

Mathematische Grundlagen der Informatik

(Comment 1:) Für diese Version muss vor einer Veröffentlichung noch eine Vielzahl von Elementen (Grafiken) mit Ursprung von Lucia in eigener Version erstellt werden!

Andreas Leiser

<https://www.andreasleiser.ch>

mail@andreasleiser.ch

Inhaltsverzeichnis

Vorwort	v
I Logik, Mengenlehre, Beweistechniken	1
1 Logik	3
1.1 Motivierung – Wieso Logik?	5
1.2 Weitere Beispiele von Anwendungsbereichen der Logik	6
1.3 Zentraler Zusammenhang der Logik mit der “Mengenlehre”	6
1.4 Aussagelogik	7
1.4.1 Was ist eine logische Aussage?	7
1.4.2 Aussagenlogische Verknüpfungen (Operatoren)	9
1.4.3 Aussagelogische Formeln	13
1.4.4 Syntax und Semantik von aussagelogischen Formeln	15
1.4.5 Rechenregeln für aussagenlogische Formeln; Umformungen	19
1.4.6 Normalformen	21
1.4.7 Modelle für eine aussagenlogische Formel	26
1.4.8 Erfüllbarkeitseigenschaften aussagelogischer Formeln	27
1.5 Prädikatenlogik	29
1.5.1 Von Aussagen zu Prädikaten	29
1.5.2 Prädikate und Quantoren	30
1.5.3 Rechenregeln für Quantoren (für prädikatenlogische Formeln); Umformungen	34
2 Mengenlehre	39
2.1 Motivierung – Wieso Mengenlehre?	41
2.2 Was ist eine Menge? Begriffe, Beispiele	41
2.3 Mengenoperationen	48
2.4 Gesetze der Mengenalgebra und ihre Anwendung	52
2.5 Weiteres zum kartesischen Produkt von Mengen	54
2.6 Die Partition einer Menge	55
2.7 Die Unendlichkeit von Mengen	59
2.8 Übersicht: Gleichheits- und Äquivalenznotationen	62
3 Beweistechniken	65
3.1 Einleitung: Beweise in der Mathematik	67
3.2 Beweisen versus Gegenbeispiel finden	67
3.3 Direkter Beweis	69
3.4 Beweis durch Kontraposition	72
3.5 Indirekter Beweis (“Widerspruchsbeweis”)	74
II Funktionen, Relationen, Graphen	77
4 Funktionen	79
4.1 Was ist eine Funktion?	81

4.2	Injectivitat, Surjektivitat, Bijektivitat, Monotonie	87
4.3	Verknpfung und Inversion von Funktionen	93
5	Relationen	97
5.1	Was ist eine Relation?	99
5.2	Homogene und heterogene binre Relationen	101
5.3	Reflexivitat, Symmetrie, Antisymmetrie, Transitivitat	102
5.4	Äquivalenzrelationen	104
5.5	Ordnungsrelationen	108
5.6	Operationen zwischen und mit Relationen	114
5.6.1	Verknpfung zweier binrer Relationen	114
5.6.2	Inversion einer binren Relationen	117
6	Graphen	121
6.1	Grundlagen	123
6.1.1	Gerichtete Graphen	123
6.1.2	Ungerichtete Graphen	128
6.1.3	(Ungerichtete) Multigraphen	134
6.2	Zusammenhang und Zusammenhangskomponenten	137
6.3	Modellierung von Anwendungsproblemen	140
6.3.1	Königsberger Brückenproblem	141
6.3.1.1	Kurzer Exkurs: Notwendige und hinreichende Bedingung	141
6.3.1.2	Das Königsberger Brückenproblem	142
6.3.1.3	Lösung des Königsberger Brückenproblems; Eulerbedingungen in (Multi)Graphen	145
6.3.1.4	Eulerwege und Eulerkreise in allgemeinen (Multi)Graphen	146
6.3.2	Bume, Spannbume und typische Anwendungen	150
6.3.2.1	Bume	150
6.3.2.2	Spannbume	153
6.3.2.3	Spannbume kantengewichteter Graphen	154
6.3.2.4	Algorithmen von Kruskal und Prim zur Lösung der Anwendungsaufgabe	155
6.4	Matrix-Darstellungen von Graphen	156
6.5	Graph- und Digraphisomorphismus	162
III	Vollstndige Induktion, Rekursion	165
7	Vollstndige Induktion	167
7.1	Einfhrung	169
7.1.1	Das Prinzip der vollstndigen Induktion	169
7.1.2	Summen- und Produktzeichen	172
7.1.2.1	Summenschreibweise	172
7.1.2.2	Produktschreibweise	174
7.2	Modell-Beispiel	175
7.3	Vollstndige Induktion bei Aussagen über Ungleichungen	176
7.4	Vollstndige Induktion bei allgemeinen Aussagen	176
7.5	Korrektheit eines Computer-Programmes	177
7.6	Einordnung der Beweismethode der vollstndigen Induktion	178
8	Rekursion	179
8.1	Was bedeutet "Rekursion"?	181
8.2	Zahlenfolgen – rekursiv und explizit	184
8.3	Explizite Darstellung von rekursiven Zahlenfolgen bestimmen	187
8.4	Was sind Rekursionsgleichungen?	190
8.5	Homogene lineare Rek.-gleichungen 1. Ordnung	194
8.5.1	Iterationsmethode	194

8.5.2 Lösungsformel für den Fall eines konstanten Koeffizienten $c(n) = c$	195
8.6 Inhomogene lineare Rekurrenzgleichungen 1. Ordnung	196
8.6.1 Iterationsmethode	196
8.6.2 Lösung einer (inhomogenen) linearen Rekursionsgleichung 1. Ordnung mit konstanten Koeffizienten und Störterm	197
8.6.3 Verallgemeinerung: Nicht konstanter Störterm	199
8.7 Homogene lineare Rekurrenzgleichungen 2. Ordnung	202
8.8 Inhomogene lineare Rekurrenzgleichungen 2. Ordnung	205
8.9 Variablentransformation	206
8.9.1 Einleitung	206
8.9.2 Variablentransformationen in Funktionen auf abzählbaren Mengen	206
8.9.3 Variablentransformation zur Analyse von Algorithmen	208
8.9.4 Ausblick: Erzeugendensumme, Master-Theorem	211
8.10 Übersicht Lösungsmethoden Rekursionsgleichungen	212
IV Formale Sprachen, endliche Automaten, Grammatiken	213
9 Einführung: Formale Sprachen	215
9.1 Motivierung	217
9.2 Alphabete und Wörter	217
9.3 Wortpotenzen	221
9.4 Sprachen	222
9.5 Entscheidungsproblem einer Sprache (Wortproblem)	223
10 Reguläre Sprachen, endliche Automaten	225
10.1 Motivation	227
10.2 Strukturelles Modell / Darstellung endlicher Automaten	228
10.3 Berechnungsmodell eines endlichen Automaten	231
10.4 Sprache eines endlichen Automaten	235
11 Formale Grammatiken	237
11.1 Gibt es nicht-reguläre Sprachen?	239
11.2 Kontextfreie Grammatiken	239
11.3 Mehr- und Eindeutigkeit bei kontextfreien Grammatiken	244
11.4 Sprach-Hierarchie zwischen regulären und kontextfreien Sprachen	245
11.5 Kontextfreie Grammatik für eine reguläre Sprache	246
11.6 Techniken für den Entwurf von kontextfreien Grammatiken	247
11.7 Zentrale Anwendungen in der Informatik	247
12 Anwendung: Reguläre Ausdrücke	249
12.1 Einführung und Motivierung	251
12.2 Syntax regulärer Ausdrücke	252
12.2.1 Die Standard-Syntax regulärer Ausdrücke	252
12.2.2 Erweiterte Syntax regulärer Ausdrücke	253
12.3 Semantik regulärer Ausdrücke	254
12.3.1 Sprache eines regulären Ausdrucks	254
12.3.2 Eigenschaften regulärer Ausdrücke	255
12.3.3 Äquivalenz DFA und RA	255
13 Reguläre Sprachen, 2. Teil	257
13.1 Übersicht über Darstellungsarten für reguläre Sprachen	259
13.2 Abschlusseigenschaften regulärer Sprachen	259
14 Die Chomsky-Hierarchie	263
14.1 Rückblick: Reguläre (oder Typ-3) und kontextfreie (oder Typ-2) Grammatiken und Sprachen	265

14.2 Kontextsensitive (oder Typ-1) Grammatiken und Sprachen	265
14.3 Unbeschränkte Grammatiken und rekursiv aufzählbare Sprachen (Typ-0)	266
14.4 Chomsky-Hierarchie	267
Appendix	271
Ergänzende Notizen	271
Symbolverzeichnis	275
Tabellenverzeichnis	276
Abbildungsverzeichnis	277
Code Listings	278
Bibliographie	279
Bildreferenzen	281

Vorwort

Dieses Skript ist die Grundlage für den Sto , welchen ich in meinem Unterricht behandle. Der Rahmen dafür war ursprünglich vor allem die Modulbeschreibung der FHNW für das dortige Modul "mgli". Danach hat sich dies aber natürlich weiterentwickelt. Dann haben vor allem die Vorlesungsmaterialien von Dr. Lucia Di Caro, FHNW, aus dem Herbstsemester 2017 Pate gestanden. Einiges entstammt auch den Vorlesungsunterlagen und -notizen während meines eigenen Studiums und ein paar hervorragender Bücher zum Thema. Grundsätzlich erwarte ich nicht, dass Studierende weitere Literatur benötigen. Im Literaturverzeichnis sind einige Texte aufgelistet, welche mir teilweise und in verschiedenem Umfang halfen dieses Skript bzw. die zusätzlichen Materialien zu erstellen. Allerdings möchte ich hier warnen, denn einige der dort genannten Bücher gehen sehr viel weiter und tiefer und haben zusätzlich einen unterschiedlichen Blickwinkel (z.B. den eines Mathematikstudierenden). Bei Interesse nach Vertiefung oder weil es zwischendurch vielleicht hilfreich sein kann, ein Thema in einer etwas anderen Darstellung zu studieren, bietet es sich an, eines dieser Werke zu konsultieren.

Ich möchte hervorheben, dass ein intensives Beschäftigen mit den Beispielen und insbesondere mit den Aufgaben entscheidend für ein Erreichen der Lern- und Kompetenzziele des Moduls ist. Die Aufgaben sind separat auf den Übungsblätter und haben als Ergänzung relativ ausführliche Musterlösungen. Das Studium der Musterlösungen alleine wird aber bei weitem nicht ausreichen die Lernziele zu erreichen! Aktives, selbständiges Beschäftigen mit Aufgaben ist unumgänglich.

Bedanken möchte ich mich ganz besonders bei Dr. Lucia Di Caro, FHNW, welche mir freundlicherweise ihre Vorlesungsunterlagen aus dem Frühlingssemester 2017 zur Verfügung gestellt hat. Ein beachtlicher Teil der Materialien in diesem Skript und in meinem Unterricht stammt von ihr. Im Weiteren waren viele interessante Gespräche mit ihr über die mathematischen Grundlagen der Informatik, das "mgli"-Modul an der FHNW und über Mathematik-/Informatik-Didaktik allgemein sehr hilfreich. Diese Gespräche sind an vielen Stellen auch später wieder in diese Unterlagen miteingeflossen. Dann bedanke ich mich auch ganz herzlich bei all den Studierenden in den vielen Semestern für mehrere konstruktive Vorschläge und Feedbacks zu meinen Materialien, welche zu Verbesserungen beigetragen haben. Im Hinblick auf das hier vorliegende neu verfasste Skript, bedanken ich mich ganz besonders bei den FHNW-Studierenden der Klasse 1Ib des Herbstsemesters 2021 (u.a. Janic Berger, Bianca Christen-Piekenbrock, Nik Irniger, Tino Heuberger, Lani Wagner) für das sehr genaue Studium und die etlichen Feedbacks.

Natürlich möchte ich aber auf alle Fälle betonen: Für alle Fehler und Oddities in meinen Unterlagen übernehme ich die alleinige Verantwortung!

Andreas Leiser
2022/07/08

Teil I

Logik, Mengenlehre, Beweistechniken

Kapitel 1

Logik

Kapitelinhalt

1.1	Motivierung – Wieso Logik?	5
1.2	Weitere Beispiele von Anwendungsgebieten der Logik	6
1.3	Zentraler Zusammenhang der Logik mit der “Mengenlehre”	6
1.4	Aussagelogik	7
1.4.1	Was ist eine logische Aussage?	7
1.4.2	Aussagenlogische Verknüpfungen (Operatoren)	9
1.4.3	Aussagelogische Formeln	13
1.4.4	Syntax und Semantik von aussagelogischen Formeln	15
1.4.5	Rechenregeln für aussagenlogische Formeln; Umformungen	19
1.4.6	Normalformen	21
1.4.7	Modelle für eine aussagenlogische Formel	26
1.4.8	Erfüllbarkeitseigenschaften aussagelogischer Formeln	27
1.5	Prädikatenlogik	29
1.5.1	Von Aussagen zu Prädikaten	29
1.5.2	Prädikate und Quantoren	30
1.5.3	Rechenregeln für Quantoren (für prädikatenlogische Formeln); Umformungen	34

L E R N Z I E L E :

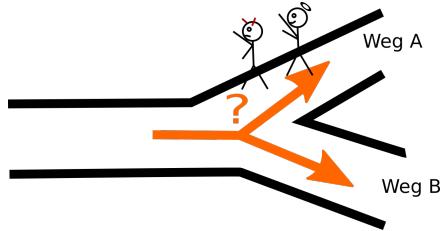
- LZ 1.1** Sie wissen, was eine logische Aussage ist.
 - LZ 1.2** Sie verstehen die aussagelogischen Operationen Disjunktion, Konjunktion, Negation, Implikation, Äquivalenz und können den Wahrheitsgehalt von zusammengesetzten aussagelogischen Formeln berechnen.
 - LZ 1.3** Sie können aussagelogische Formeln vereinfachen und die semantische Gleichheit von aussagelogischen Formeln untersuchen.
 - LZ 1.4** Sie können eine KNF/DNF einer aussagelogischen Formel bestimmen.
 - LZ 1.5** Sie kennen die Existenz- und Allquantoren der Prädikatenlogik.
 - LZ 1.6** Sie können Prädikate bilden und den Wahrheitsgehalt quantifizierter Aussagen bestimmen.
 - LZ 1.7** Sie können die Rechenregeln und Sätze der Prädikatenlogik anwenden.
-

1.1 Motivierung – Wieso Logik?

§1-1 Beispiel. (Waldspaziergang)

Als kleines Beispiel, wie man mit Logik gewisse Fragen beantworten kann, stellen wir uns folgende Situation vor:

- Wir treffen an einer Weggabelung im Wald zwei Personen.
- Die eine Person ist ein Lügner, die andre sagt immer die Wahrheit. Wir wissen aber nicht, welche Person lügt und welche nicht.
- Wir möchten nun wissen, welchen Weg wir nehmen sollen. Dabei wissen wir, dass beide Personen den richtigen Weg kennen.
- Wir haben nur eine Frage zur Verfügung, welche die beiden Personen je mit "ja" ("wahr") oder "nein" ("falsch") beantworten werden.



Das zu lösende Problem lautet also:

Welche Frage stellen wir, damit wir wissen, ob wir Weg A oder Weg B nehmen müssen um an unser Ziel zu kommen?

Beachten Sie: Die Fragen "Muss ich Weg A nehmen?" oder "Muss ich Weg B nehmen?" sind *keine* Lösungen für unser Problem!

Lösung:

Folgende Frage an die beiden Personen erlaubt es uns den richtigen Weg zu erfahren:

"Was würde dein Kollege auf die Frage antworten, ob Weg A der richtige Weg ist?"

Wieso und wie ist dies eine Lösung? Um dies zu sehen, unterscheiden wir zwei Fälle, je nachdem, welches tatsächlich der (unbekannte) richtige Weg ist:

1. Fall: Weg A ist richtig. Dann gilt:

- Der Lügner sagt "falsch" (weil er lügt und weil der Wahrheitssager "wahr" sagt, da er die Wahrheit sagt).
- Der Wahrheitssager sagt auch "falsch" (weil er die Wahrheit sagt und der Lügner lügt, also "falsch" sagt).

2. Fall: Weg A ist nicht richtig (d.h., B ist richtig). Dann gilt:

- Der Lügner sagt "wahr" (weil er lügt und weil der Wahrheitssager "falsch" sagt, da er die Wahrheit sagt).
- Der Wahrheitssager sagt auch "wahr" (weil er die Wahrheit sagt und der Lügner lügt, also "wahr" sagt).

Wir sehen nun: Was auch immer die beiden Antworten sind, wir müssen einfach das *Gegenteil* davon nehmen und haben die gesuchte richtige Antwort auf die Frage, ob Weg A der richtige sei! Was lernen wir noch? Als Logiker finden wir aus einem Wald heraus und wir finden uns auch in grösseren Wäldern zurecht! :-)

◊

Damit Sie sehen, dass wir nicht nur solche "anekdotalen" Probleme mit Logik lösen können, haben wir im folgenden Abschnitt ein paar Beispiele von Anwendungsbereichen der Logik aufgelistet.

1.2 Weitere Beispiele von Anwendungsgebieten der Logik

In der Informatik sind unter anderem folgende Anwendungsgebiete von Logik interessant bzw. zentral:

- Wie Sie möglicherweise wissen, besteht Hardware im Innersten – in den “logischen Schaltkreisen” – aus “logischen Gattern”. Diese sind nichts anderes, als Logik, welche elektronisch/technisch realisiert ist! (0 und 1 bzw. falsch und wahr)
- Als Programmierer benutzen Sie in vielen Sprachkonstrukten logische Ausdrücke (z.B. Schleifenbedingungen).
- Bei der formalen Spezifikation und der formalen Verifikation versucht man Hard- und Software vor allem in Bereichen mit höchster Sicherheits- (Verlässlichkeits-) Stufe, mittels Logik “fehlerfrei” zu machen.
- Relationale Datenbanken sind im wesentlichen Konstrukte der Logik und der Mengenlehre.
- Expertensysteme, künstliche Intelligenz \leadsto Automatisierung logischer (vernünftiger) Schlussfolgerungen zur Verarbeitung von Information und künstlicher Gewinnung von “Wissen”.
- Automatische und/oder computerunterstützte Beweissysteme (für Hardware-Verifikation, für Grundlagen der Logik und der Mathematik, usw.)
- Bei der Sprachverarbeitung geht es u.a. um die Modellierung und Analyse von natürlicher Sprache mithilfe formaler Sprachen.

Logik spielt aber auch in vielen anderen Bereichen eine Rolle, manchmal auch eher versteckt: Dies ist zum Beispiel prinzipiell in der Formalisierung der Wissenschaften der Fall, in der Philosophie, in der Medizin (Expertensysteme), im Bereich von “Business Intelligence”, in Teilen der Kriminologie/Kriminalistik, in der Logik in den Rechtswissenschaften, usw....

1.3 Zentraler Zusammenhang der Logik mit der “Mengenlehre” (folgt nach der Logik)

Es gibt einen sehr zentralen Zusammenhang zwischen der Logik und dem Thema des nächsten Kapitels, der Mengenlehre. Wie sieht dieser Zusammenhang aus? Das verstehen wir, wenn wir uns die zentralen Ziele der beiden Gebiete vor Augen führen.

Die Logik formalisiert und macht präzise “wie” man über etwas exakt/korrekt spricht (Aussagen macht) und denkt. Sie beschäftigt sich also mit “exakter, unzweideutiger Sprache” bzw. mit dem “exakten, unzweideutigen Denken”.

Gibt es dasselbe auch für die Dinge “worüber” man spricht? Ja! Genau dafür ist die Mengenlehre da, denn dies ist das Ziel der Mengenlehre. Wir merken uns folgenden zentralen Zusammenhang zwischen der Logik und der “Mengenlehre”:

In der “Mengenlehre” wird dann formalisiert, “worüber” man mithilfe der Logik exakt spricht oder denkt, d.h., worüber man formale Aussagen macht. (Dieses “worüber” wird auch “Diskursuniversum” genannt.)

In der “Mengenlehre” sollen also die Dinge und Gegenstände worüber gesprochen

wird, präzisiert und unzweideutig formalisiert werden.

Daraus folgt auch, dass wir die "formalen Wissenschaften" (wie die Mathematik, weite Teile der Informatik, ... usw.) folgendermassen sehen können:

**Formale Wissenschaft(en) = Logik (Formalisierung der Sprache) + Mengenlehre
(Formalisierung der Gegenstände worüber man spricht)**

Dies ist das Fundament von moderner Mathematik, Informatik und der theoretischen und formalisierten Bereiche aller Wissenschaften!^[E.1]

1.4 Aussagelogik

1.4.1 Was ist eine logische Aussage?

§1-2 Definition. (Logische Aussage)

Eine logische Aussage ist ein umgangssprachlicher oder formalisiert ausgedrückter Satz, von welchem man sinnvollerweise sagen kann, er sei wahr oder falsch und zwar ohne dies im konkreten Fall auch wissen zu müssen. ◇

Was meinen wir damit, dass "man sinnvollerweise sagen kann, die Aussage sei wahr oder falsch, ohne dies im konkreten Fall auch wissen zu müssen"? Wir meinen damit, dass zum Beispiel die Aussage "Der Stern Alpha Centauri wird von Planeten umkreist." prinzipiell entweder wahr oder falsch und damit eine logische Aussage ist. Und dies, obwohl wir – oder zumindest nicht jeder Mensch und schon gar nicht zu jedem Zeitpunkt der Menschheitsgeschichte – unter Umständen nicht wissen, ob die Aussage wahr oder falsch ist. Den sogenannten Wahrheitswert (wahr oder falsch) müssen wir also nicht kennen um entscheiden zu können, ob es sich bei einer Aussage oder einem Satz um eine logische Aussage im oben definierten Sinn handelt.

§1-3 Definition. (Zusammengesetzte Aussage (aussagelogische Formel))

Eine zusammengesetzte Aussage oder auch aussagenlogische Formel ist eine logische Aussage, welche aus einzelnen Teilaussagen, die selber logische Aussagen sind, besteht. ◇

Lässt sich jedoch eine logische Aussage nicht in kleinere Teile, welche selbst wieder logische Aussagen sind, unterteilen, so nennen wir diese:

§1-4 Definition. (Elementaraussage (atomare Aussage bzw. Formel))

Eine Elementaraussage oder atomare Aussage oder auch atomare Formel ist eine logische Aussage, die nicht aus mehreren Teilaussagen besteht. ◇

Mathematik zum Beispiel, besteht also im wesentlichen aus Sätzen in Form von logischen Aussagen über – vorher präzise definierte – einzelne mathematische Objekte. Für das Modul legen wir nun folgende Konvention fest: Wir meinen mit “Aussagen” immer solche präzise definierten logischen Aussagen, wenn nicht was anderes gesagt wird. Zudem, wie schon angedeutet, bezeichnen wir die beiden Werte “wahr” und “falsch” als die sogenannten Wahrheitswerte oder die logischen Konstanten. Übliche Abkürzungen dafür sind w/f oder 1/0 oder *true/false*. Dabei benutzen wir in diesem Modul wie in der Informatik üblich vorzugsweise

1 (für wahr),

0 (für falsch).

Jetzt wollen wir uns kurz anhand von ein paar Beispielen vergegenwärtigen, welche umgangssprachlichen Aussagen (Sätze) auch logische Aussagen bzw. eben gerade nicht logische Aussagen sind.

§1-5 Beispiele. (Logische Aussagen)

(1) A_1 : “Heute ist Freitag.”

Dies ist eine logische Aussage. Allerdings zeigt sich schon hier das Phänomen, dass der Kontext und die beteiligten Begriffe exakt spezifiziert sein müssen. Im Fall von A_1 hängt die Antwort natürlich davon ab, wann ich die Frage stelle. Mehr noch, auch wo ich zu einem festgelegten Zeitpunkt die Frage stelle, kann den Wahrheitswert beeinflussen. Denken Sie nur an die Datumsgrenzen: Es ist möglich, dass zu einem bestimmten Zeitpunkt an einem Ort der Wahrheitswert 1 und an einem anderen Ort der Wahrheitswert 0 ist.

(2) A_2 : “Die Sonne kreist um die Erde.”

Dies ist ebenfalls eine logische Aussage und bekannterweise eine falsche.

(3) A_3 : “Heute ist schönes Wetter.” (w?) (f?) \sim subjektiv!

Dies ist eine logische Aussage. Auch hier merken wir bei etwas nachdenken, dass aber dafür der Begriff “schönes Wetter” exakt definiert sein muss (z.B. durch eine physikalisch-meteorologische Definition), so dass wir tatsächlich von einer Aussage sprechen können, die entweder “wahr” oder “falsch” ist.

(4) A_4 : “3 teilt 1'330'647.”

Dies ist eine (wahre) logische Aussage.

(5) A_5 : “Wenn es regnet, dann bleiben die Straßen trocken.”

Dies ist eine (falsche) logische Aussage. Allerdings auch hier, nur je nach Kontext: In einem Tunnel zum Beispiel ist sie wahr. Auch muss der zeitliche und örtliche Referenzpunkt klar sein.

(6) A_6 : “Wenn der Hahn kräht auf dem Mist, dann ändert sich das Wetter, oder es bleibt, wie es ist.”

Auch dies ist eine logische Aussage. Sogar eine, welche immer wahr ist. (Wir werden dies später eine Tautologie nennen.)

(7) A_7 : “ $\pi + e$ ist transzendentale Zahl”.

Dies ist ein ziemlich spezieller Fall: Die Antwort ist unbekannt. Mehr noch, sie ist nur dann eine logische Aussage, falls dieses mathematische Problem tatsächlich entscheidbar ist. Und dies weiß man heute noch nicht, da es ein offenes Problem ist!



§1-6 Beispiele. (Gegenbeispiele: Keine logischen Aussagen)

(1) "x ist eine Primzahl^[E.2]".

Dies ist eine "Aussageform", da "x" nicht genau bestimmt ist, aber *keine Aussage!* (Siehe später: "Prädikat".)

(2) "Wie geht es Dir?"

Eine Frage ist keine logische Aussage.

(3) "Dieser Satz ist falsch." (!)

Dieser Satz stellt ein logisches Paradox dar, da logisch nie auflösbar ist, ob er "wahr" oder "falsch" ist. Also *keine logische Aussage*.

(4) "Wu , sitz!"

Dies ist eine andere umgangssprachliche Aussage (hier ein Befehl), die ebenfalls *keine logische Aussage* ist.



Ganz analog wie in der Algebra oder allgemein in der Mathematik definieren wir:

§1-7 Definition. (Aussagevariable)

Ein Symbol wie z.B. A , B , ..etc, welches für eine beliebige aussagelogische Aussage steht, bezeichnen wir als **Aussagenvariable**.



Im Fall von zusammengesetzten Aussagen, also aussagelogischen Formeln, benutzen wir auch oft kleine Buchstaben wie f, g, h , oder auch f_1, f_2, \dots . Grundsätzlich sind wir aber natürlich bei der Wahl solcher Symbole frei. Wir müssen sie jeweils nur zu Beginn klar deklarieren.

1.4.2 Aussagenlogische Verknüpfungen (Operatoren)

Ein wichtiger Grund für die Mächtigkeit der Logik als Werkzeug ist, dass wir logische Aussagen mittels logischer Operatoren miteinander verknüpfen können und so immer komplexere Aussagen bauen können. Das heisst letztlich, dass so auch sehr komplexe Information formal und präzise verarbeitet werden kann.

§1-8 Definition. (Fundamentale aussagenlogische Verknüpfungen (logische Operatoren))

Seien A und B zwei beliebige logische Aussagen.

Die drei folgenden fundamentalen aussagenlogische Verknüpfungen (logische Operatoren) liefern jeweils wieder logische Aussagen und sind mit den definierenden Wahrheitstabellen festgelegt:

A	B	$A \wedge B$	$A \vee B$	$\neg A$ bzw. \overline{A}
0	0	0	0	1
0	1	0	1	-
1	0	0	1	0
1	1	1	1	-

- \wedge heisst das logische "und" bzw. die (logische) Konjunktion und wird als "A und B" gesprochen.

Beispiel: Seien folgende zwei Aussagen gegeben: $A :=$ "Der Hund bellt." und $B :=$ "Der Hahn krähnt.". Dann ist die logische Konjunktion dieser beiden Aussagen die neue Aussage

$$(A \wedge B) = \text{"Der Hund bellt und der Hahn kräht."}$$

- \vee ist das logische "oder" bzw. die (logische) Disjunktion und wird als "A oder B" gesprochen.

Beachten Sie hier insbesondere in der letzten Zeile die "1" und damit den Unterschied zum umgangssprachlichen "oder" (dem "entweder oder") was dann der XOR -Operation ([§1-9](#)) entspricht.

Beispiel: Seien wieder $A :=$ "Der Hund bellt." und $B :=$ "Der Hahn krähnt.". Dann ist die logische Disjunktion dieser beiden Aussagen die neue Aussage

$$(A \vee B) = \text{"Der Hund bellt oder der Hahn kräht."}$$

(Und diese Aussage ist also auch wahr, wenn sowohl der Hund bellt als auch der Hahn kräht!)

- $\neg A$ oder \overline{A} schliesslich ist die logische Verneinung, auch logische Negation genannt, und wird "nicht A" gesprochen.

Beispiel: Sei $A :=$ "Der Hund bellt.". Dann ist die logische Verneinung dieser Aussage die neue Aussage

$$\neg A = \overline{A} = \text{"Der Hund bellt nicht."}$$

◊

Aussagelogische Operatoren nennt man manchmal auch Konnektoren oder Junktoren. Mit diesen drei fundamentalen Operatoren lassen sich auch weitere abgeleitete Operatoren definieren:

§1-9 Definition. (Abgeleitete logische Operatoren)

Seien A und B zwei beliebige logische Aussagen.

Dann sind weiter folgende, fundamentale Operatoren, abgeleitete logische Operatoren genannt, definiert:

A	B	$A \implies B$ oder: $\neg A \vee B$	$A \iff B$	\vee auch: XOR , $\dot{\vee}$, \oplus
0	0	1	1	0
0	1	1	0	1
1	0	0	0	1
1	1	1	1	0

- \implies heisst aussagelogische Implikation und wird "wenn A, dann B" bzw. "aus A folgt B" bzw. "A impliziert B" gesprochen. $A \implies B$ ist selbst also wieder eine logische Aussage, definiert durch obige Wahrheitstabelle.

Bei der Implikation wird A Prämisse (Voraussetzung) genannt und B die Konklusion (Folgerung).

Beispiel: Seien wieder folgende zwei Aussagen gegeben: $A :=$ "Der Hund bellt." und $B :=$ "Der Hahn krähnt.". Dann ist die logische Implikation die neue Aussage

$$(A \implies B) = \text{"Wenn der Hund bellt, dann kräht der Hahn."}$$

- \iff heisst aussagelogische Äquivalenz und wird "A genau dann, wenn B" gesprochen und auch "A gdw. B" bzw. engl. "A i B" geschrieben. $A \iff B$ ist selbst wieder eine logische Aussage.

Beispiel: Für die beiden Aussagen $A :=$ "Der Hund bellt." und $B :=$ "Der Hahn krähnt." ist die logische Äquivalenz dieser beiden Aussagen die neue Aussage

$$(A \iff B) = \text{"Der Hund bellt genau dann, wenn der Hahn kräht."}$$

- \vee heisst XOR-Operation und $A \vee B$ wird "A X-OR B" gesprochen. Auch dies ist selber wieder eine logische Aussage.

Beispiel: Für die beiden Aussagen $A :=$ "Der Hund bellt." und $B :=$ "Der Hahn krähnt." ist die XOR-Operation dieser beiden Aussagen die neue Aussage

$$(A \vee B) = \text{"Entweder bellt der Hund, oder der Hahn kräht."}$$

◊

Wie bereits gesagt, können nun mehrere bzw. viele dieser Verknüpfungen in einem "logischen Ausdruck" stehen:

$$A \vee B \vee C \vee D, \quad (A \implies B) \wedge (B \implies A), \quad \text{etc. ...}$$

Für eine längere Aneinanderreihung entweder nur von logischen Konjunktionen oder nur von logischen Disjunktionen definieren wir noch folgende Schreibweise.

