 7,8 \\
6,7 & \infty & 5,6 & \infty & \infty & \infty & \infty & \infty & 9,0 & \infty & \infty & 0 & \infty & \infty & \infty & \infty & 2,3 & \infty & \infty \\
3,8 & \infty & \infty & 6,6 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty & \infty & \infty & 6,4 & \infty & 5,8 \\
\infty & \infty & \infty & \infty & 19,5 & 2,9 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & 18,2 & 5,3 & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 11,6 & \infty & 7,1 & 10,5 & 5,1 & \infty & \infty & \infty & 18,2 & 0 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 5,3 & \infty & 0 & \infty & \infty & \infty \\
\infty & 4,9 & 4,6 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 2,3 & 6,4 & \infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 5,5 & 5,9 & \infty & 3,6 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & 4,5 \\
\infty & 6,2 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 7,8 & \infty & 5,8 & \infty & \infty & \infty & \infty & 4,5 & 0
\end{pmatrix}$$

Abbildung 23: Bewertete Matrix des Graphen der Landkarte.

Kürzeste Wege berechnen

Die Leitfrage dieser Arbeit lautet: Welche Algorithmen und mathematischen Verfahren gibt es, die mit Routenplanung in Verbindung stehen und wie funktionieren diese? An dieser Stelle haben wir nun ausreichend Grundlagen geschaffen, um zu beginnen die Frage zu beantworten.

0.4 Heuristische Herangehensweise

Bei der Routenplanung stellt sich die Frage, wie wir am kürzesten von A nach B kommen?

Menschen haben oft eine intuitive Methode, um kürzeste Wege zu finden. Bei der Planung einer Route denken sie darüber nach, wie sie von einem Startpunkt zu einem Zielpunkt gelangen können. Dabei berücksichtigen sie verschiedene Faktoren und treffen Entscheidungen, um den vermeintlich kürzesten Weg zu wählen. Diese Entscheidungen basieren auf ihrer Erfahrung, ihrem Wissen über die Umgebung und persönlichen Präferenzen. Menschen versuchen oft, direkt in Richtung des Zielpunkts zu starten und gegebenenfalls auf größere Straßen zu wechseln, um lange Strecken effizienter zurückzulegen. Sie vermeiden in der Regel, in Kreisen zu fahren und sinnlose Umwege zu machen [6].

Während die heuristische Herangehensweise beim Menschen auf intuitive Überlegungen basiert, nutzen Computer für die Berechnung kürzester Wege spezifische Algorithmen und mathematische Verfahren. Ein Ansatz ist die Brute-Force-Methode, die wir im nächsten Abschnitt genauer betrachten werden. Bevor wir uns diese ansehen, werde ich an dieser Stelle noch exakt definieren, was wir als kürzesten Weg verstehen.

Definition 10. *(Länge eines Weges) Die Länge eines Weges bzw. eines gerichteten Weges ist die Summe der Bewertungen seiner Kanten. Der Abstand $d_G(a,b)$ von einer Ecke a zu einer Ecke b in G ist die kleinste Länge eines Weges bzw. gerichteten Weges von a nach b, falls ein solche existiert und sonst ∞. Ein Weg bzw. gerichteter Weg $[a = a_0, ..., a_j = b]$ ist ein kürzester Weg von a nach b, wenn*

$$\sum_{i=0}^{j-1} \omega(\{a_i, a_{i+1}) = d_G(a,b) \quad bzw. \quad \sum_{i=0}^{j-1} \omega((a_i, a_i + 1)) = d_G(a,b)$$

ist [14].

0.5 Die Brute-Force Methode

Bei unserer heuristischen Herangehensweise haben wir einfache Wege ausprobiert und miteinander verglichen. Das gleiche macht auch der Rechner bei der Brute-Force Methode. Er rechnet alle möglichen Wege aus und sucht daraus den kürzesten aus. Während wir Menschen bei der Berechnung der Wege, absurde Möglichkeiten direkt weglassen, kann ein Computer das nicht entscheiden. Er berechnet wirklich alle Möglichkeiten. Brute-Force Methode heißt sie deshalb, weil dafür eine hohe Rechenleistung notwendig ist. Größere Aufgaben können nur mit extrem hoher Rechenleistung gelöst werden, für manche ist ein normaler Computer gar nicht mehr in der Lage, da es zu viele Möglichkeiten gibt und die Berechnung aller Wege zu viel Zeit in Anspruch nimmt. Wie viele mögliche Wege es von A nach B gibt, hängt von der Größe des Graphen ab. Zur besseren Vorstellung vergleichen wir in folgendem Beispiel die Anzahl der möglichen Wege von A nach B. Für diesen Vergleich nutzen wir vollständige Graphen. Das sind Graphen, in denen jeder Knoten mit jedem Knoten verbunden ist.

Beispiel 15. *In einem Graphen mit zwei Knoten gibt es nur eine Möglichkeit um von A nach B zu gelangen.*

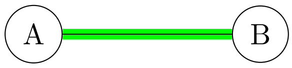


Abbildung 24: Alle Möglichkeiten von A nach B in einem vollständigen Graphen mit 2 Knoten.

Bei einem Graphen mit drei Knoten gibt es zwei Möglichkeiten um von A nach B zu gelangen.

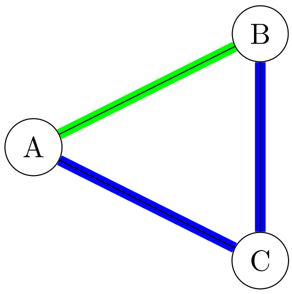


Abbildung 25: Alle Möglichkeiten von A nach B in einem vollständigen Graphen mit 3 Knoten.

Bei vier Knoten muss schon mehr überlegt werden. B kann von drei Knoten erreicht werden. Ein Weg ist direkt von A. Dazu kommen die Wege, die über C gehen. C kann von A auf 2 Wegen erreicht werden (ohne über B zu gehen). Und schließlich noch die Wege über D. D kann von A auch über 2 Wege erreicht werden. Es gibt also 1+2+2=5 mögliche Wege.

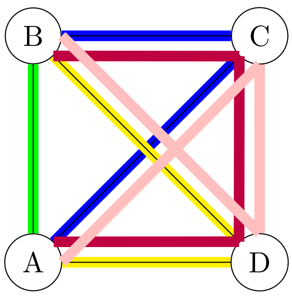


Abbildung 26: Alle Möglichkeiten von A nach B in einem vollständigen Graphen mit 4 Knoten.

Bei fünf Knoten sind die Überlegungen ähnlich. B kann von A,C,D und E erreicht werden. Der direkte Weg von A ist die erste Möglichkeit.

Weitere Möglichkeiten gehen über den Knoten C. Von A nach C gibt es, ohne über B zu gehen 5 möglich Wege. Das haben wir uns eben schon überlegt...

Genauso viele Möglichkeiten gibt es über die Knoten D und E. Wir erhalten also 1+5+5+5=16 Möglichkeiten. Diese im Graph darzustellen ist schon sehr unübersichtlich.

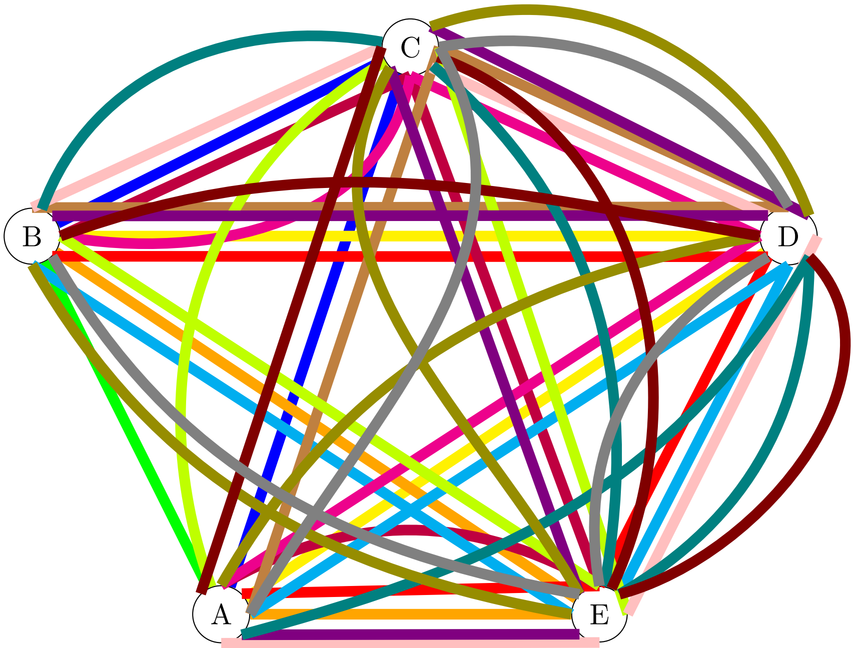


Abbildung 27: Alle Möglichkeiten von A nach B in einem vollständigen Graphen mit 5 Knoten.

Bei 6 Knoten versuche ich gar nichtmehr alle Wege zu zeichen. Denn nach den gleich Überlegungen wie schon zuvor erhalten wir 1+16+16+16+16=65 Möglichkeiten. Ich hätte nichteinmal ausreichend verschiedene Farben um diese Möglichkeiten zu zeichnen.

Bei 7 Knoten sind es dann schon 1+65+65+65+65+65=326 mögliche Wege.
*Bei 8 Knoten erhält man 1+6*326=1957 mögliche Wege.*
*Bei 9 Knoten sind es 1+7*1957=13701 Möglichkeiten.*
*Bei 10 Knoten sind es 1+8*13701=109609 Möglichkeiten.*
usw.

Anzahl der Ecken	**Anzahl der möglichen Wege von A nach B**
2	1
3	2
4	5
5	16
6	65
7	326
8	1957
9	13701
10	109609

Tabelle 1: Anzahl an Möglichen Wegen von A nach B in einem vollständigem Graphen.

Bei der Anzahl der Möglichen Wege können wir hier von einer kombinatorischen Explosion sprechen. Schon bei relativ kleinen Werten gehen die Anzahl der Möglichkeiten ins unermessliche. Deshalb eignet sich die Brute-Force Methode nur für sehr einfache Probleme. Bei schwierigeren oder komplexeren Aufgaben wird ein besserer Algorithmus benötigt. Aus dem einfachen Grund, da sonst die Rechenzeit zu lange dauert oder gar nicht mehr möglich ist. [11]

0.6 Der Algorithmus von Dijkstra

Weil sich die Brute Force Methode für größere Probleme nicht mehr eignet, hat sich der Mathematiker und Programmierer Edsger Wybe Dijkstra einen besseren Algorithmus überlegt. Diesen veröffentlichte er 1959 in einem dreiseitigen Artikel.

Um den Algorithmus zu beschreiben, werde ich zunächst noch einmal das Problem genau definieren, anschließend erläutern welche Schritte bei dem Algorithmus nötig sind und die richtige Reihenfolge festlegen. Ich werde den Algorithmus mit Hilfe eines Pseudocodes präzise formulieren und anschließend analysieren und Optimierungsmöglichkeiten zeigen.

0.6.1 Problemstellung

Um das Problem auf den Punkt zu bringen reicht im Grunde die Frage, “Wie komme ich am kürzesten von A nach B?” Welche Informationen ich dazu benötige haben wir bereits im Kapitel Vorüberlegungen zur Routenplanung diskutiert. Wie ich notwendige Daten speichern und darstellen kann ist wird im Kapitel Grundlagen der Graphentheorie dargestellt. Ziel des folgenden Algorithmus ist es, einen Weg von A nach B zu finden, der die kürzeste Distanz hat.

