

Diskrete Modellierung

**Eine Einführung in grundlegende Begriffe und
Methoden der Theoretischen Informatik**

Skript zur Vorlesung

Prof. Dr. Nicole Schweikardt

Inhaltsverzeichnis

1	Einführung ins Thema “Diskrete Modellierung”	7
1.1	Wozu “Diskrete Modellierung” im Informatik-Studium?	7
1.2	Ziele der Veranstaltung “Diskrete Modellierung”	15
1.3	Der Begriff “Diskrete Modellierung”	15
1.4	Literaturhinweise zu Kapitel 1	18
1.5	Übungsaufgaben zu Kapitel 1	21
2	Mathematische Grundlagen und Beweistechniken	24
2.1	Mengen	25
2.1.1	Was ist eine Menge?	25
2.1.2	Mengenalgebra	28
2.1.3	Das Komplement einer Menge	32
2.1.4	Mächtigkeit bzw. Kardinalität einer Menge	33
2.1.5	Die Potenzmenge	34
2.2	Kartesische Produkte und Relationen	35
2.2.1	Paare, Tupel und kartesische Produkte	35
2.2.2	Worte bzw. endliche Folgen	38
2.2.3	Relationen	39
2.3	Funktionen	40
2.3.1	Totale Funktionen und partielle Funktionen	40
2.3.2	Eigenschaften von Funktionen	41
2.3.3	Spezielle Funktionen	44
2.4	Ein Beispiel zur Modellierung mit Wertebereichen	45
2.5	Beweise verstehen und selbst formulieren	46
2.5.1	Was sind “Sätze” und “Beweise”?	46
2.5.2	Beweistechnik “direkter Beweis”	47
2.5.3	Beweistechnik “Beweis durch Kontraposition”	47
2.5.4	Beweistechnik “Beweis durch Widerspruch” (indirekter Beweis)	48
2.5.5	Beweistechnik “Beweis durch vollständige Induktion”	50
2.6	Rekursive Definitionen von Funktionen und Mengen	54
2.6.1	Rekursive Definitionen von Funktionen	54
2.6.2	Rekursive Definitionen von Mengen	57
2.7	Literaturhinweise zu Kapitel 2	60
2.8	Übungsaufgaben zu Kapitel 2	60
3	Aussagenlogik	72
3.1	Wozu “Logik” im Informatik-Studium?	72
3.2	Syntax und Semantik der Aussagenlogik	73
3.3	Erfüllbarkeit und Allgemeingültigkeit	83
3.4	Folgerung und Äquivalenz	85

3.5	Normalformen	88
3.6	Literaturhinweise	95
3.7	Übungsaufgaben zu Kapitel 3	95
4	Graphen und Bäume	107
4.1	Graphen	108
4.1.1	Grundlegende Definitionen	108
4.1.2	Wege in Graphen	113
4.1.3	Ähnlichkeit zweier Graphen	119
4.1.4	Markierte Graphen	120
4.1.5	Zuordnungsprobleme	121
4.2	Bäume	127
4.2.1	Ungerichtete Bäume	127
4.2.2	Gerichtete Bäume	131
4.2.3	Modellierungsbeispiele	136
4.3	Einige spezielle Arten von Graphen	138
4.3.1	Spezielle ungerichtete Graphen	138
4.3.2	Spezielle gerichtete Graphen	140
4.4	Literaturhinweise	143
4.5	Übungsaufgaben zu Kapitel 4	144
5	Markov-Ketten als Grundlage der Funktionsweise von Suchmaschinen im Internet	161
5.1	Die Architektur von Suchmaschinen	161
5.2	Der Page-Rank einer Webseite	163
5.3	Der Zufalls-Surfer	166
5.4	Markov-Ketten	169
5.5	Die effiziente Berechnung des Page-Rank	170
5.6	Literaturhinweise	174
5.7	Übungsaufgaben zu Kapitel 5	174
6	Logik erster Stufe (Prädikatenlogik)	178
6.1	Motivation zur Logik erster Stufe	178
6.2	Strukturen	179
6.3	Terme	182
6.4	Syntax der Logik erster Stufe	184
6.5	Semantik der Logik erster Stufe	186
6.5.1	Beispiele zur Semantik der Logik erster Stufe	187
6.5.2	Formale Definition der Semantik der Logik erster Stufe	188
6.6	Ein Anwendungsbereich der Logik erster Stufe: Datenbanken	191
6.7	Erfüllbarkeit, Allgemeingültigkeit, Folgerung und Äquivalenz	195
6.8	Grenzen der Logik erster Stufe	197
6.9	Literaturhinweise	197
6.10	Übungsaufgaben zu Kapitel 6	197
7	Endliche Automaten zur Modellierung von Abläufen	205
7.1	Deterministische endliche Automaten	207
7.2	Nichtdeterministische endliche Automaten	212
7.3	Äquivalenz von NFAs und DFAs	215
7.4	Das Pumping-Lemma für reguläre Sprachen	216

7.5	Reguläre Ausdrücke	220
7.6	Ausblick	222
7.7	Literaturhinweise	223
7.8	Übungsaufgaben zu Kapitel 7	223
8	Kontextfreie Grammatiken zur Modellierung von Strukturen	231
8.1	Definition des Begriffs „Kontextfreie Grammatik“	231
8.2	Bedeutung der Produktionen: Semantik von KFGs	233
8.3	Beispiele	236
8.4	Ausblick	240
8.5	Literaturhinweise	241
8.6	Übungsaufgaben zu Kapitel 8	241
9	Ausblick auf weitere Modellierungstechniken	246
9.1	Petri-Netze zur Modellierung von Abläufen	246
9.2	Das Entity-Relationship-Modell zur Modellierung von Datenbanken	250
9.3	Eine Fallstudie	256
	9.3.1 Datenbank-Entwurf: Autowerkstatt	256
	9.3.2 Abläufe bei der Auftragserteilung	259
9.4	Literaturhinweise	260
9.5	Übungsaufgaben zu Kapitel 9	261
10	Beispielklausuren	264
	Literaturverzeichnis	349

1 Einführung ins Thema “Diskrete Modellierung”

1.1 Wozu “Diskrete Modellierung” im Informatik-Studium?

In der Informatik wird das Modellieren mittels diskreter Strukturen als typische Arbeitsmethode in vielen Bereichen angewandt. Es dient der präzisen Beschreibung von Problemen durch spezielle Modelle und ist damit Voraussetzung für die systematische Lösung eines Problems. In den verschiedenen Gebieten der Informatik werden unterschiedliche, jeweils an die Art der Probleme und Aufgaben angepasste, diskrete Modellierungsmethoden verwendet. Ziel ist jeweils, (nur) die zur Lösung des Problems *relevanten* Aspekte präzise zu beschreiben.

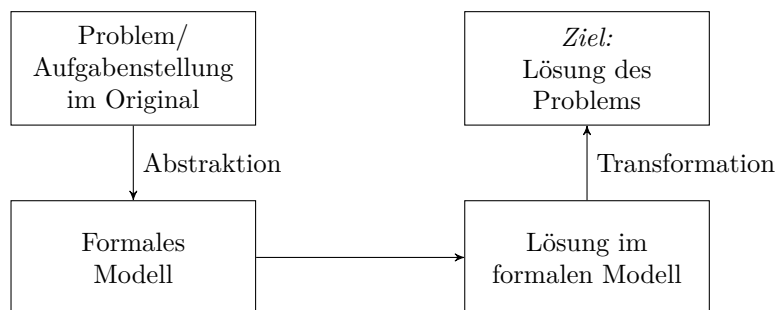
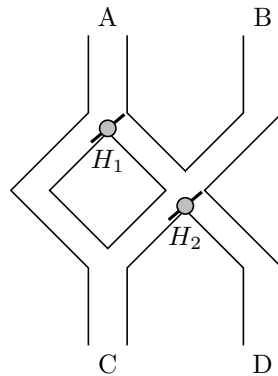


Abbildung 1.1: Generelle Vorgehensweise in der Informatik

In der Veranstaltung “Diskrete Modellierung” werden zunächst die grundlegenden Begriffe, wie z.B. “Modell” und “Modellierung”, geklärt. Anschließend werden verschiedene Ausdrucksmittel der Modellierung vorgestellt und anhand von anschaulichen Beispielen verdeutlicht.

Beispiel 1.1 (Problem “Murmeln”).

Die nachfolgende Abbildung zeigt ein Spiel, in dem Murmeln bei A oder B in die Spielbahn fallen gelassen werden.



Je nach Stellung der Hebel H_1 und H_2 rollen die Murmeln in der Spielbahn nach links oder rechts. Sobald eine Murmel auf einen dieser Hebel trifft, wird der Hebel nach dem Passieren der Murmel umgestellt, so dass die nächste Murmel in die andere Richtung rollt. Zu Beginn ist jeder der beiden Hebel so eingestellt, dass die nächste Murmel, die auf den Hebel trifft, nach links rollt. Wenn beispielsweise nacheinander drei Murmeln fallen gelassen werden, wobei die erste und dritte Murmel bei A und die zweite Murmel bei B fallen gelassen wird, dann kommen die ersten beiden Murmeln an der Öffnung C und die letzte Murmel an der Öffnung D heraus.

Frage: Aus welcher Öffnung fällt die letzte Murmel, wenn sieben Murmeln fallen gelassen werden, wobei die erste, zweite, vierte und letzte Murmel bei A und alle anderen Murmeln bei B fallen gelassen werden?

Lösungsansätze:

1. Knobeln, um eine Lösung per "Geistesblitz" zu erhalten
2. Systematisches Vorgehen unter Verwendung von Informatik-Kalkülen

Hier wird der 2. Ansatz verfolgt.

Erste Analyse des Problems:

- *relevante Objekte:*
Spielbahn, Eingänge A und B, Ausgänge C und D, Hebel H_1 und H_2 , Murmeln
- *Tätigkeit:*
Einwerfen von Murmeln an Eingängen A und/oder B
- *Start:*
Hebel H_1 und H_2 zeigen nach links
- *Ziel:*
Herausfinden, aus welchem Ausgang die letzte Murmel rollt, wenn nacheinander Murmeln an folgenden Eingängen eingeworfen werden: A, A, B, A, B, B, A
- *Eigenschaften/Beziehungen:*
 - Hebelpositionen: H_1 zeigt entweder nach links oder nach rechts, H_2 zeigt entweder nach links oder nach rechts.
 - Für jeden der beiden Hebel H_1 bzw. H_2 gilt: Wenn er nach links (bzw. rechts) zeigt so rollt die nächste an ihm vorbeierollende Murmel nach links (bzw. nach rechts) weiter.

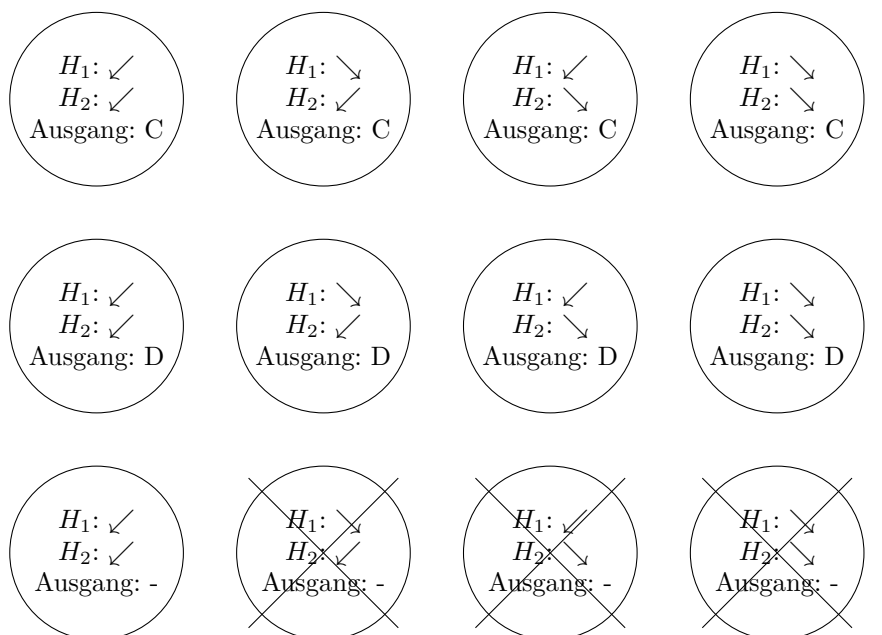
- Jeder der beiden Hebel H_1 bzw. H_2 ändert bei jedem Kontakt mit einer Murmel seine Richtung.
- Eine bei A eingeworfene Murmel rollt zu Hebel H_1 .
Eine bei B eingeworfene Murmel rollt direkt zu Hebel H_2 , ohne Hebel H_1 zu passieren.
- Zeigt H_1 nach links, so rollt eine bei A eingeworfene Murmel direkt zu Ausgang C.
Zeigt H_1 nach rechts, so rollt eine bei A eingeworfene Murmel zu Hebel H_2 .
- Zeigt H_2 nach links, so rollt eine diesen Hebel passierende Murmel zu Ausgang C.
Zeigt H_2 nach rechts, so rollt eine diesen Hebel passierende Murmel zu Ausgang D.

Abstraktionen:

1. Nutze Abkürzungen:

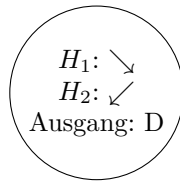
- $H_1 : \swarrow \hat{=}$ Hebel H_1 zeigt nach links
 $H_1 : \searrow \hat{=}$ Hebel H_1 zeigt nach rechts
 $H_2 : \swarrow \hat{=}$ Hebel H_2 zeigt nach links
 $H_2 : \searrow \hat{=}$ Hebel H_2 zeigt nach rechts
 Ausgang: C $\hat{=}$ die zuvor fallen gelassene Murmel ist an Ausgang C herausgerollt
 Ausgang: D $\hat{=}$ die zuvor fallen gelassene Murmel ist an Ausgang D herausgerollt
 Ausgang: - $\hat{=}$ es wurde noch keine Murmel eingeworfen

2. Betrachte die möglichen “Zustände”, die auftreten dürfen:



unzulässig, da vor dem Einwurf der ersten Murmel beide Hebel nach links zeigen müssen

3. Formale Modellierung der “Zustände”: Repräsentiere den “Zustand”



durch das Tupel (R, L, D) .

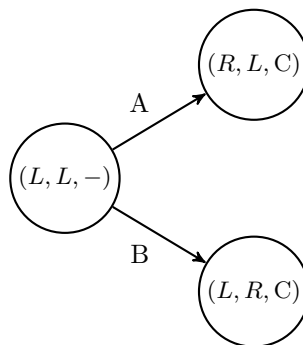
Allgemein wird ein Zustand durch ein Tupel (x, y, z) repräsentiert mit $x \in \{L, R\}$, $y \in \{L, R\}$ und $z \in \{C, D, -\}$, für das folgende Bedingung erfüllt ist: falls $z = -$, so ist $x = y = L$.

Übergänge von einem Zustand in einen anderen Zustand:

Vom Zustand $(L, L, -)$ aus kann man durch Einwerfen einer einzelnen Murmel in folgende Zustände gelangen:

- (R, L, C) , indem die Murmel bei A eingeworfen wird,
- (L, R, C) , indem die Murmel bei B eingeworfen wird.

Graphische Darstellung:



Insgesamt ergibt sich das in Abbildung 1.2 dargestellte Bild aus Zuständen und Zustandsübergängen.

Lösung des Problems “Murmeln”:

An diesem Bild lässt sich unser ursprüngliches Problem “Murmeln” (Frage: Aus welchem Ausgang rollt die letzte Murmel, wenn nacheinander Murmeln an den Eingängen A, A, B, A, B, B, A eingeworfen werden?) leicht lösen, indem man einfach einen Weg vom “Startzustand” sucht, bei dem die Pfeile nacheinander mit A, A, B, A, B, B, A beschriftet sind. In Abbildung 1.2 gibt es genau einen solchen Weg; er endet mit dem Zustand (L, R, C) . Die Antwort auf die ursprünglich gestellte Frage lautet also: Wenn nacheinander Murmeln an den Eingängen A, A, B, A, B, B, A eingeworfen werden, so rollt die letzte Murmel durch Ausgang C.

Man beachte, dass man anhand von Abbildung 1.2 auch die folgende Frage beantworten kann:

Ist es möglich, vom Startzustand aus durch geschicktes Einwerfen von Murmeln zu erreichen, dass die letzte Murmel aus Ausgang D herausrollt und danach beide Hebel nach rechts zeigen?

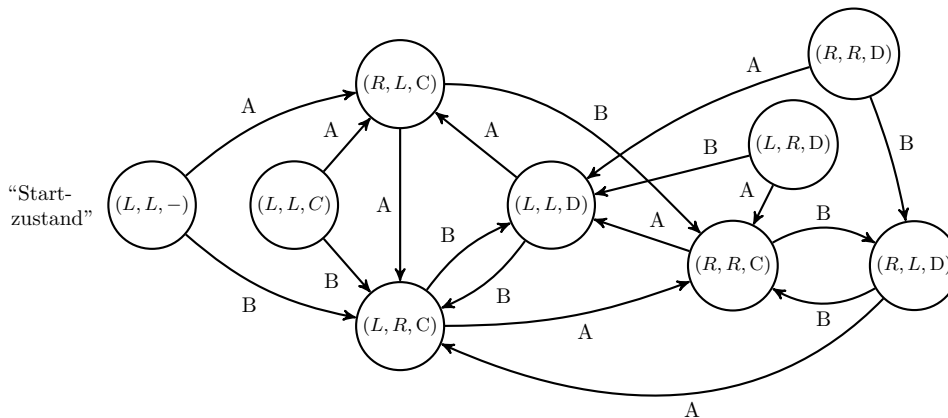


Abbildung 1.2: Übergänge zwischen den Zuständen beim Problem “Murmeln”

Um diese Frage zu beantworten muss man einfach nachprüfen, ob es in Abbildung 1.2 einen Weg vom Startzustand zum Zustand (R, R, D) gibt. Man sieht leicht, dass es in Abbildung 1.2 keinen solchen Weg gibt. Folglich lautet die korrekte Antwort auf obige Frage “nein”.

□ Ende Beispiel 1.1

Anmerkung:

Wir haben hier den Kalkül der **Transitionssysteme** (auch bekannt als **endliche Automaten** bzw. **Zustandsübergangsdiagramme** oder **Statecharts**) benutzt. Dieser Kalkül eignet sich besonders gut, wenn Abläufe in Systemen mit Übergängen zwischen verschiedenen Zuständen beschrieben werden sollen. Mehr dazu findet sich in Kapitel 7.

Wir betrachten ein weiteres Beispiel, das auf ähnliche Art gelöst werden kann.

Beispiel 1.2 (Problem “Flussüberquerung”).

Ein Mann steht mit einem Wolf, einer Ziege und einem Kohlkopf am linken Ufer eines Flusses, den er mit allen drei überqueren will. Er hat ein Boot, das gerade groß genug ist, ihn und ein weiteres Objekt zu transportieren, so dass er immer nur eines der drei mit sich hinübernehmen kann. Falls der Mann allerdings den Wolf mit der Ziege oder die Ziege mit dem Kohlkopf unbewacht an einem Ufer zurück lässt, wird die Ziege bzw. der Kohlkopf gefressen. *Frage:* Ist es möglich, den Fluss zu überqueren, ohne dass die Ziege oder der Kohlkopf gefressen wird?

Erste Analyse des Problems:

- *relevante Objekte:*
Mann, Wolf, Ziege, Kohlkopf, Boot, Fluss, Ufer (links und rechts)
- *Eigenschaften/Beziehungen:*
 - Das Boot trägt den Mann und zusätzlich maximal ein weiteres Objekt
 - Der Wolf frisst die Ziege, falls beide unbewacht am gleichen Ufer zurückgelassen werden.

Die Ziege frisst den Kohlkopf, falls beide unbewacht am gleichen Ufer zurückgelassen werden.

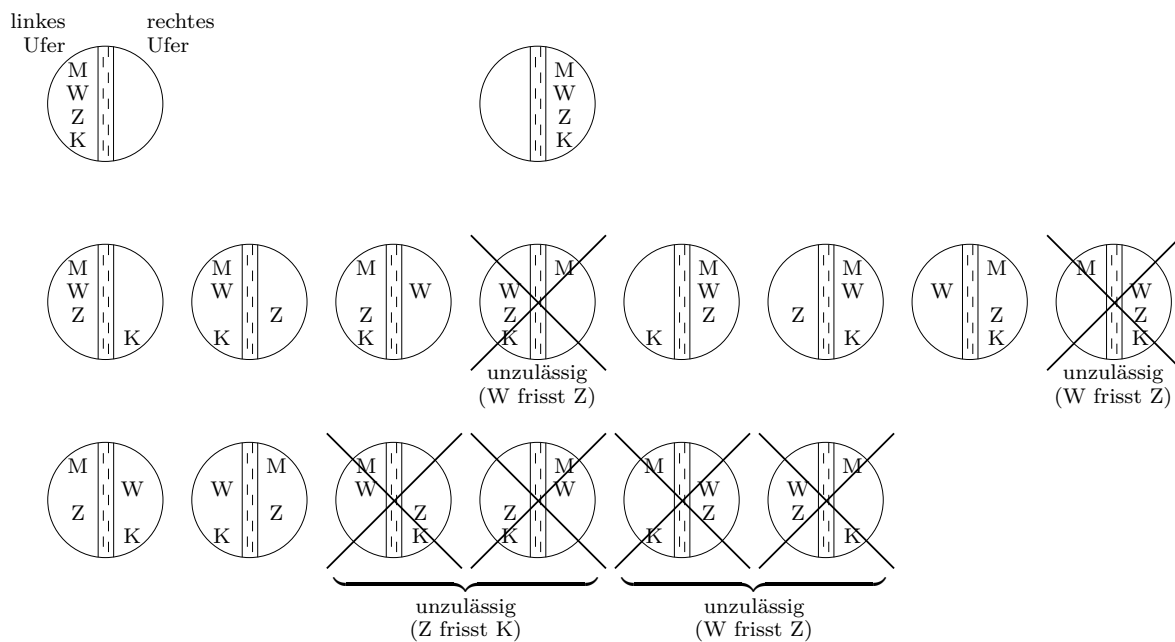
- *Tätigkeit:*
Überqueren des Flusses
- *Start:*
Mann, Wolf, Ziege, Kohlkopf (und Boot) am linken Ufer
- *Ziel:*
Mann, Wolf, Ziege, Kohlkopf (und Boot) am rechten Ufer

Abstraktionen:

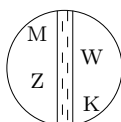
1. Nutze Abkürzungen:

- $M \hat{=}$ Mann
- $W \hat{=}$ Wolf
- $Z \hat{=}$ Ziege
- $K \hat{=}$ Kohlkopf

2. Betrachte die möglichen "Zustände", die auftreten dürfen:



3. Formale Modellierung der "Zustände": Repräsentiere den "Zustand"



durch das Tupel $(\{M, Z\}, \{W, K\})$.

Allgemein wird ein Zustand repräsentiert durch ein Tupel (ℓ, r) mit $\ell \subseteq \{M, Z, W, K\}$ und $r \subseteq \{M, Z, W, K\}$, für das folgende Bedingungen erfüllt sind:

- $\ell \cup r = \{M, Z, W, K\}$
- $\ell \cap r = \emptyset$
- falls $Z, K \in \ell$, so auch $M \in \ell$ (um zu verhindern, dass K von Z gefressen wird)
- falls $Z, K \in r$, so auch $M \in r$ (um zu verhindern, dass K von Z gefressen wird)
- falls $W, Z \in \ell$, so auch $M \in \ell$ (um zu verhindern, dass Z von W gefressen wird)
- (*) • falls $W, Z \in r$, so auch $M \in r$ (um zu verhindern, dass Z von W gefressen wird)

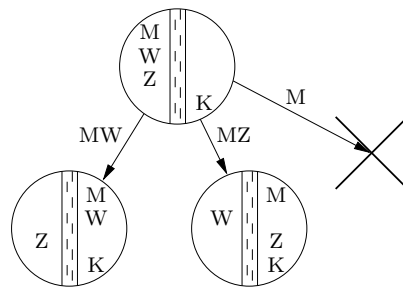
Übergänge von einem Zustand in einen anderen Zustand:

Vom Zustand $(\{M, W, Z\}, \{K\})$ aus kann man durch eine einzige Flussüberquerung in folgende Zustände gelangen:

- $(\{Z\}, \{M, W, K\})$, indem M und W im Boot fahren
- $(\{W\}, \{M, Z, K\})$, indem M und Z im Boot fahren

Beachte: wenn M allein fährt, tritt die Situation $(\{W, Z\}, \{M, K\})$ auf – dies ist aber laut (*) kein zulässiger Zustand.

Graphische Darstellung:



Insgesamt ergibt sich das in Abbildung 1.3 dargestellte Bild aus Zuständen und Zustandsübergängen.

Lösung des Problems “Flussüberquerung”:

An diesem Bild lässt sich unser ursprüngliches Problem “Flussüberquerung” (Frage: Ist es möglich, den Fluss zu überqueren, ohne dass die Ziege oder der Kohlkopf gefressen werden?) leicht lösen, indem man einfach einen Weg vom “Startzustand” zum “Zielzustand” sucht. In Abbildung 1.3 gibt es zwei verschiedene solche Wege, die jeweils mit 7 Überfahrten auskommen.

□ Ende Beispiel 1.2

Diskussion dieser beiden Modellierungsbeispiele:

Abläufe bzw. Folgen von Schritten wurden hier durch ein Zustandsübergangsdiagramm modelliert. Die **Abstraktion** bestand darin, nur die Zustände und deren Übergänge zu betrachten. Die **relevanten Objekte** wurden identifiziert: Beim “Murmelpfand” waren dies die aktuellen

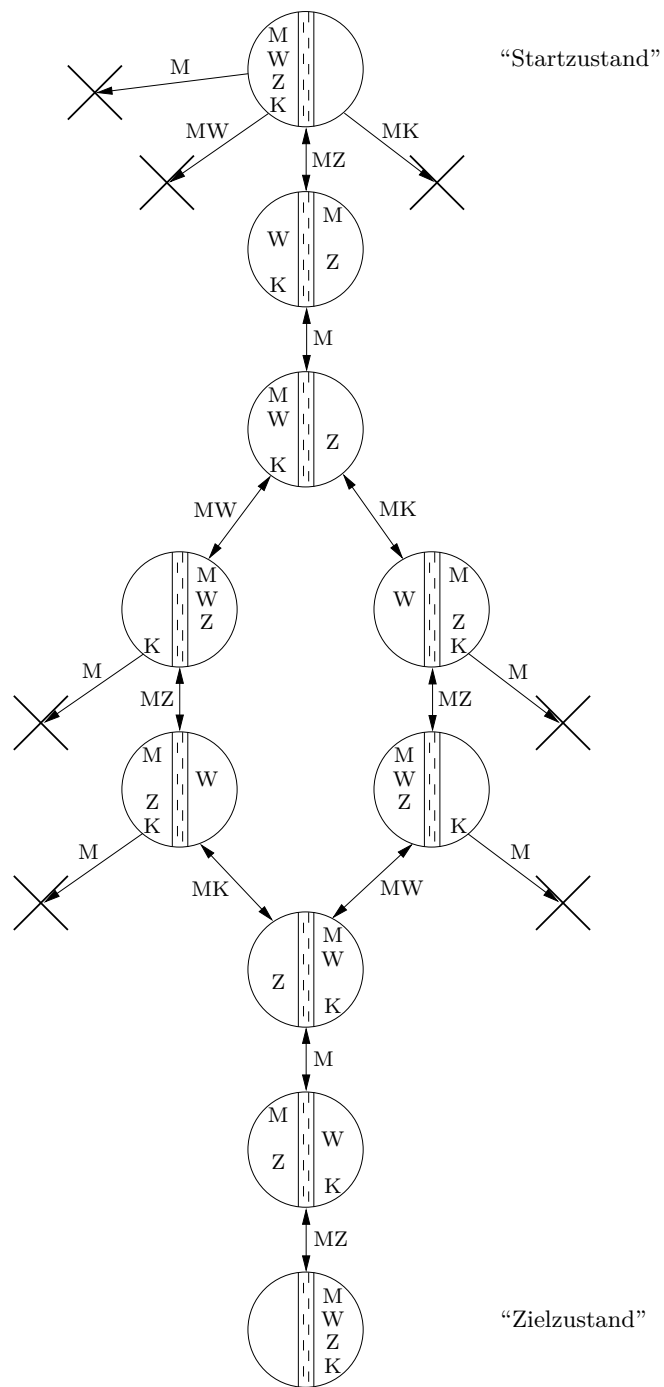


Abbildung 1.3: Übergänge zwischen den Zuständen beim Flussüberquerungsproblem

Positionen der beiden Hebel (jeweils L oder R) sowie der Ausgang, an dem die letzte Murmel herausgerollt ist (C oder D). Beim “Flussüberquerungsproblem” waren dies die Positionen von M , W , Z , K (jeweils am linken oder am rechten Ufer). Jeden **Zustand** haben wir **repräsentiert** durch ein Tupel $((x, y, z)$ im “Murmelproblem” bzw. (ℓ, r) im “Flussüberquerungsproblem”). Die möglichen Tupel wurden in **zulässige Zustände** und **unzulässige Zustände** eingeteilt. Übergänge von einem Zustand zu einem anderen Zustand wurden mit den jeweiligen Aktionen beschriftet (dem Eingang, an dem die nächste Murmel eingeworfen wird bzw. mit den Objekten, die als nächstes über den Fluss transportiert werden).

Besonders wichtig ist auch, dass Aspekte, die zur Lösung der Aufgabe irrelevant sind **nicht** modelliert wurden (beim “Murmelproblem” z.B. die genaue Anordnung der Spielbahn, die Gesetze der Schwerkraft und die Kräfte, die mechanisch auf die beiden Hebel wirken; beim “Flussüberquerungsproblem” z.B. Name, Breite, Tiefe des Flusses oder Länge, Geschwindigkeit des Boots etc.).

Die “**Kreative Leistung**”, die hier zur Lösung der beiden Probleme geleistet wurde, war, den Kalkül der Zustandsübergangsdigramme zu wählen und die Bedeutung der Zustände und Übergänge festlegen. Das konkrete Zustandsübergangsdigramm aufzustellen und einen Weg mit der entsprechenden Beschriftung (beim “Murmelproblem”) bzw. einen Weg vom Start- zum Zielzustand (beim “Flussüberquerungsproblem”) zu finden, war dann reine “**Routine-Arbeit**”.

Im Verlauf der Veranstaltung “Diskrete Modellierung” werden wir verschiedene Kalküle kennenlernen, die zur Lösung typischer Informatik-Probleme besonders geeignet sind.

1.2 Ziele der Veranstaltung “Diskrete Modellierung”

Ziel der Veranstaltung “Diskrete Modellierung” ist, einen Überblick über grundlegende Modellierungsmethoden und -kalküle zu geben — insbesondere über Aussagenlogik (Kapitel 3), Graphen und Bäume (Kapitel 4), Markov-Ketten (Kapitel 5), Logik erster Stufe / Prädikatenlogik (Kapitel 6), Transitionssysteme / endliche Automaten (Kapitel 7), Petri-Netze (Kapitel 7), kontextfreie Grammatiken (Kapitel 8) und das Entity-Relationship-Modell (Kapitel 8).

Weitere Ziele sind:

- das Verständnis des konzeptionellen Kerns der Kalküle,
- die Fähigkeit, die Kalküle an typischen Beispielen anzuwenden,
- die Fähigkeit zur präzisen und formalen Ausdrucksweise bei der Analyse von Problemen — dazu gehört auch das Verständnis und der souveräne Umgang mit mathematische Grundlagen und Beweistechniken (Kapitel 2),
- die Erkenntnis des praktischen Wertes präziser Beschreibungen.

1.3 Der Begriff “Diskrete Modellierung”

Einträge in Duden, Deutsches Universalwörterbuch (4. Auflage, 2001):

diskret <Adj.> [1,2: frz. discret < mlat. discretus = abgesondert, zu lat. discernere = absondern, unterscheiden; 3: engl. discrete] (bildungsspr.): diskret

1.a) *so unauffällig behandelt, ausgeführt, dass es von anderen nicht bemerkt wird; vertraulich: -e*

Spenden an die Parteien; eine heikle Angelegenheit d. behandeln; **b)** *taktvoll, rücksichtsvoll*: ein -es Verhalten; eine Peinlichkeit d. übergehen; d. schweigen; etw. d. übersehen; **c)** *unaufdringlich; zurückhaltend; dezent*: ein -es Parfüm, Muster; d. angezogen. **2.** (Technik, Physik, Math.) *durch endliche Intervalle od. Abstände voneinander getrennt*: -e Halbleiter, Bauteile; eine -e (*nicht integrierte*) Schaltung **3.** (Sprachw.) (*von sprachlichen Einheiten*) *abgrenzbar; abgesondert; unterschieden*.

Modell

Modell, das; -s, -e [ital. modello = Muster, Entwurf, zu lat. modulus, ↑¹Modul]: **1.a)** *Form, Beschaffenheit, Maßverhältnisse veranschaulichende Ausführung eines vorhandenen od. noch zu schaffenden Gegenstandes in bestimmtem (bes. verkleinerndem) Maßstab*: das M. eines Schiffes, Flugzeugs, einer Burg, Fabrik; ein M. entwerfen, bauen; **b)** (Technik, bild. Kunst) *Muster, Entwurf einer Plastik, eines technischen o.ä., durch Guss herzustellenden Gegenstandes, nach dem die Guss- bzw. Gipsform hergestellt wird*: das M. einer Plastik; **c)** (Wissensch.) *innere Beziehungen u. Funktionen von etw., abbildendes bzw. [schematisch] veranschaulichendes [u. vereinfachendes, idealisierendes] Objekt, Gebilde*: ein M. des Atomkerns; **d)** (math. Logik) *Interpretation eines Axiomensystems, nach der alle Axiome des Systems wahre Aussagen sind*. **2.a)** *als Gegenstand der bildnerischen, künstlerischen o.ä. Darstellung od. Gestaltung benutztes Objekt, Lebewesen usw.*; **b)** *Person, die sich [berufsmäßig] als Gegenstand bildnerischer od. fotografischer Darstellung, Gestaltung zur Verfügung stellt*: als M. arbeiten; ***[jmdm.] M. sitzen/stehen** (*jmds. Modell sein*): sie hat dem Maler für dieses Bild M. gesessen; **c)** ²*Model* (a); **d)** (verhüll.) *Hostess* (3). **3.a)** (*Gegenstand als*) *Entwurf, Muster, Vorlage für die serienweise Herstellung von etw.*; **b)** *Typ, Art der Ausführung eines Fabrikats*; **c)** (Rechtspr.) *durch Gesetz urheberrechtlich geschützte Gestaltungsform eines Gebrauchsgegenstandes*. **4.** (Mode) *[Kleidungs]stück, das eine Einzelanfertigung ist [u. ungefähr als Muster, Vorlage od. Anhaltspunkt für die serienweise Herstellung bzw. Konfektion dienen kann]*: ein Pariser M. **5.** (bildungsspr.) **a)** *etw., was (durch den Grad seiner Perfektion, Vorbildlichkeit o.Ä.) für anderes od. für andere Vorbild, Beispiel, Muster sein kann*: etw. nach dem M. von etw. gestalten; **b)** *als Muster gedachter Entwurf*: das M. eines neuen Gesetzes.

Eintrag in Dictionary of computer science, engineering, and technology, CRC Press, 2001:

model

model (1) a representation of reality of an artifact or activity intended to explain the behaviour of some aspects of it. In creating a model, an abstraction technique is used. Thus, the model is typically less complex or complete than the reality modeled and can be regarded as an abstract description. This technique identifies commonalities and, in doing so, loses details.

(2) a mathematical or schematic description of a computer or network system. Modeling usually involves an act of abstraction; i.e., the model only includes the most important properties of the original system.

Eintrag in Duden Informatik – Ein Fachlexikon für Studium und Praxis, 3. Auflage, 2001:

Modell

Modell: Abbild von etwas, oft unter Weglassen von Details, also im Sinne einer vereinfachenden Darstellung. Für die Entwicklung von (Hard- und Software-)Systemen ist die Modellbildung und der sich anschließende Entwurf einer Architektur von zentraler Bedeutung.

(1) Jede wirklichkeitsbezogene Anwendung der Datenverarbeitung basiert auf einem Modell, das einen Teil der Wirklichkeit angenähert widerspiegelt. Da die Wirklichkeit viel zu komplex ist, um sie direkt im Rechner wiedergeben zu können, werden die für die jeweilige Anwendung relevanten Anteile herauskristallisiert, analysiert, zu einer Struktur zusammengefügt und durch ein

Modell, also durch entsprechende konkrete Datentypen, Objekte, Attribute, Operationen und deren Beziehungen untereinander dargestellt. Die Darstellung erfolgt meist auf der abstrakteren Ebene der \uparrow Klassen und deren hierarchischen Beziehungen (Vererbung), der Abbildungen, die die wechselseitigen Abhängigkeiten beschreiben, und der Architekturen, durch die diese Größen miteinander verknüpft werden. Meist fügt man noch Anforderungen hinzu, die sich auf zu erfüllende Eigenschaften, auf einzuhaltende Rahmenbedingungen usw. beziehen.

Die Erstellung eines Modells zu einer Anwendung bezeichnet man als *Modellierung*. Sie wird heute meist objektorientiert durchgeführt und mithilfe vorhandener Klassenbibliotheken realisiert, in denen entsprechende „Oberklassen“ für viele Anwendungen bereits abgelegt sind. Diese Wiederverwendung erprobter und ausgetesteter Klassen macht eine Stärke des objektorientierten Entwurfs aus. Zur Unterstützung der Modellierung gibt es Schemata und Beschreibungssprachen, z.B. das \uparrow Entity-Relationship-Modell oder die \uparrow UML.

Modelle können sehr abstrakt sein (z.B. Darstellungen durch Grammatiken, Graphen, Automaten) und Gesetzmäßigkeiten enthalten, oder sie können sehr nahe an der Wirklichkeit sein und vornehmlich der Veranschaulichung von Sachverhalten dienen. Auf dem Modell setzen dann Verarbeitungsalgorithmen und Simulationen auf. Die Güte eines Modells ergibt sich daraus, inwieweit die Ergebnisse der Berechnungen mit der Wirklichkeit übereinstimmen.

In der Informatik werden meist diskrete Modelle betrachtet. In den Natur- und Ingenieurwissenschaften verwendet man vorwiegend kontinuierliche Modelle, zum Beispiel auf der Basis von Differenzialgleichungen. Mischformen bezeichnet man als hybride Modelle.

(2) Mengen mit Operationen, die eine logische Formel (\uparrow Logik) oder die Gesetze eines \uparrow Datentyps erfüllen.

Zusammenfassung:

Der Begriff „Modell“ wird in verschiedenen Zusammenhängen mit unterschiedlichen Bedeutungen verwendet. Ein Modell kann ein Abbild eines vorhandenen oder noch zu schaffenden Originals sein (z.B. ein Modellflugzeug oder ein Gebäude in kleinem Maßstab), es kann aber auch zur Repräsentation einiger Aspekte der realen Welt dienen (z.B. verschiedene Atommodelle).

Modelle werden u.a. benutzt, um

- ein konkretes Problem zu lösen (z.B. Beispiel 1.1 „Murmelpfand“),
- bestimmte Aspekte eines komplexen Gebildes zu untersuchen, zu verstehen oder zu vermitteln (z.B. Geschäftsabläufe in einer Firma),
- die Kommunikation zwischen einem Auftraggeber und einem Hersteller des Originals zu vereinfachen (z.B. beim Bau eines Hauses oder bei der Software-Entwicklung),
- Anforderungen für die Herstellung des Originals zu fixieren (z.B. Spezifikation von und Anforderungen an Software),
- Operationen durchzuführen, die man am Original nicht durchführen kann (z.B. Computer-Simulation dessen, was bei einem Flugzeugabsturz über einem Kernkraftwerk passieren könnte),
- ein Modell zu validieren (engl. *Model Checking*), d.h. um nachzuweisen, dass die relevanten Eigenschaften des Originals korrekt und vollständig im Modell erfasst sind (z.B. zur Prüfung, ob ein Finanzplan alle Kosten erfasst, sie korrekt aufsummiert und die vorgegebene Kostengrenze eingehalten wird).

Modelle sind **absichtlich** nicht originalgetreu. Sie heben bestimmte Eigenschaften hervor und lassen andere weg. Der konkrete Verwendungszweck des Modells bestimmt, welche Eigenschaften modelliert werden und welcher Kalkül dafür besonders geeignet ist.

Ein Modell beschreibt stets nur einige bestimmte Aspekte des Originals, etwa

- die Struktur oder die Zusammensetzung des Originals (z.B. das Organigramm einer Firma).
Dafür geeignete Kalküle sind z.B. Wertebereiche, Entity-Relationship-Modell, Bäume, Graphen.
- Eigenschaften von Teilen des Originals (z.B. Farbe und Wert einer Spielkarte).
Dafür geeignete Kalküle sind z.B. Wertebereiche, Logik, Entity-Relationship-Modell.
- Beziehungen zwischen Teilen des Originals (z.B. “Wolf frisst Ziege, Ziege frisst Kohlkopf” in Beispiel 1.2).
Dafür geeignete Kalküle sind z.B. Graphen, Logik, Entity-Relationship-Modell.
- Verhalten des Originals unter Operationen (z.B. aufeinanderfolgende Zustände bei wiederholtem Einwurf von Murmeln in Beispiel 1.1).
Dafür geeignete Kalküle sind z.B. Zustandsübergangsdigramme, Petri-Netze, Graphen.

Beispiel 1.3.

(a) Unterschiedliche Modelle, die beim Bau eines Hauses verwendet werden:

- Gebäudemodell: zur Vermittlung eines optischen Eindrucks
- Grundriss: zur Einteilung der Räume und des Grundstückes
- Kostenplan: zur Finanzierung

(b) Frankfurter S- und U-Bahn Netzplan: siehe Abbildung 1.4

Ziel: Beschreibung, welche Haltestellen von welchen Linien angefahren werden und welche Umsteigemöglichkeiten es gibt

Vernachlässigt: genauere topografische Informationen (Entfernung, genaue Lage, Straßenverläufe etc.), Abfahrtszeiten

(c) Fahrplan der U4 an der Haltestelle “Bockenheimer Warte”: siehe Abbildung 1.5

Ziel: Angabe der Abfahrtszeiten der U4 an der Haltestelle “Bockenheimer Warte” sowie Informationen darüber, wie viele Minuten die U4 von dort bis zu anderen Haltestellen auf ihrer Strecke braucht.

□ Ende Beispiel 1.3

1.4 Literaturhinweise zu Kapitel 1

Als vertiefende Lektüre sei Kapitel 1 in [15] empfohlen.

Quellennachweis: Teile dieses Kapitels orientieren sich an Kapitel 1 in [15]; insbesondere Beispiel 1.3 sowie das in Beispiel 1.2 betrachtete “Flussüberquerungsproblem” sind im Wesentlichen aus [15] übernommen. Das “Flussüberquerungsproblem” findet sich bereits in [12]; das “Murmelpuzzle” aus Beispiel 1.1 ist eine vereinfachte Variante von Aufgabe 2.3 in [12].

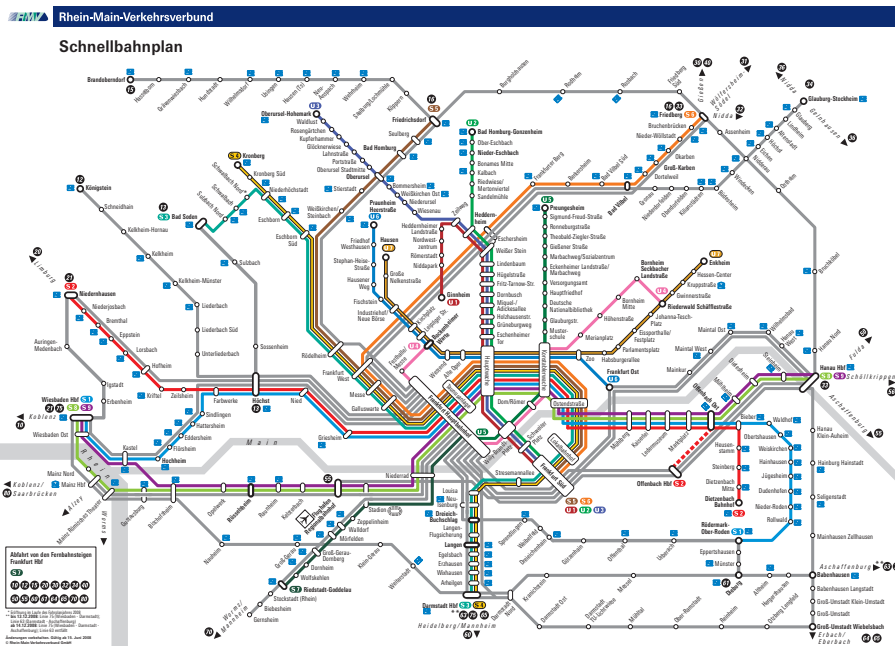


Abbildung 1.4: Schnellbahnplan des Rhein-Main-Verkehrsverbundes (RMV)



Den Namen des Verkehrsunternehmens, mit dem Sie auf dieser Linie den Beförderungsvertrag schließen, entnehmen Sie bitte dem Aushangfahrplan an der Haltestelle oder dem Fahrplanbuch.

U U4

Frankfurt (Main) Schöfflestraße



gültig vom 05.07.2008 bis 13.12.2008

Die RMV-Fahrplanauskunft wird täglich aktualisiert. Sie erhalten somit den jeweils uns bekannten aktuellen Stand. Beeinträchtigungen auf der Strecke und Sonderverkehre können zu Abweichungen vom Regelfahrplan führen. Hierüber informieren wir Sie gerne auch in unserem kostenlosen Newsletter. Oder besuchen Sie uns einfach auf www.rmv.de | Verkehrsmittele: Bus & Bahn aktuell.

Montag - Freitag			Samstag			Sonntag*		
04	18	38 58	04	18	38 58	04	18	38 58
05	18	28 ^A 38 48 ^A 58	05	18	38 58	05	18	38 58
06	08 ^A 18 28 38 ^A 45 53 ^A	06	18	38 58	06	18	38 58	
07	00 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	07	08 ^A 18 28 ^A 38 48 ^A 58	07	18	38 58		
08	00 ^b 03 ^a 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	08	08 ^A 18 28 ^A 38 48 ^A 58	08	08 ^A 18 28 ^A 38 48 ^A 58			
09	00 ^b 03 ^a 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	09	08 ^A 18 28 ^A 38 48 ^A 58	09	08 ^A 18 28 ^A 38 48 ^A 58			
10	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	10	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	10	08 ^A 18 28 ^A 38 48 ^A 58			
11	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	11	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	11	08 ^A 18 28 ^A 38 48 ^A 58			
12	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	12	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	12	08 ^A 18 28 ^A 38 48 ^A 58			
13	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	13	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	13	08 ^A 18 28 ^A 38 48 ^A 58			
14	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	14	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	14	08 ^A 18 28 ^A 38 48 ^A 58			
15	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A 58 ^a	15	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	15	08 ^A 18 28 ^A 38 48 ^A 58			
16	00 ^b 03 ^a 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	16	00 08 ^A 15 23 ^A 30 38 ^A 45 53 ^A	16	08 ^A 18 28 ^A 38 48 ^A 58			
17	00 ^b 03 ^a 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	17	00 08 ^A 15 23 ^A 30 38 48 ^A 58	17	08 ^A 18 28 ^A 38 48 ^A 58			
18	00 ^b 03 ^a 08 ^A 13 ^A 15 ^b 18 ^a 23 ^A 28 ^A 30 ^b 33 ^a 38 ^A 43 ^A 45 ^b 48 ^a 53 ^A 58 ^a	18	08 ^A 18 28 ^A 38 48 ^A 58	18	08 ^A 18 28 ^A 38 48 ^A 58			

MS/SP/PT 3.2 Alle Angaben ohne Gewähr © Rhein-Main-Verkehrsverbund

Hotline (0,14 €/Minute)* **01805/768 4636**
 Internet www.rmv.de WAP-Service wap.rmv.de Beratung vor Ort **Mobilitätszentralen**

Abbildung 1.5: Fahrplan der U4 an der Haltestelle "Bockenheimer Warte"

1.5 Übungsaufgaben zu Kapitel 1

Aufgabe 1.1. Gegeben seien drei Stapel mit Büchern. Der erste besteht aus vier Büchern, der zweite aus sechs Büchern und der dritte aus 14 Büchern. Die Stapel sollen nun ausgeglichen werden, so dass auf jedem Stapel acht Bücher liegen. Allerdings dürfen in jedem Schritt nur Bücher zwischen genau zwei Stapeln umgeschichtet werden. Zudem können auf jeden Stapel immer nur so viele Bücher gelegt werden, wie bereits darauf liegen.

- (a) Lassen sich die Stapel wie gewünscht ausgleichen? Modellieren Sie zur Beantwortung dieser Frage das Problem analog zum Beispiel 1.2.
- (b) Nehmen wir nun an, dass der erste Stapel aus vier Büchern, der zweite aus sechs Büchern und der dritte aus acht Büchern besteht. Lassen sich die Stapel so ausgleichen, dass auf jedem Stapel sechs Bücher liegen?

Hinweis: Es brauchen nur diejenigen Zustände betrachtet zu werden, die man vom Startzustand aus durch geeignete Zustandsübergänge erreichen kann.

Aufgabe 1.2. In dieser Aufgabe betrachten wir eine Variante des unter dem Namen *Nim* bekannten Spiels, die wie folgt definiert ist: Es gibt zwei Spieler namens Alice und Bob. Zu Beginn des Spiels liegen fünf Hölzer auf dem Tisch. Die beiden Spieler sind abwechselnd am Zug. In jedem Zug kann der Spieler, der gerade an der Reihe ist, entscheiden, ob er ein Holz oder zwei Hölzer vom Tisch wegnimmt. Der Spieler, der das letzte Holz vom Tisch nimmt, verliert das Spiel. Zu Beginn ist Alice am Zug.

Modellieren Sie zur Beantwortung der folgenden Fragen das Spiel analog zum Beispiel 1.1 aus der Vorlesung durch ein Transitionssystem. Überlegen Sie sich zunächst, welche Zustände und Zustandsübergänge auftreten können.

Hinweis: Jeder Zustand des Transitionssystems sollte Informationen darüber enthalten, welcher Spieler am Zug ist und wie viele Hölzer noch auf dem Tisch liegen.

- (a) Ist es eine gute Idee für Alice, im ersten Zug zwei Hölzer zu nehmen?
- (b) Eine Gewinnstrategie für einen Spieler in diesem Spiel ist eine Vorschrift, welche ihm sagt, welchen Zug er als nächstes tätigen soll. Hält sich der Spieler an diese Vorschrift, so gewinnt er auf jeden Fall. Existiert in diesem Spiel eine Gewinnstrategie für Alice?
- (c) Existiert eine Gewinnstrategie für Bob?

Aufgabe 1.3. In dieser Aufgabe betrachten wir das Spiel *Dinn*¹, das wie folgt definiert ist: Zwei Spieler, Alice und Bob, spielen gegeneinander. Zu Beginn des Spiels liegen neun Hölzer auf dem Tisch, zusätzlich besitzen Alice und Bob jeweils noch eine unbegrenzte Anzahl von Hölzern. Die beiden Spieler sind abwechselnd am Zug, Alice beginnt. In jedem Zug i kann der Spieler, der gerade an der Reihe ist, entweder i Hölzer zusätzlich auf den Tisch legen oder i Hölzer vom Tisch entfernen, wenn dort noch mindestens i Hölzer liegen. So kann Alice im ersten Zug ein Holz zu den Hölzern auf dem Tisch hinzufügen oder davon entfernen. Im zweiten Zug kann Bob entscheiden, ob er zwei Hölzer entfernt oder zwei hinzufügt und so fort. Es gewinnt der Spieler, der eine Anzahl von Hölzern auf dem Tisch hinterlässt, die eine Primzahl ist. (Achtung: 1 ist keine Primzahl.)

¹rekursives Akronym für *Dinn ist nicht Nim*.

Modellieren Sie zur Beantwortung der folgenden Fragen das Spiel analog zum Beispiel 1.1 aus der Vorlesung durch ein Transitionssystem. Überlegen Sie sich zunächst, welche Zustände und Zustandsübergänge auftreten können.

- (a) Ist es eine gute Idee für Alice, im ersten Zug ein Holz auf den Tisch zu legen?
- (b) Eine Gewinnstrategie für einen Spieler in diesem Spiel ist eine Vorschrift, die ihm sagt, welchen Zug er als nächstes tätigen soll. Hält sich der Spieler an diese Vorschrift, so gewinnt er auf jeden Fall. Existiert in diesem Spiel eine Gewinnstrategie für Alice?
- (c) Existiert eine Gewinnstrategie für Bob?

Aufgabe 1.4. In dieser Aufgabe betrachten wir das Spiel *NimHalbe*, das wie folgt definiert ist: Zwei Spieler, Alice und Bob, spielen gegeneinander. Zu Beginn des Spiels liegen 1336 Hölzer auf dem Tisch. Die beiden Spieler sind abwechselnd am Zug, Alice beginnt. In jedem Zug kann der Spieler, der gerade an der Reihe ist, entweder drei Hölzer vom Tisch entfernen oder, falls eine gerade Anzahl an Hölzern auf dem Tisch liegt, den Haufen halbieren, also die Hälfte der Hölzer vom Stapel nehmen.

So kann Alice im ersten Zug drei Hölzer vom Tisch nehmen (es verbleiben 1333) oder den Stapel halbieren (es verbleiben 668). Hiernach ist Bob am Zug und kann bei 668 Hölzern wieder zwischen beiden Optionen wählen; bei 1333 Hölzern ist er gezwungen, drei davon wegzunehmen. Es gewinnt der Spieler, der eine Anzahl von Hölzern auf dem Tisch hinterlässt, die eine Primzahl ist. (Achtung: 1 ist keine Primzahl.)

Modellieren Sie zur Beantwortung der folgenden Fragen das Spiel analog zum Beispiel 1.1 aus der Vorlesung durch ein Transitionssystem.

- (a) Ist es eine gute Idee für Alice, im ersten Zug gleich den Stapel zu halbieren ?
- (b) Eine Gewinnstrategie für einen Spieler in diesem Spiel ist eine Vorschrift, die ihm sagt, welchen Zug er als nächstes tätigen soll. Hält sich der Spieler an diese Vorschrift, so gewinnt er auf jeden Fall. Existiert in diesem Spiel eine Gewinnstrategie für Alice?
- (c) Existiert eine Gewinnstrategie für Bob?

Aufgabe 1.5. Der Polizist John McClane hat ein Déjà-vu:² Schon wieder steht er knietief in einem Brunnen und soll mithilfe von zwei Gefäßen eine bestimmte Menge Wasser daraus schöpfen. Immerhin ist die Aufgabe seit dem letzten Mal etwas einfacher geworden (Bruce ist ja auch nicht mehr der Jüngste). John startet mit zwei leeren Gefäßen, eines davon fasst genau einen Liter, das andere genau drei. Ziel ist es, in beiden Gefäßen jeweils genau die Menge von einem Liter Wasser zu haben. Da sich an den Gefäßen keine Markierungen befinden, kann John sein Ziel nur durch eine schrittweise Ausführung der folgenden Aktionen erreichen: Er kann eines der Gefäße vollständig mit Wasser aus dem Brunnen befüllen, eines der Gefäße komplett auskippen oder Wasser eines Gefäßes in das andere kippen bis eines der Gefäße voll oder leer ist. Da John es hasst, von vorne zu beginnen, wird er außerdem niemals in den Startzustand mit zwei leeren Gefäßen zurückkehren.

Modellieren Sie zur Beantwortung der folgenden Fragen das Problem durch ein Transitionssystem analog zum „Murmelbeispiel“ aus der Vorlesung.

- (a) Kann John McClane sein Ziel erreichen?

²Die Figur John McClane und die Situation in abgewandelter Form ist dem Film „Stirb langsam 3“ entnommen.

- (b) Ist es möglich, dass John sich durch eine ungeschickte Abfolge von Schritten (aber unter Berücksichtigung der Regeln) in eine Situation bringt aus der er das Ziel nicht mehr erreichen kann?
- (c) Für John ist es unerträglich, nicht voran zu kommen. Er möchte deshalb niemals eine gerade getätigte Aktion direkt wieder rückgängig machen. Nehmen Sie nun an, dass John keine Aktion direkt wieder rückgängig macht und ansonsten (unter Berücksichtigung der Regeln) wahllos vorgeht. Wird er dann zwangsläufig irgendwann in den Zustand kommen, in dem er sein Ziel erreicht hat?

2 Mathematische Grundlagen und Beweistechniken

Mathematische Notationen:

Symbol	Bedeutung
$:=$	Definition eines Wertes, z.B. $x := 5$, $M := \{1, 2, 3\}$
$:\Leftrightarrow$	Definition einer Eigenschaft oder einer Schreibweise z.B. $m \in M :\Leftrightarrow m$ ist Element von M
ex.	Abkürzung für “es gibt”, “es existiert”
f.a.	Abkürzung für “für alle”, “für jedes”
s.d.	Abkürzung für “so, dass”
\Rightarrow	Abkürzung für “impliziert” z.B.: Regen \Rightarrow nasse Straße
\Leftrightarrow	Abkürzung für “genau dann, wenn” z.B.: Klausur bestanden \Leftrightarrow die erreichte Prozentzahl z ist $\geq 50\%$
\square	markiert das Ende eines Beweises

Modellierung und Wertebereiche

In der Modellierung von Systemen, Aufgaben, Problemen oder Lösungen kommen **Objekte unterschiedlicher Art und Zusammensetzung** vor. Für Teile des Modells wird angegeben, aus welchem Wertebereich sie stammen, es wird zumeist aber offen gelassen, welchen konkreten Wert sie annehmen.

Wertebereich

Ein **Wertebereich** ist eine Menge gleichartiger Werte. Wertebereiche werden aus Mengen und Strukturen darüber gebildet.

Beispiel 2.1 (Modellierung der Karten eines (Skat-)Kartenspiels).

Die Karten eines Skat-Kartenspiels lassen sich durch folgende Wertebereiche darstellen:

$$\begin{aligned}\text{KartenArten} &:= \{ \text{Kreuz, Pik, Herz, Karo} \} \\ \text{KartenSymbole} &:= \{ 7, 8, 9, 10, \text{Bube, Dame, König, Ass} \} \\ \text{Karten} &:= \{ (\text{Kreuz}, 7), (\text{Kreuz}, 8), \dots, (\text{Kreuz}, \text{Ass}), \\ &\quad (\text{Pik}, 7), (\text{Pik}, 8), \dots, (\text{Pik}, \text{Ass}), \\ &\quad (\text{Herz}, 7), (\text{Herz}, 8), \dots, (\text{Herz}, \text{Ass}), \\ &\quad (\text{Karo}, 7), (\text{Karo}, 8), \dots, (\text{Karo}, \text{Ass}) \}.\end{aligned}$$

□Ende Beispiel 2.1

Wertebereiche sind u.a. wichtig

- zur Modellierung von Strukturen und Zusammenhängen,
- als Grundlage für alle anderen formalen Kalküle und
- als abstrakte Grundlage für Typen in Programmiersprachen.

Der grundlegende Kalkül zur Handhabung von Wertebereichen ist die **Mengenlehre**, bei der Mengen und Mengenoperationen betrachtet werden. Zur Modellierung von “zusammengesetzten Wertebereichen” kann man z.B.

- Potenzmengen,
- kartesische Produkte und Tupel,
- Relationen,
- Folgen bzw. Wörter und
- Funktionen

nutzen. Ziel von Kapitel 2 ist, diese Begriffe zu präzisieren und darüber hinaus auch einige wichtige mathematische Grundlagen und Beweistechniken zu erklären.

2.1 Mengen

2.1.1 Was ist eine Menge?

Cantors naiver Mengenbegriff: (Georg Cantor, 1845–1918)

Cantors naiver Mengenbegriff besagt folgendes: Eine Menge M ist eine Zusammenfassung von bestimmten, wohlunterschiedenen Objekten unserer Anschauung oder unseres Denkens, welche “Elemente der Menge M ” genannt werden, zu einem Ganzen.

Wir schreiben

$$m \in M$$

um auszusagen, dass M eine Menge ist und dass m ein Element in der Menge M ist.

Wir schreiben

$$m \notin M$$

um auszusagen, dass m kein Element in der Menge M ist.

Künftig werden wir solche Notationen festlegen, indem wir kurz folgendes schreiben:

$m \in M$
 $m \notin M$

Notation:

$m \in M \iff m$ ist Element der Menge M .
 $m \notin M \iff m$ ist kein Element der Menge M .

Cantors Mengenbegriff ist problematisch und führt zu Widersprüchen. Russell gab folgendes Beispiel:

Die Russellsche Antinomie:

(Bertrand Russell, 1872–1970)

Sei N die Menge aller Mengen M , die sich nicht selbst enthalten
(d.h.: $M \in N \iff M$ ist eine Menge, für die gilt: $M \notin M$).

Frage: Enthält N sich selbst (d.h. gilt $N \in N$)?

Klar: Entweder es gilt $N \in N$ oder es gilt $N \notin N$.

Fall 1: $N \notin N$. Gemäß Definition der Menge N gilt dann, dass $N \in N$.
Das ist ein Widerspruch.

Fall 2: $N \in N$. Gemäß Definition der Menge N gilt dann, dass $N \notin N$.
Das ist ein Widerspruch.

Somit führen beide Fälle zu einem Widerspruch, obwohl wir wissen, dass einer der beiden Fälle zutreffen müsste.

Fazit: Irgendetwas stimmt nicht mit Cantors naivem Mengenbegriff!

Um Russells Beispiel und den daraus resultierenden Widerspruch besser zu verstehen, betrachte man folgende Geschichte vom Barbier von Sonnenthal.

Der Barbier von Sonnenthal:

Im Städtchen Sonnenthal (in dem bekanntlich viele seltsame Dinge passieren) wohnt ein Barbier, der genau diejenigen männlichen Einwohner von Sonnenthal rasiert, die sich nicht selbst rasieren.

Frage: Rasieret der Barbier sich selbst?

Um die Russellsche Antinomie zu vermeiden, muss man die Mengenlehre sehr sorgfältig axiomatisch aufbauen (siehe z.B. [6]) — dies sprengt allerdings den Rahmen dieser Vorlesung. Sofern man sich der Problematik bewusst ist, kann man sie im “täglichen Gebrauch”, den Informatiker/innen von Mengen machen, vermeiden. Wir arbeiten daher weiter mit einem naiven Mengenbegriff, den wir nach den im Folgenden beschriebenen Grundsätzen verwenden werden.

Beschreibung bzw. Definition von Mengen:

Wir beschreiben bzw. definieren Mengen

- *extensional*, durch Aufzählen der Elemente, z.B.

$$M_1 := \{0, 1, 2, 3, 4, 5\} = \{0, 1, 2, \dots, 5\}$$

oder

- *intensional*, durch Angabe von charakteristischen Eigenschaften der Elemente der Menge, z.B.

$$\begin{aligned} M_2 &:= \{x : x \in M_1 \text{ und } x \text{ ist gerade}\} \\ &= \{x \in M_1 : x \text{ ist gerade}\} \\ &= \{x : x \text{ ist eine natürliche Zahl und } x \text{ ist gerade und } 0 \leq x \leq 5\}. \end{aligned}$$

Extensional lässt sich die Menge M_2 folgendermaßen beschreiben:

$$M_2 = \{0, 2, 4\}.$$

Oft schreibt man statt “:” auch “|” und statt “und” einfach ein “Komma”, also

$$M_2 = \{x \mid x \in M_1, x \text{ gerade}\}.$$

Vorsicht:

- $\{x : 0 \leq x \leq 5\}$ definiert nicht eindeutig eine Menge, weil nicht festgelegt ist, ob x beispielsweise eine ganze Zahl oder eine reelle Zahl ist.
- $\{M : M \text{ ist eine Menge, } M \notin M\}$ führt zur Russellschen Antinomie.

Fazit:

Um solche Probleme zu vermeiden, sollte man bei intensionalen Mengendefinitionen immer angeben, aus welcher anderen Menge die ausgewählten Elemente kommen sollen, also:

$$\{x \in M : x \text{ hat Eigenschaft(en) } E\},$$

wobei M eine Menge und E eine Eigenschaft oder eine Liste von Eigenschaften ist, die jedes einzelne Element aus M haben kann oder nicht.

Wichtige grundsätzliche Eigenschaften von Mengen:

- Alle Elemente einer Menge sind verschieden. D.h. ein Wert ist entweder Element der Menge oder eben nicht — aber er kann nicht “mehrfach” in der Menge vorkommen.
- Die Elemente einer Menge haben keine feste Reihenfolge.
- Dieselbe Menge kann auf verschiedene Weisen beschrieben werden, z.B.

$$\{1, 2, 3\} = \{1, 2, 2, 3\} = \{2, 1, 3\} = \{i : i \text{ ist eine ganze Zahl, } 0 < i \leq 3\}.$$

- Mengen können aus atomaren oder aus zusammengesetzten Elementen gebildet werden. Menge kann auch “verschiedenartige” Elemente enthalten.

Beispiel: Die Menge

$$M := \{1, (\text{Pik}, 8), \{\text{rot}, \text{blau}\}, 5, 1\}$$

besteht aus 4 Elementen: dem atomaren Wert 1, dem Tupel (Pik, 8), der Menge {rot, blau} und dem atomaren Wert 5.

Notationen für bestimmte Zahlenmengen:

- \mathbb{N} := Menge der natürlichen Zahlen := $\{0, 1, 2, 3, \dots\}$
 $\mathbb{N}_{>0}$:= Menge der positiven natürlichen Zahlen := $\{1, 2, 3, \dots\}$
 \mathbb{Z} := Menge der ganzen Zahlen := $\{0, 1, -1, 2, -2, 3, -3, \dots\}$
 \mathbb{Q} := Menge der rationalen Zahlen := $\{\frac{a}{b} : a, b \in \mathbb{Z}, b \neq 0\}$
 \mathbb{R} := Menge der reellen Zahlen

Beobachtung: Es gibt genau eine Menge, die keine Elemente enthält.

Definition 2.2 (leere Menge).

Die **leere Menge** ist die (eindeutig bestimmte) Menge, die kein(e) Element(e) enthält. Wir bezeichnen sie mit \emptyset .

leere Menge

\emptyset

Frage 2.3: Gibt es eine "Menge aller Mengen"?

Antwort: Nein! Denn wäre U die Menge aller Mengen, so wäre auch $N := \{M \in U : M \notin M\}$ eine Menge. Dies führt aber wieder zur Russellschen Antinomie, da die Frage "Ist $N \in N$?" nicht geklärt werden kann.

2.1.2 Mengenalgebra

In diesem Abschnitt werden einige grundlegende Operationen auf Mengen betrachtet. Nebenbei werden auch einige (sehr einfache) Beispiele von mathematischen Beweisen gegeben.

Definition 2.4 (Gleichheit von Mengen).

Zwei Mengen M und N sind gleich (kurz: $M = N$), falls sie dieselben Elemente enthalten, d.h. falls gilt:

$M = N$

- f.a. $x \in M$ gilt $x \in N$, und
- f.a. $x \in N$ gilt $x \in M$.

Beachte:

$\emptyset \neq \{\emptyset\}$, denn \emptyset ist die Menge, die keine Elemente enthält, während $\{\emptyset\}$ eine Menge ist, die ein Element (nämlich \emptyset) enthält.

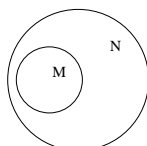
Definition 2.5 (Teilmengen). Seien M, N Mengen.

(a) M ist eine **Teilmenge** von N (kurz: $M \subseteq N$), wenn jedes Element von M auch ein Element von N ist.

Teilmenge

$M \subseteq N$

Skizze:



- (b) M ist eine **echte Teilmenge** von N (kurz: $M \subsetneq N$), wenn $M \subseteq N$ und $M \neq N$.
- (c) M ist eine **Obermenge** von N (kurz: $M \supseteq N$), wenn $N \subseteq M$.
- (d) M ist eine **echte Obermenge** von N (kurz: $M \supsetneq N$), wenn $M \supseteq N$ und $M \neq N$.

echte Teilmenge
 $M \subsetneq N$
 Obermenge
 $M \supseteq N$
 echte Obermenge
 $M \supsetneq N$

Satz 2.6. Seien M, N, P Mengen. Dann gilt:

- (a) $M = N \iff M \subseteq N$ und $M \supseteq N$.
- (b) $M \subseteq N$ und $N \subseteq P \implies M \subseteq P$.

Beweis:

(a)

$$\begin{aligned}
 M = N & \stackrel{\text{Def. 2.4}}{\iff} \begin{array}{l} \text{f.a. } x \in M \text{ gilt } x \in N \text{ und} \\ \text{f.a. } x \in N \text{ gilt } x \in M \end{array} \\
 & \iff \begin{array}{l} \text{jedes Element von } M \text{ ist auch ein Element von } N \text{ und} \\ \text{jedes Element von } N \text{ ist auch ein Element von } M \end{array} \\
 & \stackrel{\text{Def. 2.5(a)}}{\iff} M \subseteq N \text{ und } N \subseteq M \\
 & \stackrel{\text{Def. 2.5(c)}}{\iff} M \subseteq N \text{ und } M \supseteq N.
 \end{aligned}$$

(b) Es gelte $M \subseteq N$ und $N \subseteq P$.

Behauptung: $M \subseteq P$, d.h. f.a. $m \in M$ gilt $m \in P$.

Beweis: Sei $m \in M$ beliebig. Wir zeigen, dass $m \in P$:

$$m \in M \xrightarrow{\text{nach Vor.: } M \subseteq N} m \in N \xrightarrow{\text{nach Vor.: } N \subseteq P} m \in P.$$

□

Definition 2.7. Seien M und N Mengen.

(a) Der **Durchschnitt** von M und N ist die Menge

$$M \cap N := \{x : x \in M \text{ und } x \in N\}.$$

Durchschnitt
 $M \cap N$

(b) Die **Vereinigung** von M und N ist die Menge

$$M \cup N := \{x : x \in M \text{ oder } x \in N\}.$$

Vereinigung
 $M \cup N$

(c) Die **Differenz** von M und N ist die Menge

$$M \setminus N := M - N := \{x : x \in M \text{ und } x \notin N\}.$$

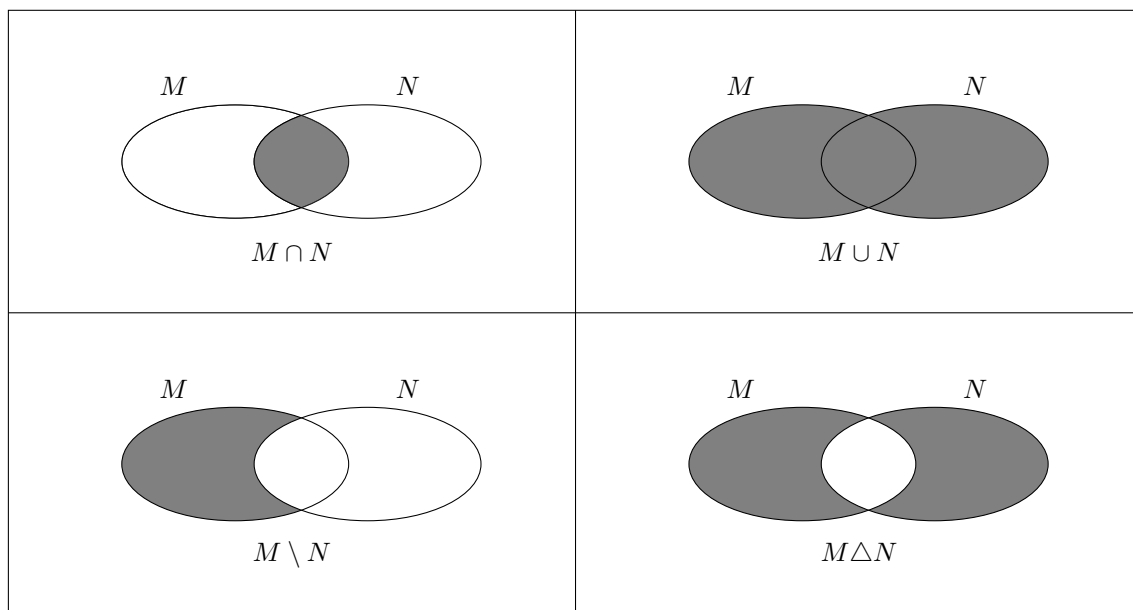
Differenz
 $M \setminus N$

(d) Die **symmetrische Differenz** von M und N ist die Menge

$$M \Delta N := (M \setminus N) \cup (N \setminus M).$$

symmetrische
 Differenz
 $M \Delta N$

Veranschaulichung durch Venn-Diagramme:



Notation 2.8 (disjunkt).

Zwei Mengen M und N heißen **disjunkt**, falls $M \cap N = \emptyset$, d.h. falls sie keine gemeinsamen Elemente besitzen. Manchmal schreiben wir

$M \dot{\cup} N$

$$M \dot{\cup} N,$$

um die Menge $M \cup N$ zu bezeichnen und gleichzeitig auszudrücken, dass $M \cap N = \emptyset$.

Rechenregeln für Durchschnitt und Vereinigung:

Satz 2.9. Seien M, N, P Mengen. Dann gelten:

Idempotenz

(a) **Idempotenz:**

$$M \cap M = M \quad \text{und} \quad M \cup M = M.$$

Kommutativität

(b) **Kommutativität:**

$$M \cap N = N \cap M \quad \text{und} \quad M \cup N = N \cup M.$$

Assoziativität

(c) **Assoziativität:**

$$M \cap (N \cap P) = (M \cap N) \cap P \quad \text{und} \quad M \cup (N \cup P) = (M \cup N) \cup P.$$

Absorption

(d) **Absorption:**

$$M \cap (M \cup N) = M \quad \text{und} \quad M \cup (M \cap N) = M.$$

(e) **Distributivität:**

Distributivität

$$M \cap (N \cup P) = (M \cap N) \cup (M \cap P) \quad \text{und} \quad M \cup (N \cap P) = (M \cup N) \cap (M \cup P).$$

Beweis:

(a)

$$\begin{aligned} M \cap M &\stackrel{\text{Def. 2.7(a)}}{=} \{x : x \in M \text{ und } x \in M\} \\ &= \{x : x \in M\} \\ &= M. \end{aligned}$$

Analog: $M \cup M = M$.

(b)

$$\begin{aligned} M \cap N &\stackrel{\text{Def. 2.7(a)}}{=} \{x : x \in M \text{ und } x \in N\} \\ &= \{x : x \in N \text{ und } x \in M\} \\ &\stackrel{\text{Def. 2.7(a)}}{=} N \cap M. \end{aligned}$$

Analog: $M \cup N = N \cup M$.

(c)

$$\begin{aligned} M \cap (N \cap P) &\stackrel{\text{Def. 2.7(a)}}{=} \{x : x \in M \text{ und } x \in N \cap P\} \\ &\stackrel{\text{Def. 2.7(a)}}{=} \{x : x \in M \text{ und } (x \in N \text{ und } x \in P)\} \\ &= \{x : (x \in M \text{ und } x \in N) \text{ und } x \in P\} \\ &\stackrel{\text{Def. 2.7(a)}}{=} \{x : x \in M \cap N \text{ und } x \in P\} \\ &\stackrel{\text{Def. 2.7(a)}}{=} (M \cap N) \cap P. \end{aligned}$$

Analog: $M \cup (N \cup P) = (M \cup N) \cup P$.

(d) Wir beweisen, dass $M \cap (M \cup N) = M$ in zwei Schritten:

Schritt 1: Zeige, dass $M \cap (M \cup N) \supseteq M$.

Schritt 2: Zeige, dass $M \cap (M \cup N) \subseteq M$.

Aus Satz 2.6(a) folgt dann, dass $M \cap (M \cup N) = M$.

ZU SCHRITT 1:

Behauptung: $M \cap (M \cup N) \supseteq M$, d.h. f.a. $m \in M$ gilt $m \in M \cap (M \cup N)$.

Beweis: Sei $m \in M$ beliebig. Zu zeigen: $m \in M \cap (M \cup N)$. Wegen $m \in M$ gilt auch $m \in M \cup N$ (gemäß Definition 2.7(b)). Wegen $m \in M$ und $m \in M \cup N$ gilt gemäß Definition 2.7(a), dass $m \in M \cap (M \cup N)$.

ZU SCHRITT 2:

Behauptung: $M \cap (M \cup N) \subseteq M$, d.h. f.a. $m \in M \cap (M \cup N)$ gilt $m \in M$.

Beweis: Sei $m \in M \cap (M \cup N)$ beliebig. Zu zeigen: $m \in M$. Wegen $m \in M \cap (M \cup N)$ gilt gemäß Definition 2.7(a), dass $m \in M$ und $m \in M \cup N$. Insbesondere ist also $m \in M$.

Insgesamt haben wir damit gezeigt, dass $M \cap (M \cup N) = M$.

Analog: $M \cup (M \cap N) = M$.

(e) Analog; Details: Übung.

□

2.1.3 Das Komplement einer Menge

Komplement
 \overline{M}

Das **Komplement** einer Menge M (kurz: \overline{M}) soll die Menge aller Elemente sein, die **nicht** zu M gehören. Bei der präzisen Definition von \overline{M} ist allerdings wieder Vorsicht geboten. Denn wenn wir einfach

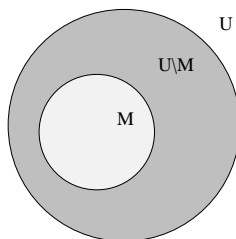
$$\overline{M} := \{x : x \notin M\}$$

setzen, so gilt für die leere Menge \emptyset , dass ihr Komplement $\overline{\emptyset}$ einfach **alles** enthält — und dann wäre

$$\{M : M \in \overline{\emptyset} \text{ und } M \text{ ist eine Menge}\}$$

die “Menge aller Mengen”, und dass es die nicht geben kann, haben wir bereits bei der Beantwortung von Frage 2.3 gesehen.

Daher betrachten wir Mengen stets innerhalb eines festen Universums U , das selbst eine Menge ist (die wir jeweils im Kontext angeben müssen). Für $M \subseteq U$ setzen wir dann $\overline{M} := U \setminus M$ und bezeichnen \overline{M} als das Komplement von M in U .



Rechenregeln für Komplemente:

Satz 2.10.

Sei U unser festes Universum, das selbst eine Menge ist, und seien $M, N \subseteq U$. Dann gelten:

Doppelte
Negation

(a) **Doppelte Negation:**

$$\overline{(\overline{M})} = M.$$

De Morgansche
Regeln

(b) **De Morgansche Regeln:**

$$\overline{M \cap N} = \overline{M} \cup \overline{N} \quad \text{und} \quad \overline{M \cup N} = \overline{M} \cap \overline{N}.$$

Inversionsregeln

(c) **Inversionsregeln:**

$$M \cap \overline{M} = \emptyset \quad \text{und} \quad M \cup \overline{M} = U.$$

Identitätsregeln

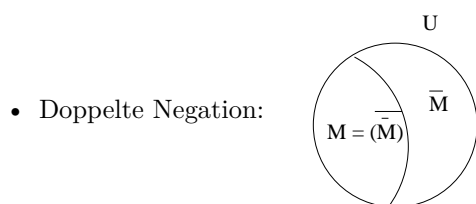
(d) **Identitätsregeln:**

$$M \cap U = M \quad \text{und} \quad M \cup \emptyset = M.$$

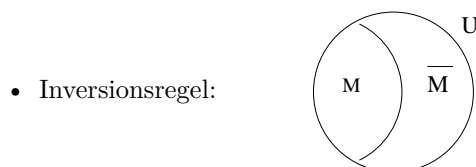
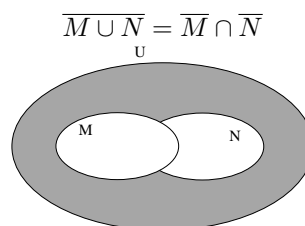
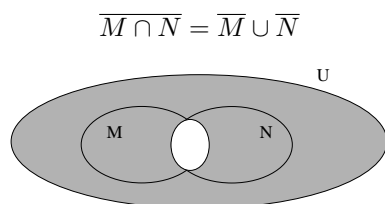
Beweis: Übung.

□

Veranschaulichung durch Venn-Diagramme:



- De Morgansche Regeln:



2.1.4 Mächtigkeit bzw. Kardinalität einer Menge

Definition 2.11.

(a) Eine Menge heißt **endlich**, wenn sie nur endlich viele Elemente enthält, d.h. wenn es eine Zahl $n \in \mathbb{N}$ gibt, so dass die Menge genau n Elemente enthält. endlich

(b) Die **Mächtigkeit** (oder **Kardinalität**) einer Menge M ist

$$|M| := \begin{cases} \text{Anzahl der Elemente in } M, & \text{falls } M \text{ endlich ist} \\ \infty \text{ (unendlich),} & \text{sonst.} \end{cases}$$

Mächtigkeit
Kardinalität
 $|M|$

Man beachte, dass “ ∞ ” keine natürliche Zahl ist (d.h. $\infty \notin \mathbb{N}$), sondern lediglich eine Abkürzung für das Wort “unendlich”.

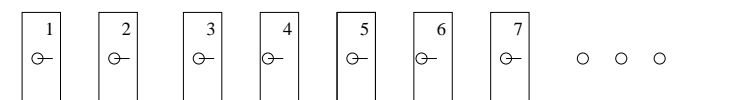
Beispiel 2.12.

- $|\{2, 4, 6\}| = 3$
- $|\emptyset| = 0$
- $|\{\emptyset\}| = 1$
- $|\mathbb{N}| = \infty$

- $|\mathbb{Z}| = \infty$
- $|\{2, 4, 6, 4\}| = 3$
- $|\{2, \{a, b\}\}| = 2$.

Vorsicht beim Vergleich der Mächtigkeit unendlicher Mengen:

Hilberts Hotel (David Hilbert, 1862–1943)
 Hilberts Hotel hat unendlich viele Zimmer, die fortlaufend mit $1, 2, 3, \dots$ (also mit allen Zahlen aus $\mathbb{N}_{>0}$) nummeriert sind. Obwohl alle Zimmer belegt sind, schafft der Angestellte an der Rezeption es, für jeden neuen Gast Platz zu schaffen.



Wie? — Er bittet alle Gäste, in das Zimmer mit der nächsthöheren Nummer umzuziehen und gibt dem neuen Gast das Zimmer mit der Nummer 1. Fügt man also zu einer unendlichen Menge ein Element hinzu, so erhält man keine “wirklich größere” Menge.

Es ist nicht schwer, zu sehen, dass im vollbesetzten Hotel sogar unendlich viele neue Gäste, die mit den Zahlen $1, 2, 3, \dots$ durchnummeriert sind, einquartiert werden können. Dazu muss einfach jeder der bisherigen Gäste in das Zimmer umziehen, dessen Nummer das Doppelte der bisherigen Zimmernummer ist. Danach sind alle “alten” Gäste in den Zimmern mit geraden Zimmernummern untergebracht, und die neuen Gäste können in die Zimmer mit ungeraden Zimmernummern einziehen.

2.1.5 Die Potenzmenge

Definition 2.13.

Potenzmenge
 $\mathcal{P}(M)$

Die **Potenzmenge** (engl.: power set) einer Menge M (kurz: $\mathcal{P}(M)$) ist die Menge aller Teilmengen von M . D.h.:

$$\mathcal{P}(M) = \{X : X \subseteq M\}.$$

Beispiel 2.14.

- $\mathcal{P}(\{a, b\}) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}.$
- $\mathcal{P}(\{1, 2, 3\}) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}.$
- $\mathcal{P}(\emptyset) = \{\emptyset\}.$

Insbesondere gilt: $\mathcal{P}(\emptyset) \neq \emptyset$.

Notation 2.15.

$\text{Pow}(M)$
 2^M
 $|\mathcal{P}(M)| = 2^{|M|}$

In manchen Büchern wird $\mathcal{P}(M)$ auch mit $\text{Pow}(M)$ (für “power set”) oder mit 2^M bezeichnet. Später, in Folgerung 2.39, werden wir nachweisen, dass für jede endliche Menge M gilt:

$$|\mathcal{P}(M)| = 2^{|M|}.$$

2.2 Kartesische Produkte und Relationen

2.2.1 Paare, Tupel und kartesische Produkte

Definition 2.16 (Paare und Tupel).

- (a) Für beliebige Objekte a und b bezeichnet (a, b) das geordnete **Paar** mit Komponenten a und b . Paar
- (b) Für $k \in \mathbb{N}$ und beliebige Objekte a_1, \dots, a_k bezeichnet (a_1, \dots, a_k) das **k -Tupel** mit Komponenten a_1, \dots, a_k . k -Tupel
- (c) Die **Gleichheit** zweier Tupel ist wie folgt definiert:
F.a. $k, \ell \in \mathbb{N}$ und $a_1, \dots, a_k, b_1, \dots, b_\ell$ gilt:

$$(a_1, \dots, a_k) = (b_1, \dots, b_\ell) \iff k = \ell \text{ und } a_1 = b_1 \text{ und } a_2 = b_2 \text{ und } \dots \text{ und } a_k = b_k.$$

Bemerkung 2.17.

- (a) Für $k = 0$ gibt es genau ein k -Tupel, nämlich das **leere Tupel** $()$, das keine Komponente(n) hat. leeres Tupel
 $()$
- (b) Man beachte den Unterschied zwischen Tupeln und Mengen: z.B.
- $(1, 2) \neq (2, 1)$, aber $\{1, 2\} = \{2, 1\}$.
 - $(1, 1, 2) \neq (1, 2)$, aber $\{1, 1, 2\} = \{1, 2\}$.

Definition 2.18.

- (a) Sei $k \in \mathbb{N}$ und sei M eine Menge. Die **k -te Potenz** von M ist die Menge k -te Potenz
 M^k

$$M^k := \{(m_1, \dots, m_k) : m_1 \in M, \dots, m_k \in M\}.$$

Insbesondere gilt: $M^0 = \{()\}$ besteht genau aus einem Element, dem leeren Tupel.

- (b) Das **kartesische Produkt** (bzw. **Kreuzprodukt**) zweier Mengen M, N ist die Menge kartesisches
Produkt
Kreuzprodukt
 $M \times N$

$$M \times N := \{(m, n) : m \in M, n \in N\}.$$

- (c) Sei $k \in \mathbb{N}_{>0}$ und seien M_1, \dots, M_k Mengen. Das kartesische Produkt von M_1, \dots, M_k ist die Menge

$$M_1 \times \dots \times M_k := \{(m_1, \dots, m_k) : m_1 \in M_1, \dots, m_k \in M_k\}.$$

Beispiel 2.19. Sei $M = \{a, b\}$ und $N = \{1, 2, 3\}$. Dann gilt:

- $M \times N = \{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$.
- $M \times \{1\} = \{(a, 1), (b, 1)\}$.
- $M \times \emptyset = \emptyset$.
- $M^2 = \{(a, a), (a, b), (b, a), (b, b)\}$.
- $M^1 = \{(a), (b)\}$.

- $M^0 = \{()\}$.
- $\emptyset^2 = \emptyset$.
- $\emptyset^1 = \emptyset$.
- $\emptyset^0 = \{()\}$.
- In Beispiel 2.1 hatten wir die Karten eines Skat-Kartenspiels durch folgende Wertebereiche modelliert:

$$\begin{aligned}\text{KartenArten} &= \{\text{Kreuz, Pik, Herz, Karo}\}, \\ \text{KartenSymbole} &= \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}, \\ \text{Karten} &= \text{KartenArten} \times \text{KartenSymbole}.\end{aligned}$$

- Uhrzeiten kann man repräsentieren durch Elemente der Menge

$$\text{Uhrzeiten} := \text{Stunden} \times \text{Minuten} \times \text{Sekunden},$$

wobei

$$\begin{aligned}\text{Stunden} &:= \{0, 1, 2, \dots, 23\}, \\ \text{Minuten} &:= \{0, 1, 2, \dots, 59\}, \\ \text{Sekunden} &:= \{0, 1, 2, \dots, 59\}.\end{aligned}$$

Das Tupel $(9, 45, 0)$ repräsentiert dann die Uhrzeit "9 Uhr, 45 Minuten und 0 Sekunden".

Notation 2.20.

- (a) Ist $k \in \mathbb{N}_{>0}$ und sind z_1, \dots, z_k Zahlen, so schreiben wir

$$\sum_{i=1}^k z_i \quad \text{bzw.} \quad \sum_{i \in \{1, \dots, k\}} z_i$$

um die Summe der Zahlen z_1, \dots, z_k zu bezeichnen (d.h. die Zahl $z_1 + \dots + z_k$).

Wir schreiben

$$\prod_{i=1}^k z_i \quad \text{bzw.} \quad \prod_{i \in \{1, \dots, k\}} z_i$$

um das Produkt der Zahlen z_1, \dots, z_k zu bezeichnen (d.h. die Zahl $z_1 \cdot \dots \cdot z_k$).

- (b) Sind M_1, \dots, M_k Mengen, so schreiben wir

$$\bigcup_{i=1}^k M_i \quad \text{bzw.} \quad \bigcup_{i \in \{1, \dots, k\}} M_i$$

um die Vereinigung der Mengen M_1, \dots, M_k zu bezeichnen (d.h. die Menge $M_1 \cup \dots \cup M_k$).

Wir schreiben

$$\bigcap_{i=1}^k M_i \quad \text{bzw.} \quad \bigcap_{i \in \{1, \dots, k\}} M_i$$

um den Durchschnitt der Mengen M_1, \dots, M_k zu bezeichnen (d.h. die Menge $M_1 \cap \dots \cap M_k$).

(c) Ist K eine Menge, deren Elemente Teilmengen einer Menge U sind (d.h.: $K \subseteq \mathcal{P}(U)$), so ist

$$\bigcup_{M \in K} M := \{x \in U : \text{ex. } M \in K \text{ s.d. } x \in M\}$$

die Vereinigung aller Mengen $M \in K$ (d.h. die Menge aller Elemente x , die in mindestens einer Menge $M \in K$ liegen).

Analog ist

$$\bigcap_{M \in K} M := \{x \in U : \text{f.a. } M \in K \text{ gilt } x \in M\}$$

der Durchschnitt aller Mengen $M \in K$ (d.h. die Menge aller Elemente x , die in jeder Menge $M \in K$ liegen).

Die Mächtigkeit von kartesischen Produkten:

Satz 2.21.

(a) Seien M und N zwei endliche Mengen. Dann gilt:

$$|M \times N| = |M| \cdot |N|.$$

(b) Sei $k \in \mathbb{N}_{>0}$ und seien M_1, \dots, M_k endliche Mengen. Dann gilt:

$$|M_1 \times \dots \times M_k| = \prod_{i=1}^k |M_i|.$$

(c) Sei $k \in \mathbb{N}$ und sei M eine endliche Menge. Dann gilt:

$$|M^k| = |M|^k.$$

Beweis:

(a) Es gilt:

$$M \times N = \{(m, n) : m \in M, n \in N\} = \bigcup_{m \in M} \{(m, n) : n \in N\} = \bigcup_{m \in M} (\{m\} \times N).$$

Außerdem gilt für alle $m, m' \in M$ mit $m \neq m'$, dass die Mengen $\{m\} \times N$ und $\{m'\} \times N$ disjunkt sind. Ferner gilt für beliebige disjunkte endliche Mengen A und B , dass $|A \cup B| = |A| + |B|$ ist. Insgesamt folgt daraus, dass

$$\begin{aligned} |M \times N| &= \left| \bigcup_{m \in M} (\{m\} \times N) \right| = \sum_{m \in M} |\{m\} \times N| \\ &= \sum_{m \in M} |N| = \underbrace{|N| + \dots + |N|}_{|M|\text{-mal}} = |M| \cdot |N|. \end{aligned}$$

(b) Analog; Details: Übung.

(c)

$$|M^k| = \underbrace{|M \times \dots \times M|}_{k\text{-mal}} \stackrel{(b)}{=} \prod_{i=1}^k |M| = \underbrace{|M| \cdot \dots \cdot |M|}_{k\text{-mal}} = |M|^k.$$

□

2.2.2 Worte bzw. endliche Folgen

Bemerkung 2.22. Sei A eine Menge.

- Wort
- Gelegentlich fassen wir ein Tupel $(a_1, \dots, a_k) \in A^k$ als **Wort** auf, dessen “Buchstaben” a_1, \dots, a_k sind. Um diese Sichtweise zu betonen, schreiben wir oft $a_1 \cdots a_k$.
Beispiel: Das Tupel (M, o, d, e, l, l) identifizieren wir mit dem Wort Modell.
- Alphabet
- A ist dann das **Alphabet**, über dem die Worte gebildet werden, und $a_1 \cdots a_k$ wird “Wort über A ” genannt.
- leeres Wort (ε)
- Das **leere Tupel** $() \in A^0$ heißt auch **leeres Wort** und wird oft als ε (epsilon) bezeichnet.
- Länge
- Die **Länge** eines Wortes $a_1 \cdots a_k$ ist die Zahl

$$|a_1 \cdots a_k| := k.$$

Insbesondere ist $|\varepsilon| = 0$, d.h. das leere Wort hat die Länge 0.

- Konkatenation
- Sind $v = a_1 \cdots a_k$ und $w = b_1 \cdots b_\ell$ zwei Worte über A , so ist die **Konkatenation** von v und w das Wort

$$vw := a_1 \cdots a_k b_1 \cdots b_\ell.$$

- Manchmal wird ein Wort $a_1 \cdots a_k$ auch als **Folge** der Länge k aufgefasst.

Definition 2.23 (A^* , A^+ , **Sprache**). Sei A ein Alphabet (d.h. eine Menge).

- Menge aller Worte über A (A^*)
- (a) Die **Menge aller Worte über A** (von beliebiger endlicher Länge) bezeichnen wir mit A^* . Es gilt also:

$$A^* = \bigcup_{k \in \mathbb{N}} A^k = \{ a_1 \cdots a_k : k \in \mathbb{N}, a_1, \dots, a_k \in A \}.$$

Beachte: Wegen $0 \in \mathbb{N}$ und $A^0 = \{()\} = \{\varepsilon\}$ enthält A^* insbesondere das leere Wort.

- A^+
- (b) Die Menge aller **nicht-leeren** Worte über A (von beliebiger endlicher Länge) bezeichnen wir mit A^+ . Es gilt:

$$A^+ = A^* \setminus \{\varepsilon\} = \{ a_1 \cdots a_k : k \in \mathbb{N}_{>0}, a_1, \dots, a_k \in A \}.$$

- Sprache
- (c) Eine **Sprache** über A ist eine Teilmenge von A^* .

Bemerkung: In vielen Büchern werden Sprachen mit dem Buchstaben L (für **L**anguage) oder mit Varianten wie L' oder L_1 bezeichnet.

Beispiel 2.24 (**Natürliche Sprachen**).

Wir betrachten das Alphabet

$$A_{\text{deutsch}} := \{A, B, \dots, Z, \ddot{A}, \ddot{O}, \ddot{U}, a, b, \dots, z, \ddot{a}, \ddot{o}, \ddot{u}, \beta, \cdot, \cdot\cdot, \cdot:, \cdot;, !, ?, -, _ \}.$$

Beispiele für Sprachen über A_{deutsch} sind:

- $L_1 :=$ Menge aller grammatikalisch korrekten Sätze der deutschen Sprache (aufgefasst, als Zeichenketten über A_{deutsch})

- $L_2 :=$ Menge aller Wörter der deutschen Sprache.

Beispiel 2.25 (Programmiersprachen).

Wir betrachten das Alphabet

ASCII := die Menge aller ASCII-Symbole

Beispiele für Sprachen über Alphabet ASCII sind:

- $L_1 :=$ die Menge aller JAVA-Schlüsselwörter,
- $L_2 :=$ die Menge aller erlaubten Variablennamen in JAVA,
- $L_3 :=$ die Menge aller syntaktisch korrekten JAVA-Programme.

2.2.3 Relationen

Relationen sind Teilmengen von kartesischen Produkten. Präzise:

Definition 2.26.

- (a) Seien M, N Mengen. Eine **Relation** von M nach N ist eine Teilmenge von $M \times N$. Relation
- (b) Sei $k \in \mathbb{N}_{>0}$ und seien M_1, \dots, M_k Mengen. Eine **Relation auf** M_1, \dots, M_k ist eine Teilmenge von $M_1 \times \dots \times M_k$. Die **Stelligkeit** einer solchen Relation ist k . Stelligkeit
- (c) Sei M eine Menge und sei $k \in \mathbb{N}$. Eine **k -stellige Relation über** M ist eine Teilmenge von M^k .

Beispiel 2.27. Um Datumsangaben im Format (Tag, Monat, Jahr) anzugeben, nutzen wir die Wertebereiche

$$\begin{aligned} \text{TagWerte} &:= \{1, 2, \dots, 31\} \\ \text{MonatsWerte} &:= \{1, 2, \dots, 12\} \\ \text{JahresWerte} &:= \mathbb{Z}. \end{aligned}$$

Die Menge "Gültig" aller **gültigen** Daten ist dann eine **Teilmenge** von

$$\text{TagWerte} \times \text{MonatsWerte} \times \text{JahresWerte},$$

d.h. eine **Relation** auf TagWerte, MonatsWerte, JahresWerte, zu der beispielsweise das Tupel (23, 6, 1912) gehört,¹ nicht aber das Tupel (30, 2, 1912).

Notation 2.28.

¹Der 23. Juni 1912 ist der Geburtstag von **Alan M. Turing**, einem der einflussreichsten Pioniere der Informatik.

- Ist R eine Relation von M nach N (für zwei Mengen M, N), so schreiben wir oft

$$mRn \quad \text{statt} \quad (m, n) \in R.$$

Beispiel:

- $m \leq n$, für natürliche Zahlen m, n
- $m \neq n$

- Ist R eine Relation auf M_1, \dots, M_k , so schreiben wir manchmal

$$R(m_1, \dots, m_k) \quad \text{statt} \quad (m_1, \dots, m_k) \in R.$$

Das soll verdeutlichen, dass R eine “Eigenschaft” ist, die ein Tupel aus $M_1 \times \dots \times M_k$ haben kann — oder eben nicht haben kann.

Im Datums-Beispiel gilt: Gültig(23, 6, 1912), aber es gilt **nicht**: Gültig(30, 2, 1912).

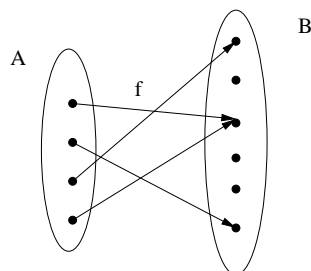
2.3 Funktionen

2.3.1 Totale Funktionen und partielle Funktionen

Funktion
Abbildung

Definition 2.29. Seien A, B Mengen. Eine **Funktion** (oder **Abbildung**) von A nach B ist eine Relation f von A nach B (d.h. $f \subseteq A \times B$) mit der Eigenschaft, dass für jedes $a \in A$ **genau ein** $b \in B$ mit $(a, b) \in f$ existiert.

Anschaulich:



Notation 2.30.

- Wir schreiben $f: A \rightarrow B$, um auszudrücken, dass f eine Funktion von A nach B ist.
- Ist $f: A \rightarrow B$ und ist $a \in A$, so bezeichnet $f(a)$ das (eindeutig bestimmte) $b \in B$ mit $(a, b) \in f$. Insbesondere schreiben wir meistens $f(a) = b$ an Stelle von $(a, b) \in f$.
- Für $f: A \rightarrow B$ und $A' \subseteq A$ sei

$$f(A') := \{f(a) : a \in A'\}.$$

$\text{Abb}(A, B)$

- Die Menge aller Funktionen von A nach B bezeichnen wir mit $\text{Abb}(A, B)$.

B^A

- In manchen Büchern wird $\text{Abb}(A, B)$ auch mit $A \rightarrow B$ oder mit B^A bezeichnet. Später, in Folgerung 2.39, werden wir sehen, dass

$$|\text{Abb}(A, B)| = |B|^{|A|}.$$

Definition 2.31.

Zwei Funktionen $f: A \rightarrow B$ und $g: A \rightarrow B$ sind **gleich** (kurz: $f = g$), falls f.a. $a \in A$ gilt: $f(a) = g(a)$.

Definition 2.32 (Definitionsbereich, Bildbereich, Bild). Sei $f: A \rightarrow B$.

- (a) Der **Definitionsbereich** von f ist die Menge $\text{Def}(f) := A$. Definitionsbereich
- (b) Der **Bildbereich** von f ist die Menge B . Def(f)
- (c) Das **Bild** von f (genauer: das Bild von A unter f) ist die Menge Bildbereich
Bild(f)

$$\text{Bild}(f) := f(A) = \{f(a) : a \in A\} \subseteq B.$$

Definition 2.33 (Restriktionen).

Sei $f: A \rightarrow B$ eine Funktion und sei $A' \subseteq A$. Die **Restriktion** (oder **Einschränkung**) von f auf A' ist die Funktion Restriktion
Einschränkung

$$f|_{A'}: A' \rightarrow B,$$

die folgendermaßen definiert ist: f.a. $a \in A'$ ist $f|_{A'}(a) := f(a)$.

Definition 2.34.

Eine **partielle Funktion** von einer Menge A in eine Menge B ist eine Funktion f mit $\text{Def}(f) \subseteq A$ und $\text{Bild}(f) \subseteq B$. partielle Funktion

Bemerkung 2.35.

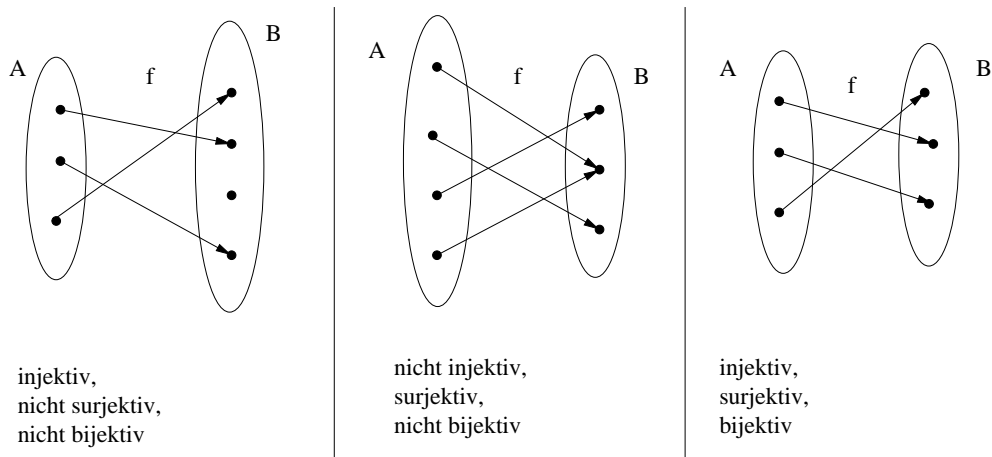
- (a) Im Gegensatz zu partiellen Funktionen nennt man Funktionen, wie wir sie in Definition 2.29 definiert haben, auch **totale Funktionen**. totale Funktion
Sprechen wir von “Funktionen”, ohne sie explizit als “partiell” zu bezeichnen, so meinen wir in dieser Vorlesung immer “totale” Funktionen.
- (b) Jede partielle Funktion von einer Menge A in eine Menge B lässt sich auch als totale Funktion von A nach $B \cup \{\perp\}$ auffassen, wobei \perp ein spezielles Zeichen ist, das für “undefiniert” steht, und das nicht zur Menge B gehört.

2.3.2 Eigenschaften von Funktionen

Definition 2.36. Sei $f: A \rightarrow B$.

- (a) f heißt **injektiv**, falls es für jedes $b \in B$ höchstens ein $a \in A$ mit $f(a) = b$ gibt. injektiv
- (b) f heißt **surjektiv**, falls es für jedes $b \in B$ mindestens ein $a \in A$ mit $f(a) = b$ gibt. surjektiv
- (c) f heißt **bijektiv**, falls es für jedes $b \in B$ genau ein $a \in A$ mit $f(a) = b$ gibt. bijektiv

Anschaulich:



Beobachtung 2.37.

(a) Für jede Funktion $f: A \rightarrow B$ gilt:

$$f \text{ ist bijektiv} \iff f \text{ ist injektiv und surjektiv.}$$

(b) Seien A und B **endliche** Mengen. Dann gilt:

$$|A| = |B| \iff \text{es gibt eine bijektive Funktion von } A \text{ nach } B.$$

Satz 2.38.

- (a) Für jede Menge M gibt es eine bijektive Funktion von $\mathcal{P}(M)$ nach $\text{Abb}(M, \{0, 1\})$.
- (b) Sei B eine Menge, sei A eine endliche Menge und sei $k := |A|$. Dann gibt es eine bijektive Funktion von $\text{Abb}(A, B)$ nach B^k .

Beweis:

- (a) Repräsentiere jedes $X \in \mathcal{P}(M)$ (d.h. $X \subseteq M$) durch die so genannte **charakteristische Funktion** $\chi_X: M \rightarrow \{0, 1\}$ mit

charakteristische Funktion

$$\chi_X(m) := \begin{cases} 1, & \text{falls } m \in X \\ 0, & \text{sonst.} \end{cases} \quad (*)$$

Sei nun $f: \mathcal{P}(M) \rightarrow \text{Abb}(M, \{0, 1\})$ definiert durch

$$f(X) := \chi_X, \quad \text{für jedes } X \in \mathcal{P}(M). \quad (**)$$

Behauptung: f ist bijektiv.

Wir zeigen dies in 2 Schritten (und nutzen Beobachtung 2.37(a)).

Schritt 1: f ist injektiv:

Seien $X, X' \in \mathcal{P}(M)$ mit $f(X) = f(X')$.

Ziel: Zeige, dass $X = X'$.

Wegen $f(X) = f(X')$ gilt gemäß (**), dass $\chi_X = \chi_{X'}$. D.h. f.a. $m \in M$ gilt $\chi_X(m) = \chi_{X'}(m)$. Gemäß (*) gilt daher f.a. $m \in M$, dass

$$m \in X \iff m \in X'.$$

Somit ist $X = X'$.

Schritt 2: f ist surjektiv:

Sei $h \in \text{Abb}(M, \{0, 1\})$, d.h. $h: M \rightarrow \{0, 1\}$.

Ziel: Finde ein $X \in \mathcal{P}(M)$ mit $f(X) = h$.

Wir wählen

$$X := \{m \in M : h(m) = 1\}.$$

Dann ist klar: $X \in \mathcal{P}(M)$. Gemäß (*) gilt $\chi_X = h$. Gemäß (**) ist daher $f(X) = h$.

- (b) *Idee:* Sei a_1, \dots, a_k eine Liste aller Elemente in A . Repräsentiere jede Funktion $h \in \text{Abb}(A, B)$ durch das k -Tupel $t_h := (h(a_1), \dots, h(a_k))$.

Rest: Übung.

□

Folgerung 2.39. Seien A, B, M endliche Mengen. Dann gilt:

(a) $|\text{Abb}(A, B)| = |B|^{|A|}$.

(b) $|\mathcal{P}(M)| = 2^{|M|}$.

Beweis:

- (a) Gemäß Satz 2.38(b) und Beobachtung 2.37(b) gilt für $k := |A|$, dass

$$|\text{Abb}(A, B)| = |B|^k.$$

Laut Satz 2.21(c) ist $|B|^k = |B|^k$. Somit $|\text{Abb}(A, B)| = |B|^k = |B|^{|A|}$.

- (b) Gemäß Satz 2.38(a) und Beobachtung 2.37(b) ist

$$|\mathcal{P}(M)| = |\text{Abb}(M, \{0, 1\})|.$$

Gemäß (a) ist

$$|\text{Abb}(M, \{0, 1\})| = |\{0, 1\}|^{|M|} = 2^{|M|}.$$

□

2.3.3 Spezielle Funktionen

Identitätsfunktion
id_M

Definition 2.40. Die **Identitätsfunktion** auf einer Menge M ist die Funktion

$$\text{id}_M: M \rightarrow M$$

mit $\text{id}_M(m) := m$, f.a. $m \in M$.

Multimenge

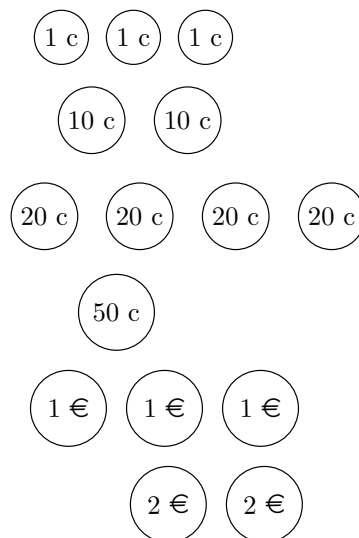
Definition 2.41 (Multimenge, engl.: bag).

Eine **Multimenge** über einer Menge M ist eine Funktion $f: M \rightarrow \mathbb{N}$.

Mit solchen “Multimengen” kann man “Mengen” beschreiben, in denen einzelne Elemente mehrfach vorkommen können: Für jedes $m \in M$ gibt $f(m)$ an, wie oft m in der “Multimenge” vorkommt.

Beispiel 2.42. Ein Geldbeutel mit

- 3 1-Cent-Münzen
- 2 10-Cent-Münzen
- 4 20-Cent-Münzen
- 1 50-Cent-Münzen
- 3 1-Euro-Münzen
- 2 2-Euro-Münzen



kann repräsentiert werden durch die Multimenge

Geldbeutelinhalt: MünzenArten $\rightarrow \mathbb{N}$,

wobei

MünzenArten := {1c, 2c, 5c, 10c, 20c, 50c, 1€, 2€}

und

Geldbeutelinhalt(1c) := 3
 Geldbeutelinhalt(2c) := 0
 Geldbeutelinhalt(5c) := 0
 Geldbeutelinhalt(10c) := 2
 Geldbeutelinhalt(20c) := 4
 Geldbeutelinhalt(50c) := 1
 Geldbeutelinhalt(1€) := 3
 Geldbeutelinhalt(2€) := 2.

Bequemere Schreibweise (die konsistent ist mit Definition 2.29):

Geldbeutelinhalt := {(1c, 3), (2c, 0), (5c, 0), (10c, 2), (20c, 4), (50c, 1), (1€, 3), (2€, 2)}.

2.4 Ein Beispiel zur Modellierung mit Wertebereichen

Beispiel 2.43 (Arbeitskreise der EU).

In der EU-Kommission sollen drei Arbeitskreise gebildet werden. Dazu entsendet jede der Nationen Deutschland, Frankreich, Österreich und Spanien drei Delegierte. Die Arbeitskreise sollen so gebildet werden, dass in jedem Arbeitskreis jede Nation vertreten ist und dass es unter Berücksichtigung der Fremdsprachenkenntnisse der Delegierten in jedem Arbeitskreis eine gemeinsame Sprache gibt, die alle beherrschen.

Aufgabe: Es soll nur die Situation modelliert werden — ein Lösungsverfahren wird hier zunächst nicht gesucht.

Formale Modellierung:

- Menge der **Nationen**:

$$\text{Nationen} := \{D, F, \text{Ö}, S\},$$

wobei D für Deutschland, F für Frankreich, Ö für Österreich und S für Spanien steht.

- Die **Delegierten** können wir repräsentieren als Paare, die aus einer Nation und einem Element aus $\{1, 2, 3\}$ bestehen, so dass beispielsweise die drei Delegierten aus Deutschland durch die Paare $(D, 1)$, $(D, 2)$ und $(D, 3)$ modelliert werden. Also:

$$\text{Delegierte} := \text{Nationen} \times \text{DelegiertenNummer},$$

wobei $\text{DelegiertenNummer} := \{1, 2, 3\}$.

- Wir nutzen eine Funktion “spricht”, die jedem Delegierten die Menge von **Sprachen** zuordnet, die er beherrscht. Formal:

$$\text{spricht} : \text{Delegierte} \rightarrow \mathcal{P}(\text{Sprachen}),$$

wobei

$$\text{Sprachen} := \{\text{deutsch, französisch, spanisch, englisch, italienisch, chinesisch, \dots}\}.$$

- Die drei Arbeitskreise bezeichnen wir mit AK1, AK2, AK3 und setzen

$$\text{Arbeitskreise} := \{\text{AK1, AK2, AK3}\}.$$

- Eine konkrete Besetzung der drei Arbeitskreise repräsentieren wir durch eine Funktion

$$\text{AK-Besetzung} : \text{Arbeitskreise} \rightarrow \mathcal{P}(\text{Delegierte}),$$

die jedem der 3 Arbeitskreise die Menge der Delegierten zuordnet, die Mitglieder des Arbeitskreises sind.

- Die Bedingung, dass jede Nation in jedem Arbeitskreis vertreten ist, lässt sich folgendermaßen formulieren:

$$\text{f.a. } a \in \text{Arbeitskreise} \text{ ist } \text{Vertretene_Nationen_in_}a = \text{Nationen},$$

wobei

$$\text{Vertretene_Nationen_in_}a := \left\{ n \in \text{Nationen} : \text{es ex. ein } i \in \text{DelegiertenNummer} \right. \\ \left. \text{s.d. } (n, i) \in \text{AK-Besetzung}(a) \right\}.$$

- Die Bedingung, dass es für jeden Arbeitskreis eine Sprache gibt, die alle Mitglieder des Arbeitskreises beherrschen, lässt sich folgendermaßen formulieren:

f.a. $a \in \text{Arbeitskreise}$ ist $\text{Gemeinsame_Sprachen_in_}a \neq \emptyset$,

wobei

$$\text{Gemeinsame_Sprachen_in_}a := \left\{ \text{sp} \in \text{Sprachen} : \text{f.a. } d \in \text{AK-Besetzung}(a) \right. \\ \left. \text{ist sp} \in \text{spricht}(d) \right\}.$$

□ Ende Beispiel 2.43

2.5 Beweise verstehen und selbst formulieren

Ziel dieses Abschnitts ist, einen kurzen Überblick über grundlegende Beweistechniken zu geben, insbesondere:

- direkter Beweis
- Beweis durch Kontraposition
- Beweis durch Widerspruch (indirekter Beweis)
- vollständige Induktion.