§1-10 Definition. (Verallgemeinerte Konjunktion und Disjunktion)

Seien beliebige n Aussagen A_1, \dots, A_n gegeben. Dann schreiben wir abkürzend:

$$\begin{aligned} \bigwedge_{i=1}^n A_i &:= A_1 \wedge A_2 \wedge \dots \wedge A_n, \\ \bigvee_{i=1}^n A_i &:= A_1 \vee A_2 \vee \dots \vee A_n. \end{aligned}$$

◊

Bei gewissen Kombinationen von logischen Operatoren müssen wir aufpassen, dass das, was gemeint ist, auch eindeutig ist. Schauen wir uns dazu ein Beispiel an:

§1-11 Beispiel. $((\neg A) \vee B)$ ist nicht logisch äquivalent mit $\neg(A \vee B)$

Sei folgender Ausdruck gegeben:

$$\neg A \vee B$$

Wenden wir zuerst $\neg A$ an und verknüpfen danach das Ergebnis davon mittels \vee mit der Aussage B , so ist dies im allgemeinen nicht dasselbe, wie wenn wir zuerst $A \vee B$ bilden und danach das Ergebnis logisch verneinen. Dies können wir durch den Vergleich der beiden Wahrheitstabellen sehen:

A	B	$\neg A$	$(\neg A) \vee B$	$A \vee B$	$\neg(A \vee B)$
0	0	1	1	0	1
0	1	1	1	1	0
1	0	0	0	1	0
1	1	0	1	1	0

Die vierte und die sechste Spalte sind nicht identisch. D.h., die beiden logischen Ausdrücken bedeuten nicht dasselbe.

Auch durch ein konkretes Beispiel können wir dies verstehen: Seien die beiden Aussagen

$$A := \text{"Madrid ist Hauptstadt von Frankreich."},$$

$$B := \text{"Die Aare fliesst durch Bern."}$$

gegeben. Dann ist es offensichtlich nicht dasselbe, ob ich sage

$$(\neg A) \vee B = \text{"Madrid ist nicht Hauptstadt von Frankreich oder die Aare fliesst durch Bern."},$$

oder ob ich sage,

$$\neg(A \vee B) = \text{"Es gilt nicht, dass Madrid die Hauptstadt von Frankreich ist oder die Aare durch Bern fliesst."}.$$

Die erste zusammengesetzte Aussage ist nämlich wahr, die zweite jedoch falsch. ◊

Unser Beispiel zeigt also: Damit bei logischen Ausdrücken mit mehreren Verknüpfungen keine Zweideutigkeiten entstehen, müssen wir eine Reihenfolge, wir sagen auch Bindungsprioritäten, für die Operatoren definieren:

§1-12 Definition. (Regeln der Bindungspriorität bei aussenlogischen Verknüpfungen)

- \neg bindet eine Aussagenvariable am stärksten.
- \wedge und \vee binden am zweitstärksten. Untereinander sind sie gleichberechtigt.
- \Rightarrow und \Leftrightarrow bilden am drittstärksten. Untereinander sind sie wieder gleichberechtigt.

Zum "Durchbrechen" dieser Regeln werden Klammern benutzt. Teile in der (innersten) Klammer werden immer zuerst ausgewertet. ◊

§1-13 Beispiele. (Bindungsprioritäten und Klammerungen)

- (1) $\neg A \vee B$ ist dasselbe wie $(\neg A) \vee B$. Ebenso ist $\neg A \wedge B$ dasselbe wie $(\neg A) \wedge B$. Die Klammern können also in beiden Fällen weggelassen werden.
- (2) $A \wedge (B \vee C)$ ist nicht dasselbe wie $(A \wedge B) \vee C$. Die Klammer muss in beiden Fällen also stehen.
- (3) $\neg A \implies B$ ist nicht dasselbe wie $\neg(A \implies B)$. Auch hier muss im zweiten Fall die Klammer stehen.
(Wenn ich aber $(\neg A) \implies B$ meine, kann ich hier die Klammer wieder weglassen.)

◊

1.4.3 Aussagenlogische Formel

Interessant sind jetzt natürlich nicht nur einzelne atomare Aussagen, sondern vor allem auch solche, welche komplizierter aus mehreren logischen Operatoren verknüpft sind (“zusammengesetzte Aussagen”). Nun wollen wir solche “aussagenlogische Formeln” noch präzise definieren:

§1-14 Definition. (Aussagenlogische Formel)

Sei Var eine Ansammlung von atomaren Aussagen (Atome, atomare Formeln) A, B, C, \dots . Eine Formel f der Aussagenlogik (AL) über Var ist “induktiv” (\leadsto später!) definiert durch:

- Induktionsbeginn: Jede atomare Aussage A, B, C, \dots aus der Gesamtheit Var ist Formel, genannt eine **atomare Formel**.
- Induktionsschritt: Sind f und g Formeln der Aussagenlogik über Var , dann sind es auch

$$\begin{aligned} &\neg f, \\ &(f \wedge g), \\ &(f \vee g). \end{aligned}$$

◊

Eine aussagenlogische Formel wird manchmal auch (aussagenlogischer) Ausdruck genannt. Allerdings ist dies manchmal auch nicht ganz korrekt, weil man im Zusammenhang mit der Syntax und “wohlgeformten Formeln” eigentlich einen Ausdruck (beliebige endliche Zeichenkette über dem Alphabet der Aussage logik) und wohlgeformten Formeln (syntaktisch, d.h., strukturell korrekte Formel) unterscheidet. Zum Beispiel ist dann $A \vee \wedge B$ zwar ein aussagenlogischer Ausdruck, aber syntaktisch eben nicht korrekt und damit keine (wohlgeformte) Formel. Eine so feine Unterscheidung wollen wir jedoch in diesem Modul nicht machen und wir unterschieden die Begriffe “Ausdruck” und “Formel” nicht so detailliert.

Kommen wir aber noch einmal zu obiger Definition einer aussagenlogischen Formel zurück: Wieso werden in der Definition nur die Verknüpfungen \neg, \wedge, \vee benutzt? Wo sind \implies, \iff, \dots ?

Die Antwort ist simpel aber auch wichtig: Wie teilweise schon gesehen, lassen sich die letzten Operatoren alle mit den drei Operatoren \neg, \wedge, \vee darstellen! Das heißt, in der Definition reicht es deshalb aus nur diese drei “wesentlichen” Operationen zu benutzen.

Zum Schluss noch ein paar grundlegende Definitionen von Begriffen und Notationen, welche immer wieder benutzt werden.

§1-15 Definition. (Gesamtheit aller aussagenlogischer Formeln)

Die Gesamtheit aller aussagenlogischer Formeln bezeichnen wir mit \mathcal{F}_{AL} . \diamond

§1-16 Definition. (Gesamtheit aller atomaren Formeln (Aussagevariablen) einer aussagenlogischen Formel)

Sei eine aussagenlogische Formel f gegeben.

Die Gesamtheit aller atomarer Formeln (= aller atomarer Aussagen, aller Aussagevariablen) in der Formel f bezeichnen wir mit $Var(f)$. \diamond

§1-17 Beispiel.

Sei $f := \neg((A \vee B) \implies C)$. Dann besteht $Var(f)$ aus den Atomen A, B, C .

\diamond

§1-18 Definition. (Literal)

Ein Literal ist entweder eine atomare Formel (eine einzelne Aussagevariable) oder die Negation einer solchen.

\diamond

§1-19 Beispiel.

In der Formel $f := \neg((A \vee \neg B) \implies (C \wedge B))$ sind die Literale: $A, B, \neg B, C$.

\diamond

Wir benutzen noch sehr oft folgenden Grundbegri:

§1-20 Definition. (Belegung von atomaren und aussagenlogischen Formeln)

- (1) Sei mit Var eine gewisse Anzahl von atomaren Formeln A, B, C, \dots bezeichnet.

Eine Belegung \mathcal{A} von Var ist eine Zuordnung, die jeder atomaren Formel A, B, C, \dots einen Wahrheitswert 1 (true) oder 0 (false) zuordnet.

- (2) Sei $Var(f)$ eine Anzahl von atomaren Formeln A, B, C, \dots einer aussagenlogischen Formel f .

Eine Belegung $\mathcal{A}(f)$ der aussagenlogischen Formel f ist dann eine Belegung der Menge $Var(f)$ all ihrer Variablen.

\diamond

§1-21 Beispiel.

Sei

$$f := (A \implies (B \vee \neg C)).$$

$Var(f)$ besteht also aus A, B, C . Dann ist zum Beispiel durch

$$\begin{aligned} A(f) : \quad & A \mapsto 1, \\ & B \mapsto 0, \\ & C \mapsto 0 \end{aligned}$$

eine Belegung von f definiert. ◊

1.4.4 Syntax und Semantik von aussagelogischen Formeln

Da die Logik das Ziel hat Aussagen und damit letztlich Teile der Sprache exakt, unzweideutig und korrekt zu fassen, gibt es auch hier – analog aller “natürlicher Sprachen” – Aspekte der Syntax und der Semantik. Beide Aspekt haben wir bei unseren Betrachtungen schon gestreift. Jetzt wollen wir sie noch etwas genauer anschauen und einander gegenüberstellen. Zuerst einmal: Was sagt der Duden?

§1-22 Definition. (Syntax; (nach Duden))

“Syntax” = lateinisch *syntaxis*; griechisch *sýntaxis*, eigentlich = Zusammenstellung, aus: *sýn* = zusammen und *táxis* = Ordnung.

1. In einer Sprache übliche Verbindung von Wörtern zu Wortgruppen und Sätzen; korrekte Verknüpfung sprachlicher Einheiten im Satz
 2. Lehre vom Bau des Satzes als Teilgebiet der Grammatik; Satzlehre
 3. wissenschaftliche Darstellung der Syntax
- ◊

§1-23 Definition. (Semantik; (nach Duden))

“Semantik” zu griechisch *sēmantikós* = bezeichnend, zu: *sēmaínein* = bezeichnen, zu: *sēma*, Sem (= kleinste Komponente einer Wortbedeutung)

1. Teilgebiet der Linguistik, das sich mit den Bedeutungen sprachlicher Zeichen und Zeichenfolgen befasst
 2. Bedeutung, Inhalt (eines Wortes, Satzes oder Textes)
- ◊

Als gut handhabbare (und merkbare) Definition können wir dies für uns folgendermassen festlegen:

§1-24 Definition. (Syntax und Semantik in der Logik)

- **Syntax** = Regeln wie elementare Zeichen (Symbole) – oder auch Symbolgruppen – korrekt zusammengesetzt werden (dürfen). \leadsto Struktur aussagenlogischer Formeln.
Frage: Ist die logische Formel formal (d.h., strukturell) korrekt?
- **Semantik** = Bedeutungslehre; Bedeutung von Zeichen (Symbole), Wörtern und Wortteile, Sätzen und Satzteile. \leadsto Bedeutung (“Interpretation”), Wahrheit aussagenlogischer Formeln.
Frage: Was bedeutet die logische Formel?

◊

Damit ist auch klar, wie wir die syntaktische Gleichheit und die semantische Gleichheit zweier aussagelogischer Formeln auseinanderhalten müssen:

§1-25 Definition. (Syntaktische Gleichheit von aussagenlogischen Formeln)

Zwei Formeln f und g sind **syntaktisch gleich**, wenn sie aus exakt derselben Folge von Zeichen (Symbolen) bestehen.

Notation:

- Syntaktische Gleichheit wird als

$$f = g$$

geschrieben.

- Wenn wir eine Formelvariable, zum Beispiel f , syntaktisch als dasselbe wie eine gegebene Formel g definieren, dann stellen wir dem Gleichheitszeichen manchmal auch ein Doppelpunkt voran,

$$f := g.$$

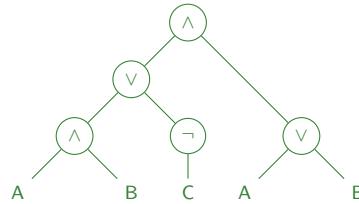
Dies wird dann "per Definition" oder "ist definiert als/durch" gelesen.

◊

Die Syntax einer aussagenlogischen Formel wird mittels eines sogenannten Ableitungsbäums (Syntaxbaum, Syntaxdiagramm, Parsebaum) dargestellt. Dabei beginnt man mit dem am wenigsten geklammerten Operationssymbol in der sogenannten "Wurzel" des Baumes und entwickelt dann die Struktur der Formel "rekursiv" nach unten, indem man die entsprechenden weiteren Symbole wieder als Knoten schreibt und sie mit Kanten verbindet. Ganz zu unterst, schreibt man dann die entsprechenden Aussagevariablen hin.

§1-26 Beispiel. (Ableitungsbau für eine aussagenlogische Formel)

$$f = ((A \wedge B) \vee (\neg C)) \wedge (A \vee B)$$



◊

Solche Syntaxbäume werden uns im Kapitel "Sprache, Grammatiken und Automaten" und in späteren Informatikmodulen wieder begegnen. Eine typische Anwendung findet sich zum Beispiel in Compiler von Programmiersprachen.

Die Semantik einer aussagenlogischen Formel wird mittels der uns bereits bekannten Wahrheitstabellen dargestellt. (Diese definieren ja gerade, was sie bedeuten, d.h., wann sie inhaltlich wahr und wann falsch sind.)

§1-27 Definition. (Sematische Gleichheit zweier aussagenlogischer Formeln)

Zwei aussagenlogische Formeln f und g heissen **semantisch gleich** oder **äquivalent** (symbolisch $f \equiv g$), wenn ihre Wahrheitstabellen identisch sind. ◊

Beachten Sie: Wir brauchen hier ein anderes Symbol für die semantische Äquivalenz (hier von aussagenlogischen Formeln), um sie vom logischen Operator der Äquivalenz von Aussagen (\iff), welche ja selbst in einer aussagenlogischen Formel vorkommen kann, unterscheiden zu können.

Ob zwei aussagelogische Formeln dasselbe bedeuten können wir also durch Vergleich ihrer beiden Wahrheitstabellen feststellen:

§1-28 Beispiel. (Sematische Äquivalenz zweier Formeln für die logische Äquivalenz, d.h., für den Äquivalenzoperator)

Zeigen Sie, dass die logische Äquivalenz $A \iff B$ aus anderen Verknüpfungsoperatoren abgeleitet bzw. mittels diesen anderen dargestellt werden kann. Vergleichen Sie dazu die Wahrheitstabelle des Ausdrucks $(A \Rightarrow B) \wedge (B \Rightarrow A)$ mit der Wahrheitstabelle der Äquivalenz.

Lösung:

Nach Definition der logischen Äquivalenz haben wir folgende Wahrheitstabelle (vgl. auch Def. §1-9):

A	B	$A \iff B$
0	0	1
0	1	0
1	0	0
1	1	1

Die Wahrheitstabelle des anderen Ausdrucks erhalten wir schrittweise ebenfalls aus den Wahrheitstabellen der Definitionen:

A	B	$A \Rightarrow B$	$B \Rightarrow A$	$(A \Rightarrow B) \wedge (B \Rightarrow A)$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	1	1	1

Die Spalten ganz rechts der beiden Wahrheitstabellen sind gleich, also sind die Formeln semantisch äquivalent! Wir können also schreiben:

$$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \wedge (B \Rightarrow A))$$

Diese semantische Äquivalenz hat übrigens in der Mathematik eine wichtige Anwendung als **Beweistechnik** (siehe auch später Kap. 3, Abschnitt 3.4). Oft ist es einfacher für den Beweis einer Äquivalenz “die beiden Richtungen separat” je als Implikationen zu zeigen! D.h., anstatt $A \Leftrightarrow B$ (in einem Schritt) zu zeigen, zeigt man $A \Rightarrow B$ und $B \Rightarrow A$ separat. Aufgrund der oben gerade gezeigten semantischen Äquivalenz hat man dann aber tatsächlich auch $A \Leftrightarrow B$ gezeigt. \diamond

Wir können Ableitungsbäume nicht nur benutzen, um die Syntax einer Formel darzustellen, sondern auch um die Semantik einer Formel für eine konkrete Variablenbelegung zu ermitteln. Dazu schreiben wir im Syntaxbaum unten die Wahrheitswerte der konkreten Variablenbelegung hin und werten Schritt-für-Schritt nach oben gehend die Knoten aus, bis wir die Auswertung der Wurzel erreicht haben. Diese entspricht dann der Auswertung der Formel für diese Belegung.

§1-29 Beispiel. (Darstellung der Semantik einer Formel in einem Ableitungsbau)

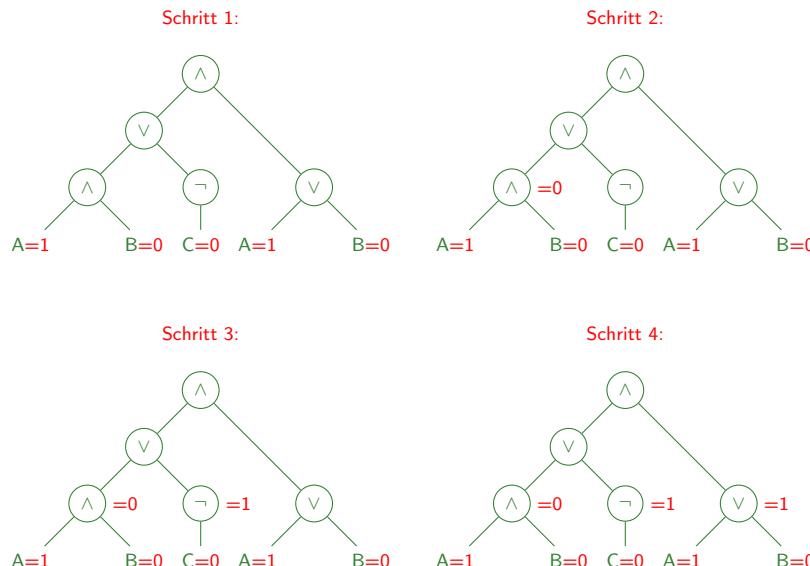
Wir stellen die Semantik der Formel

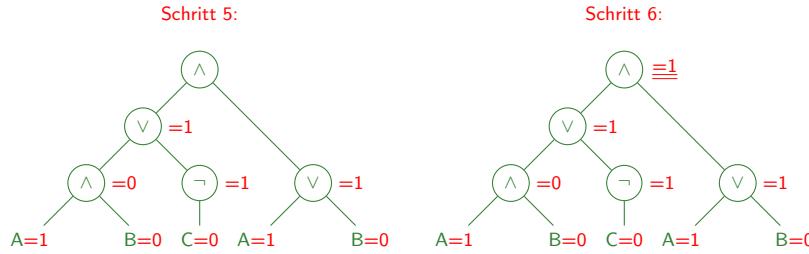
$$f = (((A \wedge B) \vee (\neg C)) \wedge (A \vee B))$$

als Ableitungsbau dar und “werten” ihn für den Fall

$$A = 1, B = 0, C = 0$$

aus:





Also ist das Ergebnis von f für diese Belegung 1, was sich leicht auch direkt verifizieren lässt. ◇

"Im Prinzip" können wir sogar Ableitungsbäume dazu benutzen, um die gesamte Semantik einer Formel darzustellen. Wir würden dazu jedoch viele Ableitungsbäume gebrauchen, nämlich für jede mögliche Variablenbelegung einen Ableitungsbau. (Wieviele also im Beispiel?) Das wird schon bei kleineren Formeln und wenigen Aussagevariablen sehr unpraktikabel und deshalb ist die Wahrheitstabelle dafür bedeutend besser geeignet. Obwohl auch diese schnell einmal sehr gross werden kann.

§1-30 Beispiel. (Darstellung der (gesamten) Semantik einer Formel mit Wahrheitstabelle)

Die gesamte Semantik, d.h., die Auswertung der Formel für alle möglichen Wahrheitswerte stellen wir am besten mittels Wahrheitstabelle dar:

A	B	C	$f = (((A \wedge B) \vee (\neg C)) \wedge (A \vee B))$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Eine Wahrheitstabelle eignet sich ebenfalls relativ gut, um eine etwas komplexere aussagelogische Formel zu definieren. ◇

1.4.5 Rechenregeln für aussagenlogische Formeln; Umformungen

Zwei grundlegende Aufgaben im Umgang mit logischen Formeln ist die Vereinfachung von aussagenlogischen Formeln und die Überprüfung, ob zwei Formeln gleich sind. Wir werden im nächsten Abschnitt auch zwei sogenannte Normalformen kennenlernen: Dies sind zu einer gegebenen Formel semantisch äquivalente Formen, welche von spezieller, einfacher Struktur sind. Jedoch sind diese oft nicht minimal in ihrer Länge und damit nicht "einfach" aus Sicht der Länge. Eine semantisch äquivalente und möglichst kurze Form erhält man normalerweise durch Anwendung der in diesem Abschnitt betrachteten Rechenregeln für aussagelogische Formeln. Ebenso kann oft mit diesen Rechenregeln die zweite erwähnte Aufgabe, der Test zweier Formeln auf Äquivalenz, am besten erledigt werden.

Grundsätzlich ist die zweite Aufgabe immer auch mittels Wahrheitstabellen lösbar. Dies ist aber sehr schnell einmal nicht mehr das geeignete Vorgehen, da die Tabellen eben relativ rasch unhandlich gross und unübersichtlich werden. Wir brauchen also ein "Kalkül", d.h., Rechenregeln, mit denen wir direkt mit den Formeln arbeiten können. (Computertechnisch kann die Situation aber natürlich gerade umgekehrt sein, z.B. wenn die Formeln bereits als – wenn auch sehr grossen, Tabellen – verfügbar sind.)

Für besagtes "Kalkül" ist folgendes Theorem mit den elementaren Rechenregeln zentral. Zuerst halten wir aber noch die folgenden zwei wichtigen "konstanten Formeln" in einer Definition fest:

§1-31 Definition. (Tautologie *true* und Kontradiktion (Widerspruch) *false*)

Wir definieren für beliebige aussagelogische Formeln f , die beiden "konstanten" Formeln

$$\text{true} := f \vee \neg f$$

(auch Tautologie genannt) und

$$\text{false} := f \wedge \neg f$$

(auch eine unerfüllbare Formel (Kontradiktion, Widerspruch) genannt, d.h., welche immer wahr bzw. immer falsch ergeben. \diamond

§1-32 Theorem. (Elementare Rechenregeln für aussagenlogische Formeln)

Seien f, g, h beliebige aussagenlogische Formeln.

Namen:	Regel/Gesetz:
Idempotenz-G.	$f \wedge f \equiv f$ $f \vee f \equiv f$
Kommutativ-G.	$f \wedge g \equiv g \wedge f$ $f \vee g \equiv g \vee f$
Identitäts-G.	$f \wedge \text{true} \equiv f$ $f \vee \text{false} \equiv f$ $f \wedge \text{false} \equiv \text{false}$ $f \vee \text{true} \equiv \text{true}$
Assoziativ-G.	$(f \wedge g) \wedge h \equiv f \wedge (g \wedge h)$ $(f \vee g) \vee h \equiv f \vee (g \vee h)$
Absorptions-G.	$f \wedge (f \vee g) \equiv f$ $f \vee (f \wedge g) \equiv f$
Distributivitäts-G.	$f \wedge (g \vee h) \equiv (f \wedge g) \vee (f \wedge h)$ $f \vee (g \wedge h) \equiv (f \vee g) \wedge (f \vee h)$
De Morgan Gesetze	$\neg(f \wedge g) \equiv \neg f \vee \neg g$ $\neg(f \vee g) \equiv \neg f \wedge \neg g$
Negations-G.	$\neg\neg f \equiv f$

(Beweis jeweils durch Vergleich der Wahrheitstabellen.) \diamond

§1-33 Beispiel. (Vereinfachung mittels der Rechenregeln)

Seien $f := (A \vee \neg B) \wedge (A \vee B)$ und $g := A$.

Gilt dann $f \equiv g$? D.h., gilt

$$(A \vee \neg B) \wedge (A \vee B) \equiv A?$$

und f lässt sich zu A vereinfachen?

Ja, denn wir können mit den Rechenregeln und der Def. der Kontradiktion umformen:

Beweis.

$$\begin{aligned}
 & (A \vee \neg B) \wedge (A \vee B) \\
 & \equiv A \vee (\neg B \wedge B) && \text{(Erstes Distributiv-G. von links nach rechts gelesen.} \\
 & && \text{Das entspricht dem bekannten "Ausklammern".)} \\
 & \equiv A \vee \text{false} && \text{(Def. Kontradiktion)} \\
 & \equiv A && \text{(Identitäts-G.)}
 \end{aligned}$$

□
◊

1.4.6 Normalformen

Zur manuellen, aber vor allem zur technischen oder computersoftware-gestützten Verarbeitung von (komplexeren) aussagenlogischen Formeln sind strukturell einfach gebaute Formeln sehr nützlich und die natürlich zu den jeweils betrachteten komplexeren Formeln semantisch äquivalent sein sollen. Die zwei Wichtigsten sind: die “konjunktive Normalform” (KNF) und die “disjunktive Normalform” (DNF). Zu ihrer Einführung benötigen wir auch noch die beiden, auch für sich wichtigen, Grundbegriffe der logischen Klausel und des logischen Monoms.

§1-34 Definition. ((Logische) Klausel; Konjunktive Normalform (KNF) einer Formel f)

- (1) Eine Klausel der konjunktiven Normalform oder einfach nur (logische) Klausel ist eine aussagenlogische Formel der Form

$$(L_1 \vee L_2 \vee \dots \vee L_m)$$

für ein $m \geq 1$, wobei alle L_i Literale sind. (Zur Erinnerung, Literale sind atomare Formeln (Aussagevariablen) oder Negationen von solchen.)

- (2) Eine aussagenlogische Formel f hat konjunktive Normalform (KNF), wenn sie in folgender Form ist:

$$f = f_1 \wedge f_2 \wedge \dots \wedge f_n$$

für ein $n \geq 1$, wobei für alle $1 \leq i \leq n$ die f_i selber logische Klauseln sind (also von der Form $f_i = (L_{i,1} \vee L_{i,2} \vee \dots \vee L_{i,m_i})$, $m_i \geq 1$, für Literale $L_{i,j}$ sind).

◊

§1-35 Definition. ((Logisches) Monom; Disjunktive Normalform (DNF) einer Formel f)

- (1) Eine Klausel der disjunktiven Normalform oder kurz (logisches) Mo-

nom ist eine aussagenlogische Formel der Form

$$(L_1 \wedge L_2 \wedge \dots \wedge L_m)$$

für ein $m \geq 1$, wobei alle L_i Literale sind.

- (2) Eine aussagenlogische Formel f hat **disjunktive Normalform (DNF)**, wenn sie in folgender Form ist:

$$f = f_1 \vee f_2 \vee \dots \vee f_n$$

für ein $n \geq 1$, wobei für alle $1 \leq i \leq n$ die f_i selber logische Monome sind (also von der Form $f_i = (L_{i,1} \wedge L_{i,2} \wedge \dots \wedge L_{i,m_i})$, $m_i \geq 1$, für Literale $L_{i,j}$ sind).

◊

Diese beiden Definitionen einer KNF bzw. einer DNF erscheinen etwas "kryptisch". Man kann sie sich aber etwas vereinfacht folgendermassen merken, wenn L_{ij} beliebige Literale und die n, m_i natürliche Zahlen grösser 0 sind:

Eine Formel f hat **KNF**, genau dann wenn $f = \bigwedge_{i=1}^n \bigvee_{j=1}^{m_i} L_{ij}$.

Eine Formel f hat **DNF**, genau dann wenn $f = \bigvee_{i=1}^n \bigwedge_{j=1}^{m_i} L_{ij}$.

Das heisst bei der konjunktiven Normalform ist die erste Operation die Konjunktion $\bigwedge_{i=1}^n$, gefolgt im Innern von "der anderen" Operation (hier die Disjunktion), welche Literale verknüpft. Und bei der disjunktiven Normalform ist es gerade umgekehr und damit ist die erste Operation entsprechend jetzt die Disjunktion $\bigvee_{i=1}^n$, gefolgt im Innern von "der anderen" Operation (jetzt die Konjunktion), welche Literale verknüpft.

Am besten studiert man dies auch noch an ein paar Beispielen:

§1-36 Beispiele. (Normalformen)

Seien im folgenden $A, B, C, \dots, T, S, \dots$ immer atomare Formeln.

- (1) $(T \vee S) \wedge A \wedge (D \vee \neg E \vee F) \wedge (\neg B) \wedge (\neg F \vee \neg S)$ ist eine KNF.
- (2) $\overline{B} \vee (A \wedge B \wedge \overline{E}) \vee (\overline{A} \wedge F) \vee C \vee \overline{S} \vee (\overline{B} \wedge S \wedge \overline{T} \wedge D)$ ist eine DNF.

(Zur Erinnerung: Für die Negation $\neg A$ können wir auch \overline{A} schreiben.)

- (3) $(\neg B) \vee (\neg C \wedge G) \wedge (A \vee \neg F \vee C) \vee (A \wedge S)$ ist weder eine KNF noch eine DNF.
- (4) $(A \vee \neg F \vee D) \wedge (A \implies B) \wedge (C \vee E) \wedge (\neg S)$ ist auch keine KNF und keine DNF.

◊

Es stellt sich natürlich die Frage, wann bzw. ob immer eine gegebene Formel in eine KNF bzw. DNF umgewandelt werden kann. In der Logik kann folgendes wichtiges Ergebnis bewiesen werden:

§1-37 Theorem.

Jede aussagenlogische Formel f lässt sich in KNF und in DNF bringen. ◊

Jetzt wo wir wissen, dass es zu einer beliebigen gegebenen aussagelogischen Formel immer sowohl eine semantisch äquivalente KNF als auch eine semantisch äquivalente DNF gibt, stellt sich die Frage, wie man diese jeweils erhält. Wir wollen zwei Verfahren kennen- und anwenden lernen:

§1-38 Definition. (Bestimmung von KNF bzw. DNF)

- Vorgehen 1: (Wahrheitstabelle)

0. (Nicht zwingend: f ev. mittels Rechenregeln in eine weniger komplexe Form bringen.)
1. Wahrheitstabelle der Formel f erstellen.
2. KNF oder DNF aus der Wahrheitstabelle ablesen:
 - a. Aufgrund der Identitätsgesetze ($f \wedge true \equiv f$ und $f \vee false \equiv f$) sind nur folgende Zeilen relevant:

KNF: Zeilen mit Wahrheitswert 0 für f .

DNF: Zeilen mit Wahrheitswert 1 für f .

- b. Jede Zeile entsprechend der Belegung und dem notwendigen Wahrheitswert mittels Literalen hinschreiben:

KNF: Kette von mit \vee verknüpften Literalen.

DNF: Kette von mit \wedge verknüpften Literalen.

- c. Die einzelnen Zeilen entsprechend der Normalform zusammenstellen.
Dies liefert die gesuchte Normalform f_{KNF} bzw. f_{DNF} von f .

- Vorgehen 2: (Vollständig mit Rechenregeln)

1. Alle Negationen, die nicht zu Literalen gehören eliminieren (Anwendung von de Morgan und doppelte Negation).
2. Mit dem entsprechenden Distributivgesetz in KNF oder DNF umformen.

◊

Folgende Beispiele demonstrieren die beiden Verfahren im Detail:

§1-39 Beispiel. (KNF bestimmen mit Vorgehen 1)

Sei die aussagelogische Formel f gegeben, welche durch folgende Wahrheitstabelle definiert ist.