0.6.2 Die Idee des Algorithmus von Dijkstra

Der Algorithmus von Dijkstra berechnet nicht nur den kürzesten Weg von A nach B aus, sondern den kürzesten Weg von A zu allen anderen Punkten . Dijkstras Algorithmus fällt in die Kategorie der sogenannten “greedy” Algorithmen. In der Informatik ist dies ein häufig auftretender Algorithmustyp, bei dem schrittweise derjenige Zustand gewählt wird, der in einem spezifischen Sinne das vielversprechendste Ergebnis verspricht. Der Algorithmus handelt daher in einer “hungrigen” Art und Weise, indem er zu jedem Zeitpunkt nach der besten Option greift. [1]

Bevor der Algorithmus mit Hilfe eines Pseudocodes formuliert wird, beschreiben wir die Schritte an einem Beispiel. Wir starten, indem wir M, die Menge der besuchten Knoten definieren. Diese Menge enthält am Anfang nur den Startpunkt A. Bei jeden Schritt wird der Algorithmus nun einen weiteren Punkt auswählen und der Menge hinzufügen. Der Algorithmus endet, wenn der Zielknoten B in der Menge enthalten ist. Alternativ kann der Algorithmus auch erst beendet werdem, wenn alle Knoten in der Menge M enthalten sind.

Zu jedem Schritt gehört auch, dass die Distanz aller Knoten zum Startpunkt aktualisiert und notiert wird. Die Distanz ist die kürzeste Entfernung zum Startpunkt und entscheidend für die Reihenfolge in der die Punkte zur Menge der besuchten Punkte hinzugefügt werden. Außerdem wird die Vorgängerecke notiert, sodass später der kürzeste Weg nachvollzogen werden kann. Aber diese Schritte nun noch etwas ausführlicher. Der Algorithmus startet am Startpunkt A, von dort trifft er seine "hungrige" Entscheidung. Er berechnet die Distanzen zu den Nachbarknoten von A, notiert diese und wählt den Knoten mit der kürzesten Distanz aus. Die Ecke wird zusammen mit ihrer Entfernung, sowie dem Vorgängerpunkt notiert und der Menge M, der Menge der besuchten Ecken hinzugefügt. In den nächsten Schritten werden nicht nur Punkte betrachtet, die direkt mit A verbunden sind, sondern alle, die eine Verbindung mit einem Knoten aus der Menge der besuchten Knoten haben. Für neu erreichbare Knoten werden die Distanzen berechnet und notiert. Bei den bereits erreichbaren wird überprüft, ob es nun eventuell einen kürzeren Weg gibt, in diesem Fall wird die Distanz aktualisiert. In jedem Schritt wählt der Algorithmus den Knoten mit der kürzesten Entfernung zum Startpunkt A aus. Die kürzeste Entfernung ist dabei immer die Summe der bewerteten Kanten des kürzesten Weges [1].

Sehen wir uns die Vorgehensweise nun Schritt für Schritt an einem Beispiel an.

Beispiel 16. *Wie geht der schnellste Weg von A nach B?*

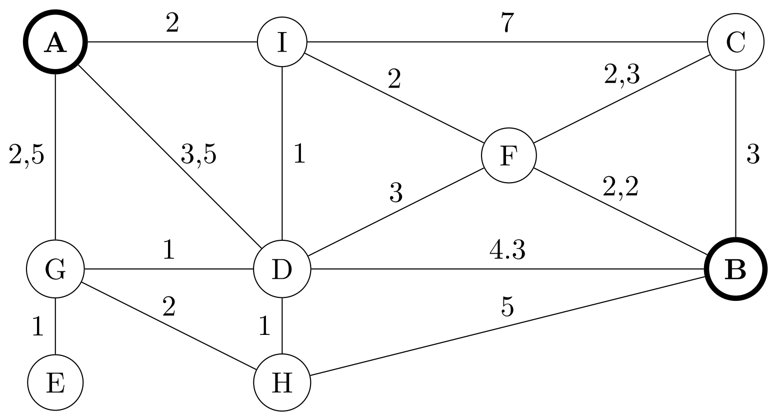


Abbildung 28: Wie finde ich den kürzesten Weg von A nach B?

Schritt 1

Ich füge den Startpunkt A zur Menge der besuchten Knoten hinzu und markiere ihn sowohl im Graphen als auch in der Tabelle in der Farbe Orange. Zudem halte ich die Entfernung zu Punkt A fest, in diesem Fall 0, sowie den Vorgänger, über den ich den aktuellen Knoten erreicht habe, der in diesem Fall ebenfalls A ist.

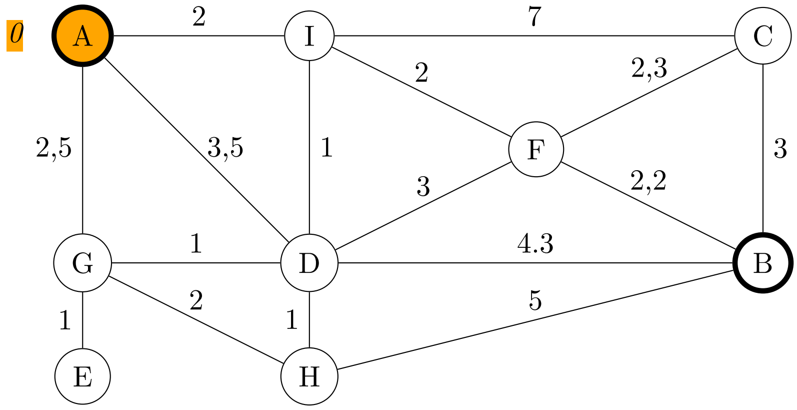


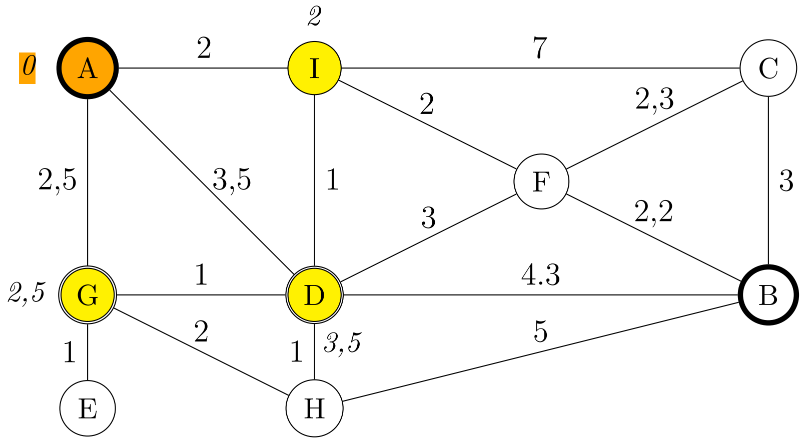
Abbildung 29: Schritt 1, Knoten A ist in der Menge der besuchten Knoten, seine Distanz ist notiert.

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	∞	∞	∞	∞	∞	∞	∞
Vorgänger	A								

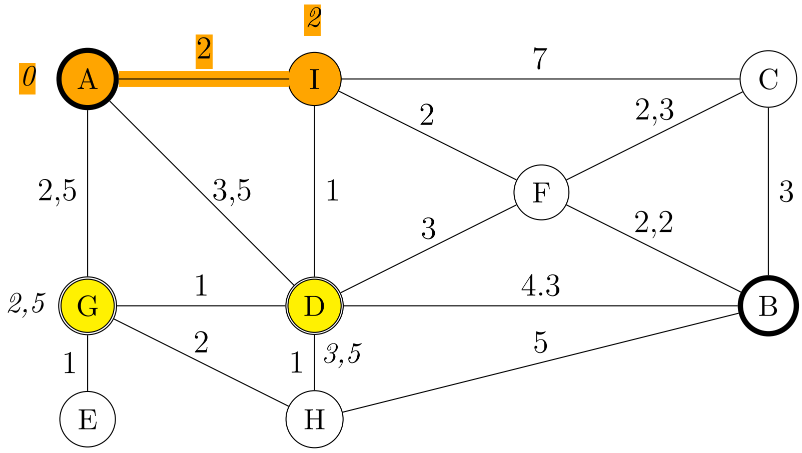
$Menge\ der\ besuchten\ Ecken := \{A\}$

Schritt 2

Nun wählt der Algorithmus den nächsten Knoten aus. Dabei betrachtet er alle direkten Nachbarn der aktuellen Menge der besuchten Knoten. Die Menge der besuchten Knoten beinhaltet zu diesem Zeitpunkt nur den Knoten A. Die Nachbarknoten sind die Knoten G,D und I, zu diesen berechnet er die Distanzen: I (Entfernung 2), G (Entfernung 2,5) und D (Entfernung 3,5) zur Auswahl. Die Distanzen werden notiert (im Graph neben dem Knoten und in der Tabelle. Dijkstra wählt den Knoten mit der kürzesten Entfernung aus. Das ist in diesem Fall I. Damit hat er den kürzesten Weg zum Knoten I gefunden. Die Entfernung von 2 sowie der vorherige Knoten werden notiert. Der Knoten I wird zur Menge der besuchten Knoten hinzugefügt.



Ausganglage für Schritt 2, der Algorithmus berechnet die Distanzen zu erreichbaren Knotenpunkten und notiert diese



Ende Schritt 2, Dijkstra hat I ausgewählt und der Menge der besuchten Knoten hinzugefügt

Abbildung 30: Schritt 2

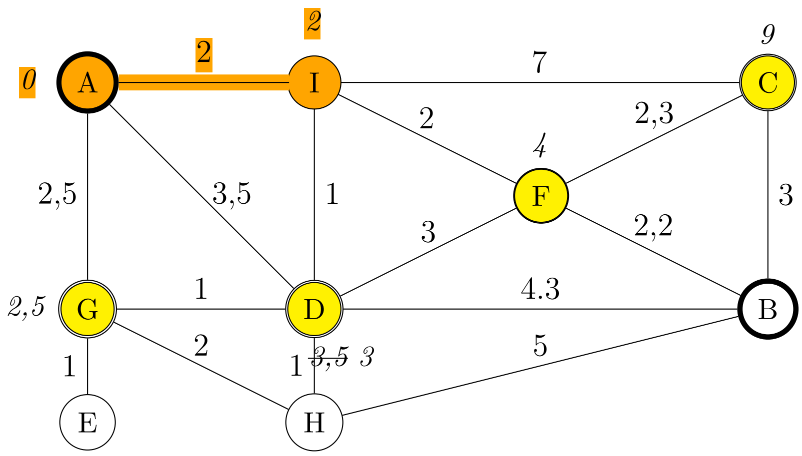
	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	∞	3,5	∞	∞	2,5	∞	2
Vorgänger	A			A			A		A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	∞	3,5	∞	∞	2,5	∞	2
Vorgänger	A			A			A		A

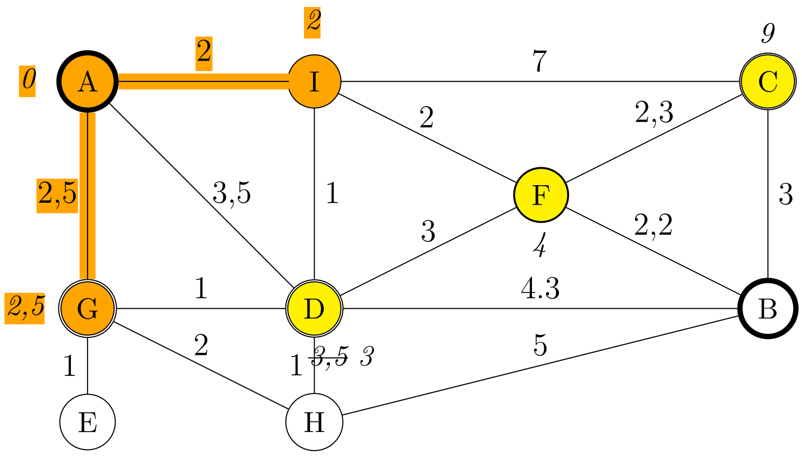
$$Menge\ der\ besuchten\ Ecken := \{A, I\}$$

Schritt 3

Aus den Nachbarn der Knoten A und I, die noch nicht in der Menge der aktuellen Knoten enthalten sind wählt der Algorithmus nun den nächsten Knoten aus, den er am schnellsten erreichen kann. Dabei werden nicht nur die Knoten berücksichtigt, die direkt von A erreichbar sind, sondern auch diejenigen, die über I erreicht werden können. Zur Auswahl stehen die gelb markierten Knoten G, D, F und C. Außerdem wird überprüft, ob die bereits berechneten Distanzen immer noch die kürzesten sind. Für den Knoten G gibt es immer noch nur einen Weg. Der Knoten D kann nun aber auch über den Knoten I erreicht werden. Dabei ist die Distanz nur noch 3. Wir haben also einen kürzeren Weg gefunden und müssen dies notieren. Die Entfernung zu einem Knoten ergibt sich als die Summe der Bewertungen entlang des Pfades. In diesem Schritt stehen die Knoten G (Entfernung 2,5), D (Entfernung von A direkt: 3,5 oder über I: 2 + 1 = 3), F (Entfernung über I: 2 + 2 = 4) und C (Entfernung über I: 2 + 7 = 9) zur Auswahl. Dijkstra wählt den Knoten mit der kürzesten Entfernung aus, in diesem Fall den Knoten G. Somit hat Dijkstra den kürzesten Weg von A nach G gefunden. Die Entfernung von 2,5 sowie der vorherige Knoten A (für den kürzesten Weg) werden notiert. Der Knoten I wird zur Menge der besuchten Knoten hinzugefügt.