2.5.1 Was sind “Sätze” und “Beweise”?

Satz
Theorem
Beweis

Ein **Satz** (bzw. **Theorem**) besteht aus Voraussetzungen und einer Behauptung. Voraussetzungen und Behauptung sind Aussagen, so dass folgendes gilt: Wenn alle Voraussetzungen erfüllt sind, dann muss auch die Behauptung wahr sein. Der **Beweis** eines Satzes muss nachweisen, dass die Behauptung des Satzes wahr ist und kann dabei verwenden:

- die Voraussetzungen des Satzes,
- Definitionen und bereits bekannte Tatsachen und Sätze,
- im Beweis selbst oder anderswo bereits als wahr bewiesene Aussagen,
- logische Schlussregeln.

Typische Fehler, die man beim Versuch, Beweise zu formulieren, **vermeiden** sollte, sind:

- unzulässiges Argumentieren mit Beispielen,
- Verwendung gleicher Symbole zur Bezeichnung verschiedener Dinge,
- Hantieren mit nicht exakt oder gar widersprüchlich definierten Begriffsbildungen,
- unzulässige Gedankensprünge beim Schlussfolgern,
- Ausnutzung von bis dahin noch unbewiesenen Behauptungen zur Begründung von einzelnen Beweisschritten.

2.5.2 Beweistechnik “direkter Beweis”

Bei einem *direkten Beweis* wird die Behauptung eines Satzes “direkt”, d.h. ohne “Umwege”, bewiesen.

Beispiele für direkte Beweise haben wir bereits kennengelernt, z.B. der Beweis von Satz 2.6, der Beweis von Satz 2.21, der Beweis von Satz 2.38, der Beweis von Folgerung 2.39.

2.5.3 Beweistechnik “Beweis durch Kontraposition”

Man beachte, dass für beliebige Aussagen A und B das Folgende gilt:

$$\begin{aligned} & \text{die folgende Aussage ist wahr:} \\ & \text{“Falls Aussage } A \text{ gilt, so gilt auch Aussage } B\text{”} \\ \iff & \text{ Aussage } B \text{ gilt oder Aussage } A \text{ gilt nicht} \\ \iff & \text{ die folgende Aussage ist wahr:} \\ & \text{“Falls Aussage } B \text{ **nicht** gilt, so gilt auch Aussage } A \text{ **nicht**.”} \end{aligned}$$

Beim *Beweis durch Kontraposition* wird ein Satz der Form

$$\text{“Falls Aussage } A \text{ gilt, so gilt auch Aussage } B\text{”}$$

dadurch bewiesen, dass man zeigt:

$$\text{“Falls Aussage } B \text{ **nicht** gilt, so kann auch Aussage } A \text{ **nicht** gelten.”}$$

Als Beispiel für einen Beweis durch Kontraposition betrachten wir folgenden Satz.

Satz 2.44.

Für jedes $n \in \mathbb{N}$ gilt: Falls n^2 eine ungerade Zahl ist, so ist auch n eine ungerade Zahl.

Beweis: Durch Kontraposition. Sei $n \in \mathbb{N}$ beliebig.

Wir zeigen: Falls n **keine** ungerade Zahl ist, so ist auch n^2 **keine** ungerade Zahl.

$n \in \mathbb{N}$ war beliebig gewählt. Falls n ungerade ist, so ist nichts weiter zu beweisen. Wir betrachten daher nur den Fall, dass n **keine** ungerade Zahl ist (d.h. n ist gerade). Wir müssen zeigen, dass dann auch n^2 **keine** ungerade Zahl ist (d.h. n^2 ist eine gerade Zahl).

Beachte: Per Definition ist eine natürliche Zahl m genau dann **gerade**, wenn es ein $k \in \mathbb{N}$ gibt, s.d. $m = 2 \cdot k$. Daher gilt:

$$\begin{aligned} n \text{ ist gerade} & \implies \text{es ex. } k \in \mathbb{N} \text{ s.d. } n = 2 \cdot k \quad (\text{gemäß Def. von “gerade”}) \\ & \implies \text{es ex. } k \in \mathbb{N} \text{ s.d. } n^2 = n \cdot (2 \cdot k) \\ & \implies \text{es ex. } k \in \mathbb{N} \text{ s.d. } n^2 = 2 \cdot (n \cdot k) \\ & \implies \text{es ex. } k' \in \mathbb{N} \text{ s.d. } n^2 = 2 \cdot k' \\ & \implies n^2 \text{ ist gerade} \quad (\text{gemäß der Definition von “geraden Zahlen”}). \end{aligned}$$

Somit ist n^2 gerade, d.h. n^2 ist keine ungerade Zahl. □

2.5.4 Beweistechnik “Beweis durch Widerspruch” (indirekter Beweis)

Beim *Beweis durch Widerspruch* wird ein Satz der Form

“Falls die Voraussetzungen A erfüllt sind, so gilt Aussage B ”

dadurch bewiesen, dass man

- annimmt, dass die Voraussetzungen A erfüllt sind, aber die Aussage B **nicht** gilt und
- daraus einen Widerspruch herleitet.

Als Beispiel für einen Beweis durch Widerspruch betrachten wir folgenden Satz:

Satz 2.45. Für alle geraden natürlichen Zahlen a und b gilt: $a \cdot b$ ist gerade.

Beweis: Durch Widerspruch.

Angenommen, a und b sind gerade natürlichen Zahlen, so dass $a \cdot b$ **nicht** gerade ist.

Da a und b gerade sind, gibt es $k, \ell \in \mathbb{N}$ s.d. $a = 2 \cdot k$ und $b = 2 \cdot \ell$.

Dann ist $a \cdot b = (2 \cdot k) \cdot (2 \cdot \ell)$. Insbesondere gibt es also ein $m \in \mathbb{N}$, s.d. $a \cdot b = 2 \cdot m$.

Gemäß der Definition von “geraden Zahlen” ist also $a \cdot b$ gerade. Dies ist ein Widerspruch zur Annahme, dass $a \cdot b$ **nicht** gerade ist. \square

Ein weiteres, etwas anspruchsvolleres Beispiel für einen Beweis durch Widerspruch ist der Beweis des folgenden Satzes, der “anschaulich” besagt, dass die Potenzmenge von \mathbb{N} **viel** größer ist als die Menge \mathbb{N} selbst.

Satz 2.46 (“ $\mathcal{P}(\mathbb{N})$ ist nicht abzählbar”).

Es gibt keine surjektive Funktion von \mathbb{N} nach $\mathcal{P}(\mathbb{N})$.

Beweis: Durch Widerspruch. Angenommen, $f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$ ist surjektiv. Sei

$$M := \{n \in \mathbb{N} : n \notin f(n)\}. \quad (*)$$

Klar: $M \in \mathcal{P}(\mathbb{N})$.

Da f surjektiv ist, muss es ein $m \in \mathbb{N}$ geben mit $f(m) = M$.

Klar: Entweder gilt $m \in M$ oder es gilt $m \notin M$.

Fall 1: $m \notin M$:

Wegen $f(m) = M$ gilt also $m \notin f(m)$.

Gemäß (*) für $n := m$ folgt, dass $m \in M$. ζ (Widerspruch zu “Fall 1: $m \notin M$ ”).

Fall 2: $m \in M$:

Wegen $f(m) = M$ gilt also: $m \in f(m)$.

Gemäß (*) für $n := m$ folgt, dass $m \notin M$. ζ (Widerspruch zu “Fall 2: $m \in M$ ”).

Somit führen beide Fälle zu einem Widerspruch. Daher muss unsere Annahme, dass es eine surjektive Funktion f von \mathbb{N} nach $\mathcal{P}(\mathbb{N})$ gibt, falsch gewesen sein. \square

Bemerkung:

Diagonalisierung

Die in diesem Beweis verwendete Technik ist unter dem Namen **Diagonalisierung** (oder **Can-**

tors zweites Diagonalargument) bekannt. Die Grundidee des obigen Beweises lässt sich nämlich folgendermaßen veranschaulichen: Eine Funktion $f : \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$ können wir durch folgende Tabelle repräsentieren

	0	1	2	3	4	5	...
0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$...
1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$...
2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$...
3	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$...
4	$a_{4,0}$	$a_{4,1}$	$a_{4,2}$	$a_{4,3}$	$a_{4,4}$	$a_{4,5}$...
5	$a_{5,0}$	$a_{5,1}$	$a_{5,2}$	$a_{5,3}$	$a_{5,4}$	$a_{5,5}$...
...	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\ddots

wobei der Eintrag $a_{i,j}$ in Zeile i und Spalte j folgendermaßen gewählt ist:

$$a_{i,j} := \begin{cases} 1 & \text{falls } j \in f(i) \\ 0 & \text{falls } j \notin f(i). \end{cases}$$

Somit repräsentiert jede Zeile i dieser Tabelle die Menge $f(i)$, und es gilt

$$f(i) = \{j \in \mathbb{N} : a_{i,j} = 1\}.$$

Wir wählen nun die Folge

$$b_0 \ b_1 \ b_2 \ b_3 \ b_4 \ b_5 \ \dots$$

so, dass sich, für jedes $j \in \mathbb{N}$, der Wert b_j von dem Eintrag $a_{j,j}$ in der Diagonalen der Tabelle unterscheidet. D.h., wir wählen

$$b_j := \begin{cases} 0 & \text{falls } a_{j,j} = 1 \\ 1 & \text{falls } a_{j,j} = 0, \end{cases}$$

für alle $j \in \mathbb{N}$. Anhand dieser Wahl von b_0, b_1, b_2, \dots wissen wir, dass diese Folge in keiner Zeile der Tabelle stehen kann, denn für jede Zeile i unterscheidet sich der in Spalte i stehende Wert $a_{i,i}$ vom Wert b_i . Somit gilt für die Menge

$$M := \{j \in \mathbb{N} : b_j = 1\},$$

dass M nicht im Bild der Funktion f liegen kann, und dass f daher nicht surjektiv sein kann.

Man kann sich leicht davon überzeugen, dass die hier gewählte Menge M mit der im Beweis von Satz 2.46 gewählten Menge M übereinstimmt, denn

$$M = \{j \in \mathbb{N} : b_j = 1\} = \{j \in \mathbb{N} : a_{j,j} = 0\} = \{j \in \mathbb{N} : j \notin f(j)\} = \{n \in \mathbb{N} : n \notin f(n)\}.$$

Ein weiteres, sehr ähnliches Beispiel für einen Beweis durch Widerspruch haben wir bereits im Zusammenhang mit der Russellschen Antinomie kennengelernt:

Satz 2.47 (“Es gibt keine Menge aller Mengen”).

Es gibt keine Menge U , so dass für jede Menge M gilt: $M \in U$.

Beweis: Durch Widerspruch. Angenommen, U ist eine Menge, so dass für jede Menge M gilt: $M \in U$. Dann ist auch

$$N := \{M \in U : M \text{ ist eine Menge und } M \notin M\} \quad (*)$$

eine Menge. Insbesondere gilt entweder $N \in N$ oder $N \notin N$.

Fall 1: $N \notin N$:

Wir wissen: N ist eine Menge, also insbesondere $N \in U$.

Da wir in Fall 1 sind, gilt außerdem: $N \notin N$.

Gemäß (*) (für $M := N$) muss dann aber gelten: $N \in N$. \downarrow (Widerspruch zu “Fall 1: $N \notin N$ ”).

Fall 2: $N \in N$:

Wegen $N \in N$ gilt gemäß (*) für $M := N$, dass $N \in U$ ist, dass N eine Menge ist, und dass $N \notin N$ ist. \downarrow (Widerspruch zu “Fall 2: $N \in N$ ”).

Somit führen beide Fälle zu einem Widerspruch. Daher kann es keine Menge U geben, so dass für jede Menge M gilt: $M \in U$. \square

Bemerkung 2.48. Jede Aussage, die durch einen *Beweis durch Kontraposition* bewiesen werden kann, kann auch durch einen *Beweis durch Widerspruch* nachgewiesen werden. Um zu zeigen, dass die Aussage

“Falls Aussage A gilt, so gilt auch Aussage B ”

wahr ist, kann man in einem Beweis durch Widerspruch folgendermaßen vorgehen: Man nimmt an, dass Aussage A gilt und Aussage B nicht gilt und leitet aus dieser Annahme dann einen Widerspruch her.

Übung: Beweisen Sie Satz 2.44 durch einen “Beweis durch Widerspruch”.

2.5.5 Beweistechnik “Beweis durch vollständige Induktion”

Um die Grundidee der vollständigen Induktion zu erklären, sei $A(n)$ eine Aussage über die natürliche Zahl n . Das Ziel ist, zu zeigen, dass die Aussage $A(n)$ für jedes $n \in \mathbb{N}$ wahr ist.

Induktionsprinzip

Eine Möglichkeit, dies zu zeigen ist, sich das so genannte **Induktionsprinzip** zu Nutze zu machen: Man zeigt, dass eine Aussage $A(n)$ für alle $n \in \mathbb{N}$ wahr ist, indem man folgendermaßen vorgeht.

Induktionsanfang

(1) Zuerst zeigt man, dass die Aussage $A(n)$ für die Zahl $n = 0$ gilt.
Diesen Schritt nennt man **Induktionsanfang** bzw. Induktionsbasis.

Induktionsschritt

(2) Danach zeigt man, dass für jede beliebige natürliche Zahl $n \in \mathbb{N}$ gilt:
Falls die Aussage $A(n)$ wahr ist, so ist auch die Aussage $A(n + 1)$ wahr.
Diesen Schritt nennt man **Induktionsschritt**.

Beachte:

Wenn man die Schritte (1) und (2) bewiesen hat, so weiß man, dass die folgenden Aussagen wahr sind:

- (i) $A(0)$ ist wahr gemäß Schritt (1).

- (ii) $A(1)$ ist wahr gemäß (i) und Schritt (2) für $n = 0$,
- (iii) $A(2)$ ist wahr gemäß (ii) und Schritt (2) für $n = 1$,
- (iv) $A(3)$ ist wahr gemäß (iii) und Schritt (2) für $n = 2$,
- (v) $A(4)$ ist wahr gemäß (iv) und Schritt (2) für $n = 3$,
- (vi) $A(5)$ ist wahr gemäß (v) und Schritt (2) für $n = 4$,
- (vii) usw.

Insgesamt hat man damit gezeigt, dass **für alle** $n \in \mathbb{N}$ die Aussage $A(n)$ wahr ist.

Als Beispiel für einen Beweis durch vollständige Induktion betrachten wir den folgenden Satz:

Satz 2.49. F.a. $n \in \mathbb{N}$ gilt:

$$\sum_{i=0}^n 2^i = 2^{n+1} - 1.$$

Beweis: Per Induktion nach n .

Die "Aussage $A(n)$ ", deren Gültigkeit hier f.a. $n \in \mathbb{N}$ bewiesen werden soll, besagt:

$$\sum_{i=0}^n 2^i = 2^{n+1} - 1.$$

INDUKTIONSANFANG: $n = 0$

Behauptung: $\sum_{i=0}^0 2^i = 2^{0+1} - 1$.

Beweis:

Es gilt: $\sum_{i=0}^0 2^i = 2^0 = 1$.

Außerdem gilt: $2^{0+1} - 1 = 2^1 - 1 = 2 - 1 = 1$.

Somit: $\sum_{i=0}^0 2^i = 1 = 2^{0+1} - 1$.

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ beliebig.

Induktionsannahme: $\sum_{i=0}^n 2^i = 2^{n+1} - 1$

(D.h. wir gehen davon aus, dass die Aussage $A(n)$ wahr ist.)

Behauptung: $\sum_{i=0}^{n+1} 2^i = 2^{(n+1)+1} - 1$

(D.h. wir müssen zeigen, dass dann auch die Aussage $A(n + 1)$ wahr ist.)

Beweis:

$$\begin{aligned}\sum_{i=0}^{n+1} 2^i &\stackrel{\text{Not. 2.20}}{=} \left(\sum_{i=0}^n 2^i \right) + 2^{n+1} \\ &\stackrel{\text{Ind. ann.}}{=} (2^{n+1} - 1) + 2^{n+1} \\ &= 2 \cdot 2^{n+1} - 1 \\ &= 2^{(n+1)+1} - 1.\end{aligned}$$

□

Zwei nützliche Varianten des Induktionsprinzips:

Um zu zeigen, dass eine Aussage $A(n)$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$ wahr ist (wobei n_0 eine geeignete natürliche Zahl ist), kann man nach einem der beiden folgenden Schemata vorgehen:

Variante 1:

INDUKTIONSANFANG: $n = n_0$

Behauptung: Die Aussage $A(n_0)$ ist wahr.

Beweis: ...

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq n_0$ beliebig.

Induktionsannahme: Die Aussage $A(n)$ ist wahr.

Behauptung: Die Aussage $A(n + 1)$ ist wahr.

Beweis: ...

Variante 2:

INDUKTIONSANFANG: $n = n_0$

Behauptung: Die Aussage $A(n_0)$ ist wahr.

Beweis: ...

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq n_0$ beliebig.

Induktionsannahme: Für jede natürliche Zahl i mit $n_0 \leq i \leq n$ ist die Aussage $A(i)$ wahr.

Behauptung: Die Aussage $A(n + 1)$ ist wahr.

Beweis: ...

Beispiel 2.50.

Wir nutzen Variante 1, um die folgende Frage zu beantworten: Welche der Funktionen $f: \mathbb{N} \rightarrow \mathbb{Z}$ und $g: \mathbb{N} \rightarrow \mathbb{Z}$ mit $f(n) := n^2 - 7$ und $g(n) := 4 \cdot n$ (f.a. $n \in \mathbb{N}$) liefert größere Funktionswerte?

Um eine Vermutung darüber zu bekommen, welche der beiden Funktionen die größeren Werte liefert, stellen wir zunächst eine Tabelle auf, die die Funktionswerte für $n = 0$, $n = 1$, $n = 2$, etc. enthält:

n	0	1	2	3	4	5	6	7	8	9
$f(n)$	-7	-6	-3	2	9	18	29	42	57	74
$g(n)$	0	4	8	12	16	20	24	28	32	36

Anhand dieser Tabelle drängt sich die Vermutung auf, dass f.a. $n \in \mathbb{N}$ mit $n \geq 6$ gilt: $f(n) > g(n)$. Die Korrektheit dieser Vermutung weisen wir im Folgenden per Induktion nach n nach.

INDUKTIONSANFANG: $n = 6$

Behauptung: $f(6) > g(6)$

Beweis:

Es gilt: $f(6) = 6^2 - 7 = 29$.

Außerdem gilt: $g(6) = 4 \cdot 6 = 24$.

Also: $f(6) = 29 > 24 = g(6)$.

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq 6$ beliebig.

Induktionsannahme: $f(n) > g(n)$, d.h. $n^2 - 7 > 4 \cdot n$.

Behauptung: $f(n + 1) > g(n + 1)$, d.h. $(n + 1)^2 - 7 > 4 \cdot (n + 1)$.

Beweis:

$$\begin{aligned}
 (n + 1)^2 - 7 &= n^2 + 2n + 1 - 7 \\
 &= (n^2 - 7) + 2n + 1 \\
 &\stackrel{\text{Ind.ann}}{>} 4n + 2n + 1 \\
 &\stackrel{n \geq 6, \text{ also } 2n + 1 \geq 13 > 4}{\geq} 4n + 4 \\
 &= 4(n + 1).
 \end{aligned}$$

Insgesamt haben wir damit bewiesen, dass f.a. $n \in \mathbb{N}$ mit $n \geq 6$ gilt: $f(n) > g(n)$.

□ Ende Beispiel 2.50

Auf ähnliche Weise kann man per Induktion auch Folgendes beweisen:

Satz 2.51.

(a) F.a. $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n i = \frac{n \cdot (n+1)}{2}$.

(b) F.a. $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n (2i - 1) = n^2$

(d.h. die Summe der ersten n ungeraden Zahlen ergibt gerade die Zahl n^2).

(c) F.a. $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n i^2 = \frac{n \cdot (n+1) \cdot (2n+1)}{6}$.

(d) F.a. $n \in \mathbb{N}$ gilt: $2^n > n$.

Beweis: Übung.

□

Das folgende Beispiel zeigt, dass man beim Führen von Induktionsbeweisen sehr sorgfältig sein muss:

Beispiel 2.52.

Der folgende Satz ist offensichtlich nicht wahr — aber wo steckt der Fehler im Beweis?

“**Satz**”: F.a. $n \in \mathbb{N}$ mit $n \geq 1$ gilt: Ist M eine Menge von Menschen mit $|M| = n$, so haben alle Menschen in M die gleiche Größe.

“*Beweis*”: Per Induktion nach n .

INDUKTIONSANFANG: $n = 1$

Behauptung: Ist M eine Menge von Menschen mit $|M| = 1$, so haben alle Menschen in M die gleiche Größe.

Beweis: Sei M eine Menge von Menschen mit $|M| = 1$. D.h. M besteht aus genau einem Menschen. Daher haben offensichtlich alle Menschen in M die gleiche Größe.

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq 1$ beliebig.

Induktionsannahme: Ist M' eine Menge von Menschen mit $|M'| = n$, so haben alle Menschen in M' die gleiche Größe.

Behauptung: Ist M eine Menge von Menschen mit $|M| = n + 1$, so haben alle Menschen in M die gleiche Größe.

Beweis: Sei M eine Menge von Menschen mit $|M| = n + 1$. Sei $a_1, a_2, \dots, a_n, a_{n+1}$ eine Liste aller Menschen in M , d.h. $M = \{a_1, a_2, \dots, a_n, a_{n+1}\}$. Sei

$$M' := \{a_1, a_2, \dots, a_n\} \quad \text{und} \quad M'' := \{a_2, \dots, a_n, a_{n+1}\}.$$

Offensichtlich sind M' und M'' Mengen von Menschen mit $|M'| = n$ und $|M''| = n$. Gemäß der Induktionsannahme gilt daher:

- (1) Alle Menschen in M' haben die gleiche Größe, und
- (2) alle Menschen in M'' haben die gleiche Größe.

Sei g' die Größe, die gemäß (1) jeder Mensch in M' hat, und sei g'' die Größe, die gemäß (2) jeder Mensch in M'' hat. Laut Definition von M' und M'' gilt: $a_2 \in M'$ und $a_2 \in M''$. Da jeder einzelne Mensch (und daher insbes. der Mensch a_2) nur *eine* Größe haben kann, gilt: $g' = g''$. Wegen $M = M' \cup M''$ gilt daher, dass alle Menschen in M die gleiche Größe haben, nämlich die Größe $g := g' = g''$. □

Frage: Wo steckt der Fehler im Beweis?

□ Ende Beispiel 2.52

2.6 Rekursive Definitionen von Funktionen und Mengen

2.6.1 Rekursive Definitionen von Funktionen

Das Induktionsprinzip lässt sich auch zur “induktiven” (bzw. “rekursiven”) Definition von Funktionen $f: \mathbb{N} \rightarrow M$ (wobei M eine beliebige Menge ist) nutzen, indem man folgendermaßen vorgeht:

- (1) Definiere $f(0)$.
Diesen Schritt bezeichnet man als **Rekursionsanfang**. Rekursionsanfang
- (2) Definiere, f.a. $n \in \mathbb{N}$, $f(n+1)$ unter Verwendung des Werts $f(n)$
(bzw. unter Verwendung der Werte $f(n), f(n-1), \dots, f(1), f(0)$).
Diesen Schritt bezeichnet man als **Rekursionsschritt**. Rekursionsschritt

Auch hier sind wieder eine Reihe von Varianten möglich.

Beispiel 2.53.

- (a) *Frage:* Wie viele Möglichkeiten gibt es, n Studierende so an n PCs zu verteilen, dass an jedem PC genau ein Studierender sitzt?

Antwort: fak(n), wobei

- fak(1) = 1 und
- fak($n+1$) = ($n+1$) · fak(n) (für alle $n \in \mathbb{N}_{>0}$).

Insbesondere ist fak eine Funktion von $\mathbb{N}_{>0}$ nach $\mathbb{N}_{>0}$, d.h. fak: $\mathbb{N}_{>0} \rightarrow \mathbb{N}_{>0}$.

Beispielsweise ist

$$\text{fak}(4) = 4 \cdot \text{fak}(3) = 4 \cdot 3 \cdot \text{fak}(2) = 4 \cdot 3 \cdot 2 \cdot \text{fak}(1) = 4 \cdot 3 \cdot 2 \cdot 1 = 24.$$

Allgemein gilt f.a. $n \in \mathbb{N}_{>0}$:

$$\text{fak}(n) = n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot 2 \cdot 1 \stackrel{\text{Not. 2.20}}{=} \prod_{i=1}^n i.$$

Notation:

Die Funktion fak wird **Fakultätsfunktion** genannt. Meistens schreibt man $n!$ um die Zahl fak(n) zu bezeichnen. Fakultätsfunktion $n!$

- (b) *Fragestellung:* Ein Bauer züchtet Kaninchen. Jedes weibliche Kaninchen bringt im Alter von zwei Monaten ein weibliches Kaninchen zur Welt und danach jeden Monat ein weiteres. Wie viele weibliche Kaninchen hat der Bauer am Ende des n -ten Monats, wenn er mit einem neu geborenen weiblichen Kaninchen startet?

Antwort: fib(n), wobei die Funktion fib: $\mathbb{N}_{>0} \rightarrow \mathbb{N}_{>0}$ rekursiv wie folgt definiert ist:

- fib(1) := 1,
- fib(2) := 1 und
- fib($n+1$) := fib(n) + fib($n-1$) (f.a. $n \in \mathbb{N}$, $n \geq 2$).

Somit gilt:

n		1		2		3		4		5		6		7		8		9		10		11		12
fib(n)		1		1		2		3		5		8		13		21		34		55		89		144

Die Funktion fib wird auch **Fibonacci-Folge** genannt; sie ist benannt nach italienischen Mathematiker Leonardo Fibonacci (13. Jh.). Die Zahl fib(n) heißt auch n -te **Fibonacci-Zahl**. Fibonacci-Folge

Um Aussagen über rekursiv definierte Funktionen zu beweisen, kann man wieder das Induktionsprinzip nutzen. Der folgende Satz gibt dazu ein Beispiel.

Satz 2.54. Sei $\text{fib}: \mathbb{N}_{>0} \rightarrow \mathbb{N}_{>0}$ die Fibonacci-Folge. Dann gilt f.a. $n \in \mathbb{N}_{>0}$: $\text{fib}(n) \leq 2^n$.

Beweis: Per Induktion nach n .

INDUKTIONSANFANG: Betrachte $n = 1$ und $n = 2$.

Behauptung: $\text{fib}(1) \leq 2^1$ und $\text{fib}(2) \leq 2^2$.

Beweis: Es gilt: $\text{fib}(1) \stackrel{\text{Def.}}{=} 1 \leq 2 = 2^1$ und $\text{fib}(2) \stackrel{\text{Def.}}{=} 1 \leq 4 = 2^2$.

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq 2$ beliebig.

Induktionsannahme: F.a. $i \in \mathbb{N}_{>0}$ mit $i \leq n$ gilt: $\text{fib}(i) \leq 2^i$.

Behauptung: $\text{fib}(n + 1) \leq 2^{n+1}$.

Beweis: $\text{fib}(n + 1) \stackrel{\text{Def.}}{=} \text{fib}(n) + \text{fib}(n - 1) \stackrel{\text{Ind.ann.}}{\leq} 2^n + 2^{n-1} \leq 2 \cdot 2^n = 2^{n+1}$. □

Bemerkung 2.55. Ein möglicher Algorithmus, um für eine Zahl $n \in \mathbb{N}_{>0}$ den Wert $\text{fib}(n)$ der Fibonacci-Folge zu berechnen, ist:

Algo 1 (bei Eingabe einer Zahl $n \in \mathbb{N}_{>0}$):

1. Falls $n = 1$ oder $n = 2$, dann gib 1 als Ergebnis zurück.
2. Falls $n \geq 3$, dann:
 3. Sei x_1 die Ausgabe von *Algo 1* bei Eingabe der Zahl $n - 1$.
 4. Sei x_2 die Ausgabe von *Algo 1* bei Eingabe der Zahl $n - 2$.
 5. Gib den Wert $(x_1 + x_2)$ als Ergebnis zurück.

Der Algorithmus benötigt bei Eingabe einer Zahl n höchstens $g_1(n)$ Schritte, wobei

$$\begin{aligned} g_1(1) &= 2 \quad \text{und} \quad g_1(2) = 3 \quad \text{und} \\ g_1(n) &= 3 + g_1(n - 1) + g_1(n - 2) + 2 \\ &= 5 + g_1(n - 1) + g_2(n - 2) \quad \text{für alle } n \in \mathbb{N} \text{ mit } n \geq 3 \end{aligned}$$

(wir zählen hier jede Addition, jeden Vergleich, und jedes Zurückgeben eines Ergebnisses als einen Schritt).

Ein anderer Algorithmus, der für eine Zahl $n \in \mathbb{N}_{>0}$ den Wert $\text{fib}(n)$ berechnet, ist:

Algo 2 (bei Eingabe einer Zahl $n \in \mathbb{N}_{>0}$):

1. Falls $n = 1$ oder $n = 2$, dann gib 1 als Ergebnis zurück.
2. Seien $a_0 := 0$, $a_1 := 1$ und $a_2 := 1$.
3. Wiederhole für alle i von 3 bis n :
 4. Ersetze a_0 durch a_1 und a_1 durch a_2 .
 5. Ersetze a_2 durch $a_0 + a_1$.

6. Gib den Wert a_2 als Ergebnis zurück.

Dieser Algorithmus benötigt bei Eingabe $n \in \mathbb{N}_{>0}$ höchstens $g_2(n) := 6 + 5 \cdot (n - 2)$ Schritte (ähnlich wie oben zählen wir jeden Vergleich, jedes Zurückgeben eines Werts und jedes Setzen eines Werts als einen Schritt. Für jeden Schleifendurchlauf berechnen wir zusätzlich 2 Schritte, um den i um eins zu erhöhen und zu testen, ob das Ergebnis kleiner oder gleich n ist).

Frage: Welcher der beiden Algorithmen läuft im Allgemeinen schneller? D.h. welche der beiden Funktionen g_1 und g_2 liefert kleinere Funktionswerte?

Mit den in diesem Kapitel bereitgestellten Werkzeugen können wir eine Antwort auf diese Frage finden, und wir können sogar beweisen, dass die Antwort korrekt ist.

Bemerkung 2.56. Es gibt auch eine “geschlossene Formel”, mit der man die n -te Fibonacci-Zahl, d.h. die Zahl $\text{fib}(n)$, direkt ausrechnen kann, ohne dafür sämtliche Werte $\text{fib}(1), \text{fib}(2), \dots, \text{fib}(n - 1)$ ausrechnen zu müssen:

F.a. $n \in \mathbb{N}_{>0}$ gilt:

$$\text{fib}(n) = \frac{1}{\sqrt{5}} \cdot \left(\left(\frac{1 + \sqrt{5}}{2} \right)^n - \left(\frac{1 - \sqrt{5}}{2} \right)^n \right).$$

Beweis: Übung (per Induktion nach n ; Details finden sich in [22]).

□

2.6.2 Rekursive Definitionen von Mengen

Oft ist es nützlich, auch **Mengen** rekursiv (bzw. induktiv) zu definieren. Eine rekursive Definition einer Menge M besteht aus:

(a) **Basisregeln** der Form “ $m \in M$ ”.

(D.h. die Basisregeln listen explizit bestimmte Elemente auf, die zur Menge M gehören.)

(b) **Rekursiven Regeln** der Form:

“Wenn $m_1, \dots, m_k \in M$, dann $m \in M$ ”,

wobei m von m_1, \dots, m_k abhängt.

Die dadurch definierte Menge M ist dann die Menge aller Elemente, deren Zugehörigkeit zu M durch endlich-maliges Anwenden der Regeln gezeigt werden kann.

Beispiel 2.57 (Die Menge PAL).

Betrachte das Alphabet $A := \{a, b\}$. Die Menge $\text{PAL} \subseteq A^*$ sei wie folgt rekursiv definiert:

Basisregeln:

(B1): $\varepsilon \in \text{PAL}$.

(B2): $a \in \text{PAL}$.

(B3): $b \in \text{PAL}$.

Rekursive Regeln:

(R1): Ist $w \in \text{PAL}$, so ist auch $awa \in \text{PAL}$.

(R2): Ist $w \in \text{PAL}$, so ist auch $bwb \in \text{PAL}$.

Beispiele für Worte, die zur Menge PAL gehören:

$\underbrace{\varepsilon, a, b}$ durch Basisregeln $\underbrace{aa, bb}$ durch rek. Regeln mit $w := \varepsilon$ $\underbrace{aaa, bab}$ durch rek. Regeln mit $w := a$ $\underbrace{aba, bbb}$ durch rek. Regeln mit $w := b$

Es gilt beispielsweise auch: $aababaa \in \text{PAL}$.

Beweis:

- $a \in \text{PAL}$ (gemäß Basisregel (B1)).
- Regel (R2) mit $w := a \implies bab \in \text{PAL}$.
- Regel (R1) mit $w := bab \implies ababa \in \text{PAL}$.
- Regel (R1) mit $w := ababa \implies aababaa \in \text{PAL}$.

□

Aber beispielsweise gilt

$$aab \notin \text{PAL},$$

denn aus den Basisregeln und den rekursiven Regeln folgt, dass für jedes Wort $w \in \text{PAL}$ der erste und der letzte Buchstabe von w identisch sind. □ Ende Beispiel 2.57

Induktionsprinzip für rekursiv definierte Mengen:

Sei M eine rekursiv definierte Menge. Dass eine Aussage $A(m)$ für alle $m \in M$ wahr ist, kann man folgendermaßen zeigen:

- (1) Zuerst betrachtet man nacheinander jede Basisregel der Form " $m \in M$ " und zeigt, dass die Aussage $A(m)$ wahr ist.
Dieser Schritt heißt **Induktionsanfang**.
- (2) Danach betrachtet man nacheinander jede rekursive Regel der Form "Wenn $m_1, \dots, m_k \in M$, dann $m \in M$ " und zeigt folgendes: Wenn die Aussagen $A(m_1), \dots, A(m_k)$ wahr sind, dann ist auch die Aussage $A(m)$ wahr.
Dieser Schritt heißt **Induktionsschritt**.

Beachte: Man kann leicht sehen, dass folgendes gilt: Wenn man die Schritte (1) und (2) bewiesen hat, so weiß man, dass die Aussage $A(m)$ für alle $m \in M$ wahr ist.

Im Folgenden betrachten wir ein Beispiel dafür, wie das Induktionsprinzip dazu genutzt werden kann, Eigenschaften von rekursiv definierten Mengen nachzuweisen.

Beispiel 2.58 (Palindrome).

Sei $A := \{a, b\}$. Für jedes Wort $w \in A^*$ sei w^R das Wort, das durch "Rückwärtslesen" von w entsteht, d.h.:

- Ist $w = \varepsilon$, so ist $w^R = \varepsilon$.
- Ist $w = w_1 \cdots w_k$ mit $k \in \mathbb{N}_{>0}$ und $w_1, \dots, w_k \in A$, so ist $w^R := w_k \cdots w_1$.

Beispiel: $aaab^R = baaa$.

Sei PAL die im Beispiel 2.57 rekursiv definierte Teilmenge von A^* .

Behauptung 1: Für jedes Wort $w \in \text{PAL}$ gilt: $w = w^R$.

Beweis: Per Induktion über den Aufbau von PAL.

INDUKTIONSANFANG: Betrachte diejenigen Worte, die aufgrund von Basisregeln zur Menge PAL gehören.

Behauptung: $\varepsilon = \varepsilon^R$, $a = a^R$ und $b = b^R$.

Beweis: Gemäß der Definition von w^R gilt offensichtlich, dass $\varepsilon = \varepsilon^R$, $a = a^R$ und $b = b^R$.

INDUKTIONSSCHRITT: Betrachte die rekursiven Regeln.

- (R1): Sei $w \in \text{PAL}$ und sei $v := awa$. Gemäß (R1) ist $v \in \text{PAL}$.

Induktionsannahme: $w = w^R$.

Behauptung: $v = v^R$.

Beweis: $v^R \stackrel{\text{Def. } v}{=} (awa)^R \stackrel{\text{Def. } (\cdot)^R}{=} aw^R a \stackrel{\text{Ind.ann.: } w = w^R}{=} awa \stackrel{\text{Def. } v}{=} v$.

- (R2): Sei $w \in \text{PAL}$ und sei $v := bwb$. Gemäß (R2) ist $v \in \text{PAL}$.

Induktionsannahme: $w = w^R$.

Behauptung: $v = v^R$.

Beweis: $v^R \stackrel{\text{Def. } v}{=} (bwb)^R \stackrel{\text{Def. } (\cdot)^R}{=} bw^R b \stackrel{\text{Ind.ann.: } w = w^R}{=} bwb \stackrel{\text{Def. } v}{=} v$. □_{Beh. 1}

Behauptung 2: Für jedes $w \in A^*$ mit $w = w^R$ gilt: $w \in \text{PAL}$.

Beweisansatz: Zeige folgende Aussage per Induktion nach n :

Für alle $n \in \mathbb{N}$ gilt: Ist $w \in A^*$ mit $w = w^R$ und $|w| \leq n$, so gilt $w \in \text{PAL}$.

Im Induktionsanfang werden $n = 0$ und $n = 1$ betrachtet; im Induktionsschritt $n \rightarrow n + 1$ werden alle $n \geq 1$ betrachtet.

Details: Übung. □_{Beh. 2}

Aus Behauptung 1 und Behauptung 2 folgt, dass $\text{PAL} = \{w \in A^* : w = w^R\}$.

□_{Ende Beispiel 2.58}

Antwort auf die Frage aus Beispiel 2.52:

Der "Induktionsschritt $n \rightarrow n + 1$ " ist für den Wert $n = 1$ nicht schlüssig, denn in diesem Fall gilt $n + 1 = 2$ und

- $M = \{a_1, a_2\}$,
- $M' = \{a_1\}$,
- $M'' = \{a_2\}$.

Insbesondere gilt also zwar, dass $a_2 \in M''$, aber es gilt nicht, dass $a_2 \in M'$.

2.7 Literaturhinweise zu Kapitel 2

Als vertiefende Lektüre seien die Kapitel 3, 6 und 7 in [22] empfohlen. Wertvolle Tipps und Tricks zur Formulierung mathematischer Gedanken und Beweise finden sich in [2]. Einen Crashkurs in die diskrete Mathematik für Informatiker/innen gibt das Buch [14]. Eine umfassende Einführung in die Mengenlehre gibt das Lehrbuch [6].

Quellennachweis: Teile der Abschnitte 2.1–2.3 sowie 2.6 orientieren sich an [9]. Das in Abschnitt 2.4 betrachtete Beispiel ist aus [15] entnommen. Teile von Abschnitt 2.5 orientieren sich an [22]. Die folgende Aufgabe 2.17 ist aus [15] entnommen.

2.8 Übungsaufgaben zu Kapitel 2

Aufgabe 2.1.

(a) Beschreiben Sie die folgenden Mengen sprachlich, wie beispielsweise in Aufgabenteil (b).

$$(i) \{3n : n \in \mathbb{N}\} \quad (ii) \{\sqrt{k} : k \in \mathbb{N}_{>0}\} \quad (iii) \{x \in \mathbb{N} : x+7 = x+11\}$$

(b) Geben Sie die folgenden Mengen in intensionaler Form, also wie in Aufgabenteil (a) an.

(i) Die Menge aller ganzen Zahlen, die größer als -12 und kleiner als 12 sind.

(ii) Die Menge aller natürlichen Zahlen, die nicht durch 3 teilbar sind.

(iii) Die Menge aller Zahlen, die das Quadrat einer natürlichen Zahl sind.

Aufgabe 2.2. Es sei $M := \{2, 5, 8\}$ und $N := \{3, 5, 7, 11\}$. Schreiben Sie die folgenden Mengen in extensionaler Form auf und geben Sie ihre Kardinalität an.

$$\begin{array}{lll} (a) M \cup N & (b) M \setminus N & (c) \mathcal{P}(M) \\ (d) \mathcal{P}(\{\emptyset\}) & (e) M \times \{a, b\} & (f) \{M\} \times \{a, b\} \\ (g) \{P : P \subseteq N \text{ und } |P| = 2\} & (h) N^2 \setminus \{(x, x) : x \in N\} \end{array}$$

Aufgabe 2.3. Sei $U := \{1, 2, \dots, 10\}$ ein festes Universum, und seien $M := \{1, 3, 5\}$, $N := \{2, 3, 5, 7\}$ und $P := \{1, 4, 9\}$. Schreiben Sie jede der folgenden Mengen in extensionaler Form auf und geben Sie ihre Kardinalität an.

$$\begin{array}{lll} (a) M \setminus (N \cup P) & (d) (M \cap \bar{P}) \cup (N \cap \bar{P}) & (g) M \times P \times \{a, b\} \\ (b) (M \setminus N) \cup (M \setminus P) & (e) M^2 \setminus (N \times P) & (h) \{Q : Q \subseteq N, |Q| = 3\} \\ (c) (M \cup N) \cap \bar{P} & (f) \mathcal{P}(N) \end{array}$$

Aufgabe 2.4. Sei $U := \{1, 2, \dots, 10\}$ ein festes Universum, und seien $M := \{7\}$, $N := \{1, 2, 9, 10\}$ und $P := \{2, 3, 5, 7\}$. Schreiben Sie jede der folgenden Mengen in extensionaler Form auf und geben Sie ihre Kardinalität an.

- (a) $(N \setminus P) \cap M$ (d) $(\overline{N \cup P}) \cup (N \cap P)$ (g) $((P \cap N) \cup M) \times \{4, 8\} \times (P \setminus M)$
 (b) $(\overline{P \setminus M})$ (e) $(P \setminus N)^2$ (h) $\{Q : Q \subseteq N, |Q| \text{ ist gerade}\}$
 (c) $(P \Delta N) \cup M$ (f) $\mathcal{P}(\overline{N \cup P})$

Aufgabe 2.5.

(a) Berechnen Sie den Wert der folgenden Ausdrücke

(i) $\sum_{i=1}^5 2i$ (ii) $\prod_{j=4}^6 (3j + 1)$ (iii) $\sum_{k=2}^4 \prod_{l=1}^3 kl$

(b) Es seien die Mengen $A_1 := \{1, 2, 3\}$, $A_2 := \{2, 3, 4\}$, $A_3 := \{3, 4, 5\}$ und $A_4 := \{4, 5, 6\}$ gegeben. Beschreiben Sie die folgenden Mengen in extensionaler Form.

(i) $\bigcup_{i=1}^3 A_i$ (ii) $\bigcap_{j=1}^4 A_j$ (iii) $\bigcup_{k=2}^4 (A_k \setminus A_{k-1})$

Aufgabe 2.6. Geben Sie an, welche der folgenden Aussagen richtig und welche falsch sind.

- (a) $\{1, 3\} \subseteq \{1, 3, \{1, 2\}\}$ (c) $\{1, \{\emptyset\}\} \subseteq \{1, 3, \{\emptyset\}\}$ (e) $\{\emptyset\} \subseteq \{1, 3, \{\emptyset\}\}$
 (b) $\{1, 3\} \in \{1, 3, \{1, 2\}\}$ (d) $\emptyset \subseteq \{1, 3, \{\emptyset\}\}$ (f) $\{\emptyset\} \in \{1, 3, \{\emptyset\}\}$

Aufgabe 2.7.

(a) Für jede der folgenden Behauptungen beweisen Sie, dass die Behauptung für alle Mengen M, N, P gilt, oder widerlegen Sie die Behauptung, indem Sie Mengen M, N, P angeben und zeigen, dass die Behauptung für diese Mengen nicht gilt:

- (i) Falls $M \subseteq N$ und $N \subsetneq P$, dann $M \subsetneq P$.
 (ii) Falls $M \subseteq N$ und $N \not\subseteq P$, dann $M \not\subseteq P$.
 (iii) Falls $M \cup N \subseteq P$, dann $M \subseteq P$ und $N \subseteq P$.
 (iv) Falls $M \cap N \subseteq P$, dann $M \subseteq P$ oder $N \subseteq P$.
 (v) Falls $M \in N$ und $N \in P$, dann $M \in P$.

(b) Geben Sie Mengen M, N und P in extensionaler Form an, so dass (v) von Aufgabenteil (a) erfüllt ist, d.h. es soll gelten $M \in N, N \in P$ und $M \in P$.

Aufgabe 2.8.

(a) Welche der Gleichungen stimmt, welche stimmt nicht?

- a) $(M \cap N) \setminus P = (M \setminus P) \cap (N \setminus P)$
 b) $(M \cap N) \setminus P = (M \setminus P) \cup (N \setminus P)$

(b) Begründen Sie Ihre Antwort aus (a) durch Betrachtung von Venn-Diagrammen.

(c) Beweisen Sie Ihre Antworten aus Teil (a).

Aufgabe 2.9.

(a) Bestimmen Sie mit Hilfe von Venn-Diagrammen, welche der folgenden Behauptungen für alle Mengen M, N, P gilt, und welche nicht für alle Mengen M, N, P gilt:

(i) $M \setminus (N \cup P) = (M \setminus N) \cup (M \setminus P)$

(ii) $M \cap N = M \setminus (M \setminus N)$

(b) Beweisen Sie, dass Ihre Antworten aus (a) korrekt sind.

Aufgabe 2.10. Beweisen Sie die Korrektheit der folgenden Gleichungen.

(a) $M \cap (N \cup P) = (M \cap N) \cup (M \cap P)$.

(b) $\overline{M \cap N} = \overline{M} \cup \overline{N}$.

Aufgabe 2.11. Seien A, B, C, D, E Teilmengen von \mathbb{N} , die wie folgt definiert sind:

$$A = \{3n : n \in \mathbb{N}\} \quad B = \{5n : n \in \mathbb{N}\} \quad C = \{15n : n \in \mathbb{N}\}$$

$$D = \{6n : n \in \mathbb{N}\} \quad E = \{12n : n \in \mathbb{N}\}$$

(a) Welche der folgenden Aussagen sind richtig und welche sind falsch?

(i) $E \subseteq D \subseteq A$ (ii) $E \subseteq C$ (iii) $A \cap B \subseteq C$ (iv) $A \cup B \subseteq C$

(b) Berechnen Sie die folgenden Mengen:

(i) $A \cup C$ (ii) $A \cap E$ (iii) $B \cap D$ (iv) $C \setminus B$

Aufgabe 2.12. Ein Informatikstudent hat 30 Informatikbücher von der Bibliothek ausgeliehen, die sich u.a. mit den Gebieten Algorithmik, Betriebssysteme und Compilerbau beschäftigen. Sei A die Menge der Bücher, die sich u.a. mit Algorithmik beschäftigen, B die Menge der Bücher, die sich u.a. mit Betriebssystemen beschäftigen und C die Menge der Bücher, die sich u.a. mit Compilerbau beschäftigen. Folgende Information über die Anzahl der Bücher und die von ihnen behandelten Themen ist bekannt:

$$|A| = 14, \quad |B| = 18, \quad |C| = 16, \quad |A \cap B| = 8, \quad |A \cap C| = 7, \quad |B \cap C| = 10, \quad |A \cap B \cap C| = 3.$$

(a) Wie viele der Bücher enthalten Material aus mindestens einem der genannten Gebiete?

D.h. berechnen Sie $|A \cup B \cup C|$.

(b) Wie viele der Bücher enthalten Material aus mindestens zwei der genannten Gebiete?

D.h. berechnen Sie $|D|$, wobei $D := (A \cap B) \cup (A \cap C) \cup (B \cap C)$.

(c) Wie viele der Bücher enthalten Material aus genau einem der genannten Gebiete?

D.h. berechnen Sie $|(A \cup B \cup C) \setminus D|$, wobei D die Menge aus (b) ist.

Hinweis: Überlegen Sie sich zunächst anhand von Venn-Diagrammen, wie man die Kardinalitäten der Mengen berechnen kann.

Aufgabe 2.13. Abgeordnete des Deutschen Bundestages bilden Ausschüsse, die sich mit einem bestimmten Thema befassen. Sei A die Menge der Abgeordneten, die im Ausschuss Arbeit/Soziales sind und F die Menge der Abgeordneten die sich im Ausschuss Finanzen befinden. Außerdem sei S die Menge der Abgeordneten, die im Sport-Ausschuss sind. Es sind folgende Informationen über die Anzahl der Abgeordneten in den verschiedenen Ausschüssen bekannt:

$$|A| = 17, \quad |F| = 18, \quad |S| = 15, \quad |A \cap F| = 8, \quad |A \cap S| = 7, \quad |F \cap S| = 9, \quad |A \cap F \cap S| = 5$$

- (a) Wie viele Abgeordnete sind in mindestens einem der Ausschüsse Mitglied?
D.h. berechnen Sie $|A \cup F \cup S|$.
- (b) Wie viele der Abgeordneten sind in genau zwei Ausschüssen?
D.h. berechnen Sie $|((A \cap F) \cup (A \cap S) \cup (F \cap S)) \setminus (A \cap F \cap S)|$.
- (c) Es soll ein Unterausschuss gebildet werden, dem alle Abgeordneten des Sport-Ausschusses angehören und zusätzlich alle Abgeordneten, die im Arbeit/Soziales- aber nicht im Finanz-Ausschuss sitzen. Wie viele Mitglieder hat dieser Unterausschuss?
D.h. berechnen Sie $|S \cup (A \setminus F)|$.

Hinweis: Überlegen Sie sich zunächst anhand von Venn-Diagrammen, wie man die Kardinalitäten der Mengen berechnen kann.

Aufgabe 2.14.

- (a) Geben Sie alle Relationen von $A := \{x, y\}$ nach $B := \{c, d\}$ an. Geben Sie für jede Relation an, ob sie eine Funktion von A nach B oder eine partielle Funktion von A nach B oder keines von beiden ist. Geben Sie außerdem für jede Funktion an, ob sie injektiv, surjektiv und/oder bijektiv ist.
- (b) Seien M und N beliebige endliche Mengen. Wieviele Relationen von M nach N gibt es?
- (c) Geben Sie für jede der folgenden Funktionen f an, ob die Funktion injektiv, surjektiv und/oder bijektiv ist. Geben Sie jeweils auch das Bild von f an.
- $f: \mathbb{Z} \rightarrow \mathbb{Z}$ mit $f(x) := x - 4$ für alle $x \in \mathbb{Z}$
 - $f: \mathbb{Z} \rightarrow \mathbb{Z}$ mit $f(x) := 2 \cdot x$ für alle $x \in \mathbb{Z}$
 - $f: \mathbb{Z} \rightarrow \{-1, 1\}$ mit $f(x) := (-1)^x$ für alle $x \in \mathbb{Z}$
 - $f: \mathbb{Z} \rightarrow \mathbb{Z}$ mit $f(x) := x^2$ für alle $x \in \mathbb{Z}$
 - $f: \mathbb{N} \rightarrow \mathbb{N}_{>0}$ mit $f(x) := x + 1$ für alle $x \in \mathbb{N}$
 - $f: A^* \rightarrow \mathbb{N}$ für eine beliebige Menge A mit $|A| = 1$ und $f(w) := |w|$ für alle $w \in A^*$
 - $f: A^* \rightarrow \mathbb{N}$ für eine beliebige Menge A mit $|A| \geq 2$ und $f(w) := |w|$ für alle $w \in A^*$
- (d) Wie viele Möglichkeiten gibt es,
- zwei Bälle B_1, B_2 so auf drei Körbe K_1, K_2, K_3 zu verteilen, dass jeder Ball in einem anderen Korb landet? D.h. wie viele injektive Funktionen von $\{B_1, B_2\}$ nach $\{K_1, K_2, K_3\}$ gibt es?
 - drei Bälle B_1, B_2, B_3 so auf zwei Körbe K_1, K_2 zu verteilen, dass kein Korb leer bleibt? D.h. wie viele surjektive Funktionen von $\{B_1, B_2, B_3\}$ nach $\{K_1, K_2\}$ gibt es?

- c) drei Bälle B_1, B_2, B_3 so auf drei Körbe K_1, K_2, K_3 zu verteilen, dass mindestens ein Korb leer bleibt? D.h. wie viele nicht surjektive Funktionen von $\{B_1, B_2, B_3\}$ nach $\{K_1, K_2, K_3\}$ gibt es?

Aufgabe 2.15. Beweisen Sie Satz 2.38(b), d.h.:

Sei B eine Menge, sei A eine endliche Menge und sei $k := |A|$. Zeigen Sie, dass es eine bijektive Funktion von $\text{Abb}(A, B)$ nach B^k gibt.

Aufgabe 2.16.

- (a) Seien A, B und C endliche Mengen und sei $f : A \rightarrow B$ eine Funktion von A nach B und $g : B \rightarrow C$ eine Funktion von B nach C . Wir definieren die Funktion $h : A \rightarrow C$ als *Komposition*, d.h. Hintereinanderausführung, von f und g als $h(x) := g(f(x))$ f.a. $x \in A$. Beweisen Sie die Gültigkeit der folgenden Aussagen:
- (i) Wenn f und g surjektiv sind, so ist auch h surjektiv.
 - (ii) Wenn f und g injektiv sind, so ist auch h injektiv.
 - (iii) Wenn f und g bijektiv sind, so ist auch h bijektiv.
- (b) Seien X und Y endliche Mengen und $f \subseteq X \times Y$ eine Relation von X nach Y . Wir definieren die Relation \tilde{f} als Relation von Y nach X wie folgt:

$$\text{Für alle } x \in X, y \in Y \text{ gilt: } (y, x) \in \tilde{f} : \iff (x, y) \in f,$$

Beweisen Sie, dass die folgende Aussage korrekt ist: f ist genau dann eine bijektive Funktion, wenn \tilde{f} eine bijektive Funktion ist.²

Aufgabe 2.17. In den folgenden Teilaufgaben sollen einige Aspekte einer Variante des Spiels Monopoly mit Wertebereichen modelliert werden. Setzen Sie dabei nur die Menge \mathbb{N} als vordefiniert voraus.

- (a) Auf dem Spielbrett gibt es 40 Felder, wobei 22 von diesen Feldern Straßen und 18 Felder Plätze sind. Die Straßen und Plätze sind von 1 bis 22 bzw. von 1 bis 18 durchnummeriert. Definieren Sie drei Mengen STRASSEN, PLÄTZE und FELDER, deren Elemente Straßen, Plätze bzw. Felder repräsentieren.
- (b) Auf ein Feld vom Typ 'Straße' können beliebig viele Häuser und Hotels platziert werden, deren Anordnung aber keine Rolle spielt.
- (i) Definieren Sie eine Menge BEBAUUNGSZUSTÄNDE, von der jedes Element den Bebauungszustand einer einzelnen Straße (d.h. die Anzahl der Häuser und die Anzahl der Hotels) repräsentiert.
 - (ii) Welches Element von BEBAUUNGSZUSTÄNDE beschreibt, dass sich drei Häuser und vier Hotels auf der Straße befinden?
- (c) Der Zustand eines Spielers ist zu jedem Zeitpunkt bestimmt durch den Geldbetrag, der ihm zur Verfügung steht, der Menge der Straßen, die er besitzt, und dem Feld, auf dem er sich gerade befindet.

²In diesem Falle wird \tilde{f} die *Umkehrfunktion* von f genannt und üblicherweise mit f^{-1} bezeichnet.

- (i) Definieren Sie eine Menge SPIELERZUSTÄNDE, von der jedes Element den Zustand eines Spielers repräsentiert.
 - (ii) Welches Element von SPIELERZUSTÄNDE beschreibt, dass dem Spieler 1000 Euro zur Verfügung stehen, dass er die Straßen 4, 6 und 7 besitzt, und dass er gerade auf der 17. Straße steht?
- (d) Ein Spieler, der eine Straße betritt, die bereits einem anderen Spieler gehört, muss Miete an den Besitzer der Straße entrichten. Die Höhe der Miete hängt von der Straße und deren Bebauungszustand ab.

Geben Sie Mengen A und B an, so dass der oben beschriebene Zusammenhang durch eine Funktion $miete: A \rightarrow B$ modelliert werden kann, d.h. $miete$ soll die Miete für die Straße in Abhängigkeit von der Straße selbst und deren Bebauungszustand angeben.

Aufgabe 2.18. In den folgenden Teilaufgaben sollen einige Aspekte des Rollenspiels *Village of Voidcraft VoV* mit Wertebereichen modelliert werden. Setzen Sie dabei nur die Menge \mathbb{N} als vordefiniert voraus.

- (a) In der Welt von VoV gibt es 80 Orte, von denen 17 Siedlungen, 30 Burgen und 33 Kerker sind. Die Siedlungen sind von 1 bis 17, die Burgen von 1 bis 30 und die Kerker von 1 bis 33 durchnummeriert. Definieren Sie die vier Mengen ORTE, SIEDLUNGEN, BURGEN und KERKER, deren Elemente die Orte, Siedlungen, Burgen und Kerker repräsentieren.
- (b) Jede Gruppe von Gegnern, auf die ein Held in VoV treffen kann, zeichnet sich durch die Anzahl der Orks und die Anzahl der Trolle darin aus, die jeweils beliebig groß sein kann.
 - (i) Definieren Sie die Menge GEGNERHEERE, von der jedes Element eine mögliche Zusammensetzung einer Gegnergruppe (d.h. die Anzahl der Orks und die Anzahl der Trolle darin) repräsentiert.
 - (ii) Welches Element von GEGNERHEERE steht für eine Gegnergruppe, die sich aus 11 Orks und 13 Trollen zusammensetzt?
- (c) Der Zustand des Helden von VoV ist zu jedem Zeitpunkt bestimmt durch den Betrag an Goldtalern, den er bei sich trägt, die Anzahl an Erfahrungspunkten, die er bisher gesammelt hat und die Menge der Burgen und Kerker, die er bisher besucht hat.
 - (i) Definieren Sie eine Menge HELDENZUSTÄNDE, von der jedes Element einen möglichen Zustand des Helden definiert.
 - (ii) Welches Element von HELDENZUSTÄNDE beschreibt, dass der Held 120 Goldtaler bei sich trägt, 4711 Erfahrungspunkte hat und bereits die Burgen 3, 12 und 23 sowie die Kerker 7 und 23 besucht hat?
- (d) Befreit der Held von VoV eine Burg oder einen Kerker komplett von Gegnern, so bekommt er Erfahrungspunkte gutgeschrieben. Die Höhe dieser Punkte hängt davon ab, welche Burg bzw. welchen Kerker er befreit hat und aus wie vielen Orks und Trollen die Gegnergruppe bestand. Geben Sie Mengen A und B an, so dass der beschriebene Zusammenhang durch eine Funktion $Erfahrungszuwachs: A \rightarrow B$ modelliert werden kann, d.h. $Erfahrungszuwachs$ soll die zusätzlichen Erfahrungspunkte in Abhängigkeit von der Burg bzw. dem Kerker selbst und der Zusammensetzung der Gegner darin angeben.

Aufgabe 2.19. Es seien m Mengen M_1, \dots, M_m für ein $m \in \mathbb{N}_{>0}$ gegeben. Beweisen Sie die folgende Aussage:

Falls die Summe der Kardinalitäten der Mengen M_1, \dots, M_m größer als $n \in \mathbb{N}$ ist, so existiert eine Menge $M \in \{M_1, M_2, \dots, M_m\}$ deren Kardinalität größer als $\frac{n}{m}$ ist.

Aufgabe 2.20. Beweisen Sie: Falls M eine endliche Teilmenge einer unendlichen Menge U ist, so ist das Komplement von M in U unendlich.

Aufgabe 2.21. Beweisen Sie, dass für alle Mengen A, B, C mit $A = B \cup C$ gilt: Falls A unendlich ist, so ist B oder C unendlich.

Aufgabe 2.22. Beweisen Sie folgendes durch vollständige Induktion nach n .

(a)

(b) Für alle $n \in \mathbb{N}$ mit $n \geq 5$ gilt: $2^n > n(n+1)$.

(c) Für alle $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n (2i-1) = n^2$.

(d) Für alle $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$.

(e) Für alle $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $\sum_{i=1}^n (4i-1) = 2n^2 + n$.

(f) Für alle $n \in \mathbb{N}$ mit $n \geq 2$ gilt: $\prod_{i=1}^n i^i \leq n^{n(n-1)}$

(g) Für alle $x \in \mathbb{R}$ mit $x \geq -1$ und alle $n \in \mathbb{N}$ mit $n \geq 1$ gilt: $1 + n \cdot x \leq (1+x)^n$.

Aufgabe 2.23. Gegeben sei folgende rekursiv definierte Funktion:

$$\text{Für alle } n \in \mathbb{N} \text{ sei } g_s(n) := \begin{cases} s & , n = 0 \\ \frac{1}{2} \cdot g_s(n-1) & , \text{ falls } g_s(n-1) \text{ gerade und } n \geq 1 \\ 3 \cdot g_s(n-1) + 1 & , \text{ falls } g_s(n-1) \text{ ungerade und } n \geq 1 \end{cases}$$

Hierbei bezeichnet $s \in \mathbb{N}_{>0}$ den Startwert der Funktion. Berechnen Sie $g_5(5)$ und $g_{23}(15)$.³

Aufgabe 2.24. Sei $G(n)$ ein Gitter bestehend aus einer Zeile und n Spalten. Formal definieren wir $G(n)$ als Menge von Kreuzungspunkten (x, y) in $\mathbb{N} \times \mathbb{N}$ und Linien, die diese Punkte verbinden, und zwar wie folgt: Sei $\{(x, y) : x, y \in \mathbb{N}, 0 \leq x \leq n, 0 \leq y \leq 1\}$ die Menge von Kreuzungspunkten von $G(n)$. Zwischen je zwei Kreuzungspunkten k_1 und k_2 verläuft eine Linie genau dann, wenn sich k_1 und k_2 in genau einer Koordinate um genau den Betrag eins unterscheiden.

Sei $R(n)$ die Anzahl der verschiedenen Rechtecke mit nicht-leerem Flächeninhalt, die ins Gitter $G(n)$ so gezeichnet werden können, dass jedes Rechteck sich aus Linien von $G(n)$ zusammensetzt. Die folgende Abbildung zeigt alle möglichen Rechtecke, die in $G(3)$ gezeichnet werden können. Insbesondere ist $R(3) = 6$.

³Bei dieser Funktion handelt es sich um die sogenannte Collatz-Funktion für den Startwert $s \in \mathbb{N}_{>0}$. Es ist kein Startwert s bekannt, für den g_s nicht irgendwann den Wert 1 erreicht, d.h. es ist unbekannt, ob für jedes $s \in \mathbb{N}_{>0}$ ein $n_0 \in \mathbb{N}$ existiert, so dass $g_s(n_0) = 1$.



Beweisen Sie durch vollständige Induktion nach n , dass f. a. $n \in \mathbb{N}_{>0}$ gilt: $R(n) = n(n+1)/2$.

Aufgabe 2.25. Ein möglicher Algorithmus, um für eine Zahl $n \in \mathbb{N}_{>0}$ den Wert $\text{fib}(n)$ der Fibonacci-Folge zu berechnen, ist:

Algo 1 (bei Eingabe einer Zahl $n \in \mathbb{N}_{>0}$):

1. Falls $n = 1$ oder $n = 2$, dann gib 1 als Ergebnis zurück.
2. Falls $n \geq 3$, dann:
3. Sei x_1 die Ausgabe von *Algo 1* bei Eingabe der Zahl $n - 1$.
4. Sei x_2 die Ausgabe von *Algo 1* bei Eingabe der Zahl $n - 2$.
5. Gib den Wert $(x_1 + x_2)$ als Ergebnis zurück.

Der Algorithmus benötigt bei Eingabe einer Zahl n höchstens $g_1(n)$ Schritte, wobei

$$\begin{aligned} g_1(1) &= 2 \quad \text{und} \quad g_1(2) = 3 \quad \text{und} \\ g_1(n) &= 3 + g_1(n-1) + g_1(n-2) + 2 \\ &= 5 + g_1(n-1) + g_2(n-2) \quad \text{für alle } n \in \mathbb{N} \text{ mit } n \geq 3 \end{aligned}$$

(wir zählen hier jede Addition, jeden Vergleich, und jedes Zurückgeben eines Ergebnisses als einen Schritt).

Ein anderer Algorithmus, der für eine Zahl $n \in \mathbb{N}_{>0}$ den Wert $\text{fib}(n)$ berechnet, ist:

Algo 2 (bei Eingabe einer Zahl $n \in \mathbb{N}_{>0}$):

1. Falls $n = 1$ oder $n = 2$, dann gib 1 als Ergebnis zurück.
2. Seien $a_0 := 0$, $a_1 := 1$ und $a_2 := 1$.
3. Wiederhole für alle i von 3 bis n :
4. Ersetze a_0 durch a_1 und a_1 durch a_2 .
5. Ersetze a_2 durch $a_0 + a_1$.
6. Gib den Wert a_2 als Ergebnis zurück.

Dieser Algorithmus benötigt bei Eingabe $n \in \mathbb{N}_{>0}$ höchstens $g_2(n) := 6 + 5 \cdot (n - 2)$ Schritte (ähnlich wie oben zählen wir jeden Vergleich, jedes Zurückgeben eines Werts und jedes Setzen eines Werts als einen Schritt. Für jeden Schleifendurchlauf berechnen wir zusätzlich 2 Schritte, um den i um eins zu erhöhen und zu testen, ob das Ergebnis kleiner oder gleich n ist).

- (a) Welcher der beiden Algorithmen läuft im Allgemeinen schneller? D.h. welche der beiden Funktionen g_1 und g_2 liefert kleinere Funktionswerte?
- (b) Beweisen Sie, dass Ihre Antwort aus (a) korrekt ist. D.h. falls Sie in (a) geantwortet haben, dass *Algo i* im Allgemeinen schneller als *Algo j* ist, dann finden Sie eine Zahl $n_0 \in \mathbb{N}_{>0}$ und beweisen Sie per Induktion nach n , dass für alle $n \in \mathbb{N}$ mit $n \geq n_0$ gilt: $g_i(n) < g_j(n)$.

Aufgabe 2.26. Betrachten Sie die beiden folgenden Algorithmen zur Sortierung der Komponenten eines Tupels $t = (a_1, \dots, a_{2^k})$ der Länge 2^k , d.h. mit 2^k Komponenten, wobei $k \in \mathbb{N}$.

Algo 1 (bei Eingabe eines Tupels $t = (a_1, \dots, a_{2^k})$):

1. Falls $k = 0$ ist, dann gib t als Ergebnis zurück.
2. Sei t_1 die Ausgabe von *Algo 1* bei Eingabe des Tupels $(a_1, \dots, a_{2^{k-1}})$.
3. Sei t_2 die Ausgabe von *Algo 1* bei Eingabe des Tupels $(a_{2^{k-1}+1}, \dots, a_{2^k})$.
4. Gib $merge(t_1, t_2)$ zurück.

Hierbei ist $merge()$ eine Funktion, die bei Eingabe zweier sortierter Tupel t_1 und t_2 mit jeweils 2^{k-1} Komponenten ein sortiertes Tupel mit 2^k Komponenten zurückgibt. Man kann sich leicht überlegen, dass die Funktion $merge()$ so implementiert werden kann, dass sie nicht mehr als $3 \cdot 2^k$ Schritte benötigt. Deshalb braucht *Algo 1* insgesamt für ein Tupel der Länge 2^k nicht mehr als $f_1(k) = 2f_1(k-1) + 3 \cdot 2^k + 4$ Schritte, wobei $f_1(0) = 2$.

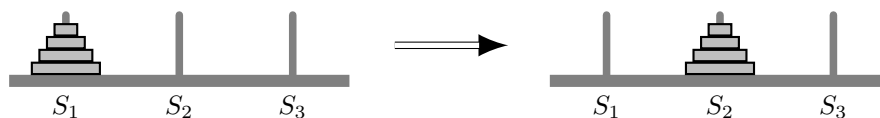
Algo 2 (bei Eingabe eines Tupels $t = (a_1, \dots, a_{2^k})$):

1. Wiederhole für jedes i von 1 bis $(2^k - 1)$:
 2. Wiederhole für jedes j von 1 bis $(2^k - i)$:
 3. Falls $a_j > a_{j+1}$, dann tausche die Komponenten a_j und a_{j+1} in t .
4. Gib t zurück.

Insgesamt braucht *Algo 2* für ein Tupel der Länge 2^k höchstens $f_2(k) = 3 \cdot 2^{2k} + 2^k - 1$ Schritte.

- (a) Welcher der beiden Algorithmen läuft im Allgemeinen schneller? D.h. welche der beiden Funktionen f_1 und f_2 liefert kleinere Funktionswerte?
- (b) Beweisen Sie, dass Ihre Antwort aus (a) korrekt ist. D.h. falls Sie in (a) geantwortet haben, dass *Algo i* im Allgemeinen schneller als *Algo j* ist, dann finden Sie eine Zahl $n_0 \in \mathbb{N}$ und beweisen Sie per Induktion nach n , dass für alle $n \in \mathbb{N}$ mit $n \geq n_0$ gilt: $f_i(n) < f_j(n)$.

Aufgabe 2.27 (Türme von Hanoi). Ein Turm aus $n \in \mathbb{N}_{>0}$ unterschiedlich großen gelochten Scheiben soll von einem Stab (S_1) auf einen zweiten Stab (S_2) unter Zuhilfenahme eines Hilfsstabes (S_3) verschoben werden (das folgende Bild zeigt die Situation für den Fall $n = 4$).



Dabei müssen die folgenden Regeln beachtet werden:

- Pro Zug darf nur eine Scheibe bewegt werden. Es kann also immer nur die oberste Scheibe eines Turmes bewegt werden.
 - Es darf nie eine größere Scheibe auf einer kleineren Scheibe liegen.
- (a) Beschreiben Sie, wie der Turm im Fall $n = 4$ von S_1 nach S_2 verschoben werden kann.
 - (b) Beweisen Sie, dass es für alle $n \in \mathbb{N}_{>0}$ möglich ist, die n Scheiben von S_1 nach S_2 zu verschieben.

Hinweis: Beweisen Sie zuerst durch vollständige Induktion nach n , dass die folgende Aussage für alle $n \in \mathbb{N}$ mit $n \geq 1$ gilt:

A(n): Seien $i, j \in \{1, 2, 3\}$ mit $i \neq j$, sei $m \in \mathbb{N}$ mit $m \geq n$, und seien m Scheiben so auf die drei Stäbe verteilt, dass gilt:

- Auf S_i liegen mindestens n Scheiben.
- Die Scheiben auf den beiden anderen Stäben sind größer als die obersten n Scheiben auf S_i .

Dann lassen sich die obersten n Scheiben von S_i so nach S_j verschieben, dass keine der anderen Scheiben bewegt wird.

Aufgabe 2.28. Sei die Sprache L über dem Alphabet $A := \{(\,,\,)\}$ wie folgt rekursiv definiert:

Basisregel: (B) $\varepsilon \in L$

Rekursive Regeln: (R1) Ist $w \in L$, so ist auch $(w) \in L$.

(R2) Sind $w_1, w_2 \in L$, so ist auch $w_1 w_2 \in L$.

(a) Welche der folgenden Wörter gehören zu L und welche nicht?

- $()$
- $()()$
- $(($
- $(())$
- $()()$
- $((())()$

(b) Beweisen Sie, dass $((()(())) \in L$ ist.

(c) Für jedes Symbol $s \in A$ und jedes Wort $w \in A^*$ bezeichne $|w|_s$ die Anzahl der Vorkommen des Symbols s in w . Beweisen Sie durch Induktion, dass für alle Wörter $w \in L$ gilt: $|w|_{(} = |w|_{)}$.

(d) Beweisen Sie, dass $()(() \notin L$ ist.

Aufgabe 2.29. Die Menge UPNZ (Umgekehrte Polnische Notation auf Ziffern) sei die rekursiv wie folgt definierte Teilmenge von A^* für $A := \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, *\}$:

Basisregel: (B) Jede Ziffer, also jedes Zeichen aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ ist in UPNZ

Rekursive Regeln: (R1) Sind w_1 und w_2 in UPNZ, so ist auch $w_1 w_2 +$ in UPNZ

(R2) Sind w_1 und w_2 in UPNZ, so ist auch $w_1 w_2 *$ in UPNZ

Somit gilt beispielsweise $7 \in \text{UPNZ}$ und $12 + \in \text{UPNZ}$, während $42 \notin \text{UPNZ}$.

Sei z eine Ziffer, also ein Zeichen aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ und seien w_1, w_2 Worte aus UPNZ. Die Funktionen $f : \text{UPNZ} \rightarrow \mathbb{N}$ und $g : \text{UPNZ} \rightarrow \{0, 1\}$ sind rekursiv entsprechend der Definition von UPNZ wie folgt definiert:

$$\begin{aligned}
 f(z) &= z & g(z) &= \begin{cases} 0, & \text{falls } z \text{ gerade ist} \\ 1, & \text{sonst} \end{cases} \\
 f(w_1 w_2 +) &= f(w_1) + f(w_2) & g(w_1 w_2 +) &= \begin{cases} 0, & \text{falls } g(w_1) = g(w_2) \\ 1, & \text{sonst} \end{cases} \\
 f(w_1 w_2 *) &= f(w_1) \cdot f(w_2) & g(w_1 w_2 *) &= \begin{cases} 0, & \text{falls } g(w_1) = 0 \text{ oder } g(w_2) = 0 \\ 1, & \text{sonst} \end{cases}
 \end{aligned}$$

Nach diesen Definitionen ist zum Beispiel $f(32 + 47 + *) = 55$ und $g(32 + 47 + *) = 1$.

(a) Welche der folgenden Wörter w_1, w_2, w_3, w_4 gehören zur Menge UPNZ, welche nicht? Berechnen Sie $f(w_i)$ und $g(w_i)$ falls $w_i \in \text{UPNZ}$ für $i \in \{1, 2, 3, 4\}$.

(i) $w_1 := 84 +$

(iii) $w_3 := 8 + 2 * 1$

(ii) $w_2 := 32 + 12 -$

(iv) $w_4 := 19 + 2 * 14 + + 0 +$

(b) Beweisen Sie den folgenden Zusammenhang durch vollständige Induktion:

Für jedes Wort $w \in \text{UPNZ}$ gilt: $f(w)$ ist gerade $\iff g(w) = 0$.

Aufgabe 2.30. Im Folgenden wird die Syntax einer sehr einfachen Programmiersprache definiert, der so genannten WHILE-Programme. Die Menge L , die hier definiert wird, ist die Menge aller Zeichenketten über dem Alphabet A , die syntaktisch korrekte WHILE-Programme sind. Hierbei ist $A := \{\mathbf{x}, :=, +, -, \neq, ;, \mathbf{while}, \mathbf{do}, \mathbf{end}\} \cup \mathbb{N}$, und L ist die folgendermaßen rekursiv definierte Menge:

Basisregeln: (B1) Für Zahlen $i, j, c \in \mathbb{N}$ gilt: $\mathbf{x}i := \mathbf{x}j + c \in L$.

(B2) Für Zahlen $i, j, c \in \mathbb{N}$ gilt: $\mathbf{x}i := \mathbf{x}j - c \in L$.

Rekursive Regeln: (R1) Sind $w_1 \in L$ und $w_2 \in L$, so ist auch $w_1; w_2 \in L$.

(R2) Ist $w \in L$ und $i \in \mathbb{N}$, so ist $\mathbf{while} \mathbf{x}i \neq 0 \mathbf{do} w \mathbf{end} \in L$.

(a) Welche der folgenden Wörter aus A^* gehören zu L und welche nicht? Begründen Sie jeweils Ihre Antwort.

(i) $\mathbf{x}3 := \mathbf{x}7 - 2$

(ii) $\mathbf{x}3 := 1; \mathbf{x}2 := \mathbf{x}3 + 5$

(iii) $\mathbf{while} \mathbf{x}1 \neq 0 \mathbf{do} \mathbf{x}0 := \mathbf{x}0 + 1; \mathbf{x}1 := \mathbf{x}1 - 1 \mathbf{end}$

(iv) $\mathbf{x}1 := \mathbf{x}1 + 42; \mathbf{while} \mathbf{x}1 \neq 0 \mathbf{do} \mathbf{x}1 := \mathbf{x}1 - 1$

(b) Für jedes Wort $w \in A^*$ und jedes Symbol $s \in A$ bezeichne $|w|_s$ die Anzahl der Vorkommen des Symbols s in w . Beweisen Sie durch Induktion, dass für alle Wörter $w \in L$ gilt: $|w|_{\mathbf{do}} = |w|_{\mathbf{end}}$.

Aufgabe 2.31. Die Menge PBA der *positiven Booleschen Ausdrücke* ist die Menge der Wörter über dem Alphabet $A = \{\mathbf{0}, \mathbf{1}, \wedge, \vee, (,)\}$, die rekursiv wie folgt definiert ist:

Basisregel: (B) Die Symbole $\mathbf{0}$ und $\mathbf{1}$ sind in PBA.

Rekursive Regeln: (R1) Sind w_1 und w_2 in PBA, so ist auch $(w_1 \wedge w_2)$ in PBA.

(R2) Sind w_1 und w_2 in PBA, so ist auch $(w_1 \vee w_2)$ in PBA.

1. Welche der folgenden Wörter gehören zur Sprache PBA, welche nicht? Sie brauchen Ihre Antworten nicht zu begründen.

a) $(\mathbf{1} \wedge \mathbf{0})$

b) $((\mathbf{1} \wedge \mathbf{V}_0) \vee \mathbf{0})$

c) $\mathbf{0} \vee ((\mathbf{1} \wedge \mathbf{0}) \wedge \mathbf{0})$

2. Offensichtlich ist jedes Wort der Sprache PBA eine aussagenlogische Formel, es gilt also $\text{PBA} \subseteq \text{AL}$. Sei $f : \text{PBA} \rightarrow \{0, 1\}$ eine Funktion, die jedem Wort aus PBA einen Wahrheitswert zuweist. Für jedes $\varphi \in \text{PBA}$ sei $f(\varphi) := \llbracket \varphi \rrbracket^{\mathcal{B}}$, wobei $\mathcal{B}(V_i) = 0$ für alle $i \in \mathbb{N}$.

a) Berechnen Sie den jeweiligen Wert der folgenden Ausdrücke. Sie brauchen Ihre Antworten nicht zu begründen.

$$\text{(I)} \ f((\mathbf{1} \wedge \mathbf{0})) \quad \text{(II)} \ f(((\mathbf{1} \wedge \mathbf{1}) \vee \mathbf{1})) \quad \text{(III)} \ f((((\mathbf{0} \vee \mathbf{0}) \vee (\mathbf{1} \vee \mathbf{0})) \wedge (\mathbf{0} \vee \mathbf{1})))$$

- b) Für jedes $\varphi \in \text{PBA}$ bezeichne $|\varphi|_{\mathbf{0}}$ die Anzahl der Vorkommen des Symbols $\mathbf{0}$ in φ .
Zeigen Sie durch Induktion über den Aufbau von PBA, dass für alle Wörter $\varphi \in \text{PBA}$ gilt:

$$|\varphi|_{\mathbf{0}} = 0 \quad \implies \quad f(\varphi) = 1$$

3 Aussagenlogik

3.1 Wozu “Logik” im Informatik-Studium?

Logik

Logik (nach dem Altgriechischen “Logos”: “Vernunft”) ist “die Lehre des vernünftigen Schlussfolgerns”. Logik ist ein Teilgebiet in den Disziplinen Philosophie, Mathematik, Informatik und Linguistik. Eine zentrale Frage, mit dem sich das Gebiet der Logik beschäftigt ist:

Wie kann man Aussagen miteinander verknüpfen, und auf welche Weise kann man formal Schlüsse ziehen und Beweise durchführen?

In einem gewissen Sinn spielt die Logik in der Informatik eine ähnlich wichtige Rolle wie die Differential- und Integralrechnung in der Physik [20, 10]. Logik wird in der Informatik u.a. genutzt

- zur Repräsentation von statischem Wissen (z.B. im Bereich der künstlichen Intelligenz),
- als Grundlage für Datenbank-Anfragesprachen,
- als Bestandteil von Programmiersprachen (z.B. um “Bedingungen” in “IF-Anweisungen” zu formulieren),
- zur automatischen Generierung von Beweisen (so genannte “Theorembeweiser”),
- zur Verifikation von
 - Schaltkreisen (*Ziel*: beweise, dass ein Schaltkreis bzw. Chip “richtig” funktioniert),
 - Programmen (*Ziel*: beweise, dass ein Programm gewisse wünschenswerte Eigenschaften hat),
 - Protokollen (*Ziel*: beweise, dass die Kommunikation zwischen zwei “Agenten”, die nach einem gewissen “Protokoll” abläuft, “sicher” ist — etwa gegen Abhören oder Manipulation durch dritte; Anwendungsbeispiel: Internet-Banking).

Aussagenlogik

Aussagen
Junktoren
Aussagenlogik

Aussagen im Sinne der Aussagenlogik sind sprachliche Gebilde, die entweder **wahr** oder **falsch** sind. Aussagen können mit **Junktoren** wie “nicht”, “und”, “oder”, “wenn ... dann” etc. zu komplexeren Aussagen verknüpft werden. Die **Aussagenlogik** beschäftigt sich mit allgemeinen Prinzipien des korrekten Argumentierens und Schließens mit Aussagen und Kombinationen von Aussagen.

Beispiel 3.1 (“Geburtstagsfeier”).

Fred möchte mit möglichst vielen seiner Freunde Anne, Bernd, Christine, Dirk und Eva seinen

Geburtstag feiern. Er weiß, dass Eva nur dann kommt, wenn Christine und Dirk kommen. Andererseits kommt Christine nur dann, wenn auch Anne kommt; und Dirk wird auf keinen Fall kommen, wenn Bernd und Eva beide zur Feier kommen. Anne wiederum wird nur dann kommen, wenn auch Bernd oder Christine dabei sind. Wenn allerdings Bernd und Anne beide zur Party kommen, dann wird Eva auf keinen Fall dabei sein.

Frage: Wie viele Freunde (und welche) werden im besten Fall zur Party kommen?

Das Wissen, das im obigen Text wiedergegeben ist, lässt sich in “atomare Aussagen” zerlegen, die mit Junktoren verknüpft werden können. Die “atomaren Aussagen”, um die sich der Text dreht, kürzen wir folgendermaßen ab:

- $A \hat{=}$ Anne kommt zur Feier
- $B \hat{=}$ Bernd kommt zur Feier
- $C \hat{=}$ Christine kommt zur Feier
- $D \hat{=}$ Dirk kommt zur Feier
- $E \hat{=}$ Eva kommt zur Feier

Das im Text zusammengefasste “Wissen” lässt sich wie folgt repräsentieren:

(Wenn E , dann (C und D))	Eva kommt nur dann, wenn Christine und Dirk kommen,
und (wenn C , dann A)	Christine kommt nur dann, wenn auch Anne kommt,
und (wenn (B und E), dann nicht D)	Dirk wird auf keinen Fall kommen, wenn Bernd und Eva beide kommen,
und (wenn A , dann (B oder C))	Anne kommt nur dann, wenn auch Bernd oder Christine dabei sind,
und (wenn (B und A), dann nicht E)	wenn Bernd und Anne beide kommen, dann wird Eva auf keinen Fall dabei sein.

Die Aussagenlogik liefert einen Formalismus, mit dessen Hilfe man solches “Wissen” modellieren und Schlüsse daraus ziehen kann — insbesondere z.B. um die Frage, mit wie vielen (und welchen) Gästen Fred bei seiner Feier rechnen kann, zu beantworten. □Ende von Beispiel 3.1

3.2 Syntax und Semantik der Aussagenlogik

Die **Syntax** legt fest, welche Zeichenketten (Worte) Formeln der Aussagenlogik sind. Die **Semantik** legt fest, welche “Bedeutung” einzelne Formeln haben.

Syntax
Semantik

Man beachte, dass dies analog zur “Syntax” und “Semantik” von JAVA-Programmen ist: Die Syntax legt fest, welche Zeichenketten JAVA-Programme sind, während die Semantik bestimmt, was das Programm tut.

Definition 3.2 (Aussagenvariablen und Alphabet der Aussagenlogik).

(a) Eine **Aussagenvariable** (kurz: Variable) hat die Form V_i , für $i \in \mathbb{N}$. Die Menge aller Aussagenvariablen bezeichnen wir mit AVAR. D.h.:

Aussagenvariable

$$\text{AVAR} = \{V_i : i \in \mathbb{N}\} = \{V_0, V_1, V_2, V_3, \dots\}.$$

(b) Das **Alphabet** der Aussagenlogik ist

$$A_{AL} := AVAR \cup \{\mathbf{0}, \mathbf{1}, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,)\}.$$

Definition 3.3 (aussagenlogische Formeln: Syntax).

Die Menge AL der aussagenlogischen Formeln (kurz: Formeln) ist die folgendermaßen rekursiv definierte Teilmenge von A_{AL}^* :

Basisregeln:

- (B0) $\mathbf{0} \in AL$.
- (B1) $\mathbf{1} \in AL$.
- (BV) Für jede Variable $X \in AVAR$ gilt: $X \in AL$.

Rekursive Regeln:

- (R1) Ist $\varphi \in AL$, so ist auch $\neg\varphi \in AL$.
- (R2) Ist $\varphi \in AL$ und $\psi \in AL$, so ist auch
 - $(\varphi \wedge \psi) \in AL$
 - $(\varphi \vee \psi) \in AL$
 - $(\varphi \rightarrow \psi) \in AL$
 - $(\varphi \leftrightarrow \psi) \in AL$.

Anmerkung 3.4 (griechische Buchstaben).

In der Literatur werden Formeln einer Logik traditionell meistens mit griechischen Buchstaben bezeichnet. Hier eine Liste der gebräuchlichsten Buchstaben:

Buchstabe	φ	ψ	χ	θ bzw. ϑ	λ	μ	ν	τ	κ
Aussprache	phi	psi	chi	theta	lambda	mü	nü	tau	kappa
Buchstabe	σ	ρ	ξ	ζ	α	β	γ	δ	ω
Aussprache	sigma	rho	xi	zeta	alpha	beta	gamma	delta	omega
Buchstabe	ε	ι	π	Δ	Γ	Σ	Π	Φ	
Aussprache	epsilon	iota	pi	Delta	Gamma	Sigma	Pi	Phi	

Beispiel 3.5.

Die folgenden Zeichenketten sind Formeln, d.h. gehören zur Menge AL:

- $(\neg V_0 \vee (V_5 \rightarrow V_1))$
- $\neg((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3)$

Die folgenden Zeichenketten sind keine Formeln, d.h. gehören nicht zur Menge AL:

- $V_1 \vee V_2 \wedge V_3$ (da die Klammern fehlen),

- $(\neg V_1)$ (da die Klammern “zu viel” sind).

□ Ende von Beispiel 3.5

Notation 3.6.

- | | |
|---|------------------------|
| (a) $\mathbf{0}, \mathbf{1}$ und die Variablen (d.h. die Elemente aus AVAR) bezeichnen wir als atomare Formeln bzw. Atome . | atomare Formel
Atom |
| (b) Die Symbole $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ heißen Junktoren . | Junktor |
| (c) Sind φ und ψ Formeln (d.h. $\varphi \in \text{AL}$ und $\psi \in \text{AL}$), so heißt: | |
| • $(\varphi \wedge \psi)$ Konjunktion (bzw. <i>Verundung</i>) von φ und ψ , | Konjunktion |
| • $(\varphi \vee \psi)$ Disjunktion (bzw. <i>Veroderung</i>) von φ und ψ , | Disjunktion |
| • $\neg\varphi$ Negation (bzw. <i>Verneinung</i>) von φ . | Negation |

Wir wissen nun, welche Zeichenketten (über dem Alphabet A_{AL}) **Formeln** genannt werden. Um festlegen zu können, welche Bedeutung (d.h. Semantik) solche Formeln haben, brauchen wir folgende Definition:

Definition 3.7.

Die **Variablenmenge** einer aussagenlogischen Formel φ (kurz: $\text{Var}(\varphi)$) ist die Menge aller Variablen $X \in \text{AVAR}$, die in φ vorkommen.

Beispiele:

- $\text{Var}\left(\left(\neg V_0 \vee (V_5 \rightarrow V_1)\right)\right) = \{V_0, V_1, V_5\}$,
- $\text{Var}\left(\neg\left((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3\right)\right) = \{V_0, V_3\}$,
- $\text{Var}\left((\mathbf{0} \vee \mathbf{1})\right) = \emptyset$.

Definition 3.8.

- | | |
|---|-------------------------------|
| (a) Eine Belegung (bzw. Wahrheitsbelegung) ist eine partielle Funktion von AVAR nach $\{0, 1\}$. ¹ | Belegung
Wahrheitsbelegung |
| (b) Eine Belegung \mathcal{B} ist eine Belegung für die Formel φ (bzw. passend zu φ), wenn | passend zu φ |

$$\text{Def}(\mathcal{B}) \supseteq \text{Var}(\varphi).$$

Definition 3.9 (Semantik der Aussagenlogik).

Rekursiv über den Aufbau von AL definieren wir eine Funktion $\llbracket \cdot \rrbracket$, die jeder Formel $\varphi \in \text{AL}$ und jeder zu φ passenden Belegung \mathcal{B} einen **Wahrheitswert** (kurz: **Wert**) $\llbracket \varphi \rrbracket^{\mathcal{B}} \in \{0, 1\}$ zuordnet:

¹Die intuitive Bedeutung dabei ist, dass 1 für den Wert “wahr” und 0 für den Wert “falsch” steht.

Rekursionsanfang:

- $\llbracket \mathbf{0} \rrbracket^{\mathcal{B}} := 0$.
- $\llbracket \mathbf{1} \rrbracket^{\mathcal{B}} := 1$.
- F.a. $X \in \text{AVAR}$ gilt: $\llbracket X \rrbracket^{\mathcal{B}} := \mathcal{B}(X)$.

Rekursionsschritt:

- Ist $\varphi \in \text{AL}$, so ist $\llbracket \neg\varphi \rrbracket^{\mathcal{B}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 0 \\ 0, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 1. \end{cases}$
- Ist $\varphi \in \text{AL}$ und $\psi \in \text{AL}$, so ist
 - $\llbracket (\varphi \wedge \psi) \rrbracket^{\mathcal{B}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 1 \text{ und } \llbracket \psi \rrbracket^{\mathcal{B}} = 1 \\ 0, & \text{sonst} \end{cases}$
 - $\llbracket (\varphi \vee \psi) \rrbracket^{\mathcal{B}} := \begin{cases} 0, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 0 \text{ und } \llbracket \psi \rrbracket^{\mathcal{B}} = 0 \\ 1, & \text{sonst} \end{cases}$
 - $\llbracket (\varphi \rightarrow \psi) \rrbracket^{\mathcal{B}} := \begin{cases} 0, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 1 \text{ und } \llbracket \psi \rrbracket^{\mathcal{B}} = 0 \\ 1, & \text{sonst} \end{cases}$
 - $\llbracket (\varphi \leftrightarrow \psi) \rrbracket^{\mathcal{B}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = \llbracket \psi \rrbracket^{\mathcal{B}} \\ 0, & \text{sonst.} \end{cases}$

Die **intuitive Bedeutung der Semantik** lässt sich wie folgt beschreiben:

- **Atome:** $\mathbf{1}$ und $\mathbf{0}$ bedeuten einfach “wahr” und “falsch”.

Die Variablen $X \in \text{AVAR}$ stehen für irgendwelche Aussagen. Uns interessiert hier nur, ob diese Aussagen “wahr” oder “falsch” sind — und dies wird durch eine Belegung \mathcal{B} angegeben.

- **Negation:** $\neg\varphi$ bedeutet “nicht φ ”.

D.h.: $\neg\varphi$ ist wahr (unter Belegung \mathcal{B}) $\iff \varphi$ ist falsch (unter Belegung \mathcal{B}). Durch eine so genannte **Verknüpfungstafel** (bzw. **Wahrheitstafel**) lässt sich dies wie folgt darstellen:

$\llbracket \varphi \rrbracket^{\mathcal{B}}$	$\llbracket \neg\varphi \rrbracket^{\mathcal{B}}$
0	1
1	0

- **Konjunktion:** $(\varphi \wedge \psi)$ bedeutet “ φ und ψ ”.

D.h.: $(\varphi \wedge \psi)$ ist wahr (unter Belegung \mathcal{B}) $\iff \varphi$ ist wahr und ψ ist wahr (unter Belegung \mathcal{B}). Zugehörige Verknüpfungstafel:

$\llbracket \varphi \rrbracket^{\mathcal{B}}$	$\llbracket \psi \rrbracket^{\mathcal{B}}$	$\llbracket (\varphi \wedge \psi) \rrbracket^{\mathcal{B}}$
0	0	0
0	1	0
1	0	0
1	1	1

- **Disjunktion:** $(\varphi \vee \psi)$ bedeutet “ φ oder ψ ”.

D.h.: $(\varphi \vee \psi)$ ist wahr (unter Belegung \mathcal{B}) $\iff \varphi$ ist wahr oder ψ ist wahr (unter Belegung \mathcal{B}). Zugehörige Verknüpfungstafel:

$\llbracket \varphi \rrbracket^{\mathcal{B}}$	$\llbracket \psi \rrbracket^{\mathcal{B}}$	$\llbracket (\varphi \vee \psi) \rrbracket^{\mathcal{B}}$
0	0	0
0	1	1
1	0	1
1	1	1

- **Implikation:** $(\varphi \rightarrow \psi)$ bedeutet “ φ impliziert ψ ”, d.h. “wenn φ , dann auch ψ ”.

D.h.: $(\varphi \rightarrow \psi)$ ist wahr (unter Belegung \mathcal{B}) \iff wenn φ wahr ist, dann ist auch ψ wahr (unter Belegung von \mathcal{B}). Zugehörige Verknüpfungstafel:

$\llbracket \varphi \rrbracket^{\mathcal{B}}$	$\llbracket \psi \rrbracket^{\mathcal{B}}$	$\llbracket (\varphi \rightarrow \psi) \rrbracket^{\mathcal{B}}$
0	0	1
0	1	1
1	0	0
1	1	1

- **Biimplikation:** $(\varphi \leftrightarrow \psi)$ bedeutet “ φ genau dann, wenn ψ ”.

D.h.: $(\varphi \leftrightarrow \psi)$ ist wahr (unter Belegung \mathcal{B}) $\iff \varphi$ ist genau dann wahr, wenn ψ wahr ist (unter Belegung von \mathcal{B}). Zugehörige Verknüpfungstafel:

$\llbracket \varphi \rrbracket^{\mathcal{B}}$	$\llbracket \psi \rrbracket^{\mathcal{B}}$	$\llbracket (\varphi \leftrightarrow \psi) \rrbracket^{\mathcal{B}}$
0	0	1
0	1	0
1	0	0
1	1	1

Beispiel 3.10.

Betrachte die Formel

$$\varphi := (\neg V_0 \vee (V_5 \rightarrow V_1)).$$

Dann ist beispielsweise die Funktion $\mathcal{B}: \{V_0, V_1, V_5\} \rightarrow \{0, 1\}$ mit $\mathcal{B}(V_0) := 1$, $\mathcal{B}(V_1) := 1$ und $\mathcal{B}(V_5) := 0$ eine Belegung für φ . Der Wahrheitswert von φ unter Belegung \mathcal{B} ist der Wert

$$\begin{aligned} \llbracket \varphi \rrbracket^{\mathcal{B}} &\stackrel{\text{Def. 3.9}}{=} \begin{cases} 1, & \text{falls } \llbracket \neg V_0 \rrbracket^{\mathcal{B}} = 1 \text{ oder } \llbracket (V_5 \rightarrow V_1) \rrbracket^{\mathcal{B}} = 1 \\ 0, & \text{sonst} \end{cases} \\ &\stackrel{\text{Def. 3.9}}{=} \begin{cases} 1, & \text{falls } \llbracket V_0 \rrbracket^{\mathcal{B}} = 0 \text{ oder } (\llbracket V_5 \rrbracket^{\mathcal{B}} = 0 \text{ oder } \llbracket V_1 \rrbracket^{\mathcal{B}} = 1) \\ 0, & \text{sonst} \end{cases} \\ &\stackrel{\text{Def. 3.9}}{=} \begin{cases} 1, & \text{falls } \mathcal{B}(V_0) = 0 \text{ oder } \mathcal{B}(V_5) = 0 \text{ oder } \mathcal{B}(V_1) = 1 \\ 0, & \text{sonst} \end{cases} \\ &= 1 \quad (\text{denn gemäß obiger Wahl von } \mathcal{B} \text{ gilt } \mathcal{B}(V_5) = 0). \end{aligned}$$

Beobachtung 3.11.

Sind \mathcal{B} und \mathcal{B}' zwei Belegungen für eine Formel φ , die auf $\text{Var}(\varphi)$ übereinstimmen (d.h.: f.a. $X \in \text{Var}(\varphi)$ ist $\mathcal{B}(X) = \mathcal{B}'(X)$), so ist $\llbracket \varphi \rrbracket^{\mathcal{B}} = \llbracket \varphi \rrbracket^{\mathcal{B}'}$.

Koinzidenzlemma

In der Literatur wird diese Beobachtung oft unter dem Namen **Koinzidenzlemma** geführt. Intuitiv ist die Beobachtung “offensichtlich richtig”, denn in der Definition von $\llbracket \varphi \rrbracket^{\mathcal{B}}$ werden ja nur diejenigen Variablen verwendet, die in φ vorkommen (also zu $\text{Var}(\varphi)$ gehören). Einen formalen Beweis der Beobachtung kann man leicht per Induktion über den Aufbau von AL führen. Aufgrund der Beobachtung des Koinzidenzlemmas werden wir im Folgenden, wenn wir Belegungen \mathcal{B} für eine Formel φ betrachten, uns meistens nur für diejenigen Werte $\mathcal{B}(X)$ interessieren, für die $X \in \text{Var}(\varphi)$ ist.

Um umgangssprachlich formuliertes Wissen (vgl. Beispiel 3.1 “Geburtstagsfeier”) durch aussagenlogische Formeln zu repräsentieren, sind folgende Konventionen bequem:

Notation 3.12.

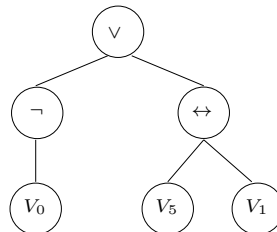
- Statt V_0, V_1, V_2, \dots bezeichnen wir Variablen oft auch mit $A, B, C, \dots, X, Y, Z, \dots$ oder mit Variablen wie X', Y_1, \dots
- Wir schreiben $\bigwedge_{i=1}^n \varphi_i$ bzw. $\varphi_1 \wedge \dots \wedge \varphi_n$ an Stelle von $((\varphi_1 \wedge \varphi_2) \wedge \varphi_3) \wedge \dots \wedge \varphi_n$ (analog für “ \vee ” an Stelle von “ \wedge ”).
- Die äußeren Klammern einer Formel lassen wir manchmal weg und schreiben z.B. $(A \wedge B) \rightarrow C$ an Stelle des (formal korrekten) $((A \wedge B) \rightarrow C)$.
- Ist φ eine Formel und \mathcal{B} eine Belegung für φ , so sagen wir “ \mathcal{B} erfüllt φ ” (bzw. “ \mathcal{B} ist eine erfüllende Belegung für φ ”), falls $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$.

Bemerkung 3.13 (Syntaxbäume zur graphischen Darstellung von Formeln).

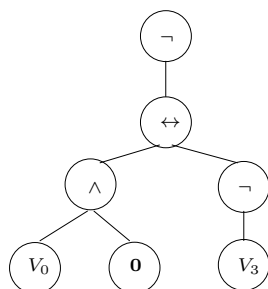
Die Struktur einer Formel lässt sich bequem durch einen **Syntaxbaum** (englisch: **parse tree**) darstellen.

Beispiele:

- Syntaxbaum der Formel $(\neg V_0 \vee (V_5 \leftrightarrow V_1))$:



- Syntaxbaum der Formel $\neg((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3)$:



Computerlesbare Darstellung von Formeln:

Definition 3.14 (ASCII-Syntax für die Aussagenlogik).

- (a) Wir betrachten das folgende Alphabet:

$\text{ASCII} :=$ Menge aller ASCII-Symbole.

- (b) Die Menge $\text{AVAR}_{\text{ASCII}}$ aller ASCII-Repräsentationen von Aussagenvariablen ist wie folgt definiert:

$$\text{AVAR}_{\text{ASCII}} := \{w \in \text{ASCII}^+ : \text{das erste Symbol in } w \text{ ist ein Buchstabe,} \\ \text{alle weiteren Symbole in } w \text{ sind Buchstaben} \\ \text{oder Ziffern}\}.$$

- (c) Die Menge AL_{ASCII} aller ASCII-Repräsentationen von aussagenlogischen Formeln ist die rekursiv wie folgt definierte Teilmenge von ASCII^* :

Basisregeln:

- $0 \in \text{AL}_{\text{ASCII}}$.
- $1 \in \text{AL}_{\text{ASCII}}$.
- Für alle $w \in \text{AVAR}_{\text{ASCII}}$ gilt: $w \in \text{AL}_{\text{ASCII}}$.

Rekursive Regeln:

- Ist $\varphi \in \text{AL}_{\text{ASCII}}$, so ist auch $\sim\varphi \in \text{AL}_{\text{ASCII}}$.
- Ist $\varphi \in \text{AL}_{\text{ASCII}}$ und $\psi \in \text{AL}_{\text{ASCII}}$, so ist auch
 - $(\varphi \wedge \psi) \in \text{AL}_{\text{ASCII}}$
 - $(\varphi \vee \psi) \in \text{AL}_{\text{ASCII}}$
 - $(\varphi \rightarrow \psi) \in \text{AL}_{\text{ASCII}}$
 - $(\varphi \leftrightarrow \psi) \in \text{AL}_{\text{ASCII}}$.

Bemerkung 3.15. Es ist offensichtlich, wie man Formeln aus AL in ihre entsprechende ASCII-Repräsentation übersetzt und umgekehrt. Zum Beispiel ist

$$((V_0 \wedge \mathbf{0}) \rightarrow \neg V_{13})$$

eine Formel in AL, deren ASCII-Repräsentation die folgende Zeichenkette aus AL_{ASCII} ist:

$$((V0/\wedge 0) -> \sim V13).$$

Wir werden meistens mit der “abstrakten Syntax”, d.h. mit der in Definition 3.3 festgelegten Menge AL, arbeiten. Um aber Formeln in Computer-Programme einzugeben, können wir die ASCII-Repräsentation verwenden.

Umgangssprachliche Aussagen lassen sich wie folgt durch aussagenlogische Formeln repräsentieren:

Beispiel 3.16.

Die Zeugenaussage

“Das Fluchtauto war rot oder grün und hatte weder vorne noch hinten ein Nummernschild.”

lässt sich durch die aussagenlogische Formel

$$((X_R \vee X_G) \wedge (\neg X_V \wedge \neg X_H))$$

repräsentieren, die die folgenden atomaren Aussagen nutzt:

- X_R : das Fluchtauto war rot,
- X_G : das Fluchtauto war grün,
- X_V : das Fluchtauto hatte vorne ein Nummernschild,
- X_H : das Fluchtauto hatte hinten ein Nummernschild.

Beispiel 3.17.

Das in Beispiel 3.1 (“Geburtstagsfeier”) aufgelistete Wissen kann folgendermaßen repräsentiert werden.

Atomare Aussagen:

- A : Anne kommt zur Feier,
- B : Bernd kommt zur Feier,
- C : Christine kommt zur Feier,
- D : Dirk kommt zur Feier,
- E : Eva kommt zur Feier.

Die Aussage des gesamten Textes aus Beispiel 3.1 wird durch folgende Formel repräsentiert:

$$\varphi := (E \rightarrow (C \wedge D)) \wedge (C \rightarrow A) \wedge ((B \wedge E) \rightarrow \neg D) \wedge (A \rightarrow (B \vee C)) \wedge ((B \wedge A) \rightarrow \neg E).$$

Die Frage

“Wie viele (und welche) Freunde werden im besten Fall zur Party kommen?”

kann dann durch Lösen der folgenden Aufgabe beantwortet werden: Finde eine Belegung \mathcal{B} für φ , so dass

- φ von \mathcal{B} erfüllt wird, d.h. $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$, und
- $|\{X \in \{A, B, C, D, E\} : \mathcal{B}(X) = 1\}|$ so groß wie möglich ist.

Um Aufgaben solcher Art lösen zu können, brauchen wir also eine Methode zum Finden der erfüllenden Belegungen für eine Formel. Eine Möglichkeit dafür ist, so genannte Wahrheitstafeln zu benutzen.

Wahrheitstafeln:

Für jede Formel φ kann man die Wahrheitswerte von φ unter allen möglichen Belegungen in einer Wahrheitstafel darstellen. Für jede Belegung $\mathcal{B} : \text{Var}(\varphi) \rightarrow \{0, 1\}$ hat die Wahrheitstafel eine Zeile, die die Werte $\mathcal{B}(X)$ f.a. $X \in \text{Var}(\varphi)$ und den Wert $\llbracket \varphi \rrbracket^{\mathcal{B}}$ enthält. Um die Wahrheitstafel für φ auszufüllen, ist es bequem, auch Spalten für (alle oder einige) “Teilformeln” von φ einzufügen.

Beispiel 3.18. (a) Wahrheitstafel für $\varphi := (\neg V_0 \vee (V_5 \rightarrow V_1))$:

V_0	V_1	V_5	$\neg V_0$	$(V_5 \rightarrow V_1)$	φ
0	0	0	1	1	1
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	0	0	0
1	1	0	0	1	1
1	1	1	0	1	1

(b) Wahrheitstafel für $\varphi := (X \wedge ((\mathbf{1} \rightarrow \mathbf{0}) \rightarrow \mathbf{0}))$:

X	$\mathbf{1}$	$\mathbf{0}$	$(\mathbf{1} \rightarrow \mathbf{0})$	$((\mathbf{1} \rightarrow \mathbf{0}) \rightarrow \mathbf{0})$	φ
0	1	0	0	1	0
1	1	0	0	1	1

Die **erfüllenden** Belegungen für eine Formel φ entsprechen gerade denjenigen Zeilen der Wahrheitstafel für φ , in denen in der mit “ φ ” beschrifteten Spalte der Wert 1 steht. Das liefert uns ein Werkzeug, um die in Beispiel 3.17 beschriebene Aufgabe zur “Geburtstagsfeier” zu lösen.

Beispiel 3.19. Sei φ die Formel aus Beispiel 3.17. Die Frage “Wie viele (und welche) Freunde werden bestenfalls zur Party kommen?” können wir lösen, in dem wir

- (1) die Wahrheitstafel für φ ermitteln,
- (2) alle Zeilen raussuchen, in denen in der mit “ φ ” beschrifteten Spalte der Wert 1 steht und

- (3) aus diesen Zeilen all jene raussuchen, bei denen in den mit A, B, C, D, E beschrifteten Spalten möglichst viele Einsen stehen. Jede dieser Zeilen repräsentiert dann eine größtmögliche Konstellation von gleichzeitigen Partybesuchern.

Prinzipiell führt diese Vorgehensweise zum Ziel. Leider ist das Verfahren aber recht aufwendig, da die Wahrheitstafel, die man dabei aufstellen muss, sehr groß wird, wie man am Beispiel der Wahrheitstafel für die Formel φ (siehe Abbildung 3.1) sieht. **Erfüllende Belegungen** für φ

A	B	C	D	E	$E \rightarrow (C \wedge D)$	$C \rightarrow A$	$(B \wedge E) \rightarrow \neg D$	$A \rightarrow (B \vee C)$	$(B \wedge A) \rightarrow \neg E$	φ
0	0	0	0	0	1	1	1	1	1	1
0	0	0	0	1	0	1	1	1	1	0
0	0	0	1	0	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	0
0	0	1	0	0	1	0	1	1	1	0
0	0	1	0	1	0	0	1	1	1	0
0	0	1	1	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1	1	1	0
0	1	0	0	0	1	1	1	1	1	1
0	1	0	0	1	0	1	1	1	1	0
0	1	0	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	0	1	1	0
0	1	1	0	0	1	0	1	1	1	0
0	1	1	0	1	0	0	1	1	1	0
0	1	1	1	0	1	0	1	1	1	0
0	1	1	1	1	1	0	0	1	1	0
1	0	0	0	0	1	1	1	0	1	0
1	0	0	0	1	0	1	1	0	1	0
1	0	0	1	0	1	1	1	0	1	0
1	0	0	1	1	0	1	1	0	1	0
1	0	1	0	0	1	1	1	1	1	1
1	0	1	0	1	0	1	1	1	1	0
1	0	1	1	0	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1
1	1	0	0	0	1	1	1	1	1	1
1	1	0	0	1	0	1	1	1	0	0
1	1	0	1	0	1	1	1	1	1	1
1	1	0	1	1	0	1	0	1	0	0
1	1	1	0	0	1	1	1	1	1	1
1	1	1	0	1	0	1	1	1	0	0
1	1	1	1	0	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	0	0

Abbildung 3.1: Wahrheitstafel für die Formel φ aus Beispiel 3.17

werden in Abbildung 3.1 durch Zeilen repräsentiert, die grau unterlegt sind.

In der Wahrheitstafel sieht man, dass es **keine** erfüllende Belegung gibt, bei der in den mit A bis E beschrifteten Spalten insgesamt 5 Einsen stehen, und dass es genau **zwei** erfüllende Belegungen gibt, bei denen in den mit A bis E beschrifteten Spalten insgesamt 4 Einsen stehen, nämlich die beiden Belegungen \mathcal{B}_1 und \mathcal{B}_2 mit

$$\mathcal{B}_1(A) = \mathcal{B}_1(C) = \mathcal{B}_1(D) = \mathcal{B}_1(E) = 1 \quad \text{und} \quad \mathcal{B}_1(B) = 0$$

und

$$\mathcal{B}_2(A) = \mathcal{B}_2(B) = \mathcal{B}_2(C) = \mathcal{B}_2(D) = 1 \quad \text{und} \quad \mathcal{B}_2(E) = 0.$$

Die Antwort auf die Frage “Wie viele (und welche) Freunde werden bestenfalls zur Party kommen?” lautet also: Bestenfalls werden 4 der 5 Freunde kommen, und dafür gibt es zwei Möglichkeiten, nämlich

- (1) dass alle außer Bernd kommen, und
- (2) dass alle außer Eva kommen.

□ Ende Beispiel 3.19

Angesichts der Wahrheitstafel aus Abbildung 3.1 stellt sich die Frage, wie groß die Wahrheitstafel für eine gegebene Formel φ ist. Die Antwort darauf gibt der folgende Satz.

Satz 3.20.

Sei φ eine aussagenlogische Formel und sei $n := |\text{Var}(\varphi)|$ die Anzahl der in φ vorkommenden Variablen. Dann gibt es 2^n verschiedene zu φ passende Belegungen \mathcal{B} mit $\text{Def}(\mathcal{B}) = \text{Var}(\varphi)$.

Beweis: Es gilt:

$$\begin{aligned} & \{\mathcal{B} : \mathcal{B} \text{ ist eine zu } \varphi \text{ passende Belegung mit } \text{Def}(\mathcal{B}) = \text{Var}(\varphi)\} \\ \stackrel{\text{Def. 3.8}}{=} & \{\mathcal{B} : \mathcal{B} : \text{Var}(\varphi) \rightarrow \{0, 1\} \text{ ist eine Funktion}\} \\ \stackrel{\text{Not. 2.30}}{=} & \text{Abb}(\text{Var}(\varphi), \{0, 1\}). \end{aligned}$$

Wir wissen außerdem, dass

$$|\text{Abb}(\text{Var}(\varphi), \{0, 1\})| \stackrel{\text{Fol. 2.39(a)}}{=} |\{0, 1\}|^{|\text{Var}(\varphi)|} \stackrel{n=|\text{Var}(\varphi)|}{=} 2^n.$$

□

Satz 3.20 besagt, dass die Wahrheitstafel einer Formel mit n Variablen genau 2^n Zeilen hat. Wie die folgende Tabelle zeigt, ergibt das bereits bei relativ kleinen Werten von n schon riesige Wahrheitstafeln:

n (Anzahl Variablen)	2^n (Anzahl Zeilen der Wahrheitstafel)
10	$2^{10} = 1.024 \approx 10^3$
20	$2^{20} = 1.048.576 \approx 10^6$
30	$2^{30} = 1.073.741.824 \approx 10^9$
40	$2^{40} = 1.099.511.627.776 \approx 10^{12}$
50	$2^{50} = 1.125.899.906.842.624 \approx 10^{15}$
60	$2^{60} = 1.152.921.504.606.846.976 \approx 10^{18}$

Zum Vergleich: Das Alter des Universums wird auf 13,7 Milliarden Jahre (das sind ungefähr 10^{18} Sekunden) geschätzt.

3.3 Erfüllbarkeit und Allgemeingültigkeit

Definition 3.21. Sei φ eine aussagenlogische Formel.

- (a) φ heißt **erfüllbar**, wenn es (mindestens) eine erfüllende Belegung für φ gibt, d.h. wenn es (mindestens) eine zu φ passende Belegung \mathcal{B} mit $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$ gibt.

unerfüllbar

(b) φ heißt **unerfüllbar**, wenn es **keine** erfüllende Belegung für φ gibt.

allgemeingültig
Tautologie

(c) φ heißt **allgemeingültig** (bzw. **Tautologie**), wenn **jede** zu φ passende Belegung φ erfüllt, d.h. wenn für jede zu φ passende Belegung \mathcal{B} gilt: $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$.

Beispiel 3.22.

(a) Die Formel $((X \vee Y) \wedge (\neg X \vee Y))$ ist

- **erfüllbar**, da z.B. die Belegung \mathcal{B} mit $\mathcal{B}(X) = 0$ und $\mathcal{B}(Y) = 1$ die Formel erfüllt.
- **nicht allgemeingültig**, da z.B. die Belegung \mathcal{B}' mit $\mathcal{B}'(X) = 0$ und $\mathcal{B}'(Y) = 0$ die Formel nicht erfüllt.