A	B	C	f (per Def.)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Da wir eine KNF bestimmen wollen, sind alle Variablenbelegungen (d.h., Zeilen) welche einen Wahrheitswert 1 (= *true*) liefern irrelevant. Weil beim Verknüpfen dieser Zeilen haben wir aufgrund des Identitätsgesetzes $f \wedge \text{true} \equiv f$ und wir können alle zu *true* ausgewerteten Zeilen weglassen. Also:

A	B	C	f (per Def.)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Da die einzelnen Zeilen als Literale (Elementaraussagen bzw. die Aussagevariablen oder ihre Verneinung) geschrieben werden können, ist es nun möglich, jede einzelne Zeile als oder-Kette zu schreiben und so den notwendigen Wahrheitswert von f zu erhalten:

A	B	C	f (per Def.)
0	0	0	0 $\sim(A \vee B \vee C)$
0	0	1	0 $\sim(A \vee B \vee C)$
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0 $\sim(\bar{A} \vee \bar{B} \vee C)$
1	1	1	1

Wir sehen, wir müssen die Literale so wählen, dass jede Belegung mit 1 zu einer 0 "geswitched" wird. Denn nur eine oder-Kette mit lauter 0 (= *false*) ist selber 0 (= *false*). Die so erhaltenen Terme können wir jetzt gemäss der Gestalt einer KNF noch mittels logischem "und" \wedge verknüpfen, was eine KNF liefert:

$$f_{\text{KNF}} = (A \vee B \vee C) \wedge (A \vee B \vee \bar{C}) \wedge (\bar{A} \vee \bar{B} \vee C)$$

◇

§1-40 Beispiel. (DNF bestimmen mit Vorgehen 1)

Sei wieder die aussagelogische Formel f von vorhin gegeben, welche durch die Wahrheitstabelle

A	B	C	f (per Def.)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

definiert ist. Da wir nun eine DNF bestimmen wollen, sind alle Variablenbelegungen (d.h., Zeilen) welche einen Wahrheitswert 0 (= *false*) liefern irrelevant. (Es dreht sich ja alles einfach um im Vergleich zur KNF.) Beim Verknüpfen dieser Zeilen haben wir so aufgrund des Identitätsgesetzes $f \vee \text{false} \equiv f$ und wir können alle zu *false* ausgewerteten Zeilen weglassen. Also:

A	B	C	f (per Def.)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Die einzelnen Zeilen dürfen wieder als Literale (Elementaraussagen bzw. die Aussagevariablen oder ihre Verneinung) geschrieben werden. Damit ist es hier möglich, jede einzelne Zeile als und-Kette zu schreiben und so den notwendigen Wahrheitswert von f zu erhalten:

A	B	C	f (per Def.)
0	0	0	0
0	0	1	0
0	1	0	1 $\rightsquigarrow (\bar{A} \wedge B \wedge \bar{C})$
0	1	1	1 $\rightsquigarrow (\bar{A} \wedge B \wedge C)$
1	0	0	1 $\rightsquigarrow (A \wedge \bar{B} \wedge \bar{C})$
1	0	1	1 $\rightsquigarrow (A \wedge \bar{B} \wedge C)$
1	1	0	0
1	1	1	1 $\rightsquigarrow (A \wedge B \wedge C)$

Wir sehen nun, dass wir die Literale so wählen müssen, dass jede Belegung mit 0 zu einer 1 "geschwichtet" wird. Hier erreichen wir auf diese Weise eine und-Kette, welche selber 1 (= true) ist. Die so erhaltenen Terme können wir jetzt gemäss der Gestalt einer DNF noch mittels logischem "oder" \vee verknüpfen, was eine DNF liefert:

$$f_{\text{DNF}} = (\bar{A} \wedge B \wedge \bar{C}) \vee (\bar{A} \wedge B \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge C) \vee (A \wedge B \wedge C)$$

◊

§1-41 Beispiel. (KNF bestimmen mit Vorgehen 2)

Sei folgende aussagelogische Formel gegeben:

$$f := \overline{(A \wedge \bar{B})} \vee ((C \vee \bar{A}) \wedge \overline{(\bar{B} \wedge C)})$$

Wir ermitteln nun eine KNF gemäss dem Vorgehen 2:

$$\begin{aligned}
 & \overline{(A \wedge \bar{B})} \vee ((C \vee \bar{A}) \wedge \overline{(\bar{B} \wedge C)}) \\
 & \equiv (\bar{A} \vee B) \vee ((C \vee \bar{A}) \wedge (B \vee \bar{C})) && \text{(de Morgan G., doppelte Neg.)} \\
 & \equiv (\bar{A} \vee B \vee C \vee \bar{A}) \wedge (\bar{A} \vee B \vee B \vee \bar{C}) && \text{(Distributiv-G.)} \\
 & \equiv (\underbrace{\bar{A} \vee \bar{A}}_{= \bar{A}} \vee B \vee C) \wedge (\bar{A} \vee \underbrace{B \vee B}_{= B} \vee \bar{C}) && \text{(noch vereinfachen)} \\
 & \equiv (\bar{A} \vee B \vee C) \wedge (\bar{A} \vee B \vee \bar{C}) \\
 & = f_{\text{KNF}}.
 \end{aligned}$$

◊

§1-42 Beispiel. (DNF bestimmen mit Vorgehen 2)

Wir ermitteln nun eine DNF gemäss dem Vorgehen 2 wiederum der aussagenlogische Formel

$$f := \overline{(A \wedge \overline{B})} \vee ((C \vee \overline{A}) \wedge \overline{(\overline{B} \wedge C)}).$$

Der erste Schritt ist noch identisch zum vorhergehenden Beispiel mit der KNF. Insgesamt erhalten wir zum Beispiel:

$$\begin{aligned} & \overline{(A \wedge \overline{B})} \vee ((C \vee \overline{A}) \wedge \overline{(\overline{B} \wedge C)}) \\ & \equiv \overline{A} \vee B \vee ((C \vee \overline{A}) \wedge (B \vee \overline{C})) && (\text{de Morgan G., doppelte Neg.}) \\ & \equiv \overline{A} \vee B \vee (((C \vee \overline{A}) \wedge B) \vee ((C \vee \overline{A}) \wedge \overline{C})) && (\text{Distributiv-G.}) \\ & \equiv \overline{A} \vee B \vee (C \wedge B) \vee (\overline{A} \wedge B) \vee \underbrace{(C \wedge \overline{C})}_{\equiv 0} \vee (\overline{A} \wedge \overline{C}) && (\text{Distributiv-G.}) \\ & \equiv \overline{A} \vee B \vee (C \wedge B) \vee (\overline{A} \wedge B) \vee (\overline{A} \wedge \overline{C}) && (\text{vereinfachen: Identitäts-G.}) \\ & = f_{\text{DNF}}. \end{aligned}$$

Das Vorgehen ist aber variabel und es gibt verschiedene Lösungen. Zum Beispiel:

$$\begin{aligned} & \overline{(A \wedge \overline{B})} \vee ((C \vee \overline{A}) \wedge \overline{(\overline{B} \wedge C)}) \\ & \equiv (\overline{A} \vee B) \vee ((C \vee \overline{A}) \wedge (B \vee \overline{C})) && (\text{de Morgan G., doppelte Neg.}) \\ & \equiv (\overline{A} \vee B \vee C \vee \overline{A}) \wedge (\overline{A} \vee B \vee B \vee \overline{C}) && (\text{Distributiv-G. anders}) \\ & \equiv (\underbrace{\overline{A} \vee \overline{A}}_{\equiv \overline{A}} \vee B \vee C) \wedge (\overline{A} \vee \underbrace{B \vee B}_{\equiv B} \vee \overline{C}) && (\text{zwischendurch vereinfachen: Komm-G., Idempotenz-G.}) \\ & \equiv (\overline{A} \vee B \vee C) \wedge (\overline{A} \vee B \vee \overline{C}) \\ & \equiv (\overline{A} \vee B) \vee \underbrace{(C \wedge \overline{C})}_{\equiv 0} && (\text{Distributiv-G.}) \\ & \equiv \overline{A} \vee B && (\text{Identitäts-G.}) \\ & = f_{\text{DNF}}. \end{aligned}$$

◊

1.4.7 Modelle für eine aussagenlogische Formel

Wertet eine Belegung $\mathcal{A}(f)$ eine Formel zu *true* aus, schreiben wir das auch als $\mathcal{A}(f) = \text{true}$ und im gegenteiligen Fall als $\mathcal{A}(f) = \text{false}$.

§1-43 Definition. (Modell für eine aussagenlogische Formel)

Ein Modell für eine aussagenlogische Formel oder wahrmachende Belegung für die Formel f ist eine Belegung $\mathcal{A}(f)$, welche die Formel f zu *true* auswertet, also wofür $\mathcal{A}(f) = \text{true}$ gilt. ◊

§1-44 Beispiel.

Sei

$$f := (A \implies (B \vee \neg C)).$$

$Var(f)$ besteht also aus A, B, C .

Dann ist die Belegung $\mathcal{A}_1(f)$ definiert durch

$$A \mapsto \text{true}, B \mapsto \text{false}, C \mapsto \text{false}$$

wahrmachend (wir schreiben $\mathcal{A}_1(f) = \text{true}$), d.h., es ist ein Modell für f .

Hingegen ist die Belegung \mathcal{A}_2 gegeben durch

$$A \mapsto \text{true}, B \mapsto \text{false}, C \mapsto \text{true}$$

nicht wahrmachend ($\mathcal{A}_2(f) = \text{false}$), d.h., kein Modell für f . ◊

1.4.8 Erfüllbarkeitseigenschaften aussagelogischer Formeln

Wir können aussagelogische Formeln nun aufgrund ihrer Erfüllbarkeitseigenschaft einteilen:

§1-45 Definition. (Klassifizierung aussagelogischer Formeln nach Erfüllbarkeit)

- (1) Eine aussagenlogische Formel f heisst erfüllbar, wenn sie mind. ein Modell besitzt, also mind. eine wahrmachende Belegung hat.

Beispiel: $A := \text{"Heute ist Freitag."}$,
 $f := \neg A = \text{"Heute ist nicht Freitag."}$

A	$f = \neg A$
0	1
1	0

- (2) Besitzt eine aussagelogische Formel f kein Modell, d.h., alle Belegungen sind falschmachend, dann nennen wir sie unerfüllbar oder einen Widerspruch bzw. eine Kontradiktion.

Beispiel: $A := \text{"Die Sonne scheint."}$,
 $f := A \wedge \neg A = \text{"Die Sonne scheint und die Sonne scheint nicht."}$

A	$f = A \wedge \neg A$
0	0
1	0

- (3) Ist jede Belegung für eine Formel f ein Modell (d.h., eine wahrmachende Belegung), so heisst die Formel eine Tautologie oder eine (allgemein)gültige Formel.

Beispiel: $A := \text{"Heute ist Freitag."}$,
 $f := A \vee \neg A = \text{"Heute ist Freitag oder heute ist nicht Freitag."}$

A	$f = A \vee \neg A$
0	1
1	1

- (4) Eine aussagenlogische Formel heisst falsifizierbar, wenn sie mind. eine falschmachende Belegung hat, d.h., mind. eine Belegung ist kein Modell.

Beispiel: $A := \text{"Heute ist Freitag."}$, $B := \text{"Heute regnet es."}$
 $f := A \wedge B = \text{"Heute ist Freitag und heute regnet es."}$

A	B	$f = A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

- (5) Eine aussagelogische Formel ist eine Neutralität (oder auch Kontingenz), wenn sie weder eine Tautologie noch eine Kontradiktion ist. (D.h., sie ist erfüllbar, aber nicht allgemeingültig.)

Beispiel: (wie vorhin in (4))

$A := \text{"Heute ist Freitag."}$, $B := \text{"Heute regnet es."}$
 $f := A \wedge B = \text{"Heute ist Freitag und heute regnet es."}$

A	B	$f = A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

◊

§1-46 Definition. (Die Probleme SAT und TAUT)

Die Frage, ob eine gegebene aussagenlogische Formel erfüllbar ist, wird in der Logik als Erfüllbarkeitsproblem oder SAT (von engl. satisfiability = "Erfüllbarkeit") bezeichnet.

Die Frage, ob eine gegebene aussagenlogische Formel eine Tautologie ist, wird in der Logik als Tautologieproblem oder TAUT bezeichnet.

◊

§1-47 Theorem. (Entscheidbarkeit von SAT und TAUT)

SAT und TAUT sind entscheidbare Probleme.

◊

D.h., man kann einen Algorithmus angeben, welcher in endlicher Zeit für jede beliebige aussagenlogische Formel f (aus der Gesamtheit \mathcal{F}_{AL}) entscheidet, ob f erfüllbar ist, d.h., f zu den SAT-Problemen gehört. Dasselbe ist möglich, um zu entscheiden, ob f Tautologie ist (gültig ist), d.h., ob f die Bedingung TAUT erfüllt.

Beweis. (siehe Informatikvorlesung → man benutze Wahrheitstabellen)

□

Eine wichtige abschliessende Bemerkung ist noch, dass man aber weiß, dass die beiden Probleme zu den "schwierigen" gehören. Stichwort:

"NP- und Co-NP-Vollständigkeit".

Das heißt, der Aufwand sie zu lösen, steigt mit wachsender Größe sehr stark an.

1.5 Prädikatenlogik

1.5.1 Von Aussagen zu Prädikaten

Wir haben in der Aussagenlogik Aussagen über mathematische Objekte gemacht:

§1-48 Beispiel.

$A := \text{"5 ist eine ungerade Zahl"}$. ◇

Wir haben im Weiteren verschiedene Aussagen verknüpft und wieder Aussagen erhalten:

§1-49 Beispiel.

$A := \text{"5 ist eine ungerade Zahl"}, B := \text{"5 ist kleiner 4"}$.

Dann ist die Aussage $A \wedge \neg B = \text{"5 ist ungerade und nicht kleiner 4"}$ wieder eine Aussage (und zwar eine wahre). ◇

Eine Aussage bestand also stets darin, etwas über die Eigenschaft eines Objektes zu sagen (oben über eine Eigenschaften der Zahl 5). Solche Eigenschaften werden in der Logik/Mathematik/Informatik Prädikate genannt und oft mit P bezeichnet.

§1-50 Beispiel.

$P := \text{"Eigenschaft einer ganzen Zahl ungerade zu sein"}$. ◇

Wir symbolisieren die Zuordnung eines Prädikats (einer Eigenschaft) P zu einem Objekt x , durch $P(x)$.

§1-51 Beispiel.

Wenn wir vorher P als die Eigenschaft einer ganzen Zahl ungerade zu sein spezifiziert haben, dann könnten wir z.B. für die Aussage

"Die Eigenschaft von 5 ungerade zu sein, ist wahr."

schreiben:

" $P(5)$ ist wahr". ◇

Dieser Formalismus (d.h., Prädikat P definieren und dann z.B. $P(5)$ schreiben) ist in der Aussagenlogik unüblich. Er leitet aber zur Prädikatenlogik über, wo er laufend benutzt wird, denn in der Prädikatenlogik wollen wir jetzt nicht mehr Eigenschaften einzelner Objekte betrachten, sondern die Eigenschaften einer ganzen Klasse von Objekten, symbolisiert durch eine Variable wie z.B. x , und machen darüber wieder Aussagen. D.h.:

Grundidee der Prädikatenlogik:

Wir betrachten beliebige Objekte x einer zuvor exakt spezifizierten ganzen Klasse von Objekten und untersuchen Prädikate (d.h., Eigenschaften) $P(x)$ dieser Klasse von Objekten und logische Aussagen über solche Prädikate, d.h., über solche Eigenschaften.

§1-52 Beispiel.

Wir wollen also festlegen, wie wir formal und exakt mit einer Aussage z.B. von folgender Art umgehen:

"Für jede natürliche Zahl gilt, dass sie entweder gerade ist (d.h., ganzzahlig durch zwei teilbar) oder ungerade ist (d.h., nicht ganzzahlig durch zwei teilbar)". ◇

Diese Idee wollen wir nun umsetzen und ihre Anwendung näher studieren.

1.5.2 Prädikate und Quantoren

§1-53 Definition. ((Logisches) Prädikat; Objektvariable)

Ein (logisches) Prädikat P (lat. *praedicare* = “zusprechen”) ist eine einem Objekt zugesprochene mögliche Eigenschaft. Bezeichnen wir mit x ein beliebiges Objekt einer ganzen Klasse von Objekten, dann nennen wir x Objektvariable.

Haben wir ein Prädikat P (also eine Eigenschaft) spezifiziert, dann schreiben wir $P(x)$ und meinen damit, dass “Objekt x die Eigenschaft P hat” (was natürlich aussagenlogisch wahr oder falsch sein kann abhängig vom konkreten x (!)). ◇

Da x in einem Prädikat noch nicht ein konkretes Objekt darstellt sondern eine Variable ist, wird $P(x)$ erst für ein konkretes Objekt zu einer Aussage (!).

Beachten Sie dies: Einem Prädikat $P(x)$ kann man keinen Wahrheitswert zuordnen. Es ist also keine logische Aussage! Deshalb:

§1-54 Definition. (Prädikat als Aussageform)

Ein Prädikat $P(x)$ für beliebiges x nennen wir auch eine Aussageform.

◇

Denken Sie auch daran, dass wir in der Wahl der Notation natürlich frei sind. Deshalb schreiben wir unter Umständen auch $A(x)$, $A(n)$, $B(x)$, ... etc. für Prädikate / Aussageformen.

§1-55 Beispiele. (Prädikate)

- (1) $P(x) := "x \text{ ist ein typisches Schweizer Haustier.}"$

Das Prädikat ist z.B. für $x = \text{“Katze”}$ und $x = \text{“Hund”}$ wahr, für $x = \text{“Säbelzahntiger”}$ jedoch falsch.

- (2) $Q(x) := "x \text{ ist eine Primzahl.}"$

$Q(7)$ ist z.B. wahr und $Q(8)$ falsch.

- (3) $S(x) := \text{“Die ganze Zahl } x \text{ erfüllt die Gleichung } 2x + 1 = 9. \text{”}$

Das Prädikat ist für $x = 4$ wahr und für alle anderen ganzen Zahlen falsch.

◇

Die Idee eines Prädikats (einer Aussageform) kann man leicht auf mehrere Objektvariablen ausdehnen:

§1-56 Definition. (n -stellige Prädikate, Aussage ist 0-stelliges Prädikat)

Enthält ein Prädikat mehrere Objektvariablen x_1, \dots, x_n , $n > 1$, (je aus vorher spezifizierten Klassen M_1, \dots, M_n von Objekten), so dass bei einer Belegung all dieser Variable das Prädikat in eine Aussage übergeht, so nennen wir das Prädikat $P(x_1, \dots, x_n)$ ein n -stelliges Prädikat oder eine n -stellige Aussageform.

Da eine Aussage keine Variable enthält, sprechen wir bei ihr auch von einem 0-stelligen Prädikat. ◇

§1-57 Beispiele.

- (1) $P(x, y) := "x \text{ und } y \text{ sind reelle Zahlen für die } x \leq y \text{ gilt."$
 $P(6, 7)$ ist wahr und $P(12, 4)$ ist falsch.
- (2) $Q(x, y, z) := "x = y \text{ oder } y < z."$
Das Prädikat $Q(3, 5, 6)$ ist z.B. wahr, $Q(1, 5, 5)$ jedoch nicht.
- (3) $S(x, y) := "x \text{ ist Schweizer Ortschaft, dessen Postleitzahl } y \text{ als erste Ziffer hat."$
Das Prädikat $S(Basel, 1)$ ist falsch, jedoch $S(Windisch, 5)$ ist wahr.
- (4) $P(5) := "5 \text{ ist Primzahl.}$ ist eine Aussage, also ein 0-stelliges Prädikat. Und die Aussage ist wahr.

◊

§1-58 Definition. (Belegung eines Prädikats)

Das Ersetzen der Objektvariablen eines Prädikats $P(x_1, \dots, x_n)$ durch konkrete Objekte der Grundgesamtheit, nennen wir wie schon in der Aussagenlogik ebenfalls eine Belegung der (Objekt-) Variablen. ◊

Wie gesehen, wird durch eine Belegung der Variablen ein Prädikat zu einer Aussage und hat damit einen Wahrheitswert zugeordnet. Dies wird Auswertung genannt:

§1-59 Definition. (Auswertung eines Prädikats)

Die Ermittlung des Wahrheitswertes durch Variablenbelegung nennen wir auch eine Auswertung des Prädikats. ◊

Konvention: Im folgenden betrachten wir oft zur Vereinfachung einstellige Prädikate $P(x), Q(x), \dots$ obwohl Analoges auch für mehrstellige Prädikate gilt.

§1-60 Definition. (Äquivalenz (semantische Gleichheit) zweier Prädikate)

Ein Prädikat $P(x)$ ist äquivalent (semantisch äquivalent) zu einem Prädikat $Q(x)$, formal

$$P(x) \iff Q(x),$$

genau dann, wenn beide Prädikate für alle Objektvariablenbelegungen dieselben Wahrheitswerte liefern. ◊

Bitte beachten Sie, dass wir die semantische Äquivalenz von Aussageformeln dagegen mit \equiv symbolisiert haben. Es gibt Autoren, die dieses Symbol auch hier benutzen. Wir wollen sie der Klarheit wegen aber auseinanderhalten.

§1-61 Theorem. (Kriterium für Äquivalenz zweier Prädikate)

Zwei Prädikate sind äquivalent, genau dann wenn $P(x) \implies Q(x)$ und $Q(x) \implies P(x)$ für jede Objektvariablenbelegung gilt. \diamond

§1-62 Beispiele. (Äquivalente Prädikate)

- (1) Seien $P(x) := "x \text{ ist durch } 12 \text{ teilbar.}"$ und $Q(x) := "x \text{ ist durch } 3 \text{ und durch } 4 \text{ teilbar.}"$
Dann ist $P(x) \iff Q(x)$.
- (2) Seien $P(x) := "x \text{ ist Primzahl.}"$ und $Q(x) := "x \text{ ist ungerade natürliche Zahl.}"$
Dann sind $P(x)$ und $Q(x)$ nicht äquivalent, denn z.B. $x = 2$ ist gerade und Primzahl.
Ebenso gibt es ungerade natürliche Zahlen, die nicht Primzahl sind.

\diamond

Um logische Aussagen über Prädikate selbst machen zu können, sog. prädikatenlogische Aussagen, benötigen wir drei weitere Operatoren und Symbole, die Quantoren genannt werden:

§1-63 Definition. (Quantoren)

Sei P ein beliebiges Prädikat und x eine Objektvariable einer vorher exakt definierten Objektklasse.

- (1) Die logische Aussage "für alle x gilt $P(x)$ " schreiben wir formal

$$\forall x : P(x).$$

Gesprochen: "Für alle x gilt $P(x)$ ".

Das Symbol \forall wird **Allquantor** oder **Generalisator** genannt.

Später nach Mengenlehre werden wir folgendermassen auch die "Menge" aus der die x stammen, so ebenfalls hinschreiben:

$$\forall x \in M : P(x).$$

Gesprochen: "Für alle x der Menge (aus der Menge) M gilt $P(x)$ ".

- (2) Die logische Aussage "es gibt (mind.) ein x , so dass $P(x)$ gilt" schreiben wir formal

$$\exists x : P(x).$$

Gesprochen: "Es gibt ein x , so dass $P(x)$ (gilt)" oder "Es gibt ein x mit $P(x)$ ".
Das Symbol \exists wird **Existenzquantor** oder **Partikularisator** genannt.

Mit Spezifizierung der "Menge" schreiben wir wieder: $\exists x \in M : P(x)$.

Gesprochen: "Es gibt ein x in der Menge M , so dass $P(x)$ (gilt)" oder "Es gibt ein x in der Menge M mit $P(x)$ ".

- (3) Die logische Aussage "es gibt genau ein x , so dass $P(x)$ gilt" schreiben wir formal

$$\exists !x : P(x).$$

Gesprochen: "Es gibt genau ein x , so dass $P(x)$ (gilt)" oder "Es gibt genau ein x mit $P(x)$ ".

Das Symbol $\exists!$ wird Eindeutigkeitsquantor oder Einzigkeitsquantor genannt.

Und mit "Mengen"angabe: $\exists!x \in M : P(x)$.

Gesprochen wird dies dann: "Es gibt genau ein x in der Menge M , so dass $P(x)$ (gilt)" oder "Es gibt genau ein x in der Menge M mit $P(x)$ ".

◊

Beachten Sie auch: Die Objektvariable x heisst im Zusammenhang mit einem Prädikat $P(x)$ auch freie (Objekt-) Variable. Dagegen nennt man x im Zusammenhang mit einer prädikatenlogischen Formel gebundene (Objekt-) Variable, weil sie vorgängig durch die Operatoren \forall , \exists , $\exists!$ "gebunden" ist (d.h., Bestandteil dieser Operatoren sind).

Es sind auch Bindungsprioritäten bzgl. der Quantoren und der anderen Operatoren zu beachten:

§1-64 Definition. (Bindungsprioritäten von Quantoren)

Ein Quantor bindet stärker als alle anderen logischen Operatoren.

◊

Ein Quantor bindet im Prinzip immer nur eine Objektvariable. Man müsste also richtigerweise $\forall x \forall y : P(x, y)$ bzw. sogar $\forall x : \forall y : P(x, y)$ und nicht $\forall x, y : P(x, y)$ schreiben. Wir definieren deshalb zur Vereinfachung folgende Notation:

§1-65 Definition. (Schreibweisen bei mehreren Quantoren)

Abkürzung:	für:
$\forall x_1, \dots, x_n : P(x_1, \dots, x_n)$	$\forall x_1 : (\forall x_2 : (\dots (\forall x_n : P(x_1, \dots, x_n)))),$
$\exists x_1, \dots, x_n : P(x_1, \dots, x_n)$	$\exists x_1 : (\exists x_2 : (\dots (\exists x_n : P(x_1, \dots, x_n)))),$
$\forall x \exists y : P(x, y)$	$\forall x : (\exists y : P(x, y)),$
$\exists x \forall y : P(x, y)$	$\exists x : (\forall y : P(x, y)).$

(Die Fälle der letzten zwei Zeilen werden bei mehr als zwei Quantoren analog verallgemeinert.) ◊

§1-66 Beispiele. (Prädikatenlogische Aussagen (Quantorenaussagen))

- (1) Bezeichne x ein Objekt der Gesamtheit aller Katzen. Dann ist

$$\forall x : "x \text{ hat Schnurrbarthaare.}"$$

eine (wahre) prädikatenlogische Aussage.

- (2) Für Schwäne x ist

$$\forall x : "x \text{ hat eine weiss-gräuliche Farbe.}"$$

eine (falsche) prädikatenlogische Aussage. (Es gibt schwarze Schwäne.)

- (3) Bezeichne a die Personen in unserem Unterrichts-Raum. Dann ist $\exists!a : "a \text{ ist Dozent.}"$ eine (im Normalfall wahre) prädikatenlogische Aussage.
- (4) $\forall x, y \text{ reelle Zahl} : xy = x + y$
ist eine (zweistellige) prädikatenlogische Aussage (und ist falsch).

◊

Nur zur Vollständigkeit wollen wir noch festhalten, was wir genau mit einer prädikatenlogischen Formel meinen, auch wenn dies wahrscheinlich nun klar ist. Denn prädikatenlogische Aussagen sind, wie gesehen, insbesondere Aussagen und wir brauchen deshalb keine spezielle Definition für den Begriff "prädikatenlogische Formel":

§1-67 Definition. (Prädikatenlogische Formel)

Eine prädikatenlogische Formel ist einfach eine aussagenlogische Formel (im früher definierten Sinn), in welcher mind. noch ein Quantor vorkommt. ◊

§1-68 Beispiele. (Prädikatenlogische Formel)

- (1) Bezeichne x ein Tier und a eine Automarke.

$$(\forall x : "x \text{ hat einen Stoßwechsel}") \wedge (\exists a : "a \text{ ist deutsche Automarke.}")$$

ist eine prädikatenlogische Formel (und ist wahr).

- (2) Bezeichne p eine natürliche, gerade Zahl mit $p > 2$ und seien x, y reelle Zahlen, wobei x nicht Null sei.

$$(\exists p : "p \text{ ist Primzahl}") \vee (\forall x \exists y : xy = 1)$$

ist eine prädikatenlogische Formel (und ist ebenfalls wahr).

- (3) Bezeichne $n, m \in \mathbb{N} \cup \{0\}$ beliebige natürliche Zahlen.

$$\forall n, m : (n \text{ gerade} \implies nm \text{ gerade})$$

ist eine prädikatenlogische Formel (und ist wahr).

- (4) Bezeichne x, y, k natürliche Zahlen ungleich 0.

$$\forall x, y \exists !k : ((y = kx) \vee (x = ky))$$

ist eine prädikatenlogische Formel (und falsch; Gegenbeispiel: $x = 2$ und $x = 3$ und viele andere mehr).

◊

1.5.3 Rechenregeln für Quantoren (für prädikatenlogische Formeln); Umformungen

Es gibt auch für den Umgang mit prädikatenlogischen Formeln ein paar Rechenregeln, welche wir anwenden können sollten:

§1-69 Theorem. (Elementare Rechenregeln für prädikatenlogische Formeln)

Seien x und y Variablen für Objekte aus zuvor spezifizierten Grundgesamtheiten ("Mengen").

Sätze:	Regel/Gesetz:
Verneinung	$\neg(\forall x : A(x)) \iff \exists x : \neg A(x)$ $\neg(\exists x : A(x)) \iff \forall x : \neg A(x)$
Vertauschbarkeit	$\forall x, y : A(x, y) \iff \forall y, x : A(x, y)$ $\exists x, y : A(x, y) \iff \exists y, x : A(x, y)$ $\exists x \forall y : A(x, y) \xrightarrow{(*)} \forall y \exists x : A(x, y)$
Verteilung von \vee, \wedge	$\exists x : (P(x) \vee Q(x)) \iff (\exists x : P(x)) \vee (\exists x : Q(x))$ $\exists x : (P(x) \wedge Q(x)) \xrightarrow{*} (\exists x : P(x)) \wedge (\exists x : Q(x))$ $\forall x : (P(x) \wedge Q(x)) \iff (\forall x : P(x)) \wedge (\forall x : Q(x))$ $\forall x : (P(x) \vee Q(x)) \xleftarrow{[E.3]} (\forall x : P(x)) \vee (\forall x : Q(x))$
	$\forall x : A(x) \xrightarrow{*} \exists x : A(x)$ $\exists !x : A(x) \xrightarrow{*} \exists x : A(x)$

◊

§1-70 Beispiele. (Es gilt nur die Implikation (*) in §1-69)

- (1) Dass in (*) die Umkehrung nicht gilt, sehen wir zum Beispiel an der Aussageform $A(x, y) := "yx = 2"$ für reelle Zahlen y ohne die Null und reelle Zahlen x .

Für jede reelle Zahl y ungleich Null gibt es eine reelle Zahl x , so dass $xy = 2$ ist. Nämlich $x = \frac{2}{y}$. Dies ist Richtung " \implies ", welche damit richtig ist.

Hingegen hängt dieses x natürlich vom jeweiligen y ab. Es gibt also sicher kein spezielles x , welches mit allen (jedem beliebigen) y multipliziert gerade 2 ergibt. Also ist die Richtung " \iff " nicht richtig.

- (2) Und noch ein weiteres Beispiel dafür:

Sei $A(x, y) := "Die Krankheit x hat Apfel y befallen".$

Dann folgt aus der Aussage, dass es eine Krankheit gibt, welche alle Äpfel befallen hat, auch dass jeder Apfel von einer Krankheit befallen ist. Dies entspricht der Richtung " \implies ".

Hingegen, wenn jeder Apfel von einer Krankheit befallen ist, heisst dies noch lange nicht, dass es eine (einzelne, bestimmte) Krankheit gibt, welche alle Äpfel befallen hat. Also ist die Richtung " \iff " nicht richtig.