Schritt 3, orange die Menge der besuchten Knoten, gelb die Auswahl der nächsten Knoten



Ende Schritt 3

Abbildung 31: Schritt 3

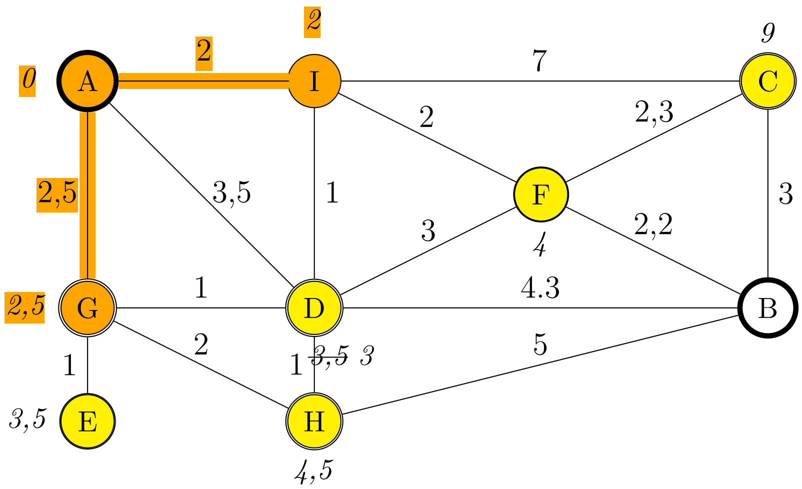
	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	2+7=9	~~3,5~~ 2+1=3	∞	2+2=4	2,5	∞	2
Vorgänger	A		I	I		I	A		A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	2+7=9	~~3,5~~ 2+1=3	∞	2+2=4	2,5	∞	2
Vorgänger	A		I	I		I	A		A

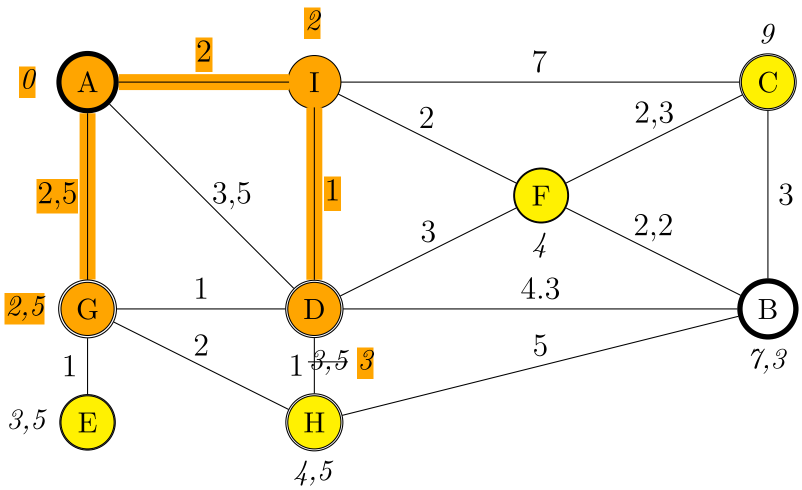
$$Menge\ der\ besuchten\ Ecken := \{A, I, G\}$$

Schritt 4

Aus den Nachbarn der aktuellen Menge der besuchten Knoten, die derzeit die Knoten A, I und G enthält, wählt der Algorithmus nun den nächsten Knoten aus, den er am schnellsten erreichen kann. Es werden alle Knoten betrachtet, die direkt von A oder über G oder I erreichbar sind. Die Entfernung zu den Knoten ergibt sich wie in den vorherigen Schritten, als die Summe der Bewertungen entlang des Pfades. In diesem Schritt stehen die Knoten E (Entfernung über G: 2,5+1=3,5), H (Entfernung über G: 2,5+2=4,5) D (Entfernung von A direkt: 3,5 oder über I: 2 + 1 = 3 oder über G: 2,5+1=3,5), F (Entfernung über I: 2 + 2 = 4) und C (Entfernung über I: 2 + 7 = 9) zur Auswahl. Wieder wählt Dijkstra den Knoten mit der kürzesten Entfernung aus, in diesem Fall den Knoten D. Dijkstra hat den kürzesten Weg von A nach D gefunden. Er geht von A über I nach D. Die Entfernung von 2+1=3 sowie der Weg von A über I werden notiert. Der Knoten D wird zur Menge der besuchten Knoten hinzugefügt.



Anfang Schritt 4



Ende Schritt 4

Abbildung 32: Schritt 4

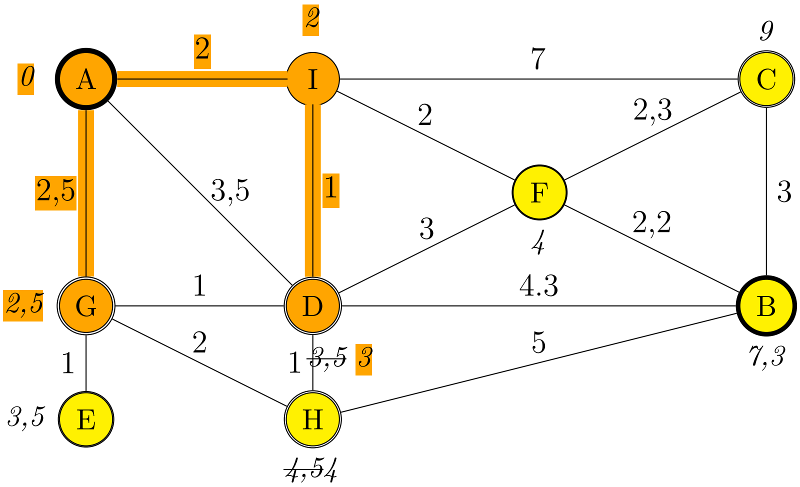
	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2,5+2=4,5	2
Vorgänger	A		I	I	G	I	A	G	A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	∞	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2,5+2=4,5	2
Vorgänger	A		I	I	G	I	A	G	A

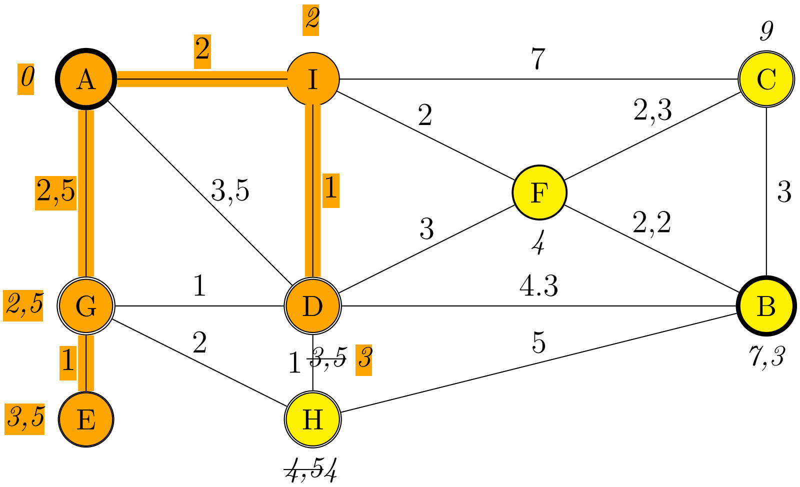
$$Menge\ der\ besuchten\ Ecken := \{A, I, G, D\}$$

Schritt 5

Der Algorithmus wiederholt das Vorgehen der vorherigen Schritte. Er betrachtet die Punkte E, H, F, B und C, die alle über Knoten aus der aktuellen Menge der besuchten Knoten erreicht werden können. Der Algorithmus berechnet die Entfernungen zu diesen Punkten: E (Entfernung über G: 2,5 + 1 = 3,5), H (Entfernung über G: 2,5 + 2 = 4,5 oder über D: 2 + 1 + 1 = 4), B (Entfernung über D: 2 + 1 + 4,3 = 7,3), F (Entfernung über I: 2 + 2 = 4 oder über D: 2 + 1 + 3 = 6) und C (Entfernung über I: 2 + 7 = 9). Dabei wählt der Algorithmus den kürzesten Weg, in diesem Fall den Weg zu E.
Dijkstra hat somit den kürzesten Weg von A nach E gefunden, der über den Knoten G verläuft. Die Entfernung von 2,5 + 1 = 3,5 sowie der Pfad von A über G werden notiert. Der Knoten E wird zur Menge der besuchten Knoten hinzugefügt.



Anfang Schritt 5



Ende Schritt 5

Abbildung 33: Schritt 5

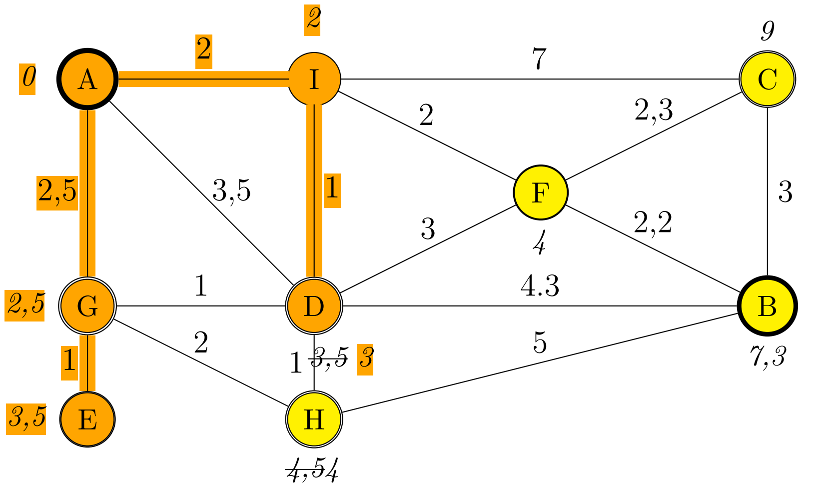
	A	B	C	D	E	F	G	H	I
Distanz zu A	0	2+1+4,3=7,3	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+1=4	2
Vorgänger	A	D	I	I	G	I	A	D	A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	2+1+4,3=7,3	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+1=4	2
Vorgänger	A	D	I	I	G	I	A	D	A