(b) Die Formel $(X \wedge \neg X)$ ist **unerfüllbar**, da für jede zur Formel passenden Belegung \mathcal{B} entweder $\mathcal{B}(X) = 1$ oder $\mathcal{B}(X) = 0$ gilt. Beachte:

Falls $\mathcal{B}(X) = 1$, so gilt:

$$\begin{aligned} \llbracket (X \wedge \neg X) \rrbracket^{\mathcal{B}} &= \begin{cases} 1, & \text{falls } \mathcal{B}(X) = 1 \text{ und } \mathcal{B}(X) = 0 \\ 0, & \text{sonst} \end{cases} \\ &= 0 \quad (\text{da } \mathcal{B}(X) = 1 \neq 0). \end{aligned}$$

Falls $\mathcal{B}(X) = 0$, so gilt:

$$\begin{aligned} \llbracket (X \wedge \neg X) \rrbracket^{\mathcal{B}} &= \begin{cases} 1, & \text{falls } \mathcal{B}(X) = 1 \text{ und } \mathcal{B}(X) = 0 \\ 0, & \text{sonst} \end{cases} \\ &= 0 \quad (\text{da } \mathcal{B}(X) = 0 \neq 1). \end{aligned}$$

(c) Die Formel $(X \vee \neg X)$ ist **allgemeingültig**, da für jede zur Formel passenden Belegung \mathcal{B} entweder $\mathcal{B}(X) = 1$ oder $\mathcal{B}(X) = 0$ gilt. Somit gilt für alle zur Formel passenden Belegungen \mathcal{B} , dass $\llbracket (X \vee \neg X) \rrbracket^{\mathcal{B}} = 1$.

Beobachtung 3.23. Für jede aussagenlogische Formel φ gilt:

- (a) φ ist erfüllbar \iff in der Wahrheitstafel für φ steht in der mit “ φ ” beschrifteten Spalte mindestens eine 1.
- (b) φ ist unerfüllbar \iff in der Wahrheitstafel für φ stehen in der mit “ φ ” beschrifteten Spalte nur Nullen.
- (c) φ ist allgemeingültig \iff in der Wahrheitstafel für φ stehen in der mit “ φ ” beschrifteten Spalte nur Einsen.
- (d) φ ist allgemeingültig \iff $\neg\varphi$ ist unerfüllbar.

aussagenlogisches
Erfüllbarkeits-
problem (SAT)

Das **aussagenlogische Erfüllbarkeitsproblem** (Kurz: **SAT**, für englisch: “satisfiability”) ist das folgendermaßen definierte Berechnungsproblem:

AUSSAGENLOGISCHES ERFÜLLBARKEITSPROBLEM (SAT)

Eingabe: Eine aussagenlogische Formel φ .

Frage: Ist φ erfüllbar?

Natürlich kann man dieses Problem dadurch lösen, dass man zur gegebenen Formel φ die Wahrheitstafel aufstellt und testet, ob es in der mit “ φ ” beschrifteten Spalte mindestens eine 1 gibt. Satz 3.20 und die darauf folgende Bemerkung über die Größe von Wahrheitstafeln besagen allerdings, dass dieses Verfahren recht aufwändig ist.

Ein unter dem Stichwort **SAT-Solving** bekannter Teilbereich der Informatik beschäftigt sich mit der Aufgabe, Verfahren zu entwickeln, die das aussagenlogische Erfüllbarkeitsproblem lösen und dabei wesentlich effizienter sind als das vorgestellte Wahrheitstafel-Verfahren. Ein relativ ernüchterndes Resultat, das Sie in weiterführenden Veranstaltungen der Theoretischen Informatik kennenlernen werden, ist allerdings der folgende Satz:

Satz von Cook (1971) (Stephen A. Cook, * 1939, Professor an der University of Toronto)
Das aussagenlogische Erfüllbarkeitsproblem ist NP-vollständig.

Eine präzise Definition des des Begriffs “NP-vollständig” zu geben, würde den Rahmen dieses Vorlesungsskripts sprengen. Grob gesagt bedeutet “NP-vollständig”, dass es wahrscheinlich keinen *effizienten* Algorithmus gibt, der das aussagenlogische Erfüllbarkeitsproblem löst. Andererseits wurden (besonders in den letzten Jahren) Heuristiken und randomisierte Algorithmen entwickelt, die das aussagenlogische Erfüllbarkeitsproblem trotzdem für viele Eingabe-Formeln erstaunlich effizient lösen können. Details zum Thema NP-Vollständigkeit und zum Satz von Cook finden sich in den Büchern [26, 29].

3.4 Folgerung und Äquivalenz

Definition 3.24 (semantische Folgerung).

Seien φ und ψ zwei aussagenlogische Formeln. Wir sagen ψ **folgt aus** φ (kurz: $\varphi \models \psi$ bzw. φ **impliziert** ψ), falls für jede zu φ und ψ passende Belegung \mathcal{B} gilt: Falls $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$, so auch $\llbracket \psi \rrbracket^{\mathcal{B}} = 1$. Somit gilt:

$\varphi \models \psi \iff$ Jede Belegung, die zu φ und ψ passt und die φ erfüllt, erfüllt auch ψ .

Beispiel 3.25.

Sei $\varphi := ((X \vee Y) \wedge (\neg X \vee Y))$ und $\psi := (Y \vee (\neg X \wedge \neg Y))$.

Dann gilt $\varphi \models \psi$, aber es gilt **nicht** $\psi \models \varphi$ (kurz: $\psi \not\models \varphi$), denn:

X	Y	$(X \vee Y)$	$(\neg X \vee Y)$	φ	ψ
0	0	0	1	0	1
0	1	1	1	1	1
1	0	1	0	0	0
1	1	1	1	1	1

Hier repräsentiert jede Zeile eine zu φ und ψ passende Belegung. In jeder Zeile, in der in der mit “ φ ” beschrifteten Spalte eine 1 steht, steht auch in der mit “ ψ ” beschrifteten Spalte eine 1. Somit gilt $\varphi \models \psi$.

Andererseits steht in Zeile 1 in der mit “ ψ ” beschrifteten Spalte eine 1 und in der mit “ φ ” beschrifteten Spalte eine 0. Für die entsprechende Belegung \mathcal{B} (mit $\mathcal{B}(X) = 0$ und $\mathcal{B}(Y) = 0$) gilt also $\llbracket \psi \rrbracket^{\mathcal{B}} = 1$ und $\llbracket \varphi \rrbracket^{\mathcal{B}} = 0$. Daher gilt $\psi \not\models \varphi$.

Beobachtung 3.26. Seien φ und ψ beliebige aussagenlogische Formeln. Dann gilt:

(a) $\mathbf{1} \models \varphi \iff \varphi$ ist allgemeingültig.

- (b) $\varphi \models \mathbf{0} \iff \varphi$ ist unerfüllbar.
 (c) $\varphi \models \psi \iff (\varphi \rightarrow \psi)$ ist allgemeingültig.
 (d) $\varphi \models \psi \iff (\varphi \wedge \neg\psi)$ ist unerfüllbar.

Beweis: Übung. □

Definition 3.27 (logische Äquivalenz).

äquivalent
 $\varphi \equiv \psi$

Zwei aussagenlogische Formeln φ und ψ heißen **äquivalent** (kurz: $\varphi \equiv \psi$), wenn für alle zu φ und ψ passenden Belegungen \mathcal{B} gilt: $\llbracket \varphi \rrbracket^{\mathcal{B}} = \llbracket \psi \rrbracket^{\mathcal{B}}$.

Beispiel 3.28. Sei $\varphi := (X \wedge (X \vee Y))$ und $\psi := X$. Dann ist $\varphi \equiv \psi$, denn

X	Y	$(X \vee Y)$	φ	ψ
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

Hier ist die mit “ φ ” beschriftete Spalte identisch zur mit “ ψ ” beschrifteten Spalte. D.h. für alle zu φ und ψ passenden Belegungen \mathcal{B} gilt $\llbracket \varphi \rrbracket^{\mathcal{B}} = \llbracket \psi \rrbracket^{\mathcal{B}}$. Somit gilt $\varphi \equiv \psi$.

Beobachtung 3.29. Seien φ und ψ aussagenlogische Formeln. Dann gilt:

- (a) $\varphi \equiv \psi \iff (\varphi \leftrightarrow \psi)$ ist allgemeingültig $\iff \varphi \models \psi$ und $\psi \models \varphi$.
 (b) φ ist allgemeingültig $\iff \varphi \equiv \mathbf{1}$.
 (c) φ ist erfüllbar $\iff \varphi \not\equiv \mathbf{0}$ (d.h. “ $\varphi \equiv \mathbf{0}$ ” gilt nicht).

Beweis: Übung. □

Fundamentale Äquivalenzen der Aussagenlogik:

Satz 3.30. Seien φ, ψ und χ aussagenlogische Formeln. Dann gilt:

- (a) **Idempotenz:**
- $(\varphi \wedge \varphi) \equiv \varphi$
 - $(\varphi \vee \varphi) \equiv \varphi$
- (b) **Kommutativität:**
- $(\varphi \wedge \psi) \equiv (\psi \wedge \varphi)$
 - $(\varphi \vee \psi) \equiv (\psi \vee \varphi)$
- (c) **Assoziativität:**
- $((\varphi \wedge \psi) \wedge \chi) \equiv (\varphi \wedge (\psi \wedge \chi))$
 - $((\varphi \vee \psi) \vee \chi) \equiv (\varphi \vee (\psi \vee \chi))$

(d) **Absorption:**

- $(\varphi \wedge (\varphi \vee \psi)) \equiv \varphi$
- $(\varphi \vee (\varphi \wedge \psi)) \equiv \varphi$

(e) **Distributivität:**

- $(\varphi \wedge (\psi \vee \chi)) \equiv ((\varphi \wedge \psi) \vee (\varphi \wedge \chi))$
- $(\varphi \vee (\psi \wedge \chi)) \equiv ((\varphi \vee \psi) \wedge (\varphi \vee \chi))$

(f) **doppelte Negation:**

- $\neg\neg\varphi \equiv \varphi$

(g) **De Morgansche Regeln:**

- $\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$
- $\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$

(h) **Tertium non Datur:**

- $(\varphi \wedge \neg\varphi) \equiv \mathbf{0}$
- $(\varphi \vee \neg\varphi) \equiv \mathbf{1}$

(i) • $(\varphi \wedge \mathbf{1}) \equiv \varphi$

• $(\varphi \wedge \mathbf{0}) \equiv \mathbf{0}$

• $(\varphi \vee \mathbf{1}) \equiv \mathbf{1}$

• $(\varphi \vee \mathbf{0}) \equiv \varphi$

(j) • $\mathbf{1} \equiv \neg\mathbf{0}$

• $\mathbf{0} \equiv \neg\mathbf{1}$

(k) **Elimination der Implikation:**

• $(\varphi \rightarrow \psi) \equiv (\neg\varphi \vee \psi)$

(l) **Elimination der Biimplikation:**

• $(\varphi \leftrightarrow \psi) \equiv ((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi))$

Beweis: Durch einfaches Nachrechnen. Details: Übung. □

Bemerkung 3.31. Durch schrittweises Anwenden der in Satz 3.30 aufgelisteten Äquivalenzen kann man eine gegebene aussagenlogische Formel in eine zu ihr äquivalente Formel umformen.

Beispiel: Sind φ und ψ aussagenlogische Formeln, so gilt:

$$(\varphi \leftrightarrow \psi) \equiv ((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)) \quad (\text{Satz 3.30(l)})$$

$$\equiv ((\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)) \quad (\text{Satz 3.30(k)})$$

3.5 Normalformen

Bisher haben wir gesehen, wie man für eine gegebene aussagenlogische Formel φ eine Wahrheitstafel aufstellen kann.

Frage: Wie kann man umgekehrt zu einer gegebenen Wahrheitstafel eine Formel φ finden, zu der die Wahrheitstafel passt?

Beispiel 3.32. Betrachte die Wahrheitstafel T :

X	Y	Z	φ
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Eine Formel φ , so dass T die Wahrheitstafel für φ ist, kann man folgendermaßen erzeugen:

- Betrachte alle Zeilen von T , in denen in der mit “ φ ” beschrifteten Spalte eine 1 steht.
- Für jede solche Zeile konstruiere eine Formel, die genau von der zur Zeile gehörenden Belegung erfüllt wird.
- Bilde die Disjunktion (d.h. Veroderung) über all diese Formeln. Dies liefert die gesuchte Formel φ .

In unserer Beispiel-Wahrheitstafel T gibt es genau 3 Zeilen, in denen in der mit “ φ ” beschrifteten Spalte eine 1 steht, nämlich die Zeilen

X	Y	Z	φ	zur Belegung der jeweiligen Zeile gehörende Formel:
0	0	0	1	$(\neg X \wedge \neg Y \wedge \neg Z)$
⋮	⋮	⋮	⋮	
1	0	0	1	$(X \wedge \neg Y \wedge \neg Z)$
1	0	1	1	$(X \wedge \neg Y \wedge Z)$
⋮	⋮	⋮	⋮	

Wir erhalten dadurch die folgende zur Wahrheitstafel T passende Formel:

$$\varphi := (\neg X \wedge \neg Y \wedge \neg Z) \vee (X \wedge \neg Y \wedge \neg Z) \vee (X \wedge \neg Y \wedge Z).$$

Generell kann man auf die beschriebene Art zu jeder beliebigen Wahrheitstafel eine aussagenlogische Formel konstruieren, die zur Wahrheitstafel passt. Die so konstruierten Formeln haben eine besonders einfache Form. Sie sind Disjunktionen von Formeln, die aus Konjunktionen von Variablen oder negierten Variablen bestehen. Formeln, die diese spezielle Struktur besitzen, nennt man auch Formeln in **disjunktiver Normalform** (kurz: **DNF**).

Definition 3.33 (disjunktive Normalform, konjunktive Normalform).

- (a) Ein **Literal** ist eine Formel der Form X oder $\neg X$, wobei $X \in \text{AVAR}$ (d.h. X ist eine Aussagenvariable). Ein Literal der Form X , mit $X \in \text{AVAR}$, wird auch **positives Literal** genannt. Eine Formel der Form $\neg X$, mit $X \in \text{AVAR}$, heißt **negatives Literal**.
 Literal
 positives Literal
 negatives Literal
- (b) Eine aussagenlogische Formel ist in **disjunktiver Normalform (DNF)**, wenn sie eine Disjunktion von Konjunktionen von Literalen ist, d.h. wenn sie die Gestalt

$$\bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right)$$

hat, wobei $n, m_1, \dots, m_n \in \mathbb{N}_{>0}$ und $\ell_{i,j}$ ein Literal ist (für jedes $i \in \{1, \dots, n\}$ und $j \in \{1, \dots, m_i\}$).

Die Teilformeln $\kappa_i := \bigwedge_{j=1}^{m_i} \ell_{i,j}$ (für $i \in \{1, \dots, n\}$) heißen **konjunktive Klauseln**.
 konjunktive Klausel

- (c) Eine aussagenlogische Formel ist in **konjunktiver Normalform (KNF)**, wenn sie eine Konjunktion von Disjunktionen von Literalen ist, d.h. wenn sie die Gestalt

$$\bigwedge_{i=1}^n \left(\bigvee_{j=1}^{m_i} \ell_{i,j} \right)$$

hat, wobei $n, m_1, \dots, m_n \in \mathbb{N}_{>0}$ und $\ell_{i,j}$ ein Literal ist (für jedes $i \in \{1, \dots, n\}$ und $j \in \{1, \dots, m_i\}$).

Die Teilformeln $\kappa_i := \bigvee_{j=1}^{m_i} \ell_{i,j}$ (für $i \in \{1, \dots, n\}$) heißen **disjunktive Klauseln**.
 disjunktive Klausel

Normalformen spielen in vielen Anwendungsgebieten eine wichtige Rolle. Beispielsweise geht man in der Schaltungstechnik (Hardware-Entwurf) oft von DNF-Formeln aus, während bei der aussagenlogischen Modellbildung oftmals KNF-Formeln auftreten, da sich eine Sammlung von einfach strukturierten Aussagen sehr gut durch eine Konjunktion von Klauseln ausdrücken lässt.

Satz 3.34.

Für jede aussagenlogische Formel φ gibt es eine Formel ψ_D in DNF und eine Formel ψ_K in KNF, so dass $\varphi \equiv \psi_D$ und $\varphi \equiv \psi_K$.

Das heißt: Jede Formel ist äquivalent zu einer Formel in DNF und zu einer Formel in KNF.

Beweisidee:

- Zur Konstruktion einer zu φ äquivalenten Formel ψ_D in DNF stellen wir zunächst die Wahrheitstafel für φ auf. Falls diese in der mit “ φ ” beschrifteten Spalte nur Nullen hat (d.h. φ ist unerfüllbar), so setzen wir $\psi_D := (V_0 \wedge \neg V_0)$. Offensichtlich ist ψ_D in DNF und unerfüllbar, also äquivalent zu φ .

Falls die mit “ φ ” beschriftete Spalte der Wahrheitstafel mindestens eine 1 enthält, so gehen wir wie in Beispiel 3.32 vor, um eine zu φ äquivalente Formel ψ_D in DNF zu konstruieren.

- Zur Konstruktion einer zu φ äquivalenten Formel ψ_K in KNF können wir folgendermaßen vorgehen:

- (1) Sei $\varphi' := \neg\varphi$.
- (2) Konstruiere eine zu φ' äquivalente Formel ψ'_D in DNF.
Sei $\bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right)$ die Gestalt von ψ'_D .
- (3) Für alle i, j sei $\tilde{\ell}_{i,j} := \begin{cases} \neg X, & \text{falls } \ell_{i,j} = X \text{ für ein } X \in \text{AVAR} \\ X, & \text{falls } \ell_{i,j} = \neg X \text{ für ein } X \in \text{AVAR}. \end{cases}$
- (4) Setze $\psi_K := \bigwedge_{i=1}^n \left(\bigvee_{j=1}^{m_i} \tilde{\ell}_{i,j} \right)$.

Offensichtlich ist ψ_K eine Formel in KNF. Außerdem gilt:

$$\begin{aligned}
\varphi &\equiv \neg\varphi' \\
&\equiv \neg\psi'_D \\
&\equiv \neg\left(\bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} \ell_{i,j}\right)\right) \\
&\stackrel{\text{Satz 3.30(g)}}{\equiv} \left(\bigwedge_{i=1}^n \neg\left(\bigwedge_{j=1}^{m_i} \ell_{i,j}\right)\right) \\
&\stackrel{\text{Satz 3.30(g)}}{\equiv} \bigwedge_{i=1}^n \left(\bigvee_{j=1}^{m_i} \neg\ell_{i,j}\right) \\
&\stackrel{\text{Def. } \tilde{\ell}_{i,j}}{\equiv} \bigwedge_{i=1}^n \left(\bigvee_{j=1}^{m_i} \tilde{\ell}_{i,j}\right) \\
&\stackrel{\text{Def. } \psi_K}{=} \psi_K.
\end{aligned}$$

□

Abgesehen von DNF und KNF gibt es noch eine weitere wichtige Normalform, die so genannte Negationsnormalform.

Definition 3.35 (Negationsnormalform).

Eine aussagenlogische Formel ist in **Negationsnormalform (NNF)**, wenn sie keines der Symbole $\rightarrow, \leftrightarrow, \mathbf{0}, \mathbf{1}$ enthält und Negationszeichen nur unmittelbar vor Variablen auftreten.

Negations-
normalform
NNF

Rekursiv lässt sich die Menge der Formeln in NNF folgendermaßen definieren.

Basisregeln:

- Für jedes $X \in \text{AVAR}$ ist sowohl X als auch $\neg X$ eine Formel in NNF.

Rekursive Regeln:

- Sind φ und ψ Formeln in NNF, so sind auch $(\varphi \wedge \psi)$ und $(\varphi \vee \psi)$ Formeln in NNF.

Beobachtung 3.36. Jede Formel, die in KNF oder in DNF ist, ist auch in NNF. Aus Satz 3.34 folgt also insbesondere, dass jede aussagenlogische Formel äquivalent zu einer Formel in NNF ist.

Beachte: Nicht jede Formel in NNF ist auch in KNF oder in DNF.

Beispiel: $((X \wedge \neg Y) \vee (\neg X \wedge Y)) \wedge \neg Z$ ist in NNF, aber weder in KNF noch in DNF.

Beobachtung 3.37. Ein einfaches Verfahren zur Transformation einer gegebenen aussagenlogischen Formel in eine äquivalente Formel in NNF beruht auf der wiederholten Anwendung der De Morganschen Regeln (Satz 3.30(g)) und der Regel für “doppelte Negation” (Satz 3.30(f)):

- Mit den De Morganschen Regeln

$$\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi) \quad \text{bzw.} \quad \neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$$

ziehen wir das Negationszeichen nach innen.

- Mit der Regel für “doppelte Negation”

$$\neg\neg\varphi \equiv \varphi$$

können wir Schritt für Schritt mehrfach hintereinander vorkommende Negationszeichen eliminieren.

- Eventuell in der Formel vorkommende Implikationspfeile “ \rightarrow ” oder Biimplikationspfeile “ \leftrightarrow ” eliminieren wir durch Verwenden von Satz 3.30(k)

$$(\varphi \rightarrow \psi) \equiv (\neg\varphi \vee \psi)$$

und Satz 3.30(l)

$$(\varphi \leftrightarrow \psi) \equiv ((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)).$$

- Eventuelle Vorkommen der Symbole **0** bzw. **1** ersetzen wir durch die Formeln

$$(V_0 \wedge \neg V_0) \quad \text{bzw.} \quad (V_0 \vee \neg V_0).$$

Beispiel 3.38.

Das Ziel ist, die Formel

$$\left((\neg V_0 \wedge \neg((V_0 \vee V_1) \rightarrow V_0)) \rightarrow \mathbf{0} \right)$$

in NNF zu bringen, d.h. eine zur gegebenen Formel äquivalente Formel in NNF zu finden.

Lösung: (der Teil einer Formel, der als nächstes ersetzt wird, ist im Folgenden jeweils unterstrichen)

$$\begin{aligned} \left((\neg V_0 \wedge \neg((V_0 \vee V_1) \rightarrow V_0)) \rightarrow \mathbf{0} \right) &\equiv \left((\neg V_0 \wedge \neg((V_0 \vee V_1) \rightarrow V_0)) \supseteq (V_0 \wedge \neg V_0) \right) \\ &\equiv \left(\neg(\neg V_0 \wedge \neg((V_0 \vee V_1) \supseteq V_0)) \vee (V_0 \wedge \neg V_0) \right) \\ &\equiv \left(\neg(\neg V_0 \wedge \neg(\neg(V_0 \vee V_1) \vee V_0)) \vee (V_0 \wedge \neg V_0) \right) \\ &\equiv \left((\neg\neg V_0 \vee \neg\neg(\neg(V_0 \vee V_1) \vee V_0)) \vee (V_0 \wedge \neg V_0) \right) \\ &\equiv \left((V_0 \vee (\neg(V_0 \vee V_1) \vee V_0)) \vee (V_0 \wedge \neg V_0) \right) \\ &\equiv \left((V_0 \vee ((\neg V_0 \wedge \neg V_1) \vee V_0)) \vee (V_0 \wedge \neg V_0) \right). \end{aligned}$$

Diese Formel ist offensichtlich in Negationsnormalform.

Unter zusätzlicher Verwendung der “Distributivitätsregel” (Satz 3.30(e)) erhält man Verfahren zur Transformation einer gegebenen Formel in eine äquivalente Formel in DNF bzw. KNF, bei denen man nicht zuerst eine Wahrheitstafel aufstellen muss. Diese Verfahren sind vor allem dann ratsam, wenn die gegebene Formel sehr viele verschiedene Variablen enthält, die zugehörige Wahrheitstafel also sehr groß wird.

Algorithmus 3.39 (Ein KNF-Algorithmus).

Eingabe: Eine aussagenlogische Formel φ .

Ausgabe: Eine zu φ äquivalente Formel φ' in KNF.

Verfahren:

- (1) Konstruiere eine zu φ äquivalente Formel φ' in NNF
(beispielsweise mit dem in Beobachtung 3.37 beschriebenen Verfahren).
- (2) Wiederhole folgende Schritte:
 - (i) Falls φ' in KNF ist, so halte mit Ausgabe φ' .
 - (ii) Falls φ' nicht in KNF ist, so ersetze eine Teilformel von φ' der Gestalt $(\psi_1 \vee (\psi_2 \wedge \psi_3))$ durch die Formel

$$((\psi_1 \vee \psi_2) \wedge (\psi_1 \vee \psi_3))$$

oder ersetze eine Teilformel von φ' der Gestalt $((\psi_2 \wedge \psi_3) \vee \psi_1)$ durch die Formel

$$((\psi_2 \vee \psi_1) \wedge (\psi_3 \vee \psi_1)).$$

Sei φ'' die resultierende Formel. Setze $\varphi' := \varphi''$.

Algorithmus 3.40 (Ein DNF-Algorithmus).

Eingabe: Eine aussagenlogische Formel φ

Ausgabe: Eine zu φ äquivalente Formel φ' in DNF.

Verfahren:

- (1) Konstruiere eine zu φ äquivalente Formel φ' in NNF.
- (2) Wiederhole folgende Schritte:
 - (i) Falls φ' in DNF ist, so halte mit Ausgabe φ' .
 - (ii) Falls φ' nicht in DNF ist, so ersetze eine Teilformel von φ' der Gestalt $(\psi_1 \wedge (\psi_2 \vee \psi_3))$ durch die Formel

$$((\psi_1 \wedge \psi_2) \vee (\psi_1 \wedge \psi_3))$$

oder ersetze eine Teilformel von φ' der Gestalt $((\psi_2 \vee \psi_3) \wedge \psi_1)$ durch die Formel

$$((\psi_2 \wedge \psi_1) \vee (\psi_3 \wedge \psi_1)).$$

Sei φ'' die resultierende Formel. Setze $\varphi' := \varphi''$.

Satz 3.41 (Korrektheit der Algorithmen 3.39 und 3.40).

Für jede aussagenlogische Formel φ gilt:

- (a) Algorithmus 3.39 hält bei Eingabe einer aussagenlogischen Formel φ nach endlich vielen Schritten an und gibt eine zu φ äquivalente Formel in KNF aus.
- (b) Algorithmus 3.40 hält bei Eingabe einer aussagenlogischen Formel φ nach endlich vielen Schritten an und gibt eine zu φ äquivalente Formel in DNF aus.

Beweis: Übung. □

Beispiel 3.42. Sei $\varphi := ((\neg V_0 \wedge (V_0 \rightarrow V_1)) \vee (V_2 \rightarrow V_3))$.

Transformation von φ in NNF:

$$\varphi = ((\neg V_0 \wedge (V_0 \Rightarrow V_1)) \vee (V_2 \Rightarrow V_3)) \equiv \underbrace{((\neg V_0 \wedge (\neg V_0 \vee V_1)) \vee (\neg V_2 \vee V_3))}_{=: \varphi'}.$$

Transformation von φ in DNF mittels Algorithmus 3.40:

- (1) Die Transformation von φ in NNF liefert $\varphi' = ((\neg V_0 \wedge (\neg V_0 \vee V_1)) \vee (\neg V_2 \vee V_3))$.
- (2) Einmaliges Anwenden von Zeile (ii) des Algorithmus (auf die unterstrichene Teilformel von φ') liefert:

$$\varphi'' := (((\neg V_0 \wedge \neg V_0) \vee (\neg V_0 \wedge V_1)) \vee (\neg V_2 \vee V_3)).$$

Diese Formel ist die DNF-Formel, die von dem Algorithmus ausgegeben wird (die einzelnen konjunktiven Klauseln sind jeweils unterstrichen).

Transformation von φ in KNF mittels Algorithmus 3.39:

- (1) Die Transformation von φ in NNF liefert $\varphi' = ((\neg V_0 \wedge (\neg V_0 \vee V_1)) \vee (\neg V_2 \vee V_3))$.
- (2) Einmaliges Anwenden von Zeile (ii) des Algorithmus (auf den unterstrichenen Teil der Formel φ') liefert:

$$\varphi'' := ((\neg V_0 \vee (\neg V_2 \vee V_3)) \wedge ((\neg V_0 \vee V_1) \vee (\neg V_2 \vee V_3))).$$

Dies ist die KNF-Formel, die von dem Algorithmus ausgegeben wird (die einzelnen disjunktiven Klauseln sind jeweils unterstrichen).

Am Ende von Abschnitt 3.3 wurde darauf hingewiesen, dass die Aufgabe, für eine gegebene Formel φ herauszufinden, ob sie erfüllbar ist, im Allgemeinen ein recht schwieriges Problem ist. Für den Spezialfall, dass φ eine Formel in DNF ist, lässt sich das Erfüllbarkeitsproblem allerdings sehr effizient lösen:

Beobachtung 3.43 (effizienter Erfüllbarkeitstest für DNF-Formeln).

Sei φ eine Formel in DNF, d.h. φ ist von der Form

$$\bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} l_{i,j} \right), \quad \text{für Literale } l_{i,j}.$$

D.h. φ ist von der Form

$$\kappa_1 \vee \cdots \vee \kappa_n,$$

wobei, für jedes $i \in \{1, \dots, n\}$, κ_i die konjunktive Klausel

$$\kappa_i := \ell_{i,1} \wedge \cdots \wedge \ell_{i,m_i}$$

ist. Offensichtlich gilt:

φ ist erfüllbar \iff für mindestens ein $i \in \{1, \dots, n\}$ ist die Formel κ_i erfüllbar.

Da κ_i eine Konjunktion von Literalen (d.h. von Variablen und/oder negierten Variablen) ist, gilt:

κ_i ist erfüllbar \iff es gibt keine $j, j' \in \{1, \dots, m_i\}$, so dass $\ell_{i,j} = \neg \ell_{i,j'}$.

Daher testet der folgende Algorithmus, ob eine gegebene DNF-Formel erfüllbar ist.

Eingabe: Eine aussagenlogische Formel $\varphi = \bigvee_{i=1}^n \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right)$ in DNF

Ziel: Entscheide, ob φ erfüllbar ist.

Verfahren:

- (1) Für $i = 1, \dots, n$
- (2) Für $j = 1, \dots, m_i$
- (3) Für $j' = j + 1, \dots, m_i$
- (4) Falls $\ell_{i,j} = \neg \ell_{i,j'}$ oder $\ell_{i,j'} = \neg \ell_{i,j}$, dann:
- (5) Falls $i = n$ ist, so mache in Zeile 7 weiter;
ansonsten setze $i := i + 1$ und mache in Zeile 2 weiter.
- (6) Halte mit Ausgabe “ φ ist erfüllbar”.
- (7) Halte mit Ausgabe “ φ ist unerfüllbar”.

Um aussagenlogische Formeln φ von **beliebiger** Form auf Erfüllbarkeit zu testen, kann man dann folgendermaßen vorgehen:

Schritt 1: Transformiere φ in eine äquivalente Formel φ' in DNF (z.B. mit Algorithmus 3.40).

Schritt 2: Entscheide, ob φ' erfüllbar ist (z.B. mit dem obigen Verfahren).

Das Ausführen von Schritt 1 kann dabei u.U. aber leider wieder sehr lange dauern, da es einige Formeln gibt, zu denen äquivalente Formeln in DNF zwangsläufig sehr groß sind. Dies wird durch den folgenden Satz präzisiert:

Satz 3.44.

Sei $n \in \mathbb{N}_{>0}$, seien $X_1, \dots, X_n, Y_1, \dots, Y_n$ genau $2n$ verschiedene aussagenlogische Variablen, und sei

$$\varphi_n := \bigwedge_{i=1}^n (X_i \leftrightarrow Y_i).$$

Dann hat jede zu φ_n äquivalente Formel in DNF mindestens 2^n konjunktive Klauseln.

Beweis: Übung. (Details finden sich in Aufgabe 3.25.) □

3.6 Literaturhinweise

Als vertiefende Lektüre seien die Kapitel 1 und 2 in [17] sowie die Einleitung und Kapitel 1 in [27] empfohlen. Einen umfassenden Überblick über die Rolle der Logik in der Informatik gibt [10]. Details zum Thema NP-Vollständigkeit und zum Satz von Cook finden sich in den Büchern [26, 29].

Quellennachweis: Teile dieses Kapitels orientieren sich an [9]. Die folgende Aufgabe 3.3 ist aus [17] entnommen. Beispiel 3.1 ist eine Variante einer Übungsaufgabe aus [17].

3.7 Übungsaufgaben zu Kapitel 3

Aufgabe 3.1.

(a) Welche der folgenden Wörter gehören gemäß Definition 3.3 zur Sprache AL, welche nicht?

- $(V_1 \wedge \mathbf{1})$
- $(V_1 \wedge \mathbf{101})$
- $(\mathbf{1} \leftrightarrow \mathbf{0})$
- $(x \vee y)$
- $(\neg(V_1 \wedge V_2) \vee V_3)$
- $\neg(V_1 \wedge V_2) \vee V_3$
- $(V_1 \rightarrow V_2)$
- $(V_1 \rightarrow V_2 \rightarrow V_3)$
- $(V_1 \leftarrow V_2)$
- $(V_1 \leftrightarrow V_2)$
- $\neg(\neg V_{42})$

(b) Beweisen Sie, dass für die Formel $\varphi := ((V_1 \leftrightarrow \mathbf{1}) \wedge (V_1 \rightarrow (V_2 \wedge \mathbf{0})))$ gilt: $\varphi \in \text{AL}$.

(c) Betrachten Sie die Formel φ aus (b) und die Belegung $\mathcal{B}: \text{Var}(\varphi) \rightarrow \{0, 1\}$ mit $\mathcal{B}(V_1) = 1$ und $\mathcal{B}(V_2) = 0$. Berechnen Sie den Wert $\llbracket \varphi \rrbracket^{\mathcal{B}}$.

(d) Geben Sie den Syntaxbaum der Formel φ aus (b) an.

Aufgabe 3.2.

(a) Betrachten Sie die folgenden Wörter und beweisen Sie jeweils, dass das Wort gemäß Definition 3.3 zur Sprache AL gehört oder begründen Sie, warum das Wort nicht zu AL gehört.

- (i) $\neg((V_3 \wedge \neg \mathbf{0}) \rightarrow (V_0 \vee (\neg \neg V_1 \wedge V_4)))$
- (ii) $(V_5 \leftrightarrow X) \wedge (V_{23} \rightarrow (V_1 \wedge \mathbf{0}))$
- (iii) $((V_{11} \leftarrow V_7) \vee \neg \neg V_5)$
- (iv) $((V_9 \vee \neg(\neg V_{42}) \vee \neg V_2) \rightarrow \mathbf{1})$

(b) Betrachten Sie die aussagenlogische Formel

$$\varphi := \left((\neg V_0 \wedge V_1) \rightarrow (V_0 \wedge (V_1 \vee \neg V_2)) \right)$$

und die Belegung $\mathcal{B}: \text{Var}(\varphi) \rightarrow \{0, 1\}$ mit $\mathcal{B}(V_0) = 1$ und $\mathcal{B}(V_1) = \mathcal{B}(V_2) = 0$. Berechnen Sie den Wert $\llbracket \varphi \rrbracket^{\mathcal{B}}$ in nachvollziehbaren Schritten analog zu Beispiel 3.10.

(c) Geben Sie den Syntaxbaum und die ASCII-Darstellung der Formel φ aus (b) an.

Aufgabe 3.3. Schon kurz nach der Geburt von Herakles und Eurystheus entstand ein Streit, wer von den beiden der rechtmäßige Herrscher sei. Dazu wurden die drei bekanntesten Orakel Griechenlands befragt.

Das Ammonion gab bekannt, dass die Orakelsprüche aus Klaros grundsätzlich falsch seien. Ebenso ließ das Orakel aus Klaros verlauten, dass die Orakelsprüche aus Delphi samt und sonders unzutreffend seien. Das Orakel aus Delphi jedoch behauptete, sowohl die Sprüche des Ammonions als auch die des Orakels in Klaros seien unwahr.

Wem sollen die armen Griechen nun glauben?

- (a) Zerlegen Sie den obigen Text in atomare Aussagen und geben Sie eine aussagenlogische Formel φ an, die das im Text zusammengefasste Wissen repräsentiert (ähnlich wie in den Beispielen 3.1, 3.17 und 3.19).
- (b) Geben Sie für Ihre Formel φ aus (a) eine Belegung \mathcal{B} an, die besagt, dass das Ammonion die Wahrheit sagt und die beiden anderen Orakel lügen. Erfüllt \mathcal{B} die Formel φ ?
- (c) Welchen Orakeln können die Griechen glauben, welchen nicht? Falls es mehrere Möglichkeiten gibt, geben Sie alle an.

Aufgabe 3.4. Die Mensa der Universität versucht ständig, ihr Angebot an die Wünsche der Studierenden anzupassen. Die neueste Idee der Mensaleitung ist es, zu jeder Hauptmahlzeit ein Stück Brot, eine Suppe und/oder ein Dessert zu reichen. Um die beliebteste Kombination aus Brot, Suppe und/oder Dessert zu erfahren, startet die Mensaleitung eine Umfrage unter den Studierenden, die zu folgenden drei Anforderungen X , Y und Z führt:

X : Wenn Suppe serviert wird, dann soll kein Dessert gereicht werden.

Y : Es soll genau dann Brot oder Dessert geben, wenn auch Suppe serviert wird.

Z : Falls Suppe aber kein Dessert gereicht wird, soll es kein Brot geben.

1. Geben Sie für jede der Anforderungen X , Y und Z eine aussagenlogische Formel an, die die jeweilige Anforderung widerspiegelt (ähnlich zu Beispiel 3.16 und 3.17 im Skript). Benutzen Sie dafür die Aussagenvariablen B (es gibt Brot), D (es gibt Dessert) und S (es gibt Suppe).
2. Stellen Sie eine aussagenlogische Formel φ auf, die die Aussagenvariablen B , D und S benutzt und die aussagt, dass alle Anforderungen gleichzeitig gelten sollen.
3. Geben Sie für Ihre Formel φ aus (b) eine Belegung an, die besagt, dass es Dessert und Suppe, aber kein Brot gibt. Erfüllt diese Belegung die Formel φ ?

4. Welche Kombination(en) von Brot, Suppe und/oder Dessert kann die Mensaleitung wählen, um allen Anforderungen gerecht zu werden? Überlegen Sie sich dazu anhand einer Wahrheitstafel, welche Belegung(en) die Formel φ aus (b) erfüllen.

Aufgabe 3.5. USA, 4. November 2008. Vor einem Wahllokal befragt ein Journalist vier Freunde A, B, C und D, die gerade das Wahllokal verlassen haben, wie sie gewählt haben. A sagt: „Falls B für Obama gestimmt hat, dann haben auch C und D für Obama gestimmt.“ B sagt: „A hat auf keinen Fall für Obama gestimmt, aber D.“ C sagt: „B hat nur dann für McCain gestimmt, wenn A für Obama gestimmt hat.“ D sagt schließlich: „Wenn C für Obama gestimmt hat, dann hat A für McCain oder B für Obama gestimmt.“ Wir nehmen an, dass jeder die Wahrheit gesagt und entweder Obama oder McCain gewählt hat.

- (a) Zerlegen Sie den obigen Text in atomare Aussagen und geben Sie eine aussagenlogische Formel φ an, die das im Text zusammengefasste Wissen repräsentiert (ähnlich wie in den Beispielen 3.1, 3.17 und 3.19).
- (b) Geben Sie für Ihre Formel φ aus (a) eine Belegung \mathcal{B} an, die besagt, dass A, B und C Obama gewählt haben und D für McCain gestimmt hat. Erfüllt \mathcal{B} die Formel φ ?
- (c) Wen haben A, B, C und D jeweils gewählt? Falls es mehrere Möglichkeiten gibt, geben Sie alle an.

Aufgabe 3.6. Auf der Insel Wafa leben zwei Stämme: Die Was, die immer die Wahrheit sagen, und die Fas, die immer lügen. Ein Reisender besucht die Insel und kommt mit drei Einwohnern A, B, C ins Gespräch. Der Reisende schreibt daraufhin folgende atomare Aussagen in sein Notizbuch:

- X_A : A sagt die Wahrheit
- X_B : B sagt die Wahrheit
- X_C : C sagt die Wahrheit

- (a) Sei $\mathcal{B}: \{X_A, X_B, X_C\} \rightarrow \{0, 1\}$ die Belegung mit $\mathcal{B}(X_A) = 1$, $\mathcal{B}(X_B) = 0$ und $\mathcal{B}(X_C) = 0$. Beschreiben Sie umgangssprachlich, welcher Sachverhalt durch die Belegung \mathcal{B} ausgedrückt wird. Was folgt daraus über die Stammesangehörigkeit der drei Einwohner A, B und C?

Die Informationen, die der Reisende im Gespräch erhalten hat, fasst er durch folgende aussagenlogische Formeln zusammen:

- $\varphi_A := (X_A \leftrightarrow (\neg X_B \vee \neg X_C))$
- $\varphi_B := (X_B \leftrightarrow (X_A \rightarrow X_C))$
- $\varphi_C := (X_C \leftrightarrow (\neg X_B \rightarrow X_A))$

Er merkt an, dass die durch $\varphi_A, \varphi_B, \varphi_C$ formalisierten Aussagen der Wahrheit entsprechen.

- (b) Beschreiben Sie umgangssprachlich, was jede der Formeln $\varphi_A, \varphi_B, \varphi_C$ aussagt.
- (b) Zu welchen Stämmen gehören A, B und C?

Aufgabe 3.7. ² Auf einer einsamen Insel leben nur Ritter und Knappen. Während die Ritter immer die Wahrheit sagen, lügen die Knappen ständig. Als Besucher kommen Sie auf die Insel und treffen die drei Bewohner Adrian, Ben und Colin. Jeder von diesen Bewohnern behauptet etwas über die anderen. Diese Behauptungen sind im Folgenden durch aussagenlogische Formeln repräsentiert:

- Adrian behauptet: $\varphi_A := (\neg B \wedge C)$
- Ben behauptet: $\varphi_B := \neg A$
- Colin behauptet: $\varphi_C := \neg(\neg A \vee \neg B)$

Dabei bedeuten die Aussagenvariablen:

- A : Adrian sagt die Wahrheit.
- B : Ben sagt die Wahrheit.
- C : Colin sagt die Wahrheit.

1. Beschreiben Sie umgangssprachlich, was jede der Formeln φ_A , φ_B und φ_C aussagt.
2. Wer von Adrian, Ben und Colin ist Ritter, wer ist Knappe?

Hinweis: Beachten Sie dazu, dass die drei Aussagen, die durch φ_A , φ_B und φ_C repräsentiert werden, wahr oder falsch sein müssen, abhängig davon, ob der jeweils Aussagende lügt oder die Wahrheit sagt.

Aufgabe 3.8. Zwei Analysten streiten sich, wer von ihnen denn nun am besten Aktienkurse voraussagen kann. Dazu wollen sie drei zufällig anwesende Anleger A, B und C befragen. Das wäre nicht weiter schwierig, wenn sich A, B und C nicht folgendes (repräsentiert durch aussagenlogische Formeln) vorwerfen würden:

- A behauptet: $\varphi_A := (\neg B \vee \neg C)$
- B behauptet: $\varphi_B := \neg A$
- C behauptet: $\varphi_C := (A \wedge \neg B)$

Hierbei bedeuten die Aussagenvariablen:

- A : A sagt die Wahrheit.
- B : B sagt die Wahrheit.
- C : C sagt die Wahrheit.

- (a) Beschreiben Sie umgangssprachlich, was jede der Formeln φ_A , φ_B , φ_C aussagt.
- (b) Wem können die Analysten glauben und wem nicht? Falls es mehrere Möglichkeiten gibt, geben Sie alle an.

Aufgabe 3.9. Das nächste Großprojekt der Deutschen Bahn ist bereits in Planung: Unter dem Namen „Frankfurt 22“ soll der Frankfurter Kopfbahnhof in einen unterirdischen Durchgangsbahnhof umgebaut werden. Die Abteilung Kundenbindung/CRM der Deutschen Bahn hat herausgefunden, dass die folgenden Zusammenhänge gelten:

²Diese Übungsaufgabe entstand in Anlehnung an eine Übungsaufgabe von Raymond M. Smullyan.

- I: Die Fahrzeiten erhöhen sich genau dann, wenn der Bahnhof nicht umgebaut wird.
- II: Die Kunden der Deutschen Bahn sind unzufrieden, wenn sich die Fahrpreise oder die Fahrzeiten erhöhen.
- III: Der Bahnhof wird nicht umgebaut, wenn die Fahrpreise nicht erhöht werden.
- (a) Geben Sie für jeden der Zusammenhänge I, II und III eine aussagenlogische Formel an, die den jeweiligen Zusammenhang widerspiegelt (ähnlich zu Beispiel 3.16 und 3.17 im Skript). Benutzen Sie dafür die Aussagenvariablen Z (die Fahrzeiten erhöhen sich), B (der Bahnhof wird umgebaut), P (die Preise erhöhen sich) und K (die Kunden sind zufrieden).
- (b) Stellen Sie eine aussagenlogische Formel φ auf, die die Aussagenvariablen Z , B , P und K benutzt und die aussagt, dass alle Zusammenhänge gleichzeitig gelten.
- (c) Geben Sie für Ihre Formel φ aus (b) eine Belegung an, die besagt, dass der Bahnhof umgebaut wird, die Fahrzeiten und Fahrpreise sich nicht erhöhen und die Kunden zufrieden sind. Erfüllt diese Belegung die Formel φ ?
- (d) Welche Kombination(en) von Bahnhofsumbau, Fahrpreis- und Fahrzeiterhöhung kann die Deutsche Bahn wählen, um die Kundenzufriedenheit sicherzustellen? Überlegen Sie sich dazu anhand einer Wahrheitstafel, welche Belegung(en) die Formel φ aus (b) erfüllen und wie es um die Kundenzufriedenheit bei diesen Belegungen steht.

Aufgabe 3.10. Es ist ein bisher gut gehütetes Geheimnis, dass Lummerland Teil der Euro-Zone ist.³ Aber ähnlich wie bei anderen Ländern auch haben die Turbulenzen an den Finanzmärkten den Staatshaushalt Lummerlands in arge Bedrängnis gebracht. Die Situation ist so ernst, dass Staatsoberhaupt König Alfons der Viertel-vor-Zwölfte die Europäische Zentralbank EZB um einen Kredit bitten muss. Doch die EZB ist streng und vergibt den Kredit nur, wenn die folgenden Anforderungen erfüllt sind:

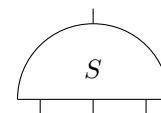
- I: Wenn die Ausgaben für Bildung nicht erhöht werden, müssen die Banken stärker kontrolliert werden.
- II: Wenn Staatseigentum verkauft wird oder die Steuern gesenkt werden, dann dürfen die Ausgaben für Bildung nicht erhöht werden.
- III: Die Banken werden genau dann stärker kontrolliert, wenn die Ausgaben für Bildung erhöht werden und die Steuern nicht gesenkt werden.
- (a) Geben Sie für jede der Anforderungen I, II und III eine aussagenlogische Formel an, die die jeweilige Anforderung widerspiegelt (ähnlich zu Beispiel 3.16 und 3.17 im Skript). Benutzen Sie dafür die atomaren Aussagen S (die Steuern werden gesenkt), B (die Ausgaben für Bildung werden erhöht), V (Staatseigentum wird verkauft) und K (die Banken werden stärker kontrolliert).
- (b) Stellen Sie eine aussagenlogische Formel φ auf, die die atomaren Aussagen S , B , V und K benutzt und die widerspiegelt, dass alle Anforderungen gleichzeitig erfüllt sein müssen.

³Dies ist so natürlich nicht ganz richtig. Richtig ist, das uns Lummerland und König Alfons der Viertel-vor-Zwölfte aus dem Kinderbuch „Jim Knopf und Lukas der Lokomotivführer“ von Michael Ende bekannt sind. Uns ist auch nicht bekannt, ob Rating-Agenturen die Kreditwürdigkeit Lummerlands anzweifeln.

- (c) Geben Sie für Ihre Formel φ aus (b) eine Belegung an, die besagt, dass die Steuern gesenkt werden, die Ausgaben für Bildung sich nicht erhöhen, Staatseigentum verkauft wird und die Banken stärker kontrolliert werden. Erfüllt diese Belegung die Formel φ ?
- (d) Welche Maßnahmen genau muss König Alfons der Viertel-vor-Zwölfte aus den Möglichkeiten Steuersenkung, Bildungsausgabenerhöhung, Verkauf des Staatseigentums und Verstärkung der Bankenkontrolle treffen und welche muss er unterlassen, um allen Anforderungen der EZB gerecht zu werden? Überlegen Sie sich dazu anhand einer Wahrheitstafel, welche Belegungen die Formel φ aus (b) erfüllen.

Aufgabe 3.11. Ein Schaltkreis mit n Eingängen und einem Ausgang ist ein elektronischer Baustein, der eine Funktion $f : \{0, 1\}^n \rightarrow \{0, 1\}$ berechnet. Dabei gilt für jede Komponente i der n Komponenten des Eingabetupels, dass sie genau dann 1 ist, falls am Schaltkreis am i -ten Eingang Strom anliegt, und der Funktionswert von f ist genau dann 1, falls am Ausgang Strom anliegt.

Wir betrachten den Schaltkreis S mit 3 Eingängen. Die Eingänge des Schaltkreises sind der Reihe nach von 1 bis 3 durchnummeriert. Über den Schaltkreis S ist nun folgendes bekannt:



Am Ausgang liegt Strom an, falls jede der folgenden drei Eigenschaften erfüllt ist:

- (I.) Aus der Tatsache, dass an dem ersten Eingang genau dann Strom anliegt wenn am zweiten Eingang Strom anliegt, folgt die Tatsache, dass am dritten Eingang Strom anliegt,
- (II.) Es gilt, am dritten Eingang liegt kein Strom an oder am zweiten Eingang liegt kein Strom an oder am ersten Eingang liegt Strom an.
- (III.) Es gilt, am dritten Eingang liegt kein Strom an, oder falls am ersten Eingang Strom anliegt, dann auch am zweiten.
- (a) Modellieren Sie das Verhalten des Schaltkreises S als aussagenlogische Formel φ , welche genau dann wahr ist, falls am Schaltkreis am Ausgang Strom anliegt. Benutzen Sie hierfür die drei atomaren Aussagen X_1 , X_2 und X_3 , wobei die atomare Aussage X_i genau dann erfüllt ist, falls am i -ten Eingang Strom anliegt. Weiterhin soll die Formel φ eine Konjunktion von drei Teilformeln φ_1 , φ_2 und φ_3 sein, wobei φ_1 die erste Eigenschaft des Schaltkreises modelliert. φ_2 die zweite und φ_3 die dritte.
- (b) Stellen Sie eine Wahrheitstabelle für φ auf.
- (c) Welche Funktion berechnet der Schaltkreis? D.h., für welche Art von Eingaben ist die Ausgabe positiv bzw. liegt Strom am Ausgang an?

Aufgabe 3.12. Es sei die aussagenlogische Formel

$$\varphi := \neg(((V_1 \leftrightarrow \neg V_2) \vee (V_3 \wedge V_2)) \rightarrow V_1)$$

gegeben.

- (a) Beweisen Sie, dass $\varphi \in \text{AL}$.
- (b) Geben Sie $\text{Var}(\varphi)$ an.

- (c) Berechnen Sie für die Belegung $\mathcal{B} : \text{Var}(\varphi) \rightarrow \{0, 1\}$ mit $\mathcal{B}(V_1) = 0$, $\mathcal{B}(V_2) = 1$ und $\mathcal{B}(V_3) = 0$ den Wert $\llbracket \varphi \rrbracket^{\mathcal{B}}$ in nachvollziehbaren Schritten analog zu Beispiel (3.10) aus der Vorlesung.
- (d) Geben Sie den Syntaxbaum der Formel φ an.
- (e) Geben Sie die ASCII-Darstellung der Formel φ an.
- (f) Eine aussagenlogische Formel φ heißt erfüllbar, falls es (mindestens) eine zu φ passende Belegung $\mathcal{B} : \text{Var}(\varphi) \rightarrow \{0, 1\}$ gibt, so dass $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$. Ist die Formel φ erfüllbar?

Aufgabe 3.13. Geben Sie für jede der folgenden aussagenlogischen Formeln eine Wahrheitstafel und alle erfüllenden Belegungen $\mathcal{B} : \text{Var}(\varphi_1) \rightarrow \{0, 1\}$ (für (a)) bzw. $\mathcal{B} : \text{Var}(\varphi_2) \rightarrow \{0, 1\}$ (für (b)) an.

(a) $\varphi_1 := \left((V_1 \leftrightarrow \neg V_2) \wedge (\neg V_2 \vee V_3) \right) \wedge (V_3 \rightarrow V_1)$

(b) $\varphi_2 := \left((V_1 \leftrightarrow \mathbf{1}) \wedge (V_1 \rightarrow (V_2 \wedge \mathbf{0})) \right)$

Aufgabe 3.14.

- (a) Geben Sie für jede der folgenden aussagenlogischen Formeln an, ob sie erfüllbar, unerfüllbar und/oder allgemeingültig ist.

- $(V_0 \vee \mathbf{0})$
- $(V_0 \wedge \neg V_1)$
- $(V_0 \leftrightarrow (\mathbf{1} \rightarrow V_0))$
- $(V_0 \leftrightarrow (V_0 \rightarrow \mathbf{0}))$
- $(V_1 \vee ((V_0 \wedge V_1) \rightarrow V_2))$
- $((V_0 \rightarrow V_1) \leftrightarrow (\neg V_1 \rightarrow \neg V_0))$

- (b) Für jedes $n \in \mathbb{N}$ sei die aussagenlogische Formel φ_n definiert durch

$$\varphi_n := \begin{cases} (V_n \vee V_{n+1}), & \text{falls } n \text{ gerade} \\ (V_n \rightarrow \neg V_{n+1}), & \text{falls } n \text{ ungerade.} \end{cases}$$

Es gilt also

$$\varphi_0 = (V_0 \vee V_1), \quad \varphi_1 = (V_1 \rightarrow \neg V_2), \quad \varphi_2 = (V_2 \vee V_3), \quad \varphi_3 = (V_3 \rightarrow \neg V_4), \quad \dots$$

Geben Sie eine Belegung \mathcal{B} an, so dass für alle $n \in \mathbb{N}$ gilt: \mathcal{B} erfüllt φ_n .

- (c) Beweisen oder widerlegen Sie die folgende Behauptung:

$$((\neg V_0 \vee V_2) \wedge (V_1 \rightarrow \neg V_2)) \models \neg((V_0 \wedge \neg V_1) \rightarrow \neg(V_0 \rightarrow V_2))$$

- (d) Zeigen Sie, dass jede aussagenlogische Formel φ semantisch äquivalent zu unendlich vielen verschiedenen aussagenlogischen Formeln ist.

Aufgabe 3.15.

- (a) Geben Sie für jede der folgenden aussagenlogischen Formeln an, ob sie erfüllbar, unerfüllbar und/oder allgemeingültig ist.

- $\neg V_1$
- $((V_0 \vee \neg V_1) \leftrightarrow V_2)$
- $(\neg V_0 \rightarrow (V_0 \rightarrow V_1))$
- $(V_0 \wedge (V_0 \rightarrow \neg V_0))$
- $((V_0 \rightarrow V_1) \leftrightarrow ((V_0 \wedge \neg V_1) \rightarrow \mathbf{0}))$
- $\psi_n := \bigwedge_{i=1}^n (V_i \leftrightarrow V_{2i})$
für $n \in \mathbb{N}$ mit $n \geq 2$

(b) Für jedes $n \in \mathbb{N}$ sei die aussagenlogische Formel φ_n definiert durch

$$\varphi_n := \begin{cases} (V_n \leftrightarrow V_{n+2}), & \text{falls } n \text{ gerade} \\ (V_n \leftrightarrow \neg V_{n-1}), & \text{falls } n \text{ ungerade.} \end{cases}$$

Es gilt also

$$\varphi_0 = (V_0 \leftrightarrow V_2), \quad \varphi_1 = (V_1 \leftrightarrow \neg V_0), \quad \varphi_2 = (V_2 \leftrightarrow V_4), \quad \varphi_3 = (V_3 \leftrightarrow \neg V_2), \quad \dots$$

Geben Sie eine Belegung $\mathcal{B}: \text{AVAR} \rightarrow \{0, 1\}$ an, so dass für alle $n \in \mathbb{N}$ gilt: \mathcal{B} erfüllt φ_n .

(c) Beweisen oder widerlegen Sie die folgenden Behauptungen:

- (i) $(\neg(V_0 \leftrightarrow V_1) \wedge (\neg V_2 \vee V_0)) \models (V_0 \vee (V_1 \wedge \neg V_2))$
- (ii) $(\neg(V_0 \leftrightarrow V_1) \wedge (\neg V_2 \vee V_0)) \equiv (V_0 \vee (V_1 \wedge \neg V_2))$

Aufgabe 3.16. Beweisen Sie Beobachtung 3.26 (b) und (d), d.h. beweisen Sie, dass für alle aussagenlogischen Formeln φ und ψ gilt:

- (a) $\varphi \models \mathbf{0} \iff \varphi$ ist unerfüllbar.
- (b) $\varphi \models \psi \iff (\varphi \wedge \neg\psi)$ ist unerfüllbar.

Aufgabe 3.17. Betrachten Sie die folgenden beiden Aussagen:

- (1) Wenn der Rechner einen Virus hat oder nicht mehr funktioniert, und wenn der Administrator erreichbar ist, dann rufen wir den Administrator.
 - (2) Wenn der Rechner einen Virus hat, so rufen wir den Administrator, falls wir ihn erreichen; und wenn der Administrator erreichbar ist und der Rechner nicht funktioniert, so rufen wir den Administrator.
- (a) Formalisieren Sie jede der beiden Aussagen (1), (2) durch eine aussagenlogische Formel.
 - (b) Zeigen Sie, dass die beiden Aussagen (1) und (2) äquivalent sind.

Aufgabe 3.18. Einer Ihrer Bekannten berichtet von seiner Zimmersuche in Frankfurt und äußert Ihnen gegenüber folgende Aussagen, die auf alle der von ihm besichtigten Wohnungen zutreffen:

- Wenn es sich um eine 1-Zimmer-Wohnung handelt, dann stehen höchstens 26 m^2 Wohnraum zur Verfügung oder der Mietpreis ist höher als 400 € .
- Wenn sich das Zimmer nicht in einer 1-Zimmer-Wohnung befindet, dann ist das Zimmer in einer WG.

- Wenn mehr als 26 m^2 Wohnraum zur Verfügung stehen, dann liegt das Zimmer nicht in einer WG.
 - Wenn mehr als 26 m^2 Wohnraum zur Verfügung stehen und der Mietpreis höher als 400 € ist, dann handelt es sich nicht um eine 1-Zimmer-Wohnung.
- (a) Zerlegen Sie den obigen Text in atomare Aussagen und geben Sie eine aussagenlogische Formel φ an, die das im Text zusammengefasste Wissen repräsentiert.

Betrachten Sie nun die nachfolgenden Aussagen:

- In jeder besichtigten Wohnung stehen Ihrem Bekannten maximal 26 m^2 zur Verfügung.
 - Für jede besichtigte Wohnung gilt: Wenn die Wohnung in einer WG liegt, dann beträgt der Mietpreis höchstens 400 € .
 - Für jede besichtigte Wohnung gilt: Wenn der verlangte Mietpreis höchstens 400 € beträgt, dann handelt es sich um eine WG oder um eine 1-Zimmer-Wohnung.
- (b) Geben Sie für jede der drei Aussagen eine aussagenlogische Formel an, die die Aussage repräsentiert.
- (c) Entscheiden Sie für jede der aussagenlogischen Formeln aus (b), ob sie aus der Formel φ in (a) folgt.

Aufgabe 3.19. Weihnachten ist nicht mehr weit und Sie haben noch kein Geschenk für Ihre Eltern. Sie wollen nicht direkt nach den Wünschen Ihrer Eltern fragen und müssen daher auf deren Andeutungen achten. Diese Andeutungen ergeben folgende Aussagen:

- Wenn das Geschenk Konzertkarten sind, dann sollen diese nicht eingepackt sein.
 - Wenn das Geschenk teuer ist und es Konzertkarten sind, dann soll das Geschenk keine Überraschung sein.
 - Das Geschenk soll genau dann eingepackt sein, wenn es eine Überraschung ist.
 - Wenn das Geschenk nicht eingepackt ist, dann soll es teuer sein oder aus Konzertkarten bestehen.
- (a) Zerlegen Sie den obigen Text in atomare Aussagen und geben Sie eine aussagenlogische Formel φ an, die alle Andeutungen Ihrer Eltern repräsentiert.

Betrachten Sie nun die nachfolgenden Aussagen:

- Das Geschenk ist teuer.
 - Wenn das Geschenk eingepackt ist, dann sind es keine Konzertkarten.
 - Wenn das Geschenk eine teure Überraschung ist, dann besteht es nicht aus eingepackten Konzertkarten.
- (b) Geben Sie für jede der drei Aussagen eine aussagenlogische Formel an, die die Aussage repräsentiert.

- (c) Entscheiden Sie für jede der drei aussagenlogischen Formeln aus (b), ob sie aus der Formel φ in (a) folgt.

Aufgabe 3.20. Um sich von der Konkurrenz abzuheben, hat das Sandwich-Restaurant *Downroad* sich auf die Belieferung von Logikern spezialisiert. Allerdings geben diese Kunden ihre Bestellungen teilweise auf recht eigenwillige Weise ab. Um zu testen, ob neue Stellenbewerber bei Downroad mit dieser Art von Bestellungen umgehen können, wird ein Test bei jedem der Bewerber durchgeführt. Stellen Sie sich vor, Sie bewerben sich auf einen Job bei Downroad und müssen folgenden Test lösen:

Ein Kunde gibt seine Bestellung in Form der folgenden Bedingungen auf:

- Wenn das Sandwich-Brot nicht getoastet ist, soll zusätzlich ein Cookie mitgeliefert werden.
 - Wenn das Sandwich keinen Extra-Käse enthält, soll kein Cookie mitgeliefert werden.
 - Wenn das Sandwich-Brot getoastet ist, so enthält das Sandwich Extra-Käse oder keine Jalapeños.
 - Wenn das Sandwich keinen Extra-Käse und keine Jalapeños enthält, so ist es auch nicht getoastet.
- (a) Zerlegen Sie den obigen Text in atomare Aussagen und geben Sie eine aussagenlogische Formel φ an, die alle Äußerungen des Kunden repräsentiert.

Betrachten Sie nun die nachfolgenden Aussagen:

- Das Sandwich enthält Extra-Käse.
 - Es wird genau dann ein Cookie mitgeliefert, wenn auch Jalapeños auf dem Sandwich sind.
 - Wenn Jalapeños auf dem Sandwich sind, dann wird zusätzlich ein Cookie mitgeliefert oder das Sandwich-Brot ist getoastet.
 - Das Sandwich enthält Extra-Käse und zusätzlich gilt, dass ein Cookie mitgeliefert wird oder das Sandwich getoastet ist.
- (b) Geben Sie für jede der vier Aussagen eine aussagenlogische Formel an, die die Aussage repräsentiert.
- (c) Entscheiden Sie für jede der vier aussagenlogischen Formeln aus (b), ob sie aus der Formel φ in (a) folgt und ob sie semantisch äquivalent dazu ist.

Aufgabe 3.21. Es sei $\varphi := ((V_0 \vee \neg V_2) \rightarrow V_1)$.

- (a) Wandeln Sie φ mittels Wahrheitstafel in eine äquivalente aussagenlogische Formel in DNF um.
- (b) Wenden Sie Algorithmus 3.39 an, um eine zu φ äquivalente Formel in KNF zu finden.

Aufgabe 3.22. Die Menge AL_{pos} sei die Menge der Wörter über dem Alphabet $A = \text{AVAR} \cup \{\mathbf{1}, \mathbf{0}, \wedge, \vee, (,), \}$, die rekursiv wie folgt definiert ist:

- Basisregel:* (B) Für jede Variable $X \in \text{AVAR}$ gilt: $X \in \text{AL}_{pos}$.
- Rekursive Regeln:* (R1) Ist $\varphi \in \text{AL}_{pos}$, so ist auch $(\varphi \vee \mathbf{0}) \in \text{AL}_{pos}$.
- (R2) Ist $\varphi \in \text{AL}_{pos}$, so ist auch $(\varphi \wedge \mathbf{1}) \in \text{AL}_{pos}$.
- (R3) Sind $\varphi \in \text{AL}_{pos}$ und $\psi \in \text{AL}_{pos}$, so ist auch $(\varphi \vee \psi) \in \text{AL}_{pos}$.
- (R4) Sind $\varphi \in \text{AL}_{pos}$ und $\psi \in \text{AL}_{pos}$, so ist auch $(\varphi \wedge \psi) \in \text{AL}_{pos}$.

Offenbar ist jedes Wort der Sprache AL_{pos} eine aussagenlogische Formel, es gilt also $\text{AL}_{pos} \subseteq \text{AL}$. Zeigen Sie durch Induktion über den Aufbau von AL_{pos} , dass jede Formel $\varphi \in \text{AL}_{pos}$ erfüllbar ist.

Aufgabe 3.23. Für Belegungen $\mathcal{B}, \mathcal{B}'$ mit $\text{Def}(\mathcal{B}) = \text{Def}(\mathcal{B}') = \text{AVAR}$ schreiben wir $\mathcal{B} \preceq \mathcal{B}'$, wenn für alle $X \in \text{AVAR}$ gilt:

$$\mathcal{B}(X) \leq \mathcal{B}'(X)$$

(d.h.: wenn $\mathcal{B}(X) = 1$ ist, so ist auch $\mathcal{B}'(X) = 1$).

Eine aussagenlogische Formel φ heißt *monoton*, falls für alle zu φ passenden Belegungen \mathcal{B} und \mathcal{B}' mit $\mathcal{B} \preceq \mathcal{B}'$ gilt:

$$\text{Falls } \llbracket \varphi \rrbracket^{\mathcal{B}} = 1, \text{ so } \llbracket \varphi \rrbracket^{\mathcal{B}'} = 1.$$

- (a) Geben Sie je ein Beispiel für eine monotone und für eine nicht monotone aussagenlogische Formel an.

Die Menge AL_+ ist die Teilmenge aller aussagenlogischen Formeln, in denen keines der Symbole $\neg, \rightarrow, \leftrightarrow$ vorkommt.

- (b) Geben Sie eine exakte rekursive Definition der Formelmenge AL_+ an.
- (c) Beweisen Sie, dass alle Formeln $\varphi \in \text{AL}_+$ monoton sind.
- (d) Existiert eine monotone aussagenlogische Formel $\varphi \in \text{AL}$, in der Negationszeichen vorkommen?
- (e) Existiert eine monotone aussagenlogische Formel $\varphi \in \text{AL}$, in der genau ein Negationszeichen vorkommt?

Aufgabe 3.24. Betrachten Sie die aussagenlogische Formel

$$\varphi := (\neg(V_0 \leftrightarrow V_1) \wedge (\neg V_2 \vee V_0)).$$

- (a) Wandeln Sie φ mittels Wahrheitstafel in eine äquivalente aussagenlogische Formel in DNF um.
- (b) Wenden Sie Algorithmus 3.39 an, um eine zu φ äquivalente Formel in KNF zu finden.

Aufgabe 3.25. Für jedes $n \in \mathbb{N}_{>0}$ sei die aussagenlogische Formel φ_n definiert durch

$$\varphi_n := \bigwedge_{i=1}^n (X_i \leftrightarrow Y_i).$$

- (a) Beschreiben Sie die erfüllenden Belegungen $\mathcal{B}: \text{Var}(\varphi_n) \rightarrow \{0, 1\}$ für φ_n . Wie viele solche Belegungen gibt es?
- (b) Geben Sie eine zu φ_n äquivalente Formel in DNF an.

- (c) Beweisen Sie Satz 3.44, d.h. zeigen Sie, dass jede zu φ_n äquivalente Formel in DNF mindestens 2^n konjunktive Klauseln hat.

Hinweis: Eine Möglichkeit, dies zu zeigen, ist einen Beweis durch Widerspruch zu führen. Nehmen Sie dazu an, dass ψ_n eine zu φ_n äquivalente Formel in DNF ist, die aus weniger als 2^n konjunktiven Klauseln besteht. D.h. es gibt eine natürliche Zahl $N < 2^n$ und N konjunktive Klauseln $\kappa_1, \dots, \kappa_N$, so dass $\psi_n = \kappa_1 \vee \dots \vee \kappa_N$. Folgern Sie aus Ihrer Antwort aus Teil (a), dass mindestens eine der Klauseln $\kappa_1, \dots, \kappa_N$ von mindestens zwei verschiedenen die Formel φ_n erfüllenden Belegungen wahr gemacht wird. Leiten Sie daraus einen Widerspruch her.

4 Graphen und Bäume

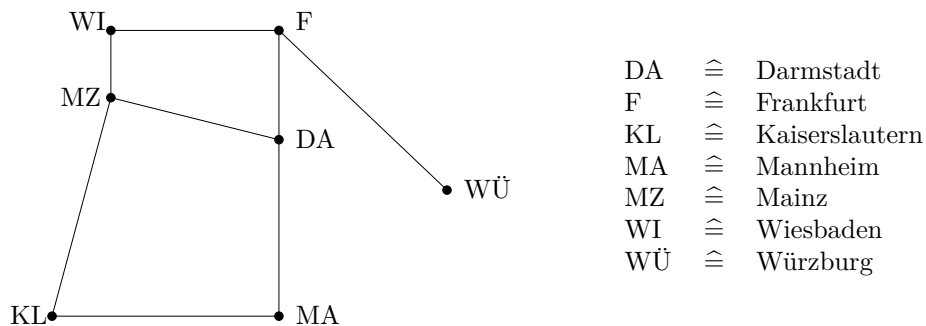
Bei Modellierungsaufgaben geht es oft darum, **Objekte** sowie **Beziehungen** zwischen Objekten zu beschreiben. Graphen und Bäume eignen sich dazu oft besonders gut. Anschaulich besteht ein Graph aus **Knoten** und **Kanten**:

- “Knoten” repräsentieren dabei “gleichartige Objekte”.
- “Kanten” repräsentieren Beziehungen zwischen je zwei “Objekten”.

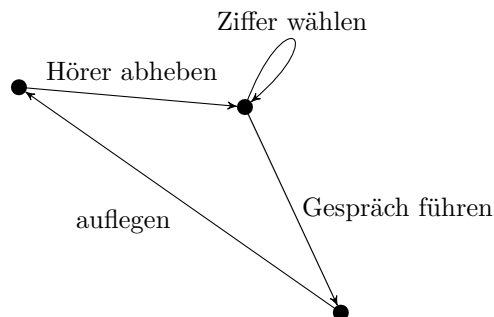
Je nach Aufgabenstellung werden **ungerichtete** Graphen oder **gerichtete** Graphen verwendet. Bäume sind Graphen mit bestimmten Eigenschaften.

Beispiel 4.1.

- (a) Skizze eines **ungerichteten** Graphen, der die Autobahnverbindungen zwischen einigen Städten darstellt:



- (b) Skizze eines **gerichteten** Graphen, der den prinzipiellen Ablauf eines Telefonats darstellt:



4.1 Graphen

4.1.1 Grundlegende Definitionen

Definition 4.2 (ungerichteter Graph).

ungerichteter
Graph

Ein **ungerichteter Graph** $G = (V, E)$ besteht aus einer Menge V , die **Knotenmenge** von G genannt wird, und einer Menge

$$E \subseteq \{\{i, j\} : i \in V, j \in V, i \neq j\},$$

Knoten
Kanten

die **Kantenmenge** von G genannt wird. Die Elemente aus V heißen **Knoten** von G (auch: "Ecken"; englisch: **vertices**, singular: vertex); die Elemente aus E heißen **Kanten** von G (englisch: **edges**, singular: edge).

endlich

Ein ungerichteter Graph G heißt **endlich**, falls seine Knotenmenge endlich ist.

Beispiel 4.3. $G = (V, E)$ mit

$$V := \{\text{MZ, WI, MA, DA, KL, F, WÜ}\} \text{ und}$$

$$E := \{\{\text{MZ, WI}\}, \{\text{WI, F}\}, \{\text{F, DA}\}, \{\text{F, WÜ}\}, \{\text{MZ, DA}\}, \{\text{MZ, KL}\}, \{\text{KL, MA}\}, \{\text{DA, MA}\}\}$$

ist ein ungerichteter Graph, der die Autobahnverbindungen zwischen Mainz (MZ), Wiesbaden (WI), Mannheim (MA), Darmstadt (DA), Kaiserslautern (KL), Frankfurt (F) und Würzburg (WÜ) repräsentiert.

Beispiel 4.1(a) zeigt diesen Graphen G in **graphischer Darstellung**: Knoten werden als Punkte dargestellt, Kanten als Verbindungslinien zwischen Punkten.

Beachte: Laut Definition 4.2 gibt es zwischen zwei Knoten i und j aus V

- **höchstens** eine Kante; diese wird mit $\{i, j\}$ bezeichnet und graphisch dargestellt als



- **keine** Kante, falls $i = j$ ist. In der graphischen Darstellung eines ungerichteten Graphs sind also "Schleifen" der Form



nicht erlaubt.

Jede Kante $\{i, j\}$ eines ungerichteten Graphen ist also eine 2-elementige Menge von Knoten des Graphen.

Bemerkung: In der Literatur wird zumeist die oben genannte Definition von ungerichteten Graphen verwendet. Davon abweichend erlauben einige Bücher in ungerichteten Graphen aber auch "Schleifen" der Form



Notation 4.4. Sei $G = (V, E)$ ein ungerichteter Graph.

- Ein Knoten $v \in V$ heißt **inzident** mit einer Kante $e \in E$, falls $v \in e$. inzident
- Die beiden mit einer Kante $e \in E$ inzidenten Knoten nennen wir die **Endknoten** von e , und wir sagen, dass e diese beiden Knoten **verbindet**. Endknoten
- Zwei Knoten $v, v' \in V$ heißen **benachbart** (bzw. **adjazent**), falls es eine Kante $e \in E$ gibt, deren Endknoten v und v' sind (d.h. $e = \{v, v'\}$). benachbart
adjazent
- Falls v und v' zwei benachbarte Knoten sind, so sagen wir auch: v' ist ein **Nachbar** von Knoten v . Nachbar

Definition 4.5 (Grad).

Sei $G = (V, E)$ ein endlicher ungerichteter Graph und sei $v \in V$ ein Knoten von G . Der **Grad von v in G** (engl.: degree), kurz: $\text{Grad}_G(v)$, ist die Anzahl der Kanten, die v als Endknoten haben. D.h.

$$\text{Grad}_G(v) = |\{e \in E : v \in e\}|.$$

Grad
 $\text{Grad}_G(v)$

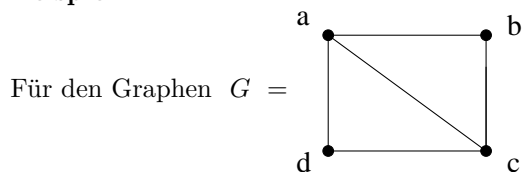
Der **Grad von G** ist

$$\text{Grad}(G) := \max \{ \text{Grad}_G(v) : v \in V \},$$

$\text{Grad}(G)$

d.h. $\text{Grad}(G)$ gibt den maximalen Grad eines Knotens von G an.¹

Beispiel:



gilt: $\text{Grad}_G(a) = 3$
 $\text{Grad}_G(b) = 2$
 $\text{Grad}_G(c) = 3$
 $\text{Grad}_G(d) = 2$ und $\text{Grad}(G) = 3$.

Definition 4.6 (gerichteter Graph).

Ein **gerichteter Graph** $G = (V, E)$ besteht aus einer Menge V , die **Knotenmenge** von G genannt wird, und einer Menge

gerichteter Graph

$$E \subseteq \{(i, j) : i \in V, j \in V\},$$

die **Kantenmenge** von G genannt wird. Die Elemente aus V heißen **Knoten** (bzw. "Ecken"), die Elemente aus E heißen (gerichtete) **Kanten** von G .

Knoten
(gerichtete) Kante

Ein gerichteter Graph G heißt **endlich**, falls seine Knotenmenge endlich ist.

endlich

Beispiel 4.7. $G = (V, E)$ mit

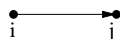
$$V := \{a, b, c\} \text{ und}$$

$$E := \{(a, b), (b, b), (b, c), (c, a), (a, c)\}$$

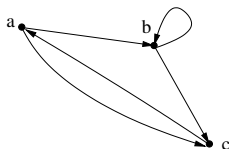
ist ein gerichteter Graph.

¹Ist M eine endliche, nicht-leere Menge von Zahlen, so bezeichnet $\max M$ das größte Element von M .

In der **graphischen Darstellung** eines gerichteten Graphen werden Knoten als Punkte dargestellt. Eine Kante der Form (i, j) wird als Pfeil von Knoten i nach Knoten j dargestellt, also



Der gerichtete Graph aus Beispiel 4.7 lässt sich in graphischer Darstellung also wie folgt darstellen:



Notation 4.8. Sei $G = (V, E)$ ein gerichteter Graph.

Ausgangsknoten
Endknoten

- Ist $e = (i, j) \in E$, so heißt i der **Ausgangsknoten** von e und j der **Endknoten** von e , und wir sagen, dass e **von i nach j verläuft**.

inzident

- Ein Knoten $v \in V$ heißt **inzident** mit einer Kante $e \in E$, falls v der Ausgangs- oder der Endknoten von e ist.

benachbart
adjazent

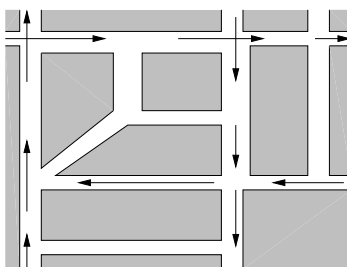
- Zwei Knoten $v, v' \in V$ heißen **benachbart** (bzw. **adjazent**), falls $(v, v') \in E$ oder $(v', v) \in E$.

Schleife

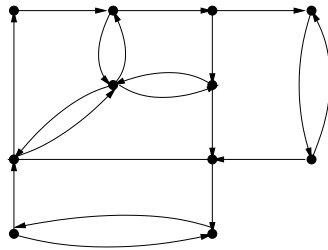
- Eine Kante der Form (v, v) wird **Schleife** genannt. D.h.: Eine Schleife ist eine Kante, deren Ausgangs- und Endknoten identisch ist.

Beispiel 4.9 (Modellierung durch gerichtete Graphen).

In der folgenden Straßenkarte sind Einbahnstraßen durch Pfeile markiert.



Diese Straßenkarte können wir durch einen gerichteten Graphen repräsentieren, der für jede Straßenkreuzung einen Knoten enthält, und in dem es eine Kante von "Kreuzung" i zu "Kreuzung" j gibt, falls man von i nach j fahren kann, ohne zwischendurch eine weitere Kreuzung zu passieren. Graphisch lässt sich dieser gerichtete Graph folgendermaßen darstellen:



Weitere Beispiele zur Modellierung durch Graphen:

- Computer-Netzwerk:
Knoten repräsentieren Computer; Kanten repräsentieren Netzwerkverbindungen
- das World Wide Web:
Knoten repräsentieren Webseiten; Kanten repräsentieren Hyperlinks
(Details dazu finden sich in Kapitel 5.)

Definition 4.10.

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

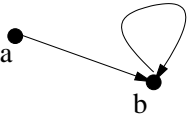
- Der **Ausgangsgrad** von v in G (engl.: out-degree), kurz: $\text{Aus-Grad}_G(v)$, ist die Anzahl der Kanten, die v als Ausgangsknoten haben. D.h.: Ausgangsgrad
Aus-Grad $_G(v)$

$$\text{Aus-Grad}_G(v) = |\{e \in E : \text{es ex. } v' \in V \text{ s.d. } e = (v, v')\}|.$$

- Der **Eingangsgrad** von v in G (engl.: in-degree), kurz: $\text{Ein-Grad}_G(v)$, ist die Anzahl der Kanten, die v als Eingangsknoten haben. D.h.: Eingangsgrad
Ein-Grad $_G(v)$

$$\text{Ein-Grad}_G(v) = |\{e \in E : \text{es ex. } v' \in V \text{ s.d. } e = (v', v)\}|.$$

Beispiel:

Für den Graphen $G =$  gilt:

Ein-Grad $_G(a) = 0$
Ein-Grad $_G(b) = 2$
Aus-Grad $_G(a) = 1$
Aus-Grad $_G(b) = 1$

Bemerkung 4.11 (Verschiedene Arten der Darstellung von Graphen).

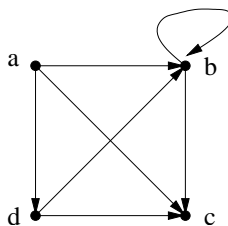
Es gibt mehrere Arten Graphen darzustellen, zum Beispiel

- **abstrakt**, durch Angabe der Knotenmenge V und der Kantenmenge E .

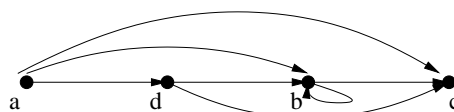
Beispiel: $G_1 = (V_1, E_1)$ mit

$$V_1 := \{a, b, c, d\} \quad \text{und} \quad E_1 := \{(a, b), (a, c), (a, d), (b, b), (b, c), (d, b), (d, c)\}.$$

- **graphisch** (bzw. **anschaulich**): Der obige Beispiel-Graph G_1 kann graphisch dargestellt werden durch



oder, äquivalent dazu, durch

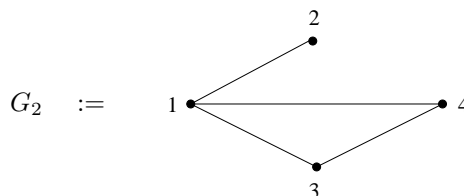


Adjazenzliste

- durch Angabe einer **Adjazenzliste**, die zu jedem Knoten i eine Liste aller Knoten angibt, zu denen eine von i ausgehende Kante führt. Der Beispiel-Graph G_1 wird durch folgende Adjazenzliste repräsentiert:

Knoten	Nachfolger
a	b, c, d
b	b, c
c	
d	b, c

Auf die gleiche Art können auch **ungerichtete** Graphen durch eine Adjazenzliste repräsentiert werden. Beispielweise der Graph



durch die Adjazenzliste

Knoten	Nachbarn
1	2, 3, 4
2	1
3	1, 4
4	1, 3

Adjazenzmatrix

- durch Angabe einer **Adjazenzmatrix**, d.h. einer Tabelle, deren Zeilen und Spalten mit Knoten beschriftet sind, und die in der mit Knoten i beschrifteten Zeile und der mit Knoten j beschrifteten Spalte

- den Eintrag 1 hat, falls es eine Kante von Knoten i nach Knoten j gibt, und
- den Eintrag 0 hat, falls es keine Kante von i nach j gibt.

Beispielsweise sieht die Adjazenzmatrix des gerichteten Graphen G_1 wie folgt aus:

	a	b	c	d
a	0	1	1	1
b	0	1	1	0
c	0	0	0	0
d	0	1	1	0

Die Adjazenzmatrix des ungerichteten Graphen G_2 ist:

	1	2	3	4
1	0	1	1	1
2	1	0	0	0
3	1	0	0	1
4	1	0	1	0

4.1.2 Wege in Graphen

Definition 4.12 (Wege und Kreise).

Sei $G = (V, E)$ ein (gerichteter oder ungerichteter) Graph.

(a) Ein **Weg** in G ist ein Tupel

$$(v_0, \dots, v_\ell) \in V^{\ell+1},$$

Weg

für ein $\ell \in \mathbb{N}$, so dass für alle $i \in \mathbb{N}$ mit $0 \leq i < \ell$ gilt:

- falls G ein gerichteter Graph ist, so ist $(v_i, v_{i+1}) \in E$,
- falls G ein ungerichteter Graph ist, so ist $\{v_i, v_{i+1}\} \in E$.

Das Tupel (v_0, \dots, v_ℓ) wird dann **ein Weg von v_0 nach v_ℓ** genannt. ℓ ist die **Länge des Weges**. D.h.: Die **Länge** des Weges gibt gerade an, wie viele **Kanten** auf dem Weg durchlaufen werden.

Weglänge

Beachte:

Gemäß dieser Definition ist für jedes $v \in V$ das Tupel (v) ein Weg der Länge 0 von v nach v .

(b) Ein Weg (v_0, \dots, v_ℓ) heißt **einfach**, wenn kein Knoten mehr als einmal in dem Weg vorkommt (d.h. die Knoten v_0, \dots, v_ℓ sind paarweise verschieden, d.h. $|\{v_0, \dots, v_\ell\}| = \ell + 1$).

einfacher Weg

(c) Ein Weg (v_0, \dots, v_ℓ) heißt **Kreis**, wenn $\ell \geq 1$ und $v_\ell = v_0$ ist.

Kreis

(d) Ein Kreis (v_0, \dots, v_ℓ) heißt **einfach**, wenn keine Kante mehrfach durchlaufen wird und — abgesehen vom Start- und Endknoten — kein Knoten mehrfach besucht wird. D.h.:

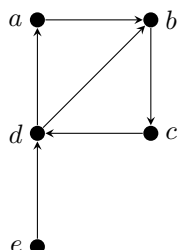
einfacher Kreis

- In einem **gerichteten** Graphen G sind **einfache** Kreise genau die Wege der Form (v_0, \dots, v_ℓ) , für die gilt: $\ell \geq 1$ und $v_\ell = v_0$ und $|\{v_0, \dots, v_{\ell-1}\}| = \ell$.

- In einem **ungerichteten** Graphen G sind **einfache** Kreise genau die Wege der Form (v_0, \dots, v_ℓ) , für die gilt: $\ell \geq 3$ und $v_\ell = v_0$ und $|\{v_0, \dots, v_{\ell-1}\}| = \ell$.

Beispiel 4.13.

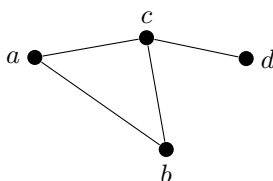
(a) Für den Graphen



gilt:

- (e, d, b, c, d) ist ein Weg der Länge 4, aber kein einfacher Weg.
- (d, b, c, d) ist ein einfacher Kreis.
- (e, d, a, b) ist ein einfacher Weg.
- (b, d, a) ist kein Weg.
- (a, b, c, d, b, c, d, a) ist ein Kreis, aber kein einfacher Kreis.

(b) Für den Graphen



gilt:

- (a, b, c, a) ist ein einfacher Kreis.
- (c, d, c) ist ein Kreis, aber kein einfacher Kreis.
- (a, c, d) ist ein einfacher Weg.
- (c, b, a, c, d) ist ein Weg, aber kein einfacher Weg.

Definition 4.14 (azyklischer Graph, DAG).

azyklisch

(a) Ein Graph heißt **azyklisch**, falls er keinen einfachen Kreis enthält.

DAG

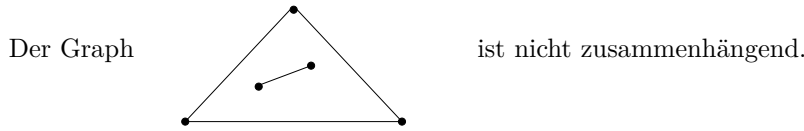
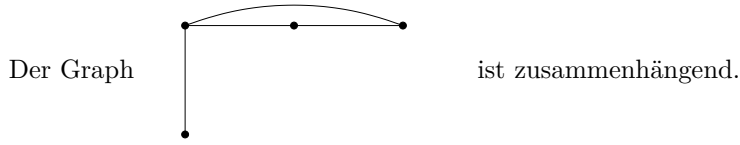
(b) Gerichtete azyklische Graphen werden im Englischen **directed acyclic graph**, kurz: DAG, genannt.

Definition 4.15 (zusammenhängend, stark zusammenhängend).

zusammenhängend

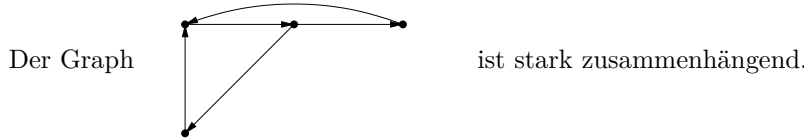
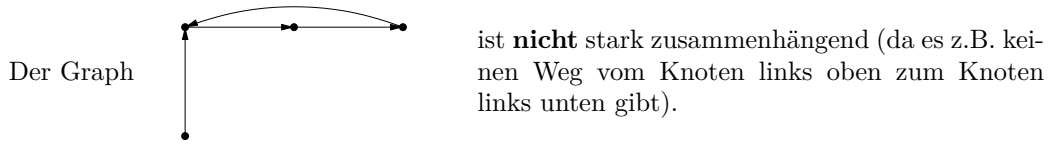
(a) Ein ungerichteter Graph $G = (V, E)$ heißt **zusammenhängend**, wenn für alle Knoten $v, w \in V$ gilt: Es gibt in G einen Weg von v nach w .

Beispiel:



- (b) Ein gerichteter Graph $G = (V, E)$ heißt **stark zusammenhängend**, wenn für alle Knoten $v, w \in V$ gilt: Es gibt in G einen Weg von v nach w . stark zusammenhängend

Beispiel:

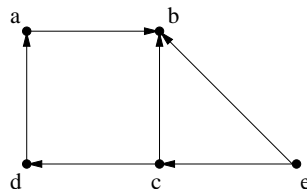


Definition 4.16 (Hamilton-Kreise und Hamilton-Wege).

Sei $G = (V, E)$ ein (gerichteter oder ein ungerichteter) Graph.

- (a) Ein Weg $W = (v_0, \dots, v_\ell)$ heißt **Hamilton-Weg**, wenn jeder **Knoten** aus V genau einmal in W vorkommt. Hamilton-Weg
- (b) Ein Weg $W = (v_0, \dots, v_\ell)$ heißt **Hamilton-Kreis**, wenn $\ell \geq 1$ und $v_\ell = v_0$ und $(v_0, \dots, v_{\ell-1})$ ein Hamilton-Weg ist. Hamilton-Kreis

Beispiel: Der Graph G



hat einen Hamilton-Weg, nämlich (e, c, d, a, b) , aber keinen Hamilton-Kreis (da $\text{Aus-Grad}_G(b) = 0$ ist).

Ein Anwendungsbeispiel:

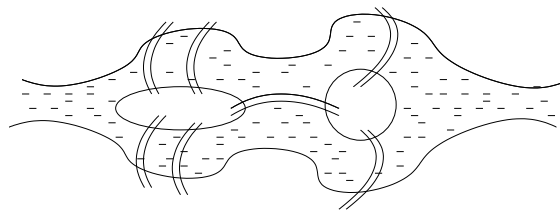
Beim Problem des **Handlungsreisenden** (engl.: **Travelling Salesman Problem**, kurz: **TSP**) Handlungsreisendenproblem

geht es darum, eine Rundreise durch n Städte so durchzuführen, dass jede Stadt genau 1 mal besucht wird. Es geht also darum, einen Hamilton-Kreis zu finden. Das Problem, zu einem gegebenen Graphen zu entscheiden, ob er einen Hamilton-Kreis besitzt, ist algorithmisch ein schwieriges Problem: Man kann zeigen, dass es (genau wie das auf Seite 84 betrachtete aussagenlogische Erfüllbarkeitsproblem) NP-vollständig ist.

Im Gegensatz zu Hamilton-Wegen (bei denen es darum geht, einen Weg zu finden, der jeden **Knoten** des Graphen genau einmal besucht), geht es bei den im Folgenden betrachteten **Euler-Wegen** darum, einen Weg zu finden, der jede **Kante** des Graphen genau einmal besucht.

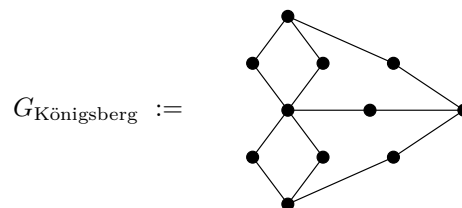
Beispiel 4.17 (Königsberger Brückenproblem).

In der Stadt Königsberg gab es im 18. Jahrhundert 7 Brücken über den Fluss Pregel, die die Ufer und 2 Inseln auf die in der folgenden Skizze dargestellten Art miteinander verbanden.



Frage: Gibt es einen Spaziergang, der jede der 7 Brücken genau einmal überquert und zum Ausgangspunkt zurückkehrt?

Die obige Skizze lässt sich folgendermaßen durch einen ungerichteten Graphen modellieren: für jedes Ufer, jede Insel und jede Brücke gibt es einen Knoten; Kanten zeigen direkte Verbindungen an. Die Skizze wird also durch folgenden Graphen repräsentiert:



Die Frage nach dem ‘‘Spaziergang’’ entspricht dann gerade der Frage:

Gibt es in $G_{\text{Königsberg}}$ einen Euler-Kreis?

Definition 4.18 (Euler-Kreise und Euler-Wege).

Sei $G = (V, E)$ ein ungerichteter Graph.

Euler-Weg

(a) Ein Weg $W = (v_0, \dots, v_\ell)$ heißt **Euler-Weg**, wenn W jede Kante aus E genau einmal durchläuft, d.h. wenn es für jedes $e \in E$ genau ein $i \in \{0, \dots, \ell-1\}$ gibt, so dass $e = \{v_i, v_{i+1}\}$.

Euler-Kreis

(b) Ein Weg $W = (v_0, \dots, v_\ell)$ heißt **Euler-Kreis**, wenn W ein Euler-Weg ist und $v_0 = v_\ell$ ist.

Satz 4.19 (Existenz von Euler-Kreisen und Euler-Wegen).

Sei $G = (V, E)$ ein ungerichteter, zusammenhängender Graph, dessen Knotenmenge endlich ist. Dann gilt:

- (a) G besitzt einen Euler-Kreis \iff jeder Knoten von G hat einen geraden Grad (d.h. ist mit einer geraden Anzahl von Kanten inzident).
- (b) G besitzt einen Euler-Weg, \iff es gibt in G genau zwei Knoten mit ungeradem Grad. der kein Euler-Kreis ist

Beweis:

- (a) " \implies ": Sei $K = (v_0, \dots, v_\ell)$ ein Euler-Kreis. Insbesondere gilt: $v_0 = v_\ell$.

Schritt 1: Jeder Knoten $v \in \{v_0, \dots, v_{\ell-1}\}$ hat geraden Grad, denn:

Sei $v \in \{v_0, \dots, v_{\ell-1}\}$ beliebig. Zu jedem $i \in \{0, \dots, \ell-1\}$ mit $v = v_i$ gibt es im Euler-Kreis K zwei verschiedene Kanten, nämlich

- $\{v_{i-1}, v_i\}$ und $\{v_i, v_{i+1}\}$, falls $i \neq 0$, bzw.
- $\{v_0, v_1\}$ und $\{v_{\ell-1}, v_0\}$, falls $i = 0$ (beachte: $v_0 = v_\ell$).

Da der Euler-Kreis K jede Kante von G genau einmal enthält, gilt somit folgendes: Ist $k = |\{i \in \{0, \dots, \ell-1\} : v = v_i\}|$ (d.h. k gibt an, wie oft v im Tupel $(v_0, \dots, v_{\ell-1})$ vorkommt), so ist $\text{Grad}_G(v) = 2 \cdot k$. Daher hat jeder Knoten $v \in \{v_0, \dots, v_{\ell-1}\}$ geraden Grad.

Schritt 2: $\{v_0, \dots, v_{\ell-1}\} = V$, denn:

Laut Voraussetzung ist G zusammenhängend. Für beliebige Knoten $v, w \in V$ gilt daher: es gibt in G einen Weg von v nach w . Da K ein Euler-Kreis ist, enthält K sämtliche Kanten, die auf dem Weg von v nach w vorkommen. Insbesondere gilt also f.a. $v, w \in V$, dass $v, w \in \{v_0, \dots, v_{\ell-1}\}$.

Zusammenfassung: Aus den Schritten 1 und 2 folgt direkt, dass jeder Knoten von G geraden Grad hat.

" \impliedby ": Sei G ein zusammenhängender ungerichteter Graph, in dem jeder Knoten geraden Grad hat. Es sei

$$W = (v_0, \dots, v_\ell)$$

ein Weg **maximaler Länge** in G , der **keine Kante(n) mehrfach** enthält. Da wir W nicht mehr verlängern können, liegen alle mit v_ℓ inzidenten Kanten auf W . Da laut unserer Voraussetzung die Anzahl dieser Kanten gerade ist, folgt $v_\ell = v_0$.

Zu zeigen: W ist ein Euler-Kreis.

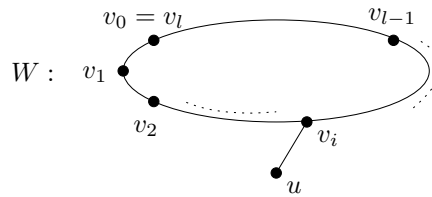
Angenommen, W ist **kein** Euler-Kreis.

Dann gibt es in G eine Kante e' , die nicht auf W liegt. Da G zusammenhängend ist, gibt es einen Weg, der von einem Endknoten von e' zu einem zu W gehörenden Knoten führt. Sei e die erste Kante auf diesem Weg, die einen Endpunkt in W hat. Sei v_i der zu e inzidente Knoten aus W und sei $u \in V$ der andere zu e inzidente Knoten, d.h. $e = \{u, v_i\}$. Dann ist der Weg

$$W' := (u, v_i, v_{i+1}, \dots, v_{\ell-1}, v_0, v_1, \dots, v_i)$$

ein Weg der Länge $\ell + 1$, der keine Kante(n) mehrfach enthält.

Skizze:



Dies widerspricht aber der Tatsache, dass W ein Weg **maximaler** Länge ist.

- (b) Die Richtung " \implies " folgt analog zu (a): Sei $K = (v_0, \dots, v_\ell)$ ein Euler-Weg, der kein Euler-Kreis ist. Man sieht leicht, dass die beiden Endknoten von K ungeraden Grad haben, und dass alle anderen Knoten geraden Grad haben.

Zum Beweis der Richtung " \impliedby " kann man (a) verwenden: Seien x und y die beiden Knoten von G , die ungeraden Grad haben. Wir betrachten den Graphen $G' := (V', E')$ mit $V' := V \cup \{z\}$ und $E' := E \cup \{\{x, z\}, \{y, z\}\}$, wobei z ein "neuer" Knoten ist, der nicht zu V gehört.

Offensichtlich hat jeder Knoten in G' geraden Grad. Außerdem ist G' zusammenhängend (da G zusammenhängend ist). Aus (a) folgt, dass G' einen Euler-Kreis besitzt. Wegen $\text{Grad}_{G'}(z) = 2$ wird z auf diesem Kreis genau einmal besucht. Durch Entfernen der Kanten $\{x, z\}$ und $\{z, y\}$ erhält man einen Euler-Weg in G , der die beiden Knoten x und y als Anfangs- und Endpunkt hat. \square

Beispiel 4.20 (Lösung des Königsberger Brückenproblems).

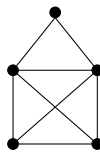
Mit Hilfe von Satz 4.19 können wir das Königsberger Brückenproblem aus Beispiel 4.17 leicht lösen: Es gibt **keinen** Spaziergang, der jede der 7 Brücken genau einmal überquert und zum Ausgangspunkt zurückkehrt.

Beweis: Ein solcher Spaziergang würde gerade einem Euler-Kreis im Graphen $G_{\text{Königsberg}}$ entsprechen. Dieser Graph besitzt aber 4 Knoten von ungeradem Grad und kann daher laut Satz 4.19(a) keinen Euler-Kreis besitzen. \square

Beispiel 4.21.

Unter Verwendung von Satz 4.19 kann man auch die folgende Frage leicht lösen.

Frage: Kann man die Figur



in einem Zug nachzeichnen? D.h: Besitzt dieser Graph einen Euler-Weg?

Unter Verwendung von Satz 4.19 kann man die Frage leicht beantworten, indem man nachzählt, wie viele Knoten von ungeradem Grad es gibt. Im obigen Graphen gibt es genau 2 Knoten von ungeradem Grad. Gemäß Satz 4.19 besitzt G also einen Euler-Weg, der kein Euler-Kreis ist.

4.1.3 Ähnlichkeit zweier Graphen

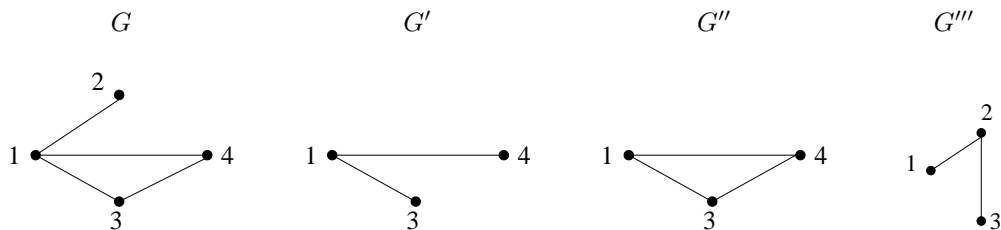
Die folgende Definition formalisiert, wann ein Graph G' in einem Graphen G "enthalten" ist.

Definition 4.22 (Teilgraph).

Seien $G = (V, E)$ und $G' = (V', E')$ zwei (gerichtete oder ungerichtete) Graphen.

- (a) G' heißt **Teilgraph von G** , falls $V' \subseteq V$ und $E' \subseteq E$. Teilgraph
- (b) Sei $W \subseteq V$. Der von W induzierte Teilgraph von G ist der Graph $G|_W$ mit Knotenmenge W und Kantenmenge $E|_W := \{e \in E : \text{alle mit } e \text{ inzidenten Knoten liegen in } W\}$. $G|_W$
- (c) $G' = (V', E')$ heißt **induzierter Teilgraph von G** , falls $V' \subseteq V$ und $G' = G|_{V'}$. induzierter Teilgraph

Beispiel 4.23. Wir betrachten die folgenden Graphen:



Dann ist

- G' ein Teilgraph von G , aber kein induzierter Teilgraph von G .
- G'' ein induzierter Teilgraph von G .
- G''' kein Teilgraph von G .

Definition 4.24 (Gleichheit von Graphen).

Zwei Graphen $G = (V, E)$ und $G' = (V', E')$ sind **gleich** (kurz: $G = G'$), falls sie dieselbe Knotenmenge und dieselbe Kantenmenge besitzen. D.h.: $G = G'$

$$G = G' \iff V = V' \text{ und } E = E'.$$

Beispielsweise sind die beiden Graphen



nicht gleich, da sie unterschiedliche Knotenmengen besitzen. Intuitiv sind die beiden Graphen aber "prinzipiell gleich" (Fachbegriff: **isomorph**, kurz: $G \cong G'$), da der zweite Graph aus dem ersten durch Umbenennung der Knoten entsteht. Der Begriff der Isomorphie wird durch die folgende Definition präzisiert:

isomorph
 $G \cong G'$

Definition 4.25 (Isomorphie von Graphen).

Seien $G = (V, E)$ und $G' = (V', E')$ zwei (gerichtete oder ungerichtete) Graphen. G und G' heißen **isomorph** (kurz: $G \cong G'$, in Worten: G ist isomorph zu G'), falls es eine bijektive Abbildung $f : V \rightarrow V'$ gibt, so dass für alle Knoten $i \in V$ und $j \in V$ gilt:

- falls G und G' gerichtet sind:

$$(i, j) \in E \iff (f(i), f(j)) \in E'$$

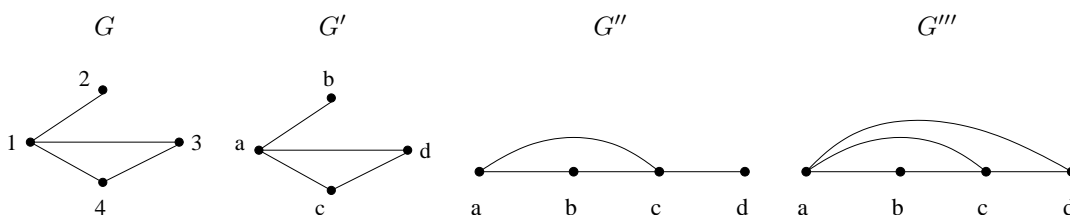
- falls G und G' ungerichtet sind:

$$\{i, j\} \in E \iff \{f(i), f(j)\} \in E'$$

Isomorphismus

Eine solche Abbildung f wird **Isomorphismus von G nach G'** genannt.

Beispiel 4.26. Es seien:



Dann gilt:

- $G \cong G'$ via $f: \{1, 2, 3, 4\} \rightarrow \{a, b, c, d\}$ mit $f(1) = a, f(2) = b, f(3) = d, f(4) = c$.
- $G \cong G''$ via $f: \{1, 2, 3, 4\} \rightarrow \{a, b, c, d\}$ mit $f(1) = c, f(2) = d, f(3) = a, f(4) = b$.
- G'' ist nicht isomorph zu G''' , kurz: $G'' \not\cong G'''$, da G''' **mehr** Kanten als G'' hat.

4.1.4 Markierte Graphen

Bemerkung 4.27.

Viele Modellierungsaufgaben erfordern, dass den Knoten oder den Kanten eines Graphen weitere Informationen zugeordnet werden. Dies wird durch so genannte **Markierungsfunktionen** für Knoten oder Kanten formalisiert:

Knoten-
markierung

- (a) Eine **Knotenmarkierung** eines (gerichteten oder ungerichteten) Graphen $G = (V, E)$ ist eine Abbildung

$$m: V \rightarrow W,$$

wobei W ein geeigneter Wertebereich ist. In dem Graph aus Beispiel 4.1(a) könnte man beispielweise eine Knotenmarkierung Einwohnerzahl: $V \rightarrow \mathbb{N}$ einführen, die jedem Knoten die Einwohnerzahl der zugehörigen Stadt zuordnet.

Kanten-
markierung

- (b) Eine **Kantenmarkierung** von G ist eine Abbildung

$$m: E \rightarrow W,$$

wobei W ein geeigneter Wertebereich ist. In dem Graph aus Beispiel 4.1(a) könnte man beispielweise eine Kantenmarkierung Entfernung: $E \rightarrow \mathbb{N}$ einführen, die jeder Kante die Länge (in km) des von der Kante repräsentierten Autobahnteilstücks zuordnet.

Kantenmarkierungen kann man auch dazu verwenden, um auszudrücken, dass es zwischen zwei Knoten mehr als eine Kante gibt. Die Markierungsfunktion gibt dann an, für wie viele Verbindungen die eine Kante des Graphen steht:

Definition 4.28 (Multigraph).

Ein **Multigraph** (G, m) besteht aus einem (gerichteten oder ungerichteten) Graphen $G = (V, E)$ und einer Kantenmarkierung $m: E \rightarrow \mathbb{N}$. Multigraph

Beispiel: Sei $G = (V, E)$ der Graph mit $V = \{a, b, c\}$ und $E = \{\{a, b\}, \{b, c\}, \{c, a\}\}$. Sei $m: E \rightarrow \mathbb{N}$ mit $m(\{a, b\}) = 1$, $m(\{b, c\}) = 1$ und $m(\{c, a\}) = 2$. Dann ist (G, m) ein Multigraph, der graphisch wie folgt dargestellt werden kann:



4.1.5 Zuordnungsprobleme

Wir betrachten zunächst zwei typische Beispiele von Zuordnungsproblemen.

Beispiel 4.29. (a) In einem Tennisverein sollen die Vereinsmitglieder für ein Turnier zu Doppelpaarungen zusammengestellt werden. Dabei möchte man jeweils nur befreundete Personen als ‘Doppel’ zusammen spielen lassen.

Um diese Aufgabe zu lösen, modellieren wir die Situation durch den ungerichteten Graphen $G_T := (V_T, E_T)$ mit

$$V_T := \{x : x \text{ ist ein Vereinsmitglied}\}$$

$$E_T := \{\{x, y\} : x \text{ und } y \text{ sind befreundete Vereinsmitglieder}\}.$$

Das **Ziel** ist, eine größtmögliche Anzahl von Doppelpaarungen zu finden. D.h., wir wollen eine möglichst große Menge $E' \subseteq E_T$ finden, so dass kein Vereinsmitglied Endpunkt von mehr als einer Kante aus E' ist.

(b) Eine Gruppe unterschiedlich ausgebildeter Piloten soll so auf Flugzeuge verteilt werden, dass jeder das ihm zugeteilte Flugzeug fliegen kann.

Auch hier modellieren wir die Situation durch einen ungerichteten Graphen $G_F := (V_F, E_F)$ mit

$$V_F := \{x : x \text{ ist ein Pilot}\} \cup \{y : y \text{ ist ein Flugzeug}\},$$

$$E_F := \{\{x, y\} : \text{Pilot } x \text{ kann Flugzeug } y \text{ fliegen}\}.$$

Das **Ziel** ist, einen Flugplan aufzustellen, so dass jeder Pilot das ihm zugeteilte Flugzeug fliegen kann. D.h.: Wir wollen eine möglichst große Menge $E' \subseteq E_F$ finden, so dass kein Element aus V_F Endpunkt von mehr als einer Kante in E' ist.

Matching
 Paarung

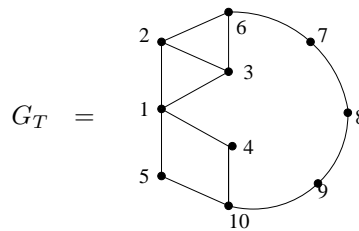
Die gesuchten Kantenmengen E' aus (a) und (b) werden **Matching** (bzw. **Paarung** oder **Menge unabhängiger Kanten**) genannt:

Definition 4.30.

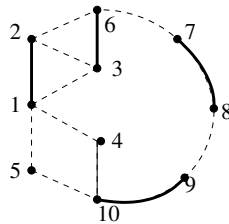
Sei $G = (V, E)$ ein ungerichteter Graph. Eine Kantenmenge $E' \subseteq E$ heißt **Matching** (bzw. **Paarung** bzw. **Menge unabhängiger Kanten**), falls kein Knoten aus V Endpunkt von mehr als einer Kante aus E' ist.

Ziel in Beispiel 4.29 (a) und (b) ist es, ein Matching maximaler Größe zu finden, d.h. ein Matching, das so viele Kanten wie möglich enthält.

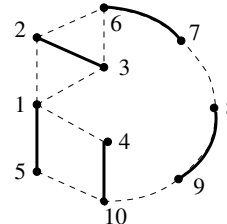
Beispiel 4.31. In einem Tennisverein mit 10 Mitgliedern und "Freundschaftsgraph"



sind z.B. die folgenden beiden Kantenmengen Matchings:



und



$$E' = \{\{1, 2\}, \{3, 6\}, \{7, 8\}, \{9, 10\}\}$$

$$E'' = \{\{1, 5\}, \{4, 10\}, \{8, 9\}, \{6, 7\}, \{2, 3\}\}.$$

In Beispiel 4.29(b) sollten Piloten auf Flugzeuge verteilt werden. Die Knotenmenge des zugehörigen Graphen G_F bestand aus zwei verschiedenen Arten von Objekten (nämlich einerseits Piloten und andererseits Flugzeuge), und Kanten konnten jeweils nur zwischen Objekten unterschiedlicher Art verlaufen (also zwischen Piloten und Flugzeugen, nicht aber zwischen Piloten und Piloten bzw. Flugzeugen und Flugzeugen). Solche Graphen werden *bipartite Graphen* genannt:

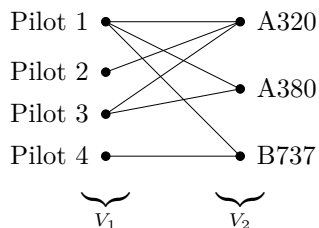
bipartit

Definition 4.32 (bipartiter Graph).

Ein ungerichteter Graph $G = (V, E)$ heißt **bipartit**, wenn seine Knotenmenge V so in zwei disjunkte Teilmengen V_1 und V_2 zerlegt werden kann (d.h. $V = V_1 \dot{\cup} V_2$), dass jede Kante aus E einen Endknoten in V_1 und einen Endknoten in V_2 hat.

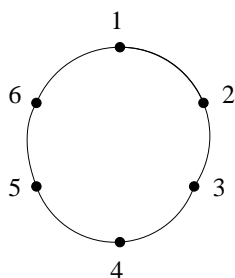
Beispiel 4.33.

(a) Der Graph

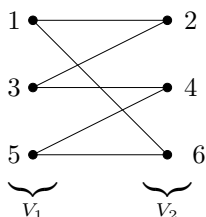


ist bipartit mit $V_1 = \{\text{Pilot 1, Pilot 2, Pilot 3, Pilot 4}\}$ und $V_2 = \{\text{A320, A380, B737}\}$.

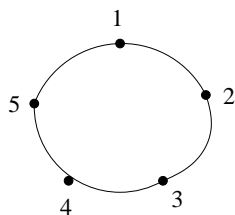
(b) Der Graph



ist bipartit mit $V_1 = \{1, 3, 5\}$ und $V_2 = \{2, 4, 6\}$. Der Graph lässt sich auch wie folgt graphisch darstellen:



(c) Der Graph



ist **nicht** bipartit.

Beweis: Durch Widerspruch. Angenommen, er ist doch bipartit. Dann seien V_1 und V_2 die beiden disjunkten Teilmengen der Knotenmenge, so dass jede Kante des Graphen einen Endknoten in V_1 und einen Endknoten in V_2 hat. Wir können ohne Beschränkung der Allgemeinheit annehmen, dass $1 \in V_1$ ist (falls nicht, vertauschen wir einfach V_1 und V_2). Dann muss aber gelten: $2 \in V_2$, $3 \in V_1$, $4 \in V_2$ und $5 \in V_1$, also $V_1 = \{1, 3, 5\}$ und $V_2 = \{2, 4\}$. Im Graphen gibt es aber auch eine Kante zwischen 1 und 5, und beide Knoten gehören zu V_1 . Dies ist ein Widerspruch zu der Annahme, dass jede Kante einen Endpunkt in V_1 und einen Endpunkt in V_2 hat. \square

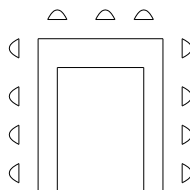
Allgemein gilt: Ist $n \in \mathbb{N}_{>0}$ und ist G ein Kreis auf n Knoten (wie in (b) für $n = 6$ und in (c) für $n = 5$), so gilt:

$$G \text{ ist bipartit} \iff n \text{ ist gerade.}$$

Wir betrachten ein weiteres typisches Beispiel für ein Zuordnungsproblem:

Beispiel 4.34 (Sitzordnung bei einer Familienfeier).

Die Gäste einer Familienfeier sollen so an einer hufeisenförmigen Tafel



platziert werden, dass niemand neben jemanden sitzt, den er nicht leiden kann.