◊

§1-71 Beispiel. (Verteilung von \exists und \vee)

Sei H die Menge aller Patientinnen in einem Spital, welche an einer medizinischen Studie teilnehmen. Wir betrachten die beiden Prädikate

$M_1(x) = "Medikament 1 wirkt signifikant bei Patientin $x".$$

$M_2(x) = "Medikament 2 wirkt signifikant bei Patientin $x".$$

Nun betrachten wir die folgenden beiden Aussagen:

$$A = \left(\exists x \in H : (M_1(x) \vee M_2(x)) \right),$$

$$B = \left((\exists x \in H : M_1(x)) \vee (\exists x \in H : M_2(x)) \right).$$

D.h., in Worten: (Beachten Sie, dass es sich hier um das "inklusive oder" handelt!)

A = "Es gibt eine Patientin x , bei welcher Medikament 1 oder Medikament 2 signifikant wirkt."

B = "Es gibt eine Patientin x , bei welcher Medikament 1 signifikant wirkt oder es gibt eine Patientin x , bei welcher Medikament 2 signifikant wirkt."

Wir überlegen uns:

\implies : Wenn Aussage A gilt, dann heisst dies, dass es genau drei Fälle geben kann: (1) es gibt mind. eine Patientin, bei welcher M_1 wirkt, (2) es gibt mind. eine Patientin, bei welcher M_2 wirkt, oder (3) es gibt mind. eine Patientin, bei welcher M_1 und M_2 wirkt. In allen drei Fällen ist auch die Aussage B erfüllt. D.h., es gilt $A \implies B$.

\impliedby : Eine analoge Überlegung zeigt uns, dass B ebenfalls heisst, dass genau einer der drei Fälle von vorhin gilt. Damit ist auch die Implikation $A \impliedby B$ gezeigt.

Die beiden Aussagen sind also äquivalent. ◊

§1-72 Beispiel. (Verteilung von \exists und \wedge)

Sei H wiederum die Menge aller Patientinnen in einem Spital, welche an einer medizinischen Studie teilnehmen. Wir betrachten wieder die beiden Prädikate

$$M_1(x) = \text{"Medikament 1 wirkt signifikant bei Patientin } x\text{"}$$

$$M_2(x) = \text{"Medikament 2 wirkt signifikant bei Patientin } x\text{"}$$

Nun studieren wir die folgenden beiden Aussagen

$$C = \left(\exists x \in H : (M_1(x) \wedge M_2(x)) \right)$$

$$D = \left((\exists x \in H : M_1(x)) \wedge (\exists x \in H : M_2(x)) \right)$$

und überlegen uns, ob C äquivalent zu D ist. In Worten haben wir nun:

C = "Es gibt eine Patientin x , bei welcher Medikament 1 und Medikament 2 signifikant wirkt."

D = "Es gibt eine Patientin x , bei welcher Medikament 1 signifikant wirkt und es gibt eine Patientin x , bei welcher Medikament 2 signifikant wirkt."

\implies : Wenn Aussage C gilt, dann heisst dies, dass es mind. eine Patientin gibt, bei welcher beide Medikamente wirken. Aussagen über andere Patientinnen spielen keine weitere Rolle. Damit ist aber auch gerade Aussage D erfüllt, denn es sind beide Teilaussagen darin erfüllt, nämlich durch dieselbe Patientin x . Also gilt $C \implies D$.

\impliedby : Wenn die Aussage D erfüllt ist, dann heisst das nur einmal, dass es eine Patientin x_1 gibt, bei welcher M_1 wirkt und dass es eine (möglicherweise andere!) Patientin x_2 gibt, bei welcher M_2 wirkt. Damit folgt also *nicht* notwendigerweise, dass es eine einzige Patientin gibt, bei welcher beide Medikamente wirken! Die Umkehrung gilt also *nicht*!

Die beiden Aussagen sind also *nicht* äquivalent, sondern es gilt nur $C \implies D$. ◊

§1-73 Beispiel. (Verteilung von \vee und \wedge)

Wir betrachten wie anhin die Menge H aller Patientinnen in einem Spital, welche an einer medizinischen Studie teilnehmen und die beiden Prädikate

$$M_1(x) = \text{"Medikament 1 wirkt signifikant bei Patientin } x\text{."}$$

$$M_2(x) = \text{"Medikament 2 wirkt signifikant bei Patientin } x\text{."}$$

Unsere Aussagen sind nun (beachten Sie, dass wir nun zuerst den Fall \wedge behandeln!):

$$E = (\forall x \in H : (M_1(x) \wedge M_2(x))),$$

$$F = ((\forall x \in H : M_1(x)) \wedge (\forall x \in H : M_2(x))).$$

In Worten:

$$E = \text{"Medikament 1 und Medikament 2 wirken bei allen Patientinnen signifikant."}$$

$$F = \text{"Medikament 1 wirkt bei allen Patientinnen signifikant und Medikament 2
wirkt bei allen Patientinnen signifikant."}$$

\implies : Wenn Aussage E gilt, dann heisst dies, dass beide Medikamente bei allen Patientinnen signifikant wirken. Dies ist natürlich nichts anderes als dass das erste Medikament bei allen Patientinnen wirkt und dass das zweite Medikament ebenfalls bei allen Patientinnen wirkt. Also gilt $E \implies F$.

\impliedby : Mit analoger Überlegung ist auch die Umkehrung ziemlich offensichtlich. D.h., auch $E \impliedby F$ gilt.

Die beiden Aussagen sind also äquivalent. ◊

§1-74 Beispiel. (Verteilung von \vee und \wedge)

Sei H erneut die Menge aller Patientinnen in einem Spital, welche an einer medizinischen Studie teilnehmen und seien die Prädikate wieder

$$M_1(x) = \text{"Medikament 1 wirkt signifikant bei Patientin } x\text{."}$$

$$M_2(x) = \text{"Medikament 2 wirkt signifikant bei Patientin } x\text{."}$$

Die noch zu studierenden Aussagen sind:

$$G = (\forall x \in H : (M_1(x) \vee M_2(x)))$$

$$I = ((\forall x \in H : M_1(x)) \vee (\forall x \in H : M_2(x)))$$

In Worten:

$$G = \text{"Bei allen Patientinnen wirkt Medikament 1 oder Medikament 2 signifikant."}$$

$$I = \text{"Bei allen Patientinnen wirkt Medikament 1 signifikant oder bei allen
Patientinnen wirkt Medikament 2 signifikant."}$$

\implies : Wir betrachten den einfachen Fall, dass $H := \{x_1, x_2, x_3\}$ ist und dass folgendes gilt: bei Patientin x_1 wirkt nur M_1 , bei Patientin x_2 wirkt nur M_2 und bei Patientin x_3 wirkt sowohl M_1 als auch M_2 . Dann ist offensichtlich G erfüllt. Diese Situation ist aber in I ausgeschlossen, denn hier haben wir nur einen der folgenden drei Fälle: (1) für alle Patientinnen gilt, dass M_1 wirkt, (2) für alle Patientinnen gilt, dass M_2 wirkt, (3) für alle Patientinnen gilt, dass sowohl M_1 als auch M_2 wirkt. Also folgt aus G nicht notwendigerweise I !

\impliedby : In jedem der drei Fälle (1), (2) und (3) folgt, dass insbesondere G gilt, weil ja bei jeder Patientin (einzelne) mind. ein Medikament wirkt. Damit gilt die Umkehrung.

Die beiden Aussagen sind also nicht äquivalent, sondern es gilt nur $G \impliedby I$. ◊

§1-75 Beispiel. (Anwendung der Rechenregeln für prädikatenlogische Formeln mit Quantoren)

Aufgabe:

Formalisieren Sie folgende Aussage und formen Sie sie mit den Rechenregeln so um, dass vor keinem Quantor mehr eine Negation steht (wir nennen dies auch "Klammer auflösen in einer prädikatenlogischen Formel"):

"Es gilt nicht, dass es eine Farbe z gibt, so dass alle Menschen x ein Auto y besitzen mit der Farbe z ."

Lösung:

Bezeichne x also Personen, y bezeichne Autos und z bezeichne Farben.

Wir formalisieren zuerst obige Aussage als prädikatenlogische Formel:

$$\neg(\exists z \forall x \exists y : A(x, y, z))$$

wobei $A(x, y, z) :=$ "Person x hat Auto y der Farbe z ."

Dann können wir die Klammer mit den Rechenregeln folgendermassen auflösen:

$$\begin{aligned} & \neg(\exists z \forall x \exists y : A(x, y, z)) \\ \iff & \forall z : \neg(\forall x \exists y : A(x, y, z)) && (2.\text{Verneinungssatz}) \\ \iff & \forall z \exists x : \neg(\exists y : A(x, y, z)) && (1.\text{Verneinungssatz}) \\ \iff & \forall z \exists x \forall y : \neg A(x, y, z) && (2.\text{Verneinungssatz}). \end{aligned}$$

Das Ergebnis lässt sich zum Beispiel so umgangssprachlich formulieren:

"Für jede Farbe z gilt, dass es eine Person x gibt, so dass alle ihre Autos nicht Farbe z haben."

◇

Kapitel 2

Mengenlehre

Kapitelinhalt

2.1	Motivierung – Wieso Mengenlehre?	41
2.2	Was ist eine Menge? Begriffe, Beispiele	41
2.3	Mengenoperationen	48
2.4	Gesetze der Mengenalgebra und ihre Anwendung	52
2.5	Weiteres zum kartesischen Produkt von Mengen	54
2.6	Die Partition einer Menge	55
2.7	Die Unendlichkeit von Mengen	59
2.8	Übersicht: Gleichheits- und Äquivalenznotationen	62

L E R N Z I E L E :

- LZ 2.1** Sie wissen, was eine Menge ist und können mit den Grundkonzepten der Mengenlehre umgehen.
- LZ 2.2** Sie verstehen die mengentheoretischen Operationen und können mit ihnen rechnen (Mengenalgebra), um insbesondere Mengenausdrücke zu vereinfachen oder zu entscheiden, ob zwei gegebene Mengenausdrücke dieselbe Menge beschreiben.
- LZ 2.3** Sie verstehen das Konzept der Partition einer Menge.
- LZ 2.4** Sie verstehen mindestens, dass die Unendlichkeit von Mengen zu unintuitiven und unter Umständen problematischen Ergebnissen führt.
- LZ 2.5** Sie können einfachere Anwendungsprobleme mengentheoretisch beschreiben und lösen.
-

2.1 Motivierung – Wieso Mengenlehre?

Jede Sprache (auch die Logik!) hat ein “Universum” von Objekten, Gegenständen, Ideen, usw. worüber gesprochen wird. Selbstverständlich ist die Aussage

“Alle Schplongs sind hülorfk. Oingpoing ist ein Schplong, also ist Oingpoing hülorfk.”

(logisch) korrekt formuliert (und zudem wahr). Aber wenn wir Objekte "Schplongs", "Oingpoing" und die Eigenschaft "hülorfk" selber nicht auch wiederum exakt fassen und beschreiben können, nützt auch die ganze Logik nichts!

§2-1 Beispiel.

Spricht zum Beispiel jemand auch noch von den Schplings, dann möchte ich wissen, ob dies nur ein anderer Name für die Schplongs ist. Und wenn nicht, ob sie trotzdem auch hülorfk sind. ◇

Das Ziel ist also folgendes: Wir müssen auch mit den (unterscheidbaren) Objekten und ihren Eigenschaften präzise und unmissverständlich umgehen können. D.h., nicht nur mit der Sprache, mit welcher wir über sie sprechen, sondern auch mit den Objekten und ihren Eigenschaften selber. Dies führt auf folgende erste Definition:

§2-2 Definition. (Diskursuniversum)

Die Gegenstände, Objekte, Ideen, usw. worüber wir zum Beispiel mittels Logik sprechen wollen, nennen wir auch das Diskursuniversum. ◇

Eine (unter gewissen Umständen) korrekte und präzise Formalisierung und Handhabung von Diskursuniversen bietet nun gerade die Mengenlehre und der dort definierte Begriff einer “Menge”.

2.2 Was ist eine Menge? Grundlegende Begriffe und Beispiele

Zur Formalisierung von Diskursuniversen definieren wir in der Mengenlehre^[E.4]:

§2-3 Definition. (Menge)

Eine Menge ist eine Zusammenfassung von bestimmten, wohlunterschiedenen Objekten unserer Anschauung oder unseres Denkens zu einem Ganzen. Die Objekte heißen Elemente der Menge.

Symbolisierung:

$x \in M$: “ x ist Element der Menge M ” oder kurz “ x in M ”.

$x \notin M$: “ x ist nicht Element von M ” oder kurz “ x nicht in M ”.

\emptyset oder $\{\}$: Menge, welche keine Elemente enthält, d.h., die leere Menge.

◇

Mengen sind also wieder selbst mathematische Objekte, über die wir Aussagen machen und die wir als "Tool" benutzen wollen. Ein paar Dinge hier noch hervorgehoben: Die Formulierung "bestimmte, wohlunterschiedene Objekte" ist wichtig, da wir die einzelnen Objekte einer Menge unterscheiden können müssen. Sie müssen als solche (vor)bestimmt sein und eine "Identität" haben. Die Formulierung "Objekte unserer Anschauung oder unseres Denkens" meint, dass die Elemente einer Menge sowohl physisch existent, visuell wahrnehmbar als auch nur abstrakt existent sein können. Letzteres bedeutet, dass wir schon eine Anzahl von Ideen zu einer Menge zusammenfassen können.

Eine Art eine konkrete Menge genau zu definieren (wir werden gleich noch andere kennenlernen) ist, alle ihre Elemente in geschweiften Klammern aufzuzählen.

§2-4 Beispiel.

$M := \{a, b, c\}$. Das ist die Menge M , welche genau aus den drei Elementen a, b, c besteht. ◇

Unbedingt zu beachten ist, dass dasselbe Objekt nicht mehrmals in der Aufzählung einer Menge vorkommen kann:

§2-5 Beispiel.

$\{a, a, b, c\}$ ist keine Menge! ◇

Weiter ist die Reihenfolge mit welcher die Elemente aufgeführt werden ohne Belang. Sie spielt keine Rolle.

§2-6 Beispiel.

Die Menge $\{a, b, c\}$ unterscheidet sich nicht von $\{c, b, a\}$, usw.. ◇

Ebenso ist zentral, dass ein Objekt entweder Element einer Menge ist oder nicht. Wir betrachten also nur sogenannte "gutartige Mengen", auch Klassen genannt. (Zum Hintergrund dafür: vgl. z.B. "Russellsche Antinomie".)

§2-7 Beispiel.

Für die Menge $M := \{a, b, c\}$ gilt $a \in M$ und $d \notin M$. ◇

Schliesslich zur Schreibweise noch eine kleine Anmerkung: Für Mengen werden wir oft die Grossbuchstaben A, B, C, \dots benutzen, wie auch bei den logischen Aussagen. Das soll aber nicht zu Verwirrung führen, da immer klar sein sollte, was gemeint ist. (Dies muss sowieso immer zuerst festgelegt werden.) Für die Elemente einer Menge werden wir oft Kleinbuchstaben a, b, c, \dots, x, y, z benutzen.

Um mit dem Konzept einer Menge noch vertrauter zu werden, noch ein paar Beispiele mehr:

§2-8 Beispiele.

- (1) $A_1 := \{\text{Löwe, Tiger, Gepard, Puma, Jaguar}\}$ ist eine Menge von Raubkatzen. Offensichtlich gilt hier $\text{Jaguar} \in A_1$ und $\text{Zebra} \notin A_1$.
- (2) Die Menge A_2 der Studierenden dieser Klasse.
- (3) $A_3 := \{0, 1, 2, 3, 4, \dots\}$ ist die Menge der natürlichen Zahlen inklusive 0. Es gilt $32254 \in A_3$ und $\frac{1}{2} \notin A_3$. (Eine Menge kann also auch unendlich viele Elemente enthalten!)

- (4) Beachte: Auch $A_4 := \{\text{Tiger}, \pi, \text{BMW}, \text{Apfel}, \text{Klaus}, 2 + 3 = 5\}$ ist eine Menge. (Die Elemente müssen im Prinzip nicht vom gleichen "Typ" sein! \rightsquigarrow In Praxis gehören sie aber oft einer bestimmten "Grundmenge" (siehe folgend §2-26) an.)

◊

§2-9 Definition. (Gleichheit zweier Mengen)

Zwei Mengen A und B heißen gleich (oder identisch; symbolisch $A = B$), wenn sie genau dieselben Elemente enthalten.

Formal: $A = B : \iff \forall x : (x \in A \Leftrightarrow x \in B)$. Sind sie nicht gleich, so schreiben wir $A \neq B$.

◊

§2-10 Beispiel.

Für $A_1 := \{3, 4, 5, 6, 7, 8, 9, 10, 11\}$ und $A_2 := \{5, 11, 3, 7, 8, 4, 10, 9, 6\}$ gilt: $A_1 = A_2$, sie sind also gleich.

◊

§2-11 Beispiel.

Es gilt $\{\text{Löwe, Tiger, Gepard, Puma, Jaguar}\} \neq \{\text{Löwe, Tiger, Gepard, Puma}\}$.

◊

Was für Möglichkeiten haben wir, um konkrete Menge zu definieren bzw. darzustellen? Wir benutzen im wesentlichen drei Arten um eine konkrete Menge zu definieren:

§2-12 Definition. (Möglichkeiten, eine konkrete Menge zu definieren)

- (1) Aufzählung der Elemente (bereits erwähnt), aufzählende Form der Mengendefinition genannt:

Beispiel: $A := \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

- (2) Beschreibung der Eigenschaften der Elemente, beschreibende Form der Mengendefinition genannt:

Sei $P(x)$ ein Prädikat.

- In Worten: "Die Menge A aller x mit (der Eigenschaft) $P(x)$."
- Mathematisch formal: $A := \{x \mid P(x)\}$. (Das Symbol \mid wird gelesen als "mit", "mit der Eigenschaft" oder "wobei".)

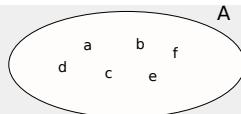
Beispiel: "Die Menge A aller natürlichen, ungeraden Zahlen kleiner 10."

Beispiel: $A := \{n \mid n \in \mathbb{N} \wedge n < 10 \wedge (\exists k \in \mathbb{Z} : n = 2k + 1)\}$

(wobei \mathbb{N} hier die Menge der natürlichen Zahlen inkl. 0 symbolisiert).

- (3) Venn-Diagramme zur grafischen Veranschaulichung: Die Elemente werden innerhalb einer geschlossenen Linie gezeichnet und ausserhalb der Linie, die Mengenvariable notiert.

Beispiel:



für eine Menge $A := \{a, b, c, d, e, f\}$.

◊

Einer der wichtigsten und grundlegendsten Typen von Mengen in der Mathematik sind selbstverständlich die Zahlenmengen^[E.5].

§2-13 Definition. (Wichtige Zahlenmengen in der Mathematik)

- (1) $\mathbb{N} := \{0, 1, 2, 3, 4, \dots\}$ ist die **Menge der natürlichen Zahlen**.
- (2) Für jede natürliche Zahl^[E.6] $n \in \mathbb{N} \setminus \{0\}$ ist die Menge $\mathbb{N}_n := \{0, 1, 2, \dots, n\}$ die **Menge der natürlichen Zahlen bis und mit n** .
- (3) $\mathbb{Z} := \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ ist die **Menge der ganzen Zahlen**.
- (4) $\mathbb{Q} := \left\{ \frac{m}{n} \mid m \in \mathbb{Z} \wedge n \in \mathbb{N} \wedge n \neq 0 \right\}$ ist die **Menge der rationalen Zahlen**. Bei einer rationale Zahl $q := \frac{m}{n}$ nennen wir m den **Zähler** und n den **Nenner**.

Beispiel: $\frac{1}{3}, -\frac{331}{4397}, \frac{4}{1} \equiv 4, -\frac{9}{3} \equiv -3, \frac{1}{4} \equiv 0.25$

Beispiel: *Nicht* rational sind z.B.: $\sqrt{2}, \pi$

- (5) $\mathbb{R} := \{x \mid x \text{ ist endlicher oder unendlicher Dezimalbruch}\}$ ist die **Menge der reellen Zahlen**.

Beispiel: $\mathbb{N}, \mathbb{Z}, \mathbb{Q}$ (d.h., \mathbb{Q} , da \mathbb{N}, \mathbb{Z} darin enthalten) zuzüglich der **irrationalen Zahlen**, wie:

$\sqrt{2} = 1.4142135\dots, \pi = 3.1415926\dots, e = 2.7182818\dots$

- (6) $\mathbb{C} := \{x + yi \mid x, y \in \mathbb{R} \wedge i := \sqrt{-1}\}$ (siehe spätere Vorlesungen) ist die **Menge der komplexen Zahlen**.

Beispiel: $1 - 2i, \frac{1}{2} - \sqrt{2}i, \cos(x) + i \sin(x) (= e^{xi})$, für reelle Zahl $x \in \mathbb{R}$, ...usw.

◊

Im Zusammenhang mit dem allgemeinen Begri einer Menge gibt es noch ein paar weitere fundamentale Begri e, welche wir hier einführen wollen.

§2-14 Definition. (Teilmengen, Obermengen)

Gilt für zwei Mengen A und B

$$x \in A \implies x \in B$$

so schreiben wir $A \subseteq B$ und sagen **A ist Teilmenge von B und B ist Obermenge von A** .

Ist zusätzlich $A \neq B$, dann schreiben wir $A \subset B$ oder zur Verdeutlichung gar $A \subsetneq B$ und sagen A ist echte Teilmenge von B und B ist echte Obermenge von A . \diamond

Beachten Sie, dass wir die Teilmengen-Relation \subseteq also über die Implikation definiert haben. Damit ist automatisch eine Menge Teilmenge von sich selbst, denn es gilt äquivalent

$$(A \subseteq B) \iff (A \subsetneq B \vee A = B).$$

Bei der Notation muss man ziemlich vorsichtig sein: Gewisse Autoren schreiben \subset für \subseteq und nur \subsetneq für echte Teilmengen.

§2-15 Beispiele.

- (1) $\{1, 2, 3\} \subset \{1, 2, 3, 4, 5\}$.
- (2) $\{x, y, z\} \subseteq \{x, y, z\}$.
- (3) Jedoch nicht $\{x, y, z\} \subset \{x, y, z\}$ (das ist falsch!)

\diamond

Für das Wort "Teilmenge" wird manchmal auch das Wort "Untermenge" benutzt. Beachten Sie auch folgende einfache Tatsache ("Korollar" [E.7]):

§2-16 Korollar.

Für jede Menge A gilt:

$$\emptyset \subseteq A$$

und

$$A \subseteq A.$$

Als Merkregel: "Die leere Menge ist Teilmenge jeder Menge. Und jede Menge ist Teilmenge von sich selbst." \diamond

Sehr häufig verwendete spezielle Teilmengen der reellen Zahlen sind die "Intervalle":

§2-17 Definition. (Intervalle in \mathbb{R})

Seien $a, b \in \mathbb{R}, a \leq b$, also beliebige reelle Zahlen, wobei a nicht grösser als b sei. Wir nennen

$$]a, b[:= \{x \in \mathbb{R} \mid a < x < b\}$$

ein offenes Intervall,

$$]a, b] := \{x \in \mathbb{R} \mid a < x \leq b\}$$

ein links-halbogenes Intervall,

$$[a, b[:= \{x \in \mathbb{R} \mid a \leq x < b\}$$

ein rechts-halbogenes Intervall und

$$[a, b] := \{x \in \mathbb{R} \mid a \leq x \leq b\}$$

ein abgeschlossenes Intervall.

Die a und b selber nennen wir dann die Intervallsgrenzen eines solchen Intervalls.



Zur Symbolisierung der linken und rechten o einen Intervallsgrenzen wird manchmal auch eine runde Klammer verwendet. D.h., für $]a, b[$ schreibt man auch (a, b) , für $]a, b]$ entsprechend $(a, b]$ und für $[a, b[$ damit auch $[a, b)$. Diese Notationsart ist meist unproblematisch, einzig (a, b) ist manchmal etwas verwirrend, da so oft auch ein geordnetes Zahlenpaar der Zahlen a und b (siehe später) bezeichnet wird. Beachten Sie weiter auch, dass wir die ganze Menge der reellen Zahlen auch als Intervall

$$]-\infty, \infty[= \mathbb{R}$$

und eine leere Menge mit jedem $a \in \mathbb{R}$ als

$$]a, a[= \emptyset$$

dargestellen können. Eine einzelne Zahl x lässt sich durch $[x, x]$ ebenfalls als (geschlossenes) Intervall darstellen. In vielen Gebieten der Ingenieursmathematik werden reelle Intervalle laufend und stillschweigend benutzt. Deshalb noch ein paar konkrete Beispiele.

§2-18 Beispiele. (Verschiedene Intervalle in \mathbb{R})

- (1) $]1, 3[:$ Die Menge aller reeller Zahlen zwischen 1 und 3, ohne die 1 und ohne die 3.
- (2) $[-\frac{2}{3}, \frac{4}{3}[$: Die Menge aller reeller Zahlen zwischen $-\frac{2}{3}$ und $\frac{4}{3}$, mit $-\frac{2}{3}$ und ohne $\frac{4}{3}$.
- (3) $[-\sqrt{2}, \sqrt{2}]:$ Die Menge aller reeller Zahlen zwischen $-\sqrt{2}$ und $\sqrt{2}$] inkl. dieser Intervallgrenzen $-\sqrt{2}$ und $\sqrt{2}$.
- (4) $]2, 2[:$ Die leere Menge \emptyset . Weil die Menge aller reeller Zahlen zwischen 2 und 2 ohne die 2 keine Elemente enthält.
- (5) $[0, \infty[:$ Die Menge aller nicht-negativer reellen Zahlen.
- (6) $[\frac{1}{2}, \frac{1}{2}]:$ Die Zahl $\frac{1}{2}$ als abgeschlossenes Intervall dargestellt.



§2-19 Definition. (Mächtigkeit (Kardinalität) von Mengen)

Die Anzahl Elemente einer Menge A wird auch ihre Mächtigkeit oder Kardinalität genannt und mit

$$|A|$$

oder (seltener) mit $\#A$ symbolisiert.

Für die Mächtigkeit einer *unendlichen* Menge – d.h., mit unendlich vielen Elementen – setzen wir

$$|A| := \infty.$$



§2-20 Beispiele.

(1) Sei $A := \{a, b, c, d, \dots, z\}$ die Menge der kleinen Buchstaben des deutschen Alphabets ohne die Umlaute. Dann ist $|A| = 26$.

Für $B := \{a, b, c, d\}$ haben wir **o** ensichtlich $B \subset A$ – wir können auch $B \subsetneq A$ schreiben – und die Mächtigkeit ist $|B| = 4$.

(2) Für die Zahlenmengen haben wir bekanntlich folgende Teilmengenbeziehungen:

$$\mathbb{N} \subsetneq \mathbb{Z} \subsetneq \mathbb{Q} \subsetneq \mathbb{R} \subsetneq \mathbb{C}.$$

Ebenso gilt für alle $n \in \mathbb{N} \setminus \{0\}$

$$\mathbb{N}_n \subsetneq \mathbb{N}.$$

◇

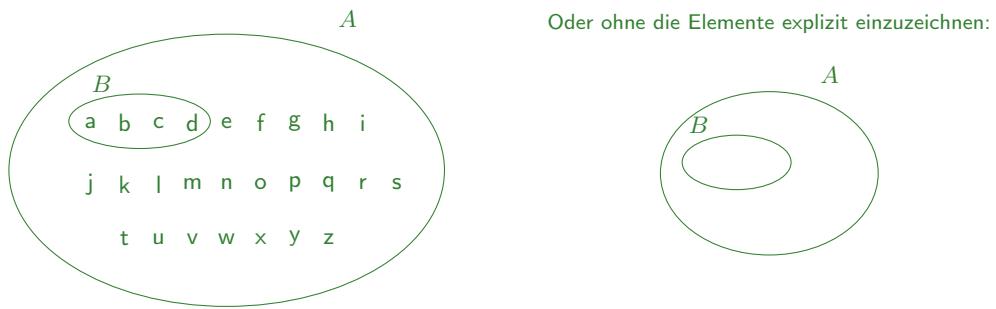
§2-21 Beispiel.

Aufgabe:

Zeichnen Sie das Venn-Diagramm des ersten Beispiels.

Lösung:

Sei also $A := \{a, b, c, d, \dots, z\}$ und $B := \{a, b, c, d\}$. Dann sieht das Venn-Diagramm für A und B z.B. so aus:



◇

§2-22 Beispiel.

Aufgabe:

Was können Sie über die Mächtigkeiten der Zahlenmengen

$$\mathbb{N}_n \subsetneq \mathbb{N} \subsetneq \mathbb{Z} \subsetneq \mathbb{Q} \subsetneq \mathbb{R} \subsetneq \mathbb{C}$$

sagen?

Lösung:

$$|\mathbb{N}_n| = n, \text{ wenn } 0 \notin \mathbb{N} \text{ und } |\mathbb{N}_n| = n + 1, \text{ wenn } 0 \in \mathbb{N},$$

$$\begin{aligned} |\mathbb{N}| &= \infty, \\ |\mathbb{Z}| &= \infty, \\ |\mathbb{Q}| &= \infty, \\ |\mathbb{R}| &= \infty, \\ |\mathbb{C}| &= \infty. \end{aligned}$$

(Wir werden auf die Unendlichkeit nochmals zurückkommen.) ◊

§2-23 Definition. (Potenzmenge einer Menge; (Mengen-) System einer Menge)

- (1) Die Menge aller Teilmengen einer Menge A wird **Potenzmenge von A** genannt und mit $\mathcal{P}(A)$ symbolisiert^[E.8].
- (2) Jede Teilmenge $S \subseteq \mathcal{P}(A)$ der Potenzmenge von A wird auch **(Mengen-) System über A** genannt.

◊

Beachten Sie: Sowohl die Potenzmenge als auch ein allgemeines Mengensystem^[E.9] sind also Mengen von Teilmengen, d.h., die Elemente der Potenzmenge und eines Mengensystems sind selbst wieder Mengen!

§2-24 Beispiele.

- (1) Sei $A := \{1, 2, 3\}$. Dann lautet die Potenzmenge von A :

$$\mathcal{P}(A) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, A\}.$$

- (2) $S := \{\emptyset, \{1\}, \{3\}, \{2, 3\}\}$ ist zum Beispiel ein weiteres Mengensystem über der Menge A des vorhergehenden Beispiels.

◊

§2-25 Theorem. (Mächtigkeit der Potenzmenge einer *endlichen* Menge)

Die Mächtigkeit der Potenzmenge einer endlichen Menge A ist

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

◊

2.3 Mengenoperationen

Wir führen nun die sechs grundlegenden Mengenoperationen ein: Vereinigung, Durchschnitt, symmetrischer Differenz, Differenz, Komplement, kartesisches Produkt. Dazu brauchen wir zuerst noch einen weiteren Begriff:

§2-26 Definition. (Grundmenge)

Werden bei einer Betrachtung verschiedene Mengen untersucht, welche alle Teil einer bestimmten, festen größeren Menge G sind, dann nennt man diese Menge G auch **Grundmenge**.

Sie wird dann im Venn-Diagramm als alles umfassendes Rechteck dargestellt



Oft wird sie aber auch stillschweigend weggelassen und wenn sie aus dem Kontext klar ist, nicht explizit erwähnt.

Beispiel: In der "Zahlentheorie" ist meist $G := \mathbb{N}$, in der "reellen Analysis" $G := \mathbb{R}$
und in der "komplexen Analysis" $G := \mathbb{C}$. \diamond

Beziehen wir uns bei der (beschreibenden) Mengendefinition auf eine feste Grundmenge G , so notieren wir diese oft vor dem "mit"-Strich | auf folgende Art:

$$A := \{x \in G \mid P(x)\}.$$

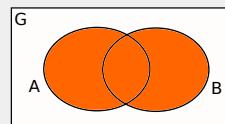
Dies ist also völlig äquivalent zu $A := \{x \mid x \in G \wedge P(x)\}$.

§2-27 Definition. (Grundlegende Mengenoperationen)

Seien A und B im folgenden Mengen einer bestimmten Grundmenge G .