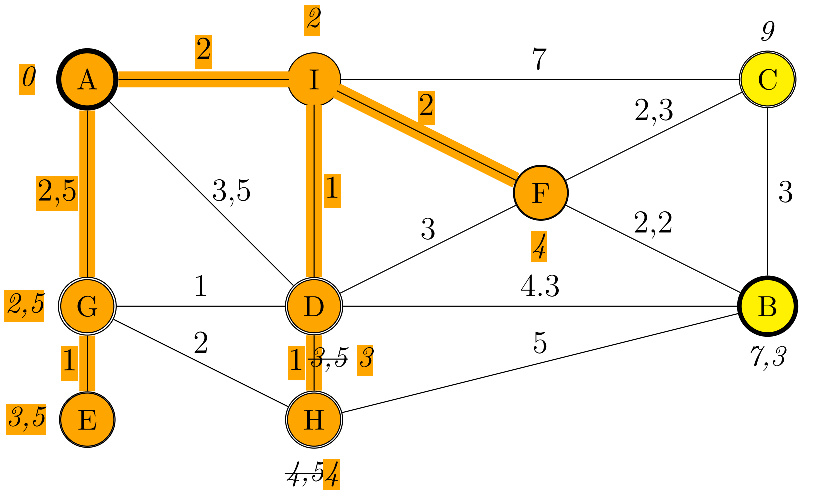
$$Menge\ der\ besuchten\ Ecken := \{A, I, G, D, E\}$$

Schritt 6

Der Algorithmus folgt erneut dem gleichen Vorgehen. Nun betrachtet er die Punkte H, F, B und C, die alle über Knoten aus der aktuellen Menge der besuchten Knoten erreicht werden können. Der Algorithmus berechnet die Entfernungen zu diesen Punkten: B (Entfernung über D: 2 + 1 + 4,3 = 7,3), F (Entfernung über I: 2 + 2 = 4 oder über D: 2 + 1 + 3 = 6), H (Entfernung über G: 2,5 + 2 = 4,5 oder über I und D: 2 + 1 + 1 = 4) und C (Entfernung über I: 2 + 7 = 9).
Nun haben wir den Fall, dass zwei Punkte die gleiche kürzeste Entfernung haben. Sowohl der Knoten F als auch der Knoten H haben eine Entfernung von 4. Es spielt keine Rolle, welchen wir zuerst auswählen, da es keinen kürzeren Weg geben wird. Daher können wir entweder zuerst den einen oder den anderen auswählen oder beide gleichzeitig. So werden wir es handhaben. Für den Knoten F notieren wir die Entfernung von 2 + 2 = 4 und den Weg von A über I. Für den Knoten H notieren wir die Entfernung von 2 + 1 + 1 = 4 und den Weg von A über I und D nach H. Wir fügen sowohl den Knoten F als auch den Knoten H der Menge der besuchten Knoten hinzu.



Anfang Schritt 6



Ende Schritt 6

Abbildung 34: Schritt 6

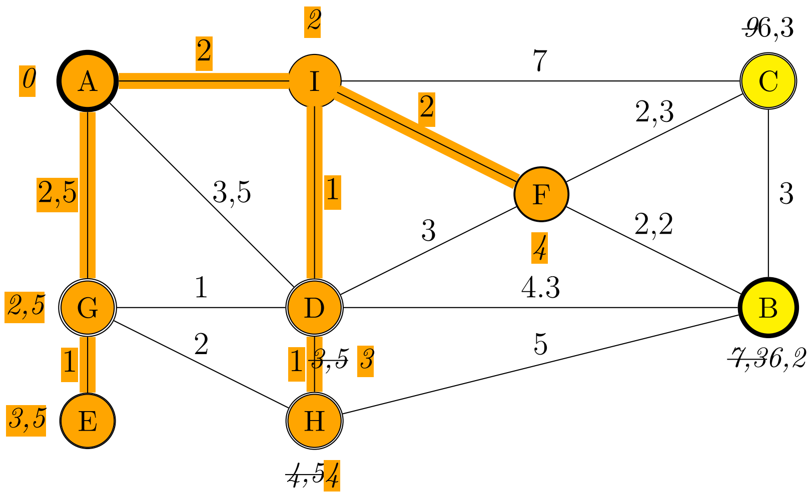
	A	B	C	D	E	F	G	H	I
Distanz zu A	0	2+1+4,3=7,3	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+1=4	2
Vorgänger	A	D	I	I	G	I	A	D	A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	2+1+4,3=7,3	2+7=9	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+1=4	2
Vorgänger	A	D	I	I	G	I	A	D	A

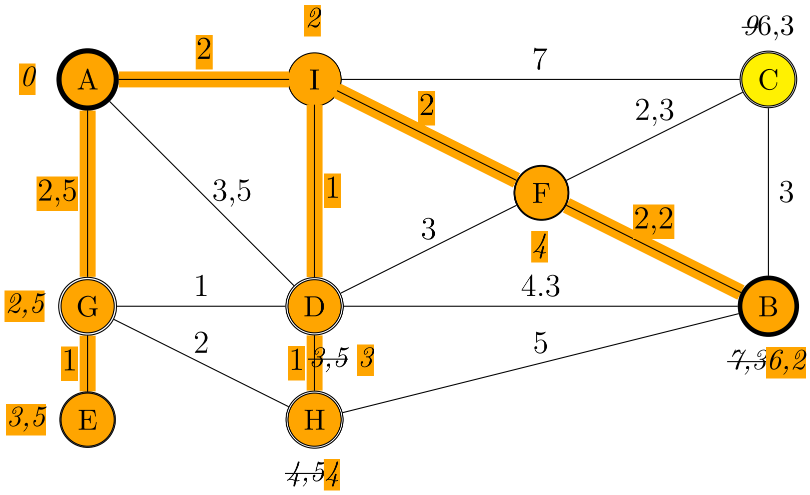
$$Menge\ der\ besuchten\ Ecken := \{A, I, G, E, F, H\}$$

Schritt 7

Der Algorithmus wiederholt das Vorgehen der vorherigen Schritte. Er berechnet die Entfernungen für die Punkte B (Entfernung über F: 2+2+2,2=6,2 oder über D: 2+1+1+4,3=8,3 oder über H: 2+1+1+5=8 und C (Entfernung über I: 2 + 7 = 9 oder über F: 2+2+2,3=6,3), Dabei wählt der Algorithmus den kürzesten Weg, in diesem Fall den Weg zu B.
Dijkstra hat somit den kürzesten Weg von A nach B gefunden, der von A über die Knoten I und F verläuft. Die Entfernung von 2+2+2,2=6,2 sowie der Pfad werden notiert. Der Knoten B wird zur Menge der besuchten Knoten hinzugefügt. Da der Weg zu diesem Knoten gesucht war kann der Algorithmus nun enden und sein Ergebnis ausgeben. Alternativ kann er in weiteren Schritten die Entfernung zu allen Knoten berechnen.



Anfang Schritt 7



Ende Schritt 7

Abbildung 35: Schritt 7 - Ende

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	~~7,3~~2+2+2,2=6,2	~~9~~2+2+2,3=6,3	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+14	2
Vorgänger	A	F	F	I	G	I	A	D	A

	A	B	C	D	E	F	G	H	I
Distanz zu A	0	~~7,3~~2+2+2,2=6,2	~~9~~2+2+2,3=6,3	~~3,5~~ 2+1=3	2,5+1=3,5	2+2=4	2,5	2+1+1=4	2
Vorgänger	A	F	F	I	G	I	A	D	A

$$Menge\ der\ besuchten\ Ecken := \{A, I, G, E, F, H, B\}$$

0.6.3 Sonderfälle

An dieser Stelle müssen wir uns auch überlegen, welche Sonderfälle es gibt und welche Ausgabe wir von unserem Algorithmus erwarten. Ein Fall wäre, dass der Endpunkt mit dem Startpunkt übereinstimmt. In diesem Fall hat der Weg von A nach B=A die Länge 0. Als Weg könnte die leere Kantenfolge [] oder [A] eine Möglichkeit sein. Ein weiterer Fall wäre, dass es keinen Weg von A nach B gibt. A könnte auf dem Festland liegen und B auf einer Insel, dann kann die Anreise mit dem AUto schwierig werden. In diesem Fall ist der Graph nicht zusammenhängend. Die Laänge des Weges kann ich mit ∞ beschreiben. Einen Weg gibt es nicht.

0.6.4 Pseudocode

Pseudocode ist eine informelle Programmiersprache, die verwendet wird, um den Ablauf eines Algorithmus oder eines Programms in einer menschenlesbaren Form darzustellen. Es handelt sich nicht um eine spezifische Programmiersprache, sondern um eine Möglichkeit, Algorithmen in natürlicher Sprache mit Elementen aus der Programmierung zu beschreiben [15].

Ich nutze im folgenden Pseudocode, da er sich gut eignet, um einen Algorithmus zu erklären. Dies bietet eine abstrakte und verständliche Darstellung des Dijkstra Algorithmus, ohne sich um die syntaktischen Details einer spezifischen Programmiersprache kümmern zu müssen. Inspirationen für die Formulierung des Algorithmus mit Hilfe des Pseudocodes stammen zum einen von Franz Pauers Skript „Algebra und diskrete Mathematik [14] und zum anderen aus dem Buch „Diskrete Mathematik für Informatiker von [7]. Trotzdem weicht der Code von diesen ab um für das Beispiel passender und leicht verständlich zu sein.

Dazu sei G:=(E,K) ein gewichteter Graph (mit einem Digraph funktioniert es analog) mit den Ecken E und den Kanten K. Sei w(X,Y) die Bewertung einer Kante von X nach Y und Sei $A \in E$ der Startpunkt und B $\in$ E der Endpunkt, dann berechnet folgender Algorithmus die Entfernung d(A,B), sowie den Weg(A,B), die Kantenfolge von A nach B mit der kürzesten Entfernung.

Algorithm 1: Dijkstra Algorithmus

Data: Variabler Startpunkt A und den variablen Endpunkt B, Graph G

Result: d(B) und Weg(A,B) der kürzesten Strecke von A nach B

Prozedur `Algorithmus(A,B)`

Definiere $M := \{A\}$ die Menge der besuchten Knoten

und $d(A,X) := \begin{cases} w(A,X) & \text{für } X \mid (A,X) \in K \\ 0 & \text{für B=A} \\ \infty & \text{sonst} \end{cases}$ und

$Weg(A,X) := \begin{cases} [A] & \text{für } X \mid (A,X) \in K \\ [A] & \text{für B=A} \\ \infty & \text{sonst} \end{cases}$

for solange B $\notin M$ **do**

Wähle U $\in$ E\M $\mid d(A,U)$ ist minimal;

Füge U der Menge M hinzu;

for für jeden Knoten V aus E\{A} $\mid (U,V) \in K$ **do**

d'(A,V):= $d(A,U) + w(U,V)$;

$d(A,V) := \begin{cases} d'(A,V) & \text{falls } d'(A,V) < d(A,V) \\ d(A,V) & \text{sonst} \end{cases}$;

Weg(A,V) := $\begin{cases} \text{Weg´(A,V):=[Weg(A,U), V]} & \text{falls d´(Á,V)< d(A,V)} \\ Weg(A,V) & \text{sonst} \end{cases}$;

end

end

return Länge des Weges von A nach B: d(A,B), Wege von A nach B: Weg(A,B);

0.7 Beweis

Nun gilt natürlich noch zu zeigen, dass der Weg den der Dijkstra Algorithmus findet auch tatsächlich immer der kürzeste ist.