Lösungsansatz:

Schritt 1: Stelle den **Konfliktgraphen** $G = (V, E)$ auf, wobei

$$V := \{x : \text{Person } x \text{ soll zur Feier kommen}\} \text{ und}$$

$$E := \left\{ \{x, y\} : \begin{array}{l} \text{Person } x \text{ kann Person } y \text{ nicht leiden oder} \\ \text{Person } y \text{ kann Person } x \text{ nicht leiden} \end{array} \right\}$$

d.h. Kanten im Konfliktgraphen zeigen auf, wer im Konflikt mit wem steht.

Schritt 2: Bilde das Komplement des Konfliktgraphen, d.h. betrachte den Graphen $\tilde{G} = (\tilde{V}, \tilde{E})$ mit

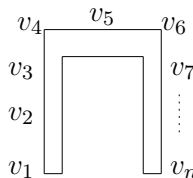
$$\tilde{V} := V \text{ und}$$

$$\tilde{E} := \{\{x, y\} : x, y \in V, x \neq y, \{x, y\} \notin E\},$$

d.h. Kanten in \tilde{G} zeigen an, wer prinzipiell neben wem platziert werden könnte.

Schritt 3: Suche einen Hamilton-Weg in \tilde{G} .

Wenn (v_1, \dots, v_n) (mit $n = |\tilde{V}|$) ein Hamilton-Weg in \tilde{G} ist, dann kann man die Sitzordnung folgendermaßen festlegen:



Falls es in \tilde{G} keinen Hamilton-Weg gibt, so weiß man, dass es **keine Möglichkeit gibt**, die geladenen Gäste so an einer hufeisenförmigen Tafel zu platzieren, dass niemand neben jemandem sitzt, den er nicht leiden kann.

Ein möglicher Ausweg ist, die Gäste an **mehrere** Tische zu verteilen. Dies kann wie folgt modelliert werden:

Beispiel 4.35 (Sitzordnung bei einer Familienfeier, Teil 2).

Die Gäste einer Familienfeier sollen so an mehreren Tischen platziert werden, dass Personen, die sich nicht leiden können, an verschiedenen Tischen sitzen. Dabei sollen so wenig Tische wie möglich verwendet werden.

Diese Aufgabe kann folgendermaßen modelliert werden: Die verfügbaren Tische werden mit den Zahlen $1, 2, 3, \dots$ durchnummeriert. Die geladenen Gäste und die herrschenden Konflikte zwischen Gästen werden durch den in Beispiel 4.34 betrachteten Konfliktgraphen $G = (V, E)$ repräsentiert. Die Zuordnung, wer an welchem Tisch sitzen soll, wird durch eine Knotenmarkierung $m: V \rightarrow \mathbb{N}_{>0}$ repräsentiert, wobei $m(x) = i$ bedeutet, dass Person x am Tisch i sitzen soll.

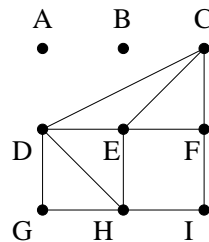
Das **Ziel** ist, eine **konfliktfreie Knotenmarkierung** $m: V \rightarrow \mathbb{N}_{>0}$ zu finden. Dabei soll $|\text{Bild}(m)|$ möglichst klein sein — dies entspricht dem Ziel, die Gäste an möglichst wenige Tische zu verteilen.

Definition 4.36 (konfliktfreie Knotenmarkierung).

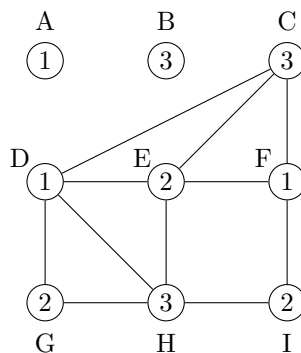
Sei $G = (V, E)$ ein ungerichteter Graph. Eine Funktion $m: V \rightarrow \mathbb{N}$ heißt **konfliktfreie Knotenmarkierung** (bzw. konfliktfreie **Färbung**), wenn für jede Kante $\{x, y\} \in E$ gilt: $m(x) \neq m(y)$.

konfliktfreie
Knotenmarkierung

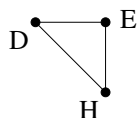
Beispiel 4.37. Um Beispiel 4.35 fortzuführen, betrachten wir eine Familienfeier mit Gästen A, B, C, D, E, F, G, H, I und folgendem Konfliktgraphen:



Die folgende Graphik gibt eine konfliktfreie Knotenmarkierung $m: V \rightarrow \mathbb{N}$ an, wobei für jeden Knoten $v \in V$ der Wert $m(v)$ in den Kreis geschrieben ist, der den Knoten v repräsentiert.



Für die hier gegebene Markierung m gilt $|\text{Bild}(m)| = 3$, die Gäste werden also an 3 Tische verteilt. Dies ist optimal, da der Konfliktgraph ein Dreieck, z.B.



als Teilgraph enthält – deshalb muss für jede konfliktfreie Knotenmarkierung m' gelten: $|\text{Bild}(m')| \geq 3$.

Bemerkung 4.38 (4-Farben-Problem).

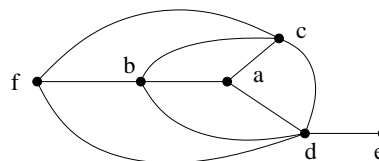
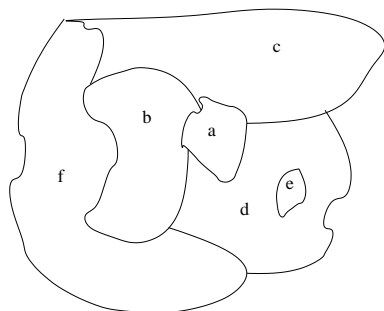
4-Farben-Problem

Ein sehr bekannter Vertreter dieser Art von Markierungs- oder Färbungsaufgaben ist das so genannte **4-Farben-Problem**. Dabei handelt es sich um die Frage, wie viele verschiedene Farben nötig sind, um jede Landkarte so einzufärben, dass zwei Staaten, die ein Stück gemeinsamer Grenze haben, durch unterschiedliche Farben dargestellt werden. 1976 wurde bewiesen, dass vier Farben ausreichen. Der Beweis basiert auf einer Fallunterscheidung mit mehr als 1000 Fällen, die mit Hilfe eines Computerprogramms analysiert wurden.

Das Problem, eine Landkarte einzufärben, kann durch einen ungerichteten Graphen modelliert werden, dessen Knoten gerade die Staaten repräsentieren, und bei dem es eine Kante zwischen zwei Staaten gibt, falls diese eine gemeinsame Grenze besitzen. Ziel ist, eine konfliktfreie Knotenmarkierung m zu finden, bei der $|\text{Bild}(m)|$ so klein wie möglich ist.

Beispiel:

Wir betrachten eine kleine Landkarte und den zugehörigen Konfliktgraphen:



Knoten $\hat{=}$ Staaten
Kanten $\hat{=}$ Staaten mit gemeinsamer Grenze

Da bei den vier Knoten a, b, c, d paarweise jeder zu jedem benachbart ist, muss eine konfliktfreie Färbung diesen vier Knoten vier verschiedene Farben zuordnen — für a, b, c, d etwa *rot, gelb, grün, blau*. Da f außerdem mit b, c, d benachbart ist, muss f dann wieder *rot* gefärbt sein; e kann jede Farbe außer *blau* erhalten.

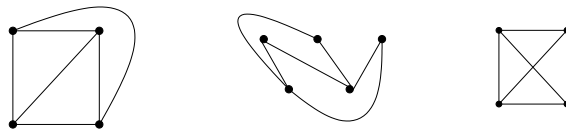
Die aus Landkarten entstehenden Konfliktgraphen haben eine besondere Eigenschaft: Sie sind **planar**.

Definition 4.39 (planare Graphen).

planar

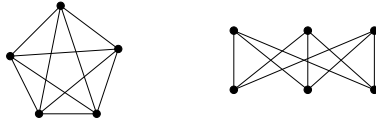
Ein Graph G heißt **planar**, wenn er so in die Ebene gezeichnet werden kann, dass seine Kanten sich nicht kreuzen.

Beispiele für planare Graphen sind:



(Der dritte Graph ist planar, da er wie der erste Graph kreuzungsfrei in die Ebene gezeichnet werden kann.)

Beispiele für nicht-planare Graphen sind:



Bemerkung 4.40 (chromatische Zahl).

Die Anzahl verschiedener ‘Farben’ bzw. ‘Markierungen’, die nötig sind, um einen Graphen $G = (V, E)$ konfliktfrei zu färben (bzw. zu markieren), nennt man die **chromatische Zahl**, kurz: $\chi(G)$. Die präzise Definition ist²

chromatische Zahl $\chi(G)$

$$\chi(G) := \min\{ |\text{Bild}(m)| : m: V \rightarrow \mathbb{N} \text{ ist eine konfliktfreie Knotenmarkierung für } G \}.$$

Weitere Beispiele für Zuordnungsprobleme, die durch Graphen modelliert werden können, finden sich in den Aufgaben 4.12, 4.13 und 4.15.

4.2 Bäume

Eine für die Informatik besonders wichtige Art von Graphen sind die so genannten **Bäume**. Wir betrachten im Folgenden zunächst ungerichtete Bäume und danach gerichtete Bäume.

4.2.1 Ungerichtete Bäume

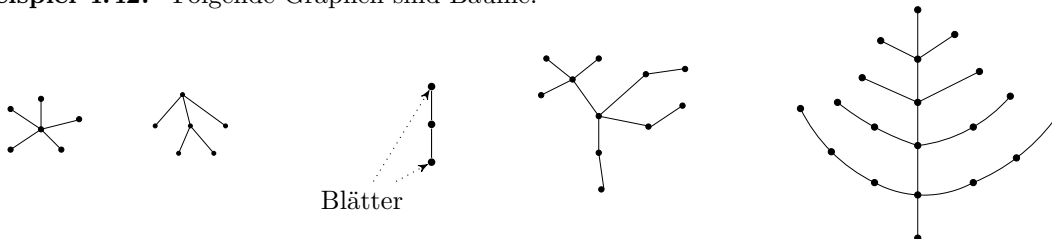
Definition 4.41 (ungerichteter Baum).

Ein **ungerichteter Baum** ist ein ungerichteter, zusammenhängender Graph $G = (V, E)$, der keinen einfachen Kreis enthält.

ungerichteter Baum
Blätter

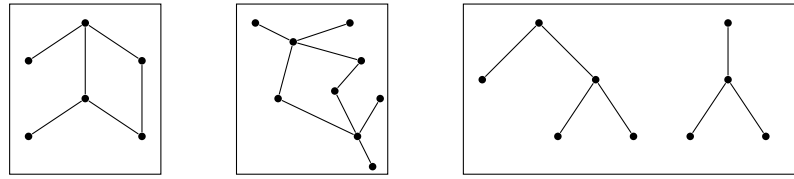
Diejenigen Knoten in V , die den Grad ≤ 1 haben, heißen **Blätter** des Baums.

Beispiel 4.42. Folgende Graphen sind Bäume:



Folgende Graphen sind keine Bäume:

²Ist M eine endliche, nicht-leere Menge von Zahlen, so bezeichnet $\min M$ das kleinste Element von M .

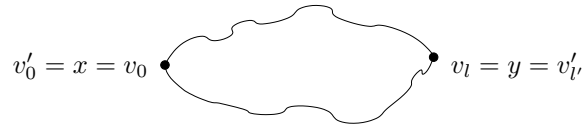


Beobachtung 4.43. Ist $B = (V, E)$ ein ungerichteter Baum, so gilt für alle Knoten $x, y \in V$:

Es gibt in B genau einen einfachen Weg von x nach y .

Denn: B ist ein ungerichteter Baum, d.h. B ist zusammenhängend und enthält keinen einfachen Kreis. Da B zusammenhängend ist, gibt es mindestens einen einfachen Weg von x nach y . Angenommen, (v_0, \dots, v_ℓ) und $(v'_0, \dots, v'_{\ell'})$ sind zwei verschiedene einfache Wege von x nach y . Insbesondere gilt dann $v_0 = x = v'_0$ und $v_\ell = y = v'_{\ell'}$.

Skizze:



Dann ist aber $(v_0, \dots, v_\ell, v'_{\ell'-1}, \dots, v'_0)$ ein Kreis. Dieser Kreis enthält einen einfachen Kreis. Dann kann B aber kein Baum sein. Widerspruch. \square

Satz 4.44.

Jeder endliche ungerichtete Baum $B = (V, E)$ mit $V \neq \emptyset$ besitzt mindestens ein Blatt.

Beweis: Sei $B = (V, E)$ ein endlicher ungerichteter Baum mit $V \neq \emptyset$.

Sei $W := (v_0, \dots, v_\ell)$ ein einfacher Weg maximaler Länge in B . Dann ist $\ell \geq 0$, da $V \neq \emptyset$. Außerdem ist $\ell < |V|$, da W einfach ist (d.h. die in W vorkommenden Knoten v_0, \dots, v_ℓ sind paarweise verschieden — und in V gibt es nur $|V|$ viele verschiedene Knoten). Wir betrachten zwei Fälle:

Fall 1: $\ell = 0$.

Da B ein Baum ist (d.h. insbesondere zusammenhängend) und $W = (v_0)$ ein einfacher Weg maximaler Länge, muss dann $V = \{v_0\}$ und $E = \emptyset$ sein. Insbesondere ist v_0 ein Blatt von B .

Fall 2: $\ell \geq 1$.

Dann ist $v_{\ell-1}$ ein Nachbar von v_ℓ . Angenommen, v_ℓ ist kein Blatt. Dann hat v_ℓ einen weiteren Nachbarn, den wir im Folgenden u nennen.

Falls u nicht in W vorkommt, so ist (v_0, \dots, v_ℓ, u) ein einfacher Weg, der länger ist als W . Dies widerspricht unserer Wahl von W als einfachem Weg maximaler Länge.

Falls u in W vorkommt, so gibt es ein i mit $u = v_i$ und $0 \leq i < \ell - 1$ (da u ein von den Knoten v_ℓ und $v_{\ell-1}$ verschiedener Knoten ist). Dann ist $(v_i, \dots, v_{\ell-1}, v_\ell, v_i)$ ein einfacher Kreis in B . Dies widerspricht aber der Tatsache, dass B ein ungerichteter Baum ist.

Insgesamt kann es also keinen von $v_{\ell-1}$ verschiedenen Nachbarn u von v_ℓ geben. D.h. v_ℓ ist ein Blatt. \square

Der folgende Satz, für dessen Beweis wir Satz 4.44 benutzen, besagt, dass die Anzahl der Kanten eines ungerichteten Baums durch die Anzahl der Knoten genau festgelegt ist.

Satz 4.45 (Anzahl der Kanten eines Baums).

Für jeden endlichen ungerichteten Baum $B = (V, E)$, mit $V \neq \emptyset$ gilt: $|E| = |V| - 1$.

Baum:
 $|E| = |V| - 1$

Beweis: Per Induktion nach $n := |V|$.

INDUKTIONSANFANG: $n = 1$

Der einzige ungerichtete Baum $B = (V, E)$ mit $|V| = 1$ ist der Graph \bullet mit $E = \emptyset$. Für diesen Graphen gilt: $|E| = 0 = 1 - 1 = |V| - 1$.

INDUKTIONSSCHRITT: $n \rightarrow n + 1$

Sei $n \in \mathbb{N}$ mit $n \geq 1$ beliebig.

Induktionsannahme:

Für jeden ungerichteten Baum $B' = (V', E')$ mit $V' \neq \emptyset$ und $|V'| \leq n$ gilt: $|E'| = |V'| - 1$.

Behauptung:

Für jeden ungerichteten Baum $B = (V, E)$ mit $V \neq \emptyset$ und $|V| = n + 1$ gilt: $|E| = |V| - 1$.

Beweis: Sei $B = (V, E)$ ein ungerichteter Baum mit $|V| = n + 1$. Gemäß Satz 4.44 besitzt B (mindestens) ein Blatt, das wir im Folgenden u nennen. Da B zusammenhängend ist und $|V| = n + 1 \geq 2$ ist, besitzt u einen Nachbarn v in B — und da u ein Blatt ist, ist v der einzige Nachbar von u in B .

Sein nun B' der Graph, der aus B entsteht, indem wir den Knoten u und die von v zu u führende Kante löschen. D.h., $B' = (V', E')$ mit $V' := V \setminus \{u\}$ und $E' := E \setminus \{\{v, u\}\}$. Man sieht leicht, dass B' zusammenhängend ist und keinen einfachen Kreis enthält. D.h., B' ist ein Baum. Außerdem ist $|V'| = n$. Aus der Induktionsannahme folgt daher, dass $|E'| = |V'| - 1$ ist. Insgesamt gilt daher für den Baum B :

$$|E| = |E'| + 1 = (|V'| - 1) + 1 = |V'| = |V| - 1.$$

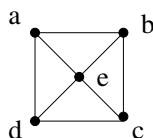
□

Bäume finden sich als Teilgraphen von zusammenhängenden Graphen. Besonders wichtig für die Informatik sind die so genannten **Spannbäume**.

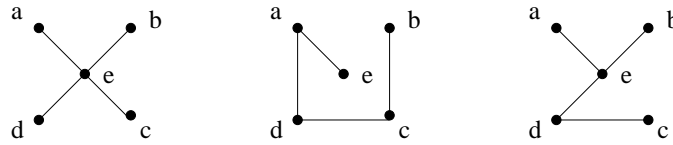
Definition 4.46 (Spannbaum).

Sei $G = (V, E)$ ein ungerichteter Graph. Ein Graph $G' = (V', E')$ heißt **Spannbaum von G** , falls G' ein ungerichteter Baum mit $V' = V$ und $E' \subseteq E$ ist. Spannbaum

Beispiel 4.47. Der Graph



hat u.a. folgende Spannbäume:



Jeder zusammenhängende Graph besitzt einen Spannbaum. Präzise gilt:

Satz 4.48. Sei $G = (V, E)$ ein ungerichteter Graph, dessen Knotenmenge endlich ist. Dann gilt:

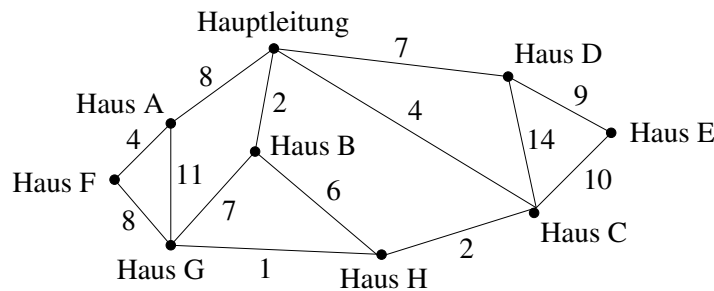
satz:Zusammenhang-vs-Spannbäume Es gibt (mindestens) einen Spannbaum von $G \iff G$ ist zusammenhängend.

Beweis: " \implies ": klar. " \impliedby ": Übung. □

Geht man von einem zusammenhängenden Graphen zu einem seiner Spannbäume über, so verkleinert man gemäß Satz 4.45 die Kantenmenge von $|E|$ auf $|V| - 1$ Kanten, ohne dabei den Zusammenhang des Graphen aufzugeben. Mit dem Begriff des Spannbäume wird also ein "bezüglich der Kantenzahl kostengünstigerer Zusammenhang" modelliert.

Manche konkreten Probleme lassen sich durch Graphen modellieren, deren Kanten mit bestimmten Werten markiert sind, so dass zur Lösung des Problems ein Spannbaum gesucht wird, bei dem die Summe seiner Kantenmarkierungen so klein wie möglich ist. Dazu betrachten wir das folgende Beispiel.

Beispiel 4.49 (Kabelfernsehen). Eine Firma will Leitungen zum Empfang von Kabelfernsehen in einem neuen Wohngebiet verlegen. Der folgende Graph skizziert das Wohngebiet:



Knoten entsprechen dabei einzelnen Häusern bzw. der Hauptleitung, die aus einem bereits verkabelten Gebiet heranzuführt. Eine Kante zwischen zwei Knoten zeigt an, dass es prinzipiell möglich ist, eine direkte Leitung zwischen den beiden Häusern zu verlegen. Der Wert, mit dem die Kante markiert ist, beschreibt, wie teuer (in 1000 €) es ist, diese Leitung zu verlegen.

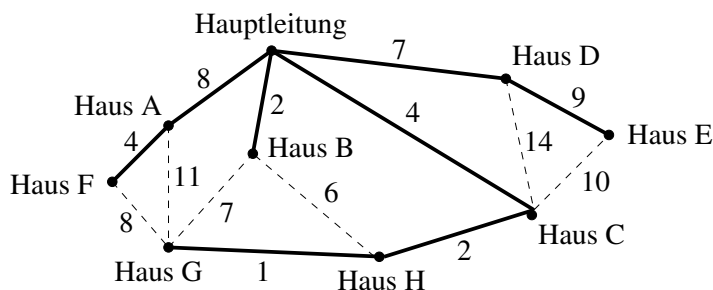
Ziel ist, Leitungen so zu verlegen, dass

- (1) jedes Haus ans Kabelfernsehen angeschlossen ist und
- (2) die Kosten für das Verlegen der Leitungen so gering wie möglich sind.

Es wird also ein Spannbaum gesucht, bei dem die Summe seiner Kantenmarkierungen so klein wie möglich ist. Ein solcher Spannbaum wird **minimaler Spannbaum** (engl.: **minimum spanning tree**) genannt.

minimaler Spannbaum

Die im Folgenden **fett** gezeichneten Kanten geben die Kanten eines minimalen Spannbaums an:



Verlegt die Firma genau diese Leitungen, so hat sie das neue Wohngebiet mit den geringstmöglichen Kosten ans Kabelfernsehen angeschlossen.

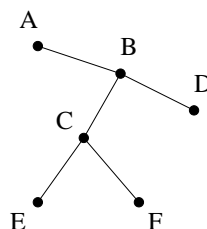
Bemerkung: Verfahren zum Finden minimaler Spann bäume werden Sie in der Vorlesung “Algorithmentheorie” (GL-1) kennenlernen.

4.2.2 Gerichtete Bäume

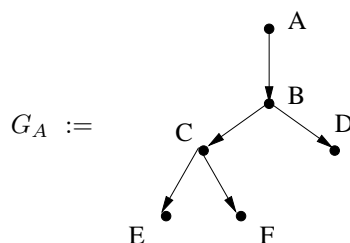
Einen **gerichteten Baum** erhält man, indem man in einem ungerichteten Baum einen Knoten als “Wurzel” auswählt und alle Kanten in die Richtung orientiert, die von der Wurzel weg führt.

gerichteten Baum

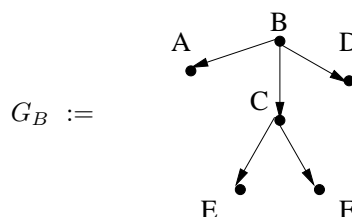
Beispiel 4.50. Ungerichteter Baum:



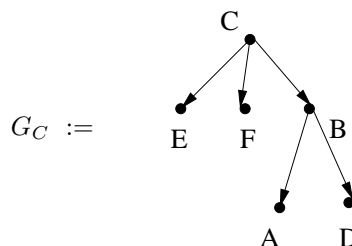
- Zugehöriger gerichteter Baum mit Wurzel A:



- Zugehöriger gerichteter Baum mit Wurzel B:



- Zugehöriger gerichteter Baum mit Wurzel C:



Die präzise Definition des Begriffs “gerichteter Baum” ist wie folgt:

Definition 4.51 (gerichteter Baum).

gerichteter Baum

Ein gerichteter Graph $G = (V, E)$ heißt **gerichteter Baum**, falls er folgende Eigenschaften hat:

Wurzel

- (1) G besitzt genau einen Knoten $w \in V$ mit $\text{Ein-Grad}_G(w) = 0$. Dieser Knoten wird **Wurzel** genannt.
- (2) Für jeden Knoten $v \in V$ gilt: Es gibt in G einen Weg von der Wurzel zum Knoten v .
- (3) Für jeden Knoten $v \in V$ gilt: $\text{Ein-Grad}_G(v) \leq 1$.

Definition 4.52 (Blätter, innere Knoten, Höhe).

Blätter

- (a) Sei $B = (V, E)$ ein gerichteter Baum. Diejenigen Knoten, deren Aus-Grad 0 ist, heißen **Blätter**.

Beispiel: In Beispiel 4.50 hat G_A die Blätter D, E, F. G_B hat die Blätter A, D, E, F und G_C die Blätter A, D, E, F.

innere Knoten

- (b) Diejenigen Knoten eines gerichteten Baums, die weder Wurzel noch Blätter sind, heißen **innere Knoten**.

Höhe
Tiefe

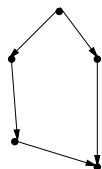
- (c) Sei $B = (V, E)$ ein gerichteter Baum. Die **Höhe** (bzw. **Tiefe**, engl.: height, depth) von B ist die Länge eines längsten Weges in B .

Beispiel: In Beispiel 4.50 hat G_A die Höhe 3, G_B die Höhe 2 und G_C die Höhe 2.

Beobachtung 4.53.

- (a) Jeder gerichtete Baum ist ein gerichteter azyklischer Graph (kurz: DAG, vgl. Definition 4.14). Aber es gibt gerichtete azyklische Graphen, die keine gerichteten Bäume sind.

Beispiel:



ist ein DAG, aber kein gerichteter Baum.

- (b) Für jeden gerichteten Baum $B = (V, E)$, dessen Knotenmenge endlich und nicht-leer ist, gilt:

$$|E| = |V| - 1.$$

Dies folgt unmittelbar aus Satz 4.45, da der ungerichtete Graph, der entsteht, indem man in B die Kantenorientierung "vergisst" (d.h. jede gerichtete Kante (i, j) durch die ungerichtete Kante $\{i, j\}$ ersetzt), ein ungerichteter Baum ist.

Alternativ zu Definition 4.51 kann man die gerichteten Bäume, deren Knotenmenge endlich und nicht-leer ist, auch folgendermaßen definieren:

Definition 4.54 (gerichtete Bäume, rekursive Definition).

Die Klasse der gerichteten Bäume mit endlicher, nicht-leerer Knotenmenge ist rekursiv wie folgt definiert:

Basisregel: Ist V eine Menge mit $|V| = 1$, so ist $B := (V, \emptyset)$ ein gerichteter Baum.

Skizze: $B := \bullet$

Der (eindeutig bestimmte) Knoten in V heißt **Wurzel** von B .

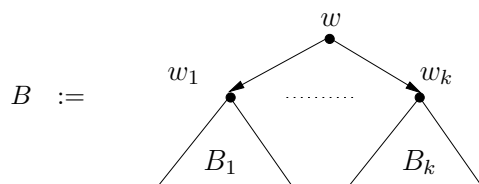
Die **Höhe** von B ist 0.

Rekursive Regel: Ist $k \in \mathbb{N}_{>0}$, sind $B_1 = (V_1, E_1), \dots, B_k = (V_k, E_k)$ gerichtete Bäume mit paarweise disjunkten Knotenmengen (d.h. $V_i \cap V_j = \emptyset$ f.a. $i, j \in \{1, \dots, k\}$ mit $i \neq j$), sind $w_1 \in V_1, \dots, w_k \in V_k$ die Wurzeln von B_1, \dots, B_k , und ist w ein Element, das nicht in $V_1 \cup \dots \cup V_k$ liegt, dann ist der Graph $B = (V, E)$ mit

$$V := \{w\} \cup V_1 \cup \dots \cup V_k \quad \text{und} \quad E := E_1 \cup \dots \cup E_k \cup \{(w, w_i) : i \in \{1, \dots, k\}\}$$

ein gerichteter Baum.

Skizze:



Der Knoten w heißt **Wurzel** von B .

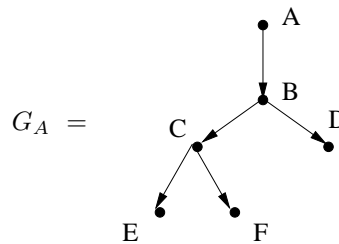
Die **Höhe** von B ist $1 + \max\{h_1, \dots, h_k\}$, wobei $h_1, \dots, h_k \in \mathbb{N}$ die Höhen der gerichteten Bäume B_1, \dots, B_k sind.

Notation 4.55 (Kinder eines Knotens).

Sei $B = (V, E)$ ein gerichteter Baum und sei $v \in V$ ein beliebiger Knoten in B . Die Knoten $v' \in V$, zu denen von v aus eine Kante führt (d.h. $(v, v') \in E$), heißen **Kinder** von v .

Kinder

Beispiel: Im Graphen



aus Beispiel 4.50 gilt: Knoten A hat genau ein Kind, nämlich B; Knoten B hat genau zwei Kinder, nämlich C und D; Knoten C hat genau zwei Kinder, nämlich E und F; und die Knoten D, E, F haben keine Kinder.

Eine besondere Rolle bei der Modellierung spielen Bäume, bei denen jeder Knoten höchstens zwei Kinder hat. Mit solchen Bäumen kann man z.B. Binär-Codierung oder Kaskaden von JA-NEIN-Entscheidungen beschreiben.

Definition 4.56 (Binärbaum, vollständiger Binärbaum).

Binärbaum

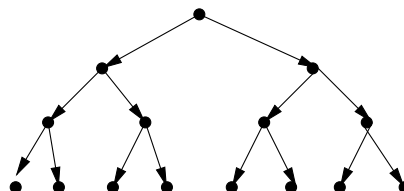
(a) Ein gerichteter Baum $B = (V, E)$ heißt **Binärbaum**, falls für jeden Knoten $v \in V$ gilt: $\text{Aus-Grad}_B(v) \leq 2$.

vollständiger Binärbaum

(b) Ein Binärbaum $B = (V, E)$ heißt **vollständiger Binärbaum**, falls gilt:

- (1) Jeder Knoten, der kein Blatt ist, hat Aus-Grad 2.
- (2) Es gibt eine Zahl $h \in \mathbb{N}$, so dass für jedes Blatt $v \in V$ gilt: Der Weg von der Wurzel zum Blatt v hat die Länge h .

Beispiel 4.57. Der Graph G_A aus Beispiel 4.50 ist ein Binärbaum, aber kein vollständiger Binärbaum. Der Graph G_B aus Beispiel 4.50 ist kein Binärbaum. Der folgende Graph B_3 ist ein **vollständiger Binärbaum** der Höhe 3:



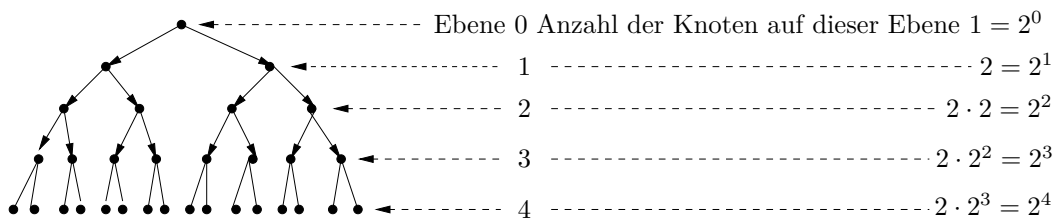
Zwischen der Höhe, der Anzahl der Blätter und der Anzahl der Knoten eines Binärbaums besteht der folgende wichtige Zusammenhang:

Satz 4.58. Sei $h \in \mathbb{N}$.

- (a) Jeder **vollständige Binärbaum der Höhe h** hat genau 2^h **Blätter** und genau $2^{h+1} - 1$ **Knoten**.
- (b) Jeder **Binärbaum der Höhe h** hat **höchstens 2^h Blätter** und **höchstens $2^{h+1} - 1$ Knoten**.

Beweis:

(a) *Skizze:*



Anhand dieser Skizze sieht man leicht, dass ein vollständiger Binärbaum der Höhe h genau 2^h Blätter und

$$2^0 + 2^1 + 2^2 + \dots + 2^h \stackrel{\text{Satz 2.49}}{=} 2^{h+1} - 1$$

Knoten besitzt.

Den formalen Beweis führen wir per Induktion nach h :

INDUKTIONSANFANG: $h = 0$:

Für jeden gerichteten Baum $B = (V, E)$ der Höhe 0 gilt: $|V| = 1$ und $|E| = 0$. D.h. B besteht aus genau einem Knoten, der gleichzeitig Wurzel und (einziges) Blatt des Baums ist. D.h: B hat genau $1 = 2^0 = 2^h$ Blätter und genau $1 = 2 - 1 = 2^1 - 1 = 2^{h+1} - 1$ Knoten.

INDUKTIONSSCHRITT: $h \rightarrow h + 1$:

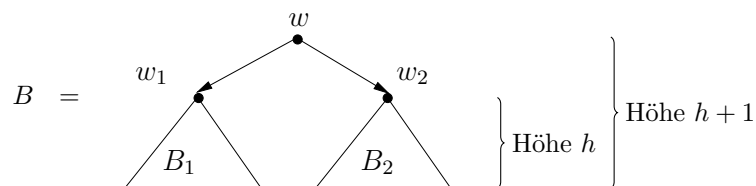
Sei $h \in \mathbb{N}$ beliebig.

Induktionsannahme: Jeder vollständige Binärbaum der Höhe h hat genau 2^h Blätter und genau $2^{h+1} - 1$ Knoten.

Behauptung: Jeder vollständige Binärbaum der Höhe $h + 1$ hat genau 2^{h+1} Blätter und genau $2^{h+2} - 1$ Knoten.

Beweis: Sei $B = (V, E)$ ein vollständiger Binärbaum der Höhe $h + 1$, und sei $w \in V$ die Wurzel von B . Wegen $h + 1 \geq 1$ hat w genau 2 Kinder. Seien $w_1 \in V$ und $w_2 \in V$ diese beiden Kinder von w . Für $i \in \{1, 2\}$ sei V_i die Menge aller Knoten aus V , zu denen von w_i aus ein Weg führt; und sei $B_i := (V_i, E_i)$ der induzierte Teilgraph von B mit Knotenmenge V_i .

Skizze:



Offensichtlich ist sowohl B_1 als auch B_2 ein vollständiger Binärbaum der Höhe h . Gemäß Induktionsannahme hat jeder der beiden Bäume B_1 und B_2 genau 2^h Blätter und genau $2^{h+1} - 1$ Knoten.

Der Baum B hat daher genau $2^h + 2^h = 2^{h+1}$ Blätter und genau $1 + (2^{h+1} - 1) + (2^{h+1} - 1) = 2 \cdot 2^{h+1} - 1 = 2^{h+2} - 1$ Knoten.

(b) Analog. Details: Übung.

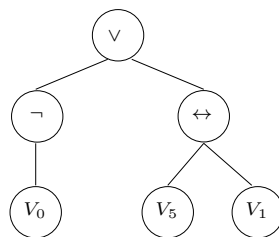
□

4.2.3 Modellierungsbeispiele

Gerichtete Bäume mit Knoten- oder Kantenmarkierungen können auf vielfältige Arten zur Modellierung genutzt werden.

Beispiel 4.59. In Kapitel 3 (Seite 78) haben wir Bäume bereits genutzt, um die Struktur einer aussagenlogischen Formel übersichtlich darzustellen. Der entsprechende Baum heißt **Syntaxbaum** der Formel.

Beispiel: Syntaxbaum der Formel $(\neg V_0 \vee (V_5 \leftrightarrow V_1))$:

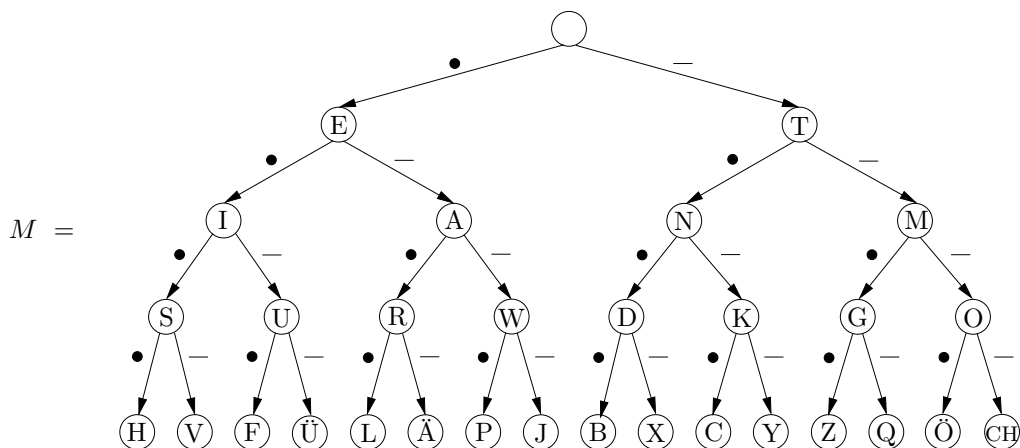


Auf ähnliche Art werden markierte Bäume genutzt, um die Struktur vieler anderer Objekte (an Stelle von aussagenlogischen Formeln) zu beschreiben — z.B. für arithmetische Terme, zur Darstellung von Klassen- und Objekthierarchien, zur Beschreibung der Struktur von Computerprogrammen oder umgangssprachlichen Texten oder auch zur Beschreibung der hierarchischen Organisationsstruktur einer Firma.

Beispiel 4.60. Folgen von Entscheidungen können in vielen Zusammenhängen durch gerichtete markierte Bäume modelliert werden. Solche Bäume heißen **Entscheidungs-bäume**. Durch einen solchen Entscheidungsbaum erhält man beispielsweise eine kompakte Darstellung des **Morse-Codes**.

Im Morse-Code wird jeder Buchstabe durch eine Folge von kurzen und langen Signalen repräsentiert. Ein “kurzes Signal” wird im folgenden Baum als Kantenmarkierung “•” dargestellt; ein “langes Signal” wird als “—” dargestellt. Insgesamt wird der Morsecode durch folgenden Entscheidungsbaum repräsentiert:

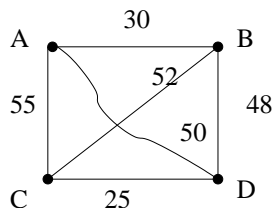
Entscheidungs-
bäume



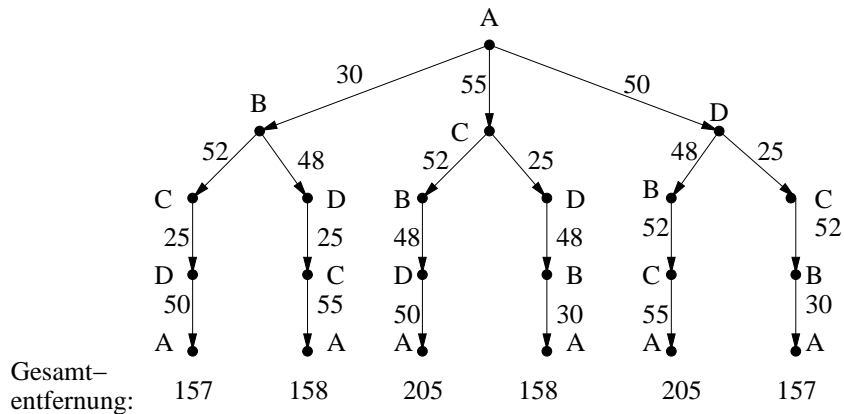
Eine eingehende Meldung aus kurzen und langen Signalen wird entschlüsselt, indem man an der Wurzel des Baums M beginnt und bei einem kurzen Signal nach links, bei einem langen nach rechts weitergeht. Eine längere Pause zeigt an, dass ein Buchstabe vollständig übermittelt ist.

In jedem Entscheidungsbaum modellieren die Knoten einen Zwischenstand bei der Entscheidungsfindung. Sie können entsprechend markiert sein, z.B. mit dem codierten Buchstaben des Morse-Codes. Die Kanten, die von einem Knoten ausgehen, modellieren die Alternativen, aus denen in dem durch den Knoten repräsentierten "Zustand" eine ausgewählt werden kann. Beim Morse-Code ist das jeweils ein kurzes oder ein langes Signal, das als Kantenmarkierung angegeben wird.

Beispiel 4.61. Markierte Bäume können auch genutzt werden, um den Lösungsraum kombinatorischer Probleme darzustellen. Als Beispiel betrachten wir einen Handlungsreisender, der einen möglichst kurzen Rundweg finden soll, auf dem er jede der Städte A, B, C, D besucht. Die Entfernungen (in km) zwischen den Städten sind als Kantenmarkierungen des folgenden Graphen gegeben:



Der folgende Baum repräsentiert alle möglichen in Stadt A startenden Rundwege:



Jeder Weg von der Wurzel zu einem Blatt repräsentiert dabei einen Rundweg, auf dem jede der Städte genau einmal besucht wird. Die Kantenmarkierungen geben die Entfernungen zwischen einzelnen Städten wieder. Eine zusätzliche Knotenmarkierung an jedem Blatt gibt die Gesamtlänge des entsprechenden Rundwegs an. Die beiden kürzesten Rundwege für unseren Handlungsreisenden sind also

$$(A, B, C, D, A) \quad \text{und} \quad (A, D, C, B, A).$$

Bemerkung 4.62. Nach dem gleichen Schema kann man auch Zugfolgen in Spielen modellieren: Jeder Knoten des Entscheidungsbaums modelliert einen Spielzustand. Die von dort ausgehenden Kanten geben an, welche Möglichkeiten für den nächsten Zug bestehen. Solche Darstellungen werden z.B. in Schachprogrammen verwendet, um die Folgen der anstehenden Entscheidung zu analysieren und zu bewerten. Ein Beispiel dazu findet sich in Aufgabe 4.26.

Beachte: Bei der Modellierung von Spielabläufen können manche “Spielzustände” (z.B. Konfigurationen eines Schachbretts) auf unterschiedlichen Wegen (d.h. Spielverläufen) erreicht werden, und trotzdem “im Sinne des Spiels” den selben Zustand beschreiben. In solchen Fällen könnte man im Entscheidungsbaum die zugehörigen Knoten zu einem einzigen Knoten zusammenfassen. Damit geht dann allerdings die Baum-Eigenschaft verloren, und es entsteht ein allgemeiner gerichteter Graph, der auch Kreise enthalten kann. Ein Kreis entspricht dann der Situation, dass eine Folge von Spielzügen in einen Zustand zurückführt, der früher schon einmal durchlaufen wurde.

4.3 Einige spezielle Arten von Graphen

In diesem Abschnitt werden einige spezielle Arten von Graphen vorgestellt, die eine wichtige Rolle in der Informatik spielen.

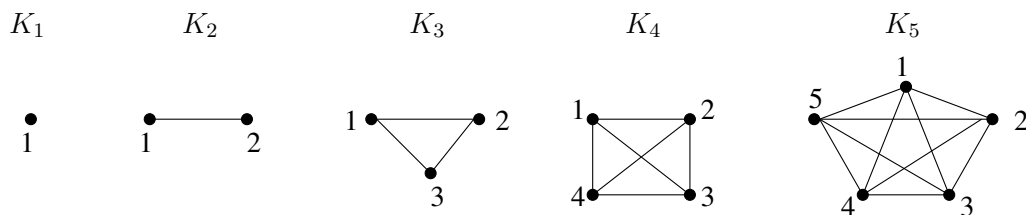
4.3.1 Spezielle ungerichtete Graphen

Definition 4.63 (Der vollständige Graph K_n).

Sei $n \in \mathbb{N}_{>0}$. Der **vollständige ungerichtete Graph** K_n hat Knotenmenge $\{1, \dots, n\}$ und Kantenmenge $\{\{i, j\} : i, j \in \{1, \dots, n\}, i \neq j\}$.

K_n
vollständiger
ungerichteter
Graph

Beispiele:



Beobachtung 4.64. Der Graph K_n hat n Knoten und $\frac{n \cdot (n-1)}{2}$ Kanten.

Definition 4.65 (Der vollständige bipartite Graph $K_{m,n}$).

Seien $m, n \in \mathbb{N}_{>0}$. Der vollständige ungerichtete bipartite Graph $K_{m,n}$ hat Knotenmenge

$$\{(1, i) : i \in \{1, \dots, m\}\} \cup \{(2, j) : j \in \{1, \dots, n\}\}$$

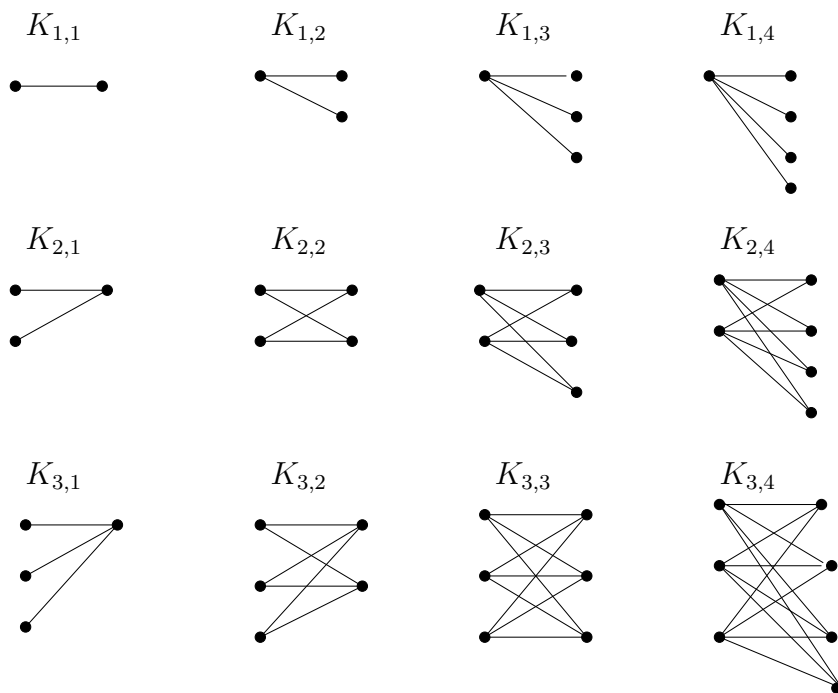
und Kantenmenge

$$\{\{(1, i), (2, j)\} : i \in \{1, \dots, m\}, j \in \{1, \dots, n\}\}.$$

$K_{m,n}$

vollständiger
ungerichteter
bipartiter Graph

Beispiele:



Beobachtung 4.66. Der Graph $K_{m,n}$ hat $m + n$ Knoten und $m \cdot n$ Kanten.

Notation 4.67. Ein ungerichteter Graph G mit endlicher, nicht-leerer Knotenmenge heißt

vollständig
vollständig
bipartit

- (a) **vollständig**, falls es ein $n \in \mathbb{N}_{>0}$ gibt, so dass $G \cong K_n$ (d.h. G ist isomorph zu K_n).
 (b) **vollständig bipartit**, falls es Zahlen $m, n \in \mathbb{N}_{>0}$ gibt, so dass $G \cong K_{m,n}$.

4.3.2 Spezielle gerichtete Graphen

Gemäß Definition 4.6 (“gerichteter Graph”) und Definition 2.26(c) (“ k -stellige Relation”) kann jeder gerichtete Graph $G = (V, E)$ als eine 2-stellige Relation über V aufgefasst werden, da die Kantenmenge E von G ja gerade eine Teilmenge von $V^2 = V \times V$ ist. Umgekehrt können wir natürlich auch jede 2-stellige Relation R über einer Menge V als gerichteten Graph mit Knotenmenge V und Kantenmenge R auffassen. Gerichtete Graphen mit Knotenmenge V sind also dasselbe wie 2-stellige Relationen über einer Menge V .

Von besonderem Interesse sind 2-stellige Relationen, die eine oder mehrere der folgenden Eigenschaften besitzen:

Definition 4.68.

Sei E eine 2-stellige Relation über einer Menge V (d.h. $G = (V, E)$ ist ein gerichteter Graph).

reflexiv

- (a) E heißt **reflexiv**, falls für alle $v \in V$ gilt:

$$(v, v) \in E. \quad (\text{Skizze: } v \begin{array}{c} \circlearrowleft \end{array})$$

symmetrisch

- (b) E heißt **symmetrisch**, falls f.a. $v, w \in V$ gilt:

$$\text{Wenn } (v, w) \in E, \text{ dann auch } (w, v) \in E.$$

(d.h. zu jeder Kante $v \longrightarrow w$ gibt es auch eine “Rückwärtskante” $w \longleftarrow v$)

antisymmetrisch

- (c) E heißt **antisymmetrisch**, falls f.a. $v, w \in V$ gilt:

$$\text{Wenn } (v, w) \in E \text{ und } (w, v) \in E, \text{ dann } v = w.$$

(d.h. Ist $v \neq w$, so gibt es in E allenfalls eine der beiden Kanten $v \longrightarrow w$ und $w \longleftarrow v$)

konnex

- (d) E heißt **konnex**, falls f.a. $v, w \in V$ gilt:

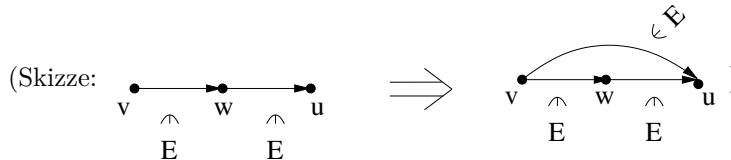
$$(v, w) \in E \text{ oder } (w, v) \in E.$$

(d.h. mindestens eine der beiden Kanten $v \longrightarrow w$ und $w \longleftarrow v$ liegt in E)

transitiv

- (e) E heißt **transitiv**, falls f.a. $v, w, u \in V$ gilt:

$$\text{Ist } (v, w) \in E \text{ und } (w, u) \in E, \text{ so auch } (v, u) \in E.$$



Äquivalenzrelationen

Definition 4.69 (Äquivalenzrelation).

Eine **Äquivalenzrelation** ist eine 2-stellige Relation, die **reflexiv**, **transitiv** und **symmetrisch** ist. Äquivalenzrelation

Beispiel 4.70. Beispiele für Äquivalenzrelationen:

(a) **Gleichheit:** Für jede Menge M ist

$$E := \{(m, m) : m \in M\}$$

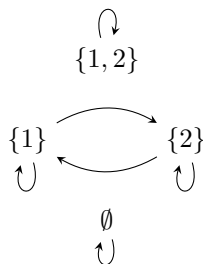
eine Äquivalenzrelation. Die Aussage " $(x, y) \in E$ " entspricht gerade der Aussage " $x = y$ ".

(b) **Gleichmächtigkeit:** Für jede endliche Menge M ist

$$E := \{(A, B) : A \subseteq M, B \subseteq M, |A| = |B|\}$$

eine Äquivalenzrelation über der Potenzmenge $\mathcal{P}(M)$.

Skizze für $M = \{1, 2\}$:



(c) **Logische Äquivalenz:** Die Relation

$$E := \{(\varphi, \psi) : \varphi, \psi \in \text{AL}, \varphi \equiv \psi\}$$

ist eine Äquivalenzrelation über der Menge AL aller aussagenlogischen Formeln.

Bemerkung 4.71 (Äquivalenzklassen). Sei E eine Äquivalenzrelation über einer Menge V . Für jedes $v \in V$ bezeichnet

$$[v]_E := \{v' \in V : (v, v') \in E\}$$

die **Äquivalenzklasse** von v bezüglich E . D.h.: Die Äquivalenzklasse $[v]_E$ besteht aus allen Elementen von V , die laut E "äquivalent" zu v sind. [v]_E
Äquivalenzklasse

Eine Menge $W \subseteq V$ heißt **Äquivalenzklasse** (bzgl. E), falls es ein $v \in V$ mit $W = [v]_E$ gibt. Das Element v wird dann ein **Vertreter** seiner Äquivalenzklasse W genannt.

Man sieht leicht, dass für alle $v, w \in V$ gilt: Entweder $[v]_E = [w]_E$ oder $[v]_E \cap [w]_E = \emptyset$. Falls V endlich und nicht leer ist, folgt daraus, dass es eine Zahl $k \in \mathbb{N}_{>0}$ und Äquivalenzklassen W_1, \dots, W_k geben muss, so dass $V = W_1 \cup \dots \cup W_k$ ist. Die Zahl k wird auch **Index** von E genannt. D.h.: Der Index einer Äquivalenzrelation gibt an, wie viele verschiedene Äquivalenzklassen es gibt. Index

Beispielsweise hat die Gleichmächtigkeits-Relation aus Beispiel 4.70 (b) den Index $|M| + 1$.

Ordnungsrelationen

Definition 4.72 (Ordnungen).

Sei E eine 2-stellige Relation über einer Menge V .

Präordnung

(a) E heißt **Präordnung**, falls E **reflexiv und transitiv** ist.

partielle Ordnung

(b) E heißt **partielle Ordnung**, falls E **reflexiv, transitiv und antisymmetrisch** ist.

lineare Ordnung

(c) E heißt **lineare Ordnung** oder **totale Ordnung**, falls E **reflexiv, transitiv, antisymmetrisch und konnex** ist.

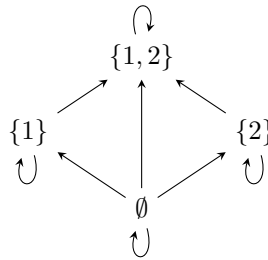
totale Ordnung

Beispiel 4.73.

(a) \leq ist eine **lineare Ordnung** auf \mathbb{N} (und \mathbb{Z} , \mathbb{Q} und \mathbb{R}). Ebenso ist \geq eine lineare Ordnung auf \mathbb{N} (und \mathbb{Z} , \mathbb{Q} und \mathbb{R}).

(b) Für jede Menge M sind \subseteq und \supseteq **partielle Ordnungen** auf der Potenzmenge $\mathcal{P}(M)$ (aber keine linearen Ordnungen).

Skizze für " \subseteq " bei $M = \{1, 2\}$:



(c) Die Folgerungsrelation für aussagenlogische Formeln (siehe Definition 3.24)

$$E := \{(\varphi, \psi) : \varphi, \psi \in \text{AL}, \varphi \models \psi\}$$

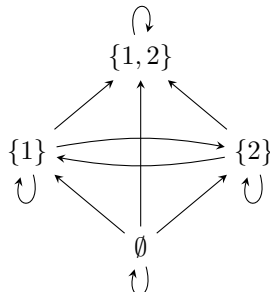
ist eine **Präordnung** auf AL (aber keine partielle Ordnung).

(d) Für jede endliche Menge M ist

$$E := \{(A, B) : A, B \subseteq M, |A| \leq |B|\}$$

eine **Präordnung** auf $\mathcal{P}(M)$ (aber keine partielle Ordnung).

Skizze für $M = \{1, 2\}$:



Die reflexive und transitive Hülle einer Relation

Definition 4.74 (reflexive und transitive Hülle).

Sei $G = (V, E)$ ein gerichteter Graph. Die **reflexive und transitive Hülle** (bzw. der **reflexive und transitive Abschluss**) von E auf V ist die rekursiv wie folgt definierte Relation $E^* \subseteq V \times V$: reflexive und transitive Hülle

Basisregeln:

- F.a. $v \in V$ ist $(v, v) \in E^*$.
- F.a. $(v, w) \in E$ ist $(v, w) \in E^*$

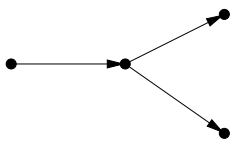
Rekursive Regel:

- Sind $(v, w) \in E^*$ und $(w, u) \in E^*$, so ist auch $(v, u) \in E^*$.

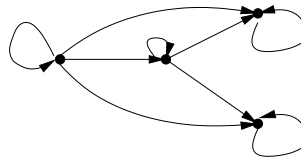
Das heißt: Der reflexive und transitive Abschluss von E auf V ist die kleinste Obermenge von E , die reflexiv und transitiv ist.

Beispiel:

$G = (V, E) :=$



$G^* = (V, E^*) :$



Beobachtung 4.75. Sei $G = (V, E)$ ein gerichteter Graph und seien $v, w \in V$. Dann sind die beiden folgenden Aussagen äquivalent:

- $(v, w) \in E^*$, wobei E^* die reflexive und transitive Hülle von E auf V ist.
- Es gibt in G einen Weg von v nach w .

Beweis: Übung. □

4.4 Literaturhinweise

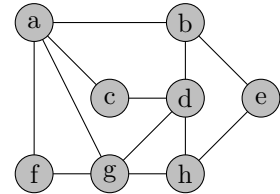
Als vertiefende Lektüre sei Kapitel 5 in [15], Kapitel 11 in [22], Teile der Kapitel 0–4 und 8 in [5], sowie Teile der Kapitel 7–10 und 13 in [19] empfohlen. Eine umfassende Einführung in die Graphentheorie gibt das Lehrbuch [5].

Quellennachweis: Viele der in diesem Kapitel angegebenen Modellierungsbeispiele sowie die folgenden Aufgaben 4.1, 4.4, 4.5 und 4.26 sind dem Buch [15] entnommen.

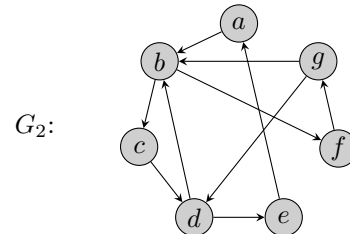
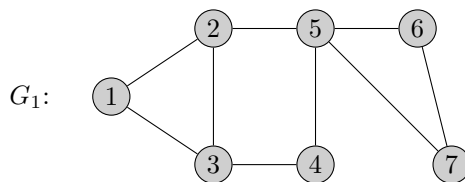
4.5 Übungsaufgaben zu Kapitel 4

Aufgabe 4.1. Betrachten Sie den ungerichteten Graphen G auf der rechten Seite.

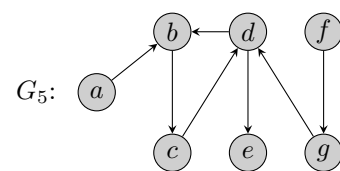
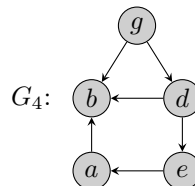
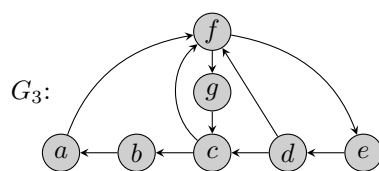
- Geben Sie die Knotenmenge V und die Kantenmenge E des Graphen G an. Repräsentieren Sie G außerdem durch eine Adjazenzmatrix und eine Adjazenzliste.
- Geben Sie einen Euler-Weg in G an. Besitzt G auch einen Euler-Kreis?
- Geben Sie einen Hamilton-Kreis in G an.
- Geben Sie einen Spannbaum von G an, den man so wurzeln kann, dass er die Höhe 2 hat. Kennzeichnen Sie die Wurzel in Ihrer Lösung.



Aufgabe 4.2. Es seien die folgenden beiden Graphen G_1 und G_2 gegeben:



- Geben Sie für jeden der beiden Graphen G_1 und G_2 die Knotenmenge und die Kantenmenge an. Repräsentieren Sie außerdem jeden der beiden Graphen durch eine Adjazenzmatrix und eine Adjazenzliste.
- Geben Sie einen Weg von 2 nach 4 in G_1 an, der *nicht* einfach ist. Geben Sie außerdem einen Kreis in G_1 an, der *nicht* einfach ist und durch den Knoten 2 verläuft.
- Ist G_1 zusammenhängend? Ist G_2 stark zusammenhängend? Ist G_2 azyklisch?
- Überprüfen Sie für jeden der folgenden Graphen G , ob folgendes gilt: (i) $G = G_2$, (ii) G ist ein Teilgraph von G_2 , (iii) G ist ein induzierter Teilgraph von G_2 , (iv) G ist isomorph zu G_2 . Geben Sie bei (d) auch einen Isomorphismus von G nach G_2 an, falls dieser existiert.



Aufgabe 4.3.

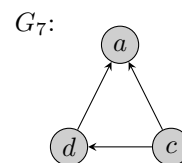
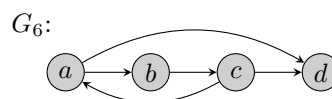
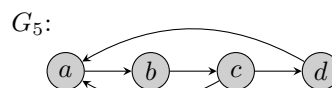
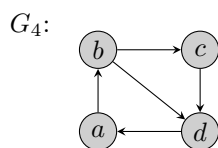
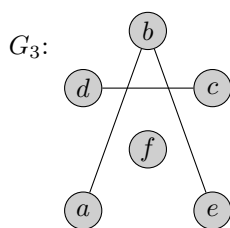
- (a) Geben Sie die folgenden Graphen G_1 und G_2 in graphischer Darstellung an.
Hinweis: Beachten Sie dabei, ob es sich jeweils um einen gerichteten oder einen ungerichteten Graphen handelt.

(i) $G_1 = (\{1, 2, 3, 4, 5, 6\}, \{\{x, y\} : x = 2 \cdot y\})$

(ii) $G_2 = (\{x \in \mathbb{N}_{>0} : 1 \leq x \leq 6\}, \{(x, y) : x = y + 1\} \cup \{(x, y) : y = 3 \cdot x\})$

Sind die Graphen G_1 und G_2 zusammenhängend bzw. stark zusammenhängend? Sind sie azyklisch?

- (b) Seien G_3, G_4, G_5, G_6 und G_7 die folgenden Graphen:



- (i) Geben Sie jeweils die Knoten- und Kantenmenge von G_3 und G_4 an.
(ii) Repräsentieren Sie die Graphen G_3 und G_4 jeweils durch eine Adjazenzliste und durch eine Adjazenzmatrix.
(iii) Gelten die folgenden Aussagen ?

(I) $G_3 \cong G_1$ (II) $G_4 \cong G_5$ (III) $G_5 \cong G_6$ (IV) $G_6 \cong G_7$

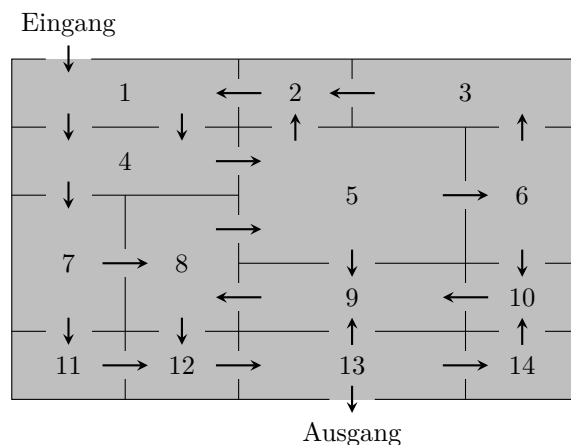
(V) G_7 ist ein induzierter Teilgraph von G_5 .

(VI) G_7 ist ein induzierter Teilgraph von G_6

Geben Sie für (I) bis (IV) jeweils einen Isomorphismus an, falls ein solcher existiert.

- (iv) Geben Sie in G_3 und G_5 jeweils einen nicht einfachen Weg an, der kein Kreis ist.
(v) Geben Sie in G_3 und G_5 jeweils einen nicht einfachen Kreis und einen einfachen Kreis an, falls ein solcher existiert.

Aufgabe 4.4. Die folgende Abbildung stellt den Grundriss eines Irrgartens dar.



Die Türen in diesem Irrgarten schwingen nur zu einer Seite auf und haben keine Klinken o.ä. Nachdem also ein Besucher die Eingangstür oder eine nachfolgende Tür durchschritten hat und die Tür hinter ihm zugefallen ist, kann der Besucher nicht mehr durch diese Tür zurück. Die Tür bleibt aber für weitere Durchgänge in der ursprünglichen Richtung benutzbar. Die allgemeinen Sicherheitsbestimmungen für Irrgärten schreiben vor, dass jeder Besucher, der den Irrgarten betritt, – egal wie er läuft – den Ausgang erreichen kann.

- Modellieren Sie den Irrgarten durch einen Graphen.
- Formulieren Sie die allgemeinen Sicherheitsbestimmungen für Irrgärten mit Begriffen der Graphentheorie.
- Überprüfen Sie anhand der Formulierungen aus (b), ob der angegebene Irrgarten den allgemeinen Sicherheitsbestimmungen entspricht.

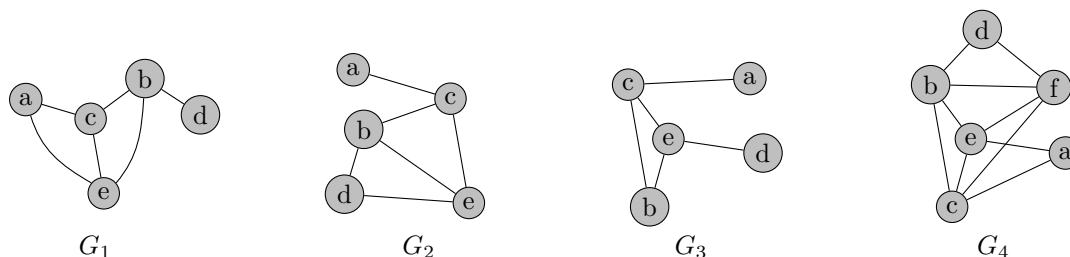
Aufgabe 4.5. Sie bekommen die Aufgabe, $n \in \mathbb{N}_{>0}$ Rechner zu vernetzen. Ihr Auftraggeber verlangt folgende Eigenschaften des Netzwerkes:

- Von jedem Rechner muss jeder andere Rechner über einen Leitungsweg erreichbar sein.
- Auch wenn genau eine Leitung zwischen zwei Rechnern ausfällt, muss jeder Rechner über einen Leitungsweg mit jedem anderen Rechner verbunden sein.
- An jedem Rechner können maximal vier Leitungen angeschlossen werden.

Dabei können auf einer Leitung Daten in beide Richtungen gesendet werden. Ein solches Netzwerk lässt sich leicht als ungerichteter Graph darstellen: ein Knoten repräsentiert einen Rechner, und eine Kante repräsentiert eine Leitung.

- Formulieren Sie die Eigenschaften (1), (2) und (3) mit Begriffen der Graphentheorie.
- Untersuchen Sie die folgenden Graphen G_1 , G_2 und G_3 auf Ihre Tauglichkeit bezüglich der Eigenschaften (1), (2) bzw. (3):
 - $G_1 = (V_1, E_1)$ mit $V_1 = \{1, 2, \dots, n\}$ und $E_1 = \{\{1, i\} : 2 \leq i \leq n\}$
 - $G_2 = (V_2, E_2)$ mit $V_2 = V_1$ und $E_2 = \{\{i, i + 1\} : 1 \leq i < n\}$
 - $G_3 = (V_3, E_3)$ mit $V_3 = V_1$ und $E_3 = E_2 \cup \{\{n, 1\}\}$

Aufgabe 4.6. Es seien die folgenden ungerichteten Graphen G_1 , G_2 , G_3 und G_4 gegeben:



- Überprüfen Sie für alle $i, j \in \{1, 2, 3, 4\}$ mit $i \neq j$, ob Folgendes gilt:
 - $G_i = G_j$

- G_i ist ein Teilgraph von G_j
 - G_i ist ein induzierter Teilgraph von G_j
- (b) Überprüfen Sie, welche der Graphen isomorph zueinander sind. Falls zwei Graphen G_i und G_j isomorph sind, so geben Sie einen Isomorphismus von G_i nach G_j an. Falls hingegen G_i und G_j nicht isomorph sind, so begründen Sie dies.
- (c) Welche der Graphen kann man nachzeichnen, ohne den Stift abzusetzen oder eine Kante doppelt zu ziehen?

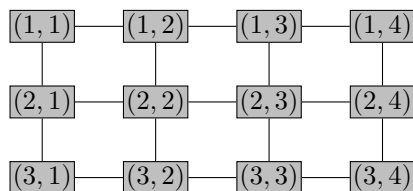
Aufgabe 4.7. Für $m, n \in \mathbb{N}_{>0}$ sei das $m \times n$ -Gitter der Graph $G_{m \times n} = (V_{m \times n}, E_{m \times n})$ mit

$$V_{m \times n} := \{ (i, j) : 1 \leq i \leq m, 1 \leq j \leq n \},$$

$$E_{m \times n} := \{ \{ (i, j), (i, j + 1) \} : 1 \leq i \leq m, 1 \leq j < n \} \cup$$

$$\{ \{ (i, j), (i + 1, j) \} : 1 \leq i < m, 1 \leq j \leq n \}.$$

Das 3×4 -Gitter $G_{3 \times 4}$ sieht z.B. wie folgt aus:



- (a) Überprüfen Sie, ob $G_{3 \times 4}$ bipartit ist. Falls $G_{3 \times 4}$ bipartit ist, so geben Sie zwei disjunkte Knotenmengen $V_1, V_2 \subseteq V_{3 \times 4}$ mit $V_1 \cup V_2 = V_{3 \times 4}$ an, so dass jede Kante aus $E_{3 \times 4}$ einen Knoten aus V_1 und einen Knoten aus V_2 miteinander verbindet. Falls $G_{3 \times 4}$ nicht bipartit ist, so begründen Sie dies.
- (b) Geben Sie ein Matching maximaler Größe in $G_{3 \times 4}$ an.
- (c) Geben Sie einen Hamilton-Kreis in $G_{3 \times 4}$ an.
- (d) Für welche $m, n \in \mathbb{N}_{>0}$ besitzt $G_{m \times n}$ einen Hamilton-Kreis, für welche nicht?

Hinweis: Stellen Sie sich vor, dass die Knoten des Gitters so mit den Farben rot oder blau eingefärbt sind, dass benachbarte Knoten unterschiedliche Farben besitzen. Jeder Weg durch das Gitter besucht daher immer abwechselnd einen blauen und einen roten Knoten.

Aufgabe 4.8. Ein ungerichteter endlicher Graph $G = (V, E)$ wird genau dann *kubisch* genannt, wenn für alle seine Knoten $v \in V$ gilt: $\text{Grad}_G(v) = 3$.

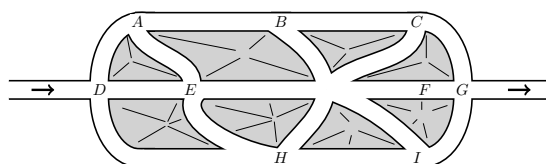
- (a) Geben Sie jeweils einen zusammenhängenden kubischen Graphen mit 4, 6 und 8 Knoten in graphischer Darstellung an.
- (b) Beweisen Sie durch vollständige Induktion, dass für jedes $n \in \mathbb{N}$ mit n gerade und $n \geq 4$ ein zusammenhängender kubischer Graph mit n Knoten existiert.

Aufgabe 4.9. Sei $G = (V, E)$ ein endlicher ungerichteter planarer Graph mit $V \neq \emptyset$. Dann gilt, dass $|E| < 3 \cdot |V|$.

- (a) Benutzen Sie diesen Zusammenhang, um zu beweisen, dass es in G einen Knoten $v \in V$ gibt mit $\text{Grad}_G(v) \leq 5$.
- (b) Beweisen Sie, dass die Knoten von G konfliktfrei mit sechs Farben gefärbt werden können. D. h. beweisen Sie, dass für G eine konfliktfreie Knotenmarkierung $m : V \rightarrow \{1, 2, 3, 4, 5, 6\}$ existiert.³
Hinweis: Benutzen Sie eine vollständige Induktion nach $n := |V|$ und die Aussage in Teilaufgabe (a) (auch wenn Sie diese nicht selbst bewiesen haben). Verwenden Sie keine anderen Resultate zur chromatischen Zahl planarer Graphen.

Aufgabe 4.10.

Es ist mal wieder so weit: Der Dude⁴ muss ganz allein die Welt vor der atomaren Zerstörung durch die Aliens retten. Er braucht nur noch die ultimative Anti-Alien-Waffe, deren Einzelteile allerdings auf Gebirgskämmen eines bisher unbekanntens Teils des Himalaja verteilt sind. Dieser Gebirgstteil wird durch die Abbildung skizziert: Die grauen Gebiete stellen Täler dar, die von den weißen Gebirgskämmen begrenzt werden. An den neun Gipfeln A, \dots, I treffen jeweils mehrere Gebirgskämme aufeinander. (So ist der Gipfel A beispielsweise mit den Gipfeln B, D und E verbunden.) Aus logistischen Gründen kann der Dude das Gebiet nur von Westen her betreten und nach Osten hin verlassen.



- (a) Modellieren Sie das Gebiet *ohne* Zu- und Abgang als graphische Darstellung eines ungerichteten Graphen, dessen Knoten die Gipfel darstellen und zwei Knoten genau dann benachbart sind, wenn die entsprechenden Gipfel durch einen Gebirgskamm miteinander verbunden sind.
- (b) Beim Durchqueren des Gebietes will der Dude, allein schon aus Gründen des Egos, jeden Gebirgskamm genau einmal benutzen. Ist das möglich? D. h., gibt es einen Euler-Weg in Ihrem Graphen aus Teilaufgabe (a), der in Knoten D startet und in Knoten G endet?
- (c) Natürlich hat der Dude sein Jetpack dabei, welches es ihm erlaubt, von einem Gipfel zu einem anderen zu fliegen, ohne einen Gebirgskamm dabei zu benutzen. Wie oft mindestens und für welche Flüge muss der Dude sein Jetpack einsetzen, um bei seiner Tour von Gipfel D zu Gipfel G jeden Gebirgskamm genau einmal zu benutzen? Geben Sie zur Beantwortung dieser Frage einen Graphen an, der den Graphen aus Teilaufgabe (a) geeignet um die Flüge mit dem Jetpack erweitert.

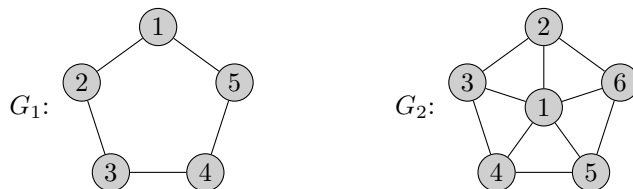
Aufgabe 4.11. Zwei Personen A und B spielen ein Spiel auf einem zusammenhängenden ungerichteten Graphen $G = (V, E)$. Die Spieler wählen abwechselnd Knoten v_1, v_2, v_3, \dots aus V , so dass v_1, v_2, v_3, \dots verschiedene Knoten sind und jeweils gilt: $\{v_i, v_{i+1}\} \in E$. Den ersten Knoten wählt A. Der letzte Spieler, der einen Knoten wählen kann, gewinnt.

Ein Spieler hat eine *Gewinnstrategie* in dem Spiel genau dann, wenn der Spieler das Spiel, unabhängig davon wie der andere Spieler spielt, gewinnen kann.

³Tatsächlich lassen sich die Knoten jedes planaren Graphen konfliktfrei mit vier Farben färben. Der Beweis dieser als „Vier-Farben-Satz“ bekannten Aussage ist sehr aufwendig. Der hier geforderte Beweis der entsprechenden Aussage für sechs Farben ist wesentlich einfacher.

⁴Der Dude steht für jeden der zahllosen Helden mit deren Hilfe man in Egoshootern die Welt retten kann.

- (a) Geben Sie für jeden der beiden folgenden Graphen G_1 und G_2 ein Matching maximaler Größe an und entscheiden Sie, welcher der beiden Spieler in dem Spiel auf dem entsprechenden Graph eine Gewinnstrategie hat.



- (b) Beweisen Sie, dass die beiden folgenden Aussagen äquivalent sind:
- G besitzt ein Matching M , so dass jeder Knoten aus V zu mindestens einer Kante aus M inzident ist.
 - Spieler B hat eine Gewinnstrategie in dem oben beschriebenen Spiel auf G .

Aufgabe 4.12. König Artus will für die Tafelrunde eine Sitzordnung für sich und neun seiner Ritter festlegen, bei der er und die neun Ritter im Kreis an einem runden Tisch sitzen. Das wäre nicht schwer, gäbe es nicht diese Rivalitäten und Eifersüchteleien zwischen den Rittern. König Artus möchte, dass Lancelot zu seiner Rechten und Kay zu seiner Linken sitzt. Erec weigert sich, neben jemand anderem als Lyonel oder Tristan zu sitzen. Galahad will weder neben Tristan noch neben Lancelot oder Lyonel sitzen. Parzival lehnt es ab, neben Gawain, Lancelot oder Lyonel zu sitzen. Gaheris möchte auf keinen Fall neben Gawain, Lancelot oder Kay sitzen. Tristan weigert sich, neben Lancelot, Parzival oder Kay zu sitzen. Gawain würde sich neben jeden anderen setzen, aber nicht neben Galahad oder Kay. Und Lyonel ist dagegen, neben Gawain zu sitzen.

- Stellen Sie den Konfliktgraphen auf.
- Verwenden Sie den Konfliktgraphen aus (a), um eine Tischordnung aufzustellen, die von allen akzeptiert wird. Zeichnen Sie den entsprechenden Graph und die Sitzordnung.

Aufgabe 4.13. Auf dem Weihnachtsmarkt von Großdorf sollen insgesamt 8 Stände rund um den Marktplatz arrangiert werden. Die 8 Stände setzen sich folgendermaßen zusammen:

- Ein Stand, in dem die traditionelle Weihnachtskrippe aufgebaut ist.
- Zwei Stände, an denen Kunsthandwerk verkauft wird: einer der beiden Stände ist die Töpferei, der andere bietet Holzschnitzwerk aus dem Erzgebirge an.
- Zwei Glühweinstände; einer davon wird von Herrn Max, der andere von Frau Peters betrieben.
- Drei Essensstände; einer davon verkauft Crêpes, der andere Waffeln und der dritte Steaks vom Holzkohlegrill.

Bei der Platzierung der 8 Stände um den Marktplatz ist folgendes zu beachten: Neben der Weihnachtskrippe darf keiner der Glühweinstände platziert werden. Essensstände dürfen nicht nebeneinander stehen, die beiden Glühweinstände dürfen nicht nebeneinander stehen, und die beiden Kunsthandwerkstände dürfen nicht nebeneinander stehen. Aus Sicherheitsgründen darf der Holzkohlegrill weder neben der Weihnachtskrippe noch neben dem Stand mit dem Holzschnitzwerk aus dem Erzgebirge stehen. Herr Max ist mit den Besitzern des Holzkohlegrills und

der Töpferei befreundet und möchte daher unbedingt die beiden als Nachbarn haben. Außerdem ist zu beachten, dass sich der Betreiber des Waffelstands weder mit Frau Peters noch mit dem Besitzer der Töpferei verträgt und daher auf keinen Fall neben einem der beiden platziert werden will.

- (a) Stellen Sie den Konfliktgraphen und das Komplement des Konfliktgraphen auf.
- (b) Gibt es im Komplement des Konfliktgraphen einen Hamiltonkreis? Falls ja, dann geben Sie einen solchen Hamiltonkreis an. Falls nein, dann begründen Sie, warum es keinen gibt.
- (c) Geben Sie eine Platzierung der 8 Stände rund um den Marktplatz an, mit der alle zufrieden sind.

Aufgabe 4.14. Die “Hyper Real Finance” Bank hat die Finanzkrise überstanden. Sie schreibt Stellenangebote aus: In den Abteilungen *Sparbücher*, *Immobilien*, *Kredite* und *Aktien* ist jeweils eine Position zu besetzen. Außerdem ist ein Job im *Vorstand* frei. Es treffen fünf Bewerbungen ein. Allerdings hat jeder der Bewerber starke Abneigungen gegen bestimmte Abteilungen, in denen er auf keinen Fall arbeiten will. Im Einzelnen ergeben sich folgende Zu- und Abneigungen:

- Alle Bewerber können sich vorstellen, in den Vorstand zu gehen.
- Andererseits will keiner außer Klaus mit den Sparbüchern zu tun haben.
- Klaus wiederum möchte weder in der Kredit- noch in der Aktienabteilung arbeiten.
- Gerhard möchte sich nicht mit den Immobilien beschäftigen.
- Auch Johann mag keine Immobilien und möchte nicht in die Aktienabteilung gehen.
- Frank will weder in die Aktien-, noch die Kredit- oder die Immobilienabteilung.
- Für Thomas gibt es bis auf die Sparbücher und die Immobilien keine Einschränkungen.

Es ist klar, dass jede Stelle nur von höchstens einem Bewerber besetzt werden kann und jeder Bewerber nur höchstens eine Stelle erhalten kann.

- (a) Geben Sie den Konfliktgraphen an, dessen Knotenmenge die Bewerber und die offenen Stellen repräsentiert. Eine Kante in diesem Graphen zwischen Stelle A und Bewerber B steht dafür, dass B nicht auf der Stelle A arbeiten will.
- (b) Geben Sie auf der Grundlage Ihres Konfliktgraphen einen “Zufriedenheitsgraphen” mit der gleichen Knotenmenge an. Eine Kante in diesem “Zufriedenheitsgraphen” zwischen Stelle A und Bewerber B soll bedeuten, dass B mit der Stelle A zufrieden wäre.
- (c) Geben sie ein Matching maximaler Größe in Ihrem “Zufriedenheitsgraphen” an.
- (d) Geben Sie eine Möglichkeit an, wie die Bank die Bewerber auf die Stellen verteilen kann, so dass möglichst viele Stellen besetzt werden und kein Bewerber eine Stelle erhält, auf der er nicht arbeiten will.
- (e) Begründen Sie, warum die von Ihnen gefundene Zuordnung bestmöglich ist, das heißt, warum es keine Zuordnung der Bewerber auf die Stellen gibt, die mehr Stellen besetzt.

Aufgabe 4.15. Es soll ein Klausurplan für 7 Klausuren A–G aufgestellt werden, bei dem kein Student mehr als eine Klausur pro Tag schreiben muss. Über die Teilnehmer an den Klausuren ist Folgendes bekannt:

- Für jede der Klausuren B,C,E und G gibt es mindestens einen Studenten, der sich für diese Klausur und A angemeldet hat.
 - Es gibt Studenten, die sich für B und C angemeldet haben, als auch Studenten, die die Kombination B und E gewählt haben.
 - Jeder Student, der D mitschreibt, hat sich auch für C und G angemeldet.
 - Mindestens ein Teilnehmer der Klausur G nimmt an F teil.
- (a) Stellen Sie den Konfliktgraphen auf.
- (b) Geben Sie einen Klausurplan an, bei dem kein Student an mehr als einer Klausur pro Tag teilnimmt.
- (c) Wie viele Tage werden für einen solchen Klausurplan benötigt?

Aufgabe 4.16. Beweisen Sie die Gültigkeit der folgenden Aussagen:

- (a) Für jeden endlichen ungerichteten Baum $B = (V, E)$ mit $V \neq \emptyset$ gilt: B ist bipartit.
- (b) Für jeden endlichen ungerichteten Graph G gibt es mindestens einen Spannbaum von G , falls G zusammenhängend ist. (Rückrichtung von Satz 4.48. aus dem Skript)
- (c) (i) Für jeden endlichen ungerichteten Graphen $G = (V, E)$ gilt: Bei jeder konfliktfreien Färbung von G mit $\chi(G)$ Farben muss es eine Menge von mindestens $\frac{|V|}{\chi(G)}$ Knoten in G geben, die mit der gleichen Farbe gefärbt sind.
- (ii) Für jeden endlichen ungerichteten Graphen $G = (V, E)$ und sein Komplement $\tilde{G} = (\tilde{V}, \tilde{E})$ gilt: $\chi(G) \cdot \chi(\tilde{G}) \geq |V|$

Aufgabe 4.17. Für einen ungerichteten Graphen $G = (V, E)$ sei der Graph $\tilde{G} = (\tilde{V}, \tilde{E})$ das Komplement von G , falls $\tilde{V} := V$ und $\tilde{E} := \{\{x, y\} : x, y \in V, x \neq y, \{x, y\} \notin E\}$. Ein ungerichteter Graph heißt *selbstkomplementär*, wenn er isomorph zu seinem Komplement ist.

Beweisen Sie die Gültigkeit der folgenden Aussagen:

- (a) Für je zwei ungerichtete Graphen G_1 und G_2 mit $G_1 \cong G_2$ gilt: Wenn G_1 zusammenhängend ist, so ist auch G_2 zusammenhängend.
- (b) Für jeden ungerichteten Graphen G gilt: G oder \tilde{G} ist zusammenhängend.
- (c) Jeder selbstkomplementäre Graph ist zusammenhängend.

Aufgabe 4.18. Sei $G = (V, E)$ ein gerichteter Graph mit mindestens zwei Knoten, so dass für alle $v, v' \in V$ mit $v \neq v'$ entweder $(v, v') \in E$ oder $(v', v) \in E$. Beweisen Sie durch vollständige Induktion über die Anzahl der Knoten von G , dass G einen Hamilton-Weg besitzt.

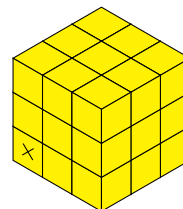
Aufgabe 4.19. Beweisen Sie, dass jeder ungerichtete, zusammenhängende Graph $G = (V, E)$, dessen Knotenmenge V endlich ist, einen Spannbaum besitzt.

Hinweis: Gehen Sie per Induktion nach $n := |E|$ vor.

Aufgabe 4.20. ⁵

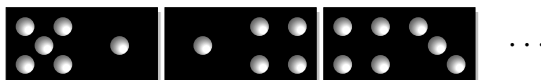
(a) Sei $G = (V, E)$ ein endlicher bipartiter Graph mit den Bipartitionsklassen V_1 und V_2 , d.h. jede Kante von E hat einen Endknoten in V_1 und einen in V_2 , außerdem ist $V = V_1 \dot{\cup} V_2$. Beweisen Sie die folgende Aussage: Falls $|V_1| > |V_2|$, so kann es keinen Hamilton-Weg in G geben, der in V_2 endet.

(b) Es sei ein Käsewürfel gegeben, der in $3 \times 3 \times 3$ kleinere Teilwürfel wie in der nebenstehenden Abbildung unterteilt ist. Zwei Teilwürfel sind *benachbart*, wenn sie sich entlang einer Fläche berühren. So ist beispielsweise der mit einem Kreuz markierte Teilwürfel mit 3 Teilwürfeln benachbart. Weiterhin sei eine Maus gegeben, die das Ziel hat, den großen Käsewürfel vollständig zu verspeisen. Sie kann dabei nur schrittweise vorgehen, indem sie in jedem Schritt einen Teilwürfel komplett verspeist und im nächsten Schritt mit einem Teilwürfel fortfährt, der mit dem gerade verspeisten Teilwürfel benachbart ist. Außerdem macht die Maus keinen Schritt, ohne einen Teilwürfel zu fressen und natürlich kann jeder Teilwürfel nur genau einmal gefressen werden.⁶



Kann die Maus bei dem markierten Teilwürfel starten und alle Teilwürfel so verspeisen, dass sie als Letztes den Teilwürfel in der Mitte des Käsewürfels frisst? Beweisen Sie, dass Ihre Antwort korrekt ist. (Sie können dafür die Aussage aus Teilaufgabe (a) benutzen, auch wenn Sie diese nicht bewiesen haben.)

Aufgabe 4.21. ⁵ Die Menge $Dom_n := \{\{x, y\} : x, y \in \mathbb{N}, x \neq y, 0 \leq x \leq n, 0 \leq y \leq n\}$ repräsentiert eine Teilmenge aller möglichen Dominosteine. Das Element $\{x, y\} = \{y, x\}$ steht für den Stein, der x Augen auf der einen und y Augen auf der anderen Hälfte zeigt, wobei $x \neq y$ für jeden Stein in Dom_n gilt. Die maximal vorkommende Augenzahl auf einer Steinhälfte wird mit $n \in \mathbb{N}$ bezeichnet. Zwei Steine können aneinander gelegt werden, wenn ihre benachbarten Augenzahlen gleich sind. Mehrere aneinander gelegte Steine ergeben eine Kette, die beispielsweise wie folgt beginnt:



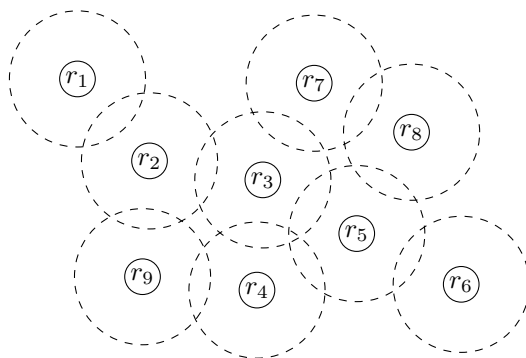
Ist es möglich, eine einzelne Kette ohne Verzweigungen so zu legen, dass jeder Stein aus Dom_n genau einmal vorkommt?

1. Modellieren Sie diese Frage als ein Problem für einen geeigneten Graphen.
2. Gibt es eine Kette für $n = 18$? Beweisen Sie, dass Ihre Antwort korrekt ist.
3. Für welche $n \in \mathbb{N}$ gibt es eine Kette, für welche nicht?
4. Zu Dom_n werden alle Steine hinzu getan, deren Augenzahl pro Hälfte höchstens n beträgt und die auf beiden Hälften die gleiche Augenzahl zeigen. Es entsteht $Dom'_n = Dom_n \cup \{\{x, x\} : x \in \mathbb{N}, 0 \leq x \leq n\}$. Für welche $n \in \mathbb{N}$ gibt es eine verzweigungsfreie Kette, die alle Steine aus Dom'_n genau einmal benutzt?

⁵Diese Übungsaufgabe entstand in Anlehnung an eine Übungsaufgabe von Manfred Nitzsche: „Graphen für Einsteiger“, Verlag Vieweg+Teubner, 2009.

⁶Es ist noch zu erwähnen, dass sich die ganze Situation in der Schwerelosigkeit abspielt, d.h. die Maus kann sich in jede Richtung fressen und es besteht auch nicht die Gefahr, dass der Würfel umkippt oder herunterfällt, wenn die untere Ebene von Teilwürfeln teilweise oder komplett gefressen ist.

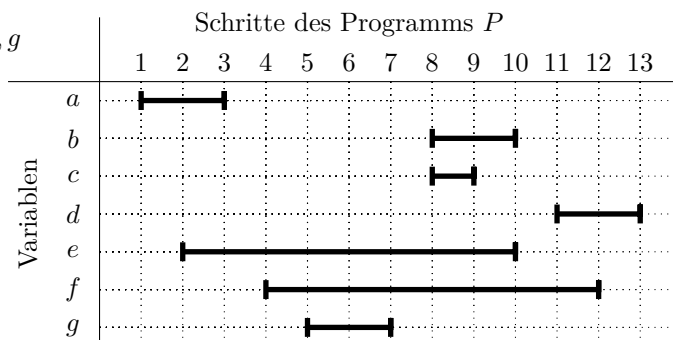
Aufgabe 4.22. Es seien die Radiostationen r_1, \dots, r_9 gegeben, denen jeweils eine Sendefrequenz zugeordnet werden soll. Radiostationen, die zu dicht beieinander liegen, dürfen allerdings nicht die gleiche Frequenz erhalten. Das nebenstehende Diagramm stellt die Lage der einzelnen Radiostationen dar. Um jede Station ist ein gestrichelter Kreis eingezeichnet, der die Reichweite einer Radiostation repräsentiert. Schneiden sich die Kreise von zwei Radiostationen r_i und r_j , so liegen r_i und r_j zu dicht beieinander und dürfen nicht die gleiche Frequenz zugeordnet bekommen.



- (a) Geben Sie den Konfliktgraphen an, der als Knotenmenge die Radiostationen besitzt und bei dem eine Kante zwischen zwei Radiostationen r_i und r_j anzeigt, dass r_i und r_j nicht die gleiche Frequenz benutzen dürfen.
- (b) Sei $G = (V, E)$ der Konfliktgraph aus Aufgabenteil (a). Geben Sie eine konfliktfreie Knotenmarkierung $m : V \rightarrow \mathbb{N}$ für G an, die möglichst wenige verschiedene Markierungen benutzt, d.h., $|\text{Bild}(m)|$ soll minimal sein.
- (c) Weisen Sie jeder der Radiostationen r_1, \dots, r_9 genau eine Frequenz zu, so dass Radiostationen, die zueinander in Konflikt stehen, nicht die gleiche Frequenz erhalten und möglichst wenige verschiedene Frequenzen benötigt werden.
- (d) Wie viele verschiedene Frequenzen werden für die Radiostationen r_1, \dots, r_9 mindestens benötigt, d.h. wie groß ist die chromatische Zahl des Konfliktgraphen?

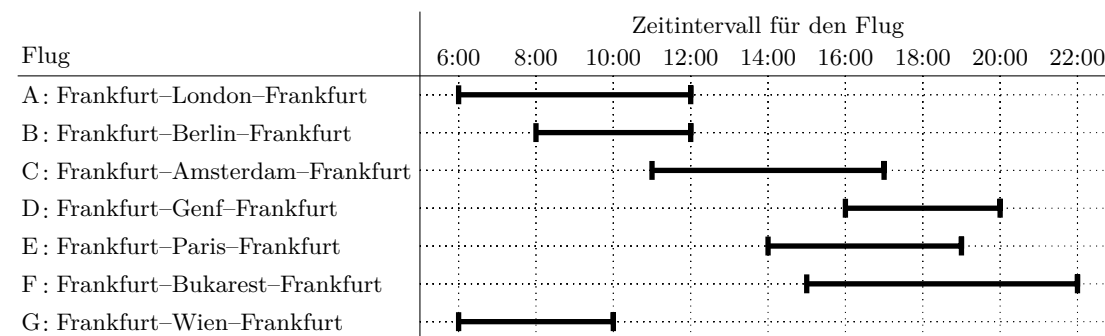
Aufgabe 4.23. Die meisten Programmiersprachen vermitteln dem Programmierer den Eindruck, ihm ständen potentiell unbeschränkt viele Variablen zur Verfügung, auf die praktisch gleichzeitig zugegriffen werden könne. Allerdings muss bei der tatsächlichen Ausführung des Programms jede Variable, auf die zugegriffen wird, im Hauptspeicher verfügbar sein. Seine Größe ist durch die Hardware begrenzt. In der Entwurfsmethode des Hardware-Software-Codesigns, die bei Eingebetteten Systemen gebräuchlich ist, wird versucht, die Hardware aus Platz-, Energie- und Kostengründen so weit zu reduzieren, dass ein gegebenes Programm gerade noch darauf ausführbar ist.

Wir stellen uns ein Programm P vor, das die sieben Variablen a, \dots, g benutzt, die jeweils eine Hauptspeicherzelle zur Speicherung benötigen. Die nebenstehende Abbildung gibt an, zu welchen Zeitpunkten der Ausführung von P welche Variablen im Hauptspeicher vorhanden sein müssen. So muss die Variable e beispielsweise in den Schritten 2 bis 10 von P im Hauptspeicher vorliegen. Zwei Variablen stehen im Konflikt miteinander, wenn sie nicht dieselbe Speicherzelle benutzen dürfen, da sie gleichzeitig im Hauptspeicher vorhanden sein müssen. Ziel der Aufgabe ist es, herauszufinden, wie viele Zellen der Hauptspeicher zur Ausführung von P haben muss, und in welche Zellen die Variablen im Verlaufe des Programms abgelegt werden können.



- (a) Geben Sie den ungerichteten Konfliktgraphen an, der als Knotenmenge die Variablen besitzt und bei dem eine Kante für einen Konflikt zwischen zwei Variablen steht.
- (b) Sei $G = (V, E)$ Ihr Konfliktgraph aus Aufgabenteil (a). Geben Sie eine konfliktfreie Knotenmarkierung $m : V \rightarrow \mathbb{N}$ an, die möglichst wenige verschiedene Markierungen verwendet, d. h. $|\text{Bild}(m)|$ soll minimal sein. Wie groß ist $\chi(G)$?
- (c) Weisen Sie jeder Variablen a bis g genau eine der Hauptspeicherzellen zu, so dass Variablen, die zueinander in Konflikt stehen, nicht derselben Zelle zugeordnet sind und möglichst wenige verschiedene Speicherzellen benötigt werden.

Aufgabe 4.24. Die kleine Frankfurter Fluggesellschaft Air-Flight hat für den kommenden Freitag sieben Flüge geplant, die wir im Folgenden mit den Buchstaben A–G bezeichnen. Für jeden Flug ist ein Zeitintervall (Abflugzeit bis Ankunftszeit) vorgesehen, in dem der Flug stattfinden soll:



Nun muss jedem Flug eines von fünf Flugzeugen F1–F5 zugeordnet werden, das für den Flug eingesetzt wird. Natürlich dürfen zwei Flüge, die zueinander in Konflikt stehen (d.h. bei denen sich die Zeitintervalle überlappen), nicht dem selben Flugzeug zugeordnet werden.

- (a) Geben Sie den Konfliktgraphen an, der als Knotenmenge die Flüge besitzt und bei dem eine Kante einen Konflikt zwischen zwei Flügen anzeigt.

- (b) Sei $G = (V, E)$ der Konfliktgraph aus Aufgabenteil (a). Geben Sie eine konfliktfreie Kontenmarkierung $m : V \rightarrow \mathbb{N}$ an, die möglichst wenige verschiedene Markierungen verwendet, d. h. $|\text{Bild}(m)|$ soll minimal sein.
- (c) Weisen Sie jedem der Flüge A–G genau eines der Flugzeuge F1–F5 zu, so dass Flüge, die zueinander in Konflikt stehen, nicht dem selben Flugzeug zugeordnet sind und möglichst wenige verschiedene Flugzeuge benötigt werden.
- (d) Wie viele Flugzeuge werden für die Flüge A–G mindestens benötigt, d. h. wie groß ist die chromatische Zahl des Konfliktgraphen?

Aufgabe 4.25. Es ist nun schon einige Zeit her, dass in Düsseldorf das Finale des 56. Eurovision Song Contest stattfand.⁷ Doch erst jetzt wird bekannt, wie erbittert die Teilnehmer hinter den Kulissen um die Ausstattung ihrer jeweiligen Show gekämpft haben. Besonders zwischen den fünf Finalteilnehmern, die schon vorher qualifiziert waren – Amaury Vassili für Frankreich, Raphael Gualazzi für Italien, die Gruppe Blue für das Vereinigte Königreich, Lucía Pérez für Spanien und Lena für Deutschland – gab es Streit. Um seine Show herauszuputzen, konnte jeder der fünf Finalteilnehmer sich einen von fünf Bühneneffekten aussuchen. Zur Wahl standen eine Lasershow, eine Windmaschine, Feuerwerk, ein Flammenwerfer und eine Schneefallsimulation. Um die Langeweile der Zuschauer zu begrenzen, durfte jeder Bühneneffekt nur höchstens einmal benutzt werden. Allerdings hatten alle Musiker ganz eigene Abneigungen gegen bestimmte Bühneneffekte, die sich im Einzelnen folgendermaßen darstellten:

- Blue mag keine Lasershow.
 - Raphael kann die Lasershow und die Schneefallsimulation nicht leiden.
 - Auch Lucía lehnt die Schneefallsimulation ab, ebenso wie das Feuerwerk.
 - Amaury missfallen das Feuerwerk, die Schneefallsimulation und der Flammenwerfer.
 - Lena hat eine Abneigung gegen alles, was keine Windmaschine ist.
- (a) Stellen Sie den Konfliktgraphen als *ungerichteten* Graphen auf, dessen Knotenmenge die Musiker und die Bühneneffekte repräsentiert. Eine Kante in diesem Graphen zwischen Musiker A und Effekt B soll dafür stehen, dass A den Effekt B nicht nutzen will.
- (b) Geben Sie auf der Grundlage Ihres Konfliktgraphen einen *ungerichteten* „Zufriedenheitsgraphen“ mit der gleichen Knotenmenge an. Eine Kante in diesem „Zufriedenheitsgraphen“ zwischen Musiker A und Effekt B soll bedeuten, dass A mit B zufrieden wäre.
- (c) Geben sie ein Matching maximaler Größe in Ihrem „Zufriedenheitsgraphen“ an.
- (d) Geben Sie eine Zuordnung zwischen Bühneneffekten und Musikern an, so dass jeder Musiker genau einen Effekt erhält, jeder Effekt genau einmal zugeordnet wird und alle Musiker mit ihrer Zuordnung zufrieden sind.

Aufgabe 4.26. Zwei Spieler A und B spielen das folgende Spiel. Das Spiel ist in Runden aufgeteilt, wobei Spieler A in den geraden Runden und Spieler B in den ungeraden Runden spielt. In der ersten Runde wählt Spieler B eine Zahl aus $\{1, 2\}$. In jeder der nachfolgenden Runden wählt der jeweilige Spieler eine Zahl aus $\{1, 2, 3\}$ mit der Einschränkung, dass die Zahl

⁷Der Eurovision Song Contest ist ein jährlich stattfindener Musikwettbewerb, welcher wirklich existiert. Der Konflikt der Kontrahenten ist frei erfunden.

aus der vorhergehenden Runde nicht gewählt werden darf. Nach jeder Runde wird die Summe der bereits gewählten Zahlen berechnet. Nimmt diese Summe den Wert 6 an, so gewinnt der Spieler der jeweiligen Runde; übersteigt sie den Wert 6, so verliert er.

- (a) Beschreiben Sie das Spiel durch einen Entscheidungsbaum.
- (b) Wer gewinnt, wenn beide Spieler optimal spielen, d.h. wenn jeder Spieler immer nur diejenigen Zahlen wählt, mit denen er – falls dies noch möglich ist – gewinnen kann?

Aufgabe 4.27. Alice und Bob spielen das Spiel *Yag*⁸ gegeneinander, das wie folgt definiert ist: Die Spieler schreiben gemeinsam eine Folge von Nullen und Einsen auf. Sie beginnen mit der leeren Zeile und sind abwechselnd am Zug, Alice beginnt. Der Spieler am Zug schreibt an das Ende der Zeile eine Null oder eine Eins. Ein Spieler gewinnt, wenn die von ihm hinzugefügte Ziffer einen Block der Länge zwei erzeugt, der in der Folge schon einmal vorkommt. (Dabei werden auch Blöcke betrachtet, die sich überlappen: Der Spieler, der die Folge 0111 erzeugt, gewinnt, da der Block 11 zweimal in der Folge vorkommt.) Haben die Spieler eine Folge der Länge vier erzeugt ohne dass es einen Gewinner gibt, so endet das Spiel unentschieden.

- (a) Beschreiben Sie das Spiel durch einen Entscheidungsbaum.
- (b) Ist der von Ihnen bei Teilaufgabe (a) aufgestellte Entscheidungsbaum ein Binärbaum? Ist er ein vollständiger Binärbaum?
- (c) Was haben alle Spielsituationen gemeinsam, die den Blättern des Entscheidungsbaumes entsprechen? Was haben alle Situationen des Spiels gemeinsam, die durch innere Knoten repräsentiert werden? Welcher Spielsituation entspricht die Wurzel?
- (d) Wie viele Runden dauert das Spiel höchstens, d. h. wie groß ist die Höhe des Entscheidungsbaumes? Wie viele Runden dauert das Spiel mindestens, d. h. was ist die kürzeste Länge eines Weges von der Wurzel zu einem Blatt?
- (e) Wer gewinnt, wenn beide Spieler optimal spielen, d. h. wenn jeder Spieler immer nur die Ziffer wählt, mit der er – falls dies noch möglich ist – gewinnen kann?

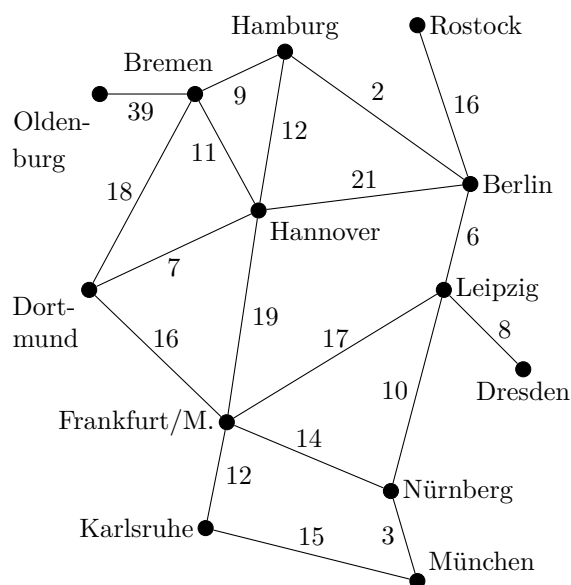
Aufgabe 4.28. Für ein $n \in \mathbb{N}$ seien 2^n Münzen gegeben, die wir im Folgenden mit M_1, \dots, M_{2^n} bezeichnen. Genau eine der Münzen ist schwerer als alle anderen. Diese Münze lässt sich mit Hilfe einer Balkenwaage wie folgt finden:

- (i.) Falls $n = 0$, ist die gesuchte Münze die einzige, die vorhanden ist.
- (ii.) Ansonsten vergleiche das Gesamtgewicht der Münzen aus der Menge $A := \{M_1, \dots, M_{2^{n-1}}\}$ mit dem Gesamtgewicht der Münzen aus der Menge $B := \{M_{2^{n-1}+1}, \dots, M_{2^n}\}$. Ist das Gesamtgewicht von A größer als das von B , muss sich die gesuchte Münze in A befinden und das beschriebene Verfahren wird rekursiv auf die Menge A angewendet, andernfalls wird es rekursiv auf die Menge B angewendet.
- (a) Beschreiben Sie das Verfahren für $n = 2$ durch einen Entscheidungsbaum. Wählen Sie hierfür geeignete Kanten- und Knotenbeschriftungen.
- (b) Ist der von Ihnen in Teilaufgabe (a) aufgestellte Entscheidungsbaum ein Binärbaum? Ist er ein vollständiger Binärbaum?

⁸Yet another game.

- (c) Welchen Situationen im Entscheidungsprozess entsprechen die inneren Knoten des Baumes? Welcher Situation entspricht ein Blatt?
- (d) Wie viele Wiegevorgänge müssen für 2^n Münzen mindestens durchgeführt werden? Wie viele Wiegevorgänge sind im schlimmsten Fall, also höchstens, nötig?

Aufgabe 4.29. Wir nehmen an, die Deutsche Bahn hat eine Studie für ausgewählte ICE-Strecken in Deutschland durchgeführt. Ziel der Studie war es, für einige Strecken jeweils zu bestimmen, welche Kosten pro unbesetztem Sitzplatz im ICE entstehen. Das Ergebnis der Studie sehen Sie im nebenstehenden Graphen. Der Zahlenwert an einer Kante zwischen zwei Städten gibt an, welche Kosten durch einen unbesetzten Sitzplatz im ICE zwischen diesen beiden Städten entstehen.



Wir nehmen weiterhin an, dass Manager der Deutschen Bahn Kosten sparen und daher so viele ICE-Strecken wie möglich abbauen möchten. Es soll ein Netz der ICE-Strecken übrig bleiben, die möglichst wenige Gesamtkosten für unbesetzte Plätze verursachen. Gleichzeitig soll aber jede der abgebildeten Städte an das ICE-Netz angeschlossen bleiben. Ähnlich wie im Beispiel (4.49) aus dem Skript ist also ein minimaler Spannbaum gesucht.

Wenden Sie folgendes Verfahren an, um einen minimalen Spannbaum B im abgebildeten Graphen zu finden: Wählen Sie einen beliebigen einfachen Kreis im Graphen. Entfernen Sie aus diesem Kreis die Kante mit den höchsten Kosten. Führen Sie dies solange fort, bis der jeweils entstehende Graph keine einfachen Kreise mehr enthält.

- (a) Geben Sie den von Ihnen bestimmten minimalen Spannbaum B an.
- (b) Welche Städte werden in B durch Blätter repräsentiert, d.h., welche Städte sind in B nur über genau eine Verbindung an das ICE-Netz angeschlossen?
- (c) Geben Sie den kürzesten Weg in B von Frankfurt/M. nach Dortmund an.
- (d) Geben Sie einen einfachen Weg maximaler Länge in B an, d.h. welches Städte-Paar ist in B so verbunden, dass ICE-Reisende dazwischen die maximale Anzahl anderer Städte besuchen müssen? Wie groß ist die Länge dieses Weges?

Aufgabe 4.30. Sei \mathcal{G} die Menge der ungerichteten Graphen $G = (V, E)$ mit $V \subseteq \mathbb{N}$.

(a) Unter (i) bis (iii) sind zweistellige Relationen über \mathcal{G} gegeben. Überprüfen Sie für jede dieser Relationen, ob sie reflexiv, symmetrisch, antisymmetrisch, konnex bzw. transitiv ist.

(i) $\{(G, G') \in \mathcal{G}^2 : G \cong G'\}$

(ii) $\{(G, G') \in \mathcal{G}^2 : G \text{ hat höchstens so viele Knoten wie } G'\}$

(iii) $\{(G, G') \in \mathcal{G}^2 : G' \text{ besitzt einen Teilgraph } G'' \text{ mit } G'' \cong G\}$

(b) Zeigen Sie, dass die zweistellige Relation

$$R := \{(G, G') \in \mathcal{G}^2 : G \text{ ist ein induzierter Teilgraph von } G'\}$$

eine partielle Ordnung, aber keine lineare Ordnung, auf \mathcal{G} ist.

Aufgabe 4.31. Ein ungerichteter Graph $G = (V, E)$ heißt *selbstkomplementär*, wenn er isomorph zu seinem Komplement $\tilde{G} = (\tilde{V}, \tilde{E})$ ist. Dabei ist $\tilde{V} := V$ und $\tilde{E} := \{\{x, y\} : x, y \in V, x \neq y, \{x, y\} \notin E\}$.

(a) Geben Sie für jedes $n \in \mathbb{N}_{>0}$ mit $n \leq 7$ einen selbstkomplementären Graphen an, der n Knoten hat, falls ein solcher Graph existiert.

(b) Beweisen Sie, dass für die Knotenanzahl $|V|$ jedes selbstkomplementären Graphen $G = (V, E)$ gilt: Es gibt ein $k \in \mathbb{N}$ so dass $|V| = 4k$ oder $|V| = 4k + 1$.

Aufgabe 4.32.

(a) Betrachten Sie die Relation $R := \{(a, a), (a, b), (a, c), (b, b), (b, c)\}$ über der Menge $A := \{a, b, c, d\}$. Welche Paare $(x, y) \in A \times A$ müssen zu R mindestens hinzugefügt werden, um aus R eine Relation zu erhalten, die jeweils

(i) reflexiv ist?

(iii) antisymmetrisch ist?

(v) transitiv ist?

(ii) symmetrisch ist?

(iv) konnex ist?

(vi) eine Präordnung ist?

(b) Betrachten Sie die folgenden Relationen R_i über der jeweiligen Menge M_i .

(i) $M_1 := \{1, 2, 3, 4, 5\}$, $R_1 := \{(1, 1), (1, 2), (2, 2), (2, 3), (3, 3), (4, 4), (4, 5), (5, 4), (5, 5)\}$

(ii) $M_2 := \{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\}$, $R_2 := \{(\clubsuit, \spadesuit), (\clubsuit, \heartsuit), (\clubsuit, \diamondsuit), (\spadesuit, \heartsuit), (\spadesuit, \diamondsuit), (\heartsuit, \diamondsuit)\}$

(iii) $M_3 := \{-4, -3, -2, -1, 0, 1, 2\}$, $R_3 := \{(x, y) \in M_3 \times M_3 : x \cdot y \leq 3\}$

(iv) $M_4 := \mathbb{N}_{>0}$, $R_4 := \{(a, b) \in M_4 \times M_4 : ggT(a, b) > 1\}$,

wobei $ggT(a, b)$ der größte gemeinsame Teiler der Zahlen a und b ist.

Stellen Sie R_1 und R_2 durch einen gerichteten Graphen in graphischer Darstellung dar.

Geben Sie für jedes $i \in \{1, 2, 3\}$ an, welche Eigenschaften (reflexiv, symmetrisch, antisymmetrisch, konnex, transitiv) die Relation R_i jeweils besitzt.

(c) Für einen gerichteten Baum $B = (V, E)$ definieren wir die Relation

$$R_B := \{(x, y) \in V \times V : \text{es gibt einen Weg von } x \text{ nach } y \text{ in } B\}$$

(i) Zeigen Sie, dass R_B für jeden gerichteten Baum B eine partielle Ordnung ist.

(ii) Geben Sie gerichtete Bäume B_1 und B_2 mit jeweils mindestens 3 Knoten an, so dass

- (I.) R_{B_1} eine lineare Ordnung ist. (II.) R_{B_2} keine lineare Ordnung ist.

(d) Für Worte über dem Alphabet A definieren wir folgende Relation:

$$\text{Prä}_A := \{(a, b) \in A^* \times A^* : \text{ex. } c \in A^*, \text{ s.d. } ac = b\}$$

Für den Fall, dass $(a, b) \in \text{Prä}_A$ heißt a *Präfix* von b .

- (i) Zeigen Sie, dass für jedes Alphabet A gilt: Prä_A ist eine partielle Ordnung.
(ii) Geben Sie zwei Alphabete A_1 und A_2 an, so dass

- (I.) Prä_{A_1} eine lineare Ordnung ist, (II.) Prä_{A_2} keine lineare Ordnung ist.

Aufgabe 4.33. Zeigen Sie, dass die im Folgenden angegebenen Relationen R_1 , R_2 und R_3 jeweils Äquivalenzrelationen sind. Geben Sie jeweils jede Äquivalenzklasse von R_1 , R_2 und R_3 sowie jeweils den Index von R_1 , R_2 und R_3 an. Geben Sie für jede Äquivalenzklasse von R_1 , R_2 und R_3 jeweils einen Vertreter an.

(a) Zwei Zahlen $a, b \in \mathbb{N}$ heißen *kongruent modulo 7* (geschrieben als $a \equiv b \pmod{7}$), wenn die Zahl $a - b$ durch 7 teilbar ist, d.h. es gibt ein $z \in \mathbb{Z}$, so dass $a - b = 7 \cdot z$. Betrachten Sie

$$R_1 := \{(a, b) \in \mathbb{N} \times \mathbb{N} : a \equiv b \pmod{7}\}$$

(b) Sei $A := \{a, b\}$ ein Alphabet und sei A^5 die Menge aller Wörter der Länge 5, die sich mit Buchstaben aus A bilden lassen. Betrachten Sie

$$R_2 := \{(w_1, w_2) \in A^5 \times A^5 : w_2 \text{ entsteht aus } w_1 \text{ durch Umsortierung der Buchstaben}\}$$

(c) Wir definieren $\text{AL}_{|V_0} := \{\varphi \in \text{AL} : \text{Var } \varphi = \{V_0\}\}$ als die Menge aller syntaktisch korrekten aussagenlogischen Formeln, die außer V_0 keine weiteren Ausagenvariablen enthalten. Dann sei

$$R_3 := \{(\varphi, \psi) \in \text{AL}_{|V_0} \times \text{AL}_{|V_0} : \varphi \equiv \psi\}.$$

Aufgabe 4.34. Um sich den Kauf eines Kühlschranks zu erleichtern, kann man sich auf drei wichtige Attribute konzentrieren. Da wäre als Erstes natürlich der Preis p , in heutigen Zeiten an zweiter Stelle der Stromverbrauch s und drittens, informatikertypisch, das Fassungsvermögen f des Tiefkühlfaches gemessen in Standardpizzen. So ordnen wir jedem in Frage kommenden Modell ein Tupel $(p, s, f) \in \mathbb{N}_{>0} \times \mathbb{N}_{>0} \times \mathbb{N}$ zu. Wir definieren die Relation „nicht schlechter als“, die wir mit \ll abkürzen, so dass für zwei Kühlschränke $k_1 = (p_1, s_1, f_1)$ und $k_2 = (p_2, s_2, f_2)$ gilt:

$$k_1 \ll k_2 \quad \iff \quad p_1 \leq p_2, s_1 \leq s_2 \text{ und } f_1 \geq f_2$$

Überprüfen Sie die Korrektheit der folgenden Aussagen:

- (a) \ll ist Präordnung. (b) \ll ist part. Ordnung. (c) \ll ist lineare Ordnung.