(1) **Vereinigung \cup (engl. union):**

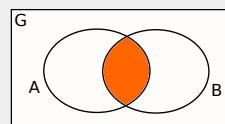
$$A \cup B := \{x \in G \mid x \in A \vee x \in B\}$$



Beispiel: $A := \{a, b, c\}, B := \{y, z\}, A \cup B = \{a, b, c, y, z\}$

(2) **Durchschnitt \cap (Schnitt, engl. intersection):**

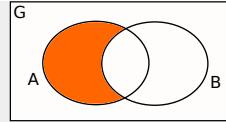
$$A \cap B := \{x \in G \mid x \in A \wedge x \in B\}$$



Beispiel: $A := \{a, b, c, d, e\}, B := \{c, d, x, y, z\}, A \cap B = \{c, d\}$

(3) **Differenz \setminus (Mengendifferenz, engl. difference):**

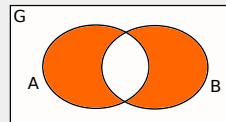
$$A \setminus B := \{x \in G \mid (x \in A \wedge x \notin B)\}$$



Beispiel: $A := \{a, b, c, d, x\}, B := \{c, d, x, y, z\}, A \setminus B = \{a, b\}$

(4) **Symmetrische Differenz Δ (engl. *symmetric difference*):**

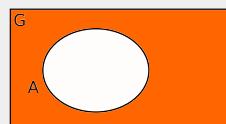
$$A \Delta B := \{x \in G \mid (x \in A \wedge x \notin B) \vee (x \in B \wedge x \notin A)\}$$



Beispiel: $A := \{a, b, c, d, x\}, B := \{c, d, x, y, z\}, A \Delta B = \{a, b, y, z\}$

(5) **Komplement (Mengenkomplement, engl. *complement*):**

$$A^c \equiv \overline{A} := \{x \in G \mid x \notin A\} = G \setminus A$$

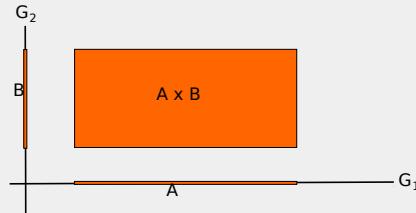


Beispiel: $G := \mathbb{N}, A := \{1, 2, 3, 4, 5\},$

$$\begin{aligned} A^c &= \overline{A} = \{6, 7, 8, 9, \dots\}, \text{ oder wenn } 0 \in \mathbb{N}, \\ A^c &= \overline{A} = \{0, 6, 7, 8, 9, \dots\} \end{aligned}$$

(6) **Kartesisches Produkt \times (Mengenprodukt, engl. *Cartesian product*):**

$$A \times B := \{(x, y) \mid x \in A \wedge y \in B\}$$



Beispiel: $A := \{1, 2\}, B := \{3, 4\},$

$$A \times B := \{(1, 3), (1, 4), (2, 3), (2, 4)\}.$$

$$B \begin{pmatrix} 4 \\ 3 \end{pmatrix} \begin{pmatrix} (1, 4)(2, 4) \\ (1, 3)(2, 3) \end{pmatrix} A \times B$$

$\circlearrowleft \begin{matrix} 1 & 2 \end{matrix} A$

◇

Die Mengenoperationen können natürlich auch wieder zu längeren Ausdrücken verknüpft werden (wir werden das noch üben!). Insbesondere gibt es auch bei der Mengenvereinigung, beim Mengendurchschnitt und beim kartesischen Produkt wieder praktische Verallgemeinerungen auf n Operanden:

§2-28 Definition. (Mengenoperationen auf n Operanden verallgemeinert)

Sei $n \in \mathbb{N}, n \geq 2$ und A_1, \dots, A_n beliebige Mengen. Dann definieren wir:

(1) n -fache Mengenvereinigung:

$$\bigcup_{k=1}^n A_k := ((\dots((A_1 \cup A_2) \cup A_3) \dots) \cup A_n).$$

(2) n -facher Mengendurchschnitt:

$$\bigcap_{k=1}^n A_k := ((\dots((A_1 \cap A_2) \cap A_3) \dots) \cap A_n).$$

(3) n -faches kartesisches Produkt:

$$\prod_{k=1}^n A_k := ((\dots((A_1 \times A_2) \times A_3) \dots) \times A_n).$$

Ein Element $(a_1, \dots, a_n) \in \prod_{k=1}^n A_k$ heisst (geordnetes) n -Tupel.

(4) Beim n -fachen kartesischen Produkt schreibt man insbesondere wenn alle Mengen $A := A_1 = \dots = A_n$ dieselben sind, abkürzend auch A^n . D.h., wir setzen noch:

$$A^n := \prod_{k=1}^n A.$$

◊

§2-29 Beispiele. (n -fache Mengenvereinigung und -durchschnitt)

Seien folgende Mengen gegeben:

$$\begin{aligned} A_1 &:= \{1, 2, 3, c\}, \\ A_2 &:= \{1, 3, a, b, c\}, \\ A_3 &:= \{1, 2, 3, 5, c\}, \\ A_4 &:= \{3, 4, a, c, d\}. \end{aligned}$$

Insbesondere ist also in §2-28 $n = 4$. Dann haben wir:

$$\begin{aligned} \bigcup_{k=1}^4 A_k &= A_1 \cup A_2 \cup A_3 \cup A_4 = \{1, 2, 3, 4, 5, a, b, c, d\}. \\ \bigcap_{k=1}^4 A_k &= A_1 \cap A_2 \cap A_3 \cap A_4 = \{3, c\}. \end{aligned}$$

◊

§2-30 Beispiele. (n -faches kartesisches Produkt)

(1) Seien folgende Mengen gegeben:

$$\begin{aligned} A_1 &:= \{1, 2\}, \\ A_2 &:= \{\alpha, \beta\}, \\ A_3 &:= \{a, b, c\}, \end{aligned}$$

Hier ist also insbesondere in §2-28 $n = 3$. Dann haben wir:

$$\begin{aligned} \prod_{k=1}^n A_k &= A_1 \times A_2 \times A_3 \\ &= \{(1, \alpha, a), (1, \alpha, b), (1, \alpha, c), (1, \beta, a), (1, \beta, b), (1, \beta, c), \\ &\quad (2, \alpha, a), (2, \alpha, b), (2, \alpha, c), (2, \beta, a), (2, \beta, b), (2, \beta, c)\} \end{aligned}$$

(2) Wir betrachten das abgeschlossene reelle Intervall $I := [0, 1]$. Das 3-fache kartesische Produkt dieser Menge ist dann

$$I^3 = I \times I \times I = [0, 1] \times [0, 1] \times [0, 1].$$

Dies ist die Menge aller 3-Tupel (Triple) von Zahlen aus dem Intervall $[0, 1]$, also

$$I^3 = \{(x, y, z) \mid x, y, z \in [0, 1]\},$$

was geometrisch einem Würfel der Länge 1 im dreidimensionalen Raum entspricht. Vielleicht wissen Sie schon, dass die x , y und z "Raumkoordinaten" darstellen. (Mehr dazu im Modul "Lineare Algebra und Geometrie (lag)".)

◊

2.4 Gesetze der Mengenalgebra und ihre Anwendung

Wenn wir den vorhergehenden Abschnitt betrachten, dann sehen wir, dass die einzige zusätzlich notwendige Relation bei den Mengen, die sogenannte Elementrelation \in ist, welche das "Element-sein einer Menge" bezeichnet. Alle anderen können durch Operatoren der Prädikatenlogik eingeführt werden! (Betrachten Sie dazu jeweils die rechten Seiten in den Definitionen von §2-27!)

Auch mit Mengen wollen wir "rechnen" können. Die Gründe für dieses "Rechnen mit Mengen" (Mengenalgebra) sind wieder dieselben wie bei der Logik: Wir wollen (nun) mengentheoretische Ausdrücke möglichst einfach auf Gleichheit testen, vereinfachen, usw. ... können. Weiter, müssen wir auch hier Bindungsprioritäten beachten. Wir setzen sie in folgender Definition gleich in Verbindung zu den Prioritäten der Logik:

§2-31 Definition. (Bindungsstärke (Prioritäten) von logischen und mengentheoretischen Operatoren)

Bindungsstärke:	Aussagen-/Prädikatenlogik:	Mengentheorie:
Höchste	Klammern $\forall, \exists, \exists!$	Klammern c (bzw. \neg)
↓	\neg	$\bigcap_{k=1}^n, \bigcup_{k=1}^n, \prod_{k=1}^n$
Niedrigste	$\in, \notin, \wedge, \vee, XOR, \subseteq, \subset, \subsetneq, =, \neq$	$\cap, \cup, \Delta, \setminus, \times$

◊

Eine ganz wichtige Warnung ist hier angebracht: Die Leserichtung von links nach rechts wird in der Mengenlehre nie als Prioritätsfestlegung angewandt! Das heisst zum Beispiel, $A \cap B \cup C$ darf nie "von links nach rechts" gelesen priorisiert werden und ohne Klammern ist dies undefiniert. Insbesondere ist damit nicht $(A \cap B) \cup C$ gemeint.

Sind aussagenlogische und mengentheoretische Teile in einem Ausdruck vorhanden, dann sind diese im Kontext voneinander abgrenzbar. Allgemein gilt aber: Besser einmal eine (überflüssige) Klammer zuviel verwenden. Dies dient manchmal auch der besseren Leserlichkeit.

Damit kommen wir nun zu den Rechenregeln der Mengenalgebra. Beachten Sie die Analogie zu den Gesetzen der Aussagenlogik, da die Mengenoperationen ja eben mit Hilfe von logischen Operationen definiert sind!

§2-32 Theorem. (Elementare Rechenregeln für Mengen)

Seien A, B, C beliebige Mengen aus einer Grundmenge G .

Namen:	Regel/Gesetz:
Idempotenz-G.	$A \cap A = A$ $A \cup A = A$
Kommutativ-G.	$A \cap B = B \cap A$ $A \cup B = B \cup A$
Identitäts-G.	$A \cap G = A$ $A \cup \emptyset = A$ $A \cap \emptyset = \emptyset$ $A \cup G = G$
Assoziativ-G.	$(A \cap B) \cap C = A \cap (B \cap C)$ $(A \cup B) \cup C = A \cup (B \cup C)$
Absorptions-G.	$A \cap (A \cup B) = A$ $A \cup (A \cap B) = A$
Distributivitäts-G.	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
De Morgan Gesetze	$(A \cap B)^c = A^c \cup B^c$ $(A \cup B)^c = A^c \cap B^c$
Komplement-G.	$A \cap A^c = \emptyset$ $A \cup A^c = G$ $(A^c)^c = A$ $G^c = \emptyset$ $\emptyset^c = G$
Teilmengenbeziehungen	$A \subseteq B \implies (A \cap B = A)$ $A \subseteq B \implies (A \cup B = B)$
Transitivität der Teilmengenbez.:	$(A \subseteq B) \wedge (B \subseteq C) \implies (A \subseteq C)$

◇

§2-33 Beispiel.

Aufgabe:

Zeigen Sie, dass für die symmetrische Differenz die praktische Gleichung

$$A \Delta B = (A \setminus B) \cup (B \setminus A)$$

gilt. (Diese Formel wird manchmal sogar als Definition der symmetrischen Differenz verwendet!)

Lösung:

Wir haben:

$$\begin{aligned} A \Delta B &= \{x \in G \mid (x \in A \wedge x \notin B) \vee (x \notin A \wedge x \in B)\} && (\text{Def. von } \Delta) \\ &= \{x \in G \mid x \in A \wedge x \notin B\} \cup \{x \in G \mid x \notin A \wedge x \in B\} && (\text{Def. von } \cup) \\ &= \underbrace{(A \setminus B)}_{\substack{\text{Def. von } \setminus}} \cup \underbrace{(B \setminus A)}_{\substack{\text{Def. von } \setminus}}. \end{aligned}$$

◊

2.5 Weiteres zum kartesischen Produkt von Mengen

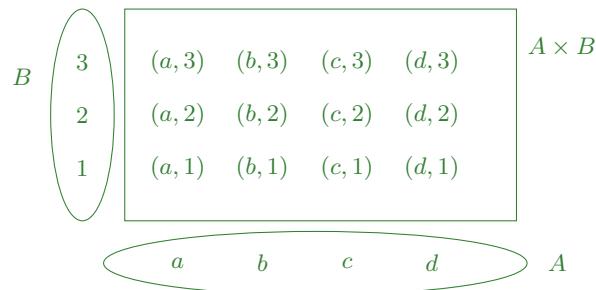
Das kartesische Produkt nimmt eine gewisse Sonderstellung ein. Warum? Ein paar Gründe dafür sind folgende:

- Es kann aus Mengen unterschiedlicher Grundmengen viele nicht-triviale neue Mengen bilden,
- es hat etwas andere rechnerische Eigenschaften (siehe anschliessend),
- es ist wichtig bei der Bildung von zwei der zentralsten Begriffe der Mathematik, den “Relationen” und den “Funktionen” (siehe spätere Kapitel),
- es erlaubt den Begriff der “Paarung” (folgt) und damit den Vergleich der Mächtigkeit von Mengen, auch von unendlichen Mengen (!) ↠ letztes Thema dieses Kapitels.

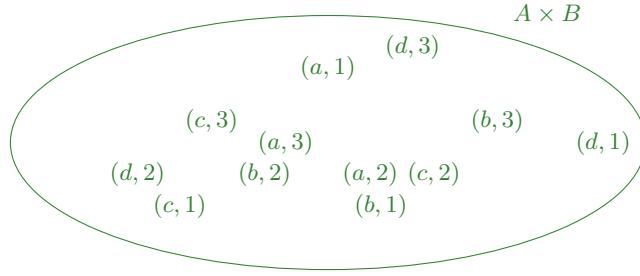
§2-34 Beispiel.

Aufgabe:

Zeichnen Sie das Venn-Diagramm des kartesischen Produkts $A \times B$ der beiden Mengen $A := \{a, b, c, d\}$ und $B := \{1, 2, 3\}$.

Lösung:

oder völlig äquivalent:



Die erste Variante das Venn-Diagramm zu zeichnen bietet sich aber an, da die Struktur des kartesischen Produkts besser ersichtlich ist. \diamond

§2-35 Theorem. (Elementare Eigenschaften des kartesischen Produkts)

Seien A, A_i, B, B_i, C, D beliebige Mengen, $1 \leq i \leq n$ und $n \in \mathbb{N}$. Dann gilt: (Beachten Sie: womöglich können Grundmengen verschieden sein!)

Stichwort/Name:	Regel/Gesetz:
Gleichheit	$(A_1 \times \dots \times A_n = B_1 \times \dots \times B_n) \iff (\forall i \in \mathbb{N}_n : A_i = B_i)$
Endliche Kardinalität	$ A \times B = A B $ $ A_1 \times \dots \times A_n = A_1 \cdot \dots \cdot A_n $
Unendliche Kardinalität	$ A_1 \times \dots \times A_n = \infty$ $\iff \exists i \in \mathbb{N}_n : A_i = \infty$
Nicht-Kommutativität	$A \times B \neq B \times A$
Nicht-Assoziativität	$A \times (B \times C) \neq (A \times B) \times C$
Mengendurchschnitt und \times - Distributivität	$(A \cap B) \times (C \cap D) = (A \times C) \cap (B \times D)$ $A \times (B \cap C) = (A \times B) \cap (A \times C)$
Mengenvereinigung und \times - Distributivität	$(A \cup B) \times (C \cup D) \neq (A \times C) \cup (B \times D)$ $A \times (B \cup C) = (A \times B) \cup (A \times C)$
Mengendifferenz und \times - Distributivität	$(A \times B) \setminus (C \times D)$ $= ((A \setminus C) \times B) \cup (A \times (B \setminus D))$ $A \times (B \setminus C) = (A \times B) \setminus (A \times C)$
Mengenkomplement und \times	$(A \times B)^c$ $= (A^c \times B^c) \cup (A^c \times B) \cup (A \times B^c)$
Teilmengenbeziehung und \times Falls $A, B \neq \emptyset$	$A \subseteq B \implies A \times C \subseteq B \times C$ $A \times B \subseteq C \times D \iff A \subseteq C \wedge B \subseteq D$

\diamond

2.6 Die Partition einer Menge

Was heisst es, wenn wir in der Informatik ein Laufwerk "partitionieren"?



Abbildung 2.1 – Partitionierungen spielen in mehrerer Hinsicht bei (älteren) Laufwerken eine Rolle.

(By User:Omegatron - Created by User:Omegatron using the GIMP, GPL,
ByEvan-Amos-Ownwork,CCBY-SA3.0,<https://commons.wikimedia.org/w/index.php?curid=27940250>)

Im einfachsten Fall hat man in einem Computersystem ein grösseres physisches Laufwerk (Festplatte), welches die gesamte Menge all seiner elementaren Speicherblöcke in mehrere logische Laufwerke "partitioniert", das heisst aufteilt. Zum Beispiel gibt es dann eine kleine Boot-Partition, eine Linux-Betriebssystem Partition, eine Linux-SWAP-Partition, eine Windows-Betriebssystem Partition und eine (oder mehrere) Partition(en) für die Daten. So eine "Partitionierung" eines Laufwerks ist genau das, was wir in der Mathematik und in der Mengenlehre als die Unterteilung einer Menge in "Partitionen" nennen. Einzig beim Sprachgebrauch muss man ein bisschen aufpassen: In der Mengenlehre spricht man oft auch von der Partition einer Menge und meint damit eigentlich die Partitionierung. In Anwendungen wie der Informatik meint man mit Partition meist eine einzelne Teilmenge einer Partitionierung (wie oben im Eingangsbeispiel). Aber das ist nur ein leichter sprachlicher Unterschied. Das Konzept dahinter ist völlig identisch:

§2-36 Definition. (Partition einer Menge)

Sei A eine beliebige Menge.

Ein Mengensystem $\mathcal{S} := \{T_1, T_2, \dots\}$ von nichtleeren Teilmengen $T_i \subseteq A$ nennen wir eine **Partition von A** , genau dann wenn

- (i) $T_1 \cup T_2 \cup \dots = A$. (Andere Schreibweise: $\bigcup_{T \in \mathcal{S}} T = A$)
- (ii) $T_i \cap T_j = \emptyset$, $\forall i \neq j$. (D.h., alle Mengen sind paarweise disjunkt.)

◇

Wir sehen, dies ist genau das, was wir aufgrund unseres Eingangsbeispiels einer Laufwerk-Partitionierung erwarten: Die einzelnen Partitionen vereinigt ergeben die Ausgangsmenge ((i) jeder Speicherblock ist einer Teilmenge zugeordnet und kann so auch benutzt werden) und die einzelnen Teilmengen der Partitionierung überschneiden sich nicht (ii), kein Speicherblock kann zu zwei verschiedenen Teilmengen gehören).

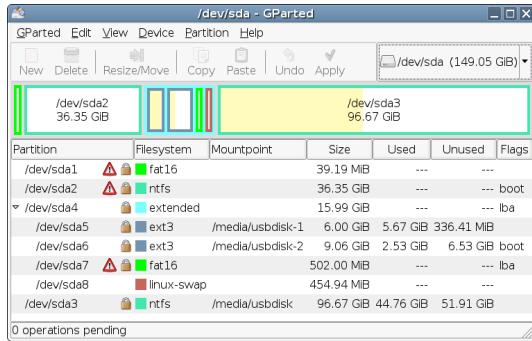


Abbildung 2.2 – Partitionierung der physischen Laufwerke in logische Laufwerke.

(By User:Omegatron - Created by User:Omegatron using the GIMP, GPL,

<https://commons.wikimedia.org/w/index.php?curid=1930356>)

Partitionen gibt es in der Realität wie Sand am Meer. (Die Sandkörner partitionieren übrigens einen Sandstrand ebenfalls, wenn auch in trivialer Weise: Jedes einzelne Sandkorn bildet für sich alleine eine Partition. Zusammengenommen vereinigen sie sich zum Sandstrand und jedes Sandkorn überschneidet sich nicht mit einem anderen Sandkorn.) Schon im Zusammenhang mit Computer-Laufwerken können wir weitere Partitionierungen entdecken:

§2-37 Beispiel. (Geometrischer Aufbau eines HDD-Laufwerks)

Ein älteres Harddisk-Laufwerk (HDD, hard disc drive) ist geometrisch folgendermassen aufgebaut:

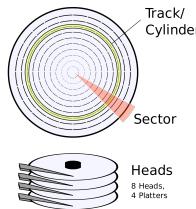


Abbildung 2.3 – Zwei verschiedene Partitionierungen der Menge aller Speichereinheiten eins Laufwerks.

(By LionKimb0 - Own work, Public Domain, <https://commons.wikimedia.org/w/index.php?curid=331092>)

Typischerweise hat so eine HDD z.B. vier Platten und damit insgesamt acht Schreib-/Leseköpfe (Heads) $\mathcal{H} := \{H_1, H_2, \dots, H_8\}$. [E.10] Weiter gibt es eine Anzahl sogenannter Zylinder $\mathcal{Z} := \{Z_1, \dots, Z_m\}$, welche dreidimensionale, konzentrische Schnitte durch den Stapel der Einzelplatten darstellen. Ebenso gibt es eine Anzahl $\mathcal{S} := \{S_1, \dots, S_n\}$ von ebenfalls dreidimensional durch den Plattenstapel geschnittenen sogenannten Sektoren.

Zwei verschiedene Partitionierungen der Menge M aller Speichereinheiten eines ganzen Laufwerks sind dann

$$M = \{Z_1, \dots, Z_m\}$$

sowie

$$M = \{S_1, \dots, S_n\}.$$

Aus Sicht eines ganzen HD-Laufwerkes kann man dann übrigens über die sogenannte CHS-Koordinaten (Cylinder, Head, Sector) eindeutig jede einzelne Speichereinheit durch das Tripel $(i, j, k) := (Z_i, H_j, S_k)$ ansprechen. [E.11] ◇

§2-38 Beispiel. (RAID-Systeme)

In sogenannten RAID-Systemen ("Redundant Array of Independent Disks") wird der Inhalt einer einzelnen Speichereinheit in verschiedene Partitionen auf *physisch unterschiedlichen Laufwerken* (HDD's oder SSD's) geschrieben, so dass Daten sicherer vor Harddisk-Crashes gespeichert werden können. Im einfachsten Fall, RAID-1, entspricht dies einer einfachen Plattenspiegelung. Die Daten werden redundant in je zwei Speichereinheiten auf Partitionen verschiedener HDD's geschrieben. Das Ganze lässt sich noch verfeinern (RAID-2, -3, -4,...) und das Ziel ist es, bei einem Defekt eines Laufwerks so keinen Datenverlust zu erleiden. Man kann unter Umständen sogar das defekte Laufwerk ohne Betriebsunterbruch und bei laufendem System ("hot swap") ersetzen. Ebenso ist eine Speicherplatz-Vergrosserung ohne Unterbruch möglich. ◇

In der Mathematik selbst treten Partitionen an unterschiedlichen Stellen auf: Zum Beispiel kann die Untersuchung der Elemente einer Mengen manchmal vereinfacht oder abgekürzt werden, indem man eine Partitionierung in Teilmengen, in welchen sich die Elemente jeweils "gleich verhalten", betrachtet. Oder man kann einen Satz leichter beweisen, wenn man die Menge der betre enden Elemente worüber der Satz eine Aussage macht, geeignet unterteilt und den Satz separat für diese Teile (Partitionen) beweist. Sie kennen das vielleicht schon als "Fallunterscheidungen" bei mathematischen Betrachtungen.

§2-39 Beispiel.

Die ganzen Zahlen \mathbb{Z} lassen sich manchmal einfacher getrennt in der Partition in gerade und ungerade Zahlen untersuchen. D.h., wir machen die Partitionierung

$$\begin{aligned}\mathbb{Z} &= \{\{m \in \mathbb{Z} \mid m \text{ gerade}\}, \{m \in \mathbb{Z} \mid m \text{ ungerade}\}\} \\ &\text{oder ganz formal:} \\ &= \{\{m \in \mathbb{Z} \mid \exists k \in \mathbb{Z} : m = 2k\}, \{m \in \mathbb{Z} \mid \exists k \in \mathbb{Z} : m = 2k + 1\}\}\end{aligned}$$

◇

§2-40 Beispiel.

Die reellen Zahlen sind oft einfacher zu handhaben, wenn wir sie in negative reelle Zahlen $\mathbb{R}_{<0}$, in positive reelle Zahlen $\mathbb{R}_{>0}$ und den Spezialfall der 0 unterteilen. D.h., wir machen folgende Partition der reellen Zahlen:

$$\mathbb{R} = \{\mathbb{R}_{<0}, \{0\}, \mathbb{R}_{>0}\}.$$

◇

Die Anzahl Partitionen einer Menge M mit n Elementen lässt sich übrigens durch die sogenannte Bell'sche Zahl B_n angeben. Diese wächst mit wachsender Grösse n der zugrundeliegenden Menge M sehr stark an:

Mächtigkeit der Menge n	2	4	6	10	16
Bell'sche Zahl (Anz. Partitionen) B_n	2	15	203	115'975	10'480'142'147

Abschliessend sei noch erwähnt, dass es eine enge Beziehung zwischen der Partitionierung einer Menge und sogenannten "Äquivalenzrelationen" auf dieser Menge (= best. Typ von "Relation") gibt. Wir werden im Thema "Relationen und Funktionen" und dort in Abschnitt 5.4 darauf zurückkommen.

2.7 Die Unendlichkeit von Mengen

Wir haben gesehen, dass wir die Mächtigkeiten (Kardinalitäten) von endlichen Mengen durch Zählen der Elemente miteinander vergleichen können. Müssen wir jedoch “unendlich lange zählen” (d.h. gibt es kein $n \in \mathbb{N}$ so dass $|A| = n$ ist), dann haben wir die Menge unendlich genannt und $|A| = \infty$ gesetzt. Zwei wichtige Fragen stellen sich im Zusammenhang mit solchen unendlichen Mengen:

- Lassen sich auch die Kardinalitäten von unendlichen Mengen vergleichen?
- Gibt es “verschiedene Unendlichkeiten”?

Die Antworten auf diese beiden Fragen, sind zweimal ja! Diesen interessanten Umstand wollen wir nun noch etwas näher untersuchen und dazu brauchen wir noch einen wichtigen Begri :

§2-41 Definition. (Paarung zweier Mengen)

Seien A und B zwei beliebige (endliche oder unendliche) Mengen.

Dann heisst eine Teilmenge

$$T := \{(a, b) \mid a \in A, b \in B\} \subseteq A \times B$$

des kartesischen Produkts von A und B eine **Paarung von A und B** , gdw. $\forall (a, b) \in T$ gilt:

- (i) $\forall a \in A \exists !b \in B : (a, b) \in T,$
- (ii) $\forall b \in B \exists !a \in A : (a, b) \in T.$

◊

§2-42 Beispiele.

Seien zwei Mengen $A := \{1, 2, 3, 4\}$ und $B := \{a, b, c, d\}$ gegeben.

- (1) $T_1 := \{(1, d), (2, b), (3, a), (4, c)\}$ ist eine Paarung von A und B .
- (2) $T_2 := \{(1, c), (2, b), (3, d), (4, a)\}$ ist eine andere Paarung dieser Mengen A und B .
- (3) Keine Paarungen von diesen A und B sind:

$$\begin{aligned} T_3 &:= \{(1, d), (2, b), (3, c)\}, \\ T_4 &:= \{(1, a), (3, b), (4, d), (3, c)\}, \\ T_5 &:= \{(2, a), (3, c), (4, c), (1, b)\}. \end{aligned}$$

- (4) $T := \{(0, 0), (1, -1), (2, -2), (3, -3), \dots\}$ ist eine Paarung von \mathbb{N} und von $\mathbb{Z}^- \cup \{0\}$.

◊

Mittels einer Paarung können wir nun definieren:

§2-43 Definition. (Gleichmächtigkeit zweier Mengen)

Zwei Mengen A und B (endlich oder unendlich) haben gleiche Mächtigkeit (gleiche Kardinalität) $|A| = |B|$, gdw. es eine Paarung von A und B gibt. ◊

Eine Paarung mit \mathbb{N} erlaubt es uns, eine Menge A auf eine oft nützliche Weise zu betrachten:

§2-44 Definition. (Nummerierung, Ordnung (Index))

Gibt es eine Paarung von \mathbb{N} mit einer Menge A , so kann man diese Paarung als **Nummerierung der Elemente von $A := \{a^*, \tilde{a}, \hat{a}, \dots\}$** auffassen:

$$\{(0, \underbrace{a^*}_{=:a_0}), (1, \underbrace{\tilde{a}}_{=:a_1}), (2, \underbrace{\hat{a}}_{=:a_2}), \dots\}$$

Wir nennen dann das n des Elements a_n auch die **Ordnung dieses Elementes oder den Index von a_n** . \diamond

Und mit Hilfe der Paarung können wir auch eine andere, gleichwertige Definition der unendlichen Mengen formulieren:

§2-45 Definition.

- (1) Eine Menge A heisst **unendlich**, gdw. es eine echte Teilmenge $T \subsetneq A$ gibt, so dass A und T eine Paarung sind.

Beispiel: Auf diese Weise sehen wir für $A := \mathbb{N}$, dass die Menge der natürlichen Zahlen \mathbb{N} (inkl. 0) unendlich ist. Denn durch

$$\{(0, 2), (1, 3), (2, 4), (3, 5), \dots\}$$

(das zweite Element im Paar immer +2) haben wir eine Paarung von $A := \mathbb{N}$ und $T := \{2, 3, 4, 5, \dots\}$, für welche $T \subsetneq \mathbb{N}$ gilt (0 und 1 fehlen ja in T).

- (2) Eine Menge heisst **abzählbar unendlich**, wenn es eine Paarung mit \mathbb{N} gibt, d.h., wenn sie gleichmächtig wie \mathbb{N} ist. Endliche und abzählbar unendliche Mengen nennen wir zusammen auch **abzählbare Mengen**.

Beispiel: $\mathbb{Z}^-, \mathbb{Z}^+, \mathbb{Z}$, die Menge \mathbb{N}_g der geraden und die Menge \mathbb{N}_u der ungeraden natürlichen Zahlen sind alle **gleichmächtig** wie \mathbb{N} , also abzählbar unendlich (wieso?).

Weiter ist auch \mathbb{Q} abzählbar unendlich (!).

Mit anderen Worten:

$$|\mathbb{Q}| = |\mathbb{Z}^-| = |\mathbb{Z}^+| = |\mathbb{Z}| = |\mathbb{N}_g| = |\mathbb{N}_u| = |\mathbb{N}|.$$

- (3) Eine Menge heisst **überabzählbar unendlich**, wenn sie unendlich ist und nicht abzählbar (d.h. "mächtiger" als \mathbb{N}).

Beispiel: \mathbb{R} ist überabzählbar unendlich. D.h.: $|\mathbb{N}| < |\mathbb{R}|$. \diamond

Unendliche Mengen verhalten sich jedoch bedeutend unintuitiver und "problematischer" als endliche Mengen. Dazu eine seltsame Geschichte:

§2-46 Beispiel. (Hilbert's Hotel – Teil 1)

Hilbert's Hotel ist ein Hotel mit *abzählbar unendlich* vielen Zimmern. D.h., wir haben Zimmer Z_1, Z_2, \dots bzw. $\forall i \in \mathbb{N} \setminus \{0\} \exists! \text{Zimmer mit Bezeichnung } Z_i$.

Am heutigen Tag sind alle Zimmer belegt. Ein neuer weiterer Gast – eben in Hilbert's ausgebuchtem Hotel angekommen – bekommt jedoch trotzdem noch ein (freies) Zimmer!

Aufgabe:

Wie ist Teil 1 von Hilbert's Hotel möglich?

Lösung:

Die Lösung ist nicht schwer und hat mit dem Begriff "unendlich" zu tun. Die Welt der unendlichen Mengen ist ziemlich anders als die der endlichen!