Dafür gibt es mehrere Möglichkeiten:

0.7.1 Beweis durch Widerspruch nach Armin Barth

Für den Fall, dass ein Graph nur aus zwei Knoten und einer verbindenden Kante besteht, ist es offensichtlich, dass der Algorithmus den kürzesten Weg findet. Der Knoten wird mit minimaler Distanz ausgewählt und notiert. Betrachtet man nun einen beliebigen Schritt, des Algorithmus, so erweitert der Algorithmus die Menge der besuchten Ecken um einen Knoten X, dessen Distanz zum Startpunkt A sich nicht mehr ändern wird. Ist es dann möglich, dass ein kürzerer Weg von A nach X existiert? Angenommen es gäbe einen kürzeren Weg über einen anderen Punkt, nennen wir ihn P, so gäbe es zwei Möglichkeiten:

- $P \in Menge\ der\ besuchten\ Punkte$
- $P \notin Menge\ der\ besuchten\ Punkte.$

Im ersten Fall würde der Algorithmus den Weg über P wählen, da er immer den kürzesten Weg auswählt. Der kürzere Weg wäre also mit unserem Weg identisch. Im zweiten Fall kann der Weg über P zu X nicht kürzer sein als der Weg zu X, denn sonst würde der Algorithmus zuerst P auswählen nicht X. Folglich gibt es keinen kürzeren Weg nach X. vgl. [1]

0.7.2 Beweis durch Widerspruch nach Brigitte Westphal

vgl. [10] Angenommen, der Algorithmus von Dijkstra hat Wege konstruiert, die nicht die kürzesten Wege sind. Sprich wir finden von A zu einem Punkt Z einen kürzeren Weg als den berechneten. Dann können wir o.B.d.A annehmen, dass Z unter den Knoten, für die es noch kürzere Wege, als die berechneten, gibt, der sogenannte „kleinste Verbrecher ist. Der Weg(A,Z) hat einen vorletzten Knoten, den wir Y nennen. Da Z der kleinste falsch berechnete Knoten ist, muss der Weg nach Y richtig berechnet sein. Betrachten wir nun den Zeitpunkt, an dem die Distanz zum Knoten Z berechnet wurde und dieser zur Menge der besuchten Ecken hinzugefügt wurde. Da die Punkte benachbart sind und auf einem Weg liegen, wurde die Distanz von Z mit dem erreichen von Y neu berechnet. Dabei gibt es vier mögliche Fälle:

1. bevor Y erreicht wurde war $d(A, Z) = \infty$
2. bevor Y erreicht wurde gab es einen Weg mit größerer Entfernung, $d_1(A, Z) > d(A, Y) + w(Y, Z)$
3. bevor Y erreicht wurde war gab es einen Weg, der die gleiche Entfernung hatte $d_2(A, Z) = d(A, Y) + w(Y, Z)$
4. bevor Y erreicht wurde, gab es einen kürzeren Weg als über Y $d_3(A, Z) < d(A, Y)+w(Y, Z)$

Im 1. und 2. Fall tauscht der Algorithmus die alte Entfernung mit der neu berechneten Distanz aus. In den Fällen 3 und 4 lässt er die alte Distanz stehen. In jedem Fall, ist die Distanz d(A,Z) nach diesem Schritt kleiner oder gleich wie die Distanz des Weges der über Y geht. Damit haben wir einen Widerspruch zur Annahme. Dijkstra kann also keinen zu langen Weg konstruieren.

0.8 Weiterentwicklungen des Dijkstra Algorithmus

Der Dijkstra-Algorithmus kann also das Problem, einen kürzesten Eg zu finden lösen. Aber löst er es auch effizient oder gibt es effizientere Möglichkeiten? Die Hauptkritik an Dijkstras Algorithmus ist, dass er sehr viele Wege berechnet, die er nicht braucht.

Um insgesamt weniger Wege zu berechnen gibt es Möglichkeiten die Wegsuche einzuschränken oder zu leiten. Der A* Algorithmus tut genau das. Er nutzt Abschätzungen für einen eingeschränkten Suchbereich für die Route [1]. Das kann eine Funktion sein, die Wege in die passende Himmelrichtung besser bewerten. Oder eine Einschränkung des Graphen auf den wahrscheinlich relevanten Bereich. Damit lassen sich kurze Wege deutlich effizienter berechnen, die mit hoher Wahrscheinlichkeit auch die kürzesten sind. Allerdings besteht ein Chance, dass der Algorithmus nicht den kürzesten Weg findet [6].

Die ALT-Algorihtmen (A* plus Landmarks plus Triangle Inequality) von Goldberg und Harrelson sind eine andere Weiterentwicklung. Sie beginnen damit Landmarks L1, L2, L3,... auszuwählen, berechnen anschließend die kürzesten Distanzen aller Punkte zu den Landmarks und speichern diese.

Die Distanz zweier Punkte kann dann mithilfe der bekannten Distanzen zu den Landmarks abgeschätzt werden. Bezeichne $d_i(v)$ eine kürzeste Distanz des Knotens v zum Landmark L_i. Nun kann ich mit Hilfe der Dreiecksungleichung abschätzen, dass

$$d(v, w) \geq d_i(v) - d_i(w)$$

sein muss. Man findet eine geeignete untere Schranke für die Distanz, indem man das Maximum all dieser unteren Schranken über alle Landmarks wählt [1].

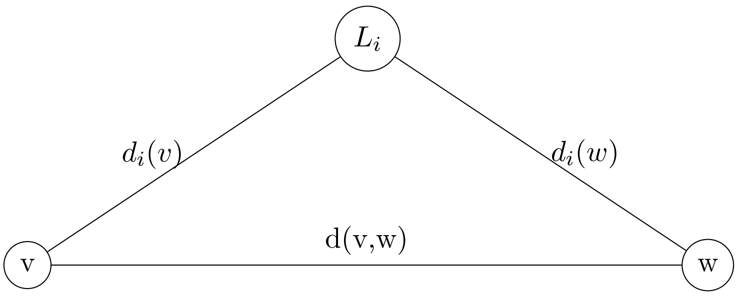


Abbildung 36: Skizze zur Abschätzung von Distanzen von Landmarks mit Hilfe der Dreiecksungleichung.

Fachdidaktischer Einschub: das EIS-Prinzip

Bei der Gestaltung von Unterricht hilft es, sich fachdidaktische Prinzipien in Erinnerung zu rufen. Für die Unterrichtsideen zum Thema "kürzeste Wege berechnen" wurde das EIS-Prinzp von Jerome Bruner gewählt. EIS-Prinzip steht für enaktiv, ikonisch und symbolisch. Es beschreibt drei Stufen der Wissensrepräsentation und des Lernens.

Die erste Stufe, die enaktive Stufe, bezieht sich auf das Lernen durch Handeln und praktisches Ausführen von Handlungen. Schüler*innen werden ermutigt, Mathematik aktiv zu erleben und zu erfahren, indem sie konkrete Aktionen durchführen, dadurch erhalten sie ein intuitives Verständnis. Der Vorteil der enaktiven Stufe besteht darin, dass sie den Lernenden ermöglicht, Mathematik auf einer handlungsorientierten Ebene zu erfassen und persönliche Erfahrungen zu sammeln. Sie können ihre eigenen Hypothesen aufstellen und durch eigenes Handeln überprüfen. Allerdings kann die enaktive Stufe auch Zeit und Raum in Anspruch nehmen, da die Durchführung praktischer Aktivitäten oft aufwändig ist.

Die ikonische Stufe bezieht sich auf die Nutzung von bildlichen Darstellungen, um mathematische Konzepte zu veranschaulichen. In diesem Kontext könnten Schüler*innen visualisieren und vergleichen. Durch das Betrachten und Vergleichen von Grafiken, Bildern und Darstellungen können sich die Schüler*innen komplexere Sachverhalte vorstellen. Der Vorteil der ikonischen Stufe liegt darin, dass sie den Schüler*innen ermöglicht, komplexe Informationen visuell zu verarbeiten und mathematische Beziehungen zu erkennen. Es kann ihnen helfen, geometrische Vorstellungen zu entwickeln und räumliche Beziehungen zu verstehen. Es muss jedoch darauf geachtet werden, dass der visuelle Aspekt nicht überbetont wird, denn das kann die Abstraktion eines mathematischen Konzepts erschweren.

Die symbolische Stufe ist die höchste Stufe der Wissensrepräsentation und bezieht sich auf die Verwendung von Symbolen und abstrakten mathematischen Konzepten. Es werden mathematische Formeln, Gleichungen und Funktionen genutzt um Sachverhalte zu beschreiben. Dadurch wird Information deutlich verdichtet und der Komplexitätgrad höher. Der Vorteil der symbolischen Stufe besteht darin, dass sie den Schüler*innen ermöglicht, mathematische Konzepte formal zu repräsentieren und abstrakte Denkprozesse zu entwickeln. Es ermöglicht ihnen, mathematische Modelle zu erstellen und mathematische Probleme systematisch zu lösen. Allerdings erfordert die symbolische Stufe oft ein hohes Maß an Abstraktionsfähigkeit und kann für manche Schüler*innen eine Herausforderung darstellen.

Durch die Integration des EIS-Prinzips im Mathematikunterricht wird den Schüler*innen ermöglicht, Mathematik auf unterschiedlichen Wahrnehmungsebenen zu erfassen und ein tieferes Verständnis für mathematische Konzepte zu entwickeln. Es unterstützt sie dabei, ihre individuellen Denkprozesse zu reflektieren und verschiedene Repräsentationen zu nutzen, um mathematische Probleme zu lösen [2].

Im folgenden Abschnitt werden Erklärungsansätze erläutert, wie die Idee des Algorithmus von Dijkstra in der Schule behandelt werden kann. Ziel ist es, den abstrakten Algorithmus zu verbildlichen und sogar in der enaktiven Ebene lebendig werden zu lassen. Ausgeführt wurden nur die Ideen zum Algorithmus. Dazu gehören selbstverständlich sämtliche Vorüberlegungen zur Routenplanung, denen auch die Schüler*innen unbedingt teilhaben müssen. Eine geeignete Methodenwahl hängt vom Alter und der Vorliebe der Schüler*innen ab, weshalb darauf verzichtet wurde diese festzulegen.

Verknüpfungen mit dem österreichischen Lehrplan

0.9 Verknüpfung mit dem österreichischen Lehrplan

In diesem Kapitel wird die Anwendung des Dijkstra-Algorithmus zur Berechnung kürzester Wege im österreichischen Curriculum untersucht. Der Dijkstra-Algorithmus ist ein grundlegendes Konzept in der Informatik und spielt eine zentrale Rolle in der Graphentheorie. Es passt also in die Fächer Informatik und Mathematik. Algorithmen und die Funktionsweise von Computern werden außerdem in dem Schulfach Digitale Grundbildung behandelt. Das Arbeiten mit Karten lässt sich zusätzlich in Geographie und Wirtschaftskunde (GW) eingliedern.

0.9.1 Lehrplan Mathematik der Allgemeinbildenden höheren Schule (AHS)

Inhalte der Graphentheorie sowie der Algorithmus von Dijkstra finden sich nicht explizit im Mathematiklehrplan der AHS wieder. Dennoch besteht die Möglichkeit, dieses Thema gerade im Rahmen eines fächerübergreifenden, projektorientierten Unterrichts zu behandeln und seine Daseinsberechtigung auch im Mathematik Lehrplan wiederzufinden. Dies lässt sich wie folgt begründen

Der Mathematikunterricht an österreichischen AHS verfolgt das Ziel, den Schüler*innen mathematische Kompetenzen zu vermitteln und sie für die Anwendung der Mathematik in verschiedenen Lebensbereichen zu motivieren. Dabei wird ein anwendungsorientierter Unterricht angestrebt, der das Lernen in Kontexten ermöglicht und den Fokus auf die praktische Anwendung mathematischer Konzepte legt [4]. Ein projektorientierter, anwendungsorientierter Unterricht bietet sich also an, um dieses Ziel zu erreichen.

Im Rahmen eines Unterrichts zum Thema "Wie finde ich den kürzesten Weg von A nach B" können die Schüler*innen die Grundlagen der Graphentheorie kennenlernen und den Dijkstra-Algorithmus verstehen und anwenden, um den kürzesten Weg in einem gegebenen Graphen zu berechnen. Durch die praktische Anwendung dieser mathematischen Konzepte und Verfahren in einem realen Kontext werden die Schüler*innen und Schüler motiviert, neues Wissen und neue Fähigkeiten zu erwerben.

Ein Unterricht zum Dijkstra-Algorithmus ermöglicht zudem fächerübergreifende Verbindungen. Es werden geografische und informatische Aspekte in den Unterricht integriert, um den Schüler*innen eine ganzheitliche Perspektive zu bieten.