Aufgabe 4.35. Sei \mathfrak{G}_3 die Menge aller Graphen $G = (V, E)$ mit $V \subseteq \mathbb{N}$ und $|V| = 3$, und es gelte für zwei Graphen $G_1, G_2 \in \mathfrak{G}_3$, dass $G_1 \cong G_2$ genau dann, wenn G_1 isomorph zu G_2 ist.

(a) Zeigen Sie, dass die Relation \cong eine Äquivalenzrelation über \mathfrak{G}_3 ist.

- (b) Geben Sie für jede Äquivalenzklasse einen Vertreter in graphischer Darstellung an.
- (c) Wie groß ist der Index von \cong auf \mathfrak{S}_3 ?
- (d) Welche Kardinalität besitzen die Äquivalenzklassen?

Aufgabe 4.36. Beweisen oder widerlegen Sie die folgenden Aussagen:

- (a) Jede transitive und symmetrische Relation ist auch reflexiv.
- (b) Eine Relation ist genau dann antisymmetrisch, wenn sie nicht symmetrisch ist.
- (c) Jede Relation, die nicht konnex ist, ist auch nicht reflexiv.
- (d) Eine Relation ist genau dann transitiv, wenn sie symmetrisch und konnex ist.
- (e) Es gibt keine Relation, die gleichzeitig antisymmetrisch, symmetrisch, konnex, transitiv und reflexiv ist.
- (f) Der Index jeder konnexen Äquivalenzrelation ist 1.
- (g) Der Index jeder antisymmetrischen Äquivalenzrelation über einer endl. Menge M ist $|M|$.
- (h) Die Vereinigung von zwei Äquivalenzrelationen ist wieder eine Äquivalenzrelation.
- (i) Die Teilbarkeitsrelation⁹ auf den natürlichen Zahlen ist eine partielle Ordnung.
- (j) Die Teilbarkeitsrelation⁹ auf den ganzen Zahlen ist eine partielle Ordnung.

⁹Für die Teilbarkeitsrelation gilt: Das Paar (a, b) von ganzen Zahlen ist genau dann Element der Teilbarkeitsrelation, falls es eine ganze Zahl k gibt, so dass $a \cdot k = b$.

5 Markov-Ketten als Grundlage der Funktionsweise von Suchmaschinen im Internet

Ziel dieses Kapitels ist, einen kurzen Überblick über die Arbeitsweise von Suchmaschinen für das Internet zu geben. Wir betrachten hierbei eine Suchmaschine, die als Eingabe ein Stichwort oder eine Liste von Stichworten erhält, und die als Ausgabe eine Liste von Links auf Webseiten geben soll, deren Inhalt relevante Informationen zu den eingegebenen Stichworten enthält. Diese Liste soll so sortiert sein, dass die informativsten Links am weitesten oben stehen.

Die Herausforderungen, die sich beim Bau einer Suchmaschine stellen, sind vielfältig. Zum einen ist die Anzahl der Webseiten *sehr* groß: Bereits im Jahr 2008 gab es mehr als 1 Billion Webseiten.¹ Beachten Sie: 1 Billion = 1.000.000.000.000 = 10^{12} . Niemand kennt den genauen Inhalt des gesamten Internets, und das Internet verändert sich ständig: Täglich kommen neue Webseiten hinzu, viele Webseiten werden täglich aktualisiert, und andere nach einiger Zeit auch wieder gelöscht.

Eine Suchmaschine muss daher eine enorm große Menge von Daten verarbeiten, die in kurzen Zeitabständen immer wieder aktualisiert werden. Trotzdem müssen Suchanfragen, die an eine Suchmaschine geschickt werden, in "Echtzeit" beantwortet werden. Um die Ergebnisse nach ihrer Relevanz für die jeweiligen Suchbegriffe sortieren zu können, benötigt man auch ein sinnvolles Maß dafür, welche Webseiten als besonders "informativ" bewertet werden sollen.

5.1 Die Architektur von Suchmaschinen

Die Herausforderung besteht darin, Anfragen für einen sich rasant ändernden Suchraum gigantischer Größe ohne merkliche Reaktionszeit zu beantworten. Um dies zu gewährleisten, nutzen Suchmaschinen u.a. die folgenden Komponenten:

- (1) **Web-Crawler:** Computerprogramme, die **Crawler** genannt werden, durchforsten das Internet, um neue oder veränderte Webseiten zu identifizieren. Die von den Crawlern gefundenen Informationen über Webseiten und deren Inhalt werden aufbereitet und gespeichert.
- (2) **Indexierung:** Die Informationen werden in einer Datenstruktur gespeichert, mit deren Hilfe bei Eingabe eines Suchworts in "Echtzeit" alle Webseiten ermittelt werden können, die das Suchwort enthalten.
- (3) **Bewertung der Webseiten:** Die ausgewählten Webseiten werden im Hinblick auf ihren Informationsgehalt (hinsichtlich möglicher Suchworte sowie hinsichtlich ihrer generellen Bedeutung im Internet) bewertet.

¹Quelle: <http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html>; zuletzt besucht am 14.12.2011.

Zu jeder vom Crawler gefundenen Webseite wird die URL (d.h. die Adresse) sowie der Inhalt der Webseite gespeichert.

Der Inhalt der Webseite wird analysiert und es werden Informationen darüber gespeichert, welches Wort mit welcher Häufigkeit und an welchen Positionen (etwa: im Titel, als Überschrift, im Fließtext, mit welcher Schriftgröße etc.) in der Webseite vorkommt. Diese Informationen werden im so genannten **Index** gespeichert.

Index

Außerdem werden die Links, die auf Webseiten angegeben sind, analysiert. Enthält Webseite i einen Link auf eine Webseite j , so wird der Text, mit dem der Link beschriftet ist, im zu j gehörenden Index-Eintrag abgelegt. Diese Linkbeschriftungen geben wertvolle Hinweise darüber, welche Informationen die Webseite j enthält.

invertierter
Index

Aus dem Index wird der so genannte **invertierte Index** generiert. Dies ist eine Datenstruktur, die zu jedem möglichen Suchwort eine Liste aller Webseiten angibt, die dieses Suchwort enthalten. Dabei werden jeweils auch Zusatzinformationen gespeichert, die die Wichtigkeit des Suchworts innerhalb der Webseite beschreiben, z.B. die Häufigkeit des Stichworts, seine Position und Schriftgröße innerhalb der Webseite sowie das Vorkommen des Stichworts in Beschriftungen von Links *auf* die Webseite.

Die **Link-Struktur** des Internets kann man durch einen gerichteten Graphen modellieren, bei dem jede Webseite (d.h. jede URL) durch einen Knoten repräsentiert wird, und bei dem es eine Kante von Knoten i zu Knoten j gibt, wenn die Webseite i einen Link auf Webseite j enthält. Dieser Graph wird **Link-Index** oder **Web-Graph** genannt. Der Web-Graph wird üblicherweise als Adjazenzliste gespeichert.

Link-Index
Web-Graph

Bearbeitung von Such-Anfragen:

Bei Eingabe einer Liste von Such-Stichworten soll die Suchmaschine die hinsichtlich dieser Stichworte informativsten Webseiten finden und diese sortiert nach ihrer Relevanz anzeigen. Dabei werden folgende Kriterien berücksichtigt:

- (1) die Häufigkeit und Positionierung der Suchbegriffe auf der jeweiligen Webseite sowie in der Beschriftung von Links, die auf diese Webseite verweisen, und
- (2) die grundlegende Bedeutung einer Webseite.

Für (1) können Methoden aus dem Bereich **Information Retrieval** verwendet werden; Details dazu finden sich z.B. in Kapitel 6 von [21].

Für (2) wird die Link-Struktur des Internets, d.h. der Web-Graph berücksichtigt. Als Rechtfertigung für die Güte dieses Ansatzes, geht man von der folgenden Annahme aus: Wenn eine Webseite i einen Link auf eine Webseite j enthält, dann

- gibt es eine inhaltliche Beziehung zwischen beiden Webseiten, und
- der Autor der Webseite i hält die Informationen auf Webseite j für wertvoll.

Es gibt verschiedene Verfahren, die Maße für die grundlegende Bedeutung einer Webseite liefern, beispielsweise das von *Google* genutzte **Page-Rank** Verfahren von Brin und Page [3] oder die **HITS** (Hypertext Induced Topic Search) Methode von Kleinberg [16]. Beide Ansätze versuchen, die in der Link-Struktur manifestierte "relative Wertschätzung" zwischen einzelnen Webseiten in eine "grundlegende Bedeutung" der Webseiten umzurechnen. Details zu den beiden Verfahren finden sich in dem Buch [18].

Bei der Bearbeitung einer Suchanfrage, bei der eine Liste s von Such-Stichworten eingegeben wird, wird dann unter Verwendung von (1) und (2) jeder Webseite i ein Wert $Score(i, s)$ zugeordnet, der als Maß für die Relevanz der Webseite i hinsichtlich der Suchanfrage s dient. Als

Trefferliste gibt die Suchmaschine dann eine Liste aller Webseiten aus, deren Score über einer bestimmten Schranke liegt und sortiert die Liste so, dass die Webseiten mit dem höchsten Score am weitesten oben stehen. Wie der Wert $Score(i, s)$ gewählt wird, ist Betriebsgeheimnis der einzelnen Betreiber von Suchmaschinen.

Im Rest dieses Kapitels werden wir uns anhand des Page-Rank Verfahrens etwas genauer ansehen, wie die “grundlegende Bedeutung” einer Webseite modelliert und berechnet werden kann.

5.2 Der Page-Rank einer Webseite

Der Page-Rank liefert ein Maß für die “grundlegende Bedeutung” einer Webseite, das allein also aus der Link-Struktur des Internets bestimmt wird, ohne dass der textuelle Inhalt einer Webseite dabei berücksichtigt wird.

Wir schreiben im Folgenden $G = (V, E)$, um den Web-Graphen zu bezeichnen. Der Einfachheit halber nehmen wir an, dass die Webseiten mit den Zahlen $1, \dots, n$ durchnummeriert sind (wobei $n = |V|$ ist), und dass $V = \{1, 2, \dots, n\}$ ist.

Jeder Knoten von G repräsentiert eine Webseite, und jede Kante $(i, j) \in E$ modelliert einen Link von Webseite i auf Webseite j . Für jeden Knoten $i \in V$ sei

$$a_i := \text{Aus-Grad}_G(i)$$

der Ausgangsgrad von i in G . D.h. a_i ist die Anzahl der Hyperlinks, die von der Webseite i auf andere Webseiten verweisen. Für eine Webseite $j \in V$ schreiben wir $\text{Vor}_G(j)$, um die Menge aller Webseiten zu bezeichnen, die einen Link auf j enthalten, d.h.

$$\text{Vor}_G(j) = \{i \in V : (i, j) \in E\}.$$

Die Elemente in $\text{Vor}_G(j)$ werden **Vorgänger** von j genannt.

Die “grundlegende Bedeutung” einer Webseite i wird im Folgenden durch eine Zahl PR_i modelliert, dem so genannten **Page-Rank** von i . Der Wert PR_i soll die Qualität (im Sinne von “Renommee” oder “Ansehen”) von Webseite i widerspiegeln; die Zahl PR_i soll umso größer sein, je höher das Renommee der Webseite i ist. Das Renommee (und damit der Wert PR_j) einer Webseite j wird als hoch bewertet, wenn viele Webseiten i mit hohem Page-Rank PR_i einen Link auf die Seite j enthalten. Die Werte PR_i , die allen Webseiten $i \in V$ zugeordnet werden, werden daher so gewählt, dass folgendes gilt:

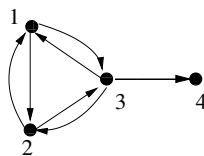
PR_i
Page-Rank

Eine Webseite i mit a_i ausgehenden Links “vererbt” ihren Page-Rank an jede Webseite j mit $(i, j) \in E$ um den Anteil $\frac{\text{PR}_i}{a_i}$.

Mit dieser Sichtweise müsste also für alle $j \in V$ mit $\text{Vor}_G(j) \neq \emptyset$ gelten:

$$\text{PR}_j = \sum_{i \in \text{Vor}_G(j)} \frac{\text{PR}_i}{a_i}. \quad (5.1)$$

Ein Problem stellen hierbei Knoten dar, deren Ausgangsgrad 0 ist, da solche Knoten ihren Page-Rank nicht an andere Knoten weitervererben und daher zu Werten PR_i führen können, die kein sinnvolles Maß für die Bedeutung einer Webseite liefern. Als Beispiel betrachte man den folgenden Graphen $G = (V, E)$:

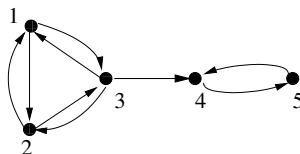


Die einzigen Werte $PR_1, PR_2, PR_3, PR_4 \in \mathbb{R}$, die die Gleichung (5.1) erfüllen, sind $PR_1 = PR_2 = PR_3 = PR_4 = 0$. Diese Werte spiegeln aber nicht die intuitive “grundlegende Bedeutung” wider, die man den Webseiten 1, 2, 3 und 4 zuordnen würde.

Senke

Im Folgenden werden **Knoten vom Ausgangsgrad 0** auch **Senken** genannt. Zur Bestimmung des Page-Ranks betrachtet man in der Regel nur **Graphen ohne Senken**, d.h. gerichtete Graphen, bei denen jeder Knoten einen Ausgangsgrad ≥ 1 hat. Natürlich gibt es keine Garantie, dass der Web-Graph keine Senken besitzt. Die Autoren von [3, 23] schlagen zwei Möglichkeiten vor, den Web-Graphen in einen Graphen ohne Senken zu transformieren: Die eine Möglichkeit ist, von jeder Senke Kanten zu *allen* Knoten hinzuzufügen. Die andere Möglichkeit ist, alle Senken zu löschen und dies rekursiv so lange zu tun, bis ein Graph übrig bleibt, der keine Senke besitzt. Wir nehmen im Folgenden an, dass eine dieser beiden Transformationen durchgeführt wurde und dass der Web-Graph durch einen endlichen gerichteten Graphen $G = (V, E)$ repräsentiert wird, der keine Senke besitzt.

Ein weiteres Problem stellen Knotenmengen dar, die unter sich zwar verbunden sind, die aber keine Kante zu einem anderen Knoten des Graphen G enthalten. Als einfaches Beispiel betrachten wir den folgenden Graphen $G = (V, E)$:



Man kann sich leicht davon überzeugen, dass Werte $PR_1, PR_2, PR_3, PR_4, PR_5 \in \mathbb{R}$ genau dann die Gleichung (5.1) erfüllen, wenn $PR_1 = PR_2 = PR_3 = 0$ und $PR_4 = PR_5$ ist. Ähnlich wie im vorherigen Beispiel spiegeln diese Werte nicht die intuitive “grundlegende Bedeutung” wider, die man den Webseiten 1–5 zuordnen würde. D.h. die durch die Gleichung (5.1) gegebenen Werte PR_1, \dots, PR_5 liefern kein sinnvolles Maß, um die grundlegende Bedeutung der einzelnen Webseiten zu bewerten.

Um dieses Problem zu vermeiden, wird die Vererbung von PR_i auf die Nachfolgeseiten j mit $(i, j) \in E$ meistens um einen **Dämpfungsfaktor** d mit $0 \leq d \leq 1$ abgeschwächt. Dies wird in der folgenden Definition präzisiert.

Definition 5.1 (Page-Rank-Eigenschaft).

Dämpfungsfaktor

Sei d eine reelle Zahl mit $0 \leq d \leq 1$. Die Zahl d wird im Folgenden **Dämpfungsfaktor** genannt. Sei $G = (V, E)$ ein gerichteter Graph, der keine Senke besitzt, und sei $n := |V| \in \mathbb{N}_{>0}$ und $V = \{1, \dots, n\}$.

Page-Rank-Eigenschaft bezüglich d

Für alle $i, j \in V$ sei $a_i := \text{Aus-Grad}_G(i)$ und $\text{Vor}_G(j) := \{i \in V : (i, j) \in E\}$.

Ein Tupel $PR = (PR_1, \dots, PR_n) \in \mathbb{R}^n$ hat die **Page-Rank-Eigenschaft bezüglich d** , wenn für alle $j \in V$ gilt:

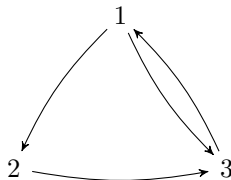
$$PR_j = \frac{1-d}{n} + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{PR_i}{a_i}. \tag{5.2}$$

Beachte: Für den Dämpfungsfaktor $d = 1$ erhält man gerade die Gleichung (5.1).

Für den Dämpfungsfaktor $d = 0$ ist $\text{PR}_1 = \text{PR}_2 = \dots = \text{PR}_n = \frac{1}{n}$.

In [3] wird empfohlen, den Wert $d = 0.85 = \frac{17}{20}$ zu wählen.

Beispiel 5.2. Zur Veranschaulichung der Page-Rank-Eigenschaft betrachten wir den Dämpfungsfaktor $d := \frac{1}{2}$ und den folgenden Graphen $G = (V, E)$:



Wir suchen ein Tupel $\text{PR} = (\text{PR}_1, \text{PR}_2, \text{PR}_3)$ von reellen Zahlen, das die Page-Rank-Eigenschaft bzgl. $d = \frac{1}{2}$ hat, d.h. es gilt:

$$(1) \quad \text{PR}_1 = \frac{1}{2 \cdot 3} + \frac{1}{2} \cdot \frac{\text{PR}_3}{1}$$

$$(2) \quad \text{PR}_2 = \frac{1}{2 \cdot 3} + \frac{1}{2} \cdot \frac{\text{PR}_1}{2}$$

$$(3) \quad \text{PR}_3 = \frac{1}{2 \cdot 3} + \frac{1}{2} \cdot \left(\frac{\text{PR}_1}{2} + \frac{\text{PR}_2}{1} \right).$$

Die Werte PR_1 , PR_2 und PR_3 können wir daher finden, indem wir das Lineare Gleichungssystem lösen, das aus den folgenden drei Gleichungen besteht:

$$(1) \quad 1 \cdot \text{PR}_1 - \frac{1}{2} \cdot \text{PR}_3 = \frac{1}{6}$$

$$(2) \quad -\frac{1}{4} \cdot \text{PR}_1 + 1 \cdot \text{PR}_2 = \frac{1}{6}$$

$$(3) \quad -\frac{1}{4} \cdot \text{PR}_1 - \frac{1}{2} \cdot \text{PR}_2 + 1 \cdot \text{PR}_3 = \frac{1}{6}$$

Die Auflösung dieses linearen Gleichungssystems (z.B. mittels **Gauß-Elimination**) liefert die Werte

$$\text{PR}_1 = \frac{14}{39}, \quad \text{PR}_2 = \frac{10}{39}, \quad \text{PR}_3 = \frac{15}{39}.$$

□ Ende Beispiel 5.2

Auf die gleiche Art wie in diesem Beispiel erhält man auch für den Web-Graphen und einen geeigneten Dämpfungsfaktor d ein entsprechendes lineares Gleichungssystem. Um den Page-Rank der einzelnen Webseiten zu berechnen, müssen wir “nur” dieses lineare Gleichungssystem lösen. Dabei stellen sich folgende Probleme:

- (1) Zunächst ist völlig unklar, ob dieses lineare Gleichungssystem überhaupt eine Lösung besitzt, und falls ja, ob die Lösung eindeutig ist. Anhand von Definition 5.1 ist nämlich prinzipiell auch denkbar, dass es gar kein Tupel gibt, das die Page-Rank-Eigenschaft bzgl. d hat, oder dass es mehrere verschiedene Tupel gibt, die die Page-Rank-Eigenschaft bzgl. d besitzen.
- (2) Das lineare Gleichungssystem hat n Unbekannte, wobei n die Anzahl der Webseiten im Internet ist — und diese Zahl ist enorm groß. Um den Page-Rank aller Webseiten zu bestimmen, benötigen daher ein extrem effizientes Verfahren zum Lösen dieses linearen Gleichungssystems.

In den folgenden beiden Abschnitten werden wir sehen, dass die Theorie der **Markov-Ketten** uns hilft, diese Probleme zu lösen. Dazu ist die im folgenden Abschnitt dargestellte Sichtweise auf den Page-Rank sehr hilfreich.

5.3 Der Zufalls-Surfer

Wir nehmen an, dass der Webgraph durch einen gerichteten Graphen $G = (V, E)$ mit Knotenmenge $V = \{1, \dots, n\}$ repräsentiert wird, der keine Senke besitzt. Des Weiteren sei d eine beliebige reelle Zahl mit $0 \leq d \leq 1$.

Zufalls-Surfer

Wir betrachten einen **Zufalls-Surfer** (englisch: **random surfer**), der auf einer beliebigen Webseite beginnt und beliebige Links verfolgt, ohne dabei auf Inhalte zu achten. Wenn der Zufalls-Surfer auf einer Webseite i ist, so wählt er

- mit Wahrscheinlichkeit d einen Link, der von Seite i ausgeht. Hierbei wird dann jeder der $a_i = \text{Aus-Grad}_G(i)$ ausgehenden Links mit derselben Wahrscheinlichkeit $\frac{d}{a_i}$ ausgewählt.
- mit Wahrscheinlichkeit $(1 - d)$ eine **beliebige** Webseite im Web-Graphen. Hierbei wird dann jede der n Webseiten mit derselben Wahrscheinlichkeit $\frac{1-d}{n}$ ausgewählt.

Für alle $i, j \in V$ gibt daher

$$p_{i,j} := \begin{cases} \frac{1-d}{n} + \frac{d}{a_i} & , \text{ falls } (i, j) \in E \\ \frac{1-d}{n} & , \text{ falls } (i, j) \notin E \end{cases} \quad (5.3)$$

die Wahrscheinlichkeit an, mit der der Zufalls-Surfer in einem Schritt von Seite i zu Seite j wechselt. Diese Wahrscheinlichkeiten, mit denen sich der Zufalls-Surfer von Knoten zu Knoten bewegt, lassen sich kompakt durch die folgende Matrix darstellen.

Definition 5.3 (Die Page-Rank-Matrix $P(G, d)$).

Sei $d \in \mathbb{R}$ mit $0 \leq d \leq 1$, sei $n \in \mathbb{N}_{>0}$ und sei $G = (V, E)$ mit $V = \{1, \dots, n\}$ ein gerichteter Graph ohne Senke. Für jedes $i \in V$ sei $a_i := \text{Aus-Grad}_G(i)$. Die **Page-Rank-Matrix** ist die $n \times n$ -Matrix

Page-Rank-Matrix

$$P(G, d) := \begin{pmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & & \vdots \\ p_{n,1} & \cdots & p_{n,n} \end{pmatrix},$$

wobei für alle $i, j \in V$ der Eintrag in Zeile i und Spalte j der in Gleichung (5.3) festgelegte Wert $p_{i,j}$ ist. Wir schreiben auch kurz $(p_{i,j})_{i,j=1,\dots,n}$, um die Matrix $P(G, d)$ zu bezeichnen.

Beispiel 5.4. Für den Wert $d = \frac{1}{2}$ und den Graphen G aus Beispiel 5.2 ist beispielsweise $p_{1,1} = \frac{1}{6}$, $p_{1,2} = \frac{1}{6} + \frac{1}{4} = \frac{5}{12}$, $p_{2,3} = \frac{1}{6} + \frac{1}{2} = \frac{2}{3}$ und insgesamt

$$P(G, d) = \begin{pmatrix} \frac{1}{6} & \frac{5}{12} & \frac{5}{12} \\ \frac{1}{6} & \frac{1}{6} & \frac{2}{3} \\ \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{pmatrix}.$$

Um den Zusammenhang zwischen dem Zufalls-Surfer, der Page-Rank-Matrix und Tupeln mit der Page-Rank-Eigenschaft beschreiben zu können, benötigen wir folgende Notation für das Rechnen mit Matrizen.

Definition 5.5 (Vektor-Matrix-Produkt).

Sei $n \in \mathbb{N}_{>0}$, und für alle $i, j \in \{1, \dots, n\}$ sei $p_{i,j}$ eine reelle Zahl. Sei $P := (p_{i,j})_{i,j=1,\dots,n}$ die $n \times n$ -Matrix, die in Zeile i und Spalte j den Eintrag $p_{i,j}$ hat (für alle $i, j \in \{1, \dots, n\}$). Ist $X = (X_1, \dots, X_n)$ ein Tupel aus n reellen Zahlen, so ist das **Vektor-Matrix-Produkt**

Vektor-Matrix-
Produkt
 $X \cdot P$

$$X \cdot P$$

das Tupel $Y = (Y_1, \dots, Y_n) \in \mathbb{R}^n$, bei dem für jedes $j \in \{1, \dots, n\}$ gilt:

$$Y_j := \sum_{i=1}^n X_i \cdot p_{i,j}.$$

Beispiel 5.6. Sei $P := P(G, d)$ die Matrix aus Beispiel 5.4 und sei $X := (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. Dann gilt:

$$\begin{aligned} X \cdot P &= \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right) \cdot \begin{pmatrix} \frac{1}{6} & \frac{5}{12} & \frac{5}{12} \\ \frac{1}{6} & \frac{1}{6} & \frac{2}{3} \\ \frac{2}{3} & \frac{1}{6} & \frac{1}{6} \end{pmatrix} \\ &= \left(\frac{1}{3} \cdot \frac{1}{6} + \frac{1}{3} \cdot \frac{1}{6} + \frac{1}{3} \cdot \frac{2}{3}, \frac{1}{3} \cdot \frac{5}{12} + \frac{1}{3} \cdot \frac{1}{6} + \frac{1}{3} \cdot \frac{1}{6}, \frac{1}{3} \cdot \frac{5}{12} + \frac{1}{3} \cdot \frac{2}{3} + \frac{1}{3} \cdot \frac{1}{6}\right) \\ &= \left(\frac{1}{3}, \frac{1}{4}, \frac{5}{12}\right). \end{aligned}$$

Der folgende Satz beschreibt den genauen Zusammenhang zwischen Zufalls-Surfer, Page-Rank-Matrix und Tupeln mit der Page-Rank-Eigenschaft.

Satz 5.7. Sei $d \in \mathbb{R}$ mit $0 \leq d < 1$, sei $n \in \mathbb{N}_{>0}$ und sei $G = (V, E)$ ein gerichteter Graph mit $V = \{1, \dots, n\}$, der keine Senke besitzt. Dann gilt:

- (a) Ist $\text{PR} = (\text{PR}_1, \dots, \text{PR}_n) \in \mathbb{R}^n$ ein Tupel, das die Page-Rank-Eigenschaft bzgl. d besitzt, so ist $\sum_{i=1}^n \text{PR}_i = 1$.
- (b) Für jedes Tupel $X = (X_1, \dots, X_n) \in \mathbb{R}^n$ mit $\sum_{i=1}^n X_i = 1$ gilt:

$$X \text{ besitzt die Page-Rank-Eigenschaft bzgl. } d \iff X \cdot P(G, d) = X.$$

Beweis:

- (a) Sei $\text{PR} = (\text{PR}_1, \dots, \text{PR}_n) \in \mathbb{R}^n$ ein Tupel, das die Page-Rank-Eigenschaft bzgl. d besitzt. D.h. es gilt f.a. $j \in \{1, \dots, n\}$, dass

$$\text{PR}_j = \frac{1-d}{n} + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{\text{PR}_i}{a_i}.$$

Somit gilt:

$$\begin{aligned}
\sum_{j=1}^n \text{PR}_j &= \sum_{j=1}^n \left(\frac{1-d}{n} + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{\text{PR}_i}{a_i} \right) = n \cdot \frac{1-d}{n} + d \cdot \sum_{j=1}^n \sum_{i \in \text{Vor}_G(j)} \frac{\text{PR}_i}{a_i} \\
&= (1-d) + d \cdot \sum_{(i,j) \in E} \frac{\text{PR}_i}{a_i} = (1-d) + d \cdot \sum_{i=1}^n \sum_{j: (i,j) \in E} \frac{\text{PR}_i}{a_i} \\
&\stackrel{\text{G ohne Senke}}{=} (1-d) + d \cdot \sum_{i=1}^n \left(a_i \cdot \frac{\text{PR}_i}{a_i} \right) = (1-d) + d \cdot \sum_{i=1}^n \text{PR}_i \\
&= (1-d) + d \cdot \sum_{j=1}^n \text{PR}_j.
\end{aligned}$$

Insbesondere gilt also:

$$(1-d) \cdot \sum_{j=1}^n \text{PR}_j = (1-d). \quad (5.4)$$

Wegen $d \neq 1$ ist $(1-d) \neq 0$, und daher erhalten wir aus Gleichung (5.4), dass $\sum_{j=1}^n \text{PR}_j = 1$ ist. Dies schließt den Beweis von Teil (a) ab.

- (b) Sei $X = (X_1, \dots, X_n) \in \mathbb{R}^n$ mit $\sum_{i=1}^n X_i = 1$. Sei $Y = (Y_1, \dots, Y_n)$ so dass $X \cdot P(G, d) = Y$. Dann gilt gemäß Definition 5.5 und Definition 5.3 für jedes $j \in \{1, \dots, n\}$, dass

$$\begin{aligned}
Y_j &= \sum_{i=1}^n X_i \cdot p_{i,j} \stackrel{\text{Gl. (5.3)}}{=} \sum_{i=1}^n X_i \cdot \frac{1-d}{n} + \sum_{i \in \text{Vor}_G(j)} X_i \cdot \frac{d}{a_i} \\
&= \frac{1-d}{n} \cdot \sum_{i=1}^n X_i + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{X_i}{a_i} \\
&\stackrel{\sum_{i=1}^n X_i = 1}{=} \frac{1-d}{n} + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{X_i}{a_i},
\end{aligned}$$

d.h. es gilt

$$Y_j = \frac{1-d}{n} + d \cdot \sum_{i \in \text{Vor}_G(j)} \frac{X_i}{a_i}. \quad (5.5)$$

Aus Definition 5.1 zusammen mit Gleichung (5.5) folgt:

$$\begin{aligned}
&X \text{ besitzt die Page-Rank-Eigenschaft bzgl. } d \\
\iff &\text{f.a. } j \in \{1, \dots, n\} \text{ gilt: } X_j = Y_j \\
\iff &X \cdot P(G, d) = X.
\end{aligned}$$

□

Beachte: Für Satz 5.7 (a) ist wichtig, dass $d \neq 1$ ist und dass G keine Senke besitzt.

Notation 5.8 (Eigenvektor).

Ein Vektor $X = (X_1, \dots, X_n)$ heißt **linker Eigenvektor zum Eigenwert 1** der $n \times n$ -Matrix P , falls gilt: $X \cdot P = X$ und $X \neq (0, \dots, 0)$. Eigenvektor

Satz 5.7 besagt also, dass ein Tupel $\text{PR} = (\text{PR}_1, \dots, \text{PR}_n) \in \mathbb{R}^n$ genau dann die Page-Rank-Eigenschaft bzgl. d besitzt, wenn es ein linker Eigenvektor zum Eigenwert 1 der Matrix $P(G, d)$ ist, für den $\sum_{i=1}^n \text{PR}_i = 1$ ist.

Diese Sichtweise auf den Page-Rank sowie die im folgenden Abschnitt vorgestellte Theorie der Markov-Ketten helfen uns, um die beiden am Ende von Abschnitt 5.2 gestellten Probleme zu lösen.

5.4 Markov-Ketten

Markov-Ketten sind nach dem russischen Mathematiker Andrei A. Markov (1856–1922) benannt. In der Literatur werden unterschiedliche Schreibweisen des Namens verwendet, z.B. Markov, Markow oder Markoff.

Definition 5.9 (Markov-Kette).

Eine (**homogene**) **Markov-Kette** mit **Übergangsmatrix** P wird durch eine $n \times n$ -Matrix Markov-Kette
Übergangsmatrix

$$P = (p_{i,j})_{i,j=1,\dots,n}$$

mit $n \in \mathbb{N}_{>0}$ beschrieben, für die gilt:

- (1) $p_{i,j} \geq 0$ für alle $i, j \in \{1, \dots, n\}$, und
- (2) für jede Zeile $i \in \{1, \dots, n\}$ gilt: $\sum_{j=1}^n p_{i,j} = 1$.

Eine Matrix P , die die Eigenschaften (1) und (2) besitzt, wird auch **stochastische Matrix** stochastische Ma-
trix genannt.

Der **zu P gehörende Graph** ist der gerichtete Graph mit Knotenmenge $V = \{1, \dots, n\}$, so dass für alle $i, j \in \{1, \dots, n\}$ gilt: Es gibt in G genau dann eine Kante von i nach j , wenn $p_{i,j} > 0$ ist. Den Eintrag $p_{i,j}$ in Zeile i und Spalte j von P kann man als Wahrscheinlichkeit dafür auffassen, dass ein Zufalls-Surfer im Graphen G in einem Schritt von Knoten i zu Knoten j springt.

Beispiel 5.10. Sei $G = (V, E)$ ein beliebiger gerichteter Graph mit Knotenmenge $V = \{1, \dots, n\}$ (für $n := |V| \in \mathbb{N}_{>0}$), der keine Senke besitzt. Sei d eine reelle Zahl mit $0 \leq d < 1$ und sei $P := P(G, d)$ die zugehörige Page-Rank-Matrix.

Gemäß der Definition von $P(G, d)$ ist $p_{i,j} > 0$ für alle $i, j \in \{1, \dots, n\}$ (dazu beachte man, dass $0 \leq d < 1$ ist). Außerdem gilt für jede Zeile $i \in \{1, \dots, n\}$, dass

$$\sum_{j=1}^n p_{i,j} = \sum_{j=1}^n \frac{1-d}{n} + \sum_{j: (i,j) \in E} \frac{d}{a_i} \stackrel{G \text{ ohne Senke}}{=} (1-d) + a_i \cdot \frac{d}{a_i} = 1.$$

Somit ist P eine stochastische Matrix, die eine Markov-Kette beschreibt. Für jedes $i, j \in \{1, \dots, n\}$ gibt der Wert $p_{i,j}$ die Wahrscheinlichkeit dafür an, dass der Zufalls-Surfer in einem Schritt von Webseite i zu Webseite j springt.

Da $p_{i,j} > 0$ ist, ist der zu P gehörende Graph der **vollständige gerichtete Graph** auf n Knoten, d.h. der Graph mit Knotenmenge $V = \{1, \dots, n\}$ und Kantenmenge $V \times V$. Diesen Graphen bezeichnen wir im Folgenden mit \vec{K}_n .

Die Theorie der Markov-Ketten und der stochastischen Matrizen wurde in der Literatur gut untersucht (siehe [13, 8]). Insbesondere ist folgendes bekannt (vgl. [7]):

Satz 5.11. Sei $n \in \mathbb{N}_{>0}$ und sei $P = (p_{i,j})_{i,j=1,\dots,n}$ eine stochastische Matrix, bei der für alle $i, j \in \{1, \dots, n\}$ gilt: $p_{i,j} > 0$. Dann gibt es genau ein Tupel $X = (X_1, \dots, X_n) \in \mathbb{R}^n$ mit $\sum_{i=1}^n X_i = 1$, das ein linker Eigenvektor zum Eigenwert 1 von P ist. Dieses Tupel hat die Eigenschaft, dass für jedes $i \in \{1, \dots, n\}$ der Wert $X_i > 0$ ist.

Einen Beweis dieses Satzes zu geben, würde den Rahmen dieses Vorlesungsskripts sprengen. Man beachte, dass sich aus der Kombination von Satz 5.11, Beispiel 5.10 und Satz 5.7 die Lösung des am Ende von Abschnitt 5.2 genannten Problems (1) ergibt.

Folgerung 5.12 (Lösung von Problem (1) auf Seite 165).

Ist $G = (V, E)$ ein gerichteter Graph mit $V = \{1, \dots, n\}$ (für $n \in \mathbb{N}_{>0}$), der keine Senke besitzt, und ist $d \in \mathbb{R}$ ein Dämpfungsfaktor mit $0 \leq d < 1$, so gibt es genau ein Tupel $\text{PR} = (\text{PR}_1, \dots, \text{PR}_n) \in \mathbb{R}^n$, das die Page-Rank-Eigenschaft bezüglich d besitzt. Für dieses Tupel gilt: $\text{PR}_i > 0$ für alle $i \in \{1, \dots, n\}$ und $\sum_{i=1}^n \text{PR}_i = 1$.

5.5 Die effiziente Berechnung des Page-Rank

Um zu sehen, dass die Theorie der Markov-Ketten uns auch eine Lösung für Problem (2) auf Seite 165 liefert, schauen wir uns die Bewegungen des Zufalls-Surfers auf dem Web-Graphen etwas genauer an.

Für unsere Betrachtungen ist folgendermaßen definierte Begriff einer **Verteilung** sehr nützlich.

Definition 5.13. Sei $n \in \mathbb{N}_{>0}$.

Eine **Verteilung** auf $V = \{1, \dots, n\}$ ist ein Tupel $X = (X_1, \dots, X_n) \in \mathbb{R}^n$, für das gilt:

- (1) für alle $i \in \{1, \dots, n\}$ ist $X_i \geq 0$ und
- (2) $\sum_{i=1}^n X_i = 1$.

Ist G ein gerichteter Graph mit Knotenmenge $V = \{1, \dots, n\}$ und ist $X = (X_1, \dots, X_n)$ eine Verteilung auf V , so fassen wir für jedes $i \in V$ die Zahl X_i als Wahrscheinlichkeit dafür auf, dass ein Zufalls-Surfer in G sich auf Knoten i befindet.

Beobachtung 5.14. Sei $n \in \mathbb{N}_{>0}$ und sei $P = (p_{i,j})_{i,j=1,\dots,n}$ eine stochastische Matrix. Ist $X = (X_1, \dots, X_n)$ eine Verteilung auf $V := \{1, \dots, n\}$, so gibt das Tupel $Y = (Y_1, \dots, Y_n)$ mit

$$X \cdot P = Y$$

folgendes an: Wenn wir in dem zu P gehörenden Graphen für jedes $i \in V$ den Zufalls-Surfer mit

Wahrscheinlichkeit X_i auf Knoten i beginnen lassen, so gibt für jedes $j \in V$ die Zahl

$$Y_j = \sum_{i=1}^n X_i \cdot p_{i,j}$$

die Wahrscheinlichkeit dafür an, dass der Zufalls-Surfer sich nach einem Schritt auf Knoten j befindet.

Rekursiv können so wir für jedes $k \in \mathbb{N}$ eine Verteilung $X^{(k)} = (X_1^{(k)}, \dots, X_n^{(k)})$ angeben, so dass für jedes $j \in V$ der Wert $X_j^{(k)}$ die Wahrscheinlichkeit dafür angibt, dass der Zufalls-Surfer sich nach k Schritten auf Knoten j befindet. Dazu wählen wir

$$X^{(0)} := X \quad \text{und} \quad X^{(k+1)} := X^{(k)} \cdot P, \quad \text{f.a. } k \in \mathbb{N}.$$

Unter Verwendung des in der folgenden Definition gegebenen Produkts von Matrizen erhalten wir per Induktion nach k , dass für alle $k \in \mathbb{N}_{>0}$ gilt:

$$X^{(k)} = X \cdot P^k.$$

Definition 5.15 (Matrix-Produkt). Sei $n \in \mathbb{N}_{>0}$, und für alle $i, j \in \{1, \dots, n\}$ sei $a_{i,j} \in \mathbb{R}$ und $b_{i,j} \in \mathbb{R}$. Wir betrachten die beiden $n \times n$ -Matrizen

$$A := (a_{i,j})_{i,j=1,\dots,n} \quad \text{und} \quad B := (b_{i,j})_{i,j=1,\dots,n}.$$

(a) Das **Produkt**

$$A \cdot B$$

ist die $n \times n$ -Matrix $C = (c_{i,j})_{i,j=1,\dots,n}$, die für alle $i, j \in \{1, \dots, n\}$ in Zeile i und Spalte j den Eintrag

$$c_{i,j} := \sum_{\ell=1}^n a_{i,\ell} \cdot b_{\ell,j}$$

hat.

(b) Für jede Zahl $k \in \mathbb{N}_{>0}$ ist die $n \times n$ -Matrix A^k folgendermaßen rekursiv definiert:

$$A^1 := A \quad \text{und} \quad A^{k+1} := A \cdot A^k \quad (\text{für alle } k \in \mathbb{N}).$$

Für alle $i, j \in \{1, \dots, n\}$ schreiben wir $(A^k)_{i,j}$, um den Eintrag in Zeile i und Spalte j der Matrix A^k zu bezeichnen.

Anhand dieser Definition sieht man leicht, dass folgendes gilt:

Beobachtung 5.16. Ist $n \in \mathbb{N}_{>0}$ und ist $P = (p_{i,j})_{i,j=1,\dots,n}$ eine stochastische Matrix, die eine Markov-Kette beschreibt, so können wir für jedes $k \in \mathbb{N}_{>0}$ den Eintrag $(P^k)_{i,j}$ in Zeile i und Spalte j der Matrix P^k als die Wahrscheinlichkeit dafür auffassen, dass der Zufalls-Surfer auf dem zu P gehörenden Graphen innerhalb von genau k Schritten von Knoten i zu Knoten j gelangt.

Zur effizienten Berechnung des Page-Ranks machen wir uns zu nutze, dass die durch die Page-Rank-Matrix $P(G, d)$ (für $0 \leq d < 1$) beschriebene Markov-Kette die folgende Eigenschaft hat:

ergodisch

Definition 5.17 (Ergodische Markov-Ketten).

Sei $n \in \mathbb{N}_{>0}$ und sei $P = (p_{i,j})_{i,j=1,\dots,n}$ eine stochastische Matrix. Die durch P beschriebene Markov-Kette heißt **ergodisch**, wenn für alle $i, i' \in \{1, \dots, n\}$ und alle $j \in \{1, \dots, n\}$ gilt: Die Grenzwerte

$$\lim_{k \rightarrow \infty} (P^k)_{i,j} \quad \text{und} \quad \lim_{k \rightarrow \infty} (P^k)_{i',j}$$

existieren und es gilt

$$\lim_{k \rightarrow \infty} (P^k)_{i,j} = \lim_{k \rightarrow \infty} (P^k)_{i',j} > 0.$$

Bemerkung 5.18 (eine Charakterisierung ergodischer Markov-Ketten).

Es ist bekannt (für einen Beweis sei auf [13] verwiesen), dass eine durch eine stochastische Matrix $P = (p_{i,j})_{i,j=1,\dots,n}$ gegebene Markov-Kette genau dann ergodisch ist, wenn sie **irreduzibel** und **aperiodisch** ist. Dabei heißt P

irreduzibel
aperiodisch

- **irreduzibel**, falls der zu P gehörende Graph stark zusammenhängend ist;
- **aperiodisch**, falls für jeden Knoten i im zu P gehörenden Graphen gilt: Der größte gemeinsame Teiler der Längen aller Wege von i nach i ist 1.

Falls $P = P(G, d)$ die Page-Rank-Matrix für einen Dämpfungsfaktor d mit $0 \leq d < 1$ und einen gerichteten Graphen G ist, der keine Senke besitzt, so wissen wir aus Beispiel 5.10, dass der zu P gehörende Graph der vollständige gerichtete Graph \vec{K}_n ist. Dieser ist offensichtlich irreduzibel und aperiodisch. Daher beschreibt die Page-Rank-Matrix $P(G, d)$ eine ergodische Markov-Kette.

Beobachtung 5.19 (Eigenschaften ergodischer Markov-Ketten).

Ist P eine stochastische Matrix, die eine ergodische Markov-Kette beschreibt, so gilt offensichtlich folgendes:

(1) Die Matrix

$$P' := \left(\lim_{k \rightarrow \infty} (P^k)_{i,j} \right)_{i,j=1,\dots,n} \tag{5.6}$$

ist wohldefiniert (da die Grenzwerte existieren), und

(2) alle Zeilen von P' sind identisch.

Wir schreiben $p' := (p'_1, \dots, p'_n)$, um die erste Zeile von P' zu bezeichnen. Die Matrix P' sieht daher folgendermaßen aus:

$$P' = \begin{pmatrix} p' \\ p' \\ \vdots \\ p' \end{pmatrix} = \begin{pmatrix} p'_1 & \cdots & p'_n \\ p'_1 & \cdots & p'_n \\ \vdots & & \vdots \\ p'_1 & \cdots & p'_n \end{pmatrix}.$$

Wegen Gleichung (5.6) gilt $P' \cdot P = P'$, und daher gilt insbesondere für die Verteilung p' , dass

$$p' \cdot P = p',$$

d.h. p' ist ein linker Eigenvektor zum Eigenwert 1 der Matrix P .

Notation: Eine Verteilung Y mit $Y \cdot P = Y$ wird auch **stationäre Verteilung** für P genannt. stationäre Verteilung

Für jede beliebige Verteilung $X = (X_1, \dots, X_n)$ gilt:

$$X \cdot P' = p', \quad (5.7)$$

denn für jedes $j \in V$ ist der j -te Eintrag im Tupel $X \cdot P'$ gerade die Zahl $\sum_{i=1}^n X_i \cdot p'_j = p'_j \cdot \sum_{i=1}^n X_i = p'_j$. Daher gilt:

- (a) $p' = (p'_1, \dots, p'_n)$ ist die **einzige** stationäre Verteilung, die P besitzt, und
- (b) wenn der Zufalls-Surfer im zu P gehörenden Graphen seinen Startknoten gemäß einer beliebigen Anfangsverteilung $X = (X_1, \dots, X_n)$ wählt und hinreichend viele Schritte macht, so ist für jedes $j \in V$ die Wahrscheinlichkeit, bei Knoten j zu landen beliebig nah bei p'_j . Die Wahl des Anfangsknotens ist für einen Zufalls-Surfer, der hinreichend lange surft, also ohne Belang.

Aufgrund der Gleichungen (5.6) und (5.7) erhalten wir:

$$p' \stackrel{(5.7)}{=} X \cdot P' \stackrel{(5.6)}{=} X \cdot \lim_{k \rightarrow \infty} P^k = \lim_{k \rightarrow \infty} (X \cdot P^k) = \lim_{k \rightarrow \infty} X^{(k)},$$

wobei $X^{(0)} := X$ und $X^{(k+1)} := X^{(k)} \cdot P$, f.a. $k \in \mathbb{N}$.

Um eine Näherung für das Tupel p' zu berechnen, können wir daher wie folgt vorgehen: Wir starten mit einer beliebigen Verteilung $X^{(0)} = X$ (etwa der **Gleichverteilung** $X = (\frac{1}{n}, \dots, \frac{1}{n})$) und berechnen nacheinander für $k = 1, 2, 3$ usw. das Tupel $X^{(k+1)} := X^{(k)} \cdot P$. Dieser Prozess wird beendet, sobald das Tupel $X^{(k+1)}$ sich nicht mehr viel vom Tupel $X^{(k)}$ unterscheidet, d.h. sobald für jedes $j \in \{1, \dots, n\}$ die Zahl $|X_j^{(k+1)} - X_j^{(k)}|$ kleiner als eine vorher festgelegte Schranke ε ist (wobei $X_j^{(k+1)}$ und $X_j^{(k)}$ der Eintrag in der j -ten Komponente von $X^{(k+1)}$ bzw. $X^{(k)}$ ist).

□Beobachtung 5.19

Folgerung 5.20 (Lösung von Problem (2) auf Seite 165).

Sei $P := P(G, d)$ die Page-Rank-Matrix für einen Dämpfungsfaktor d mit $0 \leq d < 1$ und einen gerichteten Graphen $G = (V, E)$ ohne Senke. Von Bemerkung 5.18 wissen wir, dass P ergodisch ist. Aus Beobachtung 5.19 folgt daher, dass die stationäre Verteilung p' von P das (eindeutig festgelegte) Tupel ist, das die Page-Rank-Eigenschaft bzgl. d besitzt. Das am Ende von Beobachtung 5.19 beschriebene Vorgehen liefert ein effizientes Verfahren, um eine Näherung für das Tupel p' zu berechnen.

Aus der Theorie der Markov-Ketten und den speziellen Eigenschaften der Page-Rank-Matrix $P(G, d)$ (für d mit $0 \leq d < 1$) ergibt sich, dass aufgrund des hohen Zusammenhangs des Web-Graphen die Folge der Tupel $X^{(k)}$ für $k = 0, 1, 2, 3$ usw. sehr schnell gegen die stationäre Verteilung p' konvergiert. Details dazu finden sich in [18, 13].

Für eine schnelle Berechnung des Vektor-Matrix-Produkts $X^{(k+1)} := X^{(k)} \cdot P(G, d)$ wird ausgenutzt, dass $P(G, d)$ viele identische Einträge der Form $\frac{1-d}{n}$ hat. Außerdem ist die Berechnung des Vektor-Matrix-Produkts sehr gut parallelisierbar. Details hierzu finden sich in [18].

Derzeit werden mehrere Tausend PCs eingesetzt, die mehrere Stunden zur Berechnung des Page-Ranks benötigen — was in Anbetracht der Tatsache, dass es mehrere Milliarden Webseiten gibt, erstaunlich gering ist.

5.6 Literaturhinweise

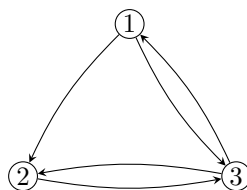
Zur vertiefenden Lektüre seien Kapitel 2 von [25] sowie das Buch [13] empfohlen, das eine Algorithmen-orientierte Einführung in die Theorie der Markov-Ketten gibt. Einen Überblick über die Architektur von Suchmaschinen gibt der Artikel [1]; Details zum Page-Rank und zum HITS Verfahren finden sich in dem Buch [18] sowie in den Originalarbeiten [3, 23, 16, 7]. Als Einführung ins Thema *Information Retrieval* sei das Buch [21] empfohlen. Das Buch [8] ist ein “Klassiker”, der eine umfassende Einführung in die Wahrscheinlichkeitstheorie (und insbesondere auch ins Thema Markov-Ketten) gibt.

Viele Informationen und Literaturhinweise zum Thema *Suchmaschinen* finden sich auf der Webseite von Martin Sauerhoffs Vorlesung *Internet Algorithmen* an der TU Dortmund; siehe <http://ls2-www.cs.uni-dortmund.de/lehre/winter200910/IntAlg/>. Ein kurzer und allgemein verständlicher Überblick über das Page-Rank Verfahren wird in dem Spiegel-Online Artikel *Wie Google mit Milliarden Unbekannten rechnet* von Holger Dambeck gegeben; siehe <http://www.spiegel.de/wissenschaft/mensch/0,1518,646448,00.html>.

Quellennachweis: Teile dieses Kapitels orientieren sich an [25].

5.7 Übungsaufgaben zu Kapitel 5

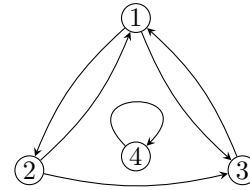
Aufgabe 5.1. Betrachten Sie den Web-Graph $G = (V, E)$, der aus den drei Webseiten 1, 2 und 3 besteht, die wie in der nebenstehenden Abbildung miteinander verlinkt sind. Die Entwickler des Page-Rank-Algorithmus und Gründer von Google, Larry Page und Sergey Brin, empfehlen einen Dämpfungsfaktor von $d := 0.85 = \frac{17}{20}$. Benutzen Sie für die folgenden Aufgaben ebenfalls diesen Dämpfungsfaktor.



- Berechnen Sie ähnlich wie in Beispiel (5.2) aus dem Skript die Page-Ranks PR_1 , PR_2 und PR_3 der drei Webseiten von G mit dem Dämpfungsfaktor d .
- Stellen Sie für den angegebenen Web-Graph G und den Dämpfungsfaktor d die Page-Rank-Matrix $P(G, d)$ auf.
- Sei P die Page-Rank-Matrix $P(G, d)$ aus Teilaufgabe (b). Angenommen der Zufalls-Surfer startet auf einer der drei Webseiten von G , wobei er jede Webseite gleichwahrscheinlich als Startpunkt wählen kann. Das bedeutet, dass die Anfangsverteilung für den Zufalls-Surfer durch $X^{(0)} := (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ beschrieben wird. Berechnen Sie die Wahrscheinlichkeitsverteilung des Zufalls-Surfers auf den Knoten von G nach einem Schritt (d.h. $X^{(1)}$), nach zwei Schritten (d.h. $X^{(2)}$) und nach drei Schritten (d.h. $X^{(3)}$). Dabei ist $X^{(1)} := X^{(0)} \cdot P$, $X^{(2)} := X^{(1)} \cdot P$ und $X^{(3)} := X^{(2)} \cdot P$.
- Gesucht ist ein Web-Graph $G' = (V', E')$ mit drei Webseiten, in dem jede Webseite auf mindestens eine Webseite verlinkt, die nicht sie selber ist. Zusätzlich soll der Zufalls-Surfer mit der Anfangsverteilung $X^{(0)} := (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ nach einem Schritt in G' genau die selbe Wahrscheinlichkeitsverteilung erreichen, es soll also $X^{(0)} \cdot P(G', d) = X^{(0)}$ gelten. Geben Sie einen solchen Graphen G' an und weisen Sie nach, dass $X^{(0)} \cdot P(G', d) = X^{(0)}$ gilt.

Aufgabe 5.2.

Betrachten Sie den Web-Graph $G = (V, E)$, der aus den vier Webseiten 1, 2, 3 und 4 besteht, die wie in der nebenstehenden Abbildung miteinander verlinkt sind. Benutzen Sie für die folgenden Aufgaben den Dämpfungsfaktor $d := \frac{1}{2}$.



- Berechnen Sie ähnlich wie in Beispiel 5.2 aus dem Skript die Page-Ranks PR_1, PR_2, PR_3 und PR_4 der vier Webseiten von G bezüglich des Dämpfungsfaktors d .
- Stellen Sie für den angegebenen Web-Graph G und den Dämpfungsfaktor d die Page-Rank-Matrix $P(G, d)$ auf.
- Sei P die Page-Rank-Matrix $P(G, d)$ aus Teilaufgabe (b). Wir nehmen an, der Zufalls-Surfer startet auf einer der vier Webseiten von G , wobei er jede Webseite gleichwahrscheinlich als Startpunkt wählen kann. Das bedeutet, dass die Anfangsverteilung für den Zufalls-Surfer durch $X^{(0)} := (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ beschrieben wird. Berechnen Sie die Wahrscheinlichkeitsverteilung des Zufalls-Surfers auf den Knoten von G nach einem Schritt (d.h. $X^{(1)}$), nach zwei Schritten (d.h. $X^{(2)}$) und nach drei Schritten (d.h. $X^{(3)}$). Dabei ist $X^{(1)} := X^{(0)} \cdot P$, $X^{(2)} := X^{(1)} \cdot P$ und $X^{(3)} := X^{(2)} \cdot P$.
- Gesucht ist ein Web-Graph $G' = (V', E')$ mit vier Webseiten, in dem jede Webseite auf mindestens eine Webseite verlinkt, die nicht sie selber ist. Zusätzlich soll der Zufalls-Surfer mit der Anfangsverteilung $X^{(0)} := (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ nach einem Schritt in G' genau dieselbe Wahrscheinlichkeitsverteilung erreichen, es soll also $X^{(0)} \cdot P(G', d) = X^{(0)}$ gelten. Geben Sie einen solchen Graphen G' an und weisen Sie nach, dass $X^{(0)} \cdot P(G', d) = X^{(0)}$ gilt.

Aufgabe 5.3. Wir nehmen an, das morgige Wetter ließe sich allein aus der Kenntnis des heutigen Wetters vorhersagen. Unter dieser Annahme kann der Wetterverlauf als Markov-Kette modelliert werden. Der Einfachheit halber unterscheiden wir im Folgenden nur die beiden Wetterbedingungen *Regen* und *Sonnenschein*. Das Wetter formt dann eine Markov-Kette mit der Zustandsmenge $Z = \{z_1, z_2\}$, wobei z_1 den Regen und z_2 den Sonnenschein bezeichnet, und der Übergangsmatrix

$$P = \begin{pmatrix} p_{z_1, z_1} & p_{z_1, z_2} \\ p_{z_2, z_1} & p_{z_2, z_2} \end{pmatrix}.$$

Dabei gibt der Wert p_{z_i, z_j} die Wahrscheinlichkeit dafür an, dass auf Wetter im Zustand z_i am folgenden Tag Wetter im Zustand z_j folgt.

Ist die Verteilung des Wetters $X^{(k)} = (X_{z_1}^{(k)}, X_{z_2}^{(k)})$ für einen Tag $k \in \mathbb{N}$ bekannt, so kann die Verteilung des Wetters am Tag $k + 1$ berechnet werden als $X^{(k+1)} = X^{(k)} \cdot P$.

- Für das Frankfurter Wetter wird oft behauptet, die beste Art der Wettervorhersage bestehe einfach darin, das morgige Wetter als identisch mit dem heutigen zu prognostizieren. Wenn diese Vorhersagemethode mit einer Wahrscheinlichkeit von $3/4$ richtig liegt (unabhängig davon, ob aktuell Regen oder Sonnenschein herrscht), dann ergibt sich für die Markov-Kette des Frankfurter Wetters die Übergangsmatrix

$$P_F = \begin{pmatrix} 3/4 & 1/4 \\ 1/4 & 3/4 \end{pmatrix}.$$

Wir nehmen an, dass die Markov-Kette für das Frankfurter Wetter an einem regnerischen Tag beginnt, d. h. es gilt $X_F^{(0)} = (1, 0)$.

- (i) Berechnen Sie die Verteilung des Frankfurter Wetters an Tag drei, d. h. berechnen Sie $X_F^{(3)}$.
 - (ii) Beweisen Sie durch vollständige Induktion, dass $X_F^{(k)} = (\frac{1}{2}(1 + 2^{-k}), \frac{1}{2}(1 - 2^{-k}))$ für jedes $k \in \mathbb{N}$ gilt.
 - (iii) Wie verhält sich $X_F^{(k)}$, wenn k gegen unendlich geht?
- (b) Wir betrachten Los Angeles als Beispiel für einen Ort, an dem der Wetterverlauf ein anderer ist als in Frankfurt. Sei die Übergangsmatrix für das Wetter in Los Angeles gegeben durch

$$P_{LA} = \begin{pmatrix} 1/2 & 1/2 \\ 1/10 & 9/10 \end{pmatrix}.$$

- (i) Zeigen Sie, dass die Verteilung $X_{LA} = (1/6, 5/6)$ eine stationäre Verteilung für das Wetter in Los Angeles ist, d. h. zeigen Sie, dass $X_{LA} = X_{LA} \cdot P_{LA}$ ist.
- (ii) Geben Sie eine stationäre Verteilung für das Frankfurter Wetter in Teilaufgabe (a) an, d. h. geben Sie eine Verteilung X_F an mit $X_F \cdot P_F = X_F$.

Aufgabe 5.4. Es ist eine allgemein bekannte Tatsache, dass der Weihnachtsmann nicht jedem Kind genau das bringt, was es sich wünscht. Wir haben durch investigative Recherchen erfahren, dass die Wahrscheinlichkeit für die Weihnachtswunscherfüllung eines Kindes allein davon abhängig ist, ob der Wunsch des Kindes im Jahr zuvor erfüllt wurde. Darum lässt sich die Folge der Ereignisse Wunscherfüllung/Nicht-Wunscherfüllung für ein Kind über die Jahre hinweg als Markov-Kette modellieren. Diese Markov-Kette hat als Zustandsmenge die Zustände z_1 für *Weihnachtswunsch erfüllt* und z_2 für *Weihnachtswunsch nicht erfüllt* sowie die Übergangsmatrix

$$P = \begin{pmatrix} p_{z_1, z_1} & p_{z_1, z_2} \\ p_{z_2, z_1} & p_{z_2, z_2} \end{pmatrix}.$$

Dabei gibt der Wert p_{z_i, z_j} die Wahrscheinlichkeit dafür an, dass nach dem Eintreten von Ereignis z_i an Weihnachten im darauf folgenden Jahr das Ereignis z_j eintritt.

Ist die Wahrscheinlichkeitsverteilung $X^{(k)} = (X_{z_1}^{(k)}, X_{z_2}^{(k)})$ der Wunscherfüllung für ein Jahr $k \in \mathbb{N}$ bekannt, so kann die Wahrscheinlichkeitsverteilung der Wunscherfüllung im Jahr $k + 1$ berechnet werden als $X^{(k+1)} = X^{(k)} \cdot P$.

- (a) Natürlich benutzt der Weihnachtsmann für jedes Kind eine andere Übergangsmatrix. Wir betrachten die Übergangsmatrix für Bob, die ausdrückt, dass sich das Ereignis der Weihnachtswunscherfüllung (bzw. -nichterfüllung) mit einer Wahrscheinlichkeit von $2/3$ aus dem Vorjahr wiederholt. Für die Markov-Kette von Bobs Wunscherfüllung lautet die Übergangsmatrix also

$$P_B = \begin{pmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{pmatrix}.$$

Wir nehmen an, dass die Markov-Kette für Bobs Wunscherfüllung damit beginnt, dass Bob seinen Wunsch erfüllt bekommt, d. h. es gelte $X_B^{(0)} = (1, 0)$.

- (i) Berechnen Sie die Wahrscheinlichkeitsverteilung für Bobs Wunscherfüllung in Jahr drei, d. h. berechnen Sie $X_B^{(3)}$.
- (ii) Beweisen Sie durch vollständige Induktion, dass $X_B^{(k)} = (\frac{1}{2}(1 + 3^{-k}), \frac{1}{2}(1 - 3^{-k}))$ für jedes $k \in \mathbb{N}$ gilt.
- (iii) Wie verhält sich $X_B^{(k)}$, wenn k gegen unendlich geht?

- (b) Es gibt Kinder, deren jeweilige Übergangsmatrix für ihre Weihnachtswunscherfüllung vorteilhafter ist als die von Bob. Als Beispiel für ein solches Kind betrachten wir Alice, deren Übergangsmatrix gegeben ist durch

$$P_A = \begin{pmatrix} 3/4 & 1/4 \\ 7/8 & 1/8 \end{pmatrix}.$$

- (i) Zeigen Sie, dass die Verteilung $X_A = (7/9, 2/9)$ eine stationäre Verteilung für die Wunscherfüllung von Alice ist, d. h. zeigen Sie, dass $X_A = X_A \cdot P_A$ ist.
- (ii) Geben Sie eine stationäre Verteilung für die Wunscherfüllung von Bob in Teilaufgabe (a) an, d. h. geben Sie eine Verteilung X_B an mit $X_B \cdot P_B = X_B$.

6 Logik erster Stufe (Prädikatenlogik)

In Kapitel 3 haben wir bereits die **Aussagenlogik** kennengelernt, die einen Formalismus darstellt, mit dessen Hilfe man “Wissen” modellieren und Schlüsse aus dem Wissen ziehen kann. In diesem Kapitel werden wir die **Logik erster Stufe** (bzw. **Prädikatenlogik**) als einen weiteren solchen Formalismus kennenlernen. Im Vergleich zur Aussagenlogik hat die Prädikatenlogik den Vorteil, dass

- eine klare Trennung zwischen “Daten” einerseits und “Logik” andererseits besteht, und dass in der Prädikatenlogik
- wesentlich umfangreichere Ausdrucksmöglichkeiten zur Verfügung stehen.

Der Preis für diese Vorteile ist allerdings, dass die Prädikatenlogik **algorithmisch** deutlich schwerer zu handhaben ist als die Aussagenlogik.

6.1 Motivation zur Logik erster Stufe

Grenzen der Aussagenlogik:

Beispiel 6.1 (Verwandtschaftsbeziehungen). Die Aussagenlogik kann helfen, um Aussagen der Art

“Anne und Bernd sind Geschwister. Wenn Christine Annes Tochter ist, dann ist Bernd Christines Onkel.”

zu modellieren und Schlüsse daraus zu ziehen. Für die Modellierung der folgenden Aussage ist die Aussagenlogik aber eher ungeeignet:

“Es gibt in Frankfurt mindestens 2 Leute, die mehr als 3 Kinder, aber selbst keine Geschwister haben.”

Beispiel 6.2 (Arithmetische Aussagen). Die Aussagenlogik kann helfen, um Sätze der Art

“Wenn eine Zahl gerade ist, dann ist sie nicht ungerade.”

zu formalisieren. Für viele andere Aussagen ist die Aussagenlogik aber eher ungeeignet, zum Beispiel:

“Es gibt eine Zahl, die nicht Summe zweier Primzahlen ist.”

Ein Überblick über die Logik erster Stufe:

Die Logik erster Stufe ist ein Formalismus, mit dem man die in den beiden obigen Beispielen genannten Aussagen bequem beschreiben kann. Genau wie die Aussagenlogik besitzt die Logik erster Stufe:

- eine **Syntax**, die festlegt, welche Zeichenketten Formeln der Logik erster Stufe sind und
- eine **Semantik**, die festlegt, welche “Bedeutung” einzelne Formeln haben.

Die Logik erster Stufe beschäftigt sich mit **Objekten** (z.B. den Einwohnern Frankfurts und deren Verwandtschaftsbeziehungen (Beispiel 6.1) oder den natürlichen Zahlen und deren Addition und Multiplikation (Beispiel 6.2)) und **Aussagen über deren Eigenschaften**. (Im Gegensatz dazu beschäftigt sich die Aussagenlogik nicht mit Objekten sondern lediglich mit “wahren” und “falschen” Aussagen und deren Kombination.)

Vor der Einführung in die Syntax und die Semantik der Logik erster Stufe wenden wir uns zunächst den Objekten zu, über die Formeln der Logik erster Stufe “reden” können.

6.2 Strukturen

Die Objekte, über die Formeln der Logik erster Stufe Aussagen treffen können, heißen **Strukturen**. Viele Objekte lassen sich auf natürliche Weise durch solche Strukturen repräsentieren, beispielsweise

Strukturen

- Graphen $G = (V, E)$
- Bäume $B = (V, E)$
- die natürlichen Zahlen mit Addition und Multiplikation, $(\mathbb{N}, +, \times)$
- die reellen Zahlen mit Addition, Multiplikation und den Konstanten 0 und 1, $(\mathbb{R}, +, \times, 0, 1)$
- Datenbanken

usw. Die im Folgenden definierten **Signaturen** legen den “Typ” (bzw. das “Format”) der entsprechenden Strukturen fest.

Definition 6.3. Eine **Signatur** (bzw. ein **Vokabular** bzw. eine **Symbolmenge**; englisch: signature, vocabulary) ist eine Menge σ von Relationssymbolen, Funktionssymbolen und/oder Konstantensymbolen. Jedes Relationssymbol $\dot{R} \in \sigma$ und jedes Funktionssymbol $\dot{f} \in \sigma$ hat eine **Stelligkeit** (bzw. Arität, engl. arity)

Signatur
Vokabular
Symbolmenge
Stelligkeit

$$\text{ar}(\dot{R}) \in \mathbb{N}_{>0} \quad \text{bzw.} \quad \text{ar}(\dot{f}) \in \mathbb{N}_{>0}.$$

Notation 6.4.

- In diesem Kapitel bezeichnet der griechische Buchstabe σ (in Worten: sigma) immer eine Signatur.
- Wir kennzeichnen Symbole aus σ immer mit einem Punkt, wie in \dot{R} bzw. \dot{f} .
- Für Relationssymbole verwenden wir meistens Großbuchstaben wie $\dot{R}, \dot{P}, \dot{E}, \dot{R}_1, \dot{R}_2, \dots$, für Funktionssymbole verwenden wir meistens Kleinbuchstaben wie $\dot{f}, \dot{g}, \dot{h}, \dots$, für Konstantensymbole verwenden wir meistens Kleinbuchstaben wie \dot{c}, \dot{d}, \dots .

- Gelegentlich verwenden wir als Relations- und Funktionssymbole auch Zeichen wie

$$\begin{aligned} \leq & \quad (2\text{-stelliges Relationssymbol}), \\ \dot{+}, \dot{\times} & \quad (2\text{-stellige Funktionssymbole}), \\ \dot{0}, \dot{1} & \quad (\text{Konstantensymbole}). \end{aligned}$$

Struktur
 σ -Struktur

Definition 6.5. Eine **Struktur über der Signatur** σ (kurz: **σ -Struktur**) ist ein Paar

$$\mathfrak{A} = (A, \alpha),$$

bestehend aus:

Universum
Träger

- einer nicht-leeren Menge A , dem so genannten **Universum** (bzw. **Träger**, engl.: universe, domain) von \mathfrak{A} , und
- einer auf σ definierten Abbildung α , die
 - jedem Relationssymbol $\dot{R} \in \sigma$ eine Relation $\alpha(\dot{R}) \subseteq A^{\text{ar}(\dot{R})}$ der Stelligkeit $\text{ar}(\dot{R})$ zuordnet,
 - jedem Funktionssymbol $\dot{f} \in \sigma$ eine Funktion $\alpha(\dot{f}): A^{\text{ar}(\dot{f})} \rightarrow A$ zuordnet,
 - jedem Konstantensymbol $\dot{c} \in \sigma$ ein Element $\alpha(\dot{c}) \in A$ zuordnet.

Notation 6.6.

- Strukturen bezeichnen wir meistens mit Fraktur-Buchstaben $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \dots$; das Universum der Strukturen durch die entsprechenden lateinischen Großbuchstaben, also A, B, G, \dots
- Ist $\mathfrak{A} = (A, \alpha)$ eine σ -Struktur, so schreiben wir für jedes Symbol $\dot{S} \in \sigma$ oft

$$\dot{S}^{\mathfrak{A}} \text{ an Stelle von } \alpha(\dot{S}).$$

An Stelle von $\mathfrak{A} = (A, \alpha)$ schreiben wir oft auch $\mathfrak{A} = (A, (\dot{S}^{\mathfrak{A}})_{\dot{S} \in \sigma})$.
Falls $\sigma = \{\dot{R}_1, \dots, \dot{R}_k, \dot{f}_1, \dots, \dot{f}_l, \dot{c}_1, \dots, \dot{c}_m\}$ ist, schreiben wir auch

$$\mathfrak{A} = (A, \dot{R}_1^{\mathfrak{A}}, \dots, \dot{R}_k^{\mathfrak{A}}, \dot{f}_1^{\mathfrak{A}}, \dots, \dot{f}_l^{\mathfrak{A}}, \dot{c}_1^{\mathfrak{A}}, \dots, \dot{c}_m^{\mathfrak{A}}).$$

Beispiel 6.7 (Arithmetische Strukturen). Sei $\sigma_{\text{Ar}} := \{\dot{+}, \dot{\times}, \dot{0}, \dot{1}\}$, wobei $\dot{+}$ und $\dot{\times}$ 2-stellige Funktionssymbole und $\dot{0}$ und $\dot{1}$ Konstantensymbole sind. Wir betrachten die σ_{Ar} -Struktur

$$\mathcal{N} := (\mathbb{N}, \dot{+}^{\mathcal{N}}, \dot{\times}^{\mathcal{N}}, \dot{0}^{\mathcal{N}}, \dot{1}^{\mathcal{N}}),$$

wobei $\dot{+}^{\mathcal{N}}$ und $\dot{\times}^{\mathcal{N}}$ die natürliche Addition bzw. Multiplikation auf \mathbb{N} sind und $\dot{0}^{\mathcal{N}} := 0$, $\dot{1}^{\mathcal{N}} := 1$. Entsprechend können wir σ_{Ar} -Strukturen $\mathcal{Z}, \mathcal{Q}, \mathcal{R}$ mit Universum $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ definieren.

Beispiel 6.8 (Graphen und Bäume). Sei $\sigma_{\text{Graph}} := \{\dot{E}\}$, wobei \dot{E} ein 2-stelliges Relationssymbol ist. Jeder gerichtete Graph bzw. gerichtete Baum (V, E) lässt sich als σ_{Graph} -Struktur $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ mit Universum $A := V$ und Relation $\dot{E}^{\mathfrak{A}} := E$ auffassen.

Beispiel 6.9 (Ordnungen). Sei $\sigma_{\text{Ord}} := \{\dot{\leq}\}$, wobei $\dot{\leq}$ ein 2-stelliges Relationssymbol ist. Jeder Präordnung, partiellen Ordnung oder linearen Ordnung \leq auf einer Menge A entspricht eine σ_{Ord} -Struktur

$$\mathfrak{A} = (A, \dot{\leq}^{\mathfrak{A}})$$

mit $\dot{\leq}^{\mathfrak{A}} := \leq$.

Frage: Wann sind zwei σ -Strukturen \mathfrak{A} und \mathfrak{B} “prinzipiell gleich” (Fachbegriff: isomorph)?

Antwort: Falls \mathfrak{B} aus \mathfrak{A} entsteht, indem man die Elemente des Universums von \mathfrak{A} umbenennt.

Analog zum Begriff der Isomorphie von Graphen (Definition 4.25) wird dies durch folgende Definition präzisiert:

Definition 6.10. Sei σ eine Signatur und seien \mathfrak{A} und \mathfrak{B} zwei σ -Strukturen. \mathfrak{A} und \mathfrak{B} heißen **isomorph** (kurz: $\mathfrak{A} \cong \mathfrak{B}$, in Worten: \mathfrak{A} ist isomorph zu \mathfrak{B}), falls es eine **bijektive** Abbildung $\pi: A \rightarrow B$ gibt, für die gilt: isomorph

- für jedes Relationssymbol $\dot{R} \in \sigma$, für $r := \text{ar}(\dot{R})$ und für alle r -Tupel $(a_1, \dots, a_r) \in A^r$ gilt:

$$(a_1, \dots, a_r) \in \dot{R}^{\mathfrak{A}} \iff (\pi(a_1), \dots, \pi(a_r)) \in \dot{R}^{\mathfrak{B}}.$$

- für jedes Konstantensymbol $\dot{c} \in \sigma$ gilt:

$$\pi(\dot{c}^{\mathfrak{A}}) = \dot{c}^{\mathfrak{B}}.$$

- für jedes Funktionssymbol $\dot{f} \in \sigma$, für $r := \text{ar}(\dot{f})$ und für alle r -Tupel $(a_1, \dots, a_r) \in A^r$ gilt:

$$\pi(\dot{f}^{\mathfrak{A}}(a_1, \dots, a_r)) = \dot{f}^{\mathfrak{B}}(\pi(a_1), \dots, \pi(a_r)).$$

Eine solche Abbildung π wird **Isomorphismus von \mathfrak{A} nach \mathfrak{B}** genannt.

Isomorphismus

Beispiel 6.11.

- (a) Ist $A = \{1, 2, 3, 4\}$, $B = \{6, 7, 8, 9\}$, und sind $\dot{\leq}^{\mathfrak{A}}$ und $\dot{\leq}^{\mathfrak{B}}$ die natürlichen linearen Ordnungen auf A und B , so sind die beiden σ_{Ord} -Strukturen $\mathfrak{A} = (A, \dot{\leq}^{\mathfrak{A}})$ und $\mathfrak{B} = (B, \dot{\leq}^{\mathfrak{B}})$ isomorph.

Skizze:



Allgemein gilt: Sind A und B endliche Mengen mit $|A| = |B|$ und sind $\dot{\leq}^{\mathfrak{A}}$ und $\dot{\leq}^{\mathfrak{B}}$ lineare Ordnungen auf den Universen A und B , so ist $\mathfrak{A} \cong \mathfrak{B}$, und die Abbildung π , die das (bzgl. $\dot{\leq}^{\mathfrak{A}}$) kleinste Element in A auf das (bzgl. $\dot{\leq}^{\mathfrak{B}}$) kleinste Element von \mathfrak{B} abbildet und, allgemein, für jedes $i \in \{1, \dots, |A|\}$ das i -kleinste Element in A (bzgl. $\dot{\leq}^{\mathfrak{A}}$) auf das i -kleinste Element in B (bzgl. $\dot{\leq}^{\mathfrak{B}}$) abbildet, ein Isomorphismus von \mathfrak{A} nach \mathfrak{B} .

- (b) Sind $\dot{\leq}^{\mathbb{N}}$ und $\dot{\leq}^{\mathbb{Z}}$ die natürlichen linearen Ordnungen auf \mathbb{N} und \mathbb{Z} , so sind die σ_{Ord} -Strukturen $\mathcal{N} := (\mathbb{N}, \dot{\leq}^{\mathbb{N}})$ und $\mathcal{Z} := (\mathbb{Z}, \dot{\leq}^{\mathbb{Z}})$ **nicht isomorph** (kurz: $\mathcal{N} \not\cong \mathcal{Z}$).

Skizze:



- (c) Sei $\sigma := \{\dot{f}, \dot{c}\}$, wobei \dot{f} ein 2-stelliges Funktionssymbol und \dot{c} ein Konstantensymbol ist. Sei $\mathfrak{A} := (A, \dot{f}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$, wobei

- $A := \mathbb{N}$
- $\dot{f}^{\mathfrak{A}} := \dot{+}^{\mathbb{N}}$ die Addition auf \mathbb{N}
- $\dot{c}^{\mathfrak{A}} := \dot{0}^{\mathbb{N}}$ die natürliche Zahl 0 ist

und sei $\mathfrak{B} := (B, \dot{f}^{\mathfrak{B}}, \dot{c}^{\mathfrak{B}})$, wobei

- $B := \{2^n : n \in \mathbb{N}\}$ die Menge aller Zweierpotenzen
- $\dot{f}^{\mathfrak{B}} : B \times B \rightarrow B$ die Funktion mit

$$\dot{f}^{\mathfrak{B}}(b_1, b_2) := b_1 \cdot b_2, \quad \text{f.a. } b_1, b_2 \in B$$

- $\dot{c}^{\mathfrak{B}} := 1 = 2^0 \in B$.

Dann gilt: $\mathfrak{A} \cong \mathfrak{B}$, und die Abbildung $\pi : A \rightarrow B$ mit $\pi(n) := 2^n$, f.a. $n \in \mathbb{N}$, ist ein Isomorphismus von \mathfrak{A} nach \mathfrak{B} , denn:

- π ist eine bijektive Abbildung von A nach B .
- Für das Konstantensymbol $\dot{c} \in \sigma$ gilt:

$$\pi(\dot{c}^{\mathfrak{A}}) \stackrel{\text{Def. } \dot{c}^{\mathfrak{A}}}{=} \pi(0) \stackrel{\text{Def. } \pi}{=} 2^0 \stackrel{\text{Def. } \dot{c}^{\mathfrak{B}}}{=} \dot{c}^{\mathfrak{B}}.$$

- Für das Funktionssymbol $\dot{f} \in \sigma$ und für alle $(a_1, a_2) \in A^2$ gilt:

$$\pi(\dot{f}^{\mathfrak{A}}(a_1, a_2)) \stackrel{\text{Def. } \dot{f}^{\mathfrak{A}}}{=} \pi(a_1 + a_2) \stackrel{\text{Def. } \pi}{=} 2^{a_1 + a_2}$$

und

$$\dot{f}^{\mathfrak{B}}(\pi(a_1), \pi(a_2)) \stackrel{\text{Def. } \pi}{=} \dot{f}^{\mathfrak{B}}(2^{a_1}, 2^{a_2}) \stackrel{\text{Def. } \dot{f}^{\mathfrak{B}}}{=} 2^{a_1} \cdot 2^{a_2} = 2^{a_1 + a_2}.$$

Also: $\pi(\dot{f}^{\mathfrak{A}}(a_1, a_2)) = \dot{f}^{\mathfrak{B}}(\pi(a_1), \pi(a_2))$. Somit ist π ein Isomorphismus von \mathfrak{A} nach \mathfrak{B} .

Wir wissen nun, über welche Objekte Formeln der Logik erster Stufe “reden” können: über σ -Strukturen, wobei σ eine Signatur ist. Als nächstes legen wir die Syntax der Logik erster Stufe fest.

6.3 Terme

Definition 6.12 (Variablen der Logik erster Stufe).

Individuenvariable
Variable

Eine **Individuenvariable** (kurz: **Variable**) hat die Form v_i , für $i \in \mathbb{N}$. Die Menge aller Variablen bezeichnen wir mit VAR , d.h. $\text{VAR} = \{v_i : i \in \mathbb{N}\}$.

Definition 6.13 (Terme der Logik erster Stufe).

(a) Für eine Signatur σ sei $A_{\sigma\text{-Terme}}$ das Alphabet, das aus allen Elementen in VAR, allen Konstanten- und Funktionssymbolen in σ , den Klammern $(,)$ und dem Komma $,$ besteht. $A_{\sigma\text{-Terme}}$

(b) Die Menge T_{σ} der σ -Terme ist die wie folgt rekursiv definierte Teilmenge von $A_{\sigma\text{-Terme}}^*$: σ -Terme

Basisregeln:

- Für jedes Konstantensymbol $\dot{c} \in \sigma$ ist $\dot{c} \in T_{\sigma}$.
- Für jede Variable $x \in \text{VAR}$ ist $x \in T_{\sigma}$.

Rekursive Regeln:

- Für jedes Funktionssymbol $\dot{f} \in \sigma$ und für $r := \text{ar}(\dot{f})$ gilt: Sind $t_1 \in T_{\sigma}, \dots, t_r \in T_{\sigma}$, so ist auch $\dot{f}(t_1, \dots, t_r) \in T_{\sigma}$.

Beispiel 6.14. Sei $\sigma = \{\dot{f}, \dot{c}\}$ die Signatur aus Beispiel 6.11(c), die aus einem 2-stelligen Funktionssymbol \dot{f} und einem Konstantensymbol \dot{c} besteht.

Folgende Worte sind σ -Terme:

$$\dot{c}, \quad v_4, \quad \dot{f}(\dot{c}, \dot{c}), \quad \dot{f}(\dot{c}, v_0), \quad \dot{f}(\dot{c}, \dot{f}(\dot{c}, v_0)).$$

Folgende Worte sind keine σ -Terme:

$$\mathbf{0}, \quad \dot{f}(\mathbf{0}, \dot{c}), \quad \dot{f}(v_0, \dot{c}, v_1), \quad f^{\mathfrak{A}}(2, 3).$$

Definition 6.15 (Belegungen und Interpretationen).

(a) Eine **Belegung** in einer σ -Struktur $\mathfrak{A} = (A, \alpha)$ ist eine partielle Funktion β von VAR nach A (d.h. β ordnet jeder Variablen $x \in \text{Def}(\beta)$ ein Element $\beta(x)$ aus dem Universum von \mathfrak{A} zu). Belegung

(b) Eine Belegung β ist eine **Belegung für einen σ -Term t** (bzw. **passend zu t**), wenn $\text{Def}(\beta)$ alle in t vorkommenden Variablen enthält. passend zu t

(c) Eine σ -**Interpretation** ist ein Paar σ -Interpretation

$$\mathcal{I} = (\mathfrak{A}, \beta),$$

bestehend aus einer σ -Struktur \mathfrak{A} und einer Belegung β in \mathfrak{A} .

(d) Eine σ -Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ ist eine **Interpretation für einen σ -Term t** (bzw. **passend zu t**), wenn β passend zu t ist.

Wir wollen Terme nun in Interpretationen “auswerten”. Die Auswertung von Term t in einer zu t passenden Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ soll dasjenige Element aus A liefern, das man erhält, wenn man die in t vorkommenden Variablen gemäß der Belegung β interpretiert, die in t vorkommenden Konstantensymbole gemäß ihrer Interpretation in \mathfrak{A} belegt, und dann nach und nach den Term t gemäß den in \mathfrak{A} gegebenen Interpretationen der Funktionssymbole berechnet.

Dies wird in der folgenden Definition präzisiert:

Definition 6.16 (Semantik von σ -Termen).

Sei σ eine Signatur. Rekursiv über den Aufbau von T_σ definieren wir eine Funktion $\llbracket \cdot \rrbracket$, die jedem σ -Term $t \in T_\sigma$ und jeder zu t passenden σ -Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ einen Wert $\llbracket t \rrbracket^{\mathcal{I}} \in A$ zuordnet:

- Für alle $x \in \text{VAR}$ ist $\llbracket x \rrbracket^{\mathcal{I}} := \beta(x)$.
- Für alle Konstantensymbole $\dot{c} \in \sigma$ ist $\llbracket \dot{c} \rrbracket^{\mathcal{I}} := \dot{c}^{\mathfrak{A}}$.
- Für alle Funktionssymbole $\dot{f} \in \sigma$, für $r := \text{ar}(\dot{f})$ und für alle σ -Terme $t_1, \dots, t_r \in T_\sigma$ gilt:

$$\llbracket \dot{f}(t_1, \dots, t_r) \rrbracket^{\mathcal{I}} := \dot{f}^{\mathfrak{A}}(\llbracket t_1 \rrbracket^{\mathcal{I}}, \dots, \llbracket t_r \rrbracket^{\mathcal{I}}).$$

Beispiel 6.17. Sei $\sigma = \{\dot{f}, \dot{c}\}$ und sei $\mathfrak{A} = (A, \dot{f}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$ mit

- $A := \mathbb{N}$
- $\dot{f}^{\mathfrak{A}} := \dot{+}^{\mathbb{N}}$ (die Addition auf \mathbb{N})
- $\dot{c}^{\mathfrak{A}} := \dot{0}^{\mathbb{N}}$ (die natürliche Zahl 0)

wie im Beispiel 6.11(c). Sei β eine Belegung mit $\beta(v_1) = 1$ und $\beta(v_2) = 7$. Und sei $\mathcal{I} := (\mathfrak{A}, \beta)$. Sei t der Term $\dot{f}(v_2, \dot{f}(v_1, \dot{c}))$. Dann gilt:

$$\begin{aligned} \llbracket t \rrbracket^{\mathcal{I}} &= \llbracket \dot{f}(v_2, \dot{f}(v_1, \dot{c})) \rrbracket^{\mathcal{I}} \\ &= \dot{f}^{\mathfrak{A}}(\beta(v_2), \dot{f}^{\mathfrak{A}}(\beta(v_1), \dot{c}^{\mathfrak{A}})) \\ &= \dot{f}^{\mathfrak{A}}(7, \dot{f}^{\mathfrak{A}}(1, 0)) \\ &= (7 + (1 + 0)) \\ &= 8. \end{aligned}$$

6.4 Syntax der Logik erster Stufe

Die Logik erster Stufe übernimmt, verändert und erweitert die Syntax der Aussagenlogik.

- Was gleich bleibt:
 - Alle Junktoren $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ werden übernommen.
- Was sich verändert:
 - Variablen stehen nicht mehr für “wahre” oder “falsche” Aussagen, sondern für Elemente im Universum einer σ -Struktur.
 - Variablen sind keine atomaren Formeln mehr.
- Was neu hinzukommt:
 - Es gibt **Quantoren** \exists (für “es existiert”) und \forall (für “für alle”).
 - Es gibt Symbole für Elemente aus der Signatur σ .
 - Es können σ -Terme benutzt werden, um Elemente im Universum einer σ -Struktur \mathfrak{A} zu bezeichnen.

Definition 6.18 (Das Alphabet der Logik erster Stufe).

Sei σ eine Signatur. Das **Alphabet** A_σ der **Logik erster Stufe über** σ besteht aus:

Alphabet A_σ

- allen Symbolen in $A_{\sigma\text{-Terme}}$
- allen Relationssymbolen in σ
- den Quantoren \exists (Existenzquantor) und \forall (Allquantor)
- dem Gleichheitssymbol \doteq
- den Junktoren $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.

D.h.

$$A_\sigma = \text{VAR} \cup \sigma \cup \{\exists, \forall\} \cup \{\doteq\} \cup \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,)\} \cup \{\{, \}\}.$$

Definition 6.19 (Formeln der Logik erster Stufe).

Sei σ eine Signatur. Die Menge $\text{FO}[\sigma]$ aller **Formeln der Logik erster Stufe über der Signatur** σ (kurz: **FO** $[\sigma]$ -**Formeln**; FO steht für die englische Bezeichnung der Logik erster Stufe: first-order logic) ist die folgendermaßen rekursiv definierte Teilmenge von A_σ^* :

FO $[\sigma]$ -Formeln

Basisregeln:

- Für alle σ -Terme t_1 und t_2 in T_σ gilt:

$$t_1 \doteq t_2 \in \text{FO}[\sigma].$$

- Für jedes Relationssymbol $\dot{R} \in \sigma$, für $r := \text{ar}(\dot{R})$ und für alle σ -Terme t_1, \dots, t_r in T_σ gilt:

$$\dot{R}(t_1, \dots, t_r) \in \text{FO}[\sigma].$$

Bemerkung: FO $[\sigma]$ -Formeln der Form $t_1 \doteq t_2$ oder $\dot{R}(t_1, \dots, t_r)$ heißen **atomare σ -Formeln**.

atomare
 σ -Formeln

Rekursive Regeln:

- Ist $\varphi \in \text{FO}[\sigma]$, so auch $\neg\varphi \in \text{FO}[\sigma]$.
- Ist $\varphi \in \text{FO}[\sigma]$ und $\psi \in \text{FO}[\sigma]$, so ist auch
 - $(\varphi \wedge \psi) \in \text{FO}[\sigma]$
 - $(\varphi \vee \psi) \in \text{FO}[\sigma]$
 - $(\varphi \rightarrow \psi) \in \text{FO}[\sigma]$
 - $(\varphi \leftrightarrow \psi) \in \text{FO}[\sigma]$.
- Ist $\varphi \in \text{FO}[\sigma]$ und ist $x \in \text{VAR}$, so ist auch
 - $\exists x \varphi \in \text{FO}[\sigma]$
 - $\forall x \varphi \in \text{FO}[\sigma]$.

Beispiel 6.20.

- (a) Sei $\sigma = \{\dot{f}, \dot{c}\}$ die Signatur aus Beispiel 6.11(c), die aus einem 2-stelligen Funktionssymbol \dot{f} und einem Konstantensymbol \dot{c} besteht.

Folgende Worte aus A_σ^* sind FO[σ]-Formeln:

- $\dot{f}(v_0, v_1) \dot{=} \dot{c}$ (atomare σ -Formel)
- $\forall v_2 \dot{f}(v_2, \dot{c}) \dot{=} v_2$
- $\neg \exists v_3 (\dot{f}(v_3, v_3) \dot{=} v_3 \wedge \neg v_3 \dot{=} \dot{c})$

Folgende Worte sind **keine** FO[σ]-Formeln:

- $(\dot{f}(v_0, v_1) \dot{=} \dot{c})$
- $(\forall v_2 (\dot{f}(v_2, \dot{c}) \dot{=} v_2))$
- $\exists \dot{c} \dot{f}(v_0, \dot{c}) \dot{=} v_0$

- (b) Sei $\sigma_{\text{Graph}} = \{\dot{E}\}$ die Signatur, die aus einem 2-stelligen Relationssymbol besteht. Folgendes ist eine FO[σ_{Graph}]-Formel:

$$\forall v_0 \forall v_1 \left((\dot{E}(v_0, v_1) \wedge \dot{E}(v_1, v_0)) \rightarrow v_0 \dot{=} v_1 \right).$$

Intuition zur Semantik: In einem Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ sagt diese Formel folgendes aus:

“für alle Knoten $a_0 \in A$ und
für alle Knoten $a_1 \in A$ gilt:
falls $(a_0, a_1) \in \dot{E}^{\mathfrak{A}}$ und $(a_1, a_0) \in \dot{E}^{\mathfrak{A}}$, so ist $a_0 = a_1$.”

Die Formel sagt in einem Graph $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ also gerade aus, dass die Kantenrelation $\dot{E}^{\mathfrak{A}}$ antisymmetrisch ist (vgl. Definition 4.68). D.h.: Ein Graph $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ **erfüllt** die Formel genau dann, wenn die Kantenrelation $\dot{E}^{\mathfrak{A}}$ antisymmetrisch ist.

Notation 6.21.

- Statt mit v_0, v_1, v_2, \dots bezeichnen wir Variablen oft auch mit x, y, z, \dots oder mit Varianten wie x', y_1, y_2, \dots
- Für gewisse 2-stellige Relationssymbole wie z.B. $\dot{\leq} \in \sigma_{\text{Ord}}$ verwenden wir **Infix- statt Präfixschreibweise** und setzen Klammern dabei auf natürliche Weise, um die eindeutige Lesbarkeit zu gewährleisten.

Beispiel:

An Stelle der (formal korrekten) atomaren Formel $\dot{\leq}(x, y)$ schreiben wir $x \dot{\leq} y$.

Wir wissen nun, welche Zeichenketten (über dem Alphabet A_σ) **FO[σ]-Formeln** genannt werden.

6.5 Semantik der Logik erster Stufe

Bevor wir die Semantik der Logik erster Stufe formal definieren, betrachten wir zunächst einige Beispiele, um ein intuitives Verständnis der Semantik der Logik erster Stufe zu erlangen.

6.5.1 Beispiele zur Semantik der Logik erster Stufe

Beispiel 6.22 (gerichtete Graphen).

Sei $\sigma_{\text{Graph}} = \{\dot{E}\}$, wobei \dot{E} ein 2-stelliges Relationssymbol ist.

(a) Die $\text{FO}[\sigma_{\text{Graph}}]$ -Formel

$$\varphi := \forall x \forall y (\dot{E}(x, y) \rightarrow \dot{E}(y, x))$$

besagt:

“Für alle Knoten x und für alle Knoten y gilt: Falls es eine Kante von x nach y gibt, so gibt es auch eine Kante von y nach x .”

Für jeden Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ gilt daher:

$$\mathfrak{A} \text{ erfüllt } \varphi \iff \dot{E}^{\mathfrak{A}} \text{ ist symmetrisch.}$$

Umgangssprachlich sagen wir auch: “Die Formel φ sagt **in einem Graphen \mathfrak{A} aus, dass dessen Kantenrelation symmetrisch ist.**”

(b) Die folgende $\text{FO}[\sigma_{\text{Graph}}]$ -Formel drückt aus, dass es von Knoten x zu Knoten y einen Weg der Länge 3 gibt:

$$\varphi(x, y) := \exists z_1 \exists z_2 \left((\dot{E}(x, z_1) \wedge \dot{E}(z_1, z_2)) \wedge \dot{E}(z_2, y) \right).$$

(c) Die $\text{FO}[\sigma_{\text{Graph}}]$ -Formel

$$\forall x \forall y \exists z_1 \exists z_2 \left((\dot{E}(x, z_1) \wedge \dot{E}(z_1, z_2)) \wedge \dot{E}(z_2, y) \right)$$

sagt in einem Graphen \mathfrak{A} aus, dass es zwischen je 2 Knoten einen Weg der Länge 3 gibt.

Beispiel 6.23 (Verwandtschaftsbeziehungen). Um Verwandtschaftsbeziehungen zu modellieren, können wir die Signatur σ benutzen, die aus den folgenden Symbolen besteht:

- 1-stellige Funktionssymbole *Vater*, *Mutter*
(Bedeutung: $x \doteq \text{Vater}(y)$ besagt “ x ist der Vater von y ”).
- 2-stellige Relationssymbole *Geschwister*, *Vorfahr*
(Bedeutung: $\text{Geschwister}(x, y)$ besagt, dass x und y Geschwister sind; $\text{Vorfahr}(x, y)$ besagt, dass x ein Vorfahr von y ist.)

Generelles Wissen über Verwandtschaftsbeziehungen lässt sich durch Formeln der Logik erster Stufe repräsentieren, beispielsweise:

- “Personen mit gleichem Vater und gleicher Mutter sind Geschwister”:

$$\forall x \forall y \left((\text{Vater}(x) \doteq \text{Vater}(y) \wedge \text{Mutter}(x) \doteq \text{Mutter}(y)) \rightarrow \text{Geschwister}(x, y) \right).$$

- “Eltern sind gerade die unmittelbaren Vorfahren”:

$$\forall x \forall y \left((x \doteq \text{Vater}(y) \vee x \doteq \text{Mutter}(y)) \leftrightarrow \left(\text{Vorfahr}(x, y) \wedge \neg \exists z (\text{Vorfahr}(x, z) \wedge \text{Vorfahr}(z, y)) \right) \right).$$

- “Die Relation *Vorfahr* ist transitiv”:

$$\forall x \forall y \forall z \left((Vorfahr(x, y) \wedge Vorfahr(y, z)) \rightarrow Vorfahr(x, z) \right).$$

- Die folgende Formel $\varphi(x, y)$ besagt, dass x Tante oder Onkel von y ist:

$$\varphi(x, y) := \exists z \left(Geschwister(x, z) \wedge (z \doteq Vater(y) \vee z \doteq Mutter(y)) \right).$$

- Die folgende Formel $\psi(x)$ besagt, dass x Vater von genau 2 Kindern ist:

$$\psi(x) := \exists y_1 \exists y_2 \left(\left((x \doteq Vater(y_1) \wedge x \doteq Vater(y_2)) \wedge \neg y_1 \doteq y_2 \right) \wedge \forall z (x \doteq Vater(z) \rightarrow (z \doteq y_1 \vee z \doteq y_2)) \right).$$

6.5.2 Formale Definition der Semantik der Logik erster Stufe

Um die formale Definition der Semantik der Logik erster Stufe angeben zu können, benötigen wir noch folgende Begriffe:

Notation 6.24.

Teilformel

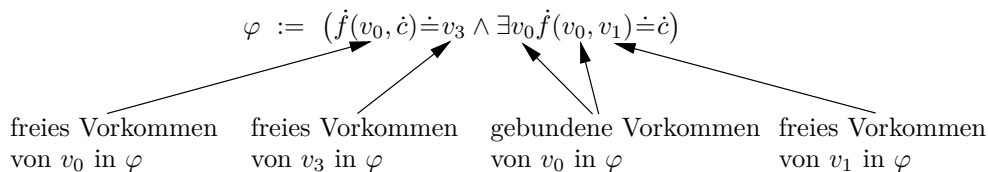
- (a) Eine Formel ψ ist **Teilformel** einer Formel φ , wenn ψ als Teil-Wort in φ vorkommt.

Beispiel: $\psi := \dot{f}(v_0, v_1) \doteq \dot{c}$ ist Teilformel der Formel $\exists v_0 \dot{f}(v_0, v_1) \doteq \dot{c}$.

gebunden
frei

- (b) Ist φ eine Formel und x eine Variable, so heißt jedes Vorkommen von x in einer Teilformel der Form $\exists x \psi$ oder $\forall x \psi$ **gebunden**. Jedes andere Vorkommen von x in φ heißt **frei**.

Beispiel:



frei(φ)
freien Variablen

- (c) Die Menge $\text{frei}(\varphi)$ aller **freien Variablen** einer FO[σ]-Formel φ besteht aus allen Variablen, die mindestens einmal frei in φ vorkommen.

Beispiele:

- $\text{frei}(\dot{f}(v_0, \dot{c}) \doteq v_3) = \{v_0, v_3\}$
- $\text{frei}(\exists v_0 \dot{f}(v_0, v_1) \doteq \dot{c}) = \{v_1\}$
- $\text{frei}(\dot{f}(v_0, \dot{c}) \doteq v_3 \wedge \exists v_0 \dot{f}(v_0, v_1) \doteq \dot{c}) = \{v_0, v_3, v_1\}$

Satz

- (d) Eine FO[σ]-Formel φ heißt **Satz** (genauer: FO[σ]-Satz), falls sie keine freien Variablen besitzt, d.h. falls $\text{frei}(\varphi) = \emptyset$.

Definition 6.25 (passende Belegungen und Interpretationen).

- (a) Eine Belegung β ist eine **Belegung für eine FO[σ]-Formel** φ (bzw. **passend zu** φ), wenn $\text{frei}(\varphi) \subseteq \text{Def}(\beta)$. passend zu φ
- (b) Eine σ -Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ ist eine **Interpretation für eine FO[σ]-Formel** φ (bzw. **passend zu** φ), wenn β passend zu φ ist. Interpretation für eine FO[σ]-Formel

Notation 6.26.

- (a) Ist β eine Belegung in einer σ -Struktur \mathfrak{A} , ist $x \in \text{VAR}$ und ist $a \in A$, so sei

$$\beta_x^a$$

die Belegung mit $\text{Def}(\beta_x^a) := \text{Def}(\beta) \cup \{x\}$ und

$$\beta_x^a(x) = a \quad \text{und} \quad \beta_x^a(y) = \beta(y) \quad \text{f.a. } y \in \text{Def}(\beta) \setminus \{x\}.$$

- (b) Ist $\mathcal{I} = (\mathfrak{A}, \beta)$ eine σ -Interpretation, ist $x \in \text{VAR}$ und ist $a \in A$, so sei

$$\mathcal{I}_x^a := (\mathfrak{A}, \beta_x^a).$$

Wir können nun (endlich) die formale Semantik der Logik erster Stufe festlegen.

Definition 6.27 (Semantik der Logik erster Stufe).

Sei σ eine Signatur. Rekursiv über den Aufbau von FO[σ] definieren wir eine Funktion $\llbracket \cdot \rrbracket$, die jeder FO[σ]-Formel φ und jeder zu φ passenden Interpretationen $\mathcal{I} = (\mathfrak{A}, \beta)$ einen **Wahrheitswert** (kurz: **Wert**) $\llbracket \varphi \rrbracket^{\mathcal{I}} \in \{0, 1\}$ zuordnet. Wahrheitswert

Rekursionsanfang:

- Für alle σ -Terme t_1 und t_2 in T_σ gilt:

$$\llbracket t_1 \doteq t_2 \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } \llbracket t_1 \rrbracket^{\mathcal{I}} = \llbracket t_2 \rrbracket^{\mathcal{I}} \\ 0, & \text{sonst.} \end{cases}$$

- Für jedes Relationssymbol $\dot{R} \in \sigma$, für $r := \text{ar}(\dot{R})$ und für alle σ -Terme t_1, \dots, t_r in T_σ gilt:

$$\llbracket \dot{R}(t_1, \dots, t_r) \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } (\llbracket t_1 \rrbracket^{\mathcal{I}}, \dots, \llbracket t_r \rrbracket^{\mathcal{I}}) \in \dot{R}^{\mathfrak{A}} \\ 0, & \text{sonst.} \end{cases}$$

Rekursionsschritt:

- Die Semantik der Junktoren $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ ist wie in der Aussagenlogik definiert, d.h. für alle $\varphi \in \text{FO}[\sigma]$ und $\psi \in \text{FO}[\sigma]$ gilt:

$$\llbracket \neg \varphi \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{I}} = 0 \\ 0, & \text{sonst.} \end{cases}$$

$$\llbracket (\varphi \wedge \psi) \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{I}} = 1 \text{ und } \llbracket \psi \rrbracket^{\mathcal{I}} = 1 \\ 0, & \text{sonst.} \end{cases}$$

$$\llbracket (\varphi \vee \psi) \rrbracket^{\mathcal{I}} := \begin{cases} 0, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{I}} = 0 \text{ und } \llbracket \psi \rrbracket^{\mathcal{I}} = 0 \\ 1, & \text{sonst.} \end{cases}$$

$$\llbracket (\varphi \rightarrow \psi) \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{I}} = 0 \text{ oder } \llbracket \psi \rrbracket^{\mathcal{I}} = 1 \\ 0, & \text{sonst.} \end{cases}$$

$$\llbracket (\varphi \leftrightarrow \psi) \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls } \llbracket \varphi \rrbracket^{\mathcal{I}} = \llbracket \psi \rrbracket^{\mathcal{I}} \\ 0, & \text{sonst.} \end{cases}$$

- Ist $\varphi \in \text{FO}[\sigma]$ und ist $x \in \text{VAR}$, so ist

$$- \llbracket \exists x \varphi \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls es (mindestens) ein } a \in A \text{ gibt, so dass } \llbracket \varphi \rrbracket^{\mathcal{I} \frac{a}{x}} = 1 \\ 0, & \text{sonst.} \end{cases}$$

$$- \llbracket \forall x \varphi \rrbracket^{\mathcal{I}} := \begin{cases} 1, & \text{falls für alle } a \in A \text{ gilt: } \llbracket \varphi \rrbracket^{\mathcal{I} \frac{a}{x}} = 1 \\ 0, & \text{sonst.} \end{cases}$$

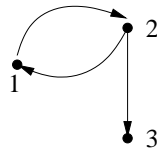
Beispiel 6.28. Sei $\sigma_{\text{Graph}} = \{\dot{E}\}$ die Signatur, die aus einem 2-stelligen Relationssymbol \dot{E} besteht. Betrachte die $\text{FO}[\sigma_{\text{Graph}}]$ -Formel

$$\varphi := \forall x \forall y (\dot{E}(x, y) \rightarrow \dot{E}(y, x)).$$

Für jede zu φ passende σ_{Graph} -Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ gilt:

$$\begin{aligned} \llbracket \varphi \rrbracket^{\mathcal{I}} = 1 &\iff \text{für alle } a \in A \text{ gilt: } \llbracket \forall y (\dot{E}(x, y) \rightarrow \dot{E}(y, x)) \rrbracket^{\mathcal{I} \frac{a}{x}} = 1 \\ &\iff \text{für alle } a \in A \text{ gilt:} \\ &\quad \text{für alle } b \in A \text{ gilt: } \llbracket (\dot{E}(x, y) \rightarrow \dot{E}(y, x)) \rrbracket^{\mathcal{I} \frac{a}{x} \frac{b}{y}} = 1 \\ &\iff \text{für alle } a \in A \text{ und alle } b \in A \text{ gilt:} \\ &\quad \text{falls } \llbracket \dot{E}(x, y) \rrbracket^{\mathcal{I} \frac{a}{x} \frac{b}{y}} = 1, \text{ so auch } \llbracket \dot{E}(y, x) \rrbracket^{\mathcal{I} \frac{a}{x} \frac{b}{y}} = 1 \\ &\iff \text{für alle } a \in A \text{ und alle } b \in A \text{ gilt:} \\ &\quad \text{falls } (a, b) \in \dot{E}^{\mathfrak{A}}, \text{ so auch } (b, a) \in \dot{E}^{\mathfrak{A}} \\ &\iff \dot{E}^{\mathfrak{A}} \text{ ist symmetrisch, (vgl. Def. 4.68).} \end{aligned}$$

Sei nun \mathfrak{A} die σ_{Graph} -Struktur, die den gerichteten Graphen



repräsentiert, d.h. $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ mit $A = \{1, 2, 3\}$ und $\dot{E}^{\mathfrak{A}} = \{(1, 2), (2, 1), (2, 3)\}$. Sei β die Belegung mit leerem Definitionsbereich und sei $\mathcal{I} := (\mathfrak{A}, \beta)$. Dann gilt: Da in unserem konkreten Graphen \mathfrak{A} für $a = 2$ und $b = 3$ gilt: $(a, b) \in \dot{E}^{\mathfrak{A}}$, aber $(b, a) \notin \dot{E}^{\mathfrak{A}}$, ist $\dot{E}^{\mathfrak{A}}$ nicht symmetrisch, und daher ist hier $\llbracket \varphi \rrbracket^{\mathcal{I}} = 0$.

Notation 6.29. Sei σ eine Signatur und sei φ eine FO[σ]-Formel.

(a) Sei $\mathcal{I} = (\mathfrak{A}, \beta)$ eine zu φ passende σ -Interpretation.

Wir sagen “ \mathcal{I} erfüllt φ ” (bzw. “ \mathcal{I} ist ein Modell von φ ”, kurz: $\mathcal{I} \models \varphi$), falls $\llbracket \varphi \rrbracket^{\mathcal{I}} = 1$. $\mathcal{I} \models \varphi$

Wir sagen “ \mathcal{I} erfüllt φ nicht” (bzw. “ \mathcal{I} ist kein Modell von φ ”, kurz: $\mathcal{I} \not\models \varphi$), falls $\llbracket \varphi \rrbracket^{\mathcal{I}} = 0$. $\mathcal{I} \not\models \varphi$

(b) Ist φ ein **Satz** (d.h. φ hat keine freien Variablen), so hängt die Tatsache, ob φ von einer Interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ erfüllt wird, nur von der Struktur \mathfrak{A} und nicht von der Belegung β ab. An Stelle von “ $\mathcal{I} \models \varphi$ ” schreiben wir dann kurz “ $\mathfrak{A} \models \varphi$ ” und sagen “die σ -Struktur \mathfrak{A} erfüllt den Satz φ .”

6.6 Ein Anwendungsbereich der Logik erster Stufe: Datenbanken

Relationale Datenbanken bestehen aus **Tabellen**, die sich als Relationen auffassen lassen. Datenbanken lassen sich daher als **Strukturen** über einer passenden Signatur auffassen. Die in der Praxis gebräuchlichste Datenbankabfragesprache ist **SQL**. Der “Kern” von SQL basiert auf der Logik erster Stufe, die in der Datenbankterminologie oft auch “relationaler Kalkül” (engl.: “relational calculus”) bezeichnet wird.

Zur Illustration von Anfragen verwenden wir eine kleine Datenbank mit Kinodaten, bestehend aus:

- einer Tabelle *Orte*, die Informationen über Kinos (Kino, Adresse, Telefonnummer) enthält,
- einer Tabelle *Filme*, die Informationen über Filme enthält (Titel, Regie, Schauspieler).
- eine Tabelle *Programm*, die Informationen zum aktuellen Kinoprogramm enthält (Kino, Titel, Zeit).

Orte-Tabelle:

Kino	Adresse	Telefon
Babylon	Dresdner Str. 2	61609693
Casablanca	Friedenstr. 12	6775752
Cinestar Cubix Alexanderplatz	Rathausstr. 1	2576110
Die Kurbel	Giesebrechtstr. 4	88915998
Filmpalast Berlin	Kurfürstendamm 225	8838551
International	Karl-Marx-Allee 33	24756011
Kino in der Kulturbrauerei	Schönhauser Allee 36	44354422
Movimiento	Kottbusser Damm 22	6924785

Filme-Tabelle:

Titel	Regie	Schauspieler
Capote	Bennet Miller	Philip Seymour Hoffman
Capote	Bennet Miller	Catherine Keener
Das Leben der Anderen	F. Henkel von Donnersmarck	Martina Gedeck
Das Leben der Anderen	F. Henkel von Donnersmarck	Ulrich Tukur
Der ewige Gärtner	Fernando Meirelles	Ralph Fiennes
Der ewige Gärtner	Fernando Meirelles	Rachel Weisz
Good Night and Good Luck	George Clooney	David Strathairn
Good Night and Good Luck	George Clooney	Patricia Clarkson
Knallhart	Detlev Buck	Jenny Elvers
Knallhart	Detlev Buck	Jan Henrik Stahlberg
Raumpatrouille Orion – Rücksturz ins Kino	Michael Braun	Dietmar Schönherr
Raumpatrouille Orion – Rücksturz ins Kino	Theo Mezger	Dietmar Schönherr
Raumpatrouille Orion – Rücksturz ins Kino	Michael Braun	Eva Pflug
Raumpatrouille Orion – Rücksturz ins Kino	Theo Mezger	Eva Pflug
Raumpatrouille Orion – Rücksturz ins Kino	Michael Braun	Wolfgang Völz
Raumpatrouille Orion – Rücksturz ins Kino	Theo Mezger	Wolfgang Völz
Requiem	Hans-Christian Schmid	Sandra Hüller
Sommer vorm Balkon	Andreas Dresen	Nadja Uhl
Sommer vorm Balkon	Andreas Dresen	Inka Friedrich
Sommer vorm Balkon	Andreas Dresen	Andreas Schmidt
Syriana	Stephen Gaghan	George Clooney
Syriana	Stephen Gaghan	Matt Damon
V wie Vendetta	James McTeigue	Natalie Portman
Walk the Line	James Mangold	Joaquin Phoenix
Walk the Line	James Mangold	Reese Witherspoon

Programm-Tabelle:

Kino	Titel	Zeit
Babylon	Capote	17:00
Babylon	Capote	19:30
Kino in der Kulturbrauerei	Capote	17:30
Kino in der Kulturbrauerei	Capote	20:15
International	Das Leben der Anderen	14:30
International	Das Leben der Anderen	17:30
International	Das Leben der Anderen	20:30
Filmpalast Berlin	Good Night and Good Luck	15:30
Filmpalast Berlin	Good Night and Good Luck	17:45
Filmpalast Berlin	Good Night and Good Luck	20:00
Kino in der Kulturbrauerei	Good Night and Good Luck	18:00
Kino in der Kulturbrauerei	Good Night and Good Luck	20:00
Kino in der Kulturbrauerei	Good Night and Good Luck	22:45
Babylon	Sommer vorm Balkon	21:45
Kino in der Kulturbrauerei	Sommer vorm Balkon	21:45
Filmuseum Potsdam	Raumpatrouille Orion – Rücksturz ins Kino	22:00

Für eine geeignete Signatur σ_{Kino} können wir diese Datenbank durch eine σ_{Kino} -Struktur $\mathfrak{A}_{\text{Kino}}$ folgendermaßen modellieren: Die Signatur σ_{Kino} besteht aus:

- einem 3-stelligen Relationssymbol *Orte*
- einem 3-stelligen Relationssymbol *Filme*
- einem 3-stelligen Relationssymbol *Programm*
- Konstantensymbolen 'c', die allen potentiellen Einträgen *c* der Datenbank entsprechen, also 'Babylon', 'Casablanca', ..., 'Capote', 'Das Leben der Anderen', ... usw., aber auch z.B. 'Stephen Spielberg' oder 'Lola rennt'. D.h.: Für jedes Wort *c* über dem ASCII-Alphabet gibt es ein Konstantensymbol 'c'.