Wir buchen einfach Gast von Zimmer Z_1 um auf Zimmer Z_2 , Gast von Zimmer Z_2 auf Zimmer Z_3 usw. Dann hat jeder der bisherigen Gäste wieder ein Zimmer und Zimmer Z_1 ist frei, bereits zum Einzug für unseren neuen Guest. :-)

◇

Es geht aber noch verrückter:

§2-47 Beispiel. (Hilbert's Hotel – Teil 2)

Seien nun in Hilbert's Hotel wiederum alle Zimmer Z_1, Z_2, \dots durch Gäste G_1, G_2, \dots besetzt. Nun trifft ein (unendlicher) Bus mit abzählbar unendlich vielen neuen Gästen NG_1, NG_2, \dots ein. Selbst jetzt lassen sich wiederum alle diese abzählbar unendlich vielen neuen Gäste im bereits vollen Hotel unterbringen!

Aufgabe:

Wie ist sogar Teil 2 von Hilbert's Hotel möglich?

Lösung:

Sogar das geht, wiederum dank der Unendlichkeit der Menge von Zimmern!

Wir buchen jetzt die ehemaligen Gäste G_1, G_2, G_3, \dots der Zimmer Z_1, Z_2, Z_3, \dots um auf die Zimmer Z_2, Z_4, Z_6, \dots D.h.,

$$G_k \mapsto Z_{2k}, \quad \forall k \in \mathbb{N} \setminus \{0\}.$$

So hat jeder der ehemaligen Gäste wieder ein Zimmer für sich bekommen und gleichzeitig haben wir alle Zimmer mit ungeraden Nummern Z_1, Z_3, Z_5, \dots frei für die neuen Gäste NG_1, NG_2, NG_3, \dots . D.h., diesen ordnen wir gemäss

$$NG_k \mapsto Z_{2k-1}, \quad \forall k \in \mathbb{N} \setminus \{0\},$$

ihre Zimmer zu. Mission accomplished! :-)

◇

Wir sehen: Der Begriff der Unendlichkeit ist in der Mathematik sehr tiefgehend, zudem unintuitiv und auch problematisch. Es gibt unter den Mathematikern verschiedene Assumptions darüber, was man vom Begriff der Unendlichkeit halten bzw. wie man ihn handhaben soll. Dieses Thema hier fortzuführen würde zu weit gehen, aber es lohnt sich schon einmal, bis hierhin der Problematik bewusst zu sein. Noch eine andere fundamentale Frage sei zum Schluss erwähnt:

Gibt es eine Menge A mit $|\mathbb{N}| < |A| < |\mathbb{R}|$?

Kurt Gödel (1940) und Paul Cohen (1963, 1964) haben gezeigt, dass diese Frage unentscheidbar ist! Neben dem weiteren delikaten Phänomen unentscheidbarer Probleme in der Mathematik, führt obige Frage also zur Unentscheidbarkeit der berühmten

§2-48 Definition. (Kontinuumshypothese (Continuum Hypothesis (CH)))

Es gibt keine Menge, deren Mächtigkeit (strikt) zwischen den Mächtigkeiten von \mathbb{N} und \mathbb{R} liegt. \diamond

Das heisst, wir haben:

§2-49 Theorem. (Unentscheidbarkeit der Kontinuumshypothese; Gödel (1940) und Paul Cohen (1963, 1964))

Die Kontinuumshypothese ist auf der heute gängigen Grundlage der Mathematik^[E.12] unentscheidbar. \diamond

2.8 Übersicht: Gleichheits- und Äquivalenznotationen in Logik und Mengenlehre

Beachten Sie für die folgenden Aufstellungen nochmals, dass dies letztlich alles nur eine Konvention zur Notation (Schreibweise) ist. Leider hat sich dies nicht in allen Fachgebieten und bei allen Autoren genau gleich "eingebürgert". Die Notation mit dem Doppelpunkt für eine Definition wird zudem nicht immer konsequent eingesetzt, v.a. wenn aus dem Kontext klar ist, dass es sich um eine Definition handelt (z.B. "Sei $A = \{a, b, c, d\} \dots$ ").

Objekte:	Syntaktische Gleichheit bzw. Äquivalenz	Semantische Gleichheit bzw. Äquivalenz
Aussagen ¹ A, B	$A = B$ in Def.: $A := B$	$A \iff B$ in Def.: $A : \iff B$
Aussagenlogische Formeln f, g	$f = g$ in Def.: $f := g$	$f \equiv g$ in Def.: $f : \equiv g$
Prädikate $P(x), Q(x)$	$P(x) = Q(x)$ in Def.: $P(x) := Q(x)$	$P(x) \iff Q(x)$ in Def.: $P(x) : \iff Q(x)$
Prädikatenlogische Formeln ³ f, g	$f = g$ in Def.: $f := g$	$f \equiv g$ in Def.: $f : \equiv g$

Tabelle 2.3 – Konventionen zur Bezeichnung von Gleichheiten / Äquivalenzen in aussagen- und prädikatenlogischen Ausdrücken.

¹ Für atomare Aussagen haben wir meistens A, B, \dots verwendet, für längere, nicht-atomare aussagenlogische Formeln f, g, \dots usw.

² Das Symbol \equiv wird ziemlich selten benutzt.

³ Sind einfach aussagenlogische Formeln mit Quantoren. Wenn sie kurz sind und insbesondere selbst keine \iff enthalten, verwendet man wie in der Aussagenlogik auch \iff für die semantische Äquivalenz selbst.

⁴ Beachten Sie auch, dass algebraische "Äquivalenzumformungen" wie z.B. $y + 2 = x \iff y = x - 2$ semantische Äquivalenzen von aussagen- bzw. meist prädikatenlogischen Formeln sind!

Objekte:	Gleichheit:
Mengen A, B	$A = B$ in Def.: $A := B$
Algebraische Ausdrücke ⁴	$(x + y)^2 = x^2 + 2xy + 2y$ in Def.: $x := 2^y$ manchmal auch "Identität": $\frac{3}{9} \equiv \frac{1}{3}$

Tabelle 2.4 – Konventionen zur Bezeichnung von Gleichheiten bei mengentheoretischen und algebraischen Ausdrücken.

Kapitel 3

Beweistechniken

Kapitelinhalt

3.1	Einleitung: Beweise in der Mathematik	67
3.2	Beweisen versus Gegenbeispiel finden	67
3.3	Direkter Beweis	69
3.4	Beweis durch Kontraposition	72
3.5	Indirekter Beweis (“Widerspruchsbeweis”)	74

L E R N Z I E L E :

- LZ 3.1** Sie haben einen ersten etwas systematischeren Überblick über die gängisten Beweistechniken (d.h., wie man deduktiv neues Wissen ableitet).
- LZ 3.2** Sie können diese Techniken an einfachen typischen Beispielen nachvollziehen und selber anwenden.
-

3.1 Einleitung: Beweise in der Mathematik

Mathematik ist im Wesentlichen eine deduktive Wissenschaft und damit verbunden ebenso gewisse theoretische Teile anderer Wissenschaften wie z.B. die theoretische Informatik oder die theoretische Physik. Deduktiv heisst, dass man von ein paar wenigen Grundannahmen ("Axiomen") ausgehend mittels exakter logischer Regeln und Begriffe definitionen weitere Aussagen (also "Wissen") ableitet ("deduziert"). Dabei können auch frühere, auf analoge Weise bewiesene Tatsachen (Sätze, Theoreme) verwendet werden. Damit lässt sich Wissen logisch "beweisen". Eine Argumentationskette, die eine Aussage aus Axiomen, Definitionen und bereits bewiesenen Sätzen als richtig begründet, nennen wir in der Mathematik und ihren angrenzenden Gebieten einen Beweis.

Dies steht im Gegensatz zu den sogenannt "empirischen" bzw. induktiven Wissenschaften, welche Wissen aus Beobachtungen, Messungen und Experimenten gewinnen und so induktiv von Einzelerkenntnissen auf allgemeine Erkenntnisse (allgemeingültiges Wissen) schliessen. Dies nennt man auch die induktive Methode und ist eine typische Vorgehensweise in vielen anderen Teilbereichen der Wissenschaften (Experimentalphysik, grosse Bereiche der anderen Naturwissenschaften, Sozial- und Wirtschaftswissenschaften, usw.). Vorsicht an dieser Stelle: Die mathematische Beweismethode der "vollständigen Induktion", welche wir später in Teil III kennenlernen werden, gehört aber gerade nicht zu einer induktiven Methode. Sondern sie ist ganz im ersteren Sinn eine deduktive Methode, und liefert einen echten mathematischen Beweis. Der Grund ist, weil diese Methode eben "vollständig" ist bzw. "vollständig induziert" (!). Aber mehr dazu im entsprechenden Teil.

In Texten ist es üblich, den Schluss eines Beweises mit einem Quadrat \square oder der Abkürzung q.e.d. bzw. qed (quod erat demonstrandum für lat. "was zu zeigen war") zu markieren.

3.2 Beweisen versus Gegenbeispiel finden

Wenn man eine logische Aussage vor sich hat und man wissen bzw. untersuchen möchte, ob sie wahr oder falsch ist, dann bietet es sich oft an, sich zuerst zu überlegen, ob die Aussage möglicherweise falsch sein könnte. Wieso? Der Grund ist der, dass um die Nicht-Gültigkeit einer Aussage über eine Vielzahl von Objekten einer ganzen Menge zu zeigen es genügt, ein einziges Gegenbeispiel zu finden. Das heisst, wir brauchen nur ein einziges konkretes Beispiel, wofür die Aussage nicht gilt und dies kann häufig bedeutend einfacher sein zu finden, als versuchen zu zeigen, dass die Aussage allgemein gilt. Und dann rechtzeitig zu merken, dass man nicht einfach den Beweis noch nicht gefunden hat, sondern dass es ihn nicht gibt, weil die Aussage falsch ist^[E.13].

Selbstverständlich ist das Gesagte eine "Heuristik". D.h., es ist ein oft sinnvolles und zum Ziel führendes Vorgehen, aber ohne Garantie, dass es in allen Fällen das richtige bzw. ein erfolgreiches Vorgehen ist. Schauen wir uns dies an zwei Beispielen an:

§3-1 Beispiel.

Untersuchen Sie, ob

$$\forall x, y \in \mathbb{R} : \quad 3^{x+y} = 3^x + 3^y$$

gilt.

Hier sehen wir sofort mit dem Gegenbeispiel für $x = y = 1$, dass einseits

$$3^{1+1} = 3^1 = 3,$$

jedoch andererseits

$$3^1 + 3^1 = 3 + 3 = 6$$

gilt. Damit ist die Behauptung schon widerlegt. ◇

§3-2 Beispiel.

Manchmal ist es so, dass irgendwelche spezielleren Werte aus einer grossen Menge von Objekten, besonders geeignet sind solche Gegenbeispiele zu finden. (Eine Aussage gilt vielleicht für die "allgemeinen Fälle", aber eben gerade nicht für ein paar Spezialfälle.) Betrachten wir zum Beispiel die Aussage

"Jede natürliche Zahl $n > 1$ ist entweder gerade Zahl oder eine ungerade Primzahl oder eine ungerade zusammengesetzte Zahl."

Vielleicht meint man auf den ersten Blick, dies sei richtig und man beginnt mit dem Suchen eines Beweises. Überlegt man sich die Aussage aber für ein paar spezielle Werte, hier die ersten paar natürliche Zahlen, so sieht man sofort anhand des Gegenbeispiels 2 (die einzige gerade Primzahl), dass die Aussage nicht wahr ist. Das heisst, dieses Gegenbeispiel widerlegt die Aussage.

Im Fall dieser konkreten Aussagen ist es ja sogar so, dass sie tatsächlich für alle natürlichen Zahlen > 1 ausser der 2 wahr wäre. Diese einzige Ausnahme macht die Aussage gerade falsch. ◇

Im Zusammenhang mit Aufgaben zu mathematischen Beweisen hat es sich eingebürgert, dass man folgende Konvention betre end den Formulierungen tri t und wir wollen uns ebenfalls daran halten:

§3-3 Definition. (Konvention zur Formulierung von Beweisaufgaben)

- Wird in einer Aufgabe eine Formulierung in der Art

"Zeigen Sie, dass [...]", "Beweisen Sie, dass [...]"

gewählt, dann können Sie davon ausgehen, dass die zu zeigende Aussage tatsächlich wahr ist.

Sie brauchen also nicht zuerst zu überlegen, ob sie ebenfalls doch falsch ist und die Suche nach einem Gegenbeispiel angebracht sein könnte. Die Wahrheit der Aussage kann von Ihnen als gegeben betrachtet werden und es ist nur der Beweis, nicht aber die Antwort "wahr" oder "falsch" gesucht.

- Ist hingegen in einer Aufgabe eine o enere Formulierung gewählt, wie zum Beispiel

"Untersuchen Sie, ob [...]", "Ist folgende Aussage wahr [...] ?",

dann ist die Antwort, ob die Aussage "wahr" oder "falsch" sei, tatsächlich o en gelassen.

Es macht dann unter Umständen Sinn, oben vorgestellter Heuristik zu folgen und sich zu überlegen, ob man ein Gegenbeispiel finden und damit die Nicht-Gültigkeit der Aussage zeigen könnte.



3.3 Direkter Beweis

Der Beweistechnik eines "direkten Beweises" liegt folgende Eigenschaft der Implikation zugrunde, welche wir uns zuerst an einem einfachen Beispiel überlegen:

§3-4 Beispiel. ("Transitivitätseigenschaft" der Implikation)

Seien die folgenden Aussagen gegeben

$$\begin{aligned} A &:= \text{"Es regnet."} \\ B &:= \text{"Die Wiese ist nass."} \\ C &:= \text{"Die Salamander freuen sich."} \end{aligned}$$

Wir nehmen an, dass die folgenden Implikationen gelten:

$$A \implies B$$

D.h., wenn es regnet, dann ist die Wiese nass.

$$B \implies C$$

D.h., wenn die Wiese nass ist, dann freuen sich die Salamander.

Wir betrachten nun die Beziehung zwischen A und C . Was können wir darüber sagen? Wir werden wahrscheinlich intuitiv (und korrekt) sagen, dass dann auch

$$A \implies C$$

gilt. D.h., dass wenn es regnet, sich die Salamander freuen. Wie können wir aber verstehen bzw. sogar exakt beweisen, dass dies korrekt ist?

Wir wollen also zeigen, dass wenn wir allgemein drei beliebige Aussagen A, B, C haben, die Aussage

$$(*) \quad ((A \implies B) \wedge (B \implies C)) \implies (A \implies C)$$

immer wahr ist, d.h., eine Tautologie ist. Der Beweis dieses Satzes (*) sieht folgendermassen aus:

Beweis. Wir beweisen (*) mit der Wahrheitstabelle:

A	B	C	$A \implies B$	$B \implies C$	$(A \implies B) \wedge (B \implies C)$	$A \implies C$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	1	0	0	1
0	1	1	1	1	1	1
1	0	0	0	1	0	0
1	0	1	0	1	0	1
1	1	0	1	0	0	0
1	1	1	1	1	1	1

A	B	C	$((A \implies B) \wedge (B \implies C)) \implies (A \implies C)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

↑
Tautologie

Die letzte Spalte beweist, dass (*) tatsächlich eine Tautologie ist, d.h., immer wahr ist. □
◊

Die eben gezeigte Eigenschaft nennt man die Transitivität der Implikation. Damit meint man, dass sich die beiden Implikationen $A \Rightarrow B$ und $B \Rightarrow C$ auf die Aussagen A und C übertragen (transitus (lat.), der "Übergang"), dass also auch $A \Rightarrow C$ gilt. Wir halten dies als wichtigen Satz fest:

§3-5 Theorem. (Transitivität der Implikation)

Wenn

$$A \Rightarrow B \text{ und } B \Rightarrow C$$

gilt, dann gilt auch

$$A \Rightarrow C.$$

◊

Dieser Satz lässt sich auf beliebig (endlich) viele Implikationen, auf sogenannte **Implikationsketten**, erweitern:

§3-6 Theorem. (Transitivität bei Implikationsketten)

Für jede natürliche Zahl $k > 1$ (welche eine Anzahl von Aussagen bezeichnet) gilt:

Wenn

$$A_1 \Rightarrow A_2,$$

$$A_2 \Rightarrow A_3,$$

$$A_3 \Rightarrow A_4,$$

⋮

$$A_{k-1} \Rightarrow A_k$$

gilt, dann gilt auch

$$A_1 \Rightarrow A_k.$$

◊

§3-7 Beispiel.

Gegeben sei als Prämisse die Aussage $A := "3x - 8 = 4"$.

Zu zeigen sei die Konklusion Z :

$Z := "x = 4 \text{ ist die einzige Lösung dieser Gleichung von Aussage } A."$

Ziel: Unser Ziel ist es, durch bekannte, bereits bewiesene Sätze, Rechenregeln, Äquivalenzumformungen ... usw. in mehreren Zwischenschritten $A \Rightarrow \dots \Rightarrow \dots \Rightarrow Z$ zu zeigen. Aufgrund des Satzes über die Transitivität der Implikation heißt dies dann, dass wir das zu Zeigende bewiesen haben. In unserem Fall geht dies sehr kurz:

Beweis.

$$\begin{aligned} 3x - 8 &= 4 \\ \implies 3x &= 12 && (\text{auf beiden Seiten } +8 \text{ gerechnet}) \\ \implies x &= 4 && (\text{auf beiden Seiten durch 3 dividiert}) \end{aligned}$$

Mit dem Satz über die Transitivität der Implikation haben wir also das Gewünschte gezeigt. Denn wenn A gilt und $A \implies Z$ gilt, muss auch Z gelten, da in unserem Fall alle benutzten Implikationen der Zwischenschritte wahr sind. □ ◇

Wir fassen dieses Vorgehen, welches einen direkten Beweis genannt wird, zusammen

§3-8 Definition. (Schema des direkten Beweises)

- **Gegeben:** Wahre Aussage A und “Werkzeuge” für die (Zwischen-) Implikationen (d. h. wahre Implikationen).
- **Zu zeigen:** Es gilt Z .

Methode:

1. Folge von Implikationen $A \implies B_1, B_1 \implies B_2, B_2 \implies B_3, \dots, B_k \implies Z$ finden, welche die Gültigkeit von $A \implies Z$ zeigen.
2. Aus der Gültigkeit von A und $A \implies Z$ kann auf die Gültigkeit von Z geschlossen werden.

◇

Weil es so wichtig ist, streichen wir nochmals hervor:

Vorsicht!

Im Beweis der Implikationskette dürfen nur Implikationen verwendet werden, deren Gültigkeit schon bewiesen sind.

Was könnte geschehen, wenn wir diese wichtige Warnung übersehen würden? Ein Beispiel dazu:

§3-9 Beispiel. (Mathematische Irrfahrt zum Beweis, dass $2 = 1$ ist)

Man könnte zum Beispiel irrtümlicherweise Folgendes folgern:

$$\begin{aligned} a &= b \\ \implies a^2 &= ab && (\cdot a \text{ gerechnet}) \\ \implies a^2 + a^2 - 2ab &= ab + a^2 - 2ab && (+ (a^2 - 2ab)) \\ \implies 2a^2 - 2ab &= a^2 - ab && (\text{zusammengefasst}) \\ \implies 2(a^2 - ab) &= a^2 - ab && (2 \text{ ausklammern}) \\ \implies 2 &= 1 && (: (a^2 - ab)). \end{aligned}$$

Wir haben gezeigt, dass

$$2 = 1$$

ist, was aber bekanntlich nicht so ist. Wo ist der Fehler?!? Oder ist jetzt plötzlich doch etwa $2 = 1?$;)

Wenn wir genau hinschauen, sehen wir, dass die letzte Implikation nicht gültig ist: Die Division durch $(a^2 - ab)$ ist nicht erlaubt, da wir in der ersten Zeile $a = b$ vorausgesetzt haben, also ist $a^2 - ab = 0$ und wir führen in der letzten Zeile eine Division durch 0 durch! \diamond

§3-10 Definition. (Gerade Zahl)

Eine ganze Zahl n heisst gerade, wenn sie ganzzahlig durch 2 teilbar ist. Das heisst, wenn

$$\frac{n}{2} \in \mathbb{Z}$$

oder äquivalent

$$\exists k \in \mathbb{Z} : n = 2k$$

gilt. \diamond

§3-11 Beispiel.

Gegeben: a und b sind zwei gerade Zahlen.

Zu zeigen: $a + b$ ist eine gerade Zahl.

Beweis.

Seien a und b zwei gerade Zahlen.

$$\begin{aligned} &\Rightarrow \exists i, j \in \mathbb{Z} : (a = 2i \wedge b = 2j) \\ &\Rightarrow \exists i, j \in \mathbb{Z} : (a + b = 2i + 2j) \\ &\Rightarrow \exists i, j \in \mathbb{Z} : (a + b = 2 \underbrace{(i + j)}_{=k}) \\ &\Rightarrow \exists k \in \mathbb{Z} : (a + b = 2k) \\ &\Rightarrow a + b \text{ ist gerade.} \end{aligned}$$

\square

\diamond

Bei Beweisen ist es wichtig, dass man die Definitionen – allenfalls sogar mehrere äquivalente Definitionen – in den zu beweisenden Aussagen gut kennt. Mit diesem “Werkzeug” lassen sich Aussagen in eine andere Form bringen, welche sich dann allenfalls besser weiter umformen lässt.

3.4 Beweis durch Kontraposition

Wir haben schon in §1-28 gezeigt:

§3-12 Theorem. (Kontraposition (Prinzip der Kontraposition))

$$A \implies B$$

ist äquivalent zu

$$\neg B \implies \neg A,$$

d. h. es gilt

$$(A \implies B) \equiv (\neg B \implies \neg A)$$

◊

Der Beweis durch Kontraposition wird für den Beweis von Aussagen der Form $A \implies B$ gebraucht, wenn sich kein direkter Beweis anbietet.

§3-13 Definition. (Schema eines Beweises durch Kontraposition)

- Gegeben: “Werkzeuge” für die Implikationen (d. h. wahre Implikationen)
- Zu zeigen: $A \implies Z$ gilt.

Methode:

1. Zeigen, dass $\neg Z \implies \neg A$.
2. Dann folgt nach dem Prinzip der Kontraposition (Satz §3-12), dass auch $A \implies Z$ gilt.

◊

Für das nachfolgende Beispiel brauchen wir folgende Definition, welche Sie sich auch unabhängig davon merken sollten:

§3-14 Definition. (Ungerade Zahl)

Eine ganze Zahl n heisst ungerade, wenn sie nicht gerade ist, also wenn es eine Zahl $k \in \mathbb{Z}$ gibt, so dass n geschrieben werden kann als

$$n = 2k + 1.$$

◊

§3-15 Beispiel.

Beh.: $\underbrace{\text{“}n^2 \text{ ist eine gerade Zahl.”}}_{=:A} \implies \underbrace{\text{“}n \text{ ist eine gerade Zahl.”}}_{=:Z}$

Beweis.

$$\begin{aligned}
 \neg Z &= "n \text{ ist eine ungerade Zahl.}" \\
 \implies n &= 2i + 1 \text{ für ein } i \in \mathbb{Z} && (\text{nach Def.}) \\
 \implies n^2 &= (2i + 1)^2 = 4i^2 + 4i + 1 = 2\underbrace{(2i^2 + 2i)}_{=k \in \mathbb{Z}} + 1 \text{ für ein } i \in \mathbb{Z} && (\text{Gl. quadrieren}) \\
 \implies n^2 &= "n^2 \text{ ist eine ungerade Zahl.}" = \neg A
 \end{aligned}$$

Also gilt mit dem Prinzip der Kontraposition, dass $A \implies Z$ ist. □ ◇

Beachten Sie noch: Die Äquivalenz $A \iff B$ von zwei Aussagen lässt sich oft am besten in zwei Schritten beweisen:

Schritt 1: $A \implies B$

Schritt 2: $B \implies A$

Für jeden dieser Schritte kann dann (unabhängig voneinander) entweder ein direkter Beweis oder ein Beweis durch Kontraposition gewählt werden!

3.5 Indirekter Beweis (“Widerspruchsbeweis”)

§3-16 Beispiel.

Seien wieder die folgenden Aussagen gegeben

$$\begin{aligned}
 A &:= \text{“Es regnet.”} \\
 B &:= \text{“Die Wiese ist nass.”} \\
 C &:= \text{“Die Salamander freuen sich.”}
 \end{aligned}$$

Wir nehmen auch wieder an, dass die folgenden Implikationen gelten:

- $A \implies B$
- $B \implies C$

Sie stellen jetzt aber fest, dass C nicht gilt, also dass $\neg C$ gilt. Was können Sie daraus schliessen?

Antwort: Dann gilt $\neg A$ und $\neg B$. Wieso?

Beweis. Dies können wir wieder durch Betrachtung der Wahrheitstabelle

A	B	C	$A \implies B$	$B \implies C$	$\neg C$
0	0	0	1	1	1
0	0	1	1	1	0
0	1	0	1	0	1
0	1	1	1	1	0
1	0	0	0	1	1
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	1	1	0

oder durch Betrachtung der Kontrapositionen von $A \implies B$ und $B \implies C$ und der Transitivität der Implikation zeigen.

(Beachten Sie, dass in der Wahrheitstabelle nur die *erste Zeile* der vorausgesetzten Situation entspricht und dass damit $\neg A$ und $\neg B$ gelten muss.) □ ◇

§3-17 Definition. (Schema eines indirekten Beweises (eines Widerspruchsbeweises))

Gegeben: “Werkzeuge” für die Implikationen (d. h. wahre Implikationen).

Zu zeigen: Es gilt Z .

Methode:

1. Folge von Implikationen $\neg Z \implies B_1, B_1 \implies B_2, B_2 \implies B_3, \dots, B_k \implies U$ finden mit einer Aussage U , die “Unsinn” ist, also die nicht wahr sein kann, weil sie im Widerspruch zu einer (bekannten) wahren Aussage $\neg U$ steht.
2. Dies impliziert die Gültigkeit von $\neg Z \implies U$.
3. Nach dem Prinzip der Kontraposition folgt daraus, dass $\neg U \implies Z$ gilt und somit dass Z gilt, weil $\neg U$ wahr ist.

◊

Weil wir in Schritt 1 ein Argument in einen Widerspruch führen (“ins Absurde führen”), nennen wir so einen Beweis eben auch Widerspruchsbeweis und das Vorgehen “reductio ad absurdum”.

§3-18 Beispiel.

Beh.: $Z := “\sqrt{2} \text{ ist keine rationale Zahl}”$.

Beweis.

$$\begin{aligned}
 & \neg Z = “\sqrt{2} \text{ ist eine rationale Zahl.”} \\
 \implies & \sqrt{2} = \frac{p}{q} \text{ für } p, q \in \mathbb{Z}, q \neq 0 \text{ und } \frac{p}{q} \text{ gekürzter Bruch,} \\
 & \text{d.h., } p \text{ und } q \text{ haben keinen gemeinsamen Teiler (*).} && (\text{nach Def.}) \\
 \implies & \sqrt{2}q = p && (\cdot q) \\
 \implies & 2q^2 = p^2, && (\text{d.h., } p^2 \text{ ist gerade}) \\
 \implies & 2q^2 = p^2 \text{ und } p \text{ ist gerade, d.h., } \exists k \in \mathbb{Z} : p = 2k && (\text{siehe §3-15}) \\
 \implies & 2q^2 = (2k)^2 = 4k^2 && (\text{einsetzen}) \\
 \implies & q^2 = 2k^2 && (\text{d.h., } q^2 \text{ ist gerade}) \\
 \implies & q \text{ ist gerade.} && (\text{siehe §3-15})
 \end{aligned}$$

Dies ist ein Widerspruch, denn wenn p und q beide gerade sind, dann haben sie mit 2 einen gemeinsamen Teiler, was im Widerspruch zu (*) steht, dass sie keine gemeinsamen Teiler haben. Nach dem Prinzip des indirekten Beweises zeigt das, dass Aussage Z wahr ist. □ ◊

Teil II

Funktionen, Relationen, Graphen

Kapitel 4

Funktionen

Kapitelinhalt

4.1	Was ist eine Funktion?	81
4.2	Injectivitat, Surjektivitat, Bijektivitat, Monotonie	87
4.3	Verknpfung und Inversion von Funktionen	93

L E R N Z I E L E :

- LZ 4.1** Sie verstehen die Begriffe partielle und totale Funktion.
 - LZ 4.2** Sie können einfache Funktionen auf die Eigenschaften Injektivität, Surjektivität und Bijektivität untersuchen.
 - LZ 4.3** Sie können Verknüpfungen und die Inversion von einfachen Funktionen berechnen.
 - LZ 4.4** Sie können einfache Probleme mittels Funktionen formulieren, darstellen und lösen.
-

4.1 Was ist eine Funktion?

Sie kennen sicher den Begriff einer Funktion schon von früher. Zum Beispiel haben Sie sicher die quadratische Funktion $f(x) := x^2$, welche eine Parabel beschreibt, kennengelernt. Möglicherweise kennen Sie auch die Sinusfunktion $f(x) := \sin(x)$, welche den Grundtyp einer sogenannten periodischen Funktion darstellt, ... und so weiter. In diesem Abschnitt werden wir den Begriff noch einmal definieren und in den darauffolgenden Abschnitten werden wir Funktionen weiter untersuchen. Dort werden dann wahrscheinlich auch für Sie neue Aspekte dazukommen. Im nächsten Kapitel werden wir mit dem Begriff der "Relation" eine sehr wichtige und für die Informatik zentrale Verallgemeinerung von Funktionen studieren.

Also, was wollen wir unter einer "Funktion" genau verstehen?

§4-1 Definition. (Funktion (Abbildung))

Seien X und Y irgendwelche Mengen.

Unter einer **Funktion (oder Abbildung)** f von der Menge X in die Menge Y verstehen wir eine Zuordnungsvorschrift, mit welcher jedem Element $x \in X$ genau ein Element $y \in Y$ zugeordnet wird. Symbolisch schreiben wir dies ausführlich

$$\begin{aligned} f : X &\rightarrow Y \\ x &\mapsto y := f(x) \end{aligned}$$

oder abgekürzt nur $f : X \rightarrow Y$, $x \mapsto y$, $f(x)$... usw., je nachdem, welchen Aspekt wir hervorheben wollen. \diamond

Dies können wir alternativ auch formal so definieren:

§4-2 Definition. (Funktion (Abbildung); Version 2)

Seien X und Y zwei beliebige Mengen.

f heisst **Funktion (Abbildung)** von X nach Y

$$\begin{aligned} \iff f = \{(x, y) \mid x \in X, y \in Y\} \subseteq X \times Y \text{ mit:} \\ \forall x \in X \exists !y \in Y : (x, y) \in f. \end{aligned}$$

Für solche $y \in Y$ schreiben wir dann

$$y := f(x).$$

\diamond

Es gibt Gebiete der Mathematik, wo die Begriffe "Funktion" und "Abbildung" nicht als das exakt Gleiche verwendet werden. Für uns hier sind die Begriffe "Funktion" und "Abbildung" (als mathematischer Begriffe) immer Synonyme. Deshalb sagen wir oft auch, dass zum Beispiel ein bestimmtes " x von f auf y abbgebildet wird". Die einzelnen Teile einer Funktion haben spezielle Namen und diese werden laufend benutzt. Sie sollten sich diese also im Laufe der Zeit gut im Gedächtnis verankert haben.

§4-3 Definition. (Definitionsbereich, Wertebereich, Bildbereich, Urbildbereich, Argument, Funktionswert)

Sei $f : X \rightarrow Y$ eine Funktion.

- (1) X heisst der **Definitionsbereich** der Funktion f , manchmal auch als D_f geschrieben.
- (2) Y ist der **Wertebereich** von f .
- (3) Jedes $x \in X$ nennen wir ein **Argument** von f .
- (4) $f(x)$, also das einem $x \in X$ zugeordnete eindeutige Element $y := f(x) \in Y$, heisst der **Funktionswert** von x unter f oder auch **Funktionswert "f von x"**.
- (5) Die Menge aller $y \in Y$ für welche es ein $x \in X$ gibt, so dass $y = f(x)$ ist, heisst der **Bildbereich** der Funktion f .