Durch den Unterricht wird nicht nur die fachliche Kompetenz der Schüler*innen gestärkt, sondern auch das Verständnis für die Anwendbarkeit der Mathematik in verschiedenen Lebensbereichen. Der Unterricht ermöglicht es den Schüler*innen, mathematische Konzepte in einem praktischen Kontext zu erleben und ihre Problemlösefähigkeiten zu erweitern. Durch die anwendungsorientierte Herangehensweise und die fächerübergreifenden Verbindungen wird eine umfassende und praxisnahe Mathematikausbildung gefördert, die den Schülerinnen und Schülern ermöglicht, Mathematik als ein lebendiges Werkzeug zur Lösung realer Probleme zu erleben

0.9.2 Lehrplan Informatik 6.-8. Klasse AHS

Der Dijkstra-Algorithmus und die Berechnung des kürzesten Weges lassen sich gut in den Lehrplan für das Fach Informatik an österreichischen AHS integrieren. Der Schwerpunkt des Informatikunterrichts liegt auf der formalen Modellierung von Sachverhalten und der Problemlösung, die aus Analyse, Beschreibung in verschiedenen Darstellungsformen, algorithmischer Lösung, Implementation, Überprüfung und Interpretation sowohl vom informatischen als auch vom sachlichen Standpunkt besteht.

Bereits ab der 6. Klasse werden grundlegende Konzepte wie Algorithmen, Datenstrukturen und Programmierung behandelt. Die Schüler*innelernen, Aufgaben und Problemstellungen algorithmisch und formalsprachlich zu beschreiben, grundlegende Algorithmen zu entwerfen, zu formalisieren, zu implementieren und zu testen. Dabei wird auch auf die Modellierung von Aufgabenstellungen mit Hilfe der Informatik eingegangen.

In der 7. Klasse werden Aspekte der prozeduralen, funktionalen und objektorientierten Programmierung erläutert und anhand von Beispielen erläutert. Die Schüler*innen werden ermutigt, vielfältige Aufgaben mithilfe der Informatik zu modellieren und verschiedene Algorithmen zu entwerfen, zu formalisieren, zu implementieren und zu testen.

In der 8. Klasse werden wesentliche Aspekte und Methoden der Softwareentwicklung und des Softwareprojektmanagements behandelt. Die Schüler*innen lernen, Softwareprojekte zu planen und durchzuführen sowie die Schritte der Softwareentwicklung zu reflektieren. Sie sind in der Lage, die Effizienz von Algorithmen zu bewerten und gezielt nach Programmfehlern zu suchen und diese zu korrigieren. [4]

Der Dijkstra-Algorithmus kann demnach in all diesen Klassen behandelt werden. Je nach Können und Vorwissen der Schüler*Innen in unterschiedlich komplexer Form. Die Schüler*innen können explorativ lernen, wie man den Algorithmus anwendet, um den kürzesten Weg in einem Graphen zu berechnen. Dabei können sie sowohl die theoretischen Konzepte der Graphentheorie als auch die praktische Umsetzung in der Programmierung kennenlernen.

Ein anwendungsorientierter, projektorientierter Unterricht zum Thema kürzeste Wege berechnen, ermöglicht es den Schüler*innen, ihre Fähigkeiten in der Problemlösung und Algorithmik weiterzuentwickeln. Durch die praktische Anwendung des Algorithmus werden die Schüler*innen dazu motiviert, ihr Wissen in der Informatik und Mathematik zu vertiefen und ihre Problemlösefähigkeiten zu erweitern.

0.9.3 Lehrplan AHS Geographie und Wirtschaftskunde

Der Dijkstra-Algorithmus und die Berechnung kürzester Wege können auch im Geographie und Wirtschaftskunde Unterricht an österreichischen AHS ihren Platz finden. Der Lehrplan legt Wert auf den Erwerb von Sprachkompetenz durch die Auswertung von Texten, Bildern und grafischen Darstellungsformen wie Karten und Geomedien. Hier ist die präzise mathematische Formulierung der Schritte des Algorithmus zu nennen. Zudem sollen die Schüler*innen Urteils- und Kritikfähigkeit, Entscheidungs- und Handelskompetenz in räumlichen und ökonomischen Fragen entwickeln. Diese werden durch das Wissen über die Funktionsweise von Routenplanern untermauert [4].

Des Weiteren ist im Geographie und Wirtschaftskunde Unterricht die Wahrnehmung und Darstellung ein Basiskonzept. Hierbei geht es darum, wie Menschen die Welt wahrnehmen, Bilder und Vorstellungen entwickeln und wie darüber kommuniziert wird. Die Schüler*innen lernen, alltagsweltliche Wahrnehmung und Orientierung im physischen Raum zu reflektieren sowie wissenschaftlich, strukturierte und technisch unterstützte Wahrnehmungsmethoden zu analysieren

[4]. Mathematische Graphen können dabei als Beispiel dienen, wie räumlichen Aspekte dargestellt und kommuniziert werden können. Durch das Verständnis für die Berechnung kürzester Wege und die Visualisierung dieser Wege auf Karten oder anderen grafischen Darstellungen können die Schüler*innen die Effizienz von Verkehrsnetzen, Transportrouten oder die Auswirkungen von Standortentscheidungen besser nachvollziehen und kritisch analysieren.

Mit dem Unterricht zum Thema "kürzeste Wege berechnen" lernen die Schüler*Innen, räumliche Orientierungshilfen und Kommunikationsmittel wie Karten und grafische Darstellungen sachgerecht zu nutzen und kritisch zu hinterfragen.

0.9.4 Lehrplan AHS Digitale Grundbildung

Die Einführung des Pflichtgegenstands "Digitale Grundbildung" an Mittelschulen und AHS-Unterstufen ab dem Schuljahr 2022/23 stellt einen neuen Schwerpunkt im österreichischen Bildungssystem dar. Die Digitale Grundbildung wird in den Klassenstufen 5 bis 8 mit mindestens einer fixen Stunde pro Woche unterrichtet. Der Lehrplan legt dabei den Fokus auf drei Blickwinkel: Wie digitale Technologien funktionieren, welche gesellschaftlichen Wechselwirkungen sich durch ihren Einsatz ergeben und welche Interaktions- und Handlungsoptionen sich für die Schüler*innen ergeben [3].

Im Bereich der Digitalen Grundbildung ist insbesondere das Konzept des Computational Thinking relevant[4]. Der Dijkstra-Algorithmus kann hierbei als Beispiel für den Umgang mit Algorithmen dienen. Schüler*innen lernen, Abläufe aus dem Alltag zu benennen und zu beschreiben. Sie können eindeutige Handlungsanleitungen (Algorithmen) formulieren, verbal oder schriftlich. Zudem können sie Algorithmen ausführen und deren Ergebnisse verstehen.

Ein weiterer Aspekt der Digitalen Grundbildung ist die kreative Nutzung von Programmiersprachen. Schüler*innen haben die Möglichkeit, einfache Programme oder Webanwendungen mit geeigneten Tools zu erstellen, um bestimmte Probleme zu lösen oder Aufgaben zu erfüllen. Dabei lernen sie verschiedene Programmiersprachen und Produktionsabläufe kennen.[4]

Die Verwendung des Dijkstra-Algorithmus im Rahmen der Digitalen Grundbildung ermöglicht es den Schüler*innen, wichtige Aspekte des Computational Thinking zu erfassen. Sie können Algorithmen verstehen, anwenden und auch eigene Handlungsanleitungen formulieren. Darüber hinaus bietet der Algorithmus die Möglichkeit, die praktische Umsetzung von Algorithmen in der realen Welt zu erleben und zu reflektieren. Die Behandlung des Dijkstra-Algorithmus zur Berechnung kürzester Wege unterstützt somit die Entwicklung von Computational Thinking und Programmierkompetenzen bei den Schüler*innen.

Abschließend lässt sich festhalten, dass das Thema des Dijkstra-Algorithmus eine Schnittstelle verschiedener Disziplinen darstellt und sich daher ideal für einen fächerübergreifenden, projektorientierten Unterricht eignet. Die Fächer Mathematik, Informatik, Geographie und Wirtschaftskunde sowie der Digitalen Grundbildung können verknüpft werden und die Schüler*innen können ihre Modellierungskompetenzen weiterentwickeln. Diese Kompetenzen beinhalten die Anwendung von fachlichem Wissen und Können, um Aussagen über die Realität treffen zu können.

Altersmäßig würde ich die Thematik für die 6. Klasse, also die 10. Schulstufe vorschlagen. In dieser Klasse werden Algorithmen im Fach Digitale Grundbildung am intensivsten behandelt werden. Der Unterricht kann auch in niedrigeren Klassen stattfinden, da wenige Grundkenntnisse notwendig sind. Ebenso kann der Komplexitätsgrad für die Oberstufe einfach erhöht werden.

Kürzeste Wege in der Schule

0.10 Kürzeste Wege und das EIS Prinzip

Die symbolische Ebene ist die abstrakteste Ebene. In dieser Arbeit wurde der Algorithmus mit Hilfe eines Pseudocodes beschrieben. Je nach Vorerfahrung sind Schüler*Innen der Oberstufe durchhaus in der Lage diesen zu verstehen. Gibt man Ihnen jedoch Beispiele und Verbildlichungen als Hilfestellung tun Sie sich deutlich einfacher. Diese können in die ikonische Ebene geordnet werden. Auch in dieser Arbeit finden sich viele Beispiele, die hier eingeordnet werden können. Dazu zählen sämtliche gezeichnete Graphen, aber auch Landkarten oder Tabellen. Die enaktive Ebene wurde nur sehr stiefmütterlich behandelt. Eventuell fährt jemand mit dem Finger auf einer Karte umher um dem Weg zu folgen. Folgende Ideen für den Unterricht wollen vor allem die letzten beiden Ebenen erreichen. Ziel ist es auch jüngeren Schüler*innen die Idee des Algorithmus von Dijkstra greifbar zu machen. Dabei ist zu betonen, dass sich im Unterricht nicht auf eine Ebene beschränkt werden muss. Im Gegenteil, ist die Kombination gerade im Sinnen der Innendifferenzierung förderlich. Schüler*Innen können auf einer Ebene abgeholt werden und haben dann die Möglichkeit auch in andere Ebenen zu folgen.

0.11 Dijkstra und die Ameisen

Die Art und Weise, wie [6] den Algorithmus von Dijkstra zu erklären ähnelt der Art und Weise, wie in dieser Arbeit die einzelnen Schritte an einem Beispiel erklärt wurden. Allerdings nutzt er, für ein leichteres Verständnis, Ameisen. In seiner Vorstellung starten sehr viele Ameisen zur gleichen Zeit am Startpunkt und laufen mit gleicher Geschwindigkeit in alle möglichen Richtungen. Wann immer sich der Weg teilt, verfolgen noch genügen Ameisen den Weg, sodass sie sich aufteilen können. Wann immer eine Ameise an einen neuen Knoten kommt, der noch nicht Markiert ist, so tut sie dies und notiert ihren Weg. Nachdem alle Ameisen gleichzeitig gestartet sind kann sie sicher sein, dass sie den kürzesten Weg gefunden hat. Trifft die Ameise entweder auf eine andere Ameise, oder auf einen schon markierten Knoten, so kann sie umkehren denn offensichtlich gibt es schon einen kürzeren Weg. Hier folgt ein ausführliches Beispiel, dass mit Schüler*Innen gut bearbeitet werden kann. Dabei sollten davor unbedingt Vorüberlegungen zur Routenplanung wie im 2. Kapitel vorangestellt werden. Methodisch kann auf verschiedene Art und Weise gearbeitet werden. Unbedingt sollten Schüler*Innen zu Beginn selbst versuchen den kürzesten Weg zu finden und Überlegungen anstellen, wie ein Algorithmus diesen finden kann. Anschließend können die Schüler*innen sich nach jeden Schritt überlegen, ob sie schon selbst eine Lösung finden, oder ob sie noch mehr Schritte mit Erklärung brauchen. So können Schüler*innen nach Ihrem eigenen Tempo und Level gleichzeitig am gleichen Thema arbeiten.