Die σ_{Kino} -Struktur $\mathfrak{A}_{\text{Kino}}$ hat als Universum die Menge aller Worte über dem ASCII-Alphabet, d.h.

$$A_{\text{Kino}} := \text{ASCII}^*,$$

die 3-stelligen Relationen

$$\begin{aligned} \text{Orte}^{\mathfrak{A}_{\text{Kino}}} := & \{(\text{Babylon, Dresdner Str. 2, 61609693}), \\ & (\text{Casablanca, Friedenstr. 12, 6775752}), \\ & \dots, \\ & (\text{Movimiento, Kottbusser Damm 22, 6924785})\}, \end{aligned}$$

$$\begin{aligned} \text{Filme}^{\mathfrak{A}_{\text{Kino}}} := & \{(\text{Capote, Bennet Miller, Philip Seymour Hoffman}), \\ & (\text{Capote, Bennet Miller, Catherine Keener}), \\ & \dots, \\ & (\text{Walk the Line, James Mangold, Reese Witherspoon})\}, \end{aligned}$$

$$\begin{aligned} \text{Programm}^{\mathfrak{A}_{\text{Kino}}} := & \{(\text{Babylon, Capote, 17:00}), \\ & (\text{Babylon, Capote, 19:30}), \\ & (\text{Kino in der Kulturbrauerei, Capote, 17:30}), \\ & \dots\} \end{aligned}$$

sowie für jedes in σ_{Kino} vorkommende Konstantensymbol 'c' die Konstante 'c' $^{\mathfrak{A}_{\text{Kino}}} := c$.
Zum Beispiel:

$$\begin{aligned} \text{'Babylon'}^{\mathfrak{A}_{\text{Kino}}} &= \text{Babylon}, \\ \text{'Capote'}^{\mathfrak{A}_{\text{Kino}}} &= \text{Capote}, \\ \text{'George Clooney'}^{\mathfrak{A}_{\text{Kino}}} &= \text{George Clooney}. \end{aligned}$$

Anfragen an die Kinodatenbank lassen sich auf unterschiedliche Art formulieren:

Beispiel 6.30. (a) Eine Anfrage an unsere Kinodatenbank:

“Gib die Titel aller Filme aus, die um 17:30 Uhr laufen.”

In der Datenbankanfragesprache SQL lässt sich dies folgendermaßen formulieren:

```
SELECT Titel
FROM Programm
WHERE Zeit = '17:30'
```

Dieselbe Anfrage lässt sich auch durch die folgende Formel der Logik erster Stufe beschreiben:

$$\varphi_{\text{Filme um 17:30 Uhr}}(x_T) := \exists x_K \text{ Programm}(x_K, x_T, '17:30').$$

(b) Die Anfrage

“Gib die Titel aller Filme aus, in denen George Clooney mitspielt oder Regie führt.”

lässt sich in Logik erster Stufe wie folgt beschreiben:

$$\begin{aligned} \varphi_{\text{Filme mit George Clooney}}(x_T) := \\ (\exists x_R \text{ Filme}(x_T, x_R, 'George Clooney') \vee \exists x_S \text{ Filme}(x_T, 'George Clooney', x_S)). \end{aligned}$$

Notation 6.31. Sei σ eine Signatur und seien x_1, \dots, x_n Variablen.

- Die Notation $\varphi(x_1, \dots, x_n)$ deutet an, dass φ eine $\text{FO}[\sigma]$ -Formel mit $\text{frei}(\varphi) = \{x_1, \dots, x_n\}$ ist, d.h. dass x_1, \dots, x_n diejenigen Variablen sind, die in φ frei vorkommen.
- Ist $\varphi(x_1, \dots, x_n)$ eine $\text{FO}[\sigma]$ -Formel, ist \mathfrak{A} eine σ -Struktur und sind a_1, \dots, a_n Elemente im Universum von \mathfrak{A} , so schreiben wir

$$\mathfrak{A} \models \varphi[a_1, \dots, a_n],$$

um auszudrücken, dass für die Belegung $\beta: \{x_1, \dots, x_n\} \rightarrow A$ mit $\beta(x_1) = a_1, \dots, \beta(x_n) = a_n$ gilt:

$$(\mathfrak{A}, \beta) \models \varphi.$$

Definition 6.32. Sei σ eine Signatur, $\varphi(x_1, \dots, x_n)$ eine $\text{FO}[\sigma]$ -Formel und \mathfrak{A} eine σ -Struktur. Die von φ in \mathfrak{A} definierte n -stellige Relation ist

$$\varphi(\mathfrak{A}) := \{ (a_1, \dots, a_n) \in A^n : \mathfrak{A} \models \varphi[a_1, \dots, a_n] \}.$$

Beispiel 6.33. Die $\text{FO}[\sigma_{\text{Kino}}]$ -Formel $\varphi_{\text{Filme um 17:30}}(x_T)$ aus Beispiel 6.30 definiert in unserer Kinodatenbank $\mathfrak{A}_{\text{Kino}}$ die 1-stellige Relation:

$$\begin{aligned} \varphi_{\text{Filme um 17:30}}(\mathfrak{A}_{\text{Kino}}) = \{ & (\text{Capote}), \\ & (\text{Das Leben der Anderen}) \}. \end{aligned}$$

Darstellung als Tabelle:

Filme um 17:30 Uhr:	Titel
	Capote
	Das Leben der Anderen

Beispiel 6.34. (a) Die Anfrage

“Gib Name und Adresse aller Kinos aus, in denen ein Film läuft, in dem George Clooney mitspielt oder Regie führt.”

lässt sich folgendermaßen formulieren:

In SQL:

```

SELECT  Orte.Kino, Orte.Adresse
FROM    Orte, Filme, Programm
WHERE   Orte.Kino = Programm.Kino AND
        Filme.Titel = Programm.Titel AND
        (Filme.Schauspieler = 'George Clooney' OR
         Filme.Regie = 'George Clooney')

```

In Logik erster Stufe:

$$\varphi_{\text{Kinos mit George Clooney}}(x_K, x_A) := \left(\exists x_{\text{Tel}} \text{Orte}(x_K, x_A, x_{\text{Tel}}) \wedge \exists x_T \exists x_Z \left(\text{Programm}(x_K, x_T, x_Z) \wedge (\exists x_R \text{Filme}(x_T, x_R, \text{'George Clooney'}) \vee \exists x_S \text{Filme}(x_T, \text{'George Clooney'}, x_S)) \right) \right)$$

In unserer konkreten Kinodatenbank $\mathfrak{A}_{\text{Kino}}$ liefert diese Formel die 2-stellige Relation:

$$\varphi_{\text{Kinos mit George Clooney}}(\mathfrak{A}_{\text{Kino}}) = \{ (\text{Filmpalast Berlin, Kurfürstendamm 225}), (\text{Kino in der Kulturbrauerei, Schönhauser Allee 36}) \}.$$

Darstellung als Tabelle:

Kinos mit George Clooney:	Kino	Adresse
	Filmpalast Berlin	Kurfürstendamm 225
	Kino in der Kulturbrauerei	Schönhauser Allee 36

(b) Die Anfrage

“Gib die Titel aller Filme aus, in denen nur Schauspieler mitspielen, die schon mal mit Stephen Spielberg zusammengearbeitet haben.”

lässt sich in Logik erster Stufe wie folgt formulieren:

$$\varphi_{\text{Filme mit Spielberg-Schauspielern}}(x_T) := \exists x_R \exists x_S \left(\text{Filme}(x_T, x_R, x_S) \wedge \forall y_S (\text{Filme}(x_T, x_R, y_S) \rightarrow \exists z_T \text{Filme}(z_T, \text{'Stephen Spielberg'}, y_S)) \right)$$

Details zum Thema Datenbanken und Datenbankanfragesprachen können Sie in den Vorlesungen “Datenbanksysteme I und II” und “Logik und Datenbanken” kennenlernen.

6.7 Erfüllbarkeit, Allgemeingültigkeit, Folgerung und Äquivalenz

Definition 6.35. Sei σ eine Signatur und sei φ eine FO[σ]-Formel.

- (a) φ heißt **erfüllbar**, wenn es (mindestens) eine zu φ passende σ -Interpretation \mathcal{I} gibt, die φ erfüllt. erfüllbar
- (b) φ heißt **unerfüllbar**, wenn φ nicht erfüllbar ist. unerfüllbar
- (c) φ heißt **allgemeingültig**, wenn jede zu φ passende σ -Interpretation φ erfüllt. allgemeingültig

Beispiel 6.36. Sei $\sigma_{\text{Graph}} := \{\dot{E}\}$ die Signatur, die aus einem 2-stelligen Relationssymbol \dot{E} besteht. Die FO[σ_{Graph}]-Formel $\varphi := \forall y \dot{E}(x, y)$ ist erfüllbar, aber nicht allgemeingültig, denn: Sei $\mathfrak{A} := (A, \dot{E}^{\mathfrak{A}})$ der gerichtete Graph



und sei β die Belegung mit $\beta(x) = 1$. Dann erfüllt die Interpretation (\mathfrak{A}, β) die Formel φ . Somit ist φ erfüllbar.

Andererseits gilt für den Graphen $\mathfrak{B} := (B, \dot{E}^{\mathfrak{B}})$



und die Belegung β mit $\beta(x) = 1$, dass die zu φ passende σ -Interpretation $\mathcal{I} := (\mathfrak{B}, \beta)$ die Formel φ **nicht** erfüllt (d.h. $\llbracket \varphi \rrbracket^{\mathcal{I}} = 0$), denn:

$$\begin{aligned} \llbracket \varphi \rrbracket^{\mathcal{I}} = 1 &\iff \text{Für jedes } b \in B \text{ gilt: } \llbracket \dot{E}(x, y) \rrbracket^{\mathcal{I}} \stackrel{b}{y} = 1 \\ &\iff \text{Für jedes } b \in B \text{ gilt: } (\beta \stackrel{b}{y}(x), \beta \stackrel{b}{y}(y)) \in \dot{E}^{\mathfrak{B}} \\ &\iff \text{Für jedes } b \in B \text{ gilt: } (1, b) \in \dot{E}^{\mathfrak{B}}. \end{aligned}$$

Aber für $b := 1$ gilt: $(1, 1) \notin \dot{E}^{\mathfrak{B}}$, und daher ist $\llbracket \varphi \rrbracket^{\mathcal{I}} = 0$. \mathcal{I} ist also eine zu φ passende σ -Interpretation, die φ **nicht** erfüllt. Somit ist φ nicht allgemeingültig.

Beobachtung 6.37. Für alle Formeln φ der Logik erster Stufe gilt:

- (a) φ ist allgemeingültig $\iff \neg\varphi$ ist unerfüllbar.
- (b) φ ist erfüllbar $\iff \neg\varphi$ ist nicht allgemeingültig.

Beweis: Übung. □

Definition 6.38 (semantische Folgerung).

ψ folgt aus φ

Sei σ eine Signatur und seien φ und ψ zwei FO[σ]-Formeln. Wir sagen ψ **folgt aus** φ (kurz: $\varphi \models \psi$, “ φ impliziert ψ ”), falls für jede zu φ und ψ passende Interpretation \mathcal{I} gilt:

$$\text{Falls } \underbrace{\mathcal{I} \models \varphi}_{\text{d.h. } \llbracket \varphi \rrbracket^{\mathcal{I}} = 1}, \text{ so auch } \underbrace{\mathcal{I} \models \psi}_{\text{d.h. } \llbracket \psi \rrbracket^{\mathcal{I}} = 1}.$$

Definition 6.39 (logische Äquivalenz).

äquivalent

Sei σ eine Signatur. Zwei FO[σ]-Formeln φ und ψ heißen **äquivalent** (kurz: $\varphi \equiv \psi$), wenn für jede zu φ und ψ passende σ -Interpretation \mathcal{I} gilt:

$$\mathcal{I} \models \varphi \iff \mathcal{I} \models \psi.$$

Beobachtung 6.40. Sei σ eine Signatur und seien φ und ψ zwei FO[σ]-Formeln. Es gilt:

- (a) $\varphi \equiv \psi \iff \varphi \models \psi$ und $\psi \models \varphi$.

- (b) $\varphi \equiv \psi \iff (\varphi \leftrightarrow \psi)$ ist allgemeingültig.
 (c) $\varphi \models \psi \iff (\varphi \rightarrow \psi)$ ist allgemeingültig.

Beweis: Übung. □

6.8 Grenzen der Logik erster Stufe

In Beispiel 6.22 und 6.23 haben wir viele Beispiele für umgangssprachliche Aussagen kennengelernt, die man durch Formeln der Logik erster Stufe beschreiben kann (siehe auch die Übungsaufgaben am Ende dieses Kapitels). Es gibt allerdings auch Aussagen, die **nicht** in der Logik erster Stufe formalisiert werden können:

Satz 6.41. Sei $\sigma_{\text{Graph}} := \{\dot{E}\}$ die Signatur, die aus einem 2-stelligen Relationssymbol \dot{E} besteht. Es gilt:

- (a) Es gibt keinen $\text{FO}[\sigma_{\text{Graph}}]$ -Satz φ , so dass für jeden gerichteten Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ gilt:

$$\mathfrak{A} \text{ erfüllt } \varphi \iff \mathfrak{A} \text{ ist azyklisch (vgl. Definition 4.14).}$$

- (b) Es gibt keinen $\text{FO}[\sigma_{\text{Graph}}]$ -Satz φ' , so dass für jeden gerichteten Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ gilt:

$$\mathfrak{A} \text{ erfüllt } \varphi' \iff \mathfrak{A} \text{ ist stark zusammenhängend (vgl. Definition 4.15).}$$

- (c) Es gibt keine $\text{FO}[\sigma_{\text{Graph}}]$ -Formel ψ mit freien Variablen x und y , so dass für jeden gerichteten Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ und jede zu ψ passende Belegung β in \mathfrak{A} gilt:

$$(\mathfrak{A}, \beta) \text{ erfüllt } \psi \iff \text{es gibt in } \mathfrak{A} \text{ einen Weg von Knoten } \beta(x) \text{ zu Knoten } \beta(y).$$

Einen Beweis dieses Satzes können Sie in der Vorlesung “Logik in der Informatik” kennenlernen.

6.9 Literaturhinweise

[27] Kapitel 2.1

[17] Kapitel 4.A

[15] Kapitel 4.2

Vorsicht: Jedes dieser Bücher verwendet unterschiedliche Notationen, die wiederum etwas von den in der Vorlesungen eingeführten Notationen abweichen.

6.10 Übungsaufgaben zu Kapitel 6

Aufgabe 6.1. Sei $\sigma := \{\dot{f}, \dot{Q}, \dot{R}, \dot{c}\}$ eine Signatur mit einem 1-stelligen Funktionssymbol \dot{f} , einem 3-stelligen Relationssymbol \dot{Q} , einem 2-stelligen Relationssymbol \dot{R} und einem Konstantensymbol \dot{c} .

- (a) Überprüfen Sie für jedes der folgenden Wörter, ob es sich jeweils um einen σ -Term (gemäß Definition 6.13), um eine atomare σ -Formel bzw. um eine $\text{FO}[\sigma]$ -Formel (gemäß Definition 6.15) handelt. Begründen Sie gegebenenfalls, warum ein Wort kein σ -Term, keine atomare σ -Formel bzw. keine $\text{FO}[\sigma]$ -Formel darstellt.

- (i) $\dot{Q}(\dot{f}(v_1), v_4, \dot{c})$ (iv) $\exists v_9 \dot{f}(\dot{f}(\dot{c})) \doteq \dot{f}(v_8)$
(ii) $\dot{R}(\dot{f}(v_1), v_4, \dot{c})$ (v) $\forall v_9 \dot{f}(\dot{f}(\dot{c})) \doteq \dot{f}(v_8 \wedge v_9)$
(iii) $(\dot{f}(v_7) \leftrightarrow \dot{R}(v_1, v_2))$ (vi) $\exists v_1 \forall v_2 \forall v_4 (\dot{c} \doteq \dot{f}(v_4) \vee \forall v_2 (\dot{R}(\dot{f}(v_1), v_4) \rightarrow \dot{Q}(v_1, v_3, v_4)))$

- (b) Betrachten Sie die σ -Strukturen $\mathfrak{A} = (A, \dot{f}^{\mathfrak{A}}, \dot{Q}^{\mathfrak{A}}, \dot{R}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$ und $\mathfrak{B} = (B, \dot{f}^{\mathfrak{B}}, \dot{Q}^{\mathfrak{B}}, \dot{R}^{\mathfrak{B}}, \dot{c}^{\mathfrak{B}})$ wobei
- $A := \{1, 2, 3, 4, 5\}$, $\dot{Q}^{\mathfrak{A}} := \{(2, 2, 4), (5, 3, 1)\}$, $\dot{R}^{\mathfrak{A}} := \{(3, 3), (5, 4), (1, 1)\}$, $\dot{c}^{\mathfrak{A}} := 2$
 - $B := \{k, l, m, n, o\}$, $\dot{Q}^{\mathfrak{B}} := \{(k, m, o), (n, n, l)\}$, $\dot{R}^{\mathfrak{B}} := \{(o, o), (m, m), (k, l)\}$, $\dot{c}^{\mathfrak{B}} := n$
- und die Funktionen $\dot{f}^{\mathfrak{A}}: A \rightarrow A$ und $\dot{f}^{\mathfrak{B}}: B \rightarrow B$ definiert sind durch

x	1	2	3	4	5
$\dot{f}^{\mathfrak{A}}(x)$	2	1	2	5	4

x	k	l	m	n	o
$\dot{f}^{\mathfrak{B}}(x)$	l	k	n	o	n

Überprüfen Sie, ob $\mathfrak{A} \cong \mathfrak{B}$ gilt. Falls ja, geben Sie einen Isomorphismus von \mathfrak{A} nach \mathfrak{B} an und begründen Sie, warum es sich um einen Isomorphismus handelt. Falls nein, begründen Sie, warum es keinen Isomorphismus von \mathfrak{A} nach \mathfrak{B} gibt.

Aufgabe 6.2. Sei $\sigma = \{\dot{B}, \dot{S}, \dot{F}, \text{Nachfolger}, \text{letzter}\}$ eine Signatur, wobei $\dot{B}, \dot{S}, \dot{F}$ 1-stellige Relationsymbole, Nachfolger ein 1-stelliges Funktionssymbol und letzter ein Konstantensymbol ist. Sei \mathfrak{A} eine σ -Struktur mit $A = \{1, 2, \dots, 34\}$ und $\text{letzter}^{\mathfrak{A}} = 34$, so dass für alle $a \in A$ gilt:

- $a \in \dot{B}^{\mathfrak{A}} \iff$ FC Bayern München ist Tabellenführer an Spieltag a
- $a \in \dot{S}^{\mathfrak{A}} \iff$ FC Schalke 04 ist Tabellenführer an Spieltag a
- $a \in \dot{F}^{\mathfrak{A}} \iff$ Eintracht Frankfurt ist Tabellenführer an Spieltag a
- $\text{Nachfolger}^{\mathfrak{A}}(a) = \begin{cases} a + 1, & \text{falls } a \in \{1, 2, \dots, 33\} \\ a, & \text{falls } a = 34. \end{cases}$

- (a) Geben Sie FO[σ]-Formeln an, die in \mathfrak{A} folgendes aussagen:
- Eintracht Frankfurt ist mindestens einmal Tabellenführer.
 - Jede der drei Mannschaften ist mindestens einmal Tabellenführer.
 - Sind die Bayern an einem Spieltag Erster, so werden sie auch Meister.
 - Schalke holt nicht den Titel, wenn sie bereits am vorletzten Spieltag Tabellenführer sind.
- (b) Beschreiben Sie umgangssprachlich, was jede der folgenden FO[σ]-Formeln in \mathfrak{A} aussagt:
- $\forall x (\neg \dot{B}(x) \rightarrow (\dot{S}(x) \vee \dot{F}(x)))$
 - $\neg \exists x \left(\dot{F}(x) \wedge (\dot{F}(\text{Nachfolger}(x)) \wedge (\dot{F}(\text{Nachfolger}(\text{Nachfolger}(x)))) \wedge \neg \text{Nachfolger}(x) \doteq \text{letzter}) \right)$
 - $\left(\neg \exists x (\dot{S}(x) \wedge \neg x \doteq \text{letzter}) \rightarrow \neg \dot{S}(\text{letzter}) \right)$

Aufgabe 6.3. Sophie, Marie und Lena nehmen im Internet an einer Auktion teil. Die Auktion dauert zehn Runden und in jeder Runde gibt es genau einen Höchstbietenden. Der Höchstbietende der zehnten Runde gewinnt die Auktion. Den Verlauf dieser Auktion modellieren wir wie folgt.

Sei $\sigma := \{\dot{S}, \dot{M}, \dot{L}, \text{Nachfolger}, \text{Endrunde}\}$ eine Signatur, wobei $\dot{S}, \dot{M}, \dot{L}$ 1-stellige Relationssymbole, Nachfolger ein 1-stelliges Funktionssymbol und Endrunde ein Konstantensymbol ist. Sei \mathfrak{A} eine σ -Struktur mit $\mathfrak{A} = (A, \dot{S}^{\mathfrak{A}}, \dot{M}^{\mathfrak{A}}, \dot{L}^{\mathfrak{A}}, \text{Nachfolger}^{\mathfrak{A}}, \text{Endrunde}^{\mathfrak{A}})$ wobei $A := \{1, \dots, 10\}$ und $\text{Endrunde}^{\mathfrak{A}} := 10$. Außerdem gilt für alle $a \in A$:

- $a \in \dot{S}^{\mathfrak{A}} \iff$ Sophie ist in Runde a die Höchstbietende
- $a \in \dot{M}^{\mathfrak{A}} \iff$ Marie ist in Runde a die Höchstbietende
- $a \in \dot{L}^{\mathfrak{A}} \iff$ Lena ist in Runde a die Höchstbietende
- $\text{Nachfolger}^{\mathfrak{A}}(a) := \begin{cases} a + 1, & \text{falls } a \in \{1, \dots, 9\} \\ a, & \text{falls } a = 10. \end{cases}$

So sagt beispielsweise die FO[σ]-Formel $\varphi = \forall x(\dot{S}(x) \vee \dot{M}(x))$ aus, dass in jeder Runde Sophie oder Marie die Höchstbietende ist.

(a) Geben Sie möglichst kurze FO[σ]-Formeln an, die in \mathfrak{A} folgendes aussagen:

- (i) Lena gewinnt die Auktion.
- (ii) Sophie ist in mindestens einer Runde die Höchstbietende.
- (iii) Lena ist in der neunten Runde die Höchstbietende.
- (iv) Ist Marie in keiner der ersten neun Runden die Höchstbietende, so gewinnt sie auch nicht die Auktion.

(b) Beschreiben Sie umgangssprachlich, was die folgenden FO[σ]-Formeln aussagen

- (i) $\forall x(\neg(\dot{L}(x) \vee \dot{S}(x)) \rightarrow \dot{M}(x))$
- (ii) $\forall x((\neg \text{Nachfolger}(x) \doteq x \wedge \dot{S}(x)) \rightarrow \dot{L}(\text{Nachfolger}(x)))$
- (iii) $\neg \exists x(\dot{M}(x) \wedge (\dot{M}(\text{Nachfolger}(x)) \wedge (\dot{M}(\text{Nachfolger}(\text{Nachfolger}(x))) \wedge \neg \text{Nachfolger}(x) \doteq \text{Endrunde})))$

Aufgabe 6.4. Sei $\sigma := \{\dot{F}, \dot{I}, \dot{M}, \dot{P}, \dot{Lzs}\}$ eine Signatur, wobei \dot{F} ein 2-stelliges Relationssymbol, $\dot{I}, \dot{M}, \dot{P}$ jeweils 1-stellige Relationssymbole und \dot{Lzs} ein Konstantensymbol ist. Sei \mathfrak{A} eine σ -Struktur mit $\mathfrak{A} := (A, \dot{F}^{\mathfrak{A}}, \dot{I}^{\mathfrak{A}}, \dot{M}^{\mathfrak{A}}, \dot{P}^{\mathfrak{A}}, \dot{Lzs}^{\mathfrak{A}})$, in der A die Menge der Studierenden ist und $\dot{Lzs}^{\mathfrak{A}}$ den Langzeitstudenten aus A bezeichnet, also die Person aus A , die schon am längsten studiert. Außerdem gilt für alle x und y aus A :

- $(x, y) \in \dot{F}^{\mathfrak{A}} \iff x$ und y sind miteinander befreundet
- $x \in \dot{I}^{\mathfrak{A}} \iff x$ studiert Informatik
- $x \in \dot{M}^{\mathfrak{A}} \iff x$ studiert Mathematik
- $x \in \dot{P}^{\mathfrak{A}} \iff x$ studiert Physik

Beachten Sie, dass $\dot{F}^{\mathfrak{A}}$ eine symmetrische Relation darstellt und niemand mit sich selbst befreundet ist. Die FO[σ]-Formel $\exists x(\dot{I}(x) \wedge \dot{M}(x))$ sagt beispielsweise aus, dass es einen Studierenden gibt, der Informatik und Mathematik studiert.

- (a) Geben Sie möglichst kurze FO[σ]-Formeln an, die in \mathfrak{A} jeweils folgendes aussagen:
- (i) Der Langzeitstudent studiert Mathematik.
 - (ii) Für jedes der Fächer Informatik, Mathematik und Physik gibt es jeweils einen Studierenden, der dieses Fach studiert.
 - (iii) Jeder Studierende der Informatik ist mit dem Langzeitstudenten befreundet.
 - (iv) Jeder Studierende der Physik ist mit einem Studierenden der Physik befreundet.
- (b) Beschreiben Sie umgangssprachlich, was jede der folgenden FO[σ]-Formeln in \mathfrak{A} aussagt:
- (i) $\exists x \neg \left((\dot{I}(x) \vee \dot{M}(x)) \vee \dot{P}(x) \right)$
 - (ii) $\forall x \forall y \left((\dot{M}(x) \wedge \dot{P}(y)) \rightarrow \neg \dot{F}(x, y) \right)$
 - (iii) $\exists x \forall y \left((\dot{F}(x, y) \vee x \doteq y) \vee \exists z \left(\dot{F}(x, z) \wedge \dot{F}(z, y) \right) \right)$

Aufgabe 6.5. Sei $\sigma = \{\dot{R}, \dot{f}, \dot{c}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{R} , einem 1-stelligen Funktionssymbol \dot{f} und einem Konstantensymbol \dot{c} .

Betrachten Sie die σ -Struktur $\mathfrak{A} = (A, \dot{R}^{\mathfrak{A}}, \dot{f}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$, wobei gilt, dass $A = \{1, 3, 5, 7, 8, 9\}$, $\dot{R}^{\mathfrak{A}} = \{(1, 3), (1, 5), (5, 1), (5, 8), (9, 9)\}$ und $\dot{c}^{\mathfrak{A}} = 1$. Weiterhin sei $\dot{f}^{\mathfrak{A}} : A \rightarrow A$, definiert durch

x	1	3	5	7	8	9
$\dot{f}^{\mathfrak{A}}(x)$	9	8	7	5	3	1

Sei $\mathcal{I} = (\mathfrak{A}, \beta)$ die σ -Interpretation mit der Belegung $\beta : \text{Var} \rightarrow A$, für die gilt:

$$\beta(v_0) = 3, \beta(v_1) = 1, \beta(v_2) = 9, \text{ und } \beta(v_i) = 5 \text{ für alle } i \geq 3.$$

Berechnen Sie $\llbracket t_1 \rrbracket^{\mathcal{I}}$ für den σ -Term t_1 analog zu Beispiel 6.23 aus dem Skript. Berechnen Sie weiterhin $\llbracket \varphi_1 \rrbracket^{\mathcal{I}}$ und $\llbracket \varphi_2 \rrbracket^{\mathcal{I}}$ für die FO[σ]-Formeln φ_1 und φ_2 analog zu Beispiel (6.28) im Skript.

- (i) $t_1 := \dot{f}(\dot{f}(v_0))$
- (ii) $\varphi_1 := (\dot{R}(v_6, v_2) \rightarrow \dot{R}(v_2, v_2))$
- (iii) $\varphi_2 := \forall v_4 ((\dot{R}(\dot{c}, v_4) \vee \dot{R}(v_3, v_4)) \wedge \exists v_0 \neg \dot{f}(v_0) \doteq v_4)$

Aufgabe 6.6. Sei $\sigma = \{\dot{f}, \dot{R}, \dot{c}\}$ eine Signatur mit einem 2-stelligen Funktionssymbol \dot{f} , einem 3-stelligen Relationssymbol \dot{R} und einem Konstantensymbol \dot{c} . Betrachten Sie die σ -Struktur $\mathfrak{A} = (A, \dot{f}^{\mathfrak{A}}, \dot{R}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$, wobei $A = \{0, 1, 2, 3, 4\}$, $\dot{R}^{\mathfrak{A}} = \{(0, 3, 4), (1, 3, 0), (4, 2, 3)\}$, $\dot{c}^{\mathfrak{A}} = 3$ und die Funktion $\dot{f}^{\mathfrak{A}} : A \times A \rightarrow A$ definiert ist durch

$\dot{f}^{\mathfrak{A}}$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Zum Beispiel gilt $\dot{f}^{\mathfrak{A}}(2, 3) = 0$ und $\dot{f}^{\mathfrak{A}}(1, 3) = 4$.

Sei $\mathcal{I} = (\mathfrak{A}, \beta)$ die Interpretation mit der Belegung $\beta : \text{VAR} \rightarrow A$, für die gilt: $\beta(v_0) = 2$, $\beta(v_1) = 0$, $\beta(v_2) = 1$, $\beta(v_3) = 4$, und $\beta(v_i) = 3$ für alle $i \geq 4$.

(a) Berechnen Sie $\llbracket t_1 \rrbracket^{\mathcal{I}}$ und $\llbracket t_2 \rrbracket^{\mathcal{I}}$ für

- $t_1 := \dot{f}(\dot{f}(v_1, v_5), \dot{c})$
- $t_2 := \dot{f}(\dot{f}(\dot{c}, \dot{f}(v_2, \dot{c})), v_1)$

(b) Berechnen Sie $\llbracket \varphi_1 \rrbracket^{\mathcal{I}}$ und $\llbracket \varphi_2 \rrbracket^{\mathcal{I}}$ für

- $\varphi_1 := (\dot{R}(v_1, v_2, \dot{f}(v_0, v_2)) \vee \exists v_0 \dot{R}(v_0, v_2, v_3))$
- $\varphi_2 := \forall v_1 (\dot{f}(v_1, \dot{c}) \dot{=} \dot{f}(v_2, \dot{c}) \rightarrow \exists v_3 (\dot{R}(v_1, v_2, v_3) \vee \dot{f}(v_1, v_5) \dot{=} v_4))$

Aufgabe 6.7.

Sei $\sigma := \{\dot{f}, \dot{c}\}$ eine Signatur mit einem 2-stelligen Funktionssymbol \dot{f} und einem Konstantensymbol \dot{c} . Wir betrachten die σ -Struktur $\mathfrak{A} := (A, \dot{f}^{\mathfrak{A}}, \dot{c}^{\mathfrak{A}})$, wobei $A := \{\text{Stein, Schere, Papier, Echse, Spock}\}$ und $\dot{c}^{\mathfrak{A}} := \text{Spock}$. Der Wert $\dot{f}^{\mathfrak{A}}(x, y)$ für $x, y \in A$ findet sich in Zeile x und Spalte y der nebenstehenden Tabelle.¹

$\dot{f}^{\mathfrak{A}}$	Stein	Schere	Papier	Echse	Spock
Stein	Stein	Stein	Papier	Stein	Spock
Schere	Stein	Schere	Schere	Schere	Spock
Papier	Papier	Schere	Papier	Echse	Papier
Echse	Stein	Schere	Echse	Echse	Echse
Spock	Spock	Spock	Papier	Echse	Spock

Sei $\mathcal{I} = (\mathfrak{A}, \beta)$ die σ -Interpretation mit der Belegung $\beta: \text{VAR} \rightarrow A$, für die gilt:

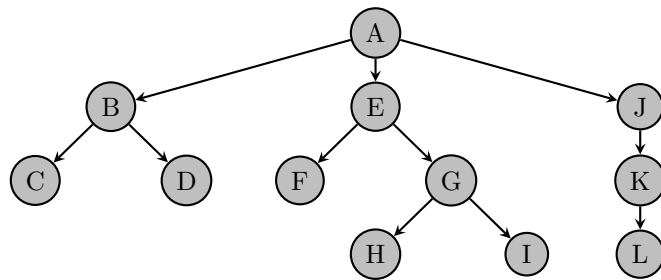
$$\beta(v_0) = \text{Stein}, \beta(v_1) = \text{Spock}, \beta(v_2) = \text{Schere}, \text{ und } \beta(v_i) = \text{Papier} \text{ für alle } i \geq 3.$$

Berechnen Sie $\llbracket t_1 \rrbracket^{\mathcal{I}}$, $\llbracket t_2 \rrbracket^{\mathcal{I}}$, $\llbracket t_3 \rrbracket^{\mathcal{I}}$ für die folgenden σ -Terme:

- (a) $t_1 := \dot{f}(v_0, \dot{c})$ (b) $t_2 := \dot{f}(v_1, \dot{f}(v_0, v_2))$ (c) $t_3 := \dot{f}(\dot{f}(\dot{f}(v_0, v_0), \dot{c}), \dot{f}(v_3, \dot{f}(v_4, v_5)))$

Aufgabe 6.8. In dieser Aufgabe sollen gerichtete Bäume durch Strukturen über einer Signatur mit einem 1-stelligen Funktionssymbol *Elternknoten* repräsentiert werden.

(a) Beschreiben Sie, wie ein gegebener gerichteter Baum $B = (V, E)$ mit $V \neq \emptyset$ durch eine Struktur über der Signatur $\sigma = \{\text{Elternknoten}\}$ modelliert werden kann. Geben Sie die entsprechende Struktur für den folgenden Baum an:



(b) Geben Sie je eine Formel $\varphi(x)$ der Logik erster Stufe an, die ausdrückt, dass der Knoten x

¹Die Struktur \mathfrak{A} spiegelt das Spiel „Stein, Schere, Papier, Echse, Spock“ wider, das Rajesh und Sheldon in Folge 25 der Serie *The Big Bang Theory* spielen, um ihre Meinungsverschiedenheiten auszuräumen – allerdings ohne Erfolg.

- (i) ein Blatt ist,
- (ii) die Wurzel ist,
- (iii) ein innerer Knoten ist,
- (iv) genau zwei Kinder hat.

- (c) Geben Sie je eine Formel $\psi(x, y)$ der Logik erster Stufe an, die ausdrückt, dass es vom Knoten x zum Knoten y
- (i) einen Weg der Länge zwei gibt,
 - (ii) einen einfachen Weg der Länge zwei gibt.

Aufgabe 6.9. Sei $\sigma := \{\dot{E}, \dot{P}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{E} und einem 1-stelligen Relationssymbol \dot{P} . Geben Sie für jede der folgenden Formeln je eine σ -Struktur an, die die Formel erfüllt, und eine, die die Formel nicht erfüllt:

- (a) $\forall x \forall y \forall z ((\dot{E}(x, y) \wedge \dot{E}(y, z)) \rightarrow \dot{E}(x, z))$
- (b) $\forall x \forall y (\dot{E}(x, y) \rightarrow ((\dot{P}(y) \wedge \neg \dot{P}(x)) \vee (\dot{P}(x) \wedge \neg \dot{P}(y))))$
- (c) $(\forall x \forall y (\dot{E}(x, y) \vee \dot{E}(y, x)) \wedge \forall x \forall y ((\dot{E}(x, y) \wedge \dot{E}(y, x)) \rightarrow x \doteq y))$

Aufgabe 6.10. Sei $\sigma := \{\dot{E}, \dot{g}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{E} und einem 1-stelligen Funktionssymbol \dot{g} . Geben Sie für jede der folgenden FO[σ]-Formeln je eine σ -Struktur an, die die Formel erfüllt, und eine, die die Formel nicht erfüllt.

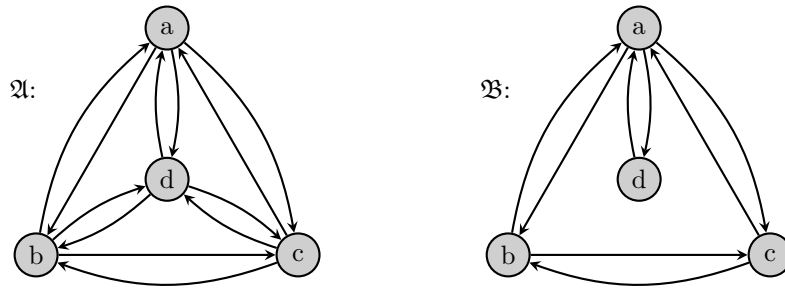
- (a) $\forall x \forall y (\dot{E}(x, y) \rightarrow \dot{E}(y, x))$
- (b) $\forall x \forall y (\dot{E}(x, y) \rightarrow \neg \dot{g}(x) \doteq \dot{g}(y))$
- (c) $(\forall x \forall y \forall z ((\dot{E}(x, y) \wedge \dot{E}(y, z)) \rightarrow \dot{E}(x, z)) \vee \forall x \forall y (\dot{E}(x, y) \leftrightarrow \neg \dot{E}(y, x)))$

Aufgabe 6.11. Bestimmen Sie für jede der folgenden $\{\dot{P}, \dot{E}, \dot{R}, \dot{f}, \dot{g}, \dot{c}\}$ -Formeln, welche Variablen gebunden und welche Variablen frei in der Formel vorkommen:

- (a) $\dot{f}(\dot{c}) \doteq \dot{c}$
- (b) $\forall y \exists y (\dot{R}(x, y) \vee \dot{R}(y, x))$
- (c) $(\dot{P}(x) \vee \neg(\dot{E}(y, z) \rightarrow \dot{f}(x) \doteq \dot{f}(y)))$
- (d) $\forall x (\dot{f}(x) \doteq \dot{g}(y, x) \vee \exists z \dot{E}(x, \dot{g}(x, z)))$
- (e) $(\forall y \neg \dot{E}(y, x) \wedge \exists z (\dot{E}(x, z) \wedge \dot{E}(z, y)))$
- (f) $\exists x \forall y \exists z (\dot{f}(y) \doteq \dot{g}(x, z) \vee \neg \dot{R}(\dot{c}, z, \dot{f}(y)))$

Aufgabe 6.12.

- (a) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{E} . Betrachten Sie die beiden σ -Strukturen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ und $\mathfrak{B} = (A, \dot{E}^{\mathfrak{B}})$, die durch die beiden folgenden Graphen repräsentiert werden.



Geben Sie einen FO[σ]-Satz φ an, so dass $\mathfrak{A} \models \varphi$ und $\mathfrak{B} \models \neg\varphi$ gilt.

(b) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{E} . Geben Sie für die Formel

$$\varphi(x) := \forall y \exists z ((\dot{E}(y, x) \rightarrow \dot{E}(x, z)) \vee x \dot{=} y)$$

eine Struktur \mathfrak{A} und zwei Interpretationen $\mathcal{I}_1 = (\mathfrak{A}, \beta_1)$ und $\mathcal{I}_2 = (\mathfrak{A}, \beta_2)$ an, so dass $\mathcal{I}_1 \models \varphi$ und $\mathcal{I}_2 \models \neg\varphi$ gilt.

Aufgabe 6.13. Betrachten Sie die Kinodatenbank $\mathfrak{A}_{\text{Kino}}$ aus der Vorlesung.

(a) Berechnen Sie für jede der folgenden Formeln φ_i die Relation $\varphi_i(\mathfrak{A}_{\text{Kino}})$ und geben Sie umgangssprachlich an, welche Anfrage durch die Formel φ_i beschrieben wird:

$$\varphi_1(x_K) = \exists x_Z \text{Programm}(x_K, \text{'Capote'}, x_Z)$$

$$\varphi_2(x_S) = \exists x_T (\exists x_R \text{Filme}(x_T, x_R, x_S) \wedge \exists x_K \exists x_Z \text{Programm}(x_K, x_T, x_Z))$$

$$\varphi_3(x_T) = \exists x_K \exists x_Z \left(\text{Programm}(x_K, x_T, x_Z) \wedge \forall y_K \forall y_Z (\text{Programm}(y_K, x_T, y_Z) \rightarrow y_Z \dot{=} x_Z) \right)$$

$$\varphi_4(x_T, x_K, x_A) = \left(\exists x_Z \exists x_{\text{Tel}} (\text{Programm}(x_K, x_T, x_Z) \wedge \text{Orte}(x_K, x_A, x_{\text{Tel}})) \wedge \exists x_S \text{Filme}(x_T, \text{'George Clooney'}, x_S) \right)$$

(b) Finden Sie Formeln der Logik erster Stufe, die die folgenden Anfragen beschreiben:

- (i) Geben Sie die Telefonnummern der Kinos aus, die um 20:15 Uhr oder um 20:30 Uhr eine Vorstellung haben.
- (ii) Geben Sie die Titel aller Filme aus, die in mindestens zwei Kinos laufen.
- (iii) Geben Sie die Titel aller Filme aus, in denen George Clooney mitspielt, aber nicht selbst Regie führt.

Beachten Sie: Es kann sein, dass ein Film mehr als einen Regisseur hat, z.B. Raumpatrouille Orion – Rücksturz ins Kino.

- (iv) Geben Sie die Titel aller Filme aus, deren Schauspieler schon mal in einem Film von Stephen Spielberg mitgespielt haben.
- (v) Geben Sie alle Filme aus, die nur in einem einzelnen Kino gespielt werden, zusammen mit den Anfangszeiten des jeweiligen Filmes.

Aufgabe 6.14.

(a) Welche der folgenden Aussagen stimmen, welche stimmen nicht?

(i) $\forall x \varphi \equiv \neg \exists x \neg \varphi$

(ii) $\exists x(\varphi \wedge \psi) \models (\exists x \varphi \wedge \exists x \psi)$

(iii) $(\exists x \varphi \wedge \exists x \psi) \models \exists x(\varphi \wedge \psi)$

(iv) $\exists x(\varphi \wedge \psi) \equiv (\exists x \varphi \wedge \exists x \psi)$

(v) $\forall x(\varphi \wedge \psi) \equiv (\forall x \varphi \wedge \forall x \psi)$

(vi) $\forall x \varphi \models \exists x \varphi$

(b) Beweisen Sie, dass Ihre Antworten zu (ii), (iii) und (vi) aus (a) korrekt sind.

(c) Zeigen Sie die Korrektheit der Beobachtung (6.40) (c) aus dem Skript, d.h. zeigen Sie, dass für jede Signatur σ und zwei beliebige FO[σ]-Formeln φ und ψ gilt:

$$\varphi \models \psi \iff (\varphi \rightarrow \psi) \text{ ist allgemeingültig.}$$

Aufgabe 6.15. Entscheiden Sie, ob FO[σ_{Graph}]-Formeln φ und ψ mit freien Variablen x , y und z existieren, so dass für jeden gerichteten Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ und jede zu φ und ψ passende Belegung β in \mathfrak{A} gilt:

(a) $(\mathfrak{A}, \beta) \models \varphi \iff$ Es ex. in \mathfrak{A} ein einfacher Weg von $\beta(x)$ über $\beta(y)$ nach $\beta(z)$ der Länge vier.

(b) $(\mathfrak{A}, \beta) \models \psi \iff$ Es ex. in \mathfrak{A} ein einfacher Weg von $\beta(x)$ über $\beta(y)$ nach $\beta(z)$.

Aufgabe 6.16. Entscheiden Sie, ob FO[σ_{Graph}]-Formeln φ und ψ mit freien Variablen x und y existieren, so dass für jeden gerichteten Graphen $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$ und jede zu φ und ψ passende Belegung β in \mathfrak{A} gilt:

(a) (\mathfrak{A}, β) erfüllt $\varphi \iff \beta(x)$ und $\beta(y)$ liegen zusammen auf einem Kreis in \mathfrak{A}

(b) (\mathfrak{A}, β) erfüllt $\psi \iff \beta(x)$ und $\beta(y)$ liegen zusammen auf einem Kreis der Länge vier in \mathfrak{A}

7 Endliche Automaten zur Modellierung von Abläufen

In diesem Kapitel geht es darum, das dynamische Verhalten von Systemen zu beschreiben, z.B.

- die Wirkung von Bedienoperationen auf reale Automaten (z.B. einen Geldautomaten bei der Bank) oder auf die Benutzungsoberflächen von Software-Systemen,
- Schaltfolgen von Ampelanlagen,
- Abläufe von Geschäftsprozessen in Firmen oder
- die Steuerung von Produktionsanlagen.

Solche Abläufe werden modelliert, indem man die Zustände angibt, die ein System annehmen kann, und beschreibt, unter welchen Bedingungen es aus einem Zustand in einen anderen übergehen kann (vgl. das Murmel-Problem aus Beispiel 1.1).

In diesem Kapitel werden die **endlichen Automaten** (bzw. **Transitionssysteme**) als ein grundlegender Kalkül vorgestellt, die sich gut zur Modellierung sequentieller Abläufe eignet und u.a. auch zur Spezifikation von realen oder abstrakten Maschinen genutzt werden kann.

Endliche Automaten

- reagieren auf äußere Ereignisse,
- ändern ggf. ihren „inneren Zustand“ und
- produzieren ggf. eine Ausgabe.

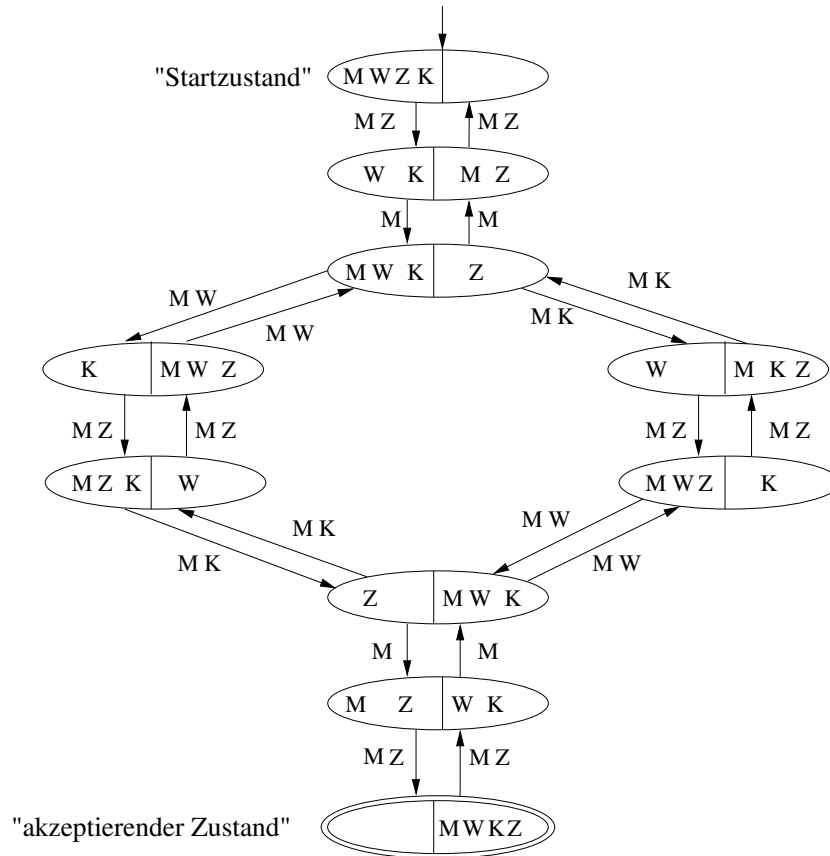
Sie werden z.B. eingesetzt, um

- das Verhalten realer Maschinen zu spezifizieren (z.B. ein Geldautomat oder ein Getränkeautomat),
- das Verhalten von Software-Komponenten zu beschreiben (z.B. das Wirken von Bedienoperationen auf Benutzungsoberflächen von Software-Systemen),
- „Sprachen“ zu spezifizieren, d.h. die Menge aller Ereignisfolgen, die den Automat von seinem „Startzustand“ in einen „akzeptierenden Zustand“ überführen. (Bei der „Flussüberquerung“ aus Beispiel 1.2 sind das genau diejenigen Folgen von „Flussüberquerungsschritten“, mit denen man vom „Startzustand“ $(\{M, W, Z, K\}, \emptyset)$ zum „Zielzustand“ $(\emptyset, \{M, W, Z, K\})$ gelangen kann).

Vor der formalen Definitionen endlicher Automaten betrachten wir zunächst zwei einführende Beispiele:

Beispiel 7.1.

Graphische Darstellung eines endlichen Automaten zum Flussüberquerungs-Problem aus Beispiel 1.2:



Dieser endliche Automat „akzeptiert“ genau diejenigen Folgen von einzelnen Flussüberquerungen, die vom Startzustand in den akzeptierenden Zustand führen.

Beispiel 7.2. Betrachte einen einfachen Getränkeautomat, der folgende Bedienoptionen hat:

- Einwerfen einer 1 €-Münze
- Einwerfen einer 2 €-Münze
- Taste „Geld Rückgabe“ drücken
- Taste „Kaffee kaufen“ drücken

und bei dem man ein einziges Getränk kaufen kann, das 2 € kostet.

Dieser Getränkeautomat kann durch den in Abbildung 7.1 dargestellten endlichen Automaten modelliert werden. Dieser endliche Automat „akzeptiert“ genau diejenigen Folgen von Bedienoperationen, die vom Grundzustand aus wieder in den Grundzustand führen.

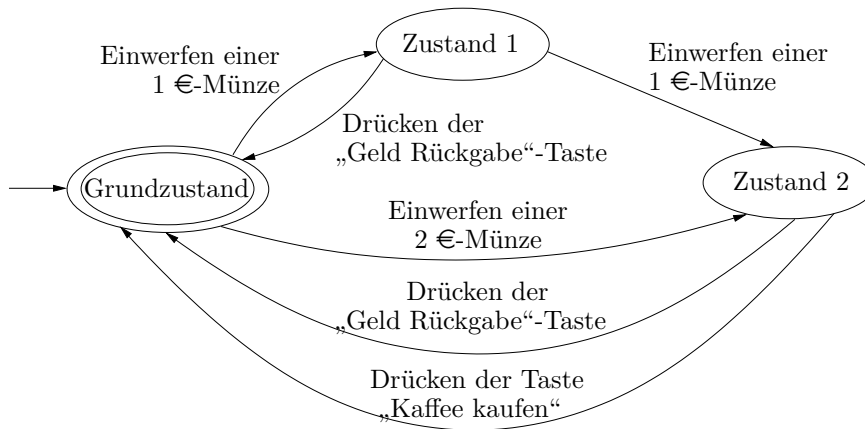


Abbildung 7.1: Graphische Darstellung eines endlichen Automaten, der einen Getränkeautomaten modelliert

7.1 Deterministische endliche Automaten

Definition 7.3 (DFA). Ein **deterministischer endlicher Automat**¹ (kurz: **DFA**)

DFA

$$A = (\Sigma, Q, \delta, q_0, F)$$

besteht aus:

- einer endlichen Menge Σ , dem so genannten **Eingabealphabet**, Eingabealphabet
- einer endlichen Mengen Q , der so genannten **Zustandsmenge** (die Elemente aus Q werden **Zustände** genannt), Zustandsmenge
- einer partiellen Funktion δ von $Q \times \Sigma$ nach Q , der so genannten (Zustands-) **Übergangsfunktion** (oder **Überföhrungsfunktion**), Übergangsfunktion
- einem Zustand $q_0 \in Q$, dem so genannten **Startzustand**, Startzustand
- einer Menge $F \subseteq Q$, der so genannten Menge der **Endzustände** bzw. **akzeptierenden Zustände** (der Buchstabe F steht für „final states“, also „Endzustände“). Endzustände

Graphische Darstellung endlicher Automaten:

Endliche Automaten lassen sich anschaulich durch beschriftete Graphen darstellen (vgl. Beispiel 7.1 und 7.2):

- Für jeden Zustand $q \in Q$ gibt es einen durch $\circlearrowleft q$ dargestellten Knoten.
- Der Startzustand q_0 wird durch einen in ihn hinein föhrenden Pfeil markiert, d.h.:



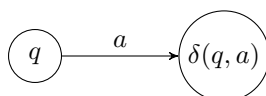
¹engl.: deterministic finite automaton, kurz: DFA

- Jeder akzeptierende Zustand $q \in F$ wird durch eine doppelte Umrandung markiert, d.h.:



- Ist $q \in Q$ ein Zustand und $a \in \Sigma$ ein Symbol aus dem Alphabet, so dass $(q, a) \in \text{Def}(\delta)$ liegt, so gibt es in der graphischen Darstellung von A einen mit dem Symbol a beschrifteten

Pfeil von Knoten q zu Knoten $\delta(q, a)$, d.h.:



Beispiel 7.4. Die graphische Darstellung aus Beispiel 7.2 repräsentiert den DFA $A = (\Sigma, Q, \delta, q_0, F)$ mit

- $\Sigma = \{\text{Einwerfen einer 1 €-Münze, Einwerfen einer 2 €-Münze, Drücken der „Geld Rückgabe“-Taste, Drücken der Taste „Kaffee kaufen“}\}$
- $Q = \{\text{Grundzustand, Zustand 1, Zustand 2}\}$
- $q_0 = \text{Grundzustand}$
- $F = \{\text{Grundzustand}\}$
- δ ist die partielle Funktion von $Q \times \Sigma$ nach Q mit

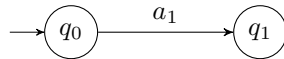
$$\begin{aligned} \delta(\text{Grundzustand, Einwerfen einer 1 €-Münze}) &= \text{Zustand 1,} \\ \delta(\text{Grundzustand, Einwerfen einer 2 €-Münze}) &= \text{Zustand 2,} \\ \delta(\text{Zustand 1, Einwerfen einer 1 €-Münze}) &= \text{Zustand 2,} \\ \delta(\text{Zustand 1, Drücken der „Geld Rückgabe“-Taste}) &= \text{Grundzustand,} \\ \delta(\text{Zustand 2, Drücken der „Geld Rückgabe“-Taste}) &= \text{Grundzustand,} \\ \delta(\text{Zustand 2, Drücken der Taste „Kaffee kaufen“}) &= \text{Grundzustand.} \end{aligned}$$

Die Verarbeitung eines Eingabeworts durch einen DFA:

Ein DFA $A = (\Sigma, Q, \delta, q_0, F)$ erhält als Eingabe ein Wort $w \in \Sigma^*$, das eine Folge von „Aktionen“ oder „Bedienoperationen“ repräsentiert. Bei Eingabe eines Worts w wird A im Startzustand q_0 gestartet. Falls w das leere Wort ist, d.h. $w = \varepsilon$, so passiert nichts weiter. Falls w ein Wort von der Form $a_1 \cdots a_n$ mit $n \in \mathbb{N}_{>0}$ und $a_1, \dots, a_n \in \Sigma$, so geschieht bei der Verarbeitung von w durch A folgendes: Durch Lesen der ersten Buchstabens von w , also a_1 , geht der Automat in den Zustand $q_1 := \delta(q_0, a_1)$ über. In der graphischen Darstellung von A wird der Zustand



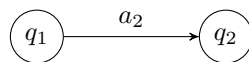
durch die mit a_1 beschriftete Kante verlassen, und q_1 ist der Endknoten dieser Kante, d.h.



Dies ist allerdings nur möglich, wenn (q_0, a_1) im Definitionsbereich von δ liegt (also $(q_0, a_1) \in \text{Def}(\delta)$) — d.h. wenn es in der graphischen Darstellung von A eine mit a_1 beschriftete Kante gibt, die aus Zustand q_0 herausführt. Falls es keine solche Kante gibt, d.h. falls $(q_0, a_1) \notin \text{Def}(\delta)$, so „stürzt A ab“, und die Verarbeitung des Wortes w ist beendet. Ansonsten ist A nach Lesen des ersten Symbols von w im Zustand $q_1 := \delta(q_0, a_1)$. Durch Lesen des zweiten Symbols von w , also a_2 , geht A nun in den Zustand $q_2 := \delta(q_1, a_2)$ über — bzw. „stürzt ab“, falls $(q_1, a_2) \notin \text{Def}(\delta)$. In der graphischen Darstellung wird



durch die mit a_2 beschriftete Kante verlassen (falls eine solche Kante existiert)



und der Automat landet in Zustand q_2 , wobei $q_2 := \delta(q_1, a_2)$ der Endknoten dieser Kante ist.

Auf diese Weise wird nach und nach das gesamte Eingabewort $w = a_1 \cdots a_n$ abgearbeitet. Ausgehend vom Startzustand q_0 werden dabei nacheinander Zustände q_1, \dots, q_n erreicht. In der graphischen Darstellung von A entspricht dies gerade dem Durchlaufen eines Weges der Länge n , der im Knoten



startet und dessen Kanten mit den Buchstaben a_1, \dots, a_n beschriftet sind. Der Knoten



der am Ende dieses Weges erreicht wird (falls der Automat nicht zwischendurch abstürzt, d.h. falls es überhaupt einen mit a_1, \dots, a_n beschrifteten in



startenden Weg gibt), ist **der von A bei Eingabe w erreichte Zustand**, kurz: $q_n = \widehat{\delta}(q_0, w)$. Im Fall, dass A bei Eingabe von w zwischendurch abstürzt, sagen wir „ $\widehat{\delta}(q_0, w)$ ist undefiniert“ und schreiben kurz $\widehat{\delta}(q_0, w) = \perp$.

Definition 7.5 (Die erweiterte Übergangsfunktion $\widehat{\delta}$ eines DFA).

Sei $A := (\Sigma, Q, \delta, q_0, F)$ ein DFA. Sei \perp ein Symbol, das nicht in der Zustandsmenge Q liegt, und sei $Q_\perp := Q \cup \{\perp\}$. Die Funktion

$$\widehat{\delta} : Q_\perp \times \Sigma^* \rightarrow Q_\perp$$

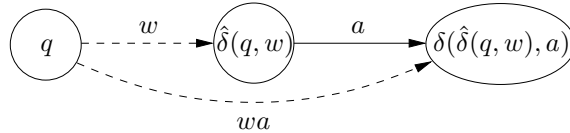
ist rekursiv wie folgt definiert:

- F.a. $w \in \Sigma^*$ ist $\widehat{\delta}(\perp, w) := \perp$.
- F.a. $q \in Q$ ist $\widehat{\delta}(q, \varepsilon) := q$.

- F.a. $q \in Q$, $w \in \Sigma^*$ und $a \in \Sigma$ gilt für $q' := \hat{\delta}(q, w)$:

$$\hat{\delta}(q, wa) := \begin{cases} \perp & \text{falls } (q', a) \notin \text{Def}(\delta), \\ \delta(q', a) & \text{falls } (q', a) \in \text{Def}(\delta). \end{cases}$$

Graphische Darstellung:



Insgesamt gilt: Falls $\hat{\delta}(q_0, w) \neq \perp$, so ist $\hat{\delta}(q_0, w)$ der Zustand, der durch Verarbeiten des Worts w erreicht wird. Falls $\hat{\delta}(q_0, w) = \perp$, so stürzt der Automat beim Verarbeiten des Worts w ab.

Die von einem DFA akzeptierte Sprache:

Das Eingabewort w wird vom DFA A **akzeptiert**, falls

$$\hat{\delta}(q_0, w) \in F,$$

d.h., A stürzt bei Eingabe von w nicht ab, und der durch Verarbeiten des Worts w erreichte Zustand gehört zur Menge F der akzeptierenden Zustände.

In der graphischen Darstellung von A heißt das für ein Eingabewort $w = a_1 \cdots a_n$, dass es einen in



startenden Weg der Länge n gibt, dessen Kanten mit den Symbolen a_1, \dots, a_n beschriftet sind, und der in einem akzeptierenden Zustand endet.

Definition 7.6 (Die von einem DFA A akzeptierte Sprache $L(A)$).

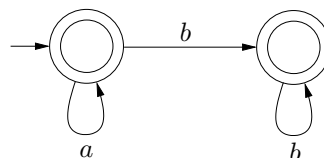
Die von einem DFA $A = (\Sigma, Q, \delta, q_0, F)$ akzeptierte Sprache $L(A)$ ist

$$L(A) := \{w \in \Sigma^* : \hat{\delta}(q_0, w) \in F\}.$$

D.h.: Ein Wort $w \in \Sigma^*$ gehört genau dann zur Sprache $L(A)$, wenn es vom DFA A akzeptiert wird.

Beispiel 7.7. Der Einfachheit halber betrachten wir das Eingabealphabet $\Sigma := \{a, b\}$.

(a) Sei A_1 ein DFA mit folgender graphischer Darstellung:



- A_1 akzeptiert z.B. folgende Worte: $\varepsilon, a, b, aaa, aaab, aaaabbbb, bbb, \dots$
- A_1 „stürzt ab“ z.B. bei Eingabe von $ba, aabba$.

Insgesamt gilt:

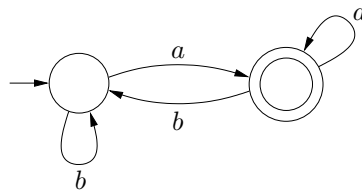
$$L(A_1) = \{a^n b^m : n \in \mathbb{N}, m \in \mathbb{N}\}$$

(**Notation:** $a^n b^m$ bezeichnet das Wort $a \cdots ab \cdots b$ der Länge $n + m$, das aus n a 's gefolgt von m b 's besteht, z.B. ist $a^3 b^4$ das Wort $aaabbbb$)

(b) Die graphische Darstellung eines DFA A_2 mit

$$L(A_2) = \{w \in \{a, b\}^* : \text{der letzte Buchstabe von } w \text{ ist ein } a\}$$

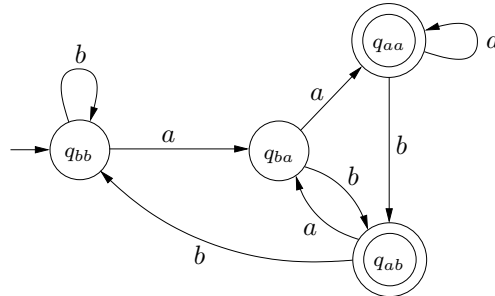
ist



(c) Die graphische Darstellung eines DFA A_3 mit

$$L(A_3) = \{w \in \{a, b\}^* : \text{der vorletzte Buchstabe von } w \text{ ist ein } a\}$$


ist



Bemerkung 7.8.

- Die in Definition 7.3 eingeführten DFAs $A = (\Sigma, Q, \delta, q_0, F)$ heißen **deterministisch**, weil es zu jedem Paar $(q, a) \in Q \times \Sigma$ höchstens einen „Nachfolgezustand“ $\delta(q, a)$ gibt (da δ eine partielle Funktion von $Q \times \Sigma$ nach Q ist). Beim Verarbeiten eines Eingabeworts ist daher zu jedem Zeitpunkt klar, ob A „abstürzt“ oder nicht — und falls nicht, welchen eindeutig festgelegten Nachfolgezustand A annimmt. deterministisch
- Ein DFA A heißt **vollständig**, wenn die Übergangsfunktion δ eine totale Funktion $\delta: Q \times \Sigma \rightarrow Q$ ist. vollständig

Beispiele: Die DFAs A_2 und A_3 aus Beispiel 7.7 sind vollständig; der DFA A_1 nicht. Die DFAs aus Beispiel 7.1 und 7.2 sind nicht vollständig.

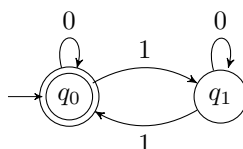
Für die graphische Darstellung eines DFAs gilt: Der DFA ist genau dann vollständig, wenn für jeden Zustand q gilt: Für jedes Symbol $a \in \Sigma$ gibt es genau eine aus  herausführende Kante, die mit a beschriftet ist.

Beachte: In manchen Büchern weicht die Definition von DFAs von Definition 7.3 ab, indem gefordert wird, dass DFAs grundsätzlich vollständig sein müssen.

Ein Anwendungsbeispiel: Paritätscheck durch einen DFA

Bei der Speicherung von Daten auf einem Speichermedium eines Computers werden Informationen durch Worte über dem Alphabet $\{0, 1\}$ kodiert. Um eventuelle Fehler beim Übertragen der Daten erkennen zu können, wird der Kodierung oft ein so genanntes **Paritätsbit** angehängt, das bewirkt, dass die Summe der Einsen im resultierenden Wort gerade ist.

Für ein beliebiges Wort $w \in \{0, 1\}^*$ sagen wir “ w besteht den Paritätscheck”, falls die Anzahl der Einsen in w gerade ist. Zur effizienten Durchführung eines Paritätschecks für ein gegebenes Wort $w \in \{0, 1\}^*$ kann man den folgenden DFA A benutzen:



Für diesen Automaten gilt: $L(A) = \{w \in \{0, 1\}^* : w \text{ besteht den Paritätscheck}\}$.

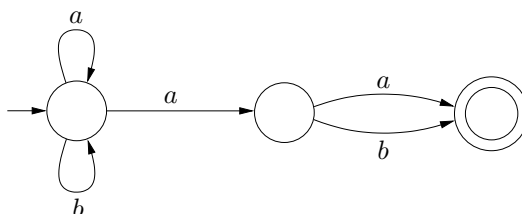
7.2 Nichtdeterministische endliche Automaten

Für manche Modellierungsaufgaben ist die Forderung, dass es für jeden Zustand q und jedes Eingabesymbol a höchstens einen Nachfolgezustand $\delta(q, a)$ gibt, zu restriktiv, da man in manchen Zuständen für den Übergang mit einem Symbol a mehrere Möglichkeiten angeben will, ohne festzulegen, welche davon gewählt wird. Solche Entscheidungsfreiheiten in der Modellierung von Abläufen bezeichnet man als **nichtdeterministisch**. Nichtdeterministische Modelle sind oft einfacher aufzustellen und leichter zu verstehen als deterministische.

Beispiel 7.9. In Beispiel 7.7(c) haben wir einen (recht komplizierten) DFA A_3 mit

$$L(A_3) = \{w \in \{a, b\}^* : \text{der vorletzte Buchstabe von } w \text{ ist ein } a\}$$

kennengelernt. Dieselbe Sprache wird auch von dem deutlich einfacheren **nichtdeterministischen endlichen Automaten** (kurz: NFA) A_4 mit der folgenden graphischen Darstellung akzeptiert:



Ein Eingabewort $w \in \{a, b\}^*$ wird von dem NFA A_4 genau dann akzeptiert, wenn es in der graphischen Darstellung mindestens einen Weg gibt, der im Startzustand $\rightarrow \bigcirc$ beginnt, dessen Kanten mit w beschriftet sind, und der im akzeptierenden Zustand $\bigcirc \bigcirc$ endet.

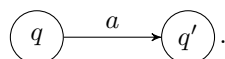
Die präzise Definition von NFAs wird in der folgenden Definition gegeben.

Definition 7.10 (NFA). Ein **nichtdeterministischer endlicher Automat**² (kurz: **NFA**) $A = (\Sigma, Q, \delta, q_0, F)$ besteht aus:

- einer endlichen Menge Σ , dem so genannten **Eingabealphabet**, Eingabealphabet
- einer endlichen Menge Q , der so genannten **Zustandsmenge**, Zustandsmenge
- einer Funktion³ $\delta: Q \times \Sigma \rightarrow \mathcal{P}(Q)$, der so genannten **Übergangsfunktion**, die jedem Zustand $q \in Q$ und jedem Symbol $a \in \Sigma$ eine **Menge $\delta(q, a)$ von möglichen Nachfolgezuständen** zuordnet (beachte: möglicherweise ist $\delta(q, a) = \emptyset$ — dann „stürzt“ der Automat ab, wenn er im Zustand q ist und das Symbol a liest), Übergangsfunktion
- einem Zustand $q_0 \in Q$, dem so genannten **Startzustand**, Startzustand
- einer Menge $F \subseteq Q$, der so genannten Menge der **Endzustände** bzw. **akzeptierenden Zustände**. Endzustände

Graphische Darstellung von NFAs:

Die graphische Darstellung von NFAs erfolgt analog der graphischen Darstellung von DFAs. Ist $q \in Q$ ein Zustand und ist $a \in \Sigma$ ein Eingabesymbol, so gibt es **für jeden Zustand $q' \in \delta(q, a)$** in der graphischen Darstellung des NFAs einen mit dem Symbol a beschrifteten Pfeil von Knoten $\bigcirc(q)$ zu Knoten $\bigcirc(q')$, d.h.



Die von einem NFA A akzeptierte Sprache $L(A)$:

Definition 7.11. Sei $A = (\Sigma, Q, \delta, q_0, F)$ ein NFA.

- (a) Sei $n \in \mathbb{N}$ und sei $w = a_1 \cdots a_n$ ein Eingabewort der Länge n . Das Wort w wird genau dann vom NFA A akzeptiert, wenn es in der graphischen Darstellung von A einen im Startzustand



beginnenden Weg der Länge n gibt, dessen Kanten mit den Symbolen a_1, \dots, a_n beschriftet sind und der in einem akzeptierenden Zustand endet.

²engl.: non-deterministic finite automaton, kurz: NFA

³Zur Erinnerung: $\mathcal{P}(Q)$ bezeichnet die **Potenzmenge** von Q .

(b) Die von A akzeptierte Sprache $L(A)$ ist

$$L(A) := \{w \in \Sigma^* : A \text{ akzeptiert } w\}.$$

Ähnlich wie bei DFAs können wir auch für NFAs eine erweiterte Überföhrungsfunktion $\widehat{\delta}$ definieren. Für einen Zustand q und ein Eingabewort w gibt $\widehat{\delta}(q, w)$ die **Menge** aller Zustände an, die durch Verarbeiten des Worts w erreicht werden können, wenn der Automat im Zustand q beginnt. Dies wird durch die folgende Definition präzisiert.

Definition 7.12 (Die erweiterte Übergangsfunktion $\widehat{\delta}$ eines NFA).

Sei $A := (\Sigma, Q, \delta, q_0, F)$ ein NFA. Die Funktion

$$\widehat{\delta} : Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$$

ist rekursiv wie folgt definiert:

- F.a. $q \in Q$ ist $\widehat{\delta}(q, \varepsilon) := \{q\}$.
- F.a. $q \in Q$, $w \in \Sigma^*$ und $a \in \Sigma$ ist $\widehat{\delta}(q, wa) := \bigcup_{q' \in \widehat{\delta}(q, w)} \delta(q', a)$.

Man beachte, dass ein Wort w genau dann von einem NFA $A := (\Sigma, Q, \delta, q_0, F)$ akzeptiert wird, wenn gilt:

$$\widehat{\delta}(q_0, w) \cap F \neq \emptyset.$$

Somit ist

$$L(A) = \{w \in \Sigma^* : \widehat{\delta}(q_0, w) \cap F \neq \emptyset\}.$$

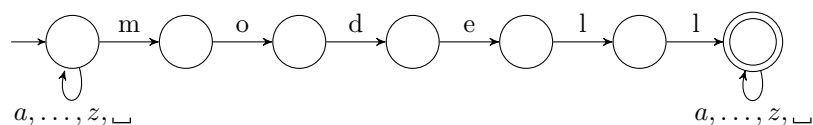
Ein Anwendungsbeispiel: Stichwort-Suche in Texten

Gegeben: Ein Stichwort, z.B. „modell“

Eingabe: Ein Text, der aus den Buchstaben „a“ bis „z“ sowie dem Leerzeichen „_“ besteht

Frage: Kommt das Stichwort „modell“ irgendwo im Eingabetext vor? Der Eingabetext soll genau dann akzeptiert werden, wenn er das Stichwort „modell“ enthält.

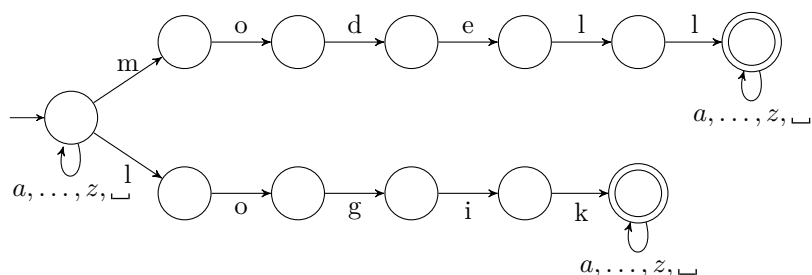
Graphische Darstellung eines NFAs, der dies bewerkstelligt:



Auf ähnliche Art können auch Varianten dieser Stichwortsuche behandelt werden, zum Beispiel die Frage

Kommt mindestens eins der Stichworte „modell“ bzw. „logik“ im Eingabetext vor?

Graphische Darstellung eines NFAs, der dies bewerkstelligt:



7.3 Äquivalenz von NFAs und DFAs

Nachdem wir mit NFAs und DFAs zwei Sorten von endlichen Automaten kennen gelernt haben, drängt sich natürlich die Frage auf, ob NFAs wirklich “mehr” können als DFAs. Der folgende Satz beantwortet diese Frage mit “nein”.

Satz 7.13. Für jeden NFA $A = (\Sigma, Q, \delta, q_0, F)$ gibt es einen DFA $A' = (\Sigma, Q', \delta', q'_0, F')$ mit $L(A') = L(A)$. D.h.: NFAs und DFAs können genau dieselben Sprachen akzeptieren.

Beweis: Sei $A = (\Sigma, Q, \delta, q_0, F)$ der gegebene NFA.

Idee: Wir konstruieren einen DFA $A' = (\Sigma, Q', \delta', q'_0, F')$, der in seinem aktuellen Zustand $q' \in Q'$ die Menge aller Zustände abspeichert, in denen der Automat A in der aktuellen Situation sein könnte. Wir definieren die Komponenten von A' daher wie folgt:

- Eingabealphabet Σ ,
- Zustandsmenge $Q' := \mathcal{P}(Q)$,
- Startzustand $q'_0 := \{q_0\}$,
- Endzustandsmenge $F' := \{X \in Q' : X \cap F \neq \emptyset\}$,
- Übergangsfunktion $\delta' : Q' \times \Sigma \rightarrow Q'$, wobei für alle $X \in Q' = \mathcal{P}(Q)$ und alle $a \in \Sigma$ gilt:

$$\delta'(X, a) := \bigcup_{q \in X} \delta(q, a).$$

Per Induktion nach n kann man leicht nachweisen (Details: **Übung!**), dass für alle $n \in \mathbb{N}$ und jedes $w \in \Sigma^*$ mit $|w| = n$ gilt:

$$\widehat{\delta}'(q'_0, w) = \widehat{\delta}(q_0, w).$$

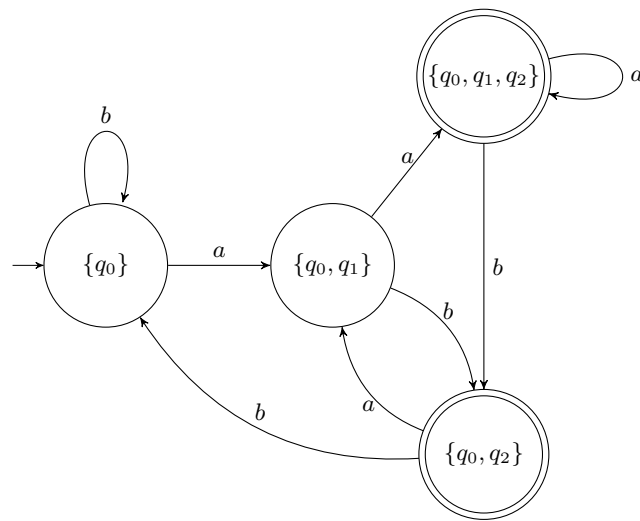
Daraus folgt, dass für jedes Eingabewort $w \in \Sigma^*$ gilt:

$$\begin{aligned} w \in L(A') &\iff \widehat{\delta}'(q'_0, w) \in F' \\ &\iff \widehat{\delta}'(q'_0, w) \cap F \neq \emptyset \\ &\iff \widehat{\delta}(q_0, w) \cap F \neq \emptyset \\ &\iff w \in L(A). \end{aligned}$$

Insbesondere ist daher $L(A') = L(A)$. □

Die im obigen Beweis durchgeführte Konstruktion eines DFAs A' wird auch **Potenzmengenkonstruktion** (engl: **subset construction**) genannt.

Beispiel 7.14. Wir führen die Potenzmengenkonstruktion an dem NFA A_3 aus Beispiel 7.9 durch, wobei wir die Zustände von A_3 , von links nach rechts gelesen, mit q_0 , q_1 und q_2 bezeichnen. Im folgenden ist die graphische Darstellung des aus der Potenzmengen-Konstruktion resultierenden DFAs A'_3 angegeben, wobei nur solche Zustände aus $\mathcal{P}(\{q_0, q_1, q_2\})$ berücksichtigt werden, die vom Startzustand $q'_0 := \{q_0\}$ aus erreicht werden können:



7.4 Das Pumping-Lemma für reguläre Sprachen

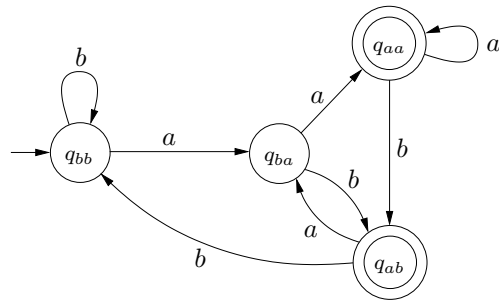
Definition 7.15 (reguläre Sprachen). Sei Σ ein endliches Alphabet. Eine Sprache $L \subseteq \Sigma^*$ heißt **regulär**, falls es einen NFA $A = (\Sigma, Q, \delta, q_0, F)$ mit $L(A) = L$ gibt.

Klar: Um zu zeigen, dass eine Sprache $L \subseteq \Sigma^*$ regulär ist, reicht es, einen NFA oder einen DFA A mit $L(A) = L$ zu finden.

Frage: Wie kann man nachweisen, dass eine bestimmte Sprache $L \subseteq \Sigma^*$ **nicht** regulär ist?

Ein nützliches Werkzeug dazu ist der folgende Satz 7.17, der unter dem Namen **Pumping-Lemma** bekannt ist. Bevor wir den Satz präzise angeben, betrachten wir zunächst ein Beispiel:

Beispiel 7.16. Sei A_3 der endliche Automat aus Beispiel 7.7(c), d.h.



A_3 akzeptiert beispielsweise das Eingabewort

$$x = babaa,$$

indem er nacheinander die Zustände

$$q_{bb} \xrightarrow{b} q_{bb} \xrightarrow{a} q_{ba} \xrightarrow{b} q_{ab} \xrightarrow{a} q_{ba} \xrightarrow{a} q_{aa}$$

besucht. Dieser Weg durch die graphische Darstellung von A_3 enthält einen Kreis

$$q_{ba} \xrightarrow{b} q_{ab} \xrightarrow{a} q_{ba},$$

der beliebig oft durchlaufen werden kann, so dass man (egal ob der Kreis 0-mal, 1-mal, 2-mal, 3-mal, ... durchlaufen wird) jedesmal ein Eingabewort erhält, das von A_3 akzeptiert wird, nämlich für jede Zahl $i \geq 0$ das Eingabewort $ba(ba)^i a$.

Der folgende Satz 7.17 beruht auf demselben Prinzip sowie der Tatsache, dass in jedem Graph auf z Knoten gilt: Jeder Weg der Länge $\geq z$ enthält einen Kreis (d.h. mindestens ein Knoten wird auf dem Weg mehr als 1-mal besucht).

Satz 7.17 (Pumping-Lemma). Sei Σ ein endliches Alphabet.

Für jede reguläre Sprache $L \subseteq \Sigma^*$ gibt es eine Zahl $n_L \in \mathbb{N}_{>0}$ (die so genannte **Pumpingkonstante**), so dass für jedes Wort $z \in L$ der Länge $|z| \geq n_L$ gilt: Es gibt eine Zerlegung von z in Worte $u, v, w \in \Sigma^*$, so dass die folgenden Bedingungen erfüllt sind:

- (1) $z = uvw$
- (2) $|uv| \leq n_L$
- (3) $|v| \geq 1$
- (4) für jedes $i \in \mathbb{N}$ gilt: $uv^i w \in L$.
(d.h.: $uw \in L, uvw \in L, uvvw \in L, uvvww \in L, \dots$)

Pumping-Lemma

Pumpingkonstante

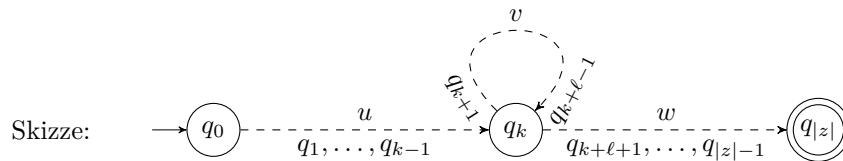
Beweis: Da L regulär ist, gibt es einen NFA $A = (\Sigma, Q, \delta, q_0, F)$ mit $L(A) = L$. Sei $n_L := |Q|$ die Anzahl der Zustände von A .

Sei nun $z \in \Sigma^*$ ein beliebiges Wort der Länge $|z| \geq n_L$, das in L liegt, d.h. das von A akzeptiert wird. Sei $q_0, q_1, \dots, q_{|z|}$ die Folge von Zuständen, die A beim Verarbeiten von z durchläuft. Da $|z| \geq n_L = |Q|$ ist, können die Zustände q_0, q_1, \dots, q_{n_L} nicht alle verschieden sein. Daher gibt es ein $k \geq 0$ und ein $\ell \geq 1$, so dass $q_k = q_{k+\ell}$ und $k + \ell \leq n_L$. Wir wählen folgende Zerlegung von z in Worte $u, v, w \in \Sigma^*$:

- u besteht aus den ersten k Buchstaben von z .
- v besteht aus den nächsten ℓ Buchstaben von z .
- w besteht aus den restlichen Buchstaben von z .

Offensichtlich gilt:

- (1) $z = uvw$,
- (2) $|uv| = k + \ell \leq n_L$,
- (3) $|v| = \ell \geq 1$.



Daher gilt für jedes $i \geq 0$: A akzeptiert das Eingabewort $uv^i w$, d.h. $uv^i w \in L$. □

Beobachtung 7.18. Unter Verwendung des Pumping-Lemmas kann man nachweisen, dass gewisse Sprachen L **nicht** regulär sind. Dies lässt sich gut durch ein 2-Personen-Spiel illustrieren, bei dem “wir” gegen einen “Gegner” spielen. Das Spiel zum Nachweis, dass L nicht regulär ist, wird nach folgenden Regeln gespielt:

- (1) Der “Gegner” wählt eine Pumpingkonstante $n_L \geq 1$.
- (2) “Wir” wählen ein Wort $z \in L$ der Länge $|z| \geq n_L$.
- (3) Der “Gegner” zerlegt z in Worte u, v, w , so dass gilt: $|uv| \leq n_L$ und $|v| \geq 1$.
- (4) “Wir” pumpen auf oder ab, d.h. wir wählen eine Zahl $i \geq 2$ oder $i = 0$.
- (5) “Wir” haben gewonnen, wenn $uv^i w \notin L$ ist; ansonsten hat der “Gegner” gewonnen.

Aus dem Pumping-Lemma folgt: Wenn “wir” eine Gewinnstrategie in diesem Spiel haben, dann ist die Sprache L nicht regulär.

Beispiel 7.19. Sei $\Sigma := \{a, b\}$. Die Sprache $L := \{a^n b^n : n \in \mathbb{N}\}$ ist **nicht** regulär. (Zum Vergleich: Gemäß Lemma 7.7(a) ist die Sprache $L_1 := \{a^n b^m : n \in \mathbb{N}, m \in \mathbb{N}\}$ regulär.)

Beweisidee: Wir versuchen, eine Gewinnstrategie im Spiel zu finden.

- (1) Der “Gegner” wählt eine Pumpingkonstante $n_L \geq 1$.
- (2) “Wir” wählen das Wort $z := a^{n_L} b^{n_L}$ (beachte: $z \in L$ und $|z| \geq n_L$, d.h. wir haben regelkonform gespielt).
- (3) Der “Gegner” zerlegt z in Worte u, v, w , so dass gilt: $|uv| \leq n_L$ und $|v| \geq 1$.
- (4) “Wir” beobachten, dass uv kein b enthalten kann, da $uvw = z = a^{n_L} b^{n_L}$ und $|uv| \leq n_L$ ist. Außerdem muss es eine Zahl $\ell \geq 1$ geben, so dass $v = a^\ell$ ist. Wir entscheiden uns daher, die Zahl “ $i = 0$ ” zu wählen (also “abzupumpen”).

- (5) Es gilt nun: $uv^i w = uw$. Dieses Wort sieht wie z aus, nur dass ein Teilstück der Form a^ℓ gelöscht wurde. D.h. $uw = a^{n_L - \ell} b^{n_L}$. Wegen $\ell \geq 1$ liegt dieses Wort nicht in der Menge $L = \{a^n b^n : n \in \mathbb{N}\}$.

Die hier gefundene Beweisidee schreiben wir nun noch als formalen Beweis auf.

Beweis: Durch Widerspruch.

Angenommen, L ist regulär. Dann sei $n_L \in \mathbb{N}_{>0}$ die gemäß dem Pumping-Lemma (Satz 7.17) gewählte Pumpingkonstante. Betrachte das Wort $z := a^{n_L} b^{n_L}$. Klar: $z \in L$ und $|z| \geq n_L$. Gemäß dem Pumping-Lemma gibt es eine Zerlegung von z in Worte $u, v, w \in \{a, b\}^*$, so dass

- (1) $z = uvw$
- (2) $|uv| \leq n_L$
- (3) $|v| \geq 1$
- (4) f.a. $i \in \mathbb{N}$ gilt: $uv^i w \in L$.

Wegen $z = a^{n_L} b^{n_L} = uvw$ und $|uv| \leq n_L$ und $|v| \geq 1$ gibt es eine Zahl $\ell \in \mathbb{N}_{>0}$, so dass $v = a^\ell$. Wegen (4) gilt insbesondere für $i = 0$, dass $uw \in L$. Wegen $z = uvw = a^{n_L} b^{n_L}$ und $v = a^\ell$ mit $\ell \geq 1$ gilt

$$uw = a^{n_L - \ell} b^{n_L} \stackrel{\ell \geq 1}{\notin} \{a^n b^n : n \in \mathbb{N}\} = L.$$

Widerspruch! □

Beispiel 7.20. Sei $\Sigma := \{a\}$. Die Sprache $L := \{w \in \{a\}^* : |w| \text{ ist eine Primzahl}\}$ ist **nicht** regulär.

Beweis: Durch Widerspruch.

Angenommen, L ist regulär. Dann sei $n_L \in \mathbb{N}_{>0}$ die gemäß dem Pumping-Lemma (Satz 7.17) gewählte Pumpingkonstante. Da es unendlich viele Primzahlen gibt, gibt es auch eine Primzahl p mit $p \geq n_L + 2$. Sei p solch eine Primzahl. Betrachte nun das Wort $z := a^p$. Klar: $z \in L$ und $|z| \geq n_L$. Gemäß dem Pumping-Lemma gibt es also eine Zerlegung von z in Worte $u, v, w \in \{a\}^*$, so dass

- (1) $z = uvw$,
- (2) $|uv| \leq n_L$,
- (3) $|v| \geq 1$,
- (4) f.a. $i \in \mathbb{N}$ gilt: $uv^i w \in L$.

Wegen (4) gilt insbesondere für $i := |uw|$, dass $uv^i w \in L$. Es gilt dann

$$|uv^i w| = |uw| + i \cdot |v| = |uw| + |uw| \cdot |v| = |uw| \cdot (1 + |v|).$$

Wegen

$$|uw| \geq |w| = |z| - |uv| \stackrel{|uv| \leq n_L}{\geq} p - n_L \stackrel{p \geq n_L + 2}{\geq} 2 \quad \text{und} \quad 1 + |v| \stackrel{|v| \geq 1}{\geq} 2$$

ist $|uv^i w|$ daher keine Primzahl, d.h. $uv^i w \notin L$. Widerspruch! □

Auf ähnliche Weise kann man auch zeigen, dass keine der folgenden Sprachen regulär ist:

- $\{a^n b^m : n, m \in \mathbb{N} \text{ mit } n \leq m\}$
- $\{ww : w \in \{a, b\}^*\}$
- $\{w : w \in \{a, b\}^*, w = w^R\}$ („Palindrome“)
- $\{w \in \{a\}^* : |w| \text{ ist eine Quadratzahl, d.h. ex. } n \in \mathbb{N} \text{ s.d. } |w| = n^2\}$.

7.5 Reguläre Ausdrücke

Reguläre Ausdrücke beschreiben Mengen von Worten, die nach bestimmten Regeln bzw. „Mustern“ aufgebaut sind.

Beispiel 7.21. Die Menge aller Wörter über dem Alphabet $\{a, b\}$, deren vorletzter Buchstabe ein a ist, wird durch den folgenden regulären Ausdruck beschrieben:

$$(a|b)^* a (a|b)$$

Definition 7.22 (Reguläre Ausdrücke: Syntax). Sei Σ ein endliches Alphabet. Die Menge aller **regulären Ausdrücke über Σ** ist rekursiv wie folgt definiert:

Basisregeln:

- \emptyset ist ein regulärer Ausdruck über Σ („leere Menge“).
- ε ist ein regulärer Ausdruck über Σ („leeres Wort“).
- Für jedes $a \in \Sigma$ gilt: a ist ein regulärer Ausdruck über Σ .

Rekursive Regeln:

- Ist R ein regulärer Ausdruck über Σ , so ist auch R^* ein regulärer Ausdruck über Σ („Kleene-Stern“).
- Sind R und S reguläre Ausdrücke über Σ , so ist auch
 - $(R \cdot S)$ ein regulärer Ausdruck über Σ („Konkatenation“).
 - $(R|S)$ ein regulärer Ausdruck über Σ („Vereinigung“).

Definition 7.23 (Reguläre Ausdrücke: Semantik). Sei Σ ein endliches Alphabet. Jeder reguläre Ausdruck R über Σ **beschreibt** (oder definiert) eine Sprache $L(R) \subseteq \Sigma^*$, die induktiv wie folgt definiert ist:

- $L(\emptyset) := \emptyset$.
- $L(\varepsilon) := \{\varepsilon\}$.
- Für jedes $a \in \Sigma$ gilt: $L(a) := \{a\}$.
- Ist R ein regulärer Ausdruck über Σ , so ist

$$L(R^*) := \{\varepsilon\} \cup \{w_1 \cdots w_k : k \in \mathbb{N}_{>0}, w_1 \in L(R), \dots, w_k \in L(R)\}.$$

reguläre
Ausdrücke über Σ

- Sind R und S reguläre Ausdrücke über Σ , so ist
 - $L((R \cdot S)) := \{wu : w \in L(R), u \in L(S)\}$.
 - $L((R|S)) := L(R) \cup L(S)$.

Notation 7.24. Zur vereinfachten Schreibweise und besseren Lesbarkeit von regulären Ausdrücken vereinbaren wir folgende Konventionen:

- Den „Punkt“ bei der Konkatenation $(R \cdot S)$ darf man weglassen.
- Bei Ketten gleichartiger Operatoren darf man Klammern weglassen: z.B. schreiben wir kurz $(R_1|R_2|R_3|R_4)$ statt $\left(\left(\left(R_1|R_2\right)|R_3\right)|R_4\right)$ und $(R_1R_2R_3R_4)$ statt $\left(\left(\left(R_1R_2\right)R_3\right)R_4\right)$.
- „Präzedenzregeln“: (1): $*$ bindet stärker als \cdot
(2): \cdot bindet stärker als $|$
- Äußere Klammern, die einen regulären Ausdruck umschließen, dürfen weggelassen werden.
- Zur besseren Lesbarkeit dürfen zusätzliche Klammern benutzt werden.

Beispiel 7.25.

(a) $a|bc^*$ ist eine verkürzte Schreibweise für den regulären Ausdruck $(a|(b \cdot c^*))$.

Die von diesem regulären Ausdruck beschriebene Sprache ist

$$L(a|bc^*) = \{a\} \cup \{w \in \{a, b, c\}^* : \text{der erste Buchstabe von } w \text{ ist ein } b \text{ und} \\ \text{alle weiteren Buchstaben von } w \text{ sind } c\text{'s}\}.$$

(b) $L((a|b)^*) = \{a, b\}^*$.

(c) Die Menge aller Worte über $\{a, b, c\}$, in denen abb als Teilwort vorkommt, wird durch den folgenden regulären Ausdruck beschrieben:

$$(a|b|c)^*abb(a|b|c)^*.$$

(d) Die Menge aller Worte über $\{a, b, c\}$, deren letzter oder vorletzter Buchstabe ein a ist, wird durch den folgenden regulären Ausdruck beschrieben:

$$(a|b|c)^*a(\varepsilon|a|b|c).$$

Beispiel 7.26.

(a) Wir wollen einen regulären Ausdruck angeben, der die Sprache all jener Wörter über dem Alphabet $\Sigma = \{0, 1, \dots, 9, /\}$ definiert, die Telefonnummern der Form

$$\textit{Vorwahl}/\textit{Nummer}$$

kodieren, wobei *Vorwahl* und *Nummer* nicht-leere Ziffernfolgen sind, *Vorwahl* mit einer Null beginnt und *Nummer* nicht mit einer Null beginnt. Wörter dieser Sprache sind z.B. 069/7980 und 06131/3923378, aber *nicht* die Wörter 069/798-0, 0697980, 69/7980 und 069/07980.

Der folgende Ausdruck definiert die gewünschte Sprache:

$$0(0|1|\dots|9)^*/(1|\dots|9)(0|1|\dots|9)^*$$

(b) Es sei R der folgende reguläre Ausdruck:

$$(\varepsilon|069/|) 798 (\varepsilon|-) (0 | (1|\dots|9)(0|1|\dots|9)^*)$$

R definiert eine Sprache, die beispielsweise die Wörter 069/798-0 und 7980 enthält, aber nicht das Wort 069/798-028362.

Frage: Welche Arten von Sprachen können durch reguläre Ausdrücke beschrieben werden?

Antwort: Genau dieselben Sprachen, die durch (deterministische oder nichtdeterministische) endliche Automaten akzeptiert werden können. Diese Sprachen werden **reguläre Sprachen** genannt.

Details: In der Veranstaltung “GL-2: Formale Sprachen und Berechenbarkeit”.

7.6 Ausblick

Reguläre Grammatiken:

Abgesehen von DFAs, NFAs und regulären Ausdrücken kann man die regulären Sprachen auch durch bestimmte Grammatiken erzeugen, so genannte **reguläre Grammatiken** — das sind kontextfreie Grammatiken (vgl. Kapitel 8), die von einer besonders einfachen Form sind.

Das Leerheitsproblem:

Eine typische Fragestellung bzgl. DFAs oder NFAs ist das so genannte **Leerheitsproblem**:

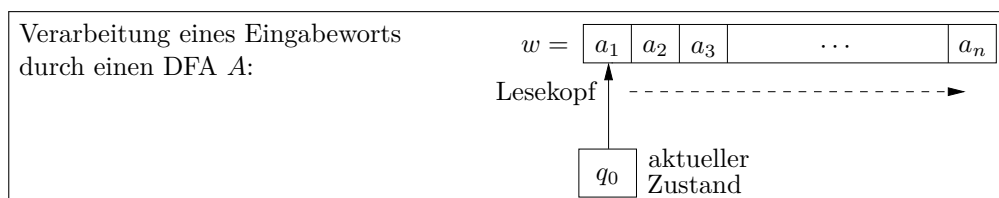
Gegeben sei ein DFA oder NFA A . Wie kann man herausfinden, ob $L(A) \neq \emptyset$ ist, d.h. ob es (mindestens) ein Eingabewort gibt, das von A akzeptiert wird?

(Man erinnere sich z.B. an das Flussüberquerungsproblem aus Beispiel 1.2.)

Durch Betrachtung der graphischen Darstellung von A kann diese Frage leicht beantwortet werden:

Teste, ob es in der graphischen Darstellung von A einen Weg gibt, der vom Startzustand zu einem akzeptierenden Zustand führt.

Eine andere Sichtweise auf DFAs und NFAs:



7.7 Literaturhinweise

[15] Kapitel 7.1

[12] Kapitel 2 und 3

[28] Kapitel 4.1 und 4.2

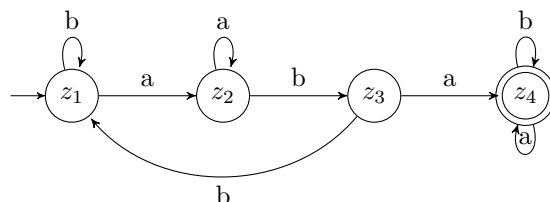
[29] Kapitel 4.1

[26] Kapitel 1.2

7.8 Übungsaufgaben zu Kapitel 7

Aufgabe 7.1.

(a) Sei A der folgende endliche Automat über dem Alphabet $\Sigma = \{a, b\}$:



(i) Geben Sie die Menge der Zustände, den Startzustand, die Menge der akzeptierenden Zustände und die Übergangsfunktion von A an.

(ii) Welche der folgenden Wörter werden von A akzeptiert, welche nicht?

- $bbaabba$
- $abbaaababba$
- $aabbaab$

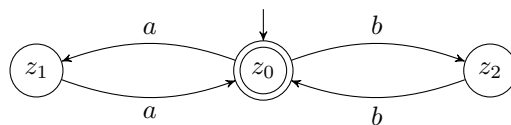
Begründen Sie Ihre Antworten.

(iii) Geben Sie ein möglichst kurzes Wort an, das von A akzeptiert wird.

(iv) Beschreiben Sie umgangssprachlich, welche Sprache $L(A)$ von A akzeptiert wird.

(b) Geben Sie die graphische Darstellung eines nichtdeterministischen endlichen Automaten an, der genau diejenigen Wörter über dem Alphabet $\{a, b\}$ akzeptiert, deren drittletzter Buchstabe ein a ist.

Aufgabe 7.2. Sei A_1 der folgende endliche Automat über dem Alphabet $\Sigma = \{a, b\}$:



(a) Geben Sie folgendes für A_1 an:

- die Menge der Zustände,
- den Startzustand,
- die Menge der Endzustände und
- die Übergangsfunktion.

(b) Stellt die obige Darstellung einen

- deterministischen Automaten dar?
- nichtdeterministischen Automaten dar?

(c) Welche der folgenden Wörter werden von A_1 akzeptiert, welche nicht? Begründen Sie Ihre Antworten.

- $aaabb$
- $baaaaabbbb$
- $aabz_0aa$
- $bbbbaab$

(d) Geben Sie die sieben kürzesten Wörter an, die A_1 akzeptiert.

(e) Beschreiben Sie umgangssprachlich, aus welchen Wörtern die Sprache $L(A_1)$ besteht, die von A_1 akzeptiert wird.

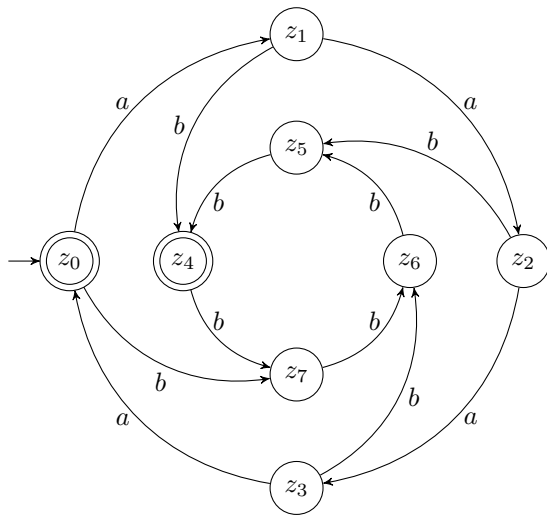
(f) Geben Sie einen DFA A_2 mit möglichst wenigen Zuständen an, der vollständig ist und für den $L(A_2) = L(A_1)$ gilt.

Aufgabe 7.3. Von einem Computervirus ist bekannt, dass in den vom ihm befallenen Dateien mindestens eine der folgenden Bitfolgen auftritt: 101 bzw. 111.

(a) Modellieren Sie potenziell befallene Dateien durch einen regulären Ausdruck. Der Ausdruck soll also die Sprache aller Wörter beschreiben, in denen 101 oder 111 als Teilwort vorkommt.

(b) Geben Sie die graphische Darstellung eines nichtdeterministischen endlichen Automaten an, der potenziell befallene Dateien erkennt. Der Automat soll also genau diejenigen Wörter akzeptieren, in denen 101 oder 111 als Teilwort vorkommt.

Aufgabe 7.4. Sei A_1 der abgebildete endliche Automat über dem Alphabet $\Sigma = \{a, b, c\}$:



- (a) Geben Sie folgendes für A_1 an:
- die Menge der Zustände,
 - den Startzustand,
 - die Menge der Endzustände und
 - die Übergangsfunktion.
- (b) Stellt die nebenstehende Abbildung einen
- deterministischen Automat dar?
 - nichtdeterministischen Automat dar?
- (c) Welche der folgenden Wörter werden von A_1 akzeptiert, welche nicht? Begründen Sie Ihre Antworten.
- $z_0z_1z_4$
 - $aaaacbbb$
 - $aabb$
 - $aaabbbbba$
 - $aaab$
 - $abbbbbbb$

- (3) Geben Sie die fünf kürzesten Wörter an, die A_1 akzeptiert.
- (3) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(A_1)$ an, die vom Automaten A_1 akzeptiert wird.
- (3) Geben Sie einen DFA A_2 mit möglichst wenigen Zuständen an, der vollständig ist und für den $L(A_2) = L(A_1)$ gilt.

Aufgabe 7.5. Der deutsch-amerikanische Komponist Gershon Kingsley hat 1969 das Instrumentalmusikstück *Popcorn* komponiert. Bis heute sind über 500 verschiedene Cover-Versionen davon entstanden, im letzten Jahr erst gab es Interpretationen von *Faith No More* und *Muse*.

Nehmen Sie an, es existiert eine elektronische Sammlung mit Notenschriften von allen jemals produzierten Musikstücken, die Kingsley nach Cover-Versionen seines Stücks durchsuchen möchte. Eine solche Cover-Version ist für Kingsley jedes Stück, das wie *Popcorn* in der Tonart h-Moll geschrieben ist bzw. nur die Noten $h, c^\sharp, d, e, f^\sharp, g$ oder a benutzt. Außerdem muss natürlich das berühmte *Popcorn*-Thema $hahf^\sharp df^\sharp h$ als Teilstück enthalten sein.

Geben Sie einen nichtdeterministischen endlichen Automaten A in graphischer Darstellung an, der eine Notenschrift in der Sammlung genau dann akzeptiert, wenn es sich für Kingsley um eine Cover-Versionen seines Stücks *Popcorn* handelt.

Aufgabe 7.6. ⁴ Jede natürliche Zahl n lässt sich als *Dualzahl*, d.h., in der Form $[n]_2 = z_l z_{l-1} \dots z_0$ darstellen, so dass $z_i \in \{0, 1\}$ für $0 \leq i \leq l$ mit $l \in \mathbb{N}$ ist und $n = \sum_{i=0}^l z_i \cdot 2^i$ gilt. Die Zahl $[n]_2$ wird als die *Dualdarstellung* der Zahl n bezeichnet. Dualzahlen können auf herkömmliche Weise schriftlich addiert werden, wobei der Übertrag bei der Zwei erfolgt.

⁴Diese Übungsaufgabe entstand in Anlehnung an eine Übungsaufgabe von Jacobo Toran.

Gegeben sei das folgende Eingabealphabet

$$\Sigma := \left\{ \begin{array}{|c|} \hline 0 \\ \hline 0 \\ \hline 0 \\ \hline \end{array}, \begin{array}{|c|} \hline 0 \\ \hline 0 \\ \hline 1 \\ \hline \end{array}, \begin{array}{|c|} \hline 0 \\ \hline 1 \\ \hline 0 \\ \hline \end{array}, \begin{array}{|c|} \hline 0 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}, \begin{array}{|c|} \hline 1 \\ \hline 0 \\ \hline 0 \\ \hline \end{array}, \begin{array}{|c|} \hline 1 \\ \hline 0 \\ \hline 1 \\ \hline \end{array}, \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 0 \\ \hline \end{array}, \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array} \right\}.$$

Geben Sie einen DFA A an, der ein Wort w aus Σ^* genau dann akzeptiert, wenn w eine korrekte Addition zweier Dualzahlen $[n]_2$ und $[m]_2$ mit $n, m \in \mathbb{N}$ darstellt. So ist beispielsweise $w \in L(A)$ für

$$w = \begin{array}{|c|c|c|c|} \hline 0 & 1 & 0 & 1 \\ \hline 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 0 \\ \hline \end{array}, \text{ weil } \begin{array}{r} 0101 = [5]_2 \\ + 0111 = [7]_2 \\ \hline 1100 = [12]_2 \end{array}.$$

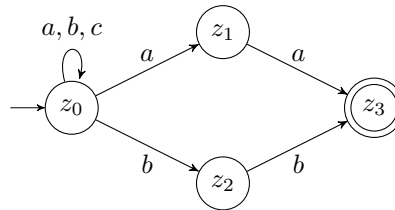
Hinweis: Beachten Sie, dass ein endlicher Automat jedes Eingabewort von links nach rechts liest. Begründen Sie kurz, warum der von Ihnen angegebene DFA die verlangte Sprache akzeptiert.

Aufgabe 7.7.

(a) Betrachten Sie das Eingabealphabet $\Sigma := \{a, b, c\}$ und die Sprache

$$L := \{w \in \Sigma^* : w \text{ endet mit dem Teilwort } aa \text{ oder } bb\}.$$

Die Sprache L wird vom NFA A akzeptiert, der durch die folgende graphische Darstellung gegeben ist:



Geben Sie einen DFA A' in graphischer Darstellung an, der die Sprache L akzeptiert. Wandeln sie dazu den NFA A mit Hilfe der Potenzmengenkonstruktion in den DFA A' um. Berücksichtigen Sie dabei nur solche Zustände von A' , die vom Startzustand $q'_0 := \{q_0\}$ aus erreicht werden können.

(b) Im Beweis von Satz 7.13 wird aus einem NFA $A = (\Sigma, Q, \delta, q_0, F)$ ein DFA $A' = (\Sigma, Q', \delta', q'_0, F')$ konstruiert mit $L(A) = L(A')$. Beweisen Sie per Induktion nach n , dass für alle $n \in \mathbb{N}$ und jedes $w \in \Sigma^*$ mit $|w| = n$ gilt:

$$\widehat{\delta}'(q'_0, w) = \widehat{\delta}(q_0, w).$$

Aufgabe 7.8. Mit Hilfe von endlichen Automaten soll die Teilbarkeit der natürlichen Zahlen in Dezimaldarstellung untersucht werden. Betrachten Sie das Eingabealphabet $\Sigma := \{0, 1, 2, 3, \dots, 9\}$. Wir identifizieren jedes Wort $w = w_0w_1w_2 \dots \in \Sigma^*$ mit der natürlichen Zahl

$$w_{\text{nat}} := \begin{cases} 0 & , \text{ falls } w = \varepsilon \\ \sum_{i=0}^{|w|-1} w_i \cdot 10^{|w|-(i+1)} & , \text{ sonst} \end{cases}$$

Aufgabe 7.11. Betrachten Sie das Eingabealphabet $\Sigma := \{a, b\}$ und die Sprache

$$L_2 := \{ww : w \in \Sigma^* \text{ und } |w| = 2\}.$$

Beweisen Sie, dass jeder NFA, der L_2 akzeptiert mindestens vier verschiedene Zustände besitzt.

Aufgabe 7.12. Beweisen Sie die Korrektheit der folgenden Aussagen:

- (a) Sei Σ ein beliebiges Alphabet und L_1 eine beliebige reguläre Sprache über Σ . Dann ist auch die Sprache $\overline{L_1} := \Sigma^* \setminus L_1$ regulär.
- (b) Es gibt eine reguläre Sprache L_2 , so dass jeder DFA A mit $L(A) = L_2$ mehr als einen Endzustand hat.

Aufgabe 7.13.

1. Geben Sie einen regulären Ausdruck an, der die Sprache der Wörter über dem Alphabet $\Sigma = \{0, 1, \dots, 9\}$ definiert, die natürliche Zahlen ohne führende Nullen kodieren. Wörter aus der Sprache sind z.B. 42, 0, 1, aber nicht 0042 oder das leere Wort.
2. Sei R der folgende reguläre Ausdruck:

$$\left(0 \mid ((1 \mid \dots \mid 9)(0 \mid 1 \mid \dots \mid 9)^*)\right) \left(\varepsilon \mid ((0 \mid 1 \mid \dots \mid 9)(0 \mid 1 \mid \dots \mid 9))\right) \in$$

- (i) Welche der folgenden Wörter liegen in der von R definierten Sprache $L(R)$, welche nicht?

1,99€	,69€	1,9€
01,99€	1€	1,09

Geben Sie jeweils eine kurze Begründung für Ihre Antwort.

- (ii) Beschreiben Sie umgangssprachlich, welche Sprache $L(R)$ von R definiert wird.

Aufgabe 7.14. Gegeben seien die folgenden regulären Ausdrücke

$$R_1 = aa^*|c \qquad R_2 = (b|aa)^*c \qquad R_3 = (a^*(\varepsilon|b)c^*)^*$$

- (a) Gehören die folgenden Worte zur Sprache $L(R_1)$, $L(R_2)$ bzw. $L(R_3)$?

$$w_1 = a \qquad w_2 = aac \qquad w_3 = baaac \qquad w_4 = cbba$$

- (b) Geben Sie das kürzeste Wort w an, so dass $w \in L(R_1)$ und $w \notin L(R_2)$.
- (c) Welche Sprachen beschreiben die regulären Ausdrücke R_1 , R_2 und R_3 ?
- (d) Geben Sie einen DFA mit möglichst wenigen Zuständen in graphischer Darstellung an, der die Sprache $L(R_2)$ akzeptiert.

Aufgabe 7.15. Gegeben seien die folgenden regulären Ausdrücke

$$R_1 = ((0|1)(0|1))^*$$

$$R_2 = (\varepsilon|00|10|11)^*$$

$$R_3 = (0|1)((0|1)(0|1))^*$$

(a) Gehören die folgenden Worte zur Sprache $L(R_1)$, $L(R_2)$ bzw. $L(R_3)$?

$$w_1 = \varepsilon$$

$$w_2 = 110$$

$$w_3 = 1111$$

$$w_4 = 101010$$

(b) Geben Sie das kürzeste nicht-leere Wort w an, so dass $w \in L(R_1)$ und $w \notin L(R_2)$.

(c) Beschreiben Sie die Sprachen $L(R_2) \setminus L(R_1)$ und $L(R_1) \cup L(R_3)$ jeweils umgangssprachlich.

(d) Geben Sie einen DFA mit möglichst wenigen Zuständen in graphischer Darstellung an, der die Sprache $L(R_2)$ akzeptiert.

Aufgabe 7.16. Geben Sie für die folgenden Sprachen je einen möglichst kurzen regulären Ausdruck an, der die Sprache beschreibt.

(a) $L_1 := \{w \in \{a, b\}^* : w \text{ beginnt mit } b \text{ und hat höchstens die Länge fünf}\}$

(b) $L_2 := \{w \in \{a, b\}^* : w \text{ besteht nur } a\text{'s oder nur aus } b\text{'s, zusätzlich ist } |w| \text{ gerade}\}$

(c) $L_3 := \{w \in \{a, b\}^* : \text{in } w \text{ kommen keine drei } a\text{'s hintereinander vor}\}$

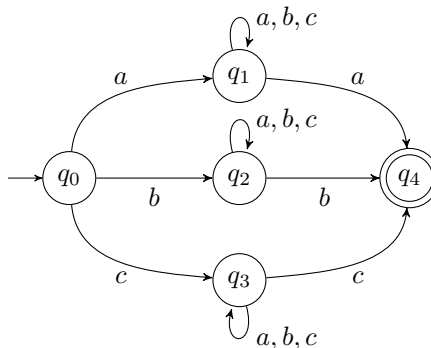
(d) $L_4 := \{w = w_0w_1w_2 \dots \in \{a, b\}^* : w_i = b, \text{ f.a. } i = 2n, n \in \mathbb{N}\}$

(e) $L_5 := \{w = w_0w_1w_2 \dots \in \{a, b\}^* : \text{Falls } w_i = b \text{ so } w_{i+1} = a \text{ oder } w_{i+2} = a, i \in \mathbb{N}\}$

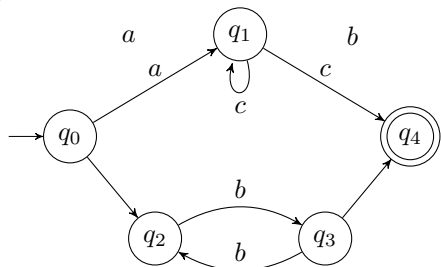
Aufgabe 7.17.

Betrachten Sie den abgebildeten NFA A über dem Eingabealphabet $\Sigma = \{a, b, c\}$.

Geben Sie einen DFA A' in graphischer Darstellung an, mit $L(A') = L(A)$. Wandeln sie dazu den NFA A mit Hilfe der Potenzmengenkonstruktion in einen DFA A' um. Berücksichtigen Sie dabei nur solche Zustände von A' , die vom Startzustand $q'_0 := \{q_0\}$ aus erreicht werden können.



Aufgabe 7.18.



Betrachten Sie den abgebildeten NFA A über dem Eingabealphabet $\Sigma = \{a, b, c\}$.

Geben Sie einen DFA A' in graphischer Darstellung an, mit $L(A') = L(A)$. Wandeln sie dazu den NFA A mit Hilfe der Potenzmengenkonstruktion in einen DFA A' um. Berücksichtigen Sie dabei nur solche Zustände von A' , die vom Startzustand $q'_0 := \{q_0\}$ aus erreicht werden können.