Formal definieren wir dies so und benutzen als Symbol $f(X)$:

$$f(X) := \{y \in Y \mid \exists x \in X : y = f(x)\}.$$

(Man spricht manchmal auch nur kurz vom **Bild** der Funktion f .)

- (6) Umgekehrt definieren wir das **Urbild** von $y \in Y$ unter f als die Menge aller Elemente $x \in X$, für welche y ein Funktionswert ist, also formal

$$f^{-1}(y) := \{x \in X \mid y = f(x)\}$$

und allgemeiner für eine ganze Teilmenge $B \subseteq Y$ des Wertebereichs das **Urbild von B unter f** , formal

$$f^{-1}(B) := \{x \in X \mid f(x) \in B\}.$$

◇

Folgende Abbildung visualisiert diese Grundbegriffe:

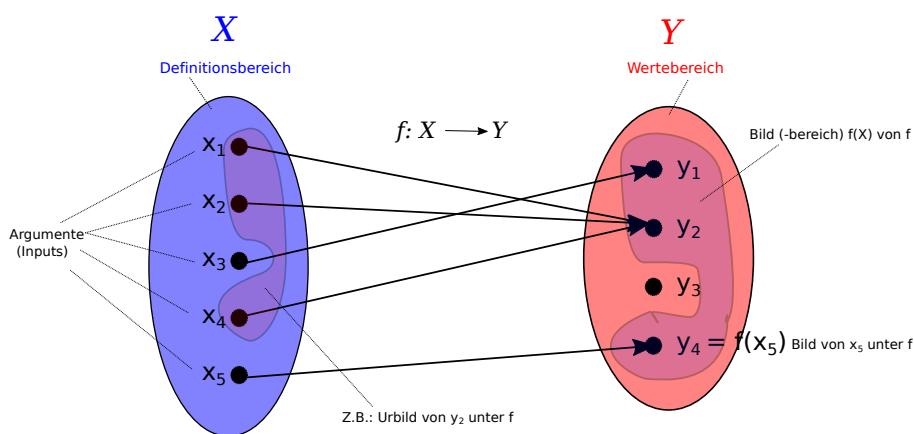


Abbildung 4.1 – Grundbegriffe im Zusammenhang mit Funktionen.

Funktionen bzw. Ausschnitte von ihnen können wir durch ihre Funktionsgraphen visualisieren:

§4-4 Definition. (Funktionsgraphen)

Sei $f : X \rightarrow Y$ eine Funktion.

Die Menge

$$G_f := \{(x, f(x)) \mid x \in X\} \quad (4.1)$$

heisst der **Graph der Funktion f** oder ihr **Funktionsgraphen**. Oft bezeichnet man auch die meist unvollständige zeichnerische ("grafische") Darstellung dieser Menge so. Dann spricht man oft auch von einem **Funktionsplot** oder einfach von einem **Plot**. \diamond

Aufgrund von Gleichung (4.1) sehen wir, dass wir eben Funktionen auch wie in Version 2 §4-2 als Teilmengen eines kartesischen Produkts definieren können. Wenn die Definitionsmenge einer Funktion endlich ist, können wir die entsprechenden geordneten Paare sogar zur Definition aufzählen. Insgesamt haben wir folgende Möglichkeiten eine Funktion zu definieren:

§4-5 Definition. (Definitionsarten für Funktionen)

Wir können im Wesentlichen eine konkrete Funktion $f : X \rightarrow Y$ auf drei Arten definieren:

- (1) In **aufzählender Form** als die Menge G_f der geordneten Paare. Aber dies ist nur möglich, wenn die Definitionsmenge endlich ist.
- (2) Als sogenannter "bipartiter Graph" (siehe später in Kap. 5 und 6). Dies entspricht der Darstellung analog zu Abb. 4.1. Auch dies ist nur möglich, wenn die Definitionsmenge endlich (und nicht zu gross) ist.
- (3) In **beschreibender Form** in Worten.
- (4) In **beschreibender Form** mathematisch-formal.

\diamond

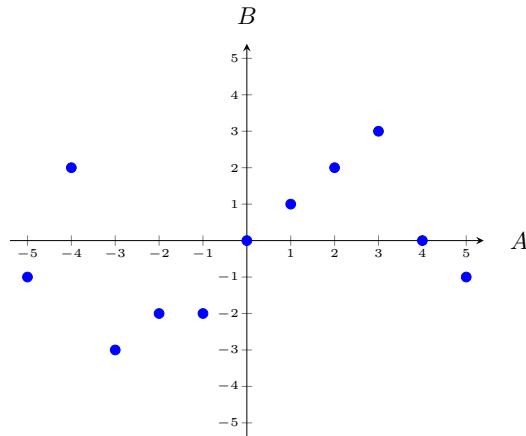
Noch einmal zur Erinnerung: Der zeichnerische Funktionsgraph einer Funktion ("Plot") ist im allgemeinen keine Definitionsart, sondern nur eine unvollständige Darstellung eines Ausschnitts der Funktion. Eine Ausnahme ist der Funktionsgraph auf einem endlichen Definitionsbereich, wenn alle Punkte (als sogenannte "Koordinatenpaare") dargestellt sind.

§4-6 Beispiele. (Typische Funktionen auf verschiedene Arten definiert)**(1) Aufzählende Form bzw. als Funktionsgraphen:**

Die Funktion $f : A \rightarrow B, x \mapsto y := f(x)$ für $A := B := \{-5, -4, \dots, 4, 5\}$ definiert durch die aufzählende Form

$$G_f := \{(-5, -1), (-4, 2), (-3, -3), (-2, -2), (-1, -2), (0, 0), (1, 1), (2, 2), (3, 3), (4, 0), (5, -1)\} \subseteq A \times B$$

mit gezeichneten Funktionsgraphen

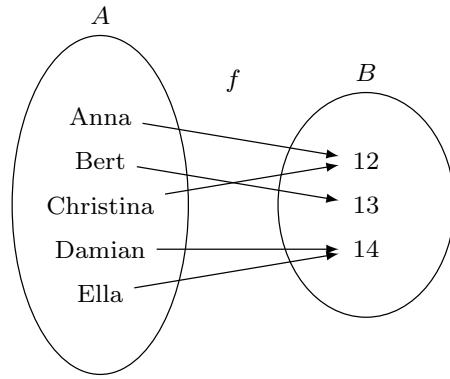


(2) Als "bipartiter Graph":

Natürlich müssen die beteiligten Mengen einer Funktion nicht nur aus Zahlen bestehen. Ein Beispiel ist die Funktion $f : A \rightarrow B$ mit

$$\begin{aligned} A &:= \{\text{Anna, Bert, Christina, Damian, Ella}\} \\ B &:= \{12, 13, 14\} \end{aligned}$$

welche einer Person $x \in A$ ihr Alter $y \in B$ in Jahren zuordnet. Als bipartiter Graph kann diese Funktion zum Beispiel so dargestellt werden:



Dies entspricht der Menge von geordneten Paaren

$$\begin{aligned} G_f &:= \{(\text{Anna}, 12), (\text{Bert}, 13), (\text{Christina}, 12), (\text{Damian}, 14), (\text{Ella}, 14)\} \\ &\subseteq A \times B \end{aligned}$$

(3) Beschreibende Form (in Worten):

"Die Funktion f , welche jeder reellen Zahl ihr Quadrat zuordnet."

Oder

"Die Funktion g , welche jeder reellen Zahl ihren Sinuswert zuordnet."

(4) Beschreibende Form (mathematisch-formal):

Die vorhergehenden beiden Funktionen können dann mathematisch-formal so definiert werden:

$$\begin{aligned} f : \mathbb{R} &\rightarrow \mathbb{R} \\ x &\mapsto \underbrace{f(x)}_{\substack{\text{formale Def.} \\ \text{d. Zuordnung}}} := x^2 \end{aligned}$$

$$g : \mathbb{R} \rightarrow \mathbb{R}$$

$$x \mapsto \underbrace{g(x) := \sin(x)}_{\substack{\text{formale Def.} \\ \text{d. Zuordnung}}}$$

Die Funktionsgraphen bzw. Ausschnitte davon können wir so zeichnen:

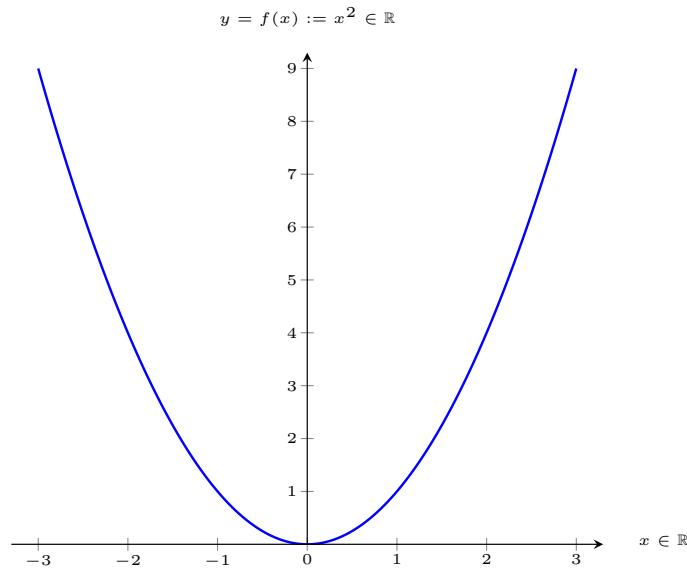


Abbildung 4.2 – Funktionsgraphen (Plot) von $f(x) := x^2$ im Bereich zwischen -3 und 3 des Arguments x .

Anhand des Plots der Funktion f vermuten wir zum Beispiel, und das lässt sich mathematisch bestätigen, dass der Bildbereich der f zwischen 0 und positiv unendlich liegt. Also

$$f(\mathbb{R}) = \mathbb{R}_{\geq 0} \quad (\text{oder } f(\mathbb{R}) = [0, \infty))$$

Für die Funktion $g := \sin$ sieht der Plot folgendermassen aus:

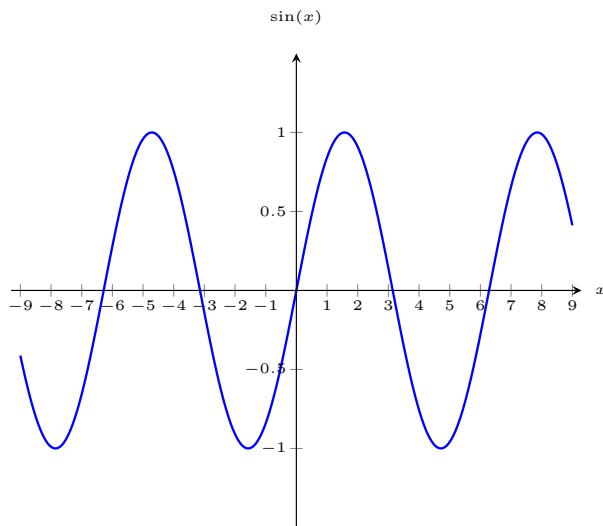


Abbildung 4.3 – Funktionsgraphen (Plot) von $g(x) := \sin(x)$ im Bereich zwischen -9 und 9 des Arguments x .

An diesem Plot erahnen wir zum Beispiel, und das ist tatsächlich so, dass der Bildbereich

der Sinusfunktion zwischen -1 und 1 liegt. Also

$$\sin(\mathbb{R}) = [-1, 1]$$

◇

Wir sehen, nur die letzten beiden Definitionsarten funktionieren auch in allgemeineren Fällen, wo die Menge der Paare $(x, f(x))$ nicht endlich sind.

Der Begri einer Funktion wie wir ihn in §4-1 definiert haben, ist der einer totalen Funktion, weil jedem Element $x \in X$ aus dem Definitionsbereich genau ein Element $y \in Y$ zugeordnet wird. Dies ist die in der Mathematik übliche Definition. In der Informatik benutzt man dazu oft auch noch den Begri einer partiellen Funktion, in welcher nicht jedem $x \in X$ ein Element $y \in Y$ zugeordnet sein muss.

§4-7 Definition. (Totale Funktion, partielle Funktion)

Eine Funktion wie in §4-1 definiert nennen wir auch eine totale Funktion.

Eine partielle Funktion ist hingegen definiert als eine Zuordnungsvorschrift $f : X \rightarrow Y$ mit $D_f \subseteq X$ und

$$\forall x \in D_f \exists !y \in Y : y = f(x)$$

D.h., nicht jedem $x \in X$ ist muss ein Wert aus Y zugeordnet sein und die Funktion kann deshalb für gewisse $x \in X$ undefiniert sein. ◇

Wenn wir nur von einer "Funktion" sprechen, dann meinen wir immer eine totale Funktion. Dies ist in der Mathematik so üblich. Es gibt aber Teilgebiete der Informatik, zum Beispiel die theoretische Informatik, wo man unter "Funktion" stillschweigend eher eine partielle Funktion meint. Es ist also manchmal Vorsicht geboten und man muss unter Umständen überprüfen, was in einem vorliegenden Kontext genau gemeint ist.

§4-8 Beispiel. (Partielle Funktion)

Eine typische partielle Funktion ist die Umkehrfunktion der Multiplikation auf den reellen Zahlen:

$$\begin{aligned} f_{\text{part.}} &: \mathbb{R} \rightarrow \mathbb{R} \\ x &\mapsto f_{\text{part.}}(x) := \frac{1}{x} \end{aligned}$$

Sie ist bekanntlich nur auf $D_f := \mathbb{R} \setminus \{0\}$ definiert, weil wir sonst eine Division durch Null hätten. Da wir in der Mathematik immer *totale* Funktionen meinen, werden wir die Umkehrfunktion der Multiplikation also so schreiben:

$$\begin{aligned} f &: \mathbb{R} \setminus \{0\} \rightarrow \mathbb{R} \\ x &\mapsto f(x) := \frac{1}{x} \end{aligned}$$

◇

§4-9 Beispiel. (Partielle Funktion)

Eine andere typische partielle Funktion ist die Paritätsfunktion, welche jeder ganzen Zahl $x \in \mathbb{Z}$

den Wert 0 zuordnet, wenn sie gerade ist, und den Wert 1, wenn sie ungerade ist. Für alle anderen reellen Zahlen, ist die Paritätsfunktion undefiniert:

$$f_{\text{part.}} : \mathbb{R} \rightarrow \{0, 1\}$$

$$x \mapsto f_{\text{part.}}(x) := \begin{cases} 0, & \text{falls } x \text{ gerade} \\ 1, & \text{falls } x \text{ ungerade} \end{cases}$$

In der Mathematik definieren wir diese wiederum nur auf ihrem Definitionsbereich, also auf \mathbb{Z} :

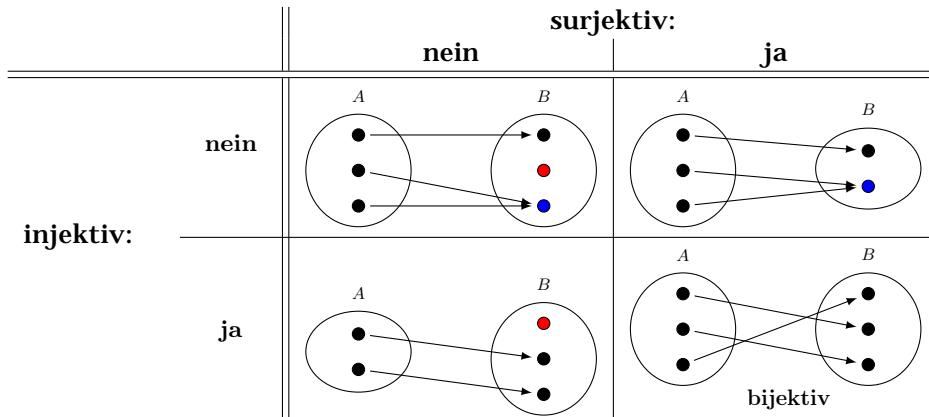
$$f : \mathbb{Z} \rightarrow \{0, 1\}$$

$$x \mapsto f(x) := \begin{cases} 0, & \text{falls } x \text{ gerade} \\ 1, & \text{falls } x \text{ ungerade} \end{cases}$$

◇

4.2 Eigenschaften von Funktionen: Injektivität, Surjektivität und Bijektivität; Monotonie

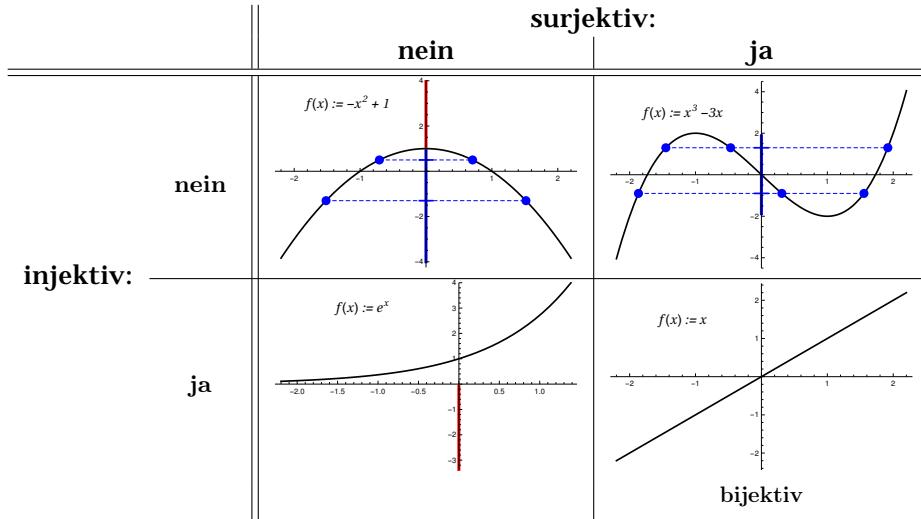
Funktionen können eine Vielzahl von interessanten und nützlichen Eigenschaften haben. Drei sehr fundamentale lernen wir nun in diesem Abschnitt kennen. Dazu überlegen wir uns, wie das zweite Element bei einer Funktion grundsätzlich auftreten kann. Wir betrachten folgende Tabelle, welche die Eigenschaften "injektiv", "surjektiv" und "bijektiv" veranschaulicht, und zwar zuerst einmal für den Fall endlicher Mengen:



Blau: Punkte, wegen welchen nicht-injektiv, Rot: Punkte, wegen welchen nicht-surjektiv

Tabelle 4.1 – Wie können die zweiten Elemente $y \in Y$ in den Paaren (x, y) einer Funktion $f : X \rightarrow Y$ prinzipiell auftreten? Der Fall von Funktionen zwischen *endlichen* Mengen.

Analog können wir uns dies für den Fall überlegen, dass die Funktion nicht auf einer endlichen Menge, sondern zum Beispiel auf einem Intervall der reellen Zahlen definiert ist:



Blau: Punkte, wegen welchen nicht-injektiv, Rot: Punkte, wegen welchen nicht-surjektiv

Tabelle 4.2 – Wie können die zweiten Elemente $y \in Y$ in den Paaren (x, y) einer Funktion $f : X \rightarrow Y$ prinzipiell auftreten? Der Fall von Funktionen zwischen *unendlichen* Mengen (hier: *Teilmenge der reellen Zahlen*).

Wie können wir nun diese Eigenschaften von Funktionen, nämlich

- entweder "injektiv" oder "nicht-injektiv",
- "surjektiv" oder "nicht-surjektiv",
- "bijektiv" oder "nicht-bijektiv"

zu sein, mathematisch exakt definieren? Das geht folgendermassen, wobei es sich lohnt diese Definitionen mit obigen Tabellen zu vergleichen und genau zu studieren:

§4-10 Definition. (Injektivität, Surjektivität, Bijektivität)

- (1) Eine Funktion $f : X \rightarrow Y$ heisst **surjektiv** bzw. ist eine **Surjektion**, falls alle Elemente der Wertemenge Y zur Bildmenge $f(X)$ gehören. Formal:

$$\begin{aligned} f : X \rightarrow Y \text{ heisst surjektiv (Surjektion)} \\ :\iff \forall y \in Y \exists x \in X : f(x) = y \\ :\iff f(X) = Y \end{aligned}$$

- (2) Eine Funktion $f : X \rightarrow Y$ heisst **injektiv** bzw. ist eine **Injectio**n, falls für je zwei verschiedene Argumente $x_1, x_2 \in X$ die dazugehörigen Funktionswerte $f(x_1)$ und $f(x_2)$ unterschiedlich sind. Formal:

$$\begin{aligned} f : X \rightarrow Y \text{ heisst injektiv (Injectio)} \\ :\iff \forall x_1, x_2 \in X : (x_1 \neq x_2 \implies f(x_1) \neq f(x_2)) \\ :\iff \forall x_1, x_2 \in X : (f(x_1) = f(x_2) \implies x_1 = x_2) \end{aligned}$$

- (3) Eine Funktion $f : X \rightarrow Y$ heisst **bijektiv** bzw. ist eine **Bijektion**, falls sie surjektiv und injektiv ist. Formal:

$$\begin{aligned} f : X \rightarrow Y \text{ heisst bijektiv (Bijektion)} \\ :\iff \forall y \in Y \exists !x \in X : y = f(x) \end{aligned}$$



Wir werden gleich diese drei zentralen Begriffe an Beispielen studieren. Es lohnt sich aber, zuerst noch den Begriff der Monotonie für Funktionen einzuführen, weil dieser damit zusammenhängt und uns oft hilft, zu entscheiden, ob eine der drei Eigenschaften surjektiv, injektiv oder bijektiv vorliegt.

§4-11 Definition. (Monotonie von Funktionen)

(Achtung: Nicht die Standarddefinition!)

Seien X und Y beliebige Mengen, worauf eine "Ordnung" und damit $<, \leq, \geq, >$ definiert sind.

- (1) Eine Funktion $f : X \rightarrow Y$ heißt **monoton nicht-fallend**, falls für alle $x_1, x_2 \in X$ mit $x_1 < x_2$ gilt, dass $f(x_1) \leq f(x_2)$ ist.
- (2) Eine Funktion $f : X \rightarrow Y$ heißt **streng monoton steigend** oder besser **nur monoton steigend**, falls für alle $x_1, x_2 \in X$ mit $x_1 < x_2$ gilt, dass $f(x_1) < f(x_2)$ ist.
- (3) Eine Funktion $f : X \rightarrow Y$ heißt **monoton nicht-steigend**, falls für alle $x_1, x_2 \in X$ mit $x_1 < x_2$ gilt, dass $f(x_1) \geq f(x_2)$ ist.
- (4) Eine Funktion $f : X \rightarrow Y$ heißt **streng monoton fallend** oder besser **nur monoton fallend**, falls für alle $x_1, x_2 \in X$ mit $x_1 < x_2$ gilt, dass $f(x_1) > f(x_2)$ ist.



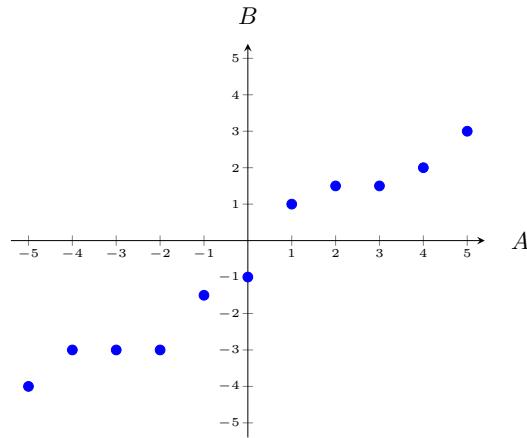
Leider haben sich folgende missverständliche Begriffe sehr verbreitet eingebürgert:

- "monoton nicht-fallend" wird als "monoton steigend" definiert,
- "(streng) monoton steigend" wird als "streng monoton steigend" definiert,
- "monoton nicht-steigend" wird als "monoton fallend" definiert,
- "(streng) monoton fallend" wird als "streng monoton fallend" definiert.

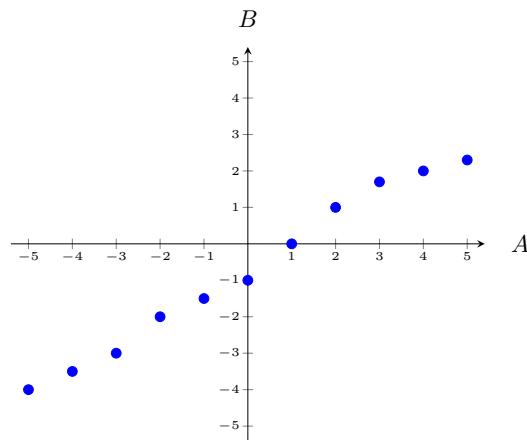
Die Details zu diesem unglücklichen Umstand und warum die verbreiteten Definitionen nicht wirklich sinnvoll sind, können in dieser kleinen Notiz nachgelesen werden: [15].

§4-12 Beispiele. (Monotonie von Funktionen)

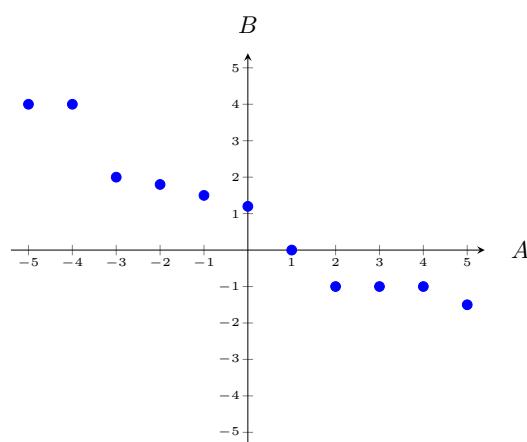
- Eine monoton nicht-fallende Funktion ist zum Beispiel die Funktion



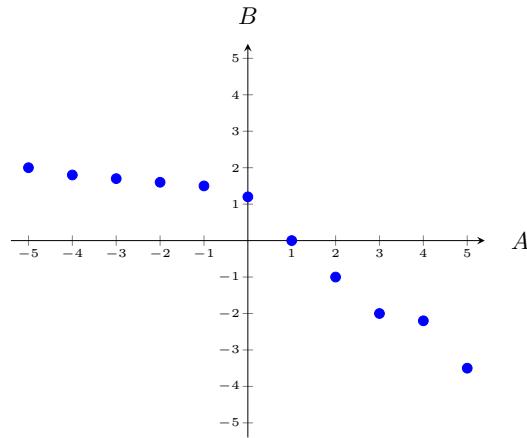
- Eine streng monoton steigende Funktion ist zum Beispiel die Funktion



- Eine monoton nicht-steigende Funktion ist zum Beispiel die Funktion



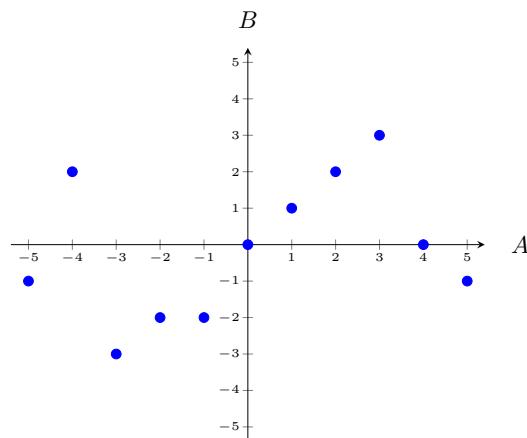
- Eine streng monoton fallende Funktion ist zum Beispiel die Funktion



- Die Funktion $f : A \rightarrow B, x \mapsto y := f(x)$ für $A := B := \{-5, -4, \dots, 4, 5\}$ definiert durch

$$G_f := \{(-5, -1), (-4, 2), (-3, -3), (-2, -2), (-1, -2), (0, 0), (1, 1), (2, 2), (3, 3), (4, 0), (5, -1)\} \subseteq A \times B$$

und mit dem Funktionsgraphen



erfüllt keine der Monotonie-Bedingungen. Sie ist also weder monoton nicht-fallend, noch monoton nicht-steigend, noch streng monoton steigend, noch streng monoton fallend.

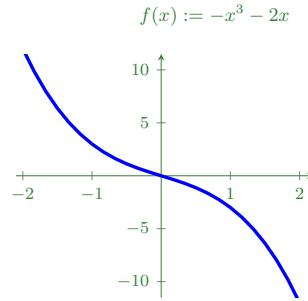
◇

§4-13 Satz. (Monotonie und Injektivität)

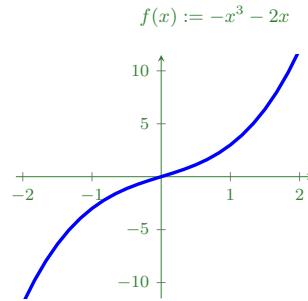
Eine Funktion welche entweder streng monoton steigend oder streng monoton fallend ist, ist injektiv. (Die Umkehrung gilt nicht!) ◇

§4-14 Beispiele. (Strenge Monotonie impliziert Injektivität)

Eine Funktion ist dann injektiv, wenn sie entweder auf dem gesamten Definitionsbereich *streng monoton fallend* ist,



oder wenn sie auf dem gesamten Definitionsbereich *streng monoton steigend* ist:



◊

§4-15 Beispiele. (Surjektivität/Injektivität/Bijektivität von Funktionen)

$A = \{\text{alle Mitarbeiter der Firma HyperData, die nur in einer Zweigstelle arbeiten}\}$

$B = \{\text{alle Zweigstellen der Firma HyperData}\}$

Die Funktion $f : A \rightarrow B$ definiert durch $f(x) = \text{"Zweigstelle, in der } x \text{ arbeitet"}$ ist:

- surjektiv, da es für jedes $y \in B$ (also jede Zweigstelle) ein $x \in A$ (also einen Mitarbeiter) gibt mit $f(x) = y$ (d.h. der Mitarbeiter x arbeitet in der Zweigstelle y).
- nicht injektiv, denn i.a. – d.h., wenn nicht jede Zweigstelle nur einen Mitarbeiter hat – kann für zwei Mitarbeiter $x_1 \neq x_2$ gelten, dass $f(x_1) = f(x_2) = y$ für eine Zweigstelle y .
- nicht bijektiv, da f nicht injektiv ist.

◊

§4-16 Beispiele. (Surjektivität/Injektivität/Bijektivität von Funktionen)

$A = \{\text{alle Studierenden in dieser Klasse}\}$

$B = \{\text{alle möglichen FHNW-E-Mail-Adressen}\}$

Die Funktion $f : A \rightarrow B$ definiert durch $f(x) = \text{"FHNW-E-Mail-Adresse von } x\text{"}$ ist:

- nicht surjektiv, denn es gibt z.B. kein Argument $x \in A$ mit

$$f(x) = \mathbf{andreas.leiser@fhmw.ch}$$

- injektiv, denn je zwei verschiedene Studierende besitzen verschiedene E-Mail-Adressen.
- nicht bijektiv, weil f nicht surjektiv ist.

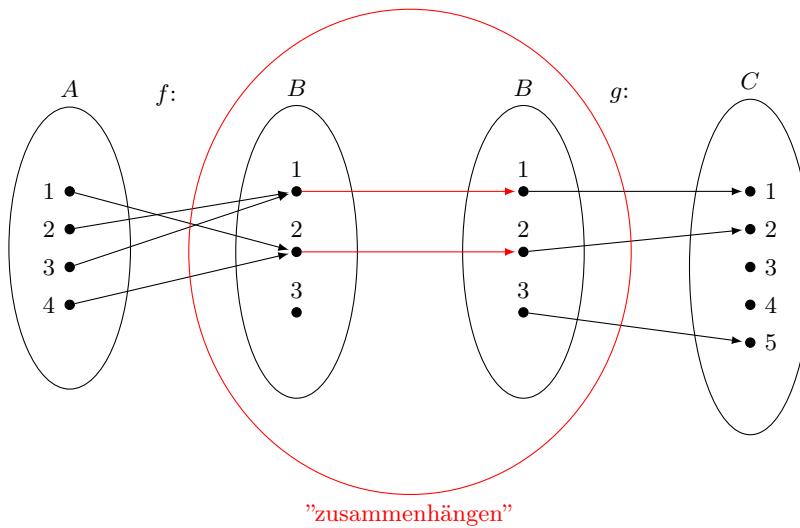
Beachten Sie: $f(A)$ ist in diesem Beispiel die Menge aller FHNW-E-Mail-Adressen dieser Klasse.
Die Funktion

$$f : A \rightarrow f(A)$$

– d.h., die Wertemenge auf den Bildbereich eingeschränkt – ist dann surjektiv und somit auch bijektiv. \diamond

4.3 Fundamentale Operationen auf Funktionen: Verknüpfung und Inversion

Die Verknüpfung von Funktionen kennen Sie wahrscheinlich schon:



Frage: Ist die Verknüpfung $g \circ f$ von zwei Funktionen immer auch eine Funktion?