Eine Kundschafterin eines Ameisenstamms hat in Oppenheim (O) ein großes Stück Futter gefunden. Welchen Weg sollen die Ameisen nehmen, um am schnellsten zur Beute zu gelangen. Der Ameisenstamm befindet sich in Imstadt (I). Von dort führen 5 Wege weg, also teilen sich unzählige Ameisen auf, um die Wege zu erkunden. Wir nehmen an, dass alle Ameisen gleich schnell laufen. (zum Beispiel 1 km pro Minute) Wir verfolgen den Weg der Ameisen.

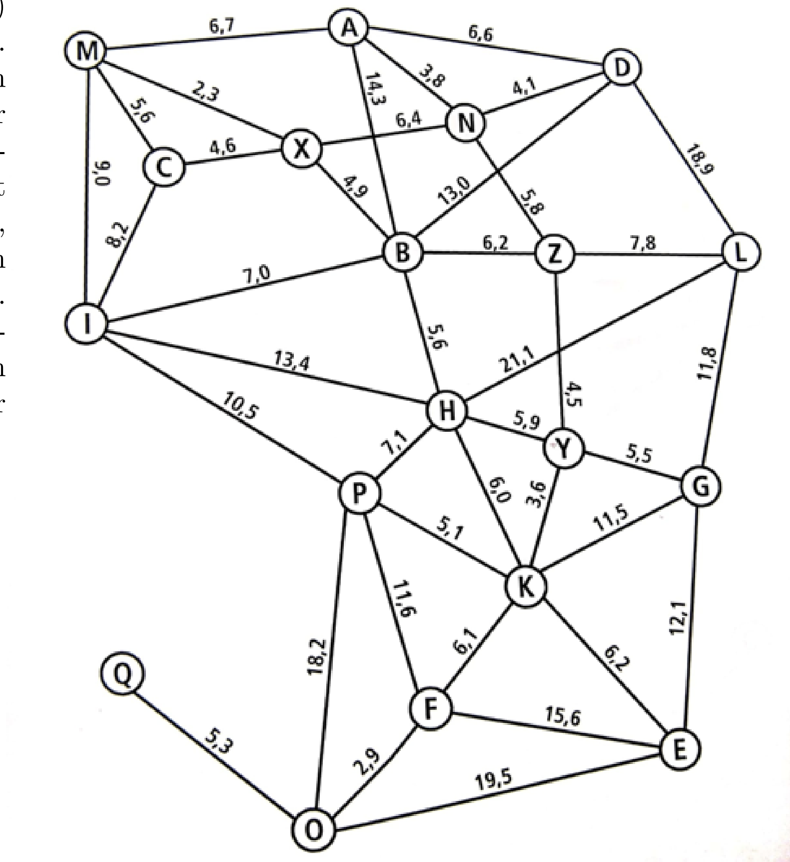


Abbildung 37: Ameisen Ausgangssituation. Quelle: [6]

Die Wege der Ameisen nach 7 Minuten. Die Ameisen haben B erreicht (auch wenn die 7km lange Strecke am längsten aussieht, nicht maßstabsgetreu) Bei den anderen Strecken sind sie noch auf dem Weg. Es gibt von I nach B garantiert keinen kürzeren Weg!

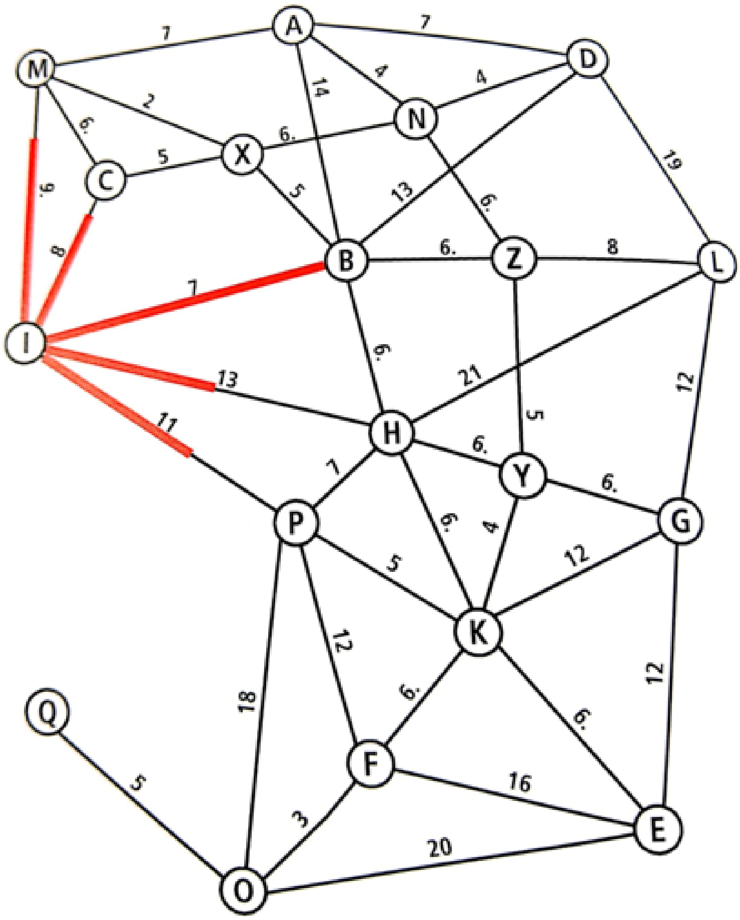


Abbildung 38: Ameisen nach 7 Minuten. Quelle: [6]

Nun: die Ameisen bei B markieren ihren Erfolg und notieren die kürzeste Strecke. Alle Ameisen gehen weiter, die bei B teilen sich auf. Es sind 5 Wege möglich. Nach einer weiteren Minute kommen Ameisen in C an. Sie sehen, dass sie die ersten sind, markieren ihre Strecke und teilen sich auf.

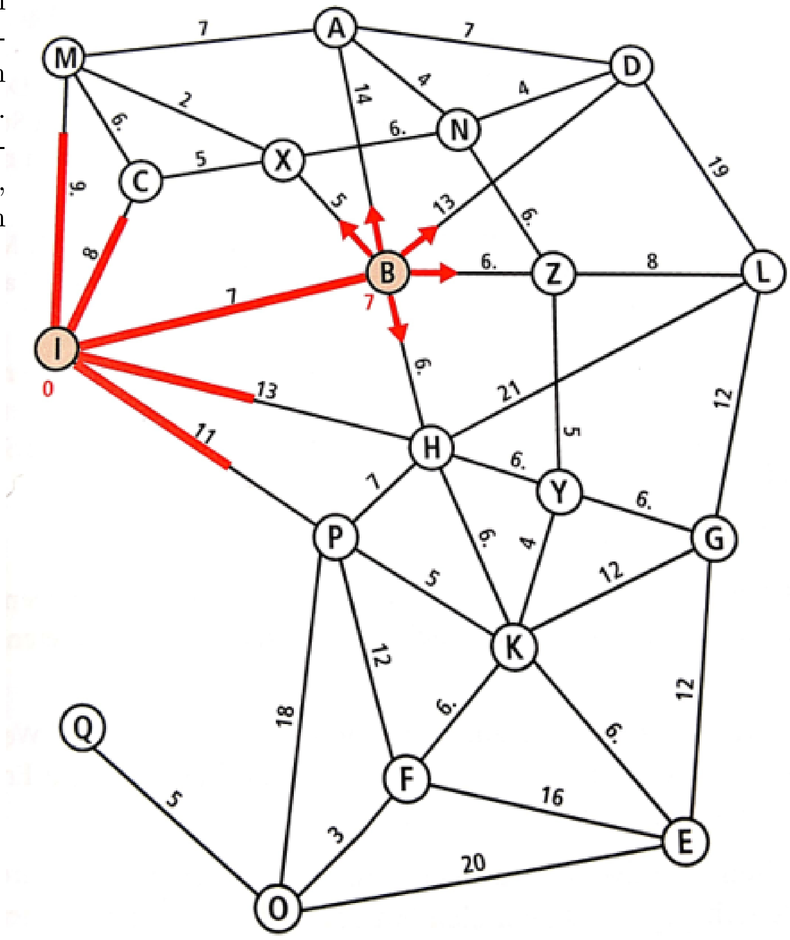


Abbildung 39: Ameisen nach 8 Minuten. Quelle: [6]

Nach insgesamt 9 Minuten kommen die ersten Ameisen in M an. Sie markieren dies und teilen sich auf. Die kürstesten Wege zu B, C und M stehen nun fest. Nun gehen Ameisen von M und C aufeinenader zu. Welche Informationen können sie austauschen, wenn sie aufeinandertreffen?

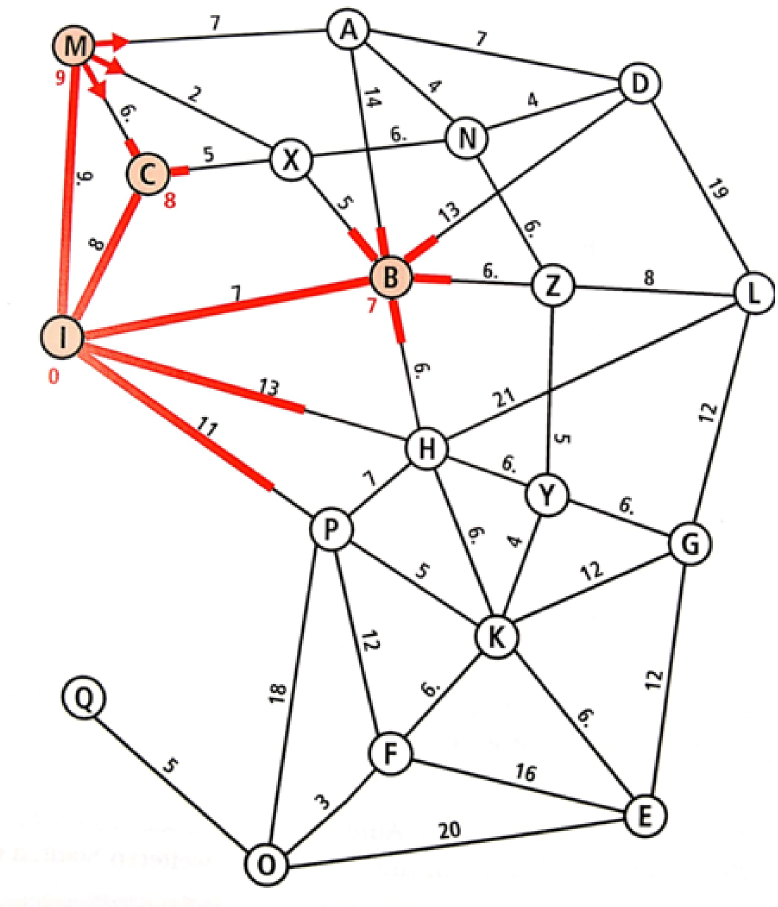


Abbildung 40: Ameisen nach etwas mehr als 9 Minuten. Quelle: [6]

Die Ameisen von C wissen, dass es schon einen kürzeren Weg nach C gibt. Die von M brauchen also dorthin nicht weiterlaufen. Andersherum genauso. Es ist also sinnlos weiterzulaufen. Der Weg wird als unbrauchbar markiert und die Ameisen können zurücklaufen. In der 11. Minute erreichen die Ameisen P und X. P wird auf direktem Weg von I aus erreicht. Bei X kommen Ameisen an, die über M gelaufen sind.