Aufgabe 7.19. Welche der Sprachen L_1 , L_2 und L_3 sind regulär, welche nicht? Beweisen Sie Ihre Antwort jeweils mit Hilfe des Pumping-Lemmas oder durch die Angabe eines endlichen Automaten.

$$(a) L_1 = \{b^n a^{(2^n)} : n \in \mathbb{N}\} \quad (b) L_2 = \{a^{(2^n)} : n \in \mathbb{N}\} \quad (c) L_3 = \{a^{(2^n)} : n \in \mathbb{N}\}$$

Aufgabe 7.20. Welche der Sprachen L_1 , L_2 und L_3 sind regulär, welche nicht? Beweisen Sie Ihre Antwort jeweils mit Hilfe des Pumping-Lemmas oder durch die Angabe eines endlichen Automaten.

$$(a) L_1 := \{a^2 a^2\} \quad (b) L_2 := \{a^n a^n : n \in \mathbb{N}\} \quad (c) L_3 := \{a^n b a^n : n \in \mathbb{N}\}$$

Aufgabe 7.21. Welche der folgenden Sprachen sind regulär, welche nicht? Beweisen Sie die Korrektheit Ihrer Antwort jeweils mit Hilfe des Pumping-Lemmas oder durch die Angabe eines endlichen Automaten.

$$(a) L_1 := \{a^i a^i : i \in \mathbb{N}\} \quad (c) L_3 := \{a^i b a^i : i \in \mathbb{N}\}$$

$$(b) L_2 := \{a^i a a^i : i \in \mathbb{N}\} \quad (d) L_4 := \{a a^{i \cdot i} : i \in \mathbb{N}\}$$

Aufgabe 7.22. Welche der folgenden Sprachen L_1 , L_2 , L_3 und L_4 über dem Alphabet $\Sigma = \{a\}$ sind regulär, welche nicht? Beweisen Sie Ihre Antwort jeweils mit Hilfe des Pumping-Lemmas oder durch die Angabe eines endlichen Automaten bzw. eines regulären Ausdrucks.

$$(a) L_1 := \{a^{n+n} : n \in \mathbb{N}_{>0}\} \quad (c) L_3 := \{a^{n \cdot n} : n \in \mathbb{N}_{>0}\}$$

$$(b) L_2 := \{a^{n-n} : n \in \mathbb{N}_{>0}\} \quad (d) L_4 := \{a^{n/n} : n \in \mathbb{N}_{>0}\}$$

8 Kontextfreie Grammatiken zur Modellierung von Strukturen

In Kapitel 4 haben wir bereits **Graphen** und **Bäume** als Möglichkeiten kennengelernt, mit denen sich Objekte sowie Beziehungen zwischen je 2 Objekten gut modellieren lassen. In Kapitel 6.2 wurden Verallgemeinerungen davon eingeführt, die so genannten σ -Strukturen, wobei σ eine Signatur ist. Abgesehen von Graphen und Bäumen kann man damit beispielsweise auch die natürlichen (oder die rationalen) Zahlen mit arithmetischen Operationen $+$, \times etc. modellieren oder – wie in Kapitel 6.6 gesehen – auch relationale Datenbanken, die z.B. Informationen über Kinofilme und das aktuelle Kinoprogramm enthalten. In Kapitel 8 werden wir nun einen weiteren Kalkül kennenlernen, mit dem man strukturelle Eigenschaften von Systemen beschreiben kann: **kontextfreie Grammatiken**.

Kontextfreie Grammatiken (kurz: KFGs) eignen sich besonders gut zur Modellierung von beliebig tief geschachtelten baumartigen Strukturen. KFGs können gleichzeitig

- hierarchische Baumstrukturen und
- Sprachen und deren textuelle Notation

spezifizieren. KFGs werden z.B. angewendet zur Definition von:

- Programmen einer Programmiersprache und deren Struktur, z.B. Java, C, Pascal (KFGs spielen z.B. beim „Compilerbau“ eine wichtige Rolle).
- Datenaustauschformaten, d.h. Sprachen als Schnittstelle zwischen Software-Werkzeugen, z.B. HTML, XML.
- Bäumen zur Repräsentation strukturierter Daten, z.B. XML.
- Strukturen von Protokollen beim Austausch von Nachrichten zwischen Prozessen oder Geräten.

KFGs sind ein grundlegender Kalkül, der für die formale Definition von Sprachen eingesetzt wird.

In dieser Veranstaltung werden nur die Grundbegriffe und einige Beispiele vorgestellt. Im Detail werden KFGs in der Veranstaltung „GL-2: Formale Sprachen und Berechenbarkeit“ behandelt.

8.1 Definition des Begriffs „Kontextfreie Grammatik“

Es gibt 2 Sichtweisen auf KFGs:

- (1) Eine KFG ist ein spezielles **Ersetzungssystem**. Seine Regeln geben an, auf welche Art man ein Symbol durch eine Folge von Symbolen ersetzen kann. Auf diese Weise definiert eine KFG eine **Sprache**, d.h. eine **Menge von Worten** über einem bestimmten Alphabet, die mit dem durch die KFG gegebenen Regeln erzeugt werden können.

- (2) Gleichzeitig definiert eine KFG eine **Menge von Baumstrukturen**, die sich durch schrittweises Anwenden der Regeln erzeugen lassen.

Für die Modellierung von Strukturen ist die zweite Sichtweise besonderes interessant. Aber es ist oft sehr nützlich, dass derselbe Kalkül auch gleichzeitig eine textuelle Notation für die Baumstrukturen liefern kann und dass Eigenschaften der zugehörigen Sprache untersucht werden können.

Definition 8.1 (KFG). Eine kontextfreie Grammatik $G = (\Sigma, V, S, P)$ besteht aus

- Terminalsymbole
Terminale • einer endlichen Menge Σ , der so genannten Menge der **Terminalsymbole** (die Elemente aus Σ werden auch **Terminale** genannt).
- Nichtterminalsymbole
Variablen
Nichtterminale • einer endlichen Menge V , der so genannten Menge der **Nichtterminalsymbole** (oder **Variablen**) (die Elemente aus V werden auch **Nichtterminale** genannt).
Die Mengen Σ und V sind disjunkt, d.h. $\Sigma \cap V = \emptyset$.
- Vokabular
Symbole Die Menge $W := \Sigma \cup V$ heißt **Vokabular** (die Elemente in W nennt man auch **Symbole**).
- Startsymbol • einem Symbol $S \in V$, dem so genannten **Startsymbol**.
- Produktionen • einer endlichen Menge $P \subseteq V \times W^*$, der so genannten Menge der **Produktionen**.
Für eine Produktion $(A, x) \in P$ schreiben wir meistens $A \rightarrow x$.

In der Literatur, in den Übungsaufgaben in Abschnitt 8.6 und in den Beispielklausuren in Kapitel 10 werden an Stelle der Buchstaben Σ und V oft auch die Buchstaben T und N verwendet, um die Menge der Terminalsymbole bzw. der Nichtterminalsymbole zu bezeichnen.

Beispiel 8.2. Als erstes Beispiel betrachten wir eine KFG, die arithmetische Ausdrücke erzeugt, die über den Zahlen 1, 2, 3 gebildet sind und die Operatoren +, −, · sowie Klammern (,) benutzt. Ein Beispiel für einen zulässigen solchen arithmetischen Ausdruck ist $(1 + 3) \cdot (2 + 2 + 3) - 1$. Wir betrachten die KFG $G_{AA} := (\Sigma, V, S, P)$ mit

- Terminalalphabet $\Sigma := \{1, 2, 3, +, -, \cdot, (,)\}$
- Nichtterminalalphabet $V := \{\text{Ausdruck}, \text{Operator}\}$
- Startsymbol $S := \text{Ausdruck}$
- Produktionsmenge $P := \left\{ \begin{array}{l} \text{Ausdruck} \rightarrow 1, \\ \text{Ausdruck} \rightarrow 2, \\ \text{Ausdruck} \rightarrow 3, \\ \text{Ausdruck} \rightarrow \text{Ausdruck Operator Ausdruck}, \\ \text{Ausdruck} \rightarrow (\text{Ausdruck}), \\ \text{Operator} \rightarrow +, \\ \text{Operator} \rightarrow -, \\ \text{Operator} \rightarrow \cdot \end{array} \right\}$.

Um Schreibarbeit zu sparen, werden wir bei der Angabe der Produktionsmenge einer KFG oft einige Zeilen, die das gleiche Nichtterminal auf der linken Seite des Pfeils aufweisen, zu einer einzigen Zeile zusammenfassen, bei der die Möglichkeiten der rechten Seite des Pfeils durch das

„oder“-Zeichen | getrennt sind.

Damit können wir die Produktionsmenge P auch kurz wie folgt beschreiben:

$$P = \left\{ \begin{array}{l} \text{Ausdruck} \rightarrow 1 \mid 2 \mid 3, \\ \text{Ausdruck} \rightarrow \text{Ausdruck Operator Ausdruck} \mid (\text{Ausdruck}), \\ \text{Operator} \rightarrow + \mid - \mid \cdot. \end{array} \right\}.$$

8.2 Bedeutung der Produktionen: Semantik von KFGs

Jede Produktion einer KFG, etwa die Produktion $\text{Ausdruck} \rightarrow \text{Ausdruck Operator Ausdruck}$, kann man auffassen als:

- eine *Strukturregel*, die besagt „Ein *Ausdruck* besteht aus einem *Ausdruck*, gefolgt von einem *Operator*, gefolgt von einem *Ausdruck* — oder als
- eine *Ersetzungsregel*, die besagt „Das Symbol *Ausdruck* kann man durch das Wort *Ausdruck Operator Ausdruck* ersetzen.“

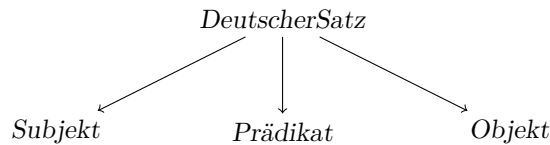
Beispielsweise kann man die Produktion

$$\text{DeutscherSatz} \rightarrow \text{Subjekt Prädikat Objekt}$$

verstehen als Aussage, die besagt:

„Ein Deutscher Satz ist aufgebaut aus Subjekt Prädikat Objekt.“

Graphische Darstellung:



Das Grundkonzept für die Anwendung von Produktionen einer KFG ist die „Ableitung“:

Definition 8.3 (Ableitung). Sei $G = (\Sigma, V, S, P)$ eine KFG.

- Falls $A \rightarrow x$ eine Produktion in P ist und $u \in W^*$ und $v \in W^*$ beliebige Worte über dem Vokabular $W = \Sigma \cup V$ sind, so schreiben wir

$$uAv \Longrightarrow_G u xv \quad (\text{bzw. kurz: } uAv \Longrightarrow u xv)$$

und sagen, dass uAv in einem **Ableitungsschritt** zu $u xv$ umgeformt werden kann.

Ableitungsschritt

- Eine **Ableitung** ist eine endliche Folge von hintereinander angewendeten Ableitungsschritten. Für Worte $w \in W^*$ und $w' \in W^*$ schreiben wir

Ableitung

$$w \Longrightarrow_G^* w' \quad (\text{bzw. kurz: } w \Longrightarrow^* w'),$$

um auszusagen, dass es eine endliche Folge von Ableitungsschritten gibt, die w zu w' umformt.

Spezialfall:

Diese Folge darf auch aus 0 Ableitungsschritten bestehen, d.h. f.a. $w \in W^*$ gilt: $w \Longrightarrow^* w$.

Beispiel 8.4. Sei G_{AA} die Grammatik für arithmetische Ausdrücke aus Beispiel 8.2. Beispiele für einzelne Ableitungsschritte:

- $(\text{Ausdruck}) \implies (\text{Ausdruck Operator Ausdruck})$
- $(\text{Ausdruck Operator Ausdruck}) \implies (\text{Ausdruck} + \text{Ausdruck})$

Ein Beispiel für eine Ableitung in G_{AA} :

$$\begin{aligned}
 \text{Ausdruck} &\implies \text{Ausdruck Operator Ausdruck} \\
 &\implies (\text{Ausdruck}) \text{ Operator Ausdruck} \\
 &\implies (\text{Ausdruck Operator Ausdruck}) \text{ Operator Ausdruck} \\
 &\implies (\text{Ausdruck} + \text{Ausdruck}) \text{ Operator Ausdruck} \\
 &\implies (\text{Ausdruck} + \text{Ausdruck}) \cdot \text{Ausdruck} \\
 &\implies (1 + \text{Ausdruck}) \cdot \text{Ausdruck} \\
 &\implies (1 + 3) \cdot \text{Ausdruck} \\
 &\implies (1 + 3) \cdot 2 ,
 \end{aligned}$$

In jedem Schritt wird jeweils eine Produktion auf ein Nichtterminal der vorangehenden Symbolfolge angewandt. Obige Kette von Ableitungsschritten zeigt, dass

$$\text{Ausdruck} \implies^* (1 + 3) \cdot 2 .$$

Definition 8.5 (Die Sprache $L(G)$ einer KFG G).

Sprache einer KFG, $L(G)$

Sei $G = (\Sigma, V, S, P)$ eine KFG. Die **von G erzeugte Sprache $L(G)$** ist die Menge aller Worte über dem Terminalalphabet Σ , die aus dem Startsymbol S abgeleitet werden können. D.h.:

$$L(G) := \{w \in \Sigma^* : S \implies_G^* w\}.$$

Man beachte, dass $L(G) \subseteq \Sigma^*$ ist. Daher kommen in Worten aus $L(G)$ keine Nichtterminale vor!

Beispiel 8.6. Die KFG G_{AA} aus Beispiel 8.2 definiert die Sprache $L(G_{AA})$, die aus allen über den Zahlen 1, 2, 3, den Operatoren +, -, · und den Klammersymbolen (,) korrekt geformten arithmetischen Ausdrücken besteht. Beispielsweise gehören folgende Worte zu $L(G_{AA})$:

$$3, \quad (3 + 1), \quad 1 + 2 \cdot 3, \quad (3 + 1) \cdot (2 + 2 + 3) - 1, \quad 2 \cdot ((3 + 1) \cdot (2 + 2 + 3) - 1), \quad ((3 + 1)).$$

Aber die Worte

$$(), \quad 4, \quad (3 + 1, \quad (\text{Ausdruck Operator Ausdruck})$$

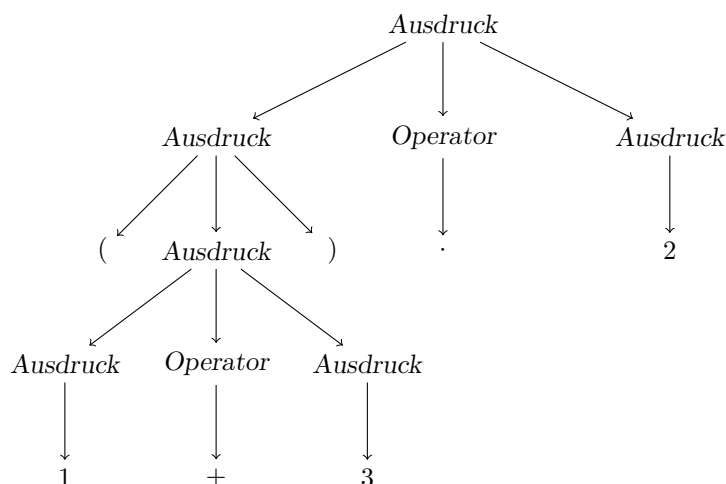
gehören **nicht** zu $L(G_{AA})$.

Beispiel 8.7. Sei G_{AA} die Grammatik für arithmetische Ausdrücke aus Beispiel 8.2.

Die Ableitung

$$\begin{aligned}
 \text{Ausdruck} &\Longrightarrow \text{Ausdruck Operator Ausdruck} \\
 &\Longrightarrow (\text{Ausdruck}) \text{ Operator Ausdruck} \\
 &\Longrightarrow (\text{Ausdruck Operator Ausdruck}) \text{ Operator Ausdruck} \\
 &\Longrightarrow (\text{Ausdruck} + \text{Ausdruck}) \text{ Operator Ausdruck} \\
 &\Longrightarrow (\text{Ausdruck} + \text{Ausdruck}) \cdot \text{Ausdruck} \\
 &\Longrightarrow (1 + \text{Ausdruck}) \cdot \text{Ausdruck} \\
 &\Longrightarrow (1 + 3) \cdot \text{Ausdruck} \\
 &\Longrightarrow (1 + 3) \cdot 2 ,
 \end{aligned}$$

wird durch den folgenden **Ableitungsbaum** dargestellt:



Ableitungsbäume:

Ableitungsbaum

Sei $G = (\Sigma, V, S, P)$ eine KFG. Jede Ableitung $S \Longrightarrow_G^* w$ lässt sich als gerichteter Baum darstellen, bei dem

- jeder Knoten mit einem Symbol aus $\Sigma \cup V \cup \{\varepsilon\}$ markiert ist und
- die Kinder jedes Knotens eine festgelegte Reihenfolge haben. In der Zeichnung eines Ableitungsbaums werden von links nach rechts zunächst das „erste Kind“ dargestellt, dann das zweite, dritte etc.

Die Wurzel des Baums ist mit dem Startsymbol S markiert. Jeder Knoten mit seinen Kindern repräsentiert die Anwendung einer Produktion aus P :

- Die Anwendung einer Produktion der Form $A \rightarrow x$ mit $A \in V$ und $x \in W^+$ wird im Ableitungsbaum repräsentiert durch einen Knoten, der mit dem Symbol A markiert ist und der $|x|$ viele Kinder hat, so dass das i -te Kind mit dem i -ten Symbol von x markiert ist (f.a. $i \in \{1, \dots, |x|\}$).
- Die Anwendung einer Produktion der Form $A \rightarrow \varepsilon$ mit $A \in V$ wird im Ableitungsbaum repräsentiert durch einen Knoten, der mit dem Symbol A markiert ist und der genau ein Kind hat, das mit ε markiert ist.

Beachte: Ein Ableitungsbaum kann mehrere Ableitungen repräsentieren. Beispielsweise repräsentiert der Ableitungsbaum aus Beispiel 8.7 auch die Ableitung

$$\begin{aligned}
 \text{Ausdruck} &\implies \text{Ausdruck Operator Ausdruck} \\
 &\implies \text{Ausdruck Operator } 2 \\
 &\implies \text{Ausdruck} \cdot 2 \\
 &\implies (\text{Ausdruck}) \cdot 2 \\
 &\implies (\text{Ausdruck Operator Ausdruck}) \cdot 2 \\
 &\implies (\text{Ausdruck Operator } 3) \cdot 2 \\
 &\implies (\text{Ausdruck} + 3) \cdot 2 \\
 &\implies (1 + 3) \cdot 2
 \end{aligned}$$

in der gegenüber der ursprünglichen Ableitung aus Beispiel 8.7 einige Ableitungsschritte vertauscht sind. Im Ableitungsbaum wird von der konkreten Reihenfolge, in der die einzelnen Ableitungsschritte vorkommen, abstrahiert.

8.3 Beispiele

Im Folgenden betrachten wir einige weitere Beispiele für kontextfreie Grammatiken.

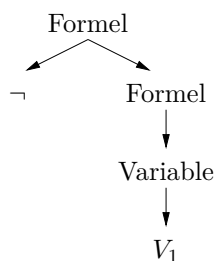
Beispiel 8.8 (Aussagenlogik). Wir konstruieren eine KFG

$$G_{AL} = (\Sigma, V, S, P),$$

deren Sprache $L(G_{AL})$ gerade die Menge aller aussagenlogischen Formeln ist, in denen nur Variablen aus $\{V_0, V_1, V_2\}$ vorkommen:

- Terminalsymbole $\Sigma := \{ V_0, V_1, V_2, \mathbf{0}, \mathbf{1}, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,) \}$
- Nichtterminalsymbole $V := \{ \text{Formel}, \text{Variable}, \text{Junktork} \}$
- Startsymbol $S := \text{Formel}$
- Produktionsmenge $P := \left\{ \begin{array}{l} \text{Formel} \rightarrow \mathbf{0} \mid \mathbf{1} \mid \text{Variable}, \\ \text{Formel} \rightarrow \neg \text{Formel} \mid (\text{Formel} \text{Junktork} \text{Formel}), \\ \text{Variable} \rightarrow V_0 \mid V_1 \mid V_2, \\ \text{Junktork} \rightarrow \wedge \mid \vee \mid \rightarrow \mid \leftrightarrow \end{array} \right\}$.

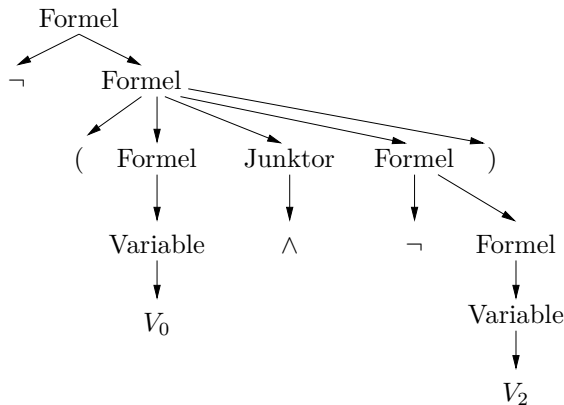
Beispiele für Ableitungsäume:



Dieser Ableitungsbaum repräsentiert die Ableitung

$$\begin{aligned}
 \text{Formel} &\implies \neg \text{Formel} \\
 &\implies \neg \text{Variable} \\
 &\implies \neg V_1
 \end{aligned}$$

Das durch diese(n) Ableitung(sbaum) erzeugte Wort in der Sprache $L(G_{AL})$ ist die Formel $\neg V_1$.



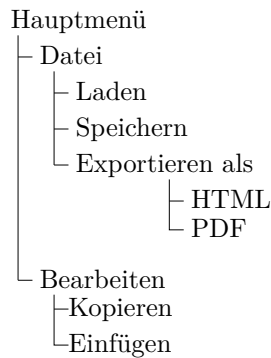
Dieser Ableitungsbaum repräsentiert die Ableitung

$$\begin{aligned}
 \text{Formel} &\Rightarrow \neg \text{Formel} \\
 &\Rightarrow \neg (\text{Formel Junktor Formel}) \\
 &\Rightarrow \neg (\text{Variable Junktor Formel}) \\
 &\Rightarrow \neg (V_0 \text{ Junktor Formel}) \\
 &\Rightarrow \neg (V_0 \wedge \text{Formel}) \\
 &\Rightarrow \neg (V_0 \wedge \neg \text{Formel}) \\
 &\Rightarrow \neg (V_0 \wedge \neg \text{Variable}) \\
 &\Rightarrow \neg (V_0 \wedge \neg V_2).
 \end{aligned}$$

Das durch diese(n) Ableitung(sbaum) erzeugte Wort in der Sprache $L(G_{AL})$ ist die Formel $\neg(V_0 \wedge \neg V_2)$.

Beispiel 8.9 (Menü-Struktur). In der graphischen Benutzeroberfläche von vielen Software-Systemen werden oftmals „Menüs“ verwendet. Ein Menü besteht aus einem Menünamen und einer Folge von Einträgen. Jeder einzelne Eintrag besteht dabei aus einem Operationsnamen oder selbst wieder einem Menü.

Beispiel:



Zur Spezifizierung der Grundstruktur solcher Menüs kann man folgende Grammatik $G_{\text{Menü}}$ verwenden:

$$G_{\text{Menü}} = (\Sigma, V, S, P)$$

mit

- $\Sigma := \{ \text{Menüname, Operationsname} \}$
- $V := \{ \text{Menü, Eintragsfolge, Eintrag} \}$
- $S := \text{Menü}$
- $P := \left\{ \begin{array}{l} \text{Menü} \rightarrow \text{Menüname Eintragsfolge}, \\ \text{Eintragsfolge} \rightarrow \text{Eintrag} \mid \text{Eintrag Eintragsfolge}, \\ \text{Eintrag} \rightarrow \text{Operationsname} \mid \text{Menü} \end{array} \right\}$.

Tag	Zeit	Raum
Mi	8:00-11:00	Magnus-Hörsaal
Do	14:00-16:00	Seminarraum 11

durch HTML-Quelltext der folgenden Form erzeugt:

```

<table>
  <tr>
    <td> Tag </td>
    <td> Zeit </td>
    <td> Raum </td>
  </tr>
  <tr>
    <td> Mi </td>
    <td> 8:00-11:00 </td>
    <td> Magnus-Hörsaal </td>
  </tr>
  <tr>
    <td> Do </td>
    <td> 14:00-16:00 </td>
    <td> Seminarraum 11 </td>
  </tr>
</table>

```

erzeugt.

Das Symbol `<table>` steht hier für den Anfang einer Tabelle, `</table>` steht für das Ende einer Tabelle. Die Symbole `<tr>` und `</tr>` stehen für den Anfang bzw. das Ende einer Zeile der Tabelle. Die Symbole `<td>` und `</td>` stehen für den Anfang bzw. das Ende eines Eintrags in einer Zelle der Tabelle. Als Einträge in einzelnen Zellen kann z.B. Text stehen oder eine weitere Tabelle.

Im Folgenden konstruieren wir eine Grammatik

$$G_{\text{HTML-Tabellen}} = (\Sigma, V, S, P),$$

so dass die von $G_{\text{HTML-Tabellen}}$ erzeugte Sprache aus (möglicherweise geschachtelten) HTML-Tabellen besteht:

- $\Sigma := \{ \langle \text{table} \rangle, \langle / \text{table} \rangle, \langle \text{tr} \rangle, \langle / \text{tr} \rangle, \langle \text{td} \rangle, \langle / \text{td} \rangle, a, \dots, z, A, \dots, Z, 0, 1, \dots, 9, :, -, _ , \grave{a}, \ddot{o}, \ddot{u}, \beta, \text{Ä}, \text{Ö}, \text{Ü} \}$
- $V := \{ \text{Tabelle}, \text{Zeile}, \text{Eintrag}, \text{Text}, \text{Zeilen}, \text{Einträge} \}$
- $S := \text{Tabelle}$
- $P := \left\{ \begin{array}{l} \text{Tabelle} \rightarrow \langle \text{table} \rangle \text{Zeilen} \langle / \text{table} \rangle , \\ \text{Zeilen} \rightarrow \text{Zeile} \mid \text{Zeile Zeilen} , \\ \text{Zeile} \rightarrow \langle \text{tr} \rangle \text{Einträge} \langle / \text{tr} \rangle , \\ \text{Einträge} \rightarrow \text{Eintrag} \mid \text{Eintrag Einträge} , \\ \text{Eintrag} \rightarrow \langle \text{td} \rangle \text{Text} \langle / \text{td} \rangle \mid \langle \text{td} \rangle \text{Tabelle} \langle / \text{td} \rangle , \\ \text{Text} \rightarrow a \mid b \mid \dots \mid z \mid A \mid B \mid \dots \mid Z \mid \dots \mid \text{Ü} , \\ \text{Text} \rightarrow a \text{Text} \mid b \text{Text} \mid \dots \mid z \text{Text} \mid A \text{Text} \mid B \text{Text} \mid \dots \mid Z \text{Text} \mid \dots \mid \text{Ü Text} \end{array} \right\}.$

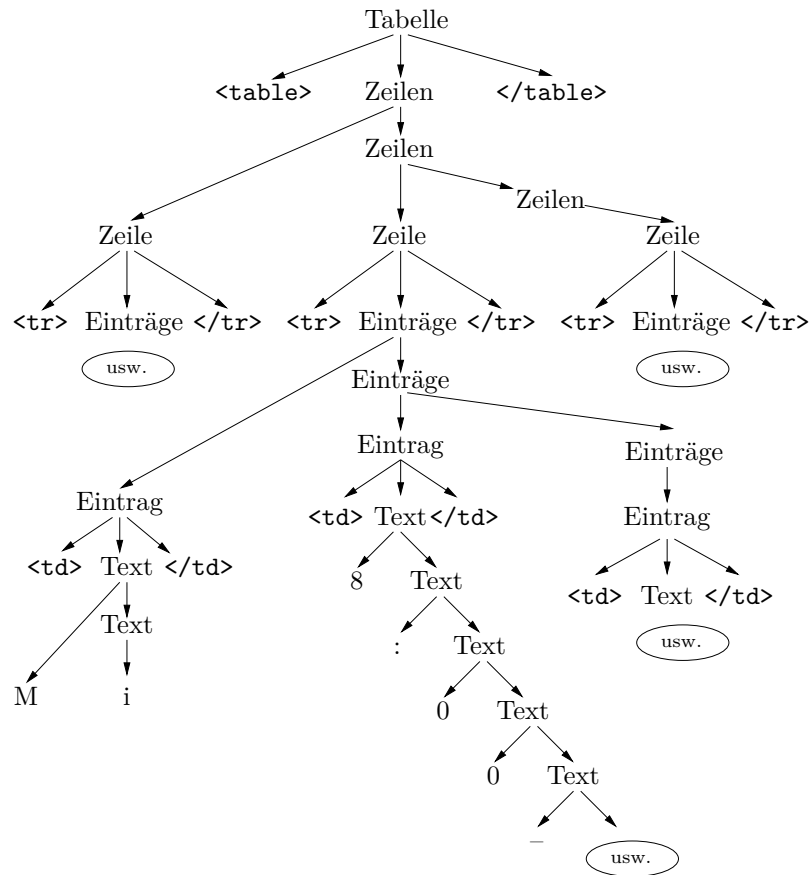


Abbildung 8.1: Ableitungsbaum für eine Beispiel-HTML-Tabelle

Die oben angegebene Beispiel-HTML-Tabelle wird z.B. durch eine Ableitung erzeugt, die durch den Ableitungsbaum in Abbildung 8.1 repräsentiert wird.

8.4 Ausblick

Typische Fragestellungen bzgl. kontextfreien Grammatiken:

- (a) Welche Sprachen können prinzipiell durch KFGs erzeugt werden, welche nicht?

Beispiel: Die Sprache $L_1 = \{a^n b^n : n \in \mathbb{N}\}$ wird von der KFG $G_1 = (\Sigma, V, S, P)$ mit $\Sigma = \{a, b\}$, $V = \{S\}$ und $P = \{S \rightarrow aSb \mid \varepsilon\}$ erzeugt, d.h. $L_1 = L(G_1)$.

Notation: $a^n b^n$ ist eine Abkürzung für $\underbrace{aa \cdots a}_{n \text{ mal}} \underbrace{bb \cdots b}_{n \text{ mal}}$.

Satz: Es gibt keine KFG, die genau die Sprache $L_2 := \{a^n b^n c^n : n \in \mathbb{N}\}$ erzeugt.

Beweis: In der Vorlesung "Theoretische Informatik 2 (GL-2)". Dort wird in Analogie zu Satz 7.17 auch ein "Pumping-Lemma für KFGs" gezeigt.

- (b) Gegeben sei eine KFG $G = (\Sigma, V, S, P)$ und ein Wort $w \in \Sigma^*$. Wie kann man herausfinden, ob $w \in L(G)$ ist, d.h. ob das Wort w zu der von G erzeugten Sprache gehört? Dies ist das so genannte **Wortproblem**.

Einen Algorithmus zum Lösen des Wortproblems für KFGs werden Sie in der Vorlesung "Theoretische Informatik 2 (GL-2)" kennenlernen: den so genannten CYK-Algorithmus, der nach seinen Erfindern Cocke, Younger und Kasami benannt ist.

Reguläre Sprachen vs. kontextfreie Grammatiken.

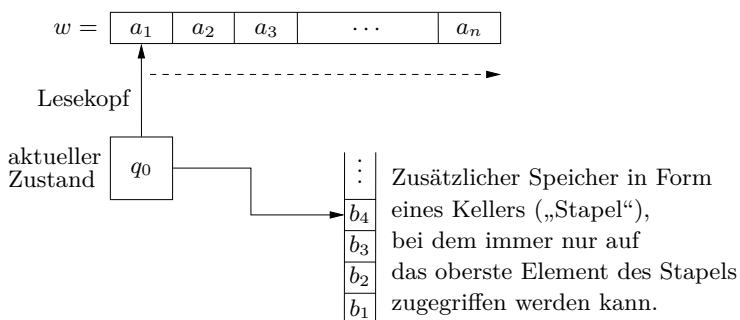
Generell gilt: Für jede reguläre Sprache L gibt es eine kontextfreie Grammatik, die die Sprache L erzeugt. Aber es gibt kontextfreie Grammatiken, die nicht-reguläre Sprachen erzeugen.

Beispiel: Die Sprache $L_1 = \{a^n b^n : n \in \mathbb{N}\}$ wird von der oben angegebenen KFG G_1 erzeugt. Andererseits wissen wir aus Beispiel 7.19, dass die Sprache L_1 nicht regulär ist.

Analog zu DFAs und NFAs gibt es auch ein erweitertes Automatenmodell, das genau diejenigen Sprachen akzeptiert, die von kontextfreien Grammatiken erzeugt werden: so genannte **Kellerautomaten**.

Kellerautomaten

Schematische Darstellung der Verarbeitung eines Eingabeworts durch einen Kellerautomaten:



Details: In der Vorlesung "Theoretische Informatik 2 (GL-2)". Dort werden auch allgemeinere Arten von Grammatiken betrachtet, z.B. so genannte **kontextsensitive Grammatiken**.

kontextsensitive Grammatiken

8.5 Literaturhinweise

- [15] Kapitel 6.1 und 6.2
- [28] Kapitel 6.1
- [29] Kapitel 6.1
- [26] Kapitel 1.3

8.6 Übungsaufgaben zu Kapitel 8

Aufgabe 8.1. Gegeben sei folgende Grammatik $G = (T, N, S, P)$ mit

- $T = \{a, b, \dots, z, @, .\}$

- $N = \{S, X, Y, Z\}$
 - $P = \{S \rightarrow X@Y,$
 $Y \rightarrow X.Z, Z \rightarrow X.Z, Z \rightarrow X,$
 $X \rightarrow \mathbf{a}X, X \rightarrow \mathbf{b}X, \dots, X \rightarrow \mathbf{z}X, X \rightarrow \mathbf{a}, X \rightarrow \mathbf{b}, \dots, X \rightarrow \mathbf{z}\}$
- (a) Überprüfen Sie für jedes der folgenden Worte, ob es in der von G erzeugten Sprache liegt. Wenn ja, dann geben Sie einen Ableitungsbaum für dieses Wort an; ansonsten begründen Sie, warum das Wort nicht zur Sprache gehört.
- (i) meier@web.de c) max.meier@web.de e) root@localhost
(ii) Meier@web.de d) meier@www.web.de

(b) Beschreiben Sie in Worten, welche Sprache $L(G)$ von der Grammatik G erzeugt wird.

Aufgabe 8.2. Im Folgenden wird die Bildungsvorschrift für die Sprache PBA der *positiven Booleschen Ausdrücke* wiederholt, die Ihnen schon von Übung 2.31 bekannt ist. Die Menge PBA ist die Menge der Wörter über dem Alphabet $A = \{\mathbf{0}, \mathbf{1}, \wedge, \vee, (,)\}$, die rekursiv wie folgt definiert ist:

- Basisregel:* (B) Die Symbole $\mathbf{0}$ und $\mathbf{1}$ sind in PBA.
Rekursive Regeln: (R1) Sind w_1 und w_2 in PBA, so ist auch $(w_1 \wedge w_2)$ in PBA.
 (R2) Sind w_1 und w_2 in PBA, so ist auch $(w_1 \vee w_2)$ in PBA.

Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = \text{PBA}$ ist. Geben Sie außerdem eine schrittweise Ableitung analog zu Beispiel 8.7 im Skript für das folgende Wort an:

$$(((\mathbf{0} \vee \mathbf{0}) \vee (\mathbf{1} \vee \mathbf{0})) \wedge (\mathbf{0} \vee \mathbf{1}))$$

Aufgabe 8.3. Für jede Signatur σ ohne Funktionssymbole sei die Sprache KBA_σ der *Konjunktiven Booleschen σ -Anfragen* die Menge der Wörter über dem Alphabet $A := \sigma \cup \{\wedge, \dot{=}, (,)\} \cup \{, \}$, die rekursiv wie folgt definiert ist:

- Basisregel:* (B1) Für alle Konstantensymbole $\dot{a}_1, \dot{a}_2 \in \sigma$ ist $\dot{a}_1 \dot{=} \dot{a}_2$ in KBA_σ .
 (B2) Für jedes Relationssymbol $\mathbb{R} \in \sigma$ und alle Konstantensymbole $\dot{a}_1, \dots, \dot{a}_{\text{ar}(\mathbb{R})} \in \sigma$ ist $\mathbb{R}(\dot{a}_1, \dots, \dot{a}_{\text{ar}(\mathbb{R})})$ in KBA_σ .
Rekursive Regel: (R) Sind w_1 und w_2 in KBA_σ , so ist auch $w_1 \wedge w_2$ in KBA_σ .

So ist bspw. das Wort $\dot{E}(\dot{c}, \dot{c})$ in $\text{KBA}_{\sigma_{\text{Graph}} \cup \{\dot{c}\}}$. Sei nun $\sigma' := \{\mathbb{R}, \dot{x}, \dot{y}, \dot{z}\}$ mit dem zweistelligen Relationssymbol \mathbb{R} und den Konstantensymbolen \dot{x}, \dot{y} und \dot{z} . Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = \text{KBA}_{\sigma'}$ ist. Geben Sie außerdem einen Ableitungsbaum für das Wort $\mathbb{R}(\dot{x}, \dot{y}) \wedge \mathbb{R}(\dot{y}, \dot{z}) \wedge \dot{y} \dot{=} \dot{z}$ entsprechend Ihrer Grammatik G an.

Aufgabe 8.4. Die Sprache REG_{ab} der *Regulären Ausdrücke* über dem Alphabet $\Sigma = \{a, b\}$ ist die Menge der Wörter über dem Alphabet $A = \{\emptyset, \varepsilon, a, b, |, \cdot, *, (,)\}$, die rekursiv wie folgt definiert ist:

- Basisregel:* (B1) Die Symbole \emptyset und ε sind in REG_{ab} .
 (B2) Die Buchstaben a und b sind in REG_{ab} .
Rekursive Regeln: (R1) Ist R in REG_{ab} , so ist auch R^* in REG_{ab} .
 (R2) Sind R und S in REG_{ab} , so ist auch $(R \cdot S)$ in REG_{ab} .
 (R3) Sind R und S in REG_{ab} , so ist auch $(R|S)$ in REG_{ab} .

Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = \mathbf{REG}_{\mathbf{ab}}$ ist. Stellen Sie einen Ableitungsbaum für das folgende Wort auf:

$$((a|b)^* \cdot \emptyset)$$

Geben Sie außerdem eine umgangssprachliche Beschreibung der gemäß Definition (7.23) durch diesen regulären Ausdruck beschriebenen Sprache $L(((a|b)^* \cdot \emptyset))$ an.

Aufgabe 8.5. ¹ Auf Madagaskar ist kürzlich eine neue Schildkrötenart entdeckt worden, die von der Fachwelt schnell den Namen Kompass-Schildkröte erhalten hat. Die erstaunliche und namensgebende Fähigkeit dieser Tiere zeigt sich, sobald man einer solchen Schildkröte (vorsichtig!) mit Kreide ein Wort x über dem Alphabet $\Sigma := \{n, s, o, w\}$ auf den Rückenpanzer schreibt und sie auf die Erde setzt. Für jedes Zeichen in x gelesen von links nach rechts läuft die Schildkröte genau einen Meter in eine bestimmte Richtung: Für n läuft sie nach Norden, für s nach Süden, für o nach Osten und nach Westen für w . Danach bleibt die Schildkröte stehen, das Wort verschwindet und die Schildkröte wartet auf das nächste.

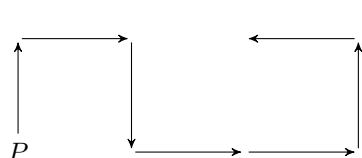


Abbildung 1

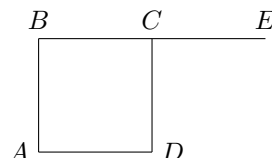


Abbildung 2

Wird der Schildkröte beispielsweise das Wort *nosoow* auf den Rückenpanzer geschrieben und sie bei Position P auf den Boden gesetzt, so läuft sie den in Abbildung 1 gezeichneten Weg ab.

- (a) Betrachten Sie Abbildung 2 mit den Positionen A, B, C, D und E . Die Länge einer Linie zwischen zwei benachbarten Positionen betrage genau ein Meter. Eine Kompass-Schildkröte soll nun auf Position A ausgesetzt werden, sich nur entlang der eingezeichneten Linien bewegen und am Ende wieder zu Position A zurückkehren. Dementsprechend können der Schildkröte beispielsweise die Wörter *ow* oder *noows* auf den Rücken geschrieben werden, nicht aber *wo* oder *onw*. Geben Sie eine kontextfreie Grammatik G an, die genau die Wörter über Σ erzeugt, die einer Kompass-Schildkröte unter diesen Bedingungen auf den Rückenpanzer geschrieben werden können.

Hinweis: Nutzen Sie für jede der Positionen in der Abbildung ein eigenes Nichtterminalsymbol. Beschreiben Sie jede mögliche Richtung, die von jedem dieser Punkte aus jeweils möglich ist durch eine Regel der Grammatik.

- (b) Betrachten Sie die folgende Grammatik $G_2 = (T_2, N_2, S_2, P_2)$ mit $T_2 = \{o, w\}$, $N_2 = \{S, O, W\}$, $S_2 = S$ und

$$P_2 = \{S \rightarrow oW, S \rightarrow wO, S \rightarrow \varepsilon, O \rightarrow oS, O \rightarrow wOO, W \rightarrow wS, W \rightarrow oWW\}$$

- (i) Überprüfen Sie für jedes der folgenden Worte, ob es in der von G_2 erzeugten Sprache liegt. Wenn ja, dann geben Sie einen Ableitungsbaum für dieses Wort an; ansonsten begründen Sie, warum das Wort nicht zur Sprache gehört.

¹Diese Übungsaufgabe entstand in Anlehnung an eine Übungsaufgabe von Thomas Schwentick. Bei der Erstellung der Aufgabe kam keine Kompass-Schildkröte zu Schaden.

$$w_1 = wO \quad w_2 = owwo \quad w_3 = owo \quad w_4 = wwwoowo$$

- (ii) Beschreiben Sie in Worten, welche Sprache $L(G_2)$ von der Grammatik G_2 erzeugt wird. Welches Verhalten zeigt eine Kompass-Schildkröte, wenn ihr ein beliebiges Wort aus $L(G_2)$ auf den Rückenpanzer geschrieben wird?

Aufgabe 8.6. Batman ist mittlerweile genervt von den langweiligen Patrouillenfahrten, die er jede Nacht mit dem Batmobil durch Gotham City unternehmen muss. Doch ihm ist aufgefallen, dass durch die getönten Scheiben des Autos sowieso nicht zu erkennen ist, ob jemand darin sitzt oder nicht. So kommt er auf die Idee, das Batmobil bequem von seiner Villa aus fernzusteuern.

Die per Funk übermittelte Route für das Batmobil besteht aus einem Wort x über dem Alphabet $\Sigma := \{n, s, o, w\}$. Für jedes Zeichen in x gelesen von links nach rechts fährt das Batmobil genau einen Häuserblock bis zur nächsten Kreuzung in eine bestimmte Richtung: Bei n fährt es nach Norden, bei s nach Süden, bei o nach Osten und nach Westen bei w . (Das funktioniert gut, weil der Straßenplan von Gotham City regelmäßig wie Millimeterpapier ist.) Erhält das Batmobil beispielsweise das Wort *sonoosw* und startet bei Position P , so fährt es die in der nebenstehenden Abbildung gezeigte Route ab.

- (a) Betrachten Sie die folgende Grammatik $G_1 = (T_1, N_1, S_1, P_1)$ mit $T_1 = \{w, o\}$, $N_1 = \{S, W, O\}$, $S_1 = S$ und

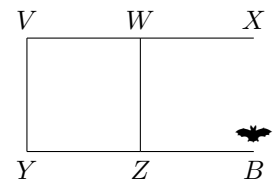
$$P_1 = \{S \rightarrow wO, S \rightarrow oW, S \rightarrow \varepsilon, W \rightarrow wS, W \rightarrow oWW, O \rightarrow oS, O \rightarrow wOO\}$$

- (i) Überprüfen Sie für jedes der folgenden Worte, ob es in $L(G_1)$ liegt. Wenn ja, geben Sie eine schrittweise Ableitung analog zu Beispiel 8.4. aus dem Skript für dieses Wort an; ansonsten begründen Sie, warum das Wort nicht zur Sprache gehört.

$$w_1 = oW \quad w_2 = woow \quad w_3 = w \quad w_4 = oowoww$$

- (ii) Beschreiben Sie, welche Sprache $L(G_1)$ von der Grammatik G_1 erzeugt wird. Was für ein Fahrverhalten zeigt das Batmobil, wenn es ein beliebiges Wort aus $L(G_1)$ als Routenanweisung erhält?

- (b) Batman hat gemerkt, dass alle Ganoven in Gotham City in einem sehr kleinem Bereich wohnen und es ausreicht, dort das Batmobil herum fahren zu lassen, um sie einzuschüchtern. Dieser Bereich besteht aus den Orten V, W, X, Y und Z sowie aus Batmans Versteck, der Bathöhle B . Diese Orte sind durch Straßen wie in der nebenstehenden Abbildung gezeigt miteinander verbunden. Batman braucht nun Fahrtrouten, die bei Position B beginnen, die eingezeichneten Straßen nicht verlassen und bei Position B enden. Mögliche Routen wären demnach *wo* oder *wnwsnoowso*, nicht aber *ns* oder *wn*. Geben Sie eine kontextfreie Grammatik G_2 an, die genau die Wörter über Σ erzeugt, die eine solche Fahrtroute beschreiben.



Hinweis: Nutzen Sie für jede der Positionen in der Abbildung ein eigenes Nichtterminalsymbol. Beschreiben Sie jede mögliche Richtung, die von jedem dieser Positionen aus jeweils möglich ist, durch eine Regel der Grammatik.

Aufgabe 8.7. Sei σ eine Signatur mit einem zweistelligen Relationssymbol \dot{E} , einem zweistelligen Funktionssymbol \dot{f} , einem einstelligen Funktionssymbol \dot{g} und einem Konstantensymbol \dot{c} .

- (a) Konstruieren Sie eine Grammatik G_{Terme} , so dass $L(G_{\text{Terme}})$ genau die Menge aller σ -Terme ist, in denen nur Variablen aus $\{v_0, v_1, v_2\}$ vorkommen. Geben Sie einen Ableitungsbaum für den Term $t := \dot{f}(v_0, \dot{g}(\dot{f}(v_2, \dot{c})))$ an.
- (b) Konstruieren Sie eine Grammatik $G_{\text{FO}[\sigma]}$, so dass $L(G_{\text{FO}[\sigma]})$ genau die Menge aller FO[σ]-Formeln ist, in denen nur Variablen aus $\{v_0, v_1, v_2\}$ vorkommen. Geben Sie einen Ableitungsbaum für die Formel $\varphi := \forall v_0 (\dot{E}(v_0, \dot{f}(v_2, \dot{c})) \rightarrow \dot{g}(v_1) \dot{=} v_0)$ an.

9 Ausblick auf weitere Modellierungstechniken

In diesem Kapitel werden die Grundzüge von zwei weiteren Kalkülen vorgestellt, die sich zur Beschreibung von Abläufen bzw. zur Modellierung von Datenbanken eignen: zum einen die **Petri-Netze** in Abschnitt 9.1, und zum anderen das **Entity-Relationship-Modell** in Abschnitt 9.2. Das Kapitel schließt in Abschnitt 9.3 mit einer **Fallstudie** ab, in der das Zusammenspiel von verschiedenen Modellierungskalkülen zur Beschreibung eines konkreten, etwas umfangreicheren Problems beleuchtet wird.

9.1 Petri-Netze zur Modellierung von Abläufen

Petri-Netze liefern einen formalen Kalkül, der besonders gut geeignet ist, um Abläufe zu modellieren, an denen mehrere Prozesse beteiligt sind. Hierbei werden die Interaktionen zwischen Prozessen, sowie die Effekte modelliert, die sich daraus ergeben, dass Operationen prinzipiell gleichzeitig ausgeführt werden können (Stichwort: „Nebenläufigkeit“). Daher eignen sich Petri-Netze besonders gut dazu, nebenläufige Prozesse zu beschreiben, bei denen Ereignisse gleichzeitig an mehreren Stellen des Systems Zustandsänderungen bewirken können. Petri-Netze wurden 1962 von C. A. Petri eingeführt.

Petri-Netze eignen sich beispielsweise gut zur Modellierung von

- realen oder abstrakten Automaten und Maschinen,
- kommunizierenden Prozessen (z.B. in Rechnern),
- Verhalten von Software- oder Hardware-Komponenten,
- Geschäftsabläufen in einer Firma,
- Spielen (bzw. Spielregeln),
- biologischen Prozessen (Bioinformatik).

Der Kalkül der Petri-Netze basiert auf bipartiten gerichteten Graphen: Es gibt zwei Sorten von Knoten:

- zum einen, Knoten, die „Bedingungen“ (so genannte **Stellen**) repräsentieren, und
- zum anderen, Knoten, die „Aktivitäten“ (so genannte **Transitionen**) repräsentieren.

Kanten verbinden „Aktivitäten“ mit ihren „Vorbedingungen“ und ihren „Nachbedingungen“. Knotenmarkierungen repräsentieren den veränderlichen „Zustand“ des Systems.

Definition 9.1 (Petri-Netz). Ein **Petri-Netz** $P = (S, T, F)$ besteht aus

Petri-Netz

- einer endlichen Menge S , den so genannten **Stellen** von P ,
- einer endlichen Menge T , den so genannten **Transitionen** von P ,
- einer Relation $F \subseteq (S \times T) \cup (T \times S)$, den so genannten **Kanten** von P .

Stellen

Transitionen

Die Mengen S und T sind disjunkt, d.h. $S \cap T = \emptyset$.

Ein Petri-Netz P bildet einen bipartiten gerichteten Graphen mit Knotenmenge $S \cup T$ und Kantenmenge F .

In der graphischen Darstellung werden **Stellen**, d.h. Knoten in S , durch Kreise repräsentiert; **Transitionen**, d.h. Knoten in T , werden durch Rechtecke dargestellt.

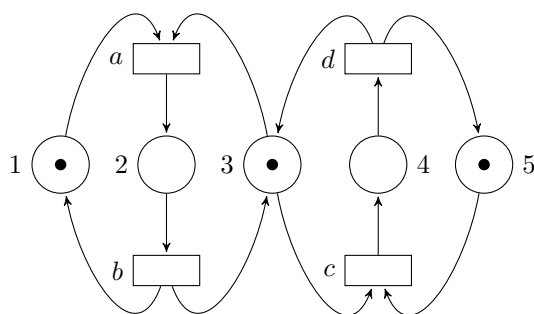
Definition 9.2 (Markierung).

Der „aktuelle Zustand“ eines Petri-Netzes $P = (S, T, F)$ wird durch eine **Markierungsfunktion** (kurz: **Markierung**) $M: S \rightarrow \mathbb{N}$ repräsentiert, die jeder Stelle $s \in S$ eine Anzahl $M(s)$ von so genannten **Marken** zuordnet.

Markierung

Marken

Beispiel 9.3. Die folgende Skizze gibt die graphische Darstellung eines Petri-Netzes $P = (S, T, F)$ und einer Markierung M an.



Die einzelnen „Marken“, die M einer Stelle $s \in S$ zuordnet, werden durch Punkte „•“ in dem die Stelle s repräsentierenden Knoten dargestellt.

Die obige Skizze repräsentiert das Petri-Netz $P = (S, T, F)$ und die Markierung $M: S \rightarrow \mathbb{N}$ mit

- $S = \{1, 2, 3, 4, 5\}$,
- $T = \{a, b, c, d\}$,
- $F = \{(1, a), (3, a), (a, 2), (2, b), (b, 1), (b, 3), (3, c), (5, c), (c, 4), (4, d), (d, 3), (d, 5)\}$,
- $M: S \rightarrow \mathbb{N}$ mit $M(1) = M(3) = M(5) = 1$ und $M(2) = M(4) = 0$.

Ein Petri-Netz P zusammen mit einer Markierung M kann man sich als eine „Momentaufnahme“, d.h. als die Beschreibung eines „aktuellen Zustands“ eines Systems vorstellen. Um Änderungen am Zustand des Systems beschreiben zu können, sind die folgenden Begriffe nützlich.

Definition 9.4 (Vorbereich und Nachbereich).

Sei $P = (S, T, F)$ ein Petri-Netz und sei $t \in T$ eine Transition von P . Wir setzen

Vorbereich(t)

$$\bullet \text{ Vorbereich}(t) := \{s \in S : (s, t) \in F\},$$

Nachbereich(t)

$$\bullet \text{ Nachbereich}(t) := \{s \in S : (t, s) \in F\}.$$

Somit ist $\text{Vorbereich}(t)$ die Menge aller Stellen, von denen aus eine Kante in t hineinführt, während $\text{Nachbereich}(t)$ die Menge aller Stellen, ist in die eine von t ausgehende Kante hinführt.

Petri-Netze verändern ihren „Zustand“, indem Transitionen „schalten“ und dadurch die Markierung des Petri-Netzes ändern. Dies wird folgendermaßen präzisiert:

Definition 9.5 (Schaltregel).

Sei $P = (S, T, F)$ ein Petri-Netz und sei $M: S \rightarrow \mathbb{N}$ eine Markierung. Das so genannte **Schalten einer Transition** $t \in T$ überführt die Markierung M in eine Markierung $M': S \rightarrow \mathbb{N}$. Die Transition t **kann schalten**, wenn gilt:

F.a. Stellen $s \in \text{Vorbereich}(t)$ ist $M(s) \geq 1$.

D.h.: Jede Stelle s in $\text{Vorbereich}(t)$ hat mindestens eine Marke.

Nachfolge-
markierung

Wenn die Transition t schaltet, so gilt für die so genannte **Nachfolgemarkierung** M' folgendes:
f.a. $s \in S$ ist

$$M'(s) = \begin{cases} M(s) - 1, & \text{falls } s \in \text{Vorbereich}(t) \setminus \text{Nachbereich}(t) \\ M(s) + 1, & \text{falls } s \in \text{Nachbereich}(t) \setminus \text{Vorbereich}(t) \\ M(s), & \text{sonst.} \end{cases}$$

D.h.: Das **Schalten von Transition** t bewirkt, dass in jeder Stelle in $\text{Vorbereich}(t)$ eine Marke entfernt wird und dass in jeder Stelle in $\text{Nachbereich}(t)$ eine Marke hinzugefügt wird. Wenn mehrere Transitionen schalten können, so wird eine davon „nichtdeterministisch“ ausgewählt.

In jedem Schritt schaltet genau eine Transition. Durch schrittweises Schalten von Transitionen wird der Ablauf von Prozessen modelliert.

Beispiel 9.6. Seien $P = (S, T, F)$ und $M: S \rightarrow \mathbb{N}$ das Petri-Netz und die Markierung aus Beispiel 9.3. Bei der angegebenen Markierung M können die Transitionen a und c schalten. Wenn wir a schalten lassen, ergibt sich als Nachfolgemarkierung die Markierung $M': S \rightarrow \mathbb{N}$ mit $M'(1) = 0$, $M'(3) = 0$, $M'(2) = 1$, $M'(4) = 0$, $M'(5) = 1$. Die graphische Darstellung von P und M' ist in Abbildung 9.1 angegeben.

Als nächstes kann Transition c **nicht** schalten, da die Stelle 3 keine Marke trägt. Die einzige Transition, die jetzt schalten kann, ist Transition b , deren Schalten bewirkt, dass die Marke bei 2 verschwindet und stattdessen Marken bei 1 und 3 erzeugt werden. Nach dem Schalten von b ist das System also wieder in seinem „ursprünglichen Zustand“, d.h. es trägt die Markierung M .

Insgesamt gilt: Das Petri-Netz P aus Beispiel 9.3, zusammen mit der Startmarkierung M , modelliert zwei zyklisch ablaufende Prozesse. Die Stelle 3 „synchronisiert“ die beiden Prozesse, so dass sich nie gleichzeitig in den beiden Stellen 2 und 4 eine Marke befinden kann. Auf diese Weise könnte man z.B. beschreiben, wie Autos eine 1-spurige Brücke von 2 Seiten überqueren, so dass sich immer nur 1 Auto auf der Brücke befindet.

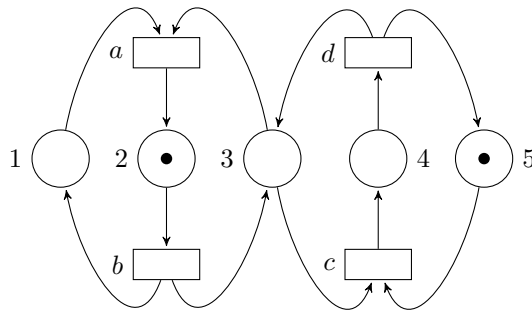


Abbildung 9.1: Skizze zu Beispiel 9.6

Beispiel 9.7. Ziel ist, die Schaltung einer Ampelanlage an einer Kreuzung zu modellieren. Dabei gibt es zwei sich zyklisch wiederholende Prozesse:

- „Grün“-Phase in Nord-Süd-Richtung,
- „Grün“-Phase in West-Ost-Richtung.

Die beiden Prozesse sollen sich immer abwechseln. Die graphische Darstellung eines Petri-Netzes, inklusive Anfangs-Markierung, das eine solche Ampelanlage modelliert, findet sich in Abbildung 9.2.

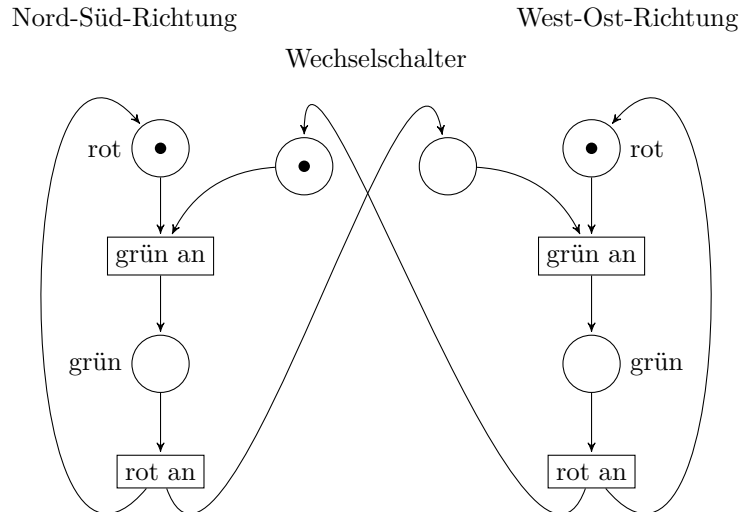


Abbildung 9.2: Petri-Netz zur Modellierung einer Ampelanlage

Die beiden Stellen namens „Wechselschalter“ koppeln die Prozesse, so dass abwechselnd in West-Ost- bzw. in Nord-Süd-Richtung die Ampel „grün“ ist.

9.2 Das Entity-Relationship-Modell zur Modellierung von Datenbanken

ER-Modell	<p>Das Entity-Relationship-Modell (kurz: ER-Modell) geht zurück auf einen grundlegenden Artikel [4] von P. P. Chen aus dem Jahr 1976.</p> <p>Das ER-Modell wird heute als Standard-Werkzeug für frühe Entwurfsphasen in der Datenbankentwicklung eingesetzt. Darüber hinaus basiert auch die Spezifikationsprache UML („Unified Modeling Language“), die zur Spezifikation von Strukturen und Beziehungen in Software-Systemen eingesetzt wird, auf dem ER-Modell.</p> <p>In dieser Vorlesung werden nur einige grundlegende Züge des ER-Modells vorgestellt. Details können Sie z.B. in der Veranstaltung „Datenbanksysteme I“ kennenlernen.</p> <p>Das ER-Modell basiert auf den 3 Grundkonzepten</p>
Entity	<ul style="list-style-type: none"> • Entity: „zu modellierende Informationseinheit“ (deutsch: „Objekt“, „Ding“, „Entität“)
Relationship	<ul style="list-style-type: none"> • Relationship: zur Modellierung von Beziehungen zwischen Entities (deutsch: „Beziehung“, „Relation“)
Attribut	<ul style="list-style-type: none"> • Attribut: Eigenschaft eines Entitys oder einer Beziehung.
	Genauer:
Entity	<ul style="list-style-type: none"> • Entity: ein Objekt der realen oder der Vorstellungswelt, über das Informationen zu speichern sind (z.B. eine Vorlesungsveranstaltung, ein Buch oder eine/n Dozent/in). Auch Informationen über Ereignisse wie Klausuren können Objekte im Sinne des ER-Modells sein.
Entity-Typ Entity-Menge	<ul style="list-style-type: none"> • Entity-Typ (bzw. Entity-Menge): eine Zusammenfassung von Entities, die im Modell als „gleichartig“ angesehen werden (z.B. „Vorlesung“, „Buch“, „Dozent/in“). Im Modell steht ein Entity-Typ für die Menge aller in Frage kommenden Objekte dieser Art.
Relationship	<ul style="list-style-type: none"> • Relationship: Beziehung zwischen Entities (z.B. welche Dozenten/innen welche Vorlesungen halten).
Attribut	<ul style="list-style-type: none"> • Attribut: Eigenschaften von Entities oder Relationships (z.B. die ISBN eines Buchs, der Titel einer Vorlesung oder die Semester, in denen Vorlesung X von Dozent/in Y gehalten wird).

Beispiel 9.8. Abbildung 9.3 zeigt eine graphische Darstellung für eine Modellierung im ER-Modell. In diesem Beispiel geht es darum, Vorlesungen, Dozenten und für die Vorlesungen empfohlene Bücher darzustellen.

Die graphische Darstellung eines ER-Modells geschieht nach folgenden Konventionen:

- **Entity-Typen** werden als Rechtecke dargestellt (hier: Dozent/in, Vorlesung, Buch).
- **Eigenschaften von Entities**, so genannte **Attribute**, werden durch Ellipsen dargestellt, die mit dem Rechteck des zugehörigen Entity-Typs verbunden sind (im Beispiel hat jede/r Dozent/in die Attribute „Name“, „Fach“ und „Email-Adresse“).

Ein Attribut ordnet jedem Entity des entsprechenden Entity-Typs einen Wert zu. Ein Attribut, dessen Wert jedes Entity eindeutig bestimmt (z.B. die ISBN von Büchern), heißt **Schlüsselattribut**. Um Schlüsselattribute im ER-Modell explizit zu kennzeichnen, werden sie unterstrichen. Auch mehrere Attribute zusammen können einen Schlüssel bilden, z.B.

Schlüsselattribut

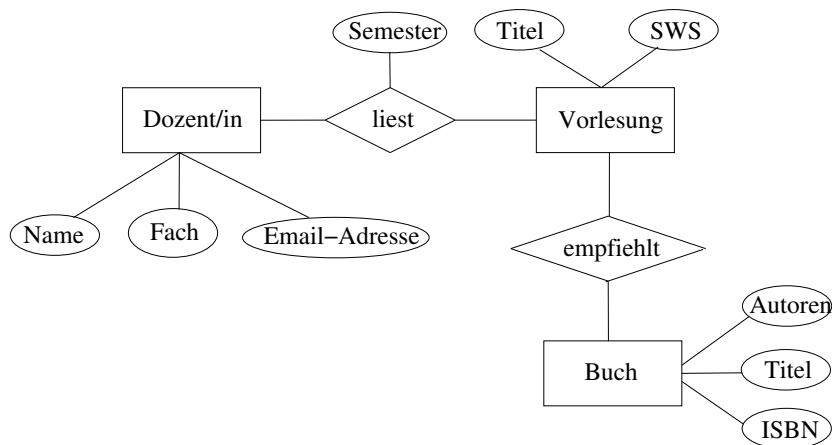
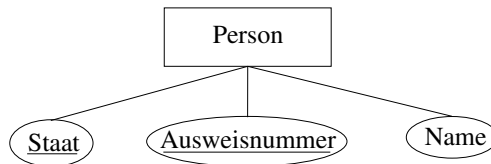


Abbildung 9.3: Ein ER-Modell, das Vorlesungen, Dozenten und für die Vorlesungen empfohlene Bücher darstellt

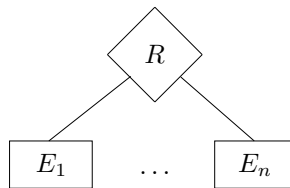


- Typen von Relationships, so genannte **Relationen-Typen**, werden durch Rauten dargestellt, die mit den betreffenden Entity-Typen durch Striche verbunden sind (z.B. ist in Beispiel 9.8 „liest“ ein Relationen-Typ, der angibt, welche/r Dozent/in welche Vorlesung liest).

Relationen-Typen

Allgemein gilt: Ein Relationen-Typ modelliert Beziehungen zwischen den Entities der betroffenen Entity-Typen. Ein **n -stelliger Relationen-Typ R** (für $n \geq 2$) verknüpft Entities aus n Entity-Typen E_1, \dots, E_n . Er wird graphisch repräsentiert durch

n -stelliger Relationen-Typ

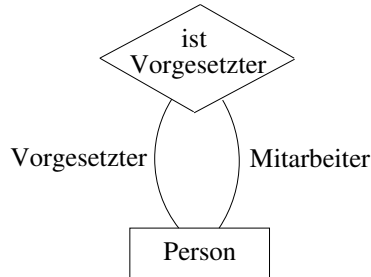


Eine **konkrete Ausprägung des Relationen-Typs R** ist eine Menge von n -Tupeln (e_1, \dots, e_n) , wobei für jedes $i \in \{1, \dots, n\}$ gilt: e_i ist ein Entity des Entity-Typs E_i .

- Auch Relationen-Typen können Attribute haben (z.B. hat der Relationen-Typ „liest“ in Beispiel 9.8 ein Attribut „Semester“, das angibt, in welchen Semestern Dozent/in X die Vorlesung Y hält).

Allgemein gilt: Ein Attribut ordnet jedem Tupel des entsprechenden Relationen-Typs einen Wert zu. Beispielsweise ordnet das Attribut „Semester“ jedem Tupel (X, Y) der „liest“-Relationen die Liste aller Semester zu, in denen Dozent/in X die Vorlesung Y hält.

- Für manche Relationen-Typen wird aus ihrem Namen und der graphischen Darstellung zunächst nicht klar, welche Bedeutung die einzelnen Entity-Typen in der Relation haben — insbesondere dann, wenn ein Entity-Typ mehrfach am Relationen-Typ beteiligt ist. Es können dann **Rollenamen** vergeben werden, etwa um die Beziehung „Person X ist Vorgesetzter von Person Y “ darzustellen:



Beispiel 9.9. Man beachte die Auswirkung von Modellierungsentscheidungen beim Entwickeln eines ER-Modells:

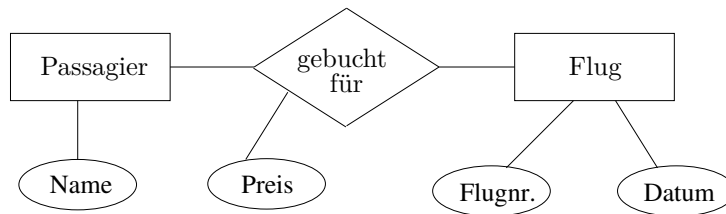
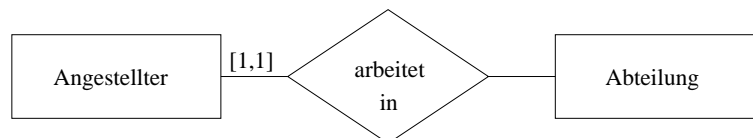


Abbildung 9.4: ER-Modell für ein Reisebüro

Nutzt ein Reisebüro das ER-Modell aus Abbildung 9.4, so besteht eine konkrete Ausprägung des Relationen-Typs „gebucht für“ aus einer Menge von Tupeln (X, Y) , die angibt, dass Person X ein Ticket für Flug Y gebucht hat — und zwar zum Preis $\text{Preis}(X, Y)$. Insbesondere heißt dies aber, dass Passagier X für Flug Y nicht zwei verschiedene Buchungen getätigt haben kann. Wenn man solche „Mehrfachbuchungen“ zulassen will, kann man das ER-Modell aus Abbildung 9.5 benutzen.

Ein weiterer Bestandteil von ER-Modellen: die Kardinalität von Relationen-Typen:

Relationen-Typen in der Form, wie wir sie bisher eingeführt haben, sagen über konkrete Ausprägungen nur aus, dass einige Entities aus den beteiligten Entity-Typen in der angegebenen Beziehung stehen können. Oft will man aber genauere Angaben (bzw. Einschränkungen) machen — z.B., dass jeder Angestellte durch eine Relation des Relationen-Typs „arbeitet in“ mit genau einer Abteilung verbunden ist. Dies kann graphisch folgendermaßen dargestellt werden:



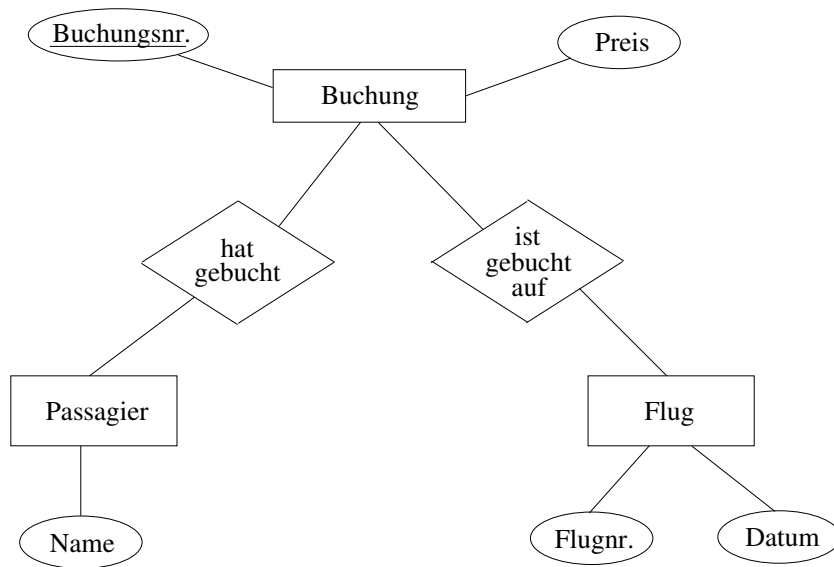
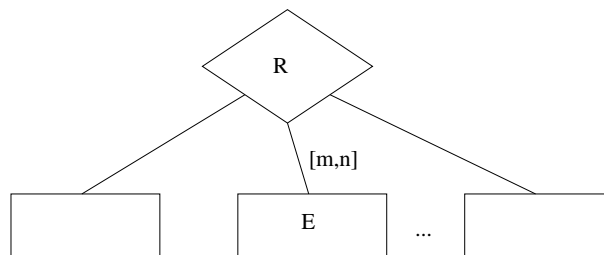


Abbildung 9.5: ER-Modell für ein Reisebüro mit „Mehrfachbuchungen“

Allgemein besagt ein Relationen-Typ der Form

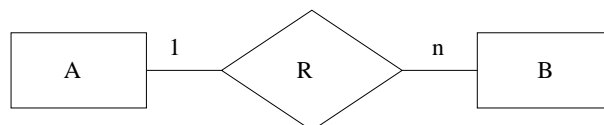


dass für jede konkrete Ausprägung dieses Typs gelten muss: Jedes Entity e der konkreten Ausprägung des Entity-Typs E kommt in mindestens m und höchstens n Tupeln vor.

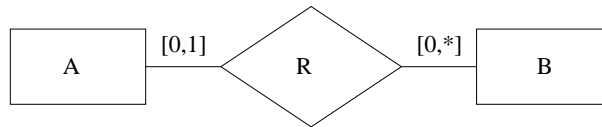
Spezialfälle für $[m, n]$:

- $[1, 1]$ bedeutet: „in genau einem Tupel“,
- $[0, 1]$ bedeutet: „in höchstens einem Tupel“,
- $[0, *]$ bedeutet: „in beliebig vielen Tupeln“. Die Angabe $[0, *]$ wird oft einfach weggelassen.

Kurznotation für 2-stellige Relationen-Typen:



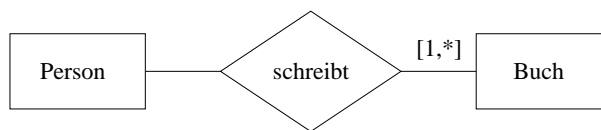
bedeutet



d.h.: „jedes Entity a des Typs A kommt in höchstens einem Tupel von R vor, und jedes Entity b des Typs B kommt in beliebig vielen Tupeln von R vor.“

Beispiel 9.10.

(a)



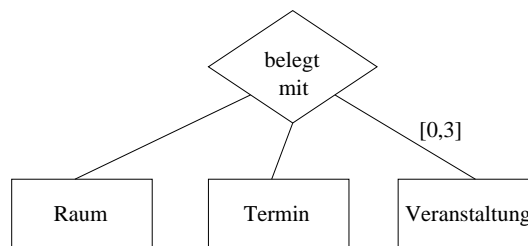
bedeutet: „Jedes Buch wird von mindestens einer Person geschrieben.“

(b)



bedeutet: „Jeder Termin im Stundenplan ist mit höchstens einer Veranstaltung belegt.“

(c)



bedeutet: „Jede Veranstaltung wird höchstens dreimal (pro Woche) angeboten.“

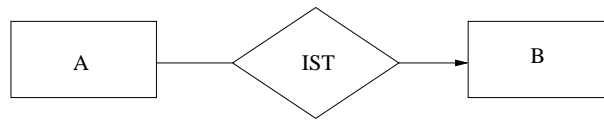
IST-Beziehung

Ein weiterer Bestandteil von ER-Modellen: Die IST-Beziehung¹

Der spezielle Relationen-Typ IST definiert eine Spezialisierungs-Hierarchie.

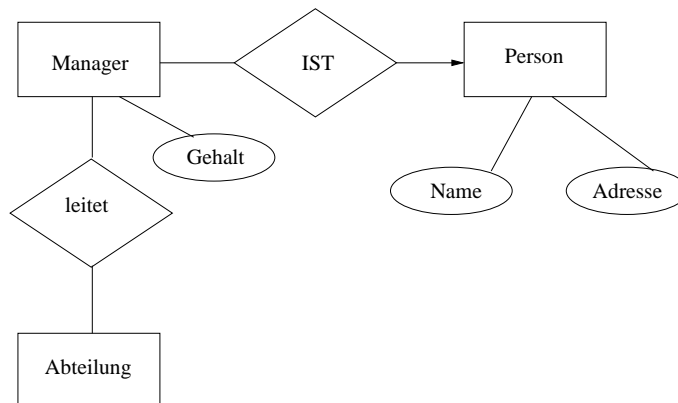
Graphische Darstellung:

¹englisch: "is-a"



Bedeutung: Jedes Entity des Typs A ist auch ein Entity des Typs B (d.h. A ist eine Spezialisierung des Typs B).

Beispiel:



Allgemein gilt: Die Entities des Typs A „erben“ alle Attribute von B und können außerdem noch weitere Attribute haben, die spezielle „ A -Eigenschaften“ beschreiben. Auch Schlüsselattribute werden als solche geerbt.

Beispiel 9.11. Ein umfangreiches ER-Modell, das einige Aspekte einer Fluggesellschaft modelliert, ist in Abbildung 9.6 dargestellt. In diesem ER-Modell wird u.a. folgendes modelliert:

- (1) Es kommt eine IST-Spezialisierung vor, die besagt, dass Piloten spezielle Angestellte sind. Das wird insbesondere benötigt, um den Relationen-Typ „kann fliegen“ hinreichend präzise formulieren zu können und das Attribut „Flugstunden“ nicht allen Angestellten zuordnen zu müssen.
- (2) Entities aller hier aufgeführten Typen (bis auf Abflug) werden durch Schlüsselattribute eindeutig identifiziert. Bei den Passagieren wird angenommen, dass Name und Adresse den jeweiligen Passagier eindeutig festlegen. Die Namen der übrigen Schlüsselattribute deuten an, dass man jeweils eine Nummerierung eingeführt hat, um die Eindeutigkeit zu erreichen (z.B. Personalnr., Flugnr., etc.).
- (3) Der Relationen-Typ „Typ“ verbindet konkrete Flugzeuge mit Flugzeugtypen, indem jedem Flugzeug genau ein Flugzeugtyp zuordnet wird. Solche Unterscheidungen zwischen „Typ“ und „Exemplar“ werden oft in Modellierungen verwendet und sind wichtig, damit Relationen und Attribute sachgerecht zugeordnet werden können. Beispielsweise sind die Fähigkeiten eines Piloten dadurch bestimmt, welche Flugzeugtypen er fliegen kann — und nicht durch die konkreten Flugzeuge (d.h. Exemplare), die er fliegen kann.

Vorsicht: Beim ersten Hinsehen mag es verlockend erscheinen, Typ-Exemplar-Beziehungen durch die IST-Spezialisierung zu modellieren. Dies ist aber ein schwerer Entwurfsfehler: „Typen“ und „Exemplare“ bezeichnen verschiedenartige Entity-Typen, zwischen denen i.d.R. keine Teilmengen-Beziehung (wie bei der IST-Spezialisierung) bestehen kann.

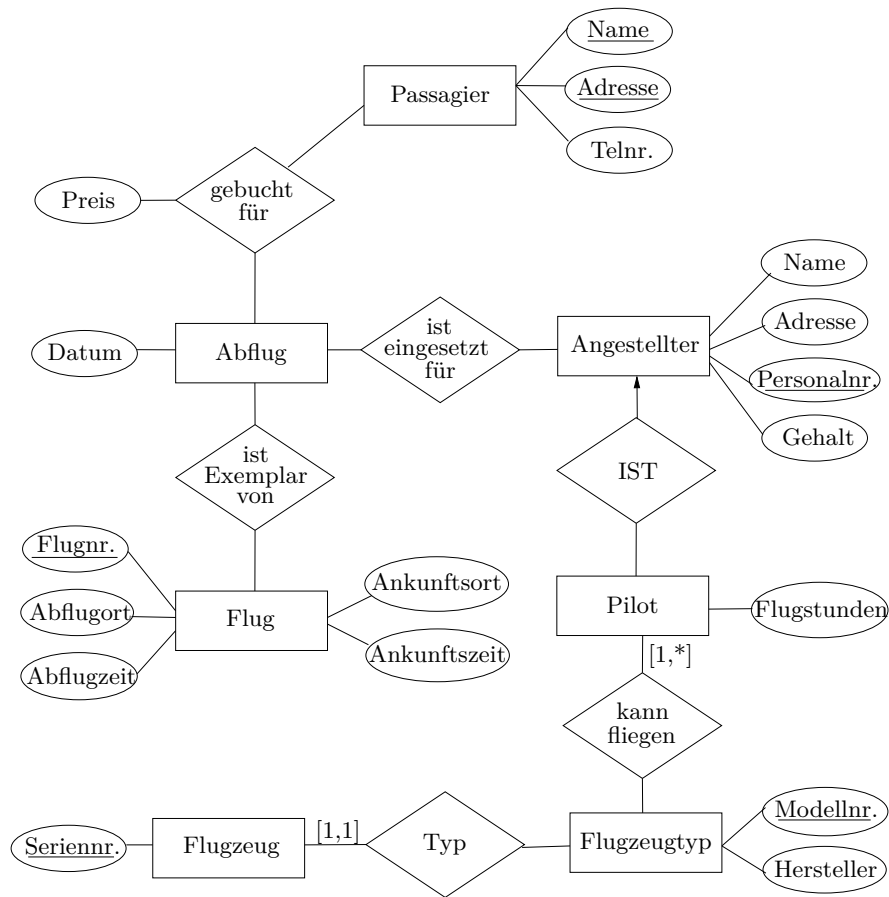


Abbildung 9.6: ER-Modell, das einige Aspekte einer Fluggesellschaft modelliert

9.3 Eine Fallstudie

In diesem Abschnitt steht ein konkretes Anwendungsbeispiel im Vordergrund. Seine Strukturen, Eigenschaften etc. werden mit verschiedenen Kalkülen modelliert. Die unterschiedlichen Kalküle werden eingesetzt, um unterschiedliche Aspekte des Anwendungsbeispiels zu beschreiben.

Als konkretes Anwendungsbeispiel betrachten wir hier eine **Autowerkstatt**. Unser Ziel ist, die Auftragsabwicklung der Autowerkstatt zu modellieren. Dabei soll zum einen eine geeignete Datenbank entworfen werden; zum anderen sollen Abläufe in der Autowerkstatt analysiert und verbessert werden.

9.3.1 Datenbank-Entwurf: Autowerkstatt

Es sollen Daten zu Kunden, Aufträgen, Kraftfahrzeugen (kurz: KFZ) und KFZ-Typen gespeichert werden:

- Jeder **Kunde** hat einen Namen, besitzt Kraftfahrzeug(e) (kurz: KFZ) und erteilt ggf. einen oder mehrere Aufträge.

- Jeder **Auftrag** hat ein Eingangsdatum, betrifft ein KFZ und wird von einem oder mehreren Mechaniker(n) bearbeitet und benötigt Ersatzteile bestimmter Arten und Mengen (im Sinne von „Anzahlen“)
- Jedes **KFZ** hat eine Fahrgestellnummer und ein Baujahr und ist entweder ein PKW oder ein Motorrad. Bei PKWs sollen Informationen zur Farbe, bei Motorrädern Informationen zum Tuningsatz gespeichert werden.
- Jedes KFZ hat einen **Typ**. Jeder Mechaniker ist für einen oder mehrere Typen ausgebildet; Ersatzteile sind für bestimmte Typen verwendbar.

Zum Entwurf eines ER-Modells zum Verwalten dieser Daten, wählen wir folgende Entity- und Relationen-Typen.

Zentrale Entity-Typen:

- Kunde
- Auftrag
- KFZ
- KFZ-Typ.

Zentrale Relationen-Typen:

- besitzt (Kunde besitzt KFZ)
- erteilt (Kunde erteilt Auftrag)
- betrifft (Auftrag betrifft KFZ)
- hat Typ (KFZ hat KFZ-Typ)
- ausgebildet für (Mechaniker ist ausgebildet für KFZ-Typen)

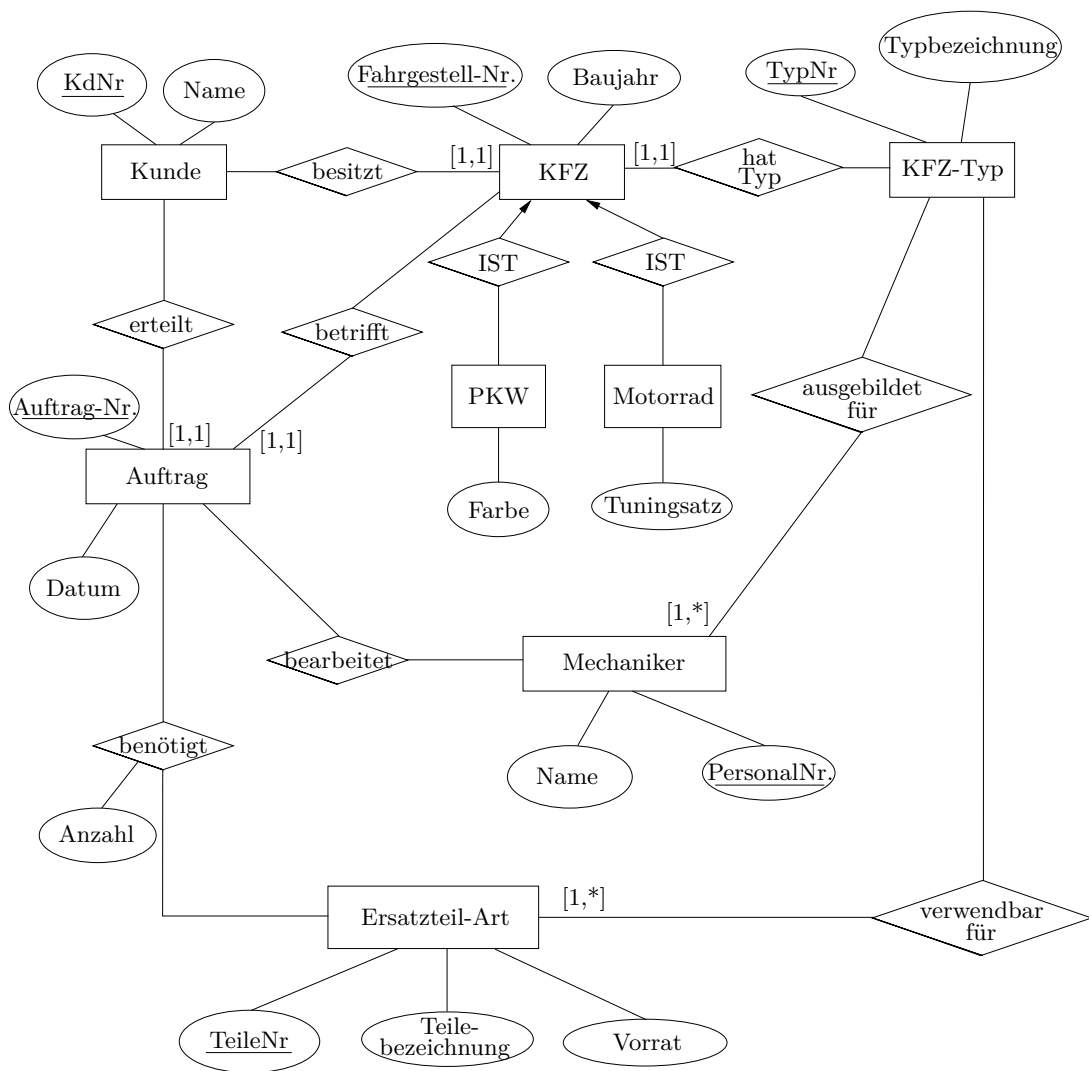
Für den letzten Relationen-Typ ist es sinnvoll einen weiteren *Entity-Typ* einzuführen:

- Mechaniker.

Des Weiteren benötigen wir noch *Relationen-Typen* zur Modellierung davon,

- welche Ersatzteile ein Auftrag benötigt,
- welche Ersatzteile für welchen KFZ-Typ geeignet sind,
- welcher Mechaniker welchen Auftrag bearbeitet.

Außerdem müssen wir eine Möglichkeit bereitstellen, KFZ in PKW und Motorräder einzuteilen. Dies wird durch das folgende ER-Modell gewährleistet:



Beachte: Durch Angabe von Kardinalitäten haben wir einige Entscheidungen getroffen:

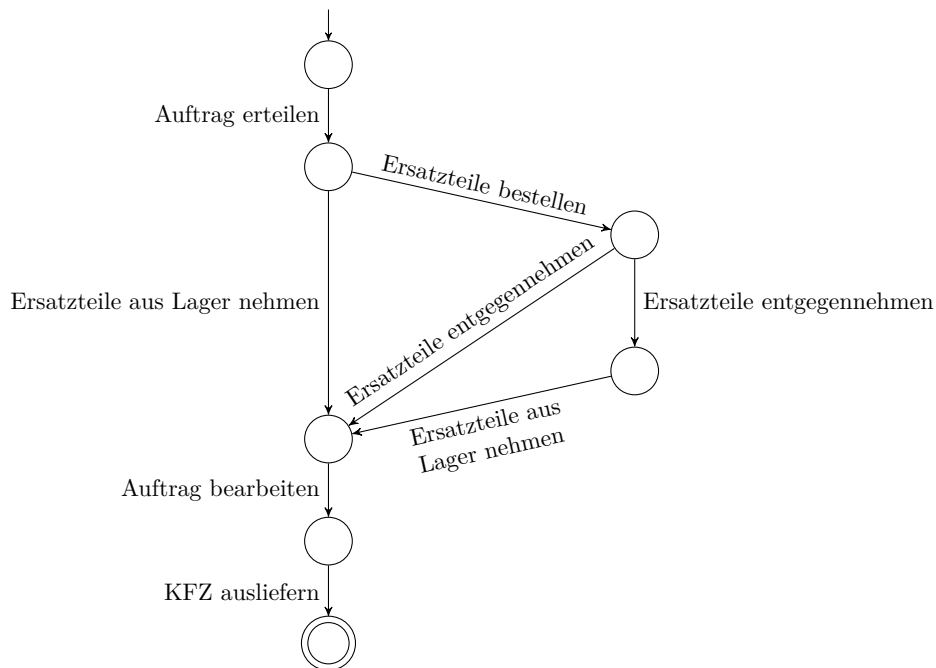
- Jedes KFZ hat genau einen KFZ-Typ.
- Jedes KFZ hat genau einen Besitzer.
- Jeder Auftrag betrifft genau ein KFZ.
- Jeder Auftrag wird von genau einem Kunden erteilt.
- Jeder Mechaniker ist für mindestens einen KFZ-Typ ausgebildet.
- Jede Ersatzteil-Art ist für mindestens einen KFZ-Typ verwendbar.

9.3.2 Abläufe bei der Auftragserteilung

Eine Untersuchung der Geschäftsabläufe in der Autowerkstatt ergibt, dass jeder Auftrag folgende Stationen durchläuft:

- (1) Der Auftrag wird erteilt.
- (2) Fehlende Ersatzteile werden bestellt und nach dem Eintreffen entgegengenommen.
- (3) Vorhandene Ersatzteile werden aus dem Lager genommen.
- (4) Der Auftrag wird von einem Mechaniker bearbeitet.
- (5) Das KFZ wird dem Kunden ausgeliefert.

Modellierung dieser Abläufe als Transitionssystem (d.h. als endlicher Automat):



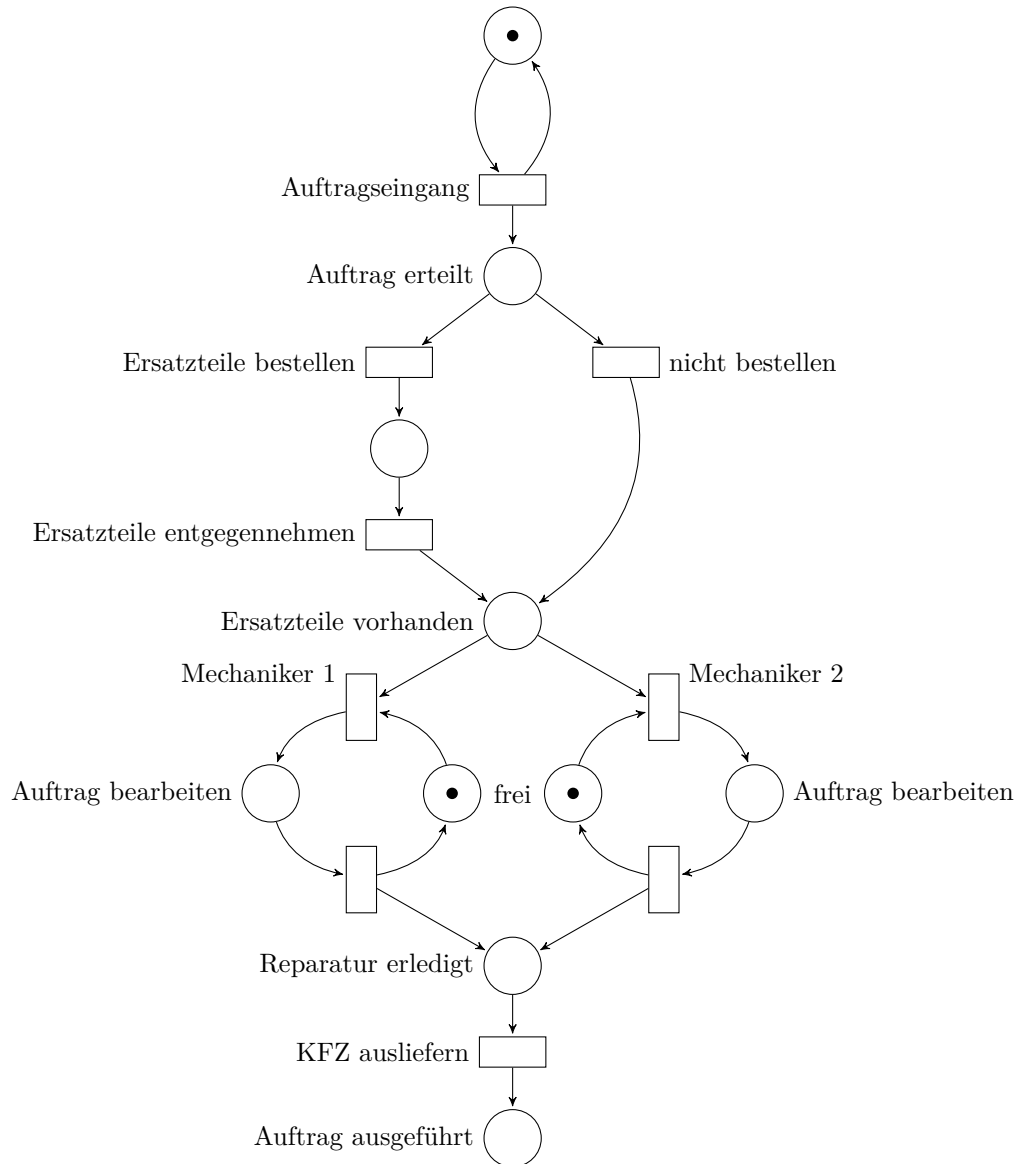
Einschränkungen dieses Modells:

- Die Abläufe in der Werkstatt werden uns aus der Sicht eines einzelnen Auftrags beschrieben.
- Das Modell spricht nur über die „Ersatzteile insgesamt“, aber nicht über ihre Art und Anzahl.
- Aktionsfolgen, bei denen mehrere Aufträge von mehreren Mechanikern bearbeitet werden, können durch dieses Modell nicht beschrieben werden.

Modellierung der Auftragsbearbeitung durch ein Petri-Netz:

Ziel: Modelliere, wie mehrere Aufträge nebenläufig von zwei miteinander um Aufträge konkurrierenden Mechanikern bearbeitet werden.

Petri-Netz:

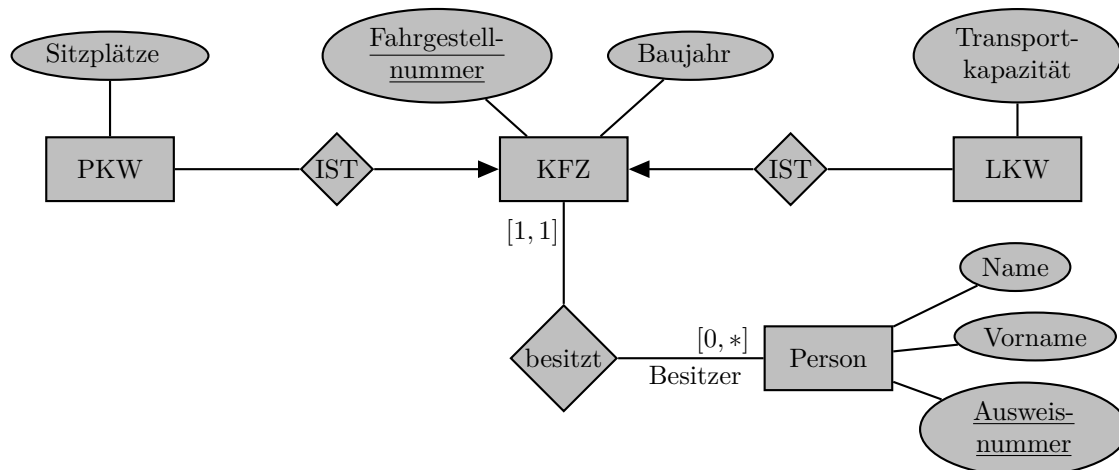


9.4 Literaturhinweise

Dieses Kapitel orientiert sich an den Kapiteln 6.3, 7.2 und 8.1 von [15]. Einen umfassenden Überblick zum Thema Petri-Netze gibt das Buch [24]. Eine Einführung in Methoden zum Datenbankentwurf und insbesondere ins Entity-Relationship-Modell wird in [11] gegeben.

9.5 Übungsaufgaben zu Kapitel 9

Aufgabe 9.1. Es sei folgendes Entity-Relationship-Modell gegeben:



Weiterhin seien die folgenden konkreten Ausprägungen der einzelnen Entity-Typen PKW, LKW, KFZ und Person gegeben: $PKW = \{PKW1, PKW2\}$, $LKW = \{LKW\}$, $KFZ = PKW \cup LKW$ und $Person = \{Person1, Person2\}$ mit:

- *PKW1*: Fahrgestellnummer: 123421, Baujahr: 1999, Sitzplätze: 4
- *PKW2*: Fahrgestellnummer: 123123, Baujahr: 2003, Sitzplätze: 6
- *LKW*: Fahrgestellnummer: 123131, Baujahr: 1994, Transportkapazität: 7 Tonnen
- *Person1*: Ausweisnummer: 1234567890, Name: Meier, Vorname: Max
- *Person2*: Ausweisnummer: 9876543210, Name: Müller, Vorname: Martha

- (a) Welche der folgenden Relationen sind zulässige Ausprägungen des Relationen-Typs „besitzt“?
- a) $\{(Person1, PKW1), (Person2, PKW2), (Person1, LKW)\}$
 - b) $\{(Person1, PKW1), (Person1, PKW2), (Person1, LKW), (Person2, PKW2)\}$
 - c) $\{(Person1, PKW1), (Person1, PKW2), (Person1, LKW)\}$
 - d) $\{(Person1, PKW1), (Person2, LKW)\}$
- (b) Würde es dem Modell widersprechen, wenn
- (a) *Person1* und *Person2* den gleichen (Nach)Namen hätten?
 - (b) *PKW1* und *PKW2* die gleiche Fahrgestellnummer hätten?
 - (c) *PKW1* und *LKW* das gleiche Baujahr hätten?
 - (d) *PKW2* ein Entity vom Typ KFZ (d.h. $PKW2 \in KFZ$), aber nicht vom Typ PKW (d.h. $PKW2 \notin PKW$) wäre?

Begründen Sie jeweils Ihre Antworten!

Aufgabe 9.2. In einer Bibliothek gibt es verschiedene Exemplare von Büchern. Die Bücher sind eindeutig über Ihre ISBN gekennzeichnet und besitzen darüberhinaus noch einen Titel, eine Seitenanzahl und ein Erscheinungsjahr. Die Exemplare eines bestimmten Buches sind fortlaufend nummeriert und weiterhin durch eine Inventarnummer und den Standort charakterisiert. Für jedes Buch wird vermerkt, welche Exemplare dieses Buches in der Bibliothek vorhanden sind. Bücher sind in einem Verlag erschienen, von dem der Name und der Verlagsort registriert wird. Für jeden Bibliotheksbenutzer ist der Name, Vorname, Wohnort und das Geburtsdatum gespeichert. Benutzer können einzelne Buchexemplare ausleihen, wobei jeweils das Datum der Ausleihe registriert wird. Weiterhin können Benutzer einzelne Bücher vorbestellen, wobei auch hier das Datum der Vorbestellung registriert wird.

Geben Sie ein Entity-Relationship-Modell für den oben beschriebenen Sachverhalt an! Geben Sie auch Kardinalitäten und Schlüsselattribute an!

Aufgabe 9.3. Die drei Schlagzeuger John Bonham, Keith Moon und Neil Peart treffen sich zu einer Jamsession. Alle drei haben jeweils ihr Schlagzeug dabei, doch nur Neil hat auch an seine beiden Drumsticks gedacht. Immerhin hat der zerstreute John noch einen weiteren Drumstick dabei, der cholerische Keith hat seine im Wutanfall zerstört. Insgesamt haben die drei also nur drei Drumsticks. Trotzdem will keiner komplett aufs Schlagzeugspielen verzichten. Sie bauen ihre Schlagzeuge im Kreis auf und setzen sich jeweils hinter ihr eigenes; jeder von ihnen sitzt nun neben den beiden anderen. Die drei Drumsticks werden so zwischen die Schlagzeuge gelegt, dass links und rechts von jedem Schlagzeuger genau ein Drumstick liegt. Will einer der drei sein Schlagzeug spielen, so muss er den Drumstick rechts *und* den Drumstick links dazu nehmen. Hört er auf zu spielen, so legt er die Drumsticks wieder an ihren Platz zurück. Sind für einen Schlagzeuger nicht *beide* Drumsticks, links und rechts, verfügbar, so spielt er sein Schlagzeug nicht (ein echter Drummer spielt nicht mit nur einem Stick!) sondern hält sich stattdessen die Ohren zu.

Modellieren Sie diesen Konflikt um die Drumsticks und die möglichen Abläufe (wer spielt wann sein Schlagzeug) durch ein Petri-Netz. Nutzen Sie für jeden Drumstick eine Stelle, die markiert ist, wenn der entsprechende Stick verfügbar ist. Nutzen Sie außerdem für jeden der drei Schlagzeuger eine oder mehrere Stellen, deren Markierung angibt, ob der Schlagzeuger gerade sein Schlagzeug spielt oder sich die Ohren zuhält. Verbinden Sie die von Ihnen genutzten Stellen durch Transitionen und wählen Sie eine passende Startmarkierung so, dass alle erreichbaren Markierungen Ihres Petri-Netzes einer möglichen Situation in der Jamsession entsprechen und andererseits auch jede mögliche Situation, die in der Jamsession auftreten kann durch eine erreichbare Markierung Ihres Petri-Netzes repräsentiert wird. Wählen Sie aussagekräftige Bezeichnungen für Ihre Stellen und Transitionen wie in Beispiel (9.7), so dass für jede Stelle und Transition klar wird, welche Situation bzw. Aktion sie beschreibt.

Achtung: Es ist nicht gefordert, dass die Schlagzeuger in irgendeiner festen Reihenfolge spielen sollen, auch müssen keine Fairness-Kriterien gelten. So ist es durchaus möglich, dass ein Schlagzeuger, nachdem er die Drumsticks abgelegt hat, diese gleich wieder aufnimmt und weiterspielt, ohne, dass einer der beiden anderen zum Zuge kam.

Aufgabe 9.4. In dieser Aufgabe soll ein ER-Modell für die Verwaltung der Uni-Sporthalle erstellt werden. Jeder einzelne Sportkurs kann über seine Anfangszeit und den Wochentag, an dem er stattfindet identifiziert werden. Darüber hinaus soll die Anzahl der Teilnehmer bekannt sein. In jedem einzelnen Sportkurs wird natürlich nur eine bestimmte Sportart ausgeübt. Und für jeden Kurs einer Sportart soll immer Material in einer bestimmten Anzahl zur Verfügung stehen (bspw. für jeden Basketballkurs immer zwölf Basketbälle, für jeden Badmintonkurs ein

Netz und acht Federbälle).

Die Kurse selbst sind entweder individuelle Kurse, d. h. die Teilnehmer treffen sich ohne Kursleiter, oder unter Anleitung genau eines Kursleiters aus dem Mitarbeiterpool. Jeder einzelne Mitarbeiter soll hierbei mindestens drei und maximal fünf Kurse leiten. Für jede Sportart zeichnet sich auch immer genau ein Mitarbeiter verantwortlich, wobei ein Mitarbeiter für beliebig viele Sportarten als Verantwortlicher zur Verfügung stehen kann.

Jeder einzelne Mitarbeiter kann mit seiner Personalnummer identifiziert werden. Weiterhin wird sein Name hinterlegt und das Datum seiner Anstellung, um jederzeit seine Erfahrung im Uni-Sporthallenalltag einschätzen zu können. Schließlich gibt es unter den Mitarbeitern, wie so oft, eine Hierarchie, wobei jeder Mitarbeiter höchstens einen Vorgesetzten hat.

Modellieren Sie den oben angegebenen Sachverhalt durch ein ER-Diagramm. Benutzen Sie dafür genau die Entitäten *Kurs*, *Sportart*, *Equipment* und *Personal*. Geben Sie die entsprechenden Beziehungen wie auch die Kardinalitäten, die modellierten Attribute, Schlüsselattribute und ggf. zusätzlich getroffene Entscheidungen bei der Modellierung an.

10 Beispielklausuren

Am Ende des Semesters findet die Modulabschlussprüfung Diskrete Modellierung in Form einer Klausur statt. Die schriftliche Prüfung dauert 120 Minuten.

Details zum Ablauf der Klausur:

- Grundsätzlich gelten die in der Ordnung Ihres Studiengangs festgelegten Regelungen. Dieses hier sind nur ergänzende Hinweise.
- Alle Klausurteilnehmer/innen müssen sich durch (1) den Studierendenausweis und (2) die „Goethe-Card“ oder einen Lichtbildausweis ausweisen.
- Die Sitzordnung wird von uns festgelegt und kurz vor Beginn der Klausur bekannt gegeben.
- Außer einem dokumentenechten Schreibstift sind keine weiteren Hilfsmittel zugelassen. (Insbesondere: Kein Vorlesungsskript, keine mitgebrachten Notizen, kein von Ihnen mitgebrachtes Papier, kein Taschenrechner, kein Handy. Bitte beachten Sie, dass ein während der Klausur eingeschaltetes Handy als Betrugsversuch gewertet wird.)
- Schreibpapier wird von uns bereitgestellt. Für jede Lösung steht eine bestimmte Lösungsbox zur Verfügung. Gegebenenfalls können auch die beigelegten Zusatzblätter benutzt. Weitere Blätter sind auf Nachfrage erhältlich.
- Begründungen sind nur dann notwendig, wenn die Aufgabenformulierung dies verlangt.
- Jedes Blatt der abgegebenen Lösung muss mit Namen, Vornamen und Matrikelnummer gekennzeichnet sein; andernfalls werden diese Blätter nicht gewertet.
- Werden zu einer Aufgabe zwei oder mehr Lösungen angegeben, so gilt die Aufgabe als nicht gelöst. Entscheiden Sie sich also immer für eine Lösung.

Checkliste - zur Klausur müssen Sie mitbringen:

- einen dokumentenechten Schreibstift
- einen gültigen Lichtbildausweis (z.B. Ihre Goethe-Card oder Ihren Personalausweis)
- Ihren Studierendenausweis (. . . es sei denn, Sie sind als „Schülerstudent“ für die Veranstaltung angemeldet)

Durch die in den Übungen gesammelten Punkte kann ein Bonus für die Klausur erworben werden. Zur Benotung wird die Summe aus dem Klausurergebnis und der Bonuspunkte verwendet. Die Klausur ist bestanden, wenn mit dem Bonus mindestens 50% der in der Klausur erzielbaren Punkte erreicht werden.

Nachfolgend finden Sie die Aufgaben der Klausuren aus den Wintersemestern 2008/09 und 2011/12 sowie den Sommersemestern 2009 und 2011.

Aufgaben der Klausur WS 11/12

Aufgabe 1:**(17 Punkte)**

- (a) ¹ Paul fühlt sich nicht so gut und geht zum Arzt. Nach eingehender Untersuchung kommen für den Mediziner nur vier Krankheiten in Frage. Möglicherweise leidet Paul unter Hirnversalzung oder Denkinsuffizienz. Gummikauzwang oder eine Sitzanomalie kommen aber auch in Frage. Mit dem Wissen des Facharztes über die Wechselwirkung der Krankheiten steht für die Diagnose nun Folgendes fest. (10 Pkte)

Fakt 1: Falls Paul keinen Gummikauzwang hat, dann leidet er unter Hirnversalzung oder Denkinsuffizienz oder gar beidem.

Fakt 2: Falls Paul eine Sitzanomalie hat, dann hat er auch Gummikauzwang und Hirnversalzung.

Fakt 3: Hat Paul jedoch weder eine Sitzanomalie noch Hirnversalzung, dann kann er auch keine Denkinsuffizienz haben.

Paul ist verwirrt. Welche Krankheiten hat er wohl — oder hält ihn der Heilkundige gar für einen Simulanten ?

Formalisieren Sie die drei Aussagen durch je eine aussagenlogische Formel, indem Sie die atomaren Aussagen D (Paul leidet unter Denkinsuffizienz), G (Paul leidet unter Gummikauzwang), H (Paul leidet unter Hirnversalzung) und S (Paul leidet unter einer Sitzanomalie) benutzen.

$$\varphi_{\text{Fakt 1}} :=$$

$$\varphi_{\text{Fakt 2}} :=$$

$$\varphi_{\text{Fakt 3}} :=$$

Stellen Sie eine aussagenlogische Formel ψ auf, die das Wissen um Pauls gesundheitlichen Zustand widerspiegelt.

$$\psi :=$$

¹Die Idee der Aufgabe beruht auf einer Aufgabe, die Dr. Peter Rossmannith und Dr. Clemens Ballarin im Sommersemester 2002 im Rahmen einer Logikvorlesung an der Technischen Universität München stellten.

Mit dem Blick in die Krankenakte ist klar. Paul hatte als Kind bereits einmal Hirnversalzung und wurde erfolgreich diesbezüglich behandelt. Wie jeder weiß, kann man Hirnversalzung nicht ein zweites Mal bekommen, d.h. es gilt zusätzlich der Fakt, dass Paul nicht unter Hirnversalzung leiden kann. Unter Einbeziehung des neuen Wissens stellt sich nun die Frage: Unter welchen Krankheiten leidet Paul und unter welchen leidet er nicht? Zeigen Sie, dass Ihre Antwort korrekt ist.

(b) Welche der folgenden Aussagen sind wahr? Welche nicht? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Jede aussagenlogische Variable ist auch eine atomare Formel aus AL. wahr falsch

Zu jeder Formel $\varphi \in \text{AL}$ existiert eine Belegung \mathcal{B} , so dass $\llbracket \varphi \rrbracket^{\mathcal{B}} = 1$. wahr falsch

Da man für φ eine zu φ äquivalente Formel in DNF aus der Wahrheitstafel erhält, indem man die erfüllenden Belegungen betrachtet, kann es wahr falsch zu φ , falls φ unerfüllbar ist, keine äquivalente Formel in DNF geben.

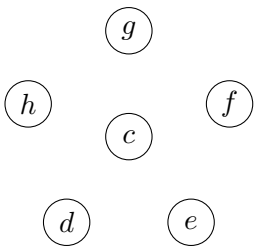
(c) Geben Sie für die aussagenlogische Formel $\varphi := \neg((X_1 \wedge X_2) \rightarrow (\mathbf{1} \leftrightarrow X_3))$ eine zu φ äquivalente aussagenlogische Formel ψ in NNF an. Geben Sie auch Ihren Lösungsweg an. (4 Pkte)

Aufgabe 2:**(23 Punkte)**(a) Sei $G = (V, E)$ der *ungerichtete* Graph mit der folgenden Adjazenzmatrix:

	c	d	e	f	g	h
c	0	1	1	1	1	0
d	1	0	1	0	0	0
e	1	1	0	0	0	0
f	1	0	0	0	0	0
g	1	0	0	0	0	1
h	0	0	0	0	1	0

(i) Geben Sie die graphische Darstellung von G an. Geben Sie außerdem die Knotenmenge V sowie die Kantenmenge E von G an und repräsentieren Sie G durch eine Adjazenzliste.

graphische Darstellung: (1 Pkt)



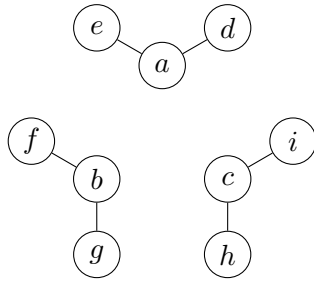
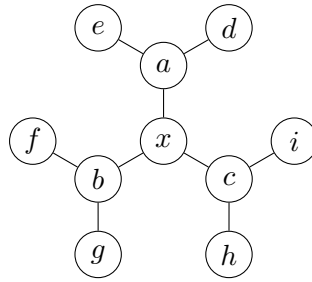
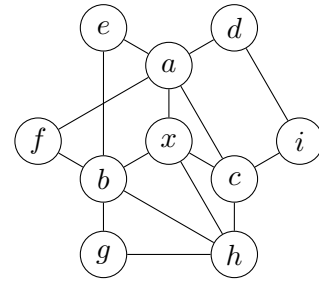
$V =$ (1 Pkt)

$E =$ (1 Pkt)

Adjazenzliste: (1 Pkt)

(ii) Geben Sie $\text{Grad}(G)$ an: (1 Pkt)(iii) Geben Sie eine Kante an, nach deren Wegnahme aus G der entstehende Graph ein Baum ist: (1 Pkt)(iv) Geben Sie eine Kante an, die zu G hinzugefügt werden muss, damit der entstehende Graph einen Eulerkreis enthält: (1 Pkt)

(b) Es seien die folgenden ungerichteten Graphen G_1 , G_2 und G_3 gegeben:

 G_1  G_2  G_3

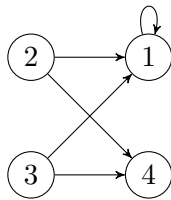
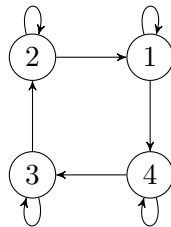
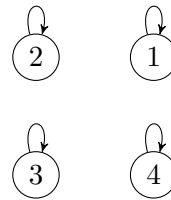
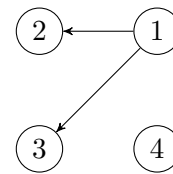
Welche der folgenden Aussagen sind wahr, welche falsch?

(3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- G_1 ist ein induzierter Teilgraph von G_2 . wahr falsch
- Jedes Matching in G_2 besteht aus höchstens 3 Kanten. wahr falsch
- G_2 ist ein Spannbaum von G_3 wahr falsch

(c) Betrachten Sie die folgenden vier gerichteten Graphen mit der Knotenmenge $V = \{1, 2, 3, 4\}$ und fassen Sie diese als 2-stellige Relationen R_1 , R_2 , R_3 und R_4 über $V \times V$ auf.

 R_1  R_2  R_3  R_4

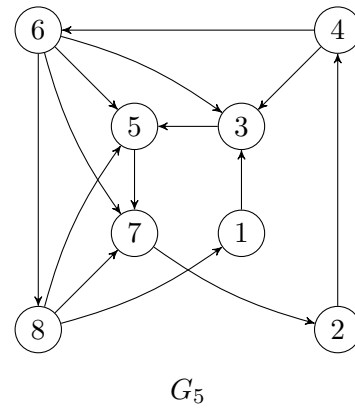
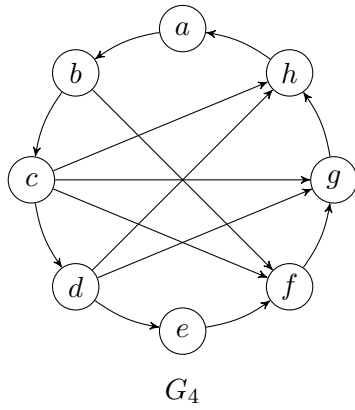
Welche der folgenden Aussagen sind wahr, welche falsch?

(4 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- R_1 ist antisymmetrisch. wahr falsch
- R_2 ist konnex. wahr falsch
- R_3 ist eine Äquivalenzrelation. wahr falsch
- R_4 ist eine partielle Funktion. wahr falsch

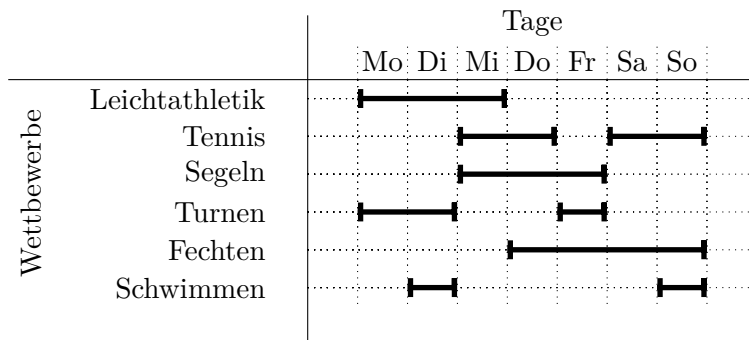
(d) Betrachten Sie die folgenden gerichteten Graphen G_4 und G_5 :



(i) G_4 und G_5 sind isomorph. Geben Sie einen Isomorphismus von G_4 nach G_5 an. (2 Pkte)

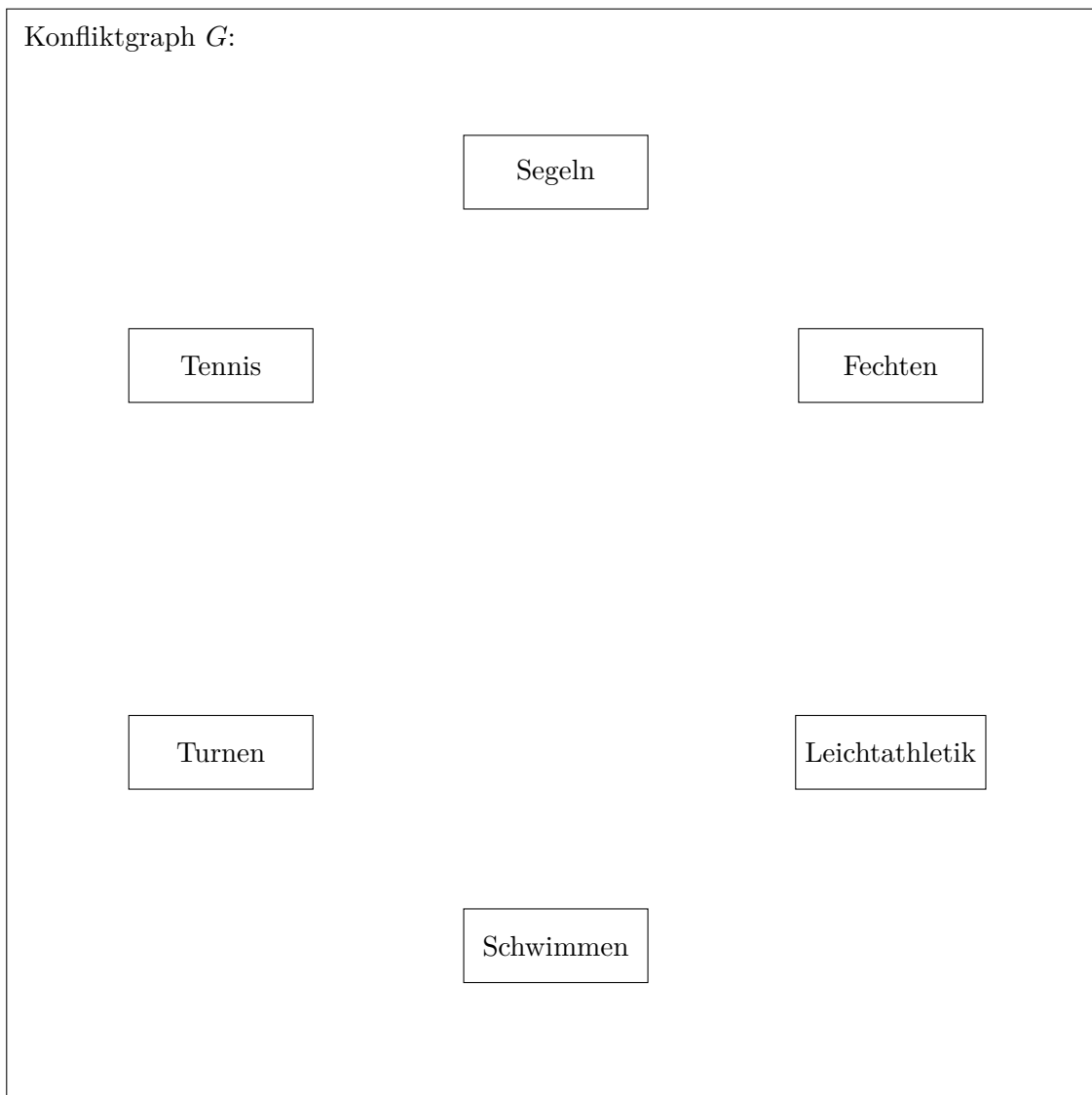
(ii) Ist der Graph G_5 stark zusammenhängend? Begründen Sie Ihre Antwort. (1 Pkt)

(e) In wenigen Monaten beginnen die Olympischen Sommerspiele 2012 in London. Die kleine Radiostation KBBL möchte gerne über die folgenden Wettbewerbe berichten: Leichtathletik, Tennis, Segeln, Turnen, Fechten und Schwimmen. Zu jedem dieser Wettbewerbe muss ein Reporter geschickt werden. Natürlich will KBBL Geld sparen und deshalb so wenige Reporter wie möglich einsetzen. Allerdings kann ein Reporter an einem Tag nur von einem Wettbewerb berichten. Also müssen zu Wettbewerben, die am gleichen Tag stattfinden, verschiedene Reporter geschickt werden. Bei den Wettbewerben gibt es folgende zeitliche Überschneidungen:



So können beispielsweise die Wettbewerbe im Tennis und Turnen vom gleichen Reporter besucht werden, weil sie an verschiedenen Tagen stattfinden. Im Gegensatz dazu benötigen die Wettbewerbe im Segeln und Turnen verschiedene Reporter, da sie sich zeitlich überschneiden.

(i) Stellen Sie den Konfliktgraphen G auf, in dem eine Kante zwischen zwei Knoten dafür steht, dass die entsprechenden Wettkämpfe sich zeitlich überschneiden. (1 Pkt)



- (ii) Sei V die Menge der Knoten des Konfliktgraphen G aus Teilaufgabe (i), es gelte also $V := \{\text{Leichtathletik, Tennis, Segeln, Turnen, Fechten, Schwimmen}\}$. Geben Sie eine konfliktfreie Knotenmarkierung $m: V \rightarrow \mathbb{N}$ an, die möglichst wenige verschiedene Markierungen verwendet, d.h. $|\text{Bild}(m)|$ soll minimal sein. (2 Pkte)

- (iii) Geben Sie eine Aufteilung aller sechs Wettbewerbe auf möglichst wenige verschiedene Reporter an, so dass kein Reporter zwei Wettbewerben zugewiesen wird, die sich zeitlich überschneiden. (1 Pkt)

- (iv) Begründen Sie, warum die von Ihnen in Teilaufgabe (iii) benutzte Anzahl von Reportern bestmöglich ist, d.h. warum es keine konfliktfreie Aufteilung der Wettbewerbe auf eine geringere Anzahl von Reportern geben kann. (2 Pkte)

Aufgabe 3:**(17 Punkte)**

- (a) Betrachten Sie die folgende Datenbank $\mathfrak{A}_{\text{Musik}}$, bestehend aus den dargestellten Tabellen $\text{Stück}^{\mathfrak{A}_{\text{Musik}}}$ und $\prec^{\mathfrak{A}_{\text{Musik}}}$.

Tabelle: $\text{Stück}^{\mathfrak{A}_{\text{Musik}}}$

Titel	Interpret	Jahr
Das Model	Kraftwerk	1978
Das Model	Rammstein	1997
Die Roboter	Fleischmann	1994
Die Roboter	Kraftwerk	1978
Fleischwolf	Fleischmann	1993
In-A-Gadda-Da-Vida	Iron Butterfly	1968
Stairway to Heaven	Iron Maiden	1981
Stairway to Heaven	Led Zeppelin	1970
Sweet Dreams	Marilyn Manson	1996
Sweet Dreams	Eurythmics	1983
Sweet Dreams	Eurythmics	1991

Tabelle: $\prec^{\mathfrak{A}_{\text{Musik}}}$

a	b
1900	1901
1900	1902
\vdots	\vdots
1900	2100
1901	1902
\vdots	\vdots
2097	2100
2098	2099
2098	2100
2099	2100

Hierbei ist die Tabelle $\text{Stück}^{\mathfrak{A}_{\text{Musik}}}$ unserer Datenbankinstanz ein Ausschnitt aus einer (denkbaren) Datenbanktabelle, welche alle zwischen 1900 und heute veröffentlichten Musikstücke enthält. Die Tabelle $\prec^{\mathfrak{A}_{\text{Musik}}}$ enthält diejenigen Paare von Wörtern aus ASCII*, welche wir als Paare von Jahreszahlen zwischen 1900 und 2100 interpretieren, falls die erste Komponente des Tupels vor der zweiten liegt. $\prec^{\mathfrak{A}_{\text{Musik}}}$ gibt uns also eine Reihenfolge auf den Worten, welche wir als Jahreszahlen zwischen 1900 und 2100 betrachten.

Die Signatur σ_{Musik} besteht aus einem 3-stelligen Relationssymbol Stück , einem 2-stelligen Relationssymbol \prec , sowie je einem Konstantensymbol 'c' für jeden möglichen Eintrag $c \in \text{ASCII}^*$ in der Datenbank.

- (i) Geben Sie eine FO[σ_{Musik}]-Anfrage φ an, welche alle Interpreten ausgibt, die den Song „Sweet Dreams“ veröffentlichten. (2 Pkte)

Berechnen Sie das Ergebnis Ihrer Anfrage auf der angegebenen Datenbankinstanz $\mathfrak{A}_{\text{Musik}}$. (1 Pkt)

- (ii) Ein Song gilt als *Cover-Version*, wenn es einen anderen Song gleichen Titels aus einem früheren Jahr in der Datenbank gibt. Geben Sie eine FO[σ_{Musik}]-Anfrage φ an, welche die Titel aller *Cover-Versionen* und die Interpreten der jeweiligen *Cover-Version* ausgibt. (2 Pkte)

Berechnen Sie das Ergebnis Ihrer Anfrage auf der angegebenen Datenbankinstanz $\mathfrak{A}_{\text{Musik}}$. (1 Pkt)

- (iii) Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0. (3 Pkte)

Es gibt eine FO[σ_{Musik}]-Anfrage φ , so dass die von φ in $\mathfrak{A}_{\text{Musik}}$ definierte Antwortrelation

... alle Titel enthält, die vor dem Jahr 1999 veröffentlicht wurden. wahr falsch

... zu jeder *Cover-Version* neben Titel und Interpreten auch den Interpreten des Originals enthält. wahr falsch

... alle Worte aus ASCII* ausgibt, welche als Bandnamen noch frei (also bisher als Interpret in der Datenbank unbenutzt) sind. wahr falsch

(b) Welche der folgenden Aussagen sind wahr, welche sind falsch? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Die Menge der aussagenlogischen Formeln ist eine Teilmenge der Menge der Formeln der Logik erster Stufe. wahr falsch

Zu jeder σ -Struktur \mathfrak{A} existiert ein $\text{FO}[\sigma]$ -Satz, welcher von \mathfrak{A} erfüllt wird. wahr falsch

Jeder σ -Term ist auch eine atomare $\text{FO}[\sigma]$ -Formel. wahr falsch

(c) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem zweistelligen Relationssymbol \dot{E} . Betrachten Sie die σ -Struktur $\mathfrak{A} = (A, \dot{E}^{\mathfrak{A}})$, welche durch die unten angegebene Abbildung repräsentiert wird. Geben Sie einen $\text{FO}[\sigma]$ -Satz φ an, der die Struktur eindeutig beschreibt. D.h. für alle σ -Strukturen \mathfrak{B} soll gelten: $\mathfrak{A} \cong \mathfrak{B} \Leftrightarrow \mathfrak{B} \models \varphi$. (5 Pkte)



$\varphi :=$

Aufgabe 4:**(19 Punkte)**

- (a) Sei Σ ein endliches Alphabet mit $\Sigma := \{V, 1, 2, 3, 4, 5, 6, 7, 8, 9, \neg, \wedge, \vee, \mathbf{1}, \mathbf{0}, \equiv, \neq\}$. R sei als folgender regulärer Ausdruck über dem Alphabet Σ gegeben:

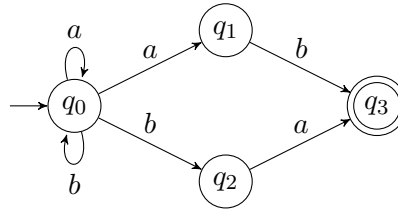
$$(\neg|\varepsilon) V(1|2|\dots|9) \left((\wedge|\vee) (\neg|\varepsilon) V(1|2|\dots|9) \right)^* (\equiv|\neq) (\mathbf{1}|\mathbf{0})$$

- (i) Welche der folgenden Wörter liegen in der von R definierten Sprache $L(R)$, welche nicht? (1,5 Pkte)
 Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen halben Punkt, für jedes **falsche Kreuz** wird **ein halber Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in $L(R)$?	
$V_1 \wedge \neg V_1 \equiv \mathbf{1}$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$V_8 \vee \neg V_9 \vee \neg V_{10} \neq \mathbf{0}$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$V_3 \wedge V_4 \vee V_3 \wedge V_4 \vee V_3 \wedge V_4 \vee V_3 \equiv \mathbf{0}$	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (ii) Geben Sie einen deterministischen endlichen Automaten B in graphischer Darstellung (2,5 Pkte) an, der genau die Sprache akzeptiert, die vom regulären Ausdruck R definiert wird, d.h. es soll gelten, dass $L(B) = L(R)$.
 Beachten Sie, dass der Automat B nicht vollständig sein muss.

(b) Sei A der folgende nichtdeterministische endliche Automat über dem Alphabet $\Sigma := \{a, b\}$:



(i) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(A)$ an, die vom Automaten A akzeptiert wird. (2 Pkte)

(ii) Wandeln Sie den NFA A mit Hilfe der Potenzmengenkonstruktion in einen DFA A' um. Berücksichtigen Sie dabei nur solche Zustände von A' , die vom Startzustand von A' aus erreicht werden können und geben Sie die graphische Darstellung von A' an. (4 Pkte)

- (c) Geben Sie für jede der folgenden beiden Sprachen L_1 und L_2 jeweils an, ob es sich um reguläre Sprachen handelt. (Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl dieser Teilaufgabe ist aber mindestens 0.) (9 Pkte)

Beweisen Sie, dass Ihre jeweilige Antwort korrekt ist.

Zur Erinnerung:

Um zu beweisen, dass eine Sprache L regulär ist, genügt es, einen endlichen Automaten A oder einen regulären Ausdruck R anzugeben, so dass $L = L(A)$ bzw. $L = L(R)$ ist.

Um zu beweisen, dass eine Sprache *nicht* regulär ist, können Sie das Pumping-Lemma aus der Vorlesung benutzen:

Sei Σ ein endliches Alphabet. Für jede *reguläre* Sprache $L \subseteq \Sigma^*$ gibt es eine Zahl $z \in \mathbb{N}$, so dass für jedes Wort $x \in L$ der Länge $|x| \geq z$ gilt: Es gibt eine Zerlegung von x in Worte $u, v, w \in \Sigma^*$, so dass die folgenden vier Bedingungen erfüllt sind:

- (1) $x = uvw$
- (2) $|uv| \leq z$
- (3) $|v| \geq 1$
- (4) für jedes $i \in \mathbb{N}$ gilt: $uv^i w \in L$.
(d.h.: $uv \in L, uvw \in L, uvvw \in L, uvvww \in L, \dots$)

$$L_1 := \{a^m b^n : m, n \in \mathbb{N}, 3 > m, m < n\}$$

regulär: ja nein

Beweis:

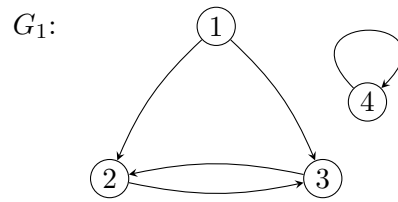
$$L_2 := \{a^m b^n : m, n \in \mathbb{N}, 3 < m, m < n\}$$

regulär: ja nein

Beweis:

Aufgabe 5:**(8 Punkte)**

Betrachten Sie den folgenden Web-Graph G_1 , der aus den vier Webseiten 1, 2, 3 und 4 besteht, die entsprechend der gerichteten Kanten in der Abbildung untereinander verlinkt sind.



- (a) Stellen Sie für G_1 und den Dämpfungsfaktor $d = \frac{1}{2}$ die Page-Rank-Matrix $P(G_1, d)$ auf. (4 Pkte)

- (b) Es seien die vier Page-Ranks $PR_1 = \frac{1}{8}$, $PR_2 = PR_3 = \frac{5}{16}$ und $PR_4 = \frac{1}{4}$ gegeben. Weisen Sie nach, dass dies die Page-Ranks für die vier Webseiten 1, 2, 3 und 4 des Web-Graphen G_1 bezüglich des Dämpfungsfaktors $d = \frac{1}{2}$ sind. Das heißt, weisen Sie nach, dass das Tupel $PR = (\frac{1}{8}, \frac{5}{16}, \frac{5}{16}, \frac{1}{4})$ die Page-Rank-Eigenschaft bezüglich $d = \frac{1}{2}$ hat.



Aufgabe 6:**(16 Punkte)**

Im Folgenden wird zu jeder Zahl $n \in \mathbb{N}$ die Sprache SET_n über dem Alphabet $\Sigma := \{\cup, \cap, (\cdot)\} \cup \{\{\} \cup \{\}\} \cup \{i : i < n, i \in \mathbb{N}\}$ rekursiv definiert:

Basisregel:

(B) Für jede Zahl $i < n$ mit $i \in \mathbb{N}$ ist das Wort $\{i\}$ in SET_n .

Rekursive Regeln:

(R1) Sind w_1 und w_2 in SET_n , so ist auch $(w_1 \cap w_2)$ in SET_n .

(R2) Sind w_1 und w_2 in SET_n , so ist auch $(w_1 \cup w_2)$ in SET_n .

(a) Welche der folgenden Wörter gehören zur Sprache SET_{42} , welche nicht? (4 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in SET_{42} ?	
\emptyset	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$\{0\}$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(\{42\} \cup (\{4\} \cap \{2\}))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(\{4, 2\} \cup (\{4\} \cap \{2\}))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein

(b) Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = SET_5$. (4 Pkte)

- (c) Betrachten Sie die von Ihnen in (b) angegebene kontextfreie Grammatik G und geben Sie (2 Pkte) einen Ableitungsbaum für das folgende Wort an:

$$(\{1\} \cup (\{4\} \cap \{2\}))$$

Zur Erinnerung wird die bereits zu Beginn der Aufgabe angegebene rekursive Definition der Sprache SET_n über dem Alphabet $\Sigma := \{\cup, \cap, (,)\} \cup \{\{\}\} \cup \{i : i < n, i \in \mathbb{N}\}$ wiederholt:

Basisregel:

(B) Für jede Zahl $i < n$ mit $i \in \mathbb{N}$ ist das Wort $\{i\}$ in SET_n .

Rekursive Regeln:

(R1) Sind w_1 und w_2 in SET_n , so ist auch $(w_1 \cap w_2)$ in SET_n .

(R2) Sind w_1 und w_2 in SET_n , so ist auch $(w_1 \cup w_2)$ in SET_n .

Für jedes $n \in \mathbb{N}$ und jedes Wort w aus SET_n ist die Funktion $f_n : SET_n \rightarrow \mathbb{N}$ wie folgt definiert:

$$\begin{aligned} f_n(w) &:= 1 && \text{für } w = \{i\} && \text{mit } 0 \leq i < n, i \in \mathbb{N} \\ f_n(w) &:= 0 && \text{für } w = (w_1 \cap w_2) && \text{mit } w_1, w_2 \in SET_n \\ f_n(w) &:= f_n(w_1) + f_n(w_2) - \min(f_n(w_1), f_n(w_2)) && \text{für } w = (w_1 \cup w_2) && \text{mit } w_1, w_2 \in SET_n \end{aligned}$$

Für $n > 4$ gilt so beispielsweise $f_n(\{4\} \cap \{2\}) = 0$ und $f_n(\{4\} \cup \{2\}) = 1$.

Die Funktion $|w|_{SET_n}$ berechnet die Kardinalität der Menge, welche man als Ergebnis erhält, wenn man $w \in SET_n$ als mengendarithmetischen Ausdruck interpretiert. Präzise heißt das: Für ein Wort $w \in SET_n$ definieren wir die Menge $M(w)$:

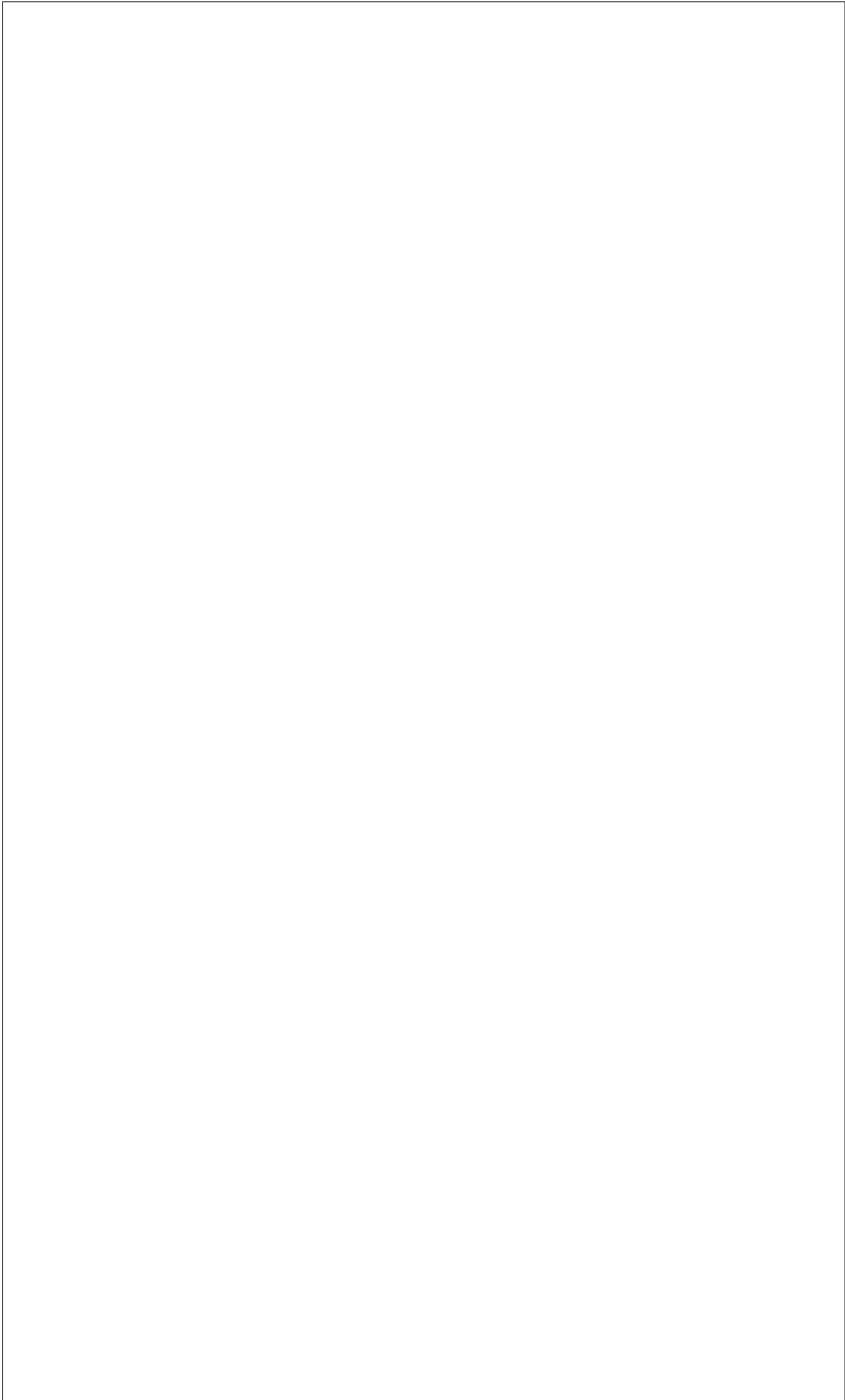
$$\begin{aligned} M(w) &:= \{i\} && \text{für } w = \{i\} && \text{mit } 0 \leq i < n, i \in \mathbb{N} \\ M(w) &:= M(w_1) \cap M(w_2) && \text{für } w = (w_1 \cap w_2) && \text{mit } w_1, w_2 \in SET_n \\ M(w) &:= M(w_1) \cup M(w_2) && \text{für } w = (w_1 \cup w_2) && \text{mit } w_1, w_2 \in SET_n \end{aligned}$$

Und wir setzen $|w|_{SET_n} := |M(w)|$.

Für $n > 4$ gilt so beispielsweise $|(\{4\} \cap \{2\})|_{SET_n} = 0$ und $|(\{4\} \cup \{2\})|_{SET_n} = 2$.

- (d) Sei $n \in \mathbb{N}$ beliebig gewählt. Beweisen Sie per Induktion nach dem Aufbau von SET_n , dass (6 Pkte)
für alle Wörter $w \in SET_n$ gilt:

$$|w|_{SET_n} \geq f_n(w)$$



Aufgaben der Klausur SoSe 2011

Aufgabe 1:**(20 Punkte)**

- (a) ¹ Nachdem Hänsel und Gretel die Hexe in den Ofen gestoßen haben, wollen sie sich über das Knusperhäuschen hermachen. Doch wie allgemein bekannt ist, muss man beim Verzehr eines solchen Hauses mit äußerster Vorsicht vorgehen, da dieses sonst zur Instabilität neigt. Die beiden wenden sich zunächst einer Wand zu, welche aus drei Lebkuchen besteht. Da Gretel erfolgreich Knusperhäuschenarchitektur studiert hat, kennt Sie die folgenden Sicherheitsregeln: (8 Pkte)

Regel 1: Von den beiden ersten Lebkuchen darf höchstens einer entfernt werden.

Regel 2: Wenn man den dritten entfernt, muss man auch den zweiten entfernen.

Regel 3: Wenn man den zweiten entfernt und den ersten nicht, dann darf man den dritten nicht entfernen.

Formalisieren Sie die drei Aussagen durch je eine aussagenlogische Formel, indem Sie die atomaren Aussagen E (die beiden entfernen den ersten Lebkuchen), Z (die beiden entfernen den zweiten Lebkuchen) und D (die beiden entfernen den dritten Lebkuchen) benutzen.

$\varphi_{\text{Regel 1}} :=$

$\varphi_{\text{Regel 2}} :=$

$\varphi_{\text{Regel 3}} :=$

Hänsel warnt Gretel davor, den dritten Lebkuchen zu entfernen. Ist seine Sorge berechtigt? Beweisen Sie, dass Ihre Aussage stimmt.

¹Diese Aufgabe ist dem Buch *Modellierung* von Uwe Kastens and Hans Kleine Büning entnommen.

- (b) Welche der folgenden Formeln ist in Negationsnormalform (NNF), disjunktiver Normalform (DNF) und/oder konjunktiver Normalform (KNF)? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen halben Punkt, für jedes **falsche Kreuz** wird **ein halber Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

	$\neg X_2$		$((X_1 \vee \neg X_2) \wedge \neg X_1) \vee (X_2 \wedge X_3)$	
in NNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein
in DNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein
in KNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (c) Welche der folgenden Aussagen sind für alle aussagenlogischen Formeln φ und ψ wahr? Welche nicht? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Jede atomare aussagenlogische Formel enthält mindestens eine aussagenlogische Variable. wahr falsch

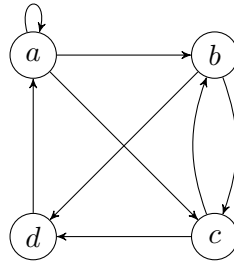
φ ist unerfüllbar gdw. $\neg\varphi$ erfüllbar ist. wahr falsch

Die Syntaxbäume zweier äquivalenter aussagenlogischer Formeln haben gleich viele Blätter. wahr falsch

- (d) Wann heißt eine aussagenlogische Formel allgemeingültig (bzw. Tautologie)? Geben Sie eine exakte Definition an! (2 Pkte)

Sei φ eine aussagenlogische Formel.
 φ heißt allgemeingültig (bzw. Tautologie), wenn

- (e) Sei $\varphi = ((X_1 \wedge X_2) \rightarrow (\neg X_2 \vee \neg X_1))$ und $\psi = (X_1 \vee X_2)$. Gilt $\varphi \models \psi$? Begründen Sie Ihre Antwort! (4 Pkte)

Aufgabe 2:**(23 Punkte)**(a) Sei $G = (V, E)$ der folgende *gerichtete* Graph:(i) Geben Sie die Knotenmenge V und die Kantenmenge E von G an. Repräsentieren Sie G außerdem durch eine Adjazenzmatrix. $V =$

(1 Pkt)

 $E =$

(1 Pkt)

Adjazenzmatrix:

(1 Pkt)

(ii) Geben Sie $\text{Ein-Grad}_G(a)$ an:

(1 Pkt)

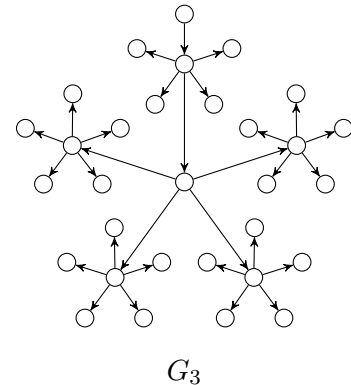
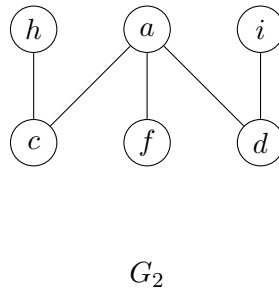
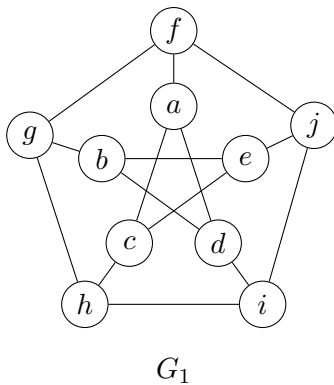
(iii) Geben Sie den längsten einfachen Weg vom Knoten b zum Knoten c an:

(1 Pkt)

(iv) Geben Sie die Kante an, die entfernt werden muss, damit der entstehende Graph nicht mehr stark zusammenhängend ist:

(1 Pkt)

- (b) Es seien die folgenden ungerichteten Graphen G_1 und G_2 sowie der gerichtete Graph G_3 gegeben:



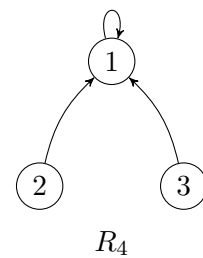
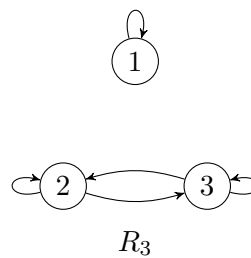
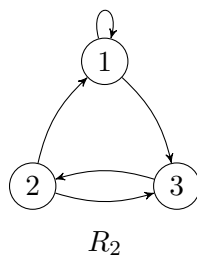
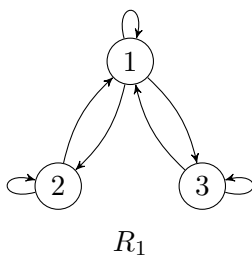
Welche der folgenden Aussagen sind wahr, welche falsch?

(3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- G_2 ist ein induzierter Teilgraph von G_1 . wahr falsch
- Die Knoten von G_2 lassen sich konfliktfrei mit zwei verschiedenen Farben färben. wahr falsch
- G_3 ist ein gerichteter Baum der Höhe 4. wahr falsch

- (c) Betrachten Sie die folgenden vier Graphen mit der Knotenmenge $V = \{1, 2, 3\}$ und fassen Sie diese als 2-stellige Relationen R_1, R_2, R_3 und R_4 über $V \times V$ auf.



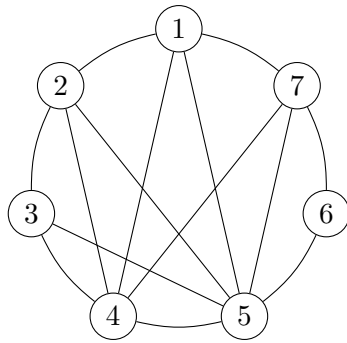
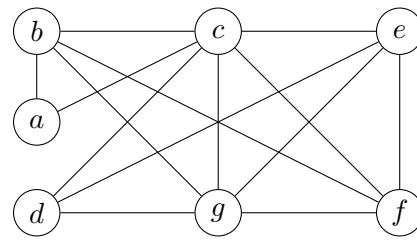
Welche der folgenden Aussagen sind wahr, welche falsch?

(4 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- R_1 ist konnex. wahr falsch
- R_2 ist transitiv. wahr falsch
- R_3 ist eine Äquivalenzrelation. wahr falsch
- R_4 ist eine Funktion. wahr falsch

(d) Betrachten Sie die folgenden ungerichteten Graphen G_4 und G_5 :

 G_4  G_5

(i) G_4 und G_5 sind isomorph. Geben Sie einen Isomorphismus von G_4 nach G_5 an. (2 Pkte)

(ii) Enthält der Graph G_5 einen Hamilton-Kreis? Begründen Sie Ihre Antwort. (2 Pkte)

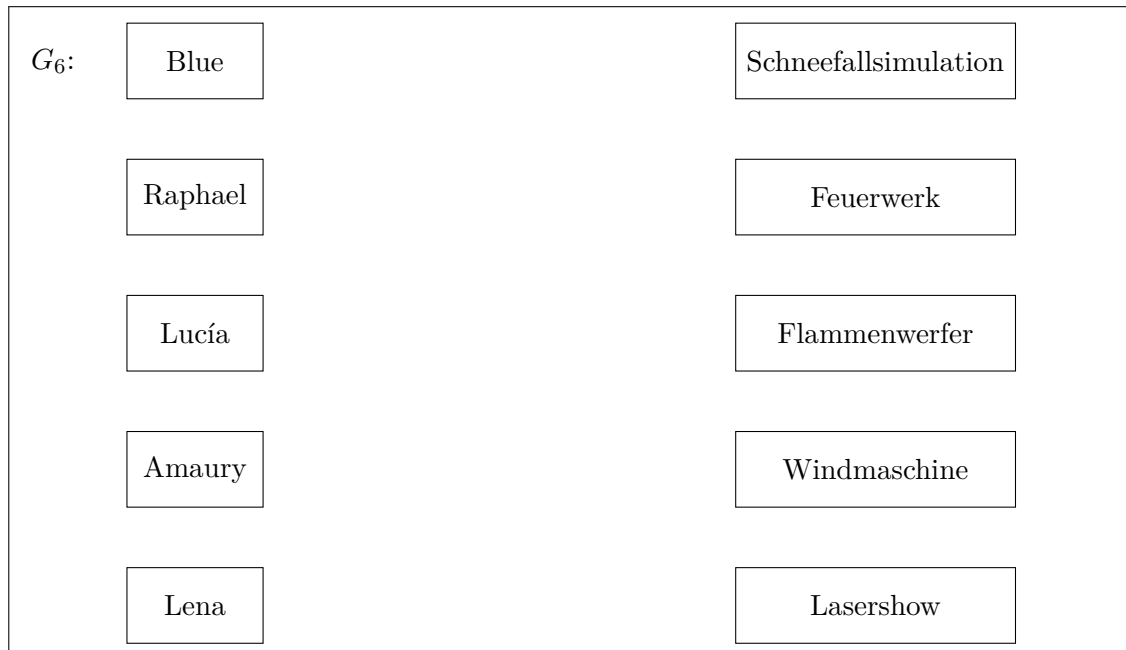
- (e) In weniger als zwei Wochen findet das Finale des 56. Eurovision Song Contest in Düsseldorf statt. Fünf Länder stehen bereits als Finalteilnehmer fest, auch die jeweiligen Musiker sind schon bekannt: Amaury Vassili geht für Frankreich, Raphael Gualazzi für Italien auf die Bühne, das Vereinigte Königreich wird von der Gruppe Blue, Spanien von Lucía Pérez vertreten, Deutschland schließlich hofft wieder einmal auf Lena.

Um die Show herauszuputzen, kann jeder der fünf Finalteilnehmer sich einen von fünf Bühneneffekten aussuchen. Zur Wahl stehen eine Lasershow, eine Windmaschine, Feuerwerk, ein Flammenwerfer und eine Schneefallsimulation. Um die Langeweile der Zuschauer zu begrenzen, darf jeder Bühneneffekt nur höchstens einmal benutzt werden.

Natürlich haben alle Musiker ganz eigene Vorlieben für bestimmte Bühneneffekte, die sich im Einzelnen folgendermaßen darstellen:

- Blue sind mit jedem Bühneneffekt zufrieden, der keine Lasershow ist.
- Raphael mag Feuerwerk, den Flammenwerfer und die Windmaschine.
- Auch Lucía steht auf Flammenwerfer und Windmaschine, außerdem auf die Lasershow.
- Amaury schwärmt für die Windmaschine und die Lasershow.
- Für Lena kommt nur die Windmaschine in Frage.

- (i) Stellen Sie den Graphen G_6 als *ungerichteten* Graphen auf, in dem jede Kante zwischen einem Musiker M und einem Bühneneffekt B verläuft und dafür steht, dass M mit B zufrieden wäre. (2 Pkte)



- (ii) Handelt es sich bei G_6 um einen bipartiten Graphen? Begründen Sie Ihre Antwort. (1 Pkt)

(iii) Geben Sie ein Matching maximaler Größe in G_6 an.

(2 Pkte)

Matching maximaler Größe in G_6 :

Blue	Schneefallsimulation
Raphael	Feuerwerk
Lucía	Flammenwerfer
Amaury	Windmaschine
Lena	Lasershow

(iv) Geben Sie eine Zuordnung zwischen Bühneneffekten und Musikern an, so dass jeder Musiker genau einen Effekt erhält, jeder Effekt genau einmal zugeordnet wird und alle Musiker mit ihrer Zuordnung zufrieden sind.

(1 Pkt)

Aufgabe 3:**(13 Punkte)**

- (a) Sei $\sigma = \{\mathit{Mult}, \mathit{N\ddot{a}ch}, \mathit{Eins}\}$ eine Signatur mit einem dreistelligen Relationssymbol Mult , einem einstelligen Funktionssymbol $\mathit{N\ddot{a}ch}$ und einem Konstantensymbol Eins . Wir betrachten die σ -Struktur $\mathfrak{A} := (\mathbb{N}, \mathit{Mult}^{\mathfrak{A}}, \mathit{N\ddot{a}ch}^{\mathfrak{A}}, \mathit{Eins}^{\mathfrak{A}})$ für die gilt:

$$\mathit{Mult}^{\mathfrak{A}} := \{(x, y, z) \mid x, y, z \in \mathbb{N}, x \cdot y = z\},$$

$$\mathit{N\ddot{a}ch}^{\mathfrak{A}} : \mathbb{N} \rightarrow \mathbb{N} \text{ mit } \mathit{N\ddot{a}ch}^{\mathfrak{A}}(x) = x + 1,$$

$$\mathit{Eins}^{\mathfrak{A}} := 1.$$

- (i) Geben Sie eine Formel φ der Logik erster Stufe über der Signatur σ an, die in der σ -Struktur $\mathfrak{A} := (\mathbb{N}, \mathit{Mult}^{\mathfrak{A}}, \mathit{N\ddot{a}ch}^{\mathfrak{A}}, \mathit{Eins}^{\mathfrak{A}})$ aussagt, dass für die modellierte Multiplikation das Kommutativgesetz gilt. (Zur Erinnerung: das Kommutativgesetz besagt, dass $n \cdot m = m \cdot n$ für alle natürliche Zahlen n und m gilt.) (2 Pkte)

- (ii) Geben Sie eine Formel $\varphi(x)$ der Logik erster Stufe über der Signatur σ an, die in der σ -Struktur $\mathfrak{A} := (\mathbb{N}, \mathit{Mult}^{\mathfrak{A}}, \mathit{N\ddot{a}ch}^{\mathfrak{A}}, \mathit{Eins}^{\mathfrak{A}})$ aussagt, dass die natürliche Zahl x eine Quadratzahl ist. (2 Pkte)

- (iii) Was sagt die Formel (2 Pkte)

$$\varphi(x) := \exists y \exists z (\mathit{N\ddot{a}ch}(\mathit{Eins}) = y \wedge \mathit{Mult}(z, y, x))$$

in der σ -Struktur $\mathfrak{A} := (\mathbb{N}, \mathit{Mult}^{\mathfrak{A}}, \mathit{N\ddot{a}ch}^{\mathfrak{A}}, \mathit{Eins}^{\mathfrak{A}})$ aus ?

- (b) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem zweistelligen Relationssymbol \dot{E} . (4 Pkte)

Geben Sie für die Formel

$$\varphi(x) := \forall y ((\neg x \dot{=} y) \rightarrow (\dot{E}(x, y) \wedge (\neg \exists z \dot{E}(y, z))))$$

eine σ -Struktur \mathfrak{A} und zwei Interpretationen $\mathcal{I}_1 = (\mathfrak{A}, \beta_1)$ und $\mathcal{I}_2 = (\mathfrak{A}, \beta_2)$ an, so dass gilt:

$$\mathcal{I}_1 \models \varphi \quad \text{und} \quad \mathcal{I}_2 \not\models \varphi.$$

- (c) Für eine Formel φ der Logik erster Stufe ist $\text{Var}(\varphi)$ die Menge aller Variablen, die in der Formel φ vorkommen und $\text{frei}(\varphi)$ die Menge aller Variablen, die mindestens einmal frei in φ vorkommen. Zusätzlich sei $\text{geb}(\varphi)$ die Menge aller Variablen, die mindestens einmal gebunden in φ vorkommen. (3 Pkte)

Welche der folgenden Aussagen für Formeln der Logik erster Stufe sind wahr, welche sind falsch?

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

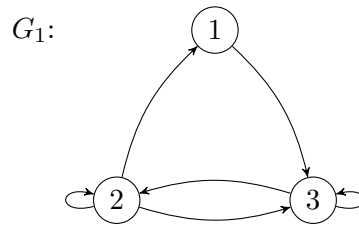
$\text{Var}(\varphi) = \text{frei}(\varphi) \cup \text{geb}(\varphi)$ wahr falsch

$\text{Var}(\varphi) \setminus \text{frei}(\varphi) = \text{geb}(\varphi)$ wahr falsch

Falls φ ein Satz ist, gilt: $\text{frei}(\varphi) = \text{Var}(\varphi) \setminus \text{geb}(\varphi)$ wahr falsch

Aufgabe 4:**(8 Punkte)**

Betrachten Sie den folgenden Web-Graph G_1 , der aus den drei Webseiten 1, 2 und 3 besteht, die entsprechend der gerichteten Kanten in der Abbildung untereinander verlinkt sind.



- (a) Stellen Sie für G_1 und den Dämpfungsfaktor $d = \frac{1}{2}$ die Page-Rank-Matrix $P(G_1, d)$ auf. (4 Pkte)

- (b) Es seien die drei Page-Ranks $PR_1 = \frac{2}{9}$, $PR_2 = \frac{1}{3}$ und $PR_3 = \frac{4}{9}$ gegeben. Weisen Sie nach, (4 Pkte) dass dies die Page-Ranks für die drei Webseiten 1, 2 und 3 des Web-Graphen G_1 bezüglich des Dämpfungsfaktors $d = \frac{1}{2}$ sind. Das heißt, weisen Sie nach, dass das Tupel $PR = (\frac{2}{9}, \frac{1}{3}, \frac{4}{9})$ die Page-Rank-Eigenschaft bezüglich $d = \frac{1}{2}$ hat.



Aufgabe 5:**(20 Punkte)**

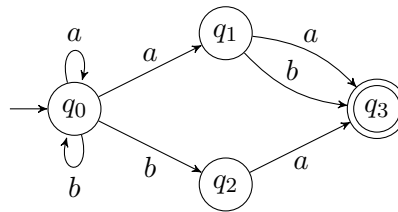
- (a) Geben Sie einen regulären Ausdruck R an, der die Sprache aller Wörter über dem Alphabet $\Sigma = \{\mathbf{a}, \mathbf{b}, \dots, \mathbf{z}, /, :, \dots\}$ definiert, die die URL einer Webseite beschreiben, welche die Form

$$\mathbf{http:// 3rd-level-label 2nd-level-label . top-level-domain}$$

haben. Dabei ist *3rd-level-label* entweder das Wort **www.** oder die leere Zeichenkette und *2nd-level-label* ein nicht-leeres Wort, in dem keines der Zeichen **/**, **:** und **.** (also nur **a, b, ..., z**) vorkommt. Die Zeichenkette *top-level-domain* kann entweder aus **de**, **com** oder **org** bestehen. Wörter dieser Sprache sind also z. B. **http://dismod.com** und **http://www.informatik.de**, aber *nicht* **www.uni.com** oder **http://de.wikipedia.org** oder **http://www..de**.

$R =$

- (b) Sei A der folgende nichtdeterministische endliche Automat über dem Alphabet $\Sigma := \{a, b\}$:



- (i) Welche der folgenden Wörter liegen in der von A akzeptierten Sprache $L(A)$, welche nicht? (3 Pkte)
 Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in $L(A)$?	
<i>bbbab</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>bbbabb</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>aaaaaaaaaaaa</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (ii) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(A)$ an, die vom Automaten A akzeptiert wird. (1,5 Pkte)

- (iii) Wandeln Sie den NFA A mit Hilfe der Potenzmengenkonstruktion in einen DFA A' um. (4 Pkte)
Berücksichtigen Sie dabei nur solche Zustände von A' , die vom Startzustand von A' aus erreicht werden können. Geben Sie die graphische Darstellung von A' an.



- (c) Geben Sie für jede der folgenden beiden Sprachen L_1 und L_2 jeweils an, ob es sich um reguläre Sprachen handelt. (Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl dieser Teilaufgabe ist aber mindestens 0.) (9 Pkte)

Beweisen Sie, dass Ihre jeweilige Antwort korrekt ist.

Zur Erinnerung:

Um zu beweisen, dass eine Sprache L regulär ist, genügt es, einen endlichen Automaten A oder einen regulären Ausdruck R anzugeben, so dass $L = L(A)$ bzw. $L = L(R)$ ist.

Um zu beweisen, dass eine Sprache *nicht* regulär ist, können Sie das Pumping-Lemma aus der Vorlesung benutzen:

Sei Σ ein endliches Alphabet. Für jede *reguläre* Sprache $L \subseteq \Sigma^*$ gibt es eine Zahl $z \in \mathbb{N}$, so dass für jedes Wort $x \in L$ der Länge $|x| \geq z$ gilt: Es gibt eine Zerlegung von x in Wörter $u, v, w \in \Sigma^*$, so dass die folgenden vier Bedingungen erfüllt sind:

- (1) $x = uvw$
- (2) $|uv| \leq z$
- (3) $|v| \geq 1$
- (4) für jedes $i \in \mathbb{N}$ gilt: $uv^i w \in L$.
(d.h.: $uv \in L, uvw \in L, uvvw \in L, uvvww \in L, \dots$)

$$L_1 := \{b^n b^k : n, k \in \mathbb{N}, n = k\}$$

regulär: ja nein

Beweis:

$$L_1 := \{b^n a^k : n, k \in \mathbb{N}, n = k\}$$

regulär: ja nein

Beweis:

Aufgabe 6:**(16 Punkte)**

- (a) Im Folgenden wird die Menge AT der *arithmetischen Terme mit den Variablen a, b und c* über dem Alphabet $\Sigma := \{a, b, c, +, -, \cdot, :, (,)\}$ rekursiv definiert:

Basisregel:

- (B) Jede der Variablen a, b, c ist in AT .

Rekursive Regeln:

- (R1) Ist w in AT , so ist auch $-w$ in AT .
 (R2) Sind w_1 und w_2 in AT , so sind auch $(w_1 + w_2)$ und $(w_1 - w_2)$ in AT .
 (R3) Sind w_1 und w_2 in AT , so sind auch $(w_1 \cdot w_2)$ und $(w_1 : w_2)$ in AT .

- (i) Welche der folgenden Wörter gehören zur Sprache AT , welche nicht? (4 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in AT ?	
$-a$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(a + (b - a))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(b - - - - a)$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(b \cdot (b - +a))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (ii) Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = AT$. (4 Pkte)

- (iii) Betrachten Sie die von Ihnen in (b) angegebene kontextfreie Grammatik G und geben (2 Pkte)
Sie einen Ableitungsbaum für das folgende Wort an:

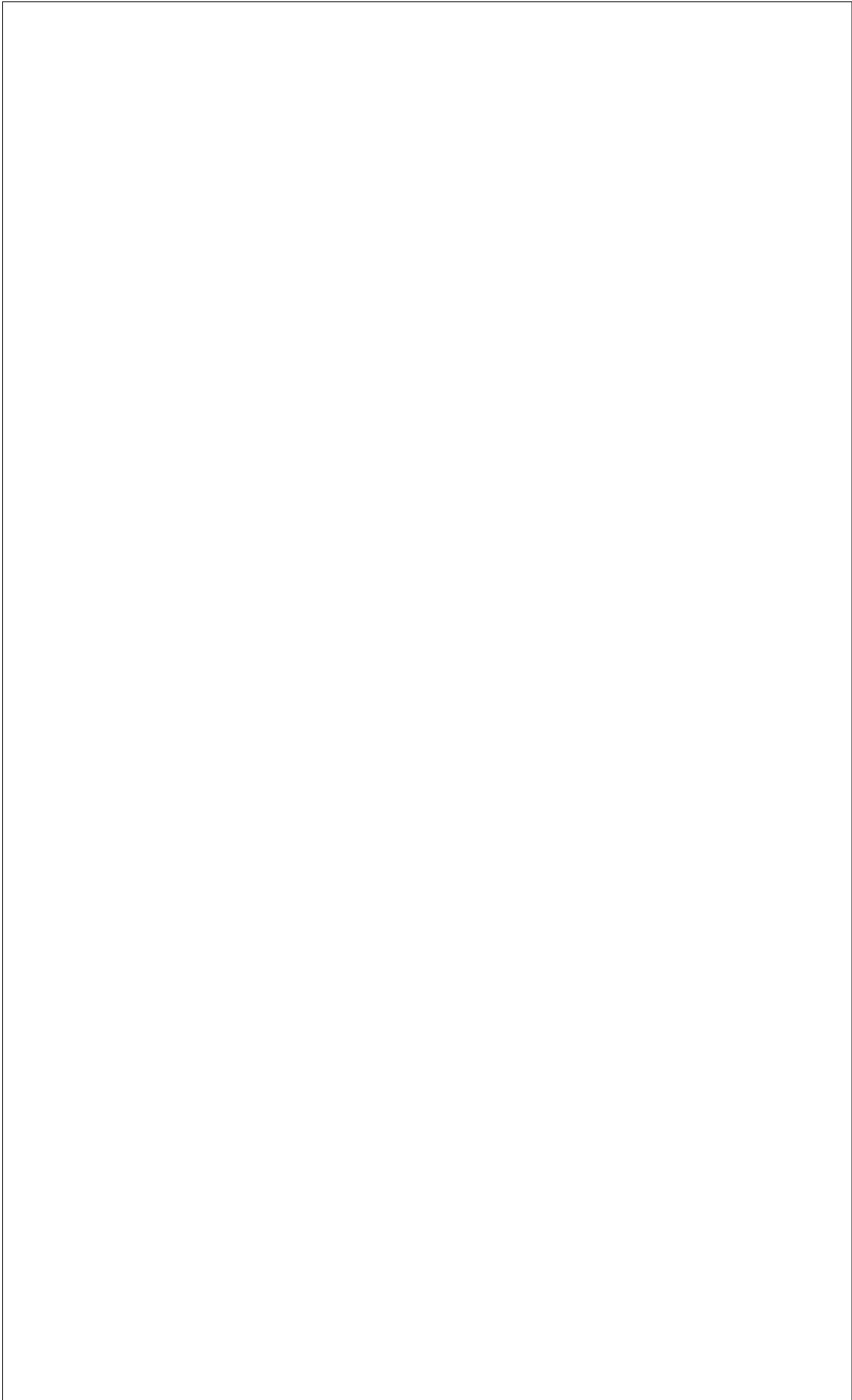
$$-(a \cdot (b - c))$$

- (b) Die zwei Algorithmen \mathcal{A}_1 und \mathcal{A}_2 lösen bei Eingabe von mindestens zwei unterschiedlichen Wörtern das selbe Problem. Die Laufzeit g_1 von \mathcal{A}_1 bzw. g_2 von \mathcal{A}_2 ist abhängig von der Anzahl n der Eingabewörter und der Länge l des längsten Wortes der Eingabe. Insbesondere ist (6 Pkte)

$$g_1(n) = (1 + l)^n \quad \text{und} \quad g_2(n) = 1 + n \cdot l$$

Zeigen Sie, dass der Algorithmus \mathcal{A}_1 im Allgemeinen langsamer als \mathcal{A}_2 ist, d.h. zeigen Sie per Induktion über n für $n \in \mathbb{N}_{>0}$ mit $n \geq 2$, dass für jedes $l \in \mathbb{N}_{>0}$ gilt:

$$g_1(n) > g_2(n) \text{ , d.h. } (1 + l)^n > 1 + n \cdot l$$



Aufgaben der Klausur SoSe 2009

Aufgabe 1:**(22 Punkte)**

(a) Betrachten Sie die folgende Aussage:

(2 Pkte)

Wenn mein Wecker nicht kaputt geht, komme ich morgen pünktlich zur Vorlesung, falls die Tram nach Fahrplan fährt.

Formalisieren Sie die Aussage durch eine aussagenlogische Formel, in der Sie die atomaren Aussagen K (der Wecker geht kaputt), P (ich komme pünktlich zur Vorlesung) und F (die Tram fährt nach Fahrplan) verwenden.

(b) Bekanntlich sagt nicht jedermann die Wahrheit. Markus, Otto und Paul geben folgende Auskunft: (4 Pkte)

- 1) Markus sagt: „Otto lügt.“
- 2) Otto sagt: „Paul lügt.“
- 3) Paul sagt: „Otto und Markus lügen.“

Dieser Sachverhalt wird mit Hilfe der atomaren Aussagen M (Markus sagt die Wahrheit), O (Otto sagt die Wahrheit) und P (Paul sagt die Wahrheit) durch die drei Formeln

$$\varphi_1 := (M \leftrightarrow \neg O), \quad \varphi_2 := (O \leftrightarrow \neg P) \text{ und } \varphi_3 := (P \leftrightarrow (\neg O \wedge \neg M))$$

formalisiert.

Aber wer sagt die Wahrheit und wer lügt? Gibt es eine Belegung der aussagenlogischen Variablen M , P und O , derart dass das System aus den Aussagen 1) - 3) widerspruchsfrei ist? Geben Sie Ihren Lösungsweg an.

- (c) Geben Sie für jede der folgenden aussagenlogischen Formeln an, ob sie erfüllbar und/oder allgemeingültig ist. (Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.) (7 Pkte)

Geben Sie außerdem folgendes für jede Formel an: Falls die Formel erfüllbar ist, geben Sie eine zur Formel passende Belegung an, die die Formel erfüllt. Falls die Formel nicht allgemeingültig ist, geben Sie eine zur Formel passende Belegung an, die die Formel *nicht* erfüllt.

- $\varphi = \left(((X_1 \vee X_2) \leftrightarrow \neg X_3) \wedge (X_1 \wedge \neg X_3) \right)$

erfüllbar:	<input type="checkbox"/> ja	<input type="checkbox"/> nein
allgemeingültig:	<input type="checkbox"/> ja	<input type="checkbox"/> nein
Falls φ erfüllbar ist, geben Sie hier eine zu φ passende Belegung an, die φ erfüllt:		
Falls φ nicht allgemeingültig ist, geben Sie hier eine zu φ passende Belegung an, die φ nicht erfüllt:		

- $\psi = \left(((X_1 \wedge X_2) \leftrightarrow X_3) \vee (X_1 \vee X_3) \right)$

erfüllbar:	<input type="checkbox"/> ja	<input type="checkbox"/> nein
allgemeingültig:	<input type="checkbox"/> ja	<input type="checkbox"/> nein
Falls ψ erfüllbar ist, geben Sie hier eine zu ψ passende Belegung an, die ψ erfüllt:		
Falls ψ nicht allgemeingültig ist, geben Sie hier eine zu ψ passende Belegung an, die ψ nicht erfüllt:		

- (d) Geben Sie eine präzise Definition für die logische Äquivalenz zweier aussagenlogischer Formeln an. (2 Pkte)

Definition: (logische Äquivalenz)

(e) Seien φ , ψ und χ beliebige aussagenlogische Formeln. Gelten folgende Äquivalenzen? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

$$\psi \equiv (\psi \vee \varphi) \quad \square \text{ ja} \quad \square \text{ nein}$$

$$(\varphi \vee (\psi \wedge \chi)) \equiv ((\varphi \wedge \psi) \vee (\varphi \wedge \chi)) \quad \square \text{ ja} \quad \square \text{ nein}$$

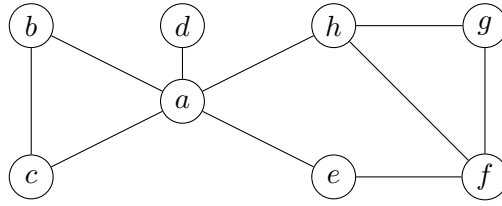
$$(\psi \leftrightarrow \varphi) \equiv ((\neg\varphi \vee \psi) \wedge (\psi \rightarrow \varphi)) \quad \square \text{ ja} \quad \square \text{ nein}$$

(f) (4 Pkte)

Geben Sie eine zur Formel

$$\varphi := ((\neg X_1 \rightarrow \mathbf{0}) \rightarrow ((X_1 \vee \neg X_2) \wedge X_3))$$

äquivalente Formel in Negationsnormalform an. Geben Sie auch Ihren Lösungsweg an.

Aufgabe 2:**(23 Punkte)**(a) Sei $G = (V, E)$ der folgende Graph:(i) Geben Sie die Knotenmenge V und die Kantenmenge E von G an. Repräsentieren Sie G außerdem durch eine Adjazenzliste. $V =$

(1 Pkt)

 $E =$

(1 Pkt)

Adjazenzliste:

(1 Pkt)

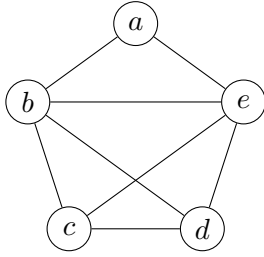
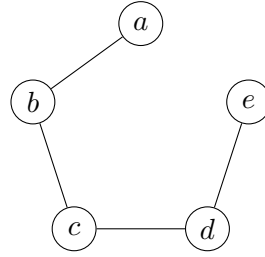
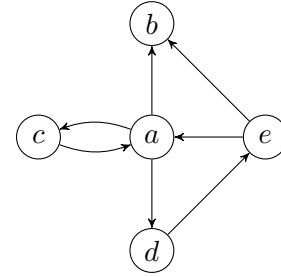
(ii) Geben Sie einen einfachen Weg der Länge 3 vom Knoten d zum Knoten g an:

(1 Pkt)

(iii) Wie viele verschiedene einfache Kreise enthalten den Knoten a als Start- und Endknoten?

(1 Pkt)

(b) Betrachten Sie die folgenden ungerichteten Graphen G_1, G_2 und den gerichteten Graphen G_3 :

 G_1  G_2  G_3

Welche der folgenden Aussagen sind wahr, welche falsch?

(3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- G_1 besitzt einen Eulerkreis. wahr falsch
- G_2 ist ein Spannbaum von G_1 . wahr falsch
- G_3 ist stark zusammenhängend. wahr falsch

(c) Welche der folgenden Aussagen über Binärbäume sind wahr, welche falsch?

(3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

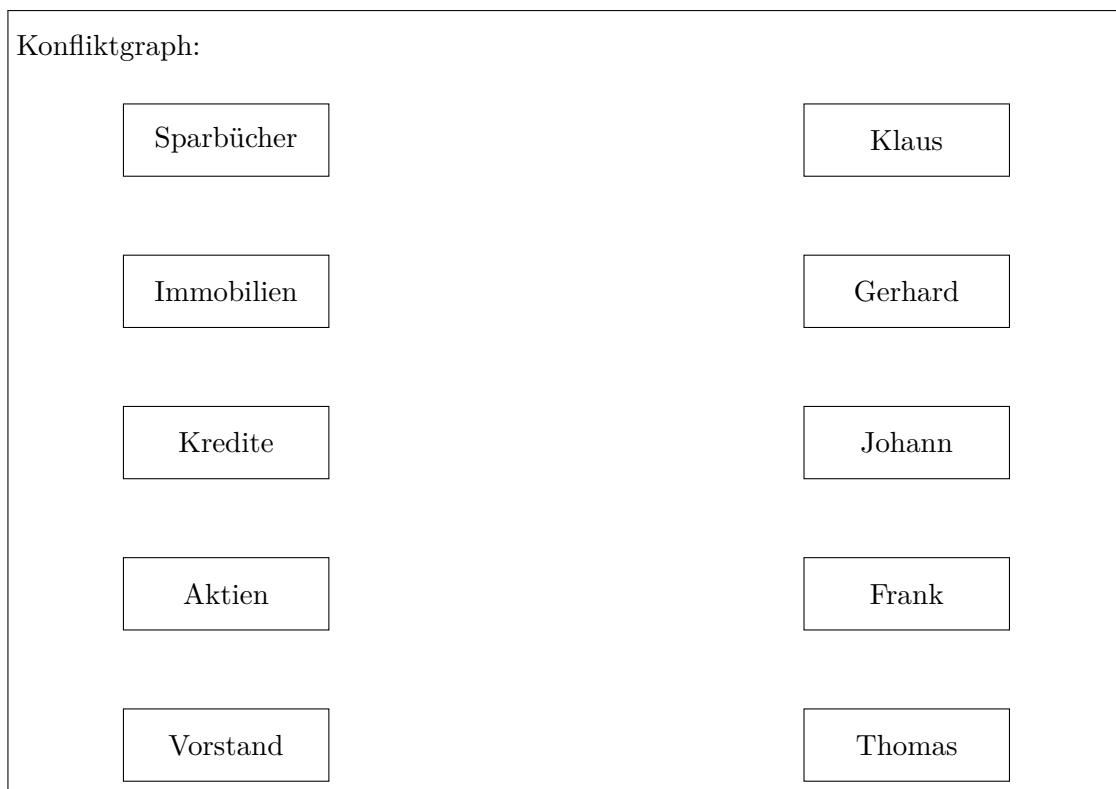
- In jedem vollständigen Binärbaum ist die Anzahl der Kanten genau doppelt so groß wie die Anzahl der Knoten. wahr falsch
- Es existiert ein vollständiger Binärbaum B , der einen Knoten v enthält mit $\text{Ein-Grad}_B(v) = \text{Aus-Grad}_B(v) = 0$. wahr falsch
- In jedem Binärbaum gibt es einen einfachen Weg von der Wurzel zu einem Blatt. wahr falsch

(d) Die “Hyper Real Finance” Bank hat die Finanzkrise überstanden. Sie schreibt Stellenangebote aus: In den vier Abteilungen *Sparbücher*, *Immobilien*, *Kredite* und *Aktien* ist jeweils eine Position zu besetzen. Außerdem ist ein Job im *Vorstand* frei. Passend dazu treffen fünf Bewerbungen ein. Die Bewerber sind allerdings recht eigenwillig, jeder von ihnen hat starke Abneigungen gegen bestimmte Abteilungen, in denen er auf keinen Fall arbeiten will. Im Einzelnen ergeben sich folgende Zu- und Abneigungen:

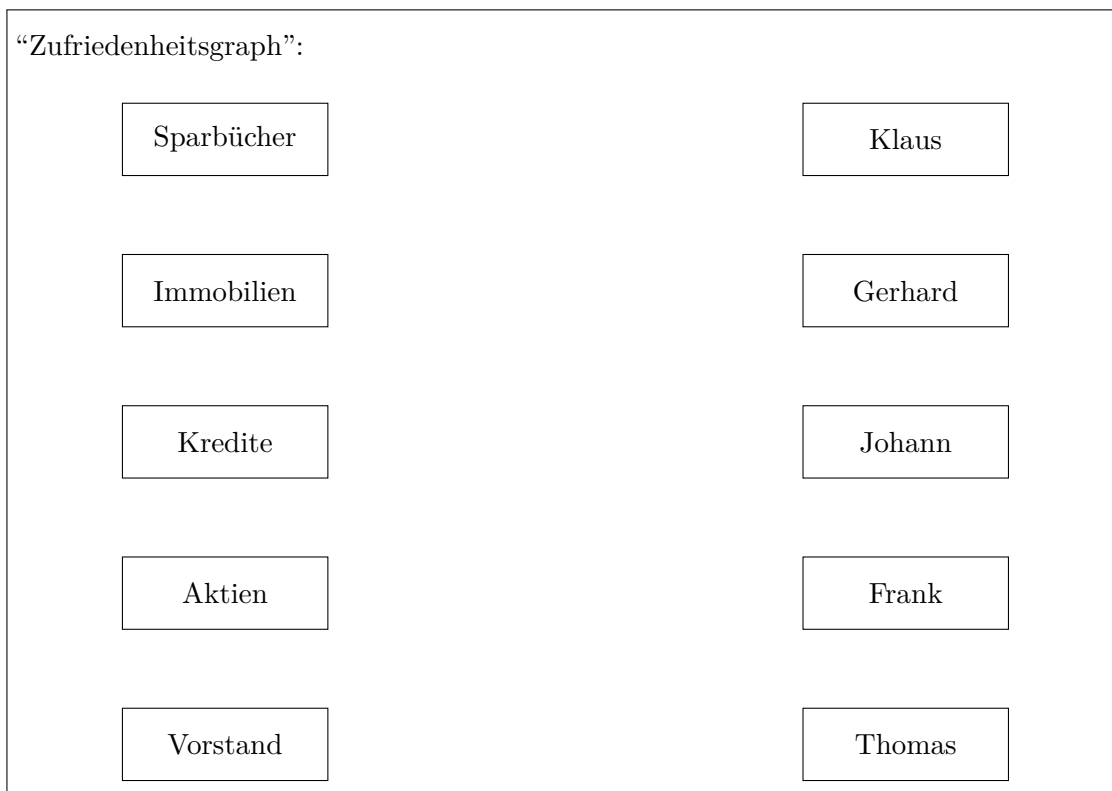
- Alle Bewerber können sich vorstellen, in den Vorstand zu gehen.
- Andererseits will keiner außer Klaus mit den Sparbüchern zu tun haben.
- Klaus wiederum möchte weder in der Kredit- noch in der Aktienabteilung arbeiten.
- Gerhard möchte sich nicht mit den Immobilien beschäftigen.
- Auch Johann mag keine Immobilien und zusätzlich möchte er nicht in die Aktienabteilung gehen.
- Für Frank kommt weder die Aktien- noch die Kredit- oder die Immobilienabteilung in Frage.
- Für Thomas gibt es bis auf die Sparbücher und die Immobilien keine Einschränkungen.

Es ist klar, dass jede Stelle nur von höchstens einem Bewerber besetzt werden kann und jeder Bewerber nur höchstens eine Stelle erhalten kann.

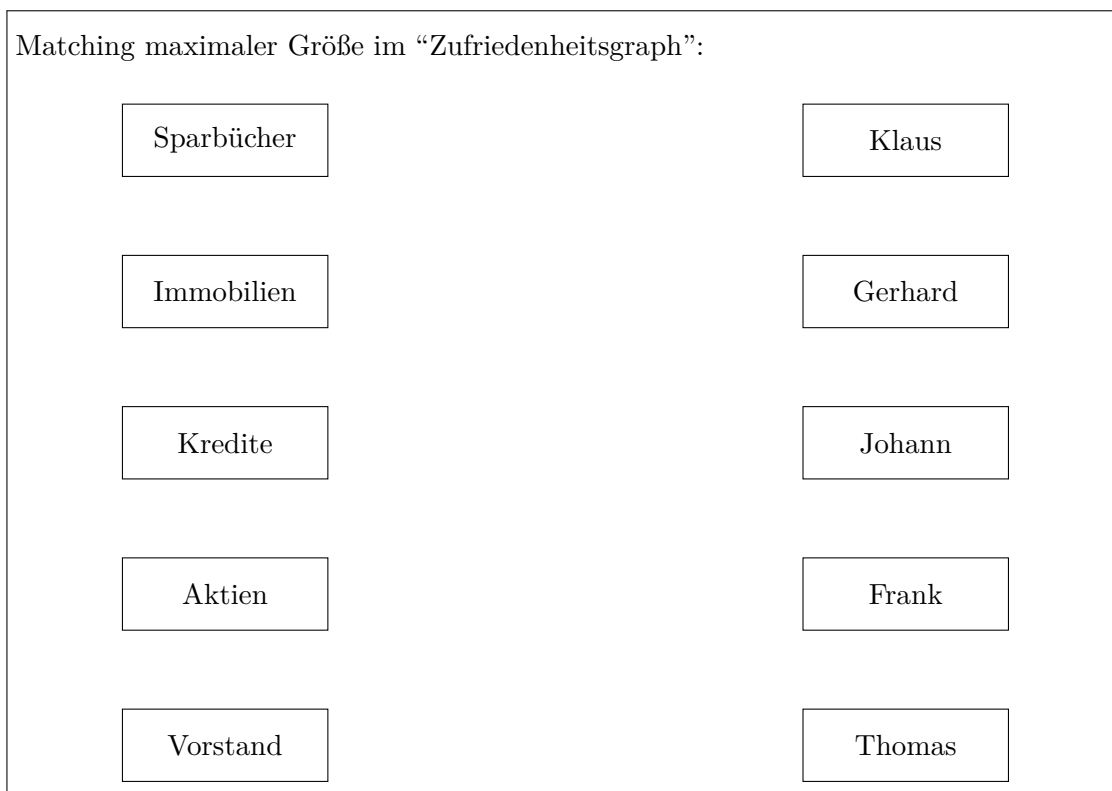
(i) Stellen Sie den Konfliktgraphen auf, in dem eine Kante zwischen Stelle A und Bewerber B dafür steht, dass B nicht auf der Stelle A arbeiten will. (2 Pkte)



- (ii) Bilden Sie auf der Grundlage Ihres Konfliktgraphen einen “Zufriedenheitsgraphen”, in dem eine Kante zwischen Stelle A und Bewerber B dafür steht, dass B mit der Stelle A zufrieden wäre. (2 Pkte)



- (iii) Geben sie ein Matching maximaler Größe in Ihrem “Zufriedenheitsgraphen” an (2 Pkte)



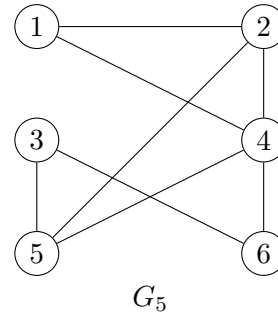
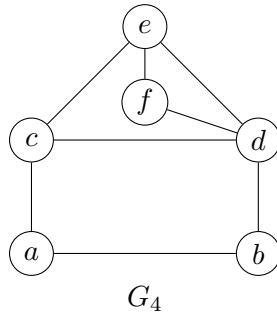
- (iv) Geben Sie eine Möglichkeit an, wie die Bank die Bewerber auf die Stellen verteilen kann, (1 Pkt)
so dass möglichst viele Stellen besetzt werden und kein Bewerber eine Stelle erhält, auf
der er nicht arbeiten will.

A large, empty rectangular box with a thin black border, intended for the student to write their answer to question (iv).

- (v) Begründen Sie, warum die von Ihnen gefundene Zuordnung bestmöglich ist, das heißt, (2 Pkte)
warum es keine Zuordnung von den Bewerbern auf die Stellen gibt, die mehr Stellen
besetzt.

A large, empty rectangular box with a thin black border, intended for the student to write their answer to question (v).

(e) Betrachten Sie die folgenden ungerichteten Graphen G_4 und G_5 :



(i) Geben Sie einen Isomorphismus von G_4 nach G_5 an.

(2 Pkte)

(ii) Ist der Graph G_5 planar? Begründen Sie Ihre Antwort.

(1 Pkt)

Aufgabe 3:**(12 Punkte)**

- (a) Sei $\sigma = \{\dot{B}\}$ eine Signatur mit einem zweistelligen Relationssymbol \dot{B} , wobei $\dot{B}(x, y)$ besagt, dass das Bauteil y Bestandteil des Bauteils x ist. (4 Pkte)
- (i) Geben Sie eine Formel φ der Logik erster Stufe über die Signatur σ an, die aussagt, dass mindestens zwei Bauteile existieren, welche nicht Bestandteil eines anderen Bauteils sind.

$$\varphi =$$

- (ii) Beschreiben Sie umgangssprachlich, was die folgende Formel

$$\forall x \forall y \forall z \left(((\dot{B}(x, y) \wedge \dot{B}(y, z)) \rightarrow \dot{B}(x, z)) \wedge (\dot{B}(x, y) \rightarrow \neg \dot{B}(y, x)) \right)$$

aussagt.

- (b) Sei φ eine FO[σ]-Formel. Wann heißt eine Belegung β *passend* zu φ ? Geben Sie eine präzise Definition an. (2 Pkte)

Definition: (passende Belegung)

- (c) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem zweistelligen Relationssymbol \dot{E} . Geben Sie für die (4 Pkte)
Formel

$$\varphi(x) := \forall y \exists z ((\dot{E}(y, x) \rightarrow \dot{E}(x, z)) \vee x \dot{=} y)$$

eine Struktur \mathfrak{A} und zwei Interpretationen $\mathcal{I}_1 = (\mathfrak{A}, \beta_1)$ und $\mathcal{I}_2 = (\mathfrak{A}, \beta_2)$ an, so dass \mathcal{I}_1 die Formel φ erfüllt und \mathcal{I}_2 nicht.

- (d) Sei $\sigma = \{\dot{E}\}$ eine Signatur mit einem zweistelligen Relationssymbol \dot{E} . Strukturen über (2 Pkte)
dieser Signatur kann man als gerichtete Graphen auffassen. Wir betrachten zwei beliebige σ -Strukturen \mathfrak{A} und \mathfrak{B} , welche zueinander isomorph sind.

Kann es einen Satz φ der Logik erster Stufe über der Signatur σ geben, so dass \mathfrak{A} den Satz erfüllt, aber \mathfrak{B} nicht? Begründen Sie Ihre Antwort.

Aufgabe 4:**(15 Punkte)**

Die Sprache UPNZ (Umgekehrte Polnische Notation auf Ziffern) sei über dem Alphabet $\Sigma := \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, *\}$ wie folgt rekursiv definiert:

Basisregel: (B) Jede Ziffer, also jedes Zeichen aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ ist in UPNZ

Rekursive Regeln: (R1) Sind w_1 und w_2 in UPNZ, so ist auch w_1w_2+ in UPNZ

(R2) Sind w_1 und w_2 in UPNZ, so ist auch w_1w_2* in UPNZ

(a) Welche der folgenden Wörter gehören zur Sprache UPNZ, welche nicht? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in UPNZ?	
123	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$69 + 38 * +$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$9 + 3$	<input type="checkbox"/> ja	<input type="checkbox"/> nein

(b) Geben Sie eine kontextfreie Grammatik G an, die genau die Sprache UPNZ erzeugt.

(4 Pkte)

- (c) Zur Erinnerung wird die bereits zu Beginn der Aufgabe 4 angegebene Definition der Sprache UPNZ wiederholt:

Die Sprache UPNZ (Umgekehrte Polnische Notation auf Ziffern) sei über dem Alphabet $\Sigma := \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, *\}$ wie folgt rekursiv definiert:

Basisregel: (B) Jede Ziffer, also jedes Zeichen aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ ist in UPNZ

Rekurs. Regeln: (R1) Sind w_1 und w_2 in UPNZ, so ist auch w_1w_2+ in UPNZ

(R2) Sind w_1 und w_2 in UPNZ, so ist auch w_1w_2* in UPNZ

Wir definieren zwei Funktionen f und g , die jedem Wort $w \in \text{UPNZ}$ jeweils einen Wert zuweisen.

Sei z eine Ziffer, also ein Zeichen aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ und seien w_1, w_2 Worte aus UPNZ. Die Funktionen $f : \text{UPNZ} \rightarrow \mathbb{N}$ und $g : \text{UPNZ} \rightarrow \{0, 1\}$ sind induktiv entsprechend der Definition von UPNZ wie folgt definiert:

$$f(z) = z \qquad g(z) = \begin{cases} 0, & \text{falls } z \text{ gerade ist} \\ 1, & \text{sonst} \end{cases}$$

$$f(w_1w_2+) = f(w_1) + f(w_2) \qquad g(w_1w_2+) = \begin{cases} 0, & \text{falls } g(w_1) = g(w_2) \\ 1, & \text{sonst} \end{cases}$$

$$f(w_1w_2*) = f(w_1) \cdot f(w_2) \qquad g(w_1w_2*) = \begin{cases} 0, & \text{falls } g(w_1) = 0 \text{ oder } g(w_2) = 0 \\ 1, & \text{sonst} \end{cases}$$

Nach diesen Definitionen ist zum Beispiel $f(32 + 47 + *) = 55$ und $g(32 + 47 + *) = 1$.

- (i) Berechnen Sie die folgenden Funktionswerte

(2 Pkte)

$$f(24 * 91 * +) = \boxed{} \qquad g(24 * 91 * +) = \boxed{}$$

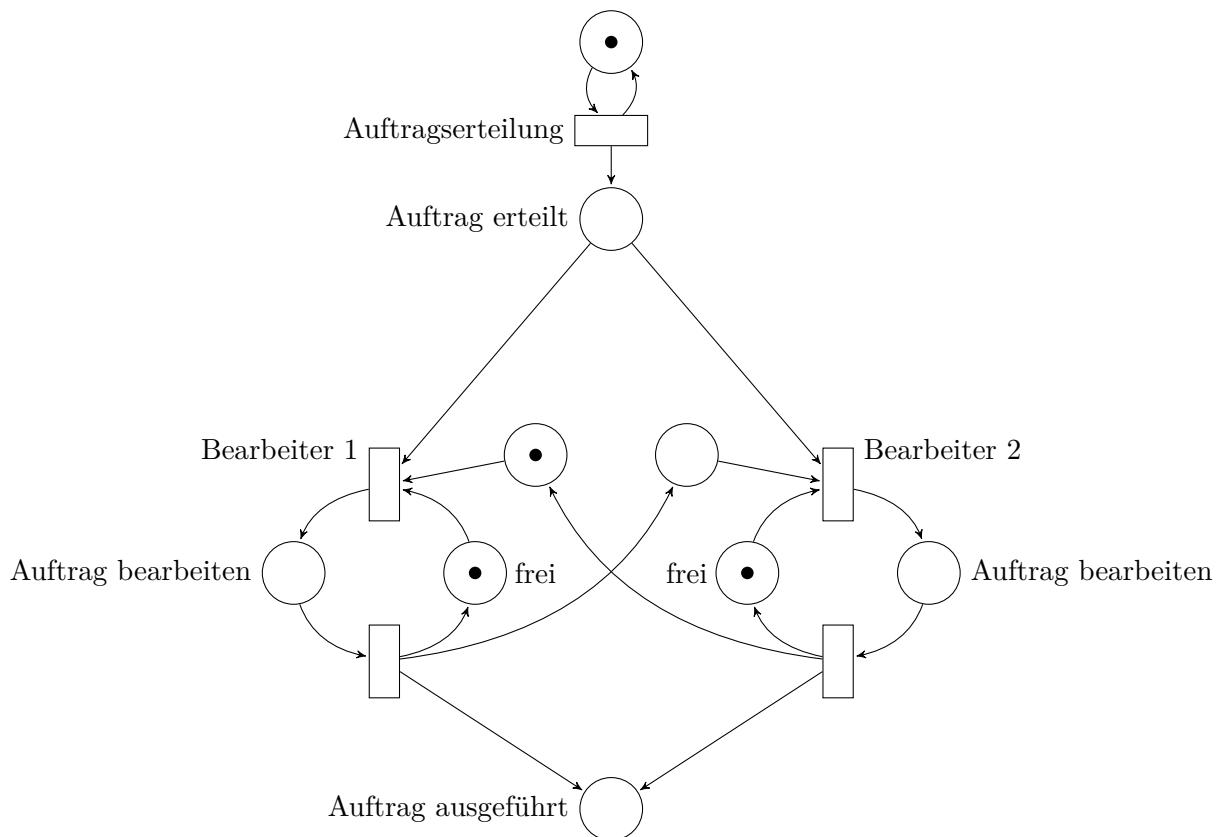
(ii) Beweisen Sie den folgenden Zusammenhang durch vollständige Induktion:

(6 Pkte)

Für jedes Wort $w \in \text{UPNZ}$ gilt: $f(w)$ ist gerade $\iff g(w) = 0$

Aufgabe 5:**(8 Punkte)**

- (a) Die Bearbeitung von Aufträgen durch zwei Bearbeiter wurde durch folgendes Petri-Netz (4 Pkte) modelliert.



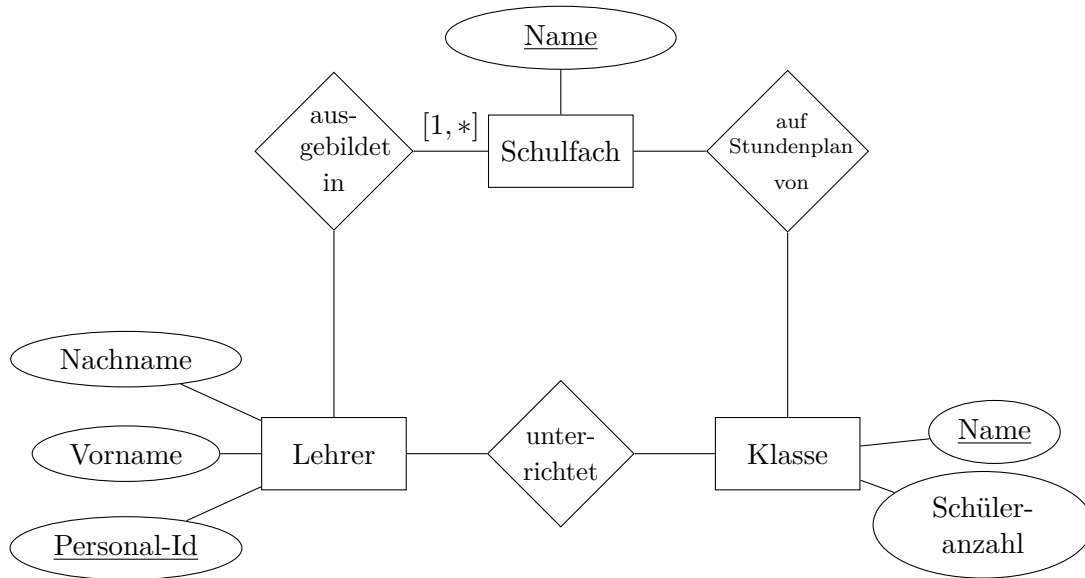
Welche der folgenden Aussagen sind im Sinne des modellierten Petri-Netzes wahr, welche nicht?

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- (i) Es können sich beliebig viele erteilte Aufträge für die Bearbeiter anhäufen. wahr falsch
- (ii) Ein neuer Auftrag kann nur erteilt werden, wenn der letzte ausgeführt wurde. wahr falsch
- (iii) Es können zwei Aufträge gleichzeitig bearbeitet werden. wahr falsch
- (iv) Es ist möglich, dass der Bearbeiter 1 bereits 42 Aufträge mehr als der Bearbeiter 2 bearbeitet hat. wahr falsch

(b) Es sei folgendes Entity-Relationship-Modell gegeben:

(4 Pkte)



Welche der folgenden Aussagen sind im Sinne des oben angegebenen Entity-Relationship-Modells wahr, welche nicht?

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- (i) In jeder Klasse wird jedes Schulfach unterrichtet. wahr falsch
- (ii) Für jedes Schulfach ist mindestens ein Lehrer ausgebildet. wahr falsch
- (iii) In jeder Klasse müssen alle Schüler unterschiedliche Namen haben. wahr falsch
- (iv) Es ist möglich, dass es zwei Lehrer mit demselben Nachnamen und demselben Vornamen gibt. wahr falsch

Aufgabe 6:**(20 Punkte)**

- (a) R sei als folgender regulärer Ausdruck über dem Alphabet $\Sigma := \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, :\}$ gegeben:

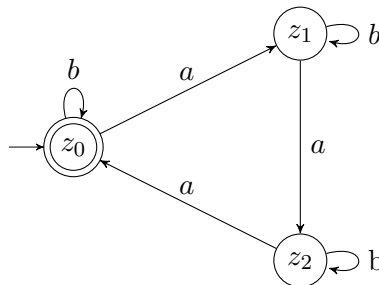
$$((1(0|1|2))|(0|1|\dots|9)) : (0|1|2|3|4|5)(0|1|\dots|9)$$

- (i) Welche der folgenden Wörter liegen in der von R definierten Sprache $L(R)$, welche nicht? (3 Pkte)
 Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in $L(R)$?	
9 : 40	<input type="checkbox"/> ja	<input type="checkbox"/> nein
125	<input type="checkbox"/> ja	<input type="checkbox"/> nein
13 : 28	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (ii) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(R)$ an, die von R definiert wird. (2 Pkte)

- (b) Sei A der folgende deterministische endliche Automat:



- (i) Welche der folgenden Wörter werden von A akzeptiert, welche nicht? (3 Pkte)
 Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... wird akzeptiert?	
<i>aababbb</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>bbabbabb</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>ababaaaba</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein

- (ii) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(A)$ an, die vom Automaten A akzeptiert wird. (2 Pkte)

- (c) Geben Sie die graphische Darstellung eines nicht-deterministischen endlichen Automaten an, (3 Pkte) der genau diejenigen Wörter über dem Alphabet $\{a, b\}$ akzeptiert, die das Teilwort bb zweimal enthalten, getrennt durch mindestens ein weiteres Zeichen.

D.h. der Automat soll genau die Wörter $w \in \{a, b\}^*$ akzeptieren, für die es Wörter $x, y, z \in \{a, b\}^*$ mit $y \neq \varepsilon$ gibt, so dass $w = xbybbz$.

(d) Betrachten Sie das Alphabet $\Sigma := \{a, b\}$ und zeigen Sie, dass die Sprache

(7 Pkte)

$$L := \{w \in \{a, b\}^* : w \text{ enthält genau doppelt so viele } a\text{'s wie } b\text{'s}\}$$

nicht regulär ist.

Hinweis: Verwenden Sie das Pumping-Lemma aus der Vorlesung:

Sei Σ ein endliches Alphabet. Für jede reguläre Sprache $L \subseteq \Sigma^*$ gibt es eine Zahl $z \in \mathbb{N}$, so dass für jedes Wort $x \in L$ der Länge $|x| \geq z$ gilt: Es gibt eine Zerlegung von x in Worte $u, v, w \in \Sigma^*$, so dass die folgenden vier Bedingungen erfüllt sind:

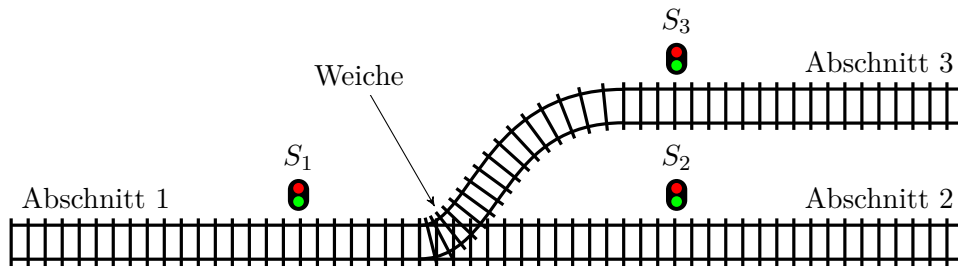
- (i) $x = uvw$
- (ii) $|uv| \leq z$
- (iii) $|v| \geq 1$
- (iv) für jedes $i \in \mathbb{N}$ gilt: $uv^i w \in L$.
(d.h.: $uw \in L, uvw \in L, uvvw \in L, uvvww \in L, \dots$)



Aufgaben der Klausur WS 08/09

Aufgabe 1:**(21 Punkte)**

(a) Es sei die folgende Weiche einer Bahnanlage gegeben:



Die drei Signale S_1 , S_2 und S_3 geben an, ob ein Zug vom entsprechenden Gleisabschnitt über die Weiche fahren darf: Leuchtet S_i grün, dann darf ein Zug vom Gleisabschnitt i aus über die Weiche fahren.

Die Weiche ist entweder auf „geradeaus fahren“ oder „abbiegen“ eingestellt. Wenn sie auf „geradeaus fahren“ eingestellt ist, dann fahren Züge, die vom Gleisabschnitt 1 aus über die Weiche fahren, auf den Gleisabschnitt 2 und umgekehrt. Ansonsten fahren Züge, die vom Gleisabschnitt 1 aus über die Weiche fahren, auf den Gleisabschnitt 3 und umgekehrt.

Mit Hilfe der folgenden atomaren Aussagen lassen sich nun einfache Anforderungen an die Weichenanlage formulieren:

- X_1 : S_1 leuchtet grün.
- X_2 : S_2 leuchtet grün.
- X_3 : S_3 leuchtet grün.
- X_W : Die Weiche ist auf „geradeaus fahren“ eingestellt.

Beispielsweise besagt die aussagenlogische Formel $(\neg X_W \wedge (X_1 \wedge \neg X_3))$, dass die Weiche auf „abbiegen“ eingestellt ist, S_1 grün leuchtet und S_3 rot leuchtet.

Geben Sie aussagenlogische Formeln an, die nur die Variablen X_1 , X_2 , X_3 und X_W benutzen und Folgendes aussagen:

- Wenn S_2 grün leuchtet, dann ist die Weiche auf „geradeaus fahren“ eingestellt, und wenn S_3 grün leuchtet, dann ist die Weiche auf „abbiegen“ eingestellt. (2 Pkte)

- Höchstens eines der drei Signale S_1 , S_2 , S_3 leuchtet grün. D.h. wenn S_1 grün leuchtet, dann leuchten S_2 und S_3 nicht grün und analog für die anderen Signale. Beachten Sie, dass auch der Fall eintreten kann, bei dem keines der drei Signale grün leuchtet. (3 Pkte)

- (b) Geben Sie für jede der folgenden aussagenlogischen Formeln an, ob sie erfüllbar und/oder allgemeingültig ist. (Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.) (7 Pkte)

Geben Sie außerdem folgendes für jede Formel an: Falls die Formel erfüllbar ist, geben Sie eine zur Formel passende Belegung an, die die Formel erfüllt. Falls die Formel nicht allgemeingültig ist, geben Sie eine zur Formel passende Belegung an, die die Formel *nicht* erfüllt.

• $\varphi = \left(((X_1 \vee X_2) \leftrightarrow X_3) \wedge (X_1 \wedge \neg X_3) \right)$

erfüllbar:	<input type="checkbox"/> ja	<input type="checkbox"/> nein	
allgemeingültig:	<input type="checkbox"/> ja	<input type="checkbox"/> nein	
Falls φ erfüllbar ist, geben Sie hier eine zu φ passende Belegung an, die φ erfüllt:			
Falls φ nicht allgemeingültig ist, geben Sie hier eine zu φ passende Belegung an, die φ nicht erfüllt:			

• $\psi = \left(\left(X_1 \wedge (X_1 \rightarrow (X_2 \vee X_3)) \right) \rightarrow X_3 \right)$

erfüllbar:	<input type="checkbox"/> ja	<input type="checkbox"/> nein	
allgemeingültig:	<input type="checkbox"/> ja	<input type="checkbox"/> nein	
Falls ψ erfüllbar ist, geben Sie hier eine zu ψ passende Belegung an, die ψ erfüllt:			
Falls ψ nicht allgemeingültig ist, geben Sie hier eine zu ψ passende Belegung an, die ψ nicht erfüllt:			

- (c) Welche der folgenden Formeln sind in Negationsnormalform (NNF), disjunktiver Normalform (DNF) und/oder konjunktiver Normalform (KNF)? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen halben Punkt, für jedes **falsche Kreuz** wird **ein halber Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

	$\left((\neg X_1 \wedge (X_2 \vee \neg X_3)) \vee \neg X_2 \right)$		$\left((X_1 \vee \neg X_2) \wedge ((\neg X_1 \vee X_3) \vee X_4) \right)$	
in NNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein
in DNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein
in KNF?	<input type="checkbox"/> ja	<input type="checkbox"/> nein	<input type="checkbox"/> ja	<input type="checkbox"/> nein

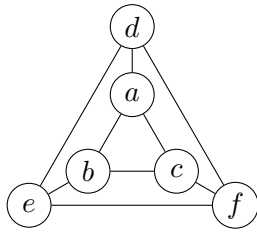
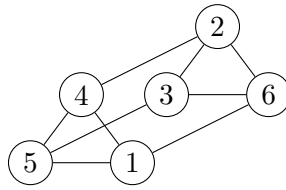
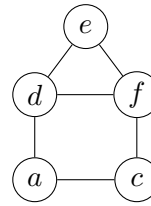
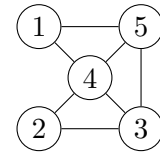
(d) Geben Sie eine zur Formel

(6 Pkte)

$$\varphi := ((X_2 \rightarrow X_1) \vee (X_2 \wedge \neg X_3))$$

äquivalente aussagenlogische Formel in konjunktiver Normalform an. Geben Sie auch Ihren Lösungsweg an.

(b) Betrachten Sie die folgenden ungerichteten Graphen G_1, G_2, G_3, G_4 :

 G_1  G_2  G_3  G_4

(i) Welche der folgenden Aussagen sind wahr, welche falsch? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- G_4 ist ein Teilgraph von G_2 . wahr falsch
- G_3 ist ein induzierter Teilgraph von G_1 . wahr falsch
- G_3 und G_4 sind isomorph. wahr falsch

(ii) Geben Sie einen Isomorphismus von G_1 nach G_2 an. (2 Pkte)

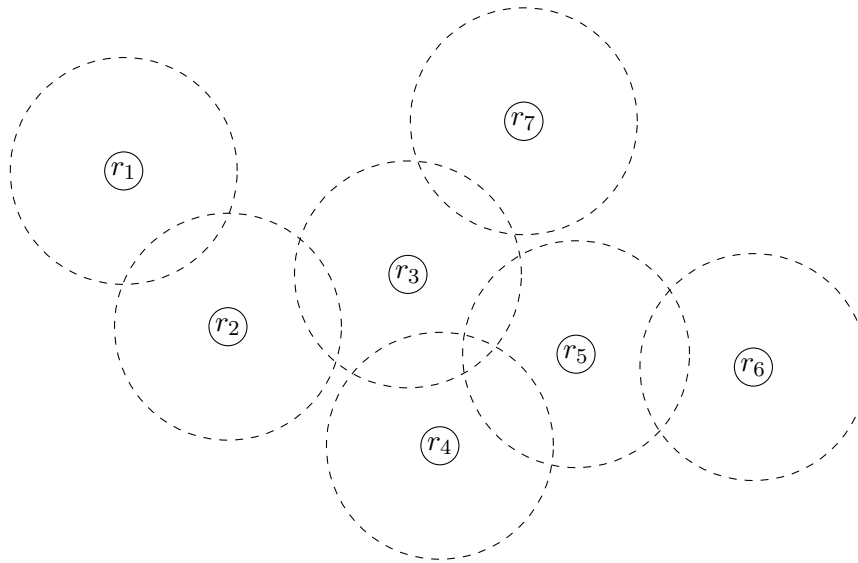
(c) Welche der folgenden Aussagen über Bäume sind wahr, welche falsch? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- Sind x und y Knoten in einem ungerichteten Baum B , so gibt es in B genau einen einfachen Weg von x nach y . wahr falsch
- Es gibt einen Baum B und zwei Knoten x und y in B , so dass es in B mindestens zwei verschiedene einfache Wege von x nach y gibt. wahr falsch
- Ist $B = (V, E)$ ein Baum, so gilt $|E| = |V| + 1$. wahr falsch

- (d) Es seien Radiostationen r_1, \dots, r_7 gegeben. Jeder Radiostation soll eine von drei Frequenzen f_1, f_2, f_3 zugeordnet werden. Radiostationen, die zu dicht beieinander liegen, dürfen allerdings nicht die gleichen Frequenzen zugewiesen bekommen.

Das folgende Diagramm stellt die Lage der einzelnen Radiostationen dar.



Um jede Station ist ein gestrichelter Kreis eingezeichnet. Schneiden sich die Kreise von zwei Radiostationen r_i und r_j , so liegen r_i und r_j zu dicht beieinander und dürfen nicht die gleiche Frequenz zugeordnet bekommen. Wir sagen auch, dass r_i und r_j in Konflikt zueinander stehen. Zum Beispiel stehen r_1 und r_2 in Konflikt zueinander, r_1 und r_3 aber nicht. r_1 und r_2 dürfen also nicht die gleiche Frequenz zugeordnet bekommen, wohingegen r_1 und r_3 auf der gleichen Frequenz senden dürfen.

- (i) Geben Sie den Konfliktgraph G an (in graphischer Darstellung).

(2 Pkte)

- (ii) Weisen Sie jeder der Radiostationen r_1, \dots, r_7 genau eine der drei Frequenzen f_1, f_2, f_3 zu, so dass Radiostationen, die zueinander in Konflikt stehen, nicht der gleichen Frequenz zugeordnet sind. (3 Pkte)

D.h.: Sei V die Menge der Knoten des Konfliktgraphen G aus (i). Geben Sie eine konfliktfreie Knotenmarkierung $m: V \rightarrow \{1, 2, 3\}$ an.

- (iii) Wie viele Frequenzen werden für die Radiostationen mindestens benötigt? Begründen Sie Ihre Antwort. (2 Pkte)

Aufgabe 3:**(12 Punkte)**

(a) Sei $\sigma := \{\dot{S}, \dot{G}\}$ eine Signatur mit 2-stelligen Relationssymbolen \dot{S}, \dot{G} . Sei $\mathfrak{A} = (A, \dot{S}^{\mathfrak{A}}, \dot{G}^{\mathfrak{A}})$ die σ -Struktur, in der

- A die Menge der Spieler eines Turniers ist,
- $\dot{S}^{\mathfrak{A}}$ alle Tupel $(x, y) \in A \times A$ enthält, so dass x gegen y gespielt hat und
- $\dot{G}^{\mathfrak{A}}$ alle Tupel $(x, y) \in A \times A$ enthält, so dass x gegen y gewonnen hat.

(i) Geben Sie eine Formel φ der Logik erster Stufe über der Signatur σ an, die in \mathfrak{A} aussagt, dass es mindestens zwei *verschiedene* Spieler gibt, die noch nicht gegeneinander gespielt haben. (2 Pkte)

$\varphi :=$

(ii) Geben Sie eine Formel ψ der Logik erster Stufe über der Signatur σ an, die in \mathfrak{A} aussagt, dass es keinen Spieler gibt, der in allen Spielen, an denen er teilgenommen hat, gewonnen hat. (2 Pkte)

$\psi :=$

(iii) Beschreiben Sie umgangssprachlich, was die folgende Formel in \mathfrak{A} aussagt: (2 Pkte)

$$\forall x \forall y \left(\dot{S}(x, y) \rightarrow (\dot{G}(x, y) \leftrightarrow \neg \dot{G}(y, x)) \right)$$

(b) Sei $\sigma := \{\dot{E}\}$ eine Signatur mit einem 2-stelligen Relationssymbol \dot{E} . Geben Sie für die Formel (6 Pkte)

$$\varphi(x, y) := \left(\dot{E}(x, y) \wedge \exists z(\dot{E}(y, z) \wedge \dot{E}(z, x)) \right)$$

zwei σ -Interpretationen $\mathcal{I}_1, \mathcal{I}_2$ an, so dass \mathcal{I}_1 die Formel φ erfüllt und \mathcal{I}_2 die Formel φ *nicht* erfüllt.

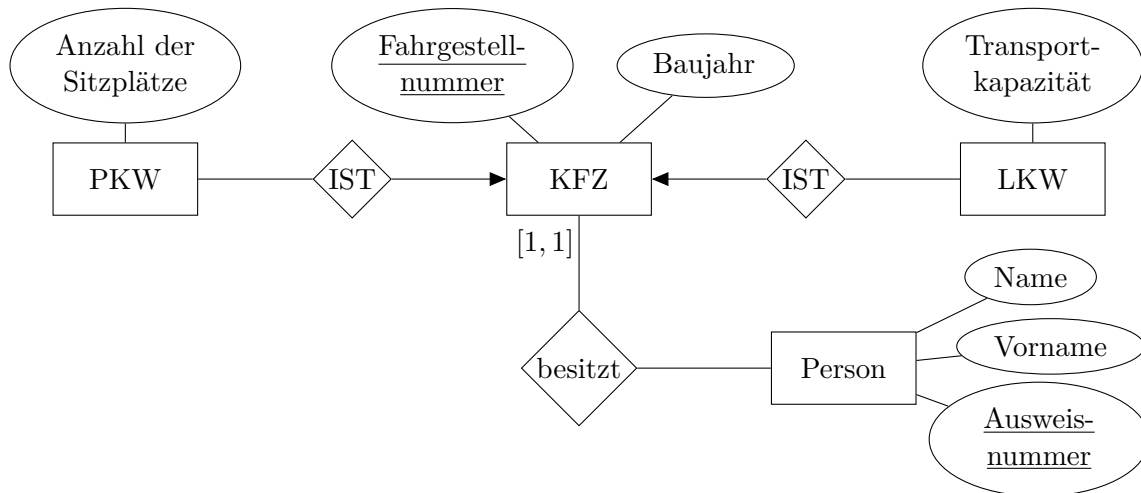
$\mathcal{I}_1 :=$

$\mathcal{I}_2 :=$

Aufgabe 4:**(11 Punkte)**

(a) Es sei folgendes Entity-Relationship-Modell gegeben:

(4 Pkte)



Welche der folgenden Aussagen sind im Sinne des oben angegebenen Entity-Relationship-Modells wahr, welche nicht?

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

- (i) Es ist möglich, dass es ein KFZ ohne Besitzer gibt. wahr falsch
- (ii) Jedes KFZ hat genau einen Besitzer. wahr falsch
- (iii) Es ist möglich, dass es ein PKW (z.B. mit Baujahr 2002) und einen LKW (z.B. mit Baujahr 1997) gibt, die dieselbe Fahrgestellnummer besitzen. wahr falsch
- (iv) Es ist möglich, dass es zwei Personen mit demselben (Nach)namen und demselben Vornamen gibt. wahr falsch

(b) Sei $G = (T, N, S, P)$ die kontextfreie Grammatik mit

- der Menge der Terminalsymbole $T = \{a, b, c, d\}$
- der Menge der Nichtterminalsymbole $N = \{S, X, Y\}$
- dem Startsymbol S
- der Menge der Produktionen $P = \{S \rightarrow XcYd, X \rightarrow aY, Y \rightarrow aY, Y \rightarrow bY, Y \rightarrow \varepsilon\}$

(i) Geben Sie für jedes der folgenden Wörter an, ob es zu der von G erzeugten Sprache $L(G)$ gehört. Geben Sie außerdem für jedes Wort, das zur Sprache $L(G)$ gehört, einen Ableitungsbaum an. (4 Pkte)

- *abcbad*

gehört zu $L(G)$? ja nein

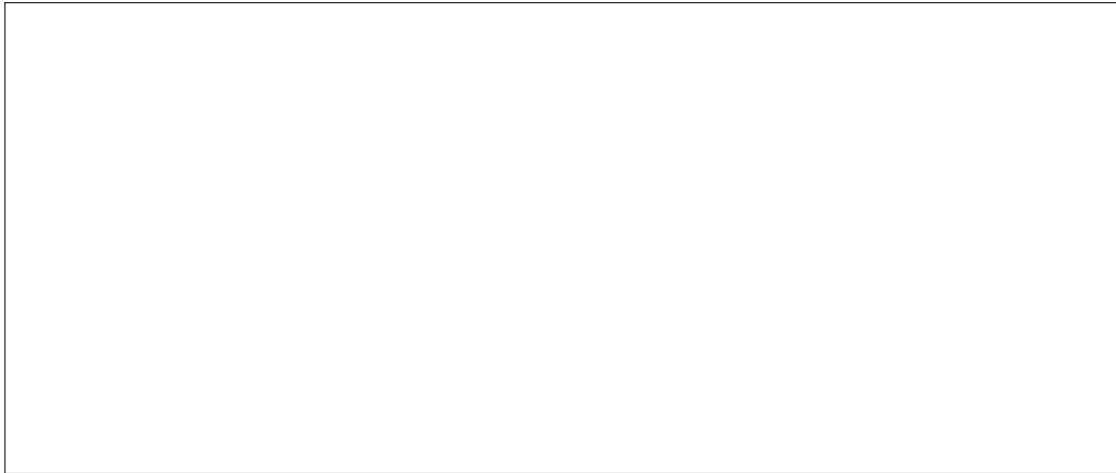
Falls das Wort zu $L(G)$ gehört, geben Sie hier einen Ableitungsbaum an:

- *bacabad*

gehört zu $L(G)$? ja nein

Falls das Wort zu $L(G)$ gehört, geben Sie hier einen Ableitungsbaum an:

- (ii) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache (3 Pkte) $L(G)$ an, die von der oben angegebenen Grammatik G erzeugt wird.



Aufgabe 5:**(15 Punkte)**

Die Sprache SFR über dem Alphabet $\Sigma := \{a, b, \emptyset, \sim, |, \cdot, (,)\}$ sei wie folgt rekursiv definiert:

Basisregeln: (B1) Das Symbol \emptyset ist in SFR.
 (B2) Die Symbole a und b sind in SFR.

Rekursive Regeln: (R1) Ist $w \in \text{SFR}$, so ist auch $\sim w$ in SFR.
 (R2) Sind w_1 und w_2 in SFR, so sind auch $(w_1 | w_2)$ und $(w_1 \cdot w_2)$ in SFR.

(a) Welche der folgenden Wörter gehören zur Sprache SFR, welche nicht? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in SFR?	
$((\sim \emptyset \cdot a) \cdot (\sim \emptyset \cdot b))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$\sim((\sim \emptyset \cdot a) \cdot b) b$	<input type="checkbox"/> ja	<input type="checkbox"/> nein
$(\sim(\sim \emptyset \cdot a) ((b \cdot a) \cdot \emptyset))$	<input type="checkbox"/> ja	<input type="checkbox"/> nein

(b) Für jedes Wort $w \in \Sigma^*$ bezeichne $s(w)$ die Anzahl der Vorkommen der Symbole \emptyset, a, b in w und $o(w)$ die Anzahl der Symbole $\sim, |$ und \cdot in w . Zum Beispiel gilt für $w = ((a \cdot \sim b) | \sim a)$, dass $s(w) = 3$ und $o(w) = 4$.

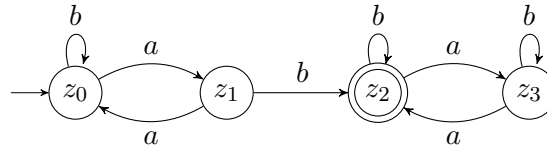
Beweisen Sie durch Induktion, dass für alle Wörter $w \in \text{SFR}$ gilt: $s(w) \leq o(w) + 1$.

(c) Geben Sie eine kontextfreie Grammatik G an, die genau die Sprache SFR erzeugt.

(4 Pkte)

Aufgabe 6:**(20 Punkte)**(a) Sei A der folgende deterministische endliche Automat:

(3 Pkte)

Welche der folgenden Wörter werden von A akzeptiert, welche nicht?

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... wird akzeptiert?	
<i>abbabb</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>babaab</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein
<i>aababaa</i>	<input type="checkbox"/> ja	<input type="checkbox"/> nein

(b) Es sei R der folgende reguläre Ausdruck über dem Alphabet $\Sigma := \{0, 1, \dots, 9, .\}$:

$$(\varepsilon \mid (1 \mid \dots \mid 9)(0 \mid 1 \mid \dots \mid 9)^*) \cdot (0 \mid 1 \mid \dots \mid 9)(0 \mid 1 \mid \dots \mid 9)^*$$

(i) Welche der folgenden Wörter liegen in der von R definierten Sprache $L(R)$, welche nicht? (3 Pkte)

Kreuzen Sie alle richtigen Antworten an. Für jedes korrekte Kreuz bekommen Sie einen Punkt, für jedes **falsche Kreuz** wird **ein Punkt abgezogen**. Die Gesamtpunktzahl ist aber mindestens 0.

Wort	... liegt in $L(R)$?	
.075	<input type="checkbox"/> ja	<input type="checkbox"/> nein
12	<input type="checkbox"/> ja	<input type="checkbox"/> nein
012.56637	<input type="checkbox"/> ja	<input type="checkbox"/> nein

(ii) Geben Sie eine (mathematische oder umgangssprachliche) Beschreibung der Sprache $L(R)$ an, die von R definiert wird. (3 Pkte)

- (c) Geben Sie die graphische Darstellung eines nicht-deterministischen endlichen Automaten an, (4 Pkte) der genau diejenigen Wörter über dem Alphabet $\{a, b\}$ akzeptiert, die die Teilwörter ab und ba enthalten, so dass ba mindestens einmal hinter einem Vorkommen von ab steht. D.h. der Automat soll genau die Wörter $w \in \{a, b\}^*$ akzeptieren, für die es Wörter $x, y, z \in \{a, b\}^*$ gibt, so dass $w = xabybaz$.

- (d) Betrachten Sie das Alphabet $\Sigma := \{a, b\}$ und zeigen Sie, dass die Sprache (7 Pkte)

$$L := \{w \in \{a, b\}^* : w \text{ enthält genau so viele } as \text{ wie } bs\}$$

nicht regulär ist.

Hinweis: Verwenden Sie das Pumping-Lemma aus der Vorlesung:

Sei Σ ein endliches Alphabet. Für jede *reguläre* Sprache $L \subseteq \Sigma^*$ gibt es eine Zahl $z \in \mathbb{N}$, so dass für jedes Wort $x \in L$ der Länge $|x| \geq z$ gilt: Es gibt eine Zerlegung von x in Worte $u, v, w \in \Sigma^*$, so dass die folgenden vier Bedingungen erfüllt sind:

- (i) $x = uvw$
- (ii) $|uv| \leq z$
- (iii) $|v| \geq 1$
- (iv) für jedes $i \in \mathbb{N}$ gilt: $uv^i w \in L$.
(d.h.: $uw \in L, uvw \in L, uvvw \in L, uvvww \in L, \dots$)

Literaturverzeichnis

- [1] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the web. *ACM Transactions on Internet Technology*, 1(1):2–43, 2001.
- [2] Albrecht Beutelspacher. “*Das ist o.B.d.A. trivial!*”. *Tipps und Tricks zur Formulierung mathematischer Gedanken*. Vieweg Studium, Braunschweig, 2002.
- [3] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.
- [4] P. P. Chen. The Entity-Relationship-Model — Towards a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976.
- [5] Reinhard Diestel. *Graphentheorie*. Springer-Verlag, Berlin, 2006.
- [6] Heinz-Dieter Ebbinghaus. *Einführung in die Mengenlehre*. Spektrum Akademischer Verlag, Heidelberg Berlin, 2003.
- [7] Ayman Farahat, Thomas LoFaro, Joel C. Miller, Gregory Rae, and Lesley A. Ward. Authority rankings from HITS, PageRank, and SALSA: Existence, uniqueness, and effect of initialization. *SIAM Journal on Scientific Computing*, 27(4):1181–1201, 2006.
- [8] William Feller. *An Introduction to Probability Theory and Its Applications: Volume I*. Wiley, 3rd edition, 1968. ISBN: 978-0-471-25708-0.
- [9] Martin Grohe. *Theoretische Informatik I*. Skript zur Vorlesung am Institut für Informatik, Humboldt-Universität zu Berlin, 2007.
- [10] J. Y. Halpern, R. Harper, N. Immerman, P. G. Kolaitis, M. Y. Vardi, and V. Vianu. On the unusual effectiveness of logic in computer science. *The Bulletin of Symbolic Logic*, 7(2):213–236, June 2001.
- [11] Andreas Heuer and Gunter Saake. *Datenbanken: Konzepte und Sprachen*. MITP-Verlag, 2. Auflage edition, 2000.
- [12] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.
- [13] Olle Häggström. *Finite Markov chains and algorithmic applications*. Number 52 in London Mathematical Society Student Texts. Cambridge University Press, 2002. ISBN-10: 0521890012.
- [14] Stasys Jukna. *Crashkurs Mathematik für Informatiker*. Leitfäden der Informatik. Teubner Verlag, 2008. ISBN 978-3-8351-0216-3.
- [15] Uwe Kastens and Hans Kleine Büning. *Modellierung. Grundlagen und formale Methoden*. Carl Hanser Verlag, München, 2005.

- [16] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [17] Martin Kreuzer and Stefan Kühling. *Logik für Informatiker*. Pearson Studium, München, 2006.
- [18] Amy N. Langville and Carl D. Meyer. *Google's Pagerank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, 2006.
- [19] L. Lovász, J. Pelikán, and K. Vesztergombi. *Discrete Mathematics*. Springer Science+Business Media, LLC, New York, 2003.
- [20] Zohar Manna and Richard Waldinger. *The logical basis for computer programming*. Addison-Wesley, 1985.
- [21] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [22] Christoph Meinel and Martin Mundhenk. *Mathematische Grundlagen der Informatik. Mathematisches Denken und Beweisen - Eine Einführung*. B.G. Teubner, Stuttgart, 2000.
- [23] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66 (previous number: SIDL-WP-1999-0120), Stanford InfoLab, November 1999. The article is available from <http://ilpubs.stanford.edu:8090/422/>.
- [24] Wolfgang Reisig. *Petrinetze*. Springer-Verlag, Berlin, 1982.
- [25] Georg Schnitger. *Internet Algorithmen*. Skript zur Vorlesung am Institut für Informatik, Goethe-Universität Frankfurt am Main, 2009.
- [26] Uwe Schöning. *Theoretische Informatik – kurzgefasst*. Spektrum Akademischer Verlag, Heidelberg, 2001.
- [27] Uwe Schöning. *Logik für Informatiker*. Spektrum Akademischer Verlag, Heidelberg, 2005.
- [28] Ingo Wegener. *Kompodium Theoretische Informatik – eine Ideensammlung*. B.G. Teubner, Stuttgart, 1996.
- [29] Ingo Wegener. *Theoretische Informatik*. B.G. Teubner, Stuttgart, 1999.