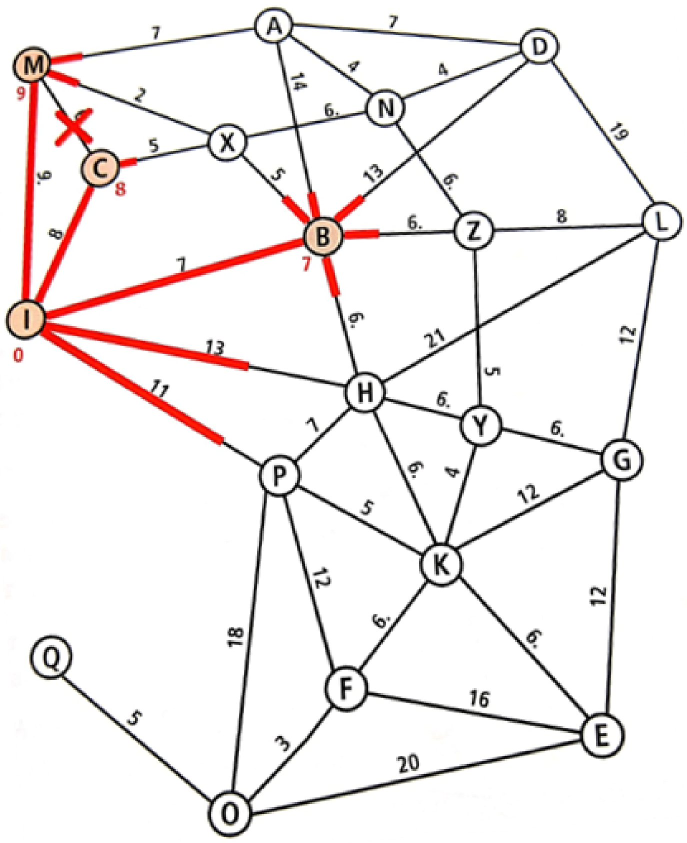


Abbildung 41: Ameisen vor der 11. Minute. Quelle: [6]

Wieder teilen sich die Ameisen auf. Bei X gibt es nur noch einen offenen Weg, bei den anderen treffen sie schnell auf Kameraden. Prinzip verstanden?

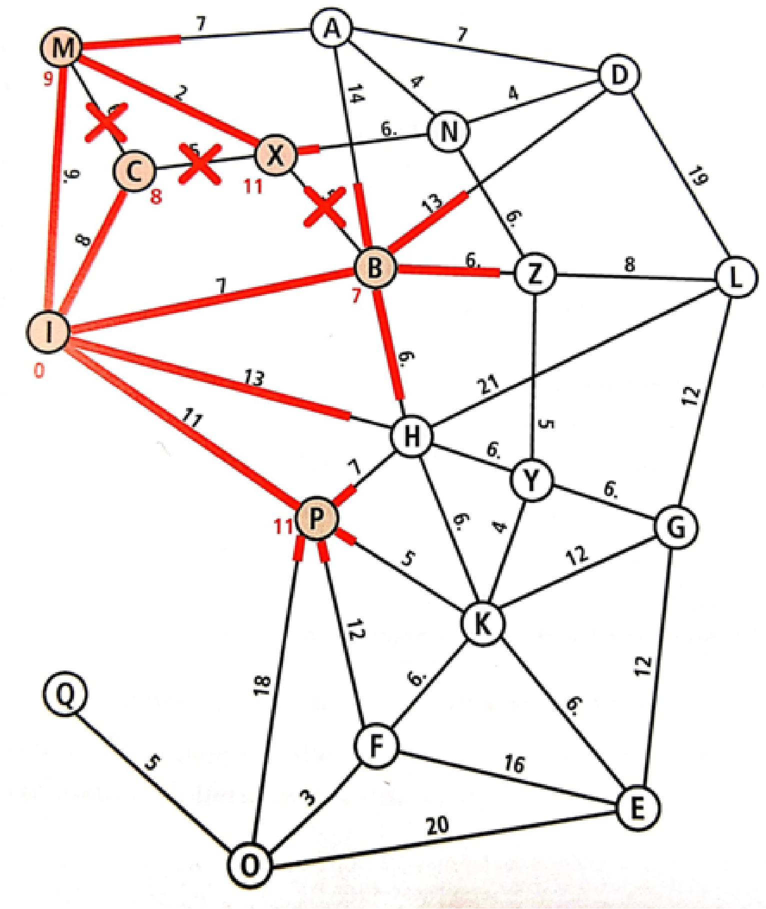


Abbildung 42: Ameisen nach der 11. Minute. [6]

Statt einzelne Wege auszuprobieren und wieder zu verwerfen, probieren die Ameisen alle Wege gleichzeitig. Wenn sie als erster an einem Punkt ankommen, wissen sie, dass sie sicher den kürzesten Weg gefunden haben. (Sonst wäre schon jemand da) Treffen die Ameisen auf Artgenossen, wissen sie, dass das anvisierte Ziel schon früher erreicht wurde.

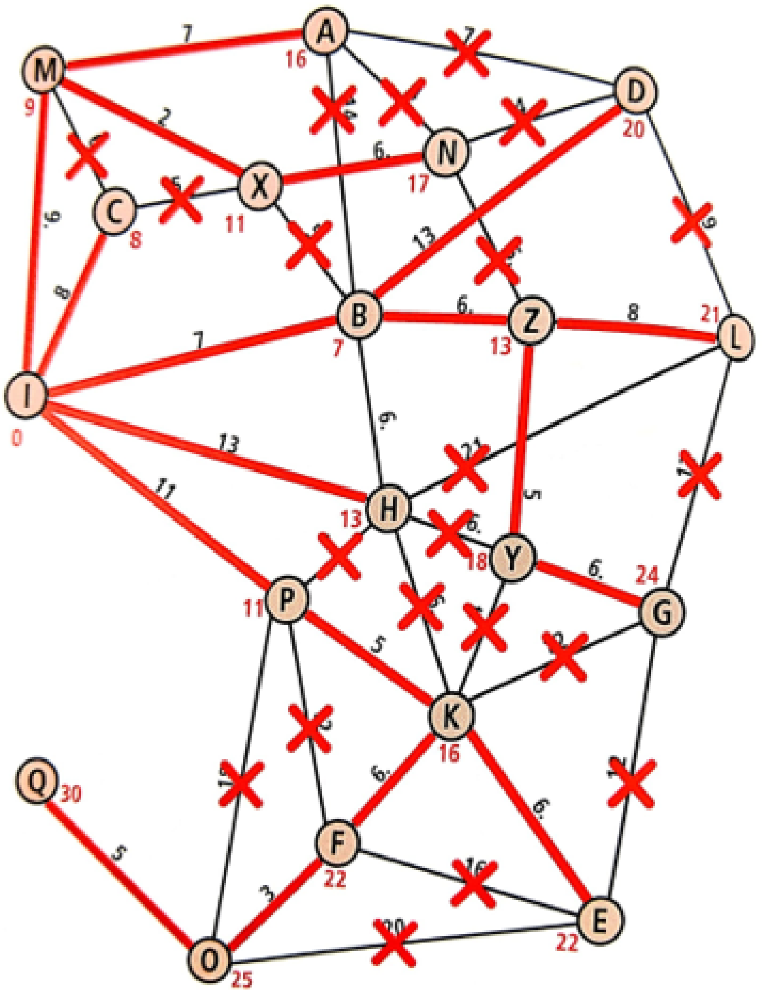


Abbildung 43: Ameisen haben zu allen Knoten kürzeste Wege gefunden. Quelle: [6]

0.12 Kürzeste Wege - Straßenkreide

[8] beschreibt eine Methode, die auch mit deutlich jüngeren Schüler*Innen behandelt werden kann, da sie hauptsächlich auf der enaktiven Ebene stattfindet. Dazu muss zur Vorbereitung ein Kartennetz gezeichnet werden. Dieses ähnelt einem unbewerteten Graphen kann jedoch beliebig illustriert werden und somit mehr den Charakter einer Karte annehmen. Die Handlung der Schüler*Innen sieht dann wie folgt aus:
Die Schüler*innen starten am Ausgangspunkt. Jede/r Schüler*in geht zu einem unmittelbar benachbarten Knoten, zählt dabei seine Schritte und notiert die Schrittzahl bei dem Knoten. Von den erreichten Knoten gehen die Schüler*innen nun in alle möglichen Richtungen weiter und addieren laufend ihre Schritte. Dabei achten sie darauf, dass jeder Weg einmal abgeschritten wird. Trifft ein Schüler auf einen Knoten, der schon eine kleinere Schrittzahl aufweist, so ist er fertig. Trifft ein/e Schüler*in auf eine/n andere/n Schüler*in, so gibt es bereits einen kürzeren Weg, beide Schüler*innen sind also fertig. So wird der ganze Graph abgelaufen. Wichtig dabei ist, dass die Schüler*innen gleich schnell gehen und gleich große Schritte machen. Vergleicht man dies mit Gallenbachers Erklärung, so lässt Hartmann die Schüler*innen im Grunde selbst die Ameisen sein. Damit holt sie die Schüler*Innen in der enaktiven Ebene ab. Im Unterricht könnte man nach dieser Handlung den Inhalt abstrahieren und mit Graphen und Zeichnungen

auf der symbolischen Ebene wie Gallenbacher fortfahren.

Bibliography

[1] Armin Barth. Algorithmik für Einsteiger. Wiesbaden: Springer, 2013.

[2] Jerome Bruner. The Process of Education. Cambridge: Harvard University Press, 1960.

[3] Bundesministerium Bildung, Wissenschaft und Forschung. Digitale Grundbildung. https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/dgb.html. Zugriff am 13. August 2023. 2022.

[4] Bundesministerium Bildung, Wissenschaft und Forschung. Lehrplan der allgemeinbildenden höheren Schu https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10008568. Zugriff am 8. Mai 2023. 2023.

[5] Rolf Dreyer. Sportbootführerschein See, mit amtlichem Fragenkatalog. Bielefeld: Delius Klasing, 2021. ISBN: 9783667118134.

[6] Jens Gallenbacher. Abenteuer Informatik: IT zum Anfassen für alle von 9 bis 99, vom Navi bis Social M 4. Auflage. Berlin and Heidelberg: Springer, 2017. ISBN: 9783662539644. URL: http://www.springer.com/.

[7] Rod Haggarty. Diskrete Mathematik für Informatiker. Informatik Mathematik. München: Pearson Studium, 2004. ISBN: 3827370957.

[8] W. Hartmann. Informatikunterricht planen und durchführen. Heidelberg: Springer, 2006.

[9] Carina Homrighausen. "Das GPS-System Eine theoretische Annäherung und Ansätze zur Anwendung im Physikunterricht". Masterarbeit. Universität Bielefeld, 2008.

[10] Stephan Hußmann and Brigitte Lutz-Westphal, eds. Diskrete Mathematik erleben: Anwendungsbasierte u 2., erweiterte Auflage. Wiesbaden: Springer Spektrum, 2015. ISBN: 9783658069926.

[11] Stephan Hußmann and Brigitte Lutz-Westphal, eds. Kombinatorische Optimierung erleben. Wiesbaden: Vieweg, 2007.

[12] Neumann, Ulrich. Navigation, Von der Kogge zur modernen Seeschifffahrt. https://www.planet-wissen.de/technik/schifffahrt/geschichte_der_schifffahrt/pwienavigation100.html. Zugriff am 24. Juli 2023. 2020.

[13] Norddeutscher Rundfunk. Navi an, Kopf aus: Die lustigsten GPS-Fails. https://www.n-joy.de/entertainment/GPS-Navigation-Die-lustigsten-Navi-Fails,navifails100.html. Zugriff am 24. Juli 2023. 2022.

[14] Franz Pauer. Algebra und diskrete Mathematik, Skriptum zur Vorlesung im Sommersemester 2019. Innsbruck, 2019.

[15] Thomas Widhalm Richard; Mück. Topic Maps: Semantische Suche im Internet. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 283–341. ISBN: 978-3-64-56285-3. URL: https://doi.org/10.1007/978-3-642-56285-3_10.