Ja!

§4-17 Theorem.

Seien $f : A \rightarrow B$ und $g : B \rightarrow C$ zwei (totale) Funktionen.
Dann ist die Verknüpfung $g \circ f$ eine Funktion von A nach C , also

$$g \circ f : A \rightarrow C.$$

Für die Funktion $g \circ f$ gilt

$$(g \circ f)(x) = g(f(x)).$$

(Die Funktion f wird also zuerst auf x ausgeführt, und auf den Wert $f(x)$ wird die Funktion g ausgeführt.) \diamond

Beweis. (nicht Prüfungssto)

Da f eine (totale) Funktion ist, gilt nach Definition, dass es für alle $x \in A$ genau ein $y \in B$ gibt, so dass $y = f(x)$ ist.

Und da g ebenfalls (totale) Funktion ist, gilt wieder nach Definition, dass es für alle $y \in B$ genau ein $z \in C$ gibt, so dass $z = g(y)$ ist.

Das heisst aber gerade, dass es für alle $x \in A$ genau ein $z \in C$ gibt, – nämlich $z = g(y) = g(f(x))$, so dass $z = (g \circ f)(x)$ ist.

Also ist gemäss Definition einer Funktion auch $g \circ f$ eine Funktion. □

§4-18 Beispiel.

Seien $f : \mathbb{R} \rightarrow \mathbb{R}$ und $g : \mathbb{R} \rightarrow \mathbb{R}$ durch

$$\begin{aligned}f(x) &= 3x + 1 \\g(x) &= x^2 - 3\end{aligned}$$

gegeben. Wir bestimmen $g \circ f$ und $f \circ g$.

Lösung:

$$\begin{aligned}(g \circ f)(x) &= g(f(x)) = g(3x + 1) = (3x + 1)^2 - 3 = 9x^2 + 6x + 1 - 3 \\&= 9x^2 + 6x - 2. \\(f \circ g)(x) &= f(g(x)) = f(x^2 - 3) = 3(x^2 - 3) + 1 = 3x^2 - 9 + 1 \\&= 3x^2 - 8.\end{aligned}$$

Wir sehen: $g \circ f \neq f \circ g$ und merken uns also: ◊

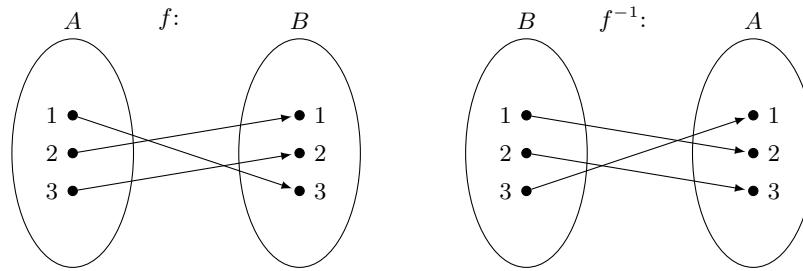
§4-19 Theorem.

Die Verknüpfung von Funktionen ist nicht kommutativ. D.h., im Allgemeinen gilt:

$$g \circ f \neq f \circ g.$$

◊

Auch die Inverse einer Funktion ist Ihnen eventuell schon bekannt:



Frage: Ist das "Inverse" (die Umkehrung der Pfeile) einer beliebigen Funktion immer auch eine Funktion?

Nur, wenn die Funktion eine bijektive Funktion ist! [E.14]

Wieso? Folgende Überlegungen zeigen dies:

- Wenn f nicht surjektiv ist, dann ist die Umkehrung der Pfeile keine (totale) Funktion mehr! (Es gibt dann einen Knoten, von dem kein Pfeil ausgeht.)
~ Schauen Sie sich die Def. von Surjektivität und einer totalen Funktion an!
- Wenn f nicht injektiv ist, dann ist die Umkehrung der Pfeile keine Funktion, da es mind. einen Knoten gibt, von dem mind. zwei Pfeile ausgehen!
~ Schauen Sie sich die Def. von Injektivität und einer Funktion an!
- Also muss f surjektiv und injektiv sein, d.h., eine Bijektion sein.

§4-20 Satz und Definition. (Inverse einer Funktion)

Sei $f : A \rightarrow B$ eine bijektive Funktion.

Dann ist die Funktion

$$f^{-1} : B \rightarrow A$$

mit $\forall b \in B$

$$f^{-1}(b) = a \iff f(a) = b :$$

eine Funktion. Wir nennen diese Funktion die **inverse Funktion** oder **Umkehrfunktion (Inverse)** von f . \diamond

§4-21 Beispiel.

Wir berechnen die Umkehrfunktion von $f : \mathbb{R} \rightarrow \mathbb{R}$ definiert durch $f(x) = 3x + 1$.

Lösung:

Wir müssen die definierende Gleichung der Funktion f , d.h.,

$$y = f(x) = 3x + 1$$

nach x auflösen. Also

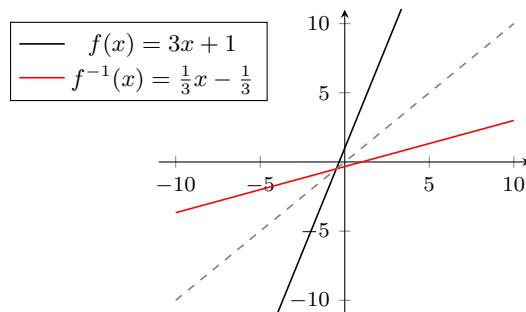
$$\begin{aligned} y = 3x + 1 &\iff y - 1 = 3x \\ &\iff \frac{1}{3}(y - 1) = x. \end{aligned}$$

Vertauschung der Variablen x und y liefert dann die Umkehrfunktion

$$y = f^{-1}(x) = \frac{1}{3}(x - 1) = \frac{x - 1}{3}.$$

\diamond

Wenn wir den gleichen Massstab für beide Achsen verwenden, dann ergibt sich die inverse Funktion in der Darstellung im Koordinatensystem durch Spiegelung an der ersten Winkelhalbierenden:



Kapitel 5

Relationen

Kapitelinhalt

5.1	Was ist eine Relation?	99
5.2	Homogene und heterogene binäre Relationen	101
5.3	Reflexivität, Symmetrie, Antisymmetrie, Transitivität	102
5.4	Äquivalenzrelationen	104
5.5	Ordnungsrelationen	108
5.6	Operationen zwischen und mit Relationen	114
5.6.1	Verknüpfung zweier binärer Relationen	114
5.6.2	Inversion einer binären Relationen	117

L E R N Z I E L E :

- LZ 5.1** Sie erkennen Relationen als eine Verallgemeinerung des Begriffs einer Funktion.
 - LZ 5.2** Sie verstehen die Grundbegriffe, können die Relationen darstellen und können diese auf Reflexivität, Symmetrie, Transitivität und Antisymmetrie untersuchen.
 - LZ 5.3** Sie können die Komposition zweier Relationen und die Inversion einer Relation bestimmen.
 - LZ 5.4** Sie kennen Äquivalenzrelationen und Ordnungsrelationen, sowie den Zusammenhang zwischen Äquivalenzrelationen und Partitionen.
-

5.1 Was ist eine Relation?

Wir haben bereits gesehen, dass wir Funktionen (Abbildungen) als spezielle Teilmengen eines kartesischen Produkts $f := G_f \subseteq X \times Y$ ansehen können. Dabei haben wir X den Definitionsbereich und Y den Wertebereich der Funktion f genannt. Das spezielle bei diesen Teilmengen war, dass jedem Element $x \in X$ genau ein Element $y \in Y$ zugeordnet wurde. Diese Einschränkung wollen wir nun nicht mehr machen und ganz allgemein beliebige Teilmengen $R \subseteq A_1 \times A_2 \times \dots \times A_n$ von irgendwelchen kartesischen Produkten von irgendwelchen Mengen A_1, A_2, \dots, A_n betrachten. Diese Verallgemeinerung sind genau die Relationen. Dabei veranschaulicht das Wort "Relation" sehr schön die Bedeutung: "Relation" heisst ja auch "Beziehung": Es werden mit Relationen Elemente verschiedener Mengen bzgl. einer oder mehrerer interessierenden Eigenschaften miteinander in Beziehung gesetzt. Das ist ein sehr mächtiges Tool um viele Dinge zu modellieren und Anwendungsprobleme zu lösen. Wir definieren also:

§5-1 Definition. (n -stellige Relation, n -Tupel)

Sei $n \in \mathbb{N}$ eine beliebige natürliche Zahl grösser 1.

Eine n -stellige Relation R auf den (oder zwischen den) Mengen A_1, A_2, \dots, A_n ist eine Teilmenge

$$R \subseteq A_1 \times A_2 \times \dots \times A_n$$

des kartesischen Produkts der A_1, A_2, \dots, A_n . Die Elemente einer Relation sind sogenannte (geordnete) n -Tupel:

$$(x_1, x_2, \dots, x_n) \in R$$

◊

Einen speziellen Namen erhalten 2-stellige Relationen, da sie der am einfachsten zu studierende Fall sind:

§5-2 Definition. (Binäre Relation)

Im Fall von $n := 2$ sprechen wir auch von einer binären Relation $R \subseteq A_1 \times A_2$ zwischen A_1 und A_2 . Für ihre Elemente, die geordneten Paare $(x, y) \in R$ schreiben wir häufig auch xRy .

◊

Da es sich hier um eine Einführung handelt, werden wir uns fast ausschliesslich mit binären Relationen beschäftigen und diese untersuchen. Schauen wir uns zwei erste typische Beispiele von Relationen an:

§5-3 Beispiel. (Relation zwischen Ländern und Kontinenten)

Für die (willkürlichen) Mengen

$$A := \{\text{Andorra, Belgien, Brasilien, Chile, Indien, Italien, Marokko, Schweiz, \\ Türkei, USA}\}$$

von Ländern und

$$B := \{\text{Amerika, Asien, Europa}\}$$

von Kontinenten, ist z.B.

$$\begin{aligned} R &= \{(Andorra, Europa), (Belgien, Europa), (Brasilien, Amerika), \\ &\quad (\text{Chile}, \text{Amerika}), (\text{Indien}, \text{Asien}), (\text{Italien}, \text{Europa}), (\text{Schweiz}, \text{Europa}), \\ &\quad (\text{Türkei}, \text{Europa}), (\text{Türkei}, \text{Asien}), (\text{USA}, \text{Amerika})\} \\ &\subseteq A \times B \end{aligned}$$

die Relation "x ist Land des Kontinents y". D.h., $(x, y) \in R$ bedeutet "x ist Land des Kontinents y".

Wir können uns in diesem Beispiel fragen, ob die Teilmenge R nicht auch eine Funktion ist? Wenn wir genau hinschauen, dann sehen wir, dass dies *nicht* der Fall ist. Denn die Türkei gehört bekanntlich sowohl zum Kontinent Europa, wie auch zu Asien. Das heisst, hier sind einem Element aus der ersten Menge A *zwei* Elemente – also nicht "genau ein Element", wie in der Definition einer Funktion gefordert – zugeordnet. \diamond

§5-4 Beispiel. (Relationen aus Logik und Mengenlehre)

Relationen haben wir in der Logik und in der Mengenlehre zum Beispiel schon mit "=". "∈" oder " \leq " kennengelernt:

Sei x allgemein eine reelle Zahl. $x \in \mathbb{Q}$ ist dann die Relation

$$\begin{aligned} R &:= \{(x, q) \mid x \in \mathbb{R}, q \in \mathbb{Q} : x = q\} \\ &= \{(x, \frac{m}{n}) \mid x \in \mathbb{R}, \exists m \in \mathbb{Z}, n \in \mathbb{N} : x = \frac{m}{n}\} \\ &\subseteq \mathbb{R} \times \mathbb{Q}. \end{aligned}$$

D.h., $x \in \mathbb{Q} \iff \exists q \in \mathbb{Q} : xRq$.

Als konkretes Beispiel: $0.\bar{3}R\frac{1}{3}$ und deshalb ist $0.\bar{3}$ eine rationale Zahl. \diamond

Speziell in der Informatik sind Relationen an vielen Orten von zentraler Bedeutung. Zum Beispiel

- als logisch-mathematische Formulierungen in Programmen,
- als abstrakte und präzise Problemformulierungen im weitesten Sinn (zum Beispiel zur Formulierung von Anwendungsproblemen),
- bei "relationalen Datenbanken" und damit ganz allgemein in der "Informationsverarbeitung",
- usw.

Ähnlich wie bei den Funktionen können wir auch konkrete Relationen auf verschiedene Arten definieren bzw. darstellen:

§5-5 Definition. (Definitionsarten von Relationen)

- (1) In der **aufzählenden Form** wird die Relation als Menge geordneter Paare angegeben bzw. definiert. Dies ist wiederum nur möglich, wenn die Menge der Elemente der Relation endlich ist.
- (2) In der **graphischen Darstellung** als bipartiter Graph (wir werden wie schon erwähnt noch genauer sehen, was Graphen bzw. bipartite Graphen sind). Auch hier gilt, dass sie nur als Definition gebraucht werden kann, wenn die Menge der Paare endlich, und meist auch einigermassen klein, ist.

- (3) In der beschreibenden Form wird die Relation entweder in Worten oder mathematisch-formal definiert. Da es sich um eine Teilmenge handelt, benutzen wir dabei bei der mathematisch-formalen Form die kennengelernten Schreibweisen der Mengenlehre und der Logik.

◊

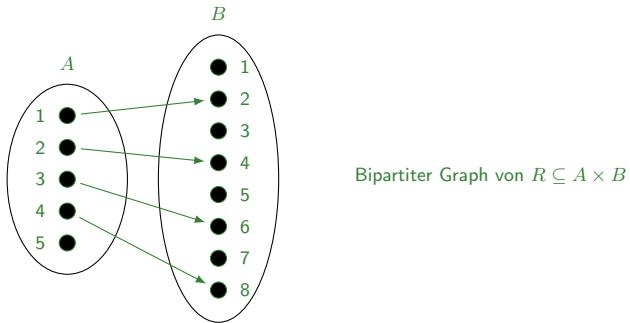
§5-6 Beispiel. (Definitionsarten von Relationen)

- (1) Sei $A := \{1, 2, \dots, 5\}$, $B := \{1, 2, \dots, 8\}$. Dann ist mit

$$R := \{(1, 2), (2, 4), (3, 6), (4, 8)\} \subseteq A \times B$$

eine Relation in aufzählender Form gegeben. Diese ist im Übrigen sogar eine Funktion. Bitte überlegen Sie sich wieso.

- (2) A, B, R wie oben.



- (3) A und B wie oben.

$$R := "y \text{ ist doppelt so gross wie } x."$$

ist dieselbe Relation zwischen A und B in Worten definiert und

$$R := \{(x, y) \in A \times B \mid y = 2x\}$$

ist eine mathematisch-formale Definition derselben Relation.

◊

5.2 Homogene und heterogene binäre Relationen

§5-7 Definition. (Homogenen und heterogenen binären Relationen)

Eine binäre Relation $R \subseteq A_1 \times A_2$ heisst **homogen**, gdw. $A_1 = A_2$. Ansonsten heisst sie **heterogen**.

Eine binäre homogene Relation $R \subseteq A \times A$ nennen wir dann auch eine (**binäre Relation auf A**)

◊

Bei homogenen binären Relationen vereinfacht sich der darstellende bipartite Graph zu einem sogenannt gerichteten Graphen (diese Struktur werden wir später genau definieren). Was heisst dies? Da die beiden beteiligten Mengen identisch sind, müssen wir diese Mengen im Graphen nicht mehr getrennt zeichnen (wie im bipartiten Graphen). Sondern es reicht, wenn wir die Punkte der Menge A einmal zeichnen und dann die Pfeile welche die geordneten Paare darstellen, direkt eintragen. Genau dies führt zu einem Gebilde, was wir später allgemeiner als gerichteten Graphen kennenlernen werden. Im homogenen Fall sind also die folgenden Darstellungen äquivalent und wir benutzen meist diejenige des gerichteten Graphen auf der rechten Seite, da sie einfacher ist. Dies ist im folgenden Beispiel dargestellt:

§5-8 Beispiel. (Äquivalente Darstellung als "gerichteter Graph")

Sei $A := \{1, 2, 3, 4, 5, 6\}$, $R := \{(1, 3), (2, 1), (3, 2), (5, 2), (5, 3), (5, 5), (6, 5)\}$.

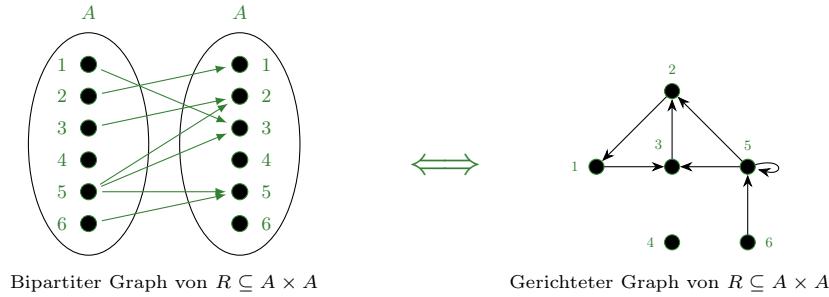


Abbildung 5.1 – Homogene binäre Relation äquivalent als "gerichteter Graph" dargestellt.

◇

5.3 Reflexivität, Symmetrie, Antisymmetrie und Transitivität einer homogenen binären Relation $R \subseteq A \times A$

§5-9 Definition. (Typische Eigenschaften (und Klassen) von homogenen binären Relationen)

Eine homogene binäre Relation $R \subseteq A \times A$ heisst

- (i) **reflexiv**, gdw. $\forall x \in A : xRx$,
- (ii) **symmetrisch**, gdw. $\forall x, y \in A : (xRy \Rightarrow yRx)$,
- (iii) **antisymmetrisch**, gdw. $\forall x, y \in A : (x \neq y \wedge xRy \Rightarrow (y, x) \notin R)$,
- (iv) **transitiv**, gdw. $\forall x, y, z \in A : (xRy \wedge yRz \Rightarrow xRz)$.

◊

§5-10 Theorem. (Eigenschaften der vier Vergleichsoperatoren $\circ \in \{<, \leq, \neq, =\}$)

Seien die Relationen $R_\circ := \{(x, y) \in \mathbb{R}^2 \mid x \circ y\} \subseteq \mathbb{R}^2$ gegeben, wobei $\circ \in \{<, \leq, \neq, =\}$. Dann gilt:

Eigenschaft:	$<$	\leq	\neq	$=$
reflexiv	nein	ja	nein	ja
symmetrisch	nein	nein	ja	ja
antisymmetrisch	ja	ja	nein	ja
transitiv	ja	ja	nein	ja

◊

Beweis.Seien x, y, z reelle Zahlen.**1.Spalte “ $<$ ”-Relation:**

- Für jede reelle Zahl x ist $\neg(x < x)$ (da ja $x = x$ trivialerweise gilt), also ist die Relation nicht reflexiv.
- Wenn $x < y$ ist (z.B. $3 < 5$), dann ist sicher nicht $y < x$ (im Bsp. $\neg(5 < 3)$), also nicht symmetrisch.
- Letzteres ist sogar immer der Fall (insbesondere, wenn $x \neq y$ zwei unterschiedliche reelle Zahlen sind), also anti-symmetrisch.
- Die Transitivität gilt: wenn $(x < y) \wedge (y < z)$ ist, dann folgt, dass $x < z$ ist (z.B. aus $(3 < 5) \wedge (5 < 6)$ folgt $3 < 6$).

2.Spalte “ \leq ”-Relation:

- Für jedes x gilt trivialerweise $x \leq x$ (z.B. $5 \leq 5$), also reflexiv.
- Mit analogem Argument wie bei $<$: nicht symmetrisch, anti-symmetrisch und transitiv.

3.Spalte “ \neq ”-Relation:

- Gleichheit reeller Zahlen ist reflexiv, also ist es die Nichtgleichheit nicht.
- Symmetrisch: Ist $x \neq y$, dann natürlich auch $y \neq x$.
- Nicht anti-symmetrisch, konkretes Gegenbeispiel: $3 \neq 5$, aber daraus folgt nicht, dass $\neg(5 \neq 3) \equiv (5 = 3)$ gilt.
- Nicht transitiv, Gegenbeispiel: Gilt z.B. $(3 \neq 5) \wedge (5 \neq 3)$ dann folgt nicht, dass $3 \neq 3$ gilt.

4.Spalte “ $=$ ”-Relation:

- Die Gleichheitsrelation ist natürlich reflexiv, dies per Definition.
- Sie ist auch symmetrisch: Aus $x = y$ folgt immer $y = x$.
- Wenn $x \neq y$ ist, dann folgt aus $x = y$ (eine falsche Prämisse!), dass $y \neq x$ gilt, also auch anti-symmetrisch! (Erinnerung: aus einer falschen Prämisse kann jede Folgerung gezogen werden!)
- Aus $(x = y) \wedge (y = z)$ folgt immer $x = z$, also transitiv.

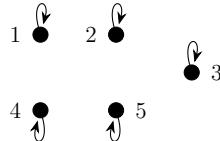
□

Insbesondere merken wir uns anhand der “ $=$ ”-Relation (4.Spalte):**§5-11 Satz.**

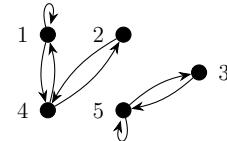
Eine Relation kann symmetrisch und anti-symmetrisch sein!

◊

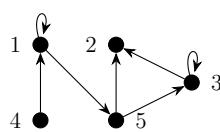
Wieso das? Das logische Gegenteil von symmetrisch ist eben die Eigenschaft nicht-symmetrisch zu sein. Und das ist was anderes als anti-symmetrisch zu sein! Dies, und die Intuition für alle vier Eigenschaften können wir uns ganz gut an folgender Skizze überlegen:



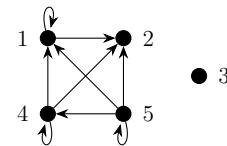
Reflexivität: “Jeder Knoten hat eine Schleife.”



Symmetrie: “Für jeden Pfeil gibt es einen Pfeil in Gegenrichtung.” (Beachte: Schleifen sind gleichzeitig Pfeil und Pfeil in Gegenrichtung (!))



Antisymmetrie: “Für jeden Pfeil, der nicht Schleife ist, gibt es keinen Pfeil in Gegenrichtung.”



Transitivität: “Jeder Pfad entlang zweier Pfeile (mit gleichem Richtungssinn) hat einen abkürzenden Pfeil vom Anfangs- zum Endknoten des Pfades.”

Abbildung 5.2 – Visualisierung der vier Grundeigenschaften homogener binärer Relationen.

5.4 Äquivalenzrelationen

Wie kann man Elemente einer beliebigen Menge (d.h. von verschiedensten mathematischen Objekten, denken Sie hier nicht nur an Zahlen!) “vom gleichen Typ” zusammenfassen? D.h., wir haben eine bestimmte Eigenschaft vorgegeben und wollen wissen, welche Elemente der Menge bzgl. dieser Eigenschaft gleich sind (d.h., die betreute Eigenschaft haben).

§5-12 Beispiel. (Gleichheit von Elementen einer Menge bzgl. einer Eigenschaft)

Sei

$$M := \{\text{Ananas, Banane, Erdbeere, Melone, Apfel, Litschi, Birne, Aprikose, Mango, Mandarine}\}.$$

Wir betrachten folgende Eigenschaft der Elemente von M :

“Frucht x hat denselben Anfangsbuchstaben wie Frucht y .”

Dann ist bzgl. dieser Eigenschaft z.B.

- Ananas und Aprikose gleich,
- Melone, Mango und Mandarine gleich,
- Ananas und Litschi nicht gleich,
- Banane und Erdbeere nicht gleich.

(Anm.: Natürlich ist jede Frucht bzgl. dieser Eigenschaft mit sich selbst gleich.) ◊

Diese “Gleichheit bzgl. einer bestimmten, uns interessierenden Eigenschaft” lässt sich elegant mittels dem Begriff einer Relation und zwar einer speziellen Art von Relation formulieren:

§5-13 Definition. (Äquivalenzrelation; äquivalente Objekte (Elemente))

Eine (binäre) Relation $R \subseteq A \times A$ auf A heisst **Äquivalenzrelation**, genau dann wenn R

- (i) reflexiv,
- (ii) symmetrisch,
- (iii) transitiv

ist. Eine Äquivalenzrelation wird oft allgemein mit \sim symbolisiert.

Zwei Objekte $x, y \in A$ mit xRy heissen dann **äquivalent zueinander** und wir schreiben $x \sim y$ (oder auch $\sim(x, y)$, wenn $(x, y) \in \sim$ ($:= R$) ist). \diamond

Äquivalente Elemente einer Menge werden dann bzgl. der diese Äquivalenzklasse definierenden Eigenschaft als gleich (“ununterscheidbar”) – d.h., eben als äquivalent – angesehen. Die Zusammenfassung der Elemente gleichen Typs nennen wir auch eine **Klasse**, da wir bzgl. einer Äquivalenzrelation eine ganze Menge “klassifizieren” können:

§5-14 Definition. (Äquivalenzklasse; Repräsentant)

Sei \sim eine Äquivalenzrelation auf der Menge A . Für jedes Element $x \in A$ heisst dann

$$[x]_{\sim} := \{y \in A \mid x \sim y\}$$

die **Äquivalenzklasse von x bzgl. \sim** . Jedes Element $y \in [x]_{\sim}$ heisst dann ein **Repräsentant** oder **Vertreter** der Äquivalenzklasse. \diamond

§5-15 Beispiel.

In unserem Beispiel von vorhin mit

$$M := \{\text{Ananas, Banane, Erdbeere, Melone, Apfel, Litschi, Birne, Aprikose, Mango, Mandarine}\}.$$

haben wir also folgende Äquivalenzklassen bzgl. der dort definierten Äquivalenzrelation \sim :

$$\begin{aligned} [\text{Ananas}]_{\sim} &= \{\text{Ananas, Apfel, Aprikose}\}, & [\text{Banane}]_{\sim} &= \{\text{Banane, Birne}\}, \\ [\text{Erdbeere}]_{\sim} &= \{\text{Erdbeere}\}, & [\text{Litschi}]_{\sim} &= \{\text{Litschi}\}, \\ [\text{Mandarine}]_{\sim} &= \{\text{Mandarine, Mango, Melone}\}. \end{aligned}$$

\diamond

Wenn die Äquivalenzrelation aus dem Kontext klar, schreiben wir auch nur $[x]$ für eine Äquivalenzklasse. Üblich ist auch: x / \sim . Das Bilden von Äquivalenzklassen bzgl. einer Eigenschaft ist in der Mathematik ein sehr wichtiges Prinzip: Es erlaubt zum Beispiel eine “ökonomische” (d.h., zeitsparendere, einfachere) Untersuchung einer Menge. Denn wir müssen dann bei passend gewählter Äquivalenzklasse bzw. Eigenschaft nur je die einzelnen Äquivalenzklassen untersuchen bzw. je einen Vertreter (Repräsentanten) daraus!

§5-16 Theorem. (Fundamentale Eigenschaften von Äquivalenzklassen)

Für jede Äquivalenzrelation \sim auf einer beliebigen Menge A gilt:

- (1) $y \in [x]_{\sim} \Rightarrow [y]_{\sim} = [x]_{\sim}$.
- (2) Je zwei verschiedene Äquivalenzklassen $[x]_{\sim} \neq [y]_{\sim}$ sind disjunkt.
- (3) Die Vereinigung aller Äquivalenzklassen ist gleich der Menge A .

◊

Damit gilt auch unmittelbar der folgende Korollar: [E.15]

§5-17 Korollar. (Äquivalenzrelationen und Partitionen)

Es gilt also:

- (1) Eine Äquivalenzrelation auf einer Menge A definiert durch ihre Äquivalenzklassen "in natürlicher Weise" eine Partition auf der Menge. (Wir sagen auch, eine Äquivalenzrelation "induziert" eine Partition auf ihrer Menge.)
- (2) Jede Partition einer Menge A induziert umgekehrt eindeutig eine Äquivalenzrelation auf A , d.h. eine Relation $\sim \subseteq A \times A$.

◊

Diesen wichtigen Zusammenhang wollen wir noch genauer an einem Beispiel studieren:

§5-18 Beispiel. (Äquivalenzrelation und Partition)

- (1) Zuerst betrachten wir die Richtung, dass jede Äquivalenzrelation eine Partition definiert:

Sei wieder

$$M := \{\text{Ananas, Banane, Erdbeere, Melone, Apfel, Litschi, Birne, Aprikose, Mango, Mandarine}\}.$$

mit der Äquivalenzrelation

$$x \sim y \iff \text{"Frucht } x \text{ hat denselben Anfangsbuchstaben wie } y."$$

auf M . Dann haben wir wie gesehen folgende Äquivalenzklassen:

$$\begin{aligned} [\text{Ananas}]_{\sim} &= \{\text{Ananas, Apfel, Aprikose}\}, \\ [\text{Banane}]_{\sim} &= \{\text{Banane, Birne}\}, \\ [\text{Erdbeere}]_{\sim} &= \{\text{Erdbeere}\}, \\ [\text{Litschi}]_{\sim} &= \{\text{Litschi}\}, \\ [\text{Mandarine}]_{\sim} &= \{\text{Mandarine, Mango, Melone}\}. \end{aligned}$$

Diese fünf Äquivalenzklassen bilden nun offensichtlich eine Partition von M : Denn ihre Vereinigung ist M selbst und sie sind gegenseitig disjunkt. Und genau so haben wir eine Partition einer Menge definiert!

- (2) Nun zum umgekehrten Fall, dass eine Partition immer auch in natürlicher Weise eine Äquivalenzrelation definiert:

Sei

$$M := \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

und die einfache Partition von M gegeben durch $\{A, B\}$ mit

$$\begin{aligned} A &:= \{x \in M \mid x \text{ gerade}\} \\ B &:= \{x \in M \mid x \text{ ungerade}\}. \end{aligned}$$

Dann ist $A \cup B = M$ und $A \cap B = \emptyset$, also klar eine Partition. Die "zugehörige" Äquivalenzrelation auf M ist jetzt definiert durch

$$x \sim y \iff x \text{ und } y \text{ haben dieselbe Parität.} \quad (5.1)$$

("Dieselbe Parität" meint, sie sind beide entweder gerade oder ungerade.) Die Äquivalenzklassen symbolisiert durch die Repräsentanten 1 und 2 (man könnte aber durchaus auch andere verwenden), sind damit:

$$\begin{aligned} [1]_{\sim} &= \{1, 3, 5, 7, 9\} && (\text{Die ungeraden Zahlen in } M.) \\ [2]_{\sim} &= \{2, 4, 6, 8\} && (\text{Die geraden Zahlen in } M.) \end{aligned}$$

Ist dies tatsächlich eine Äquivalenzrelation? Testen wir dies kurz. Die Elemente von $\sim \subseteq M \times M$ sind aufgezählt aufgrund der Definition (5.1) die folgenden:

$$\begin{aligned} \sim = \{(1, 1), (1, 3), (1, 5), (1, 7), (1, 9), (2, 2), (2, 4), (2, 6), (2, 8), \\ (3, 1), (3, 3), (3, 5), (3, 7), (3, 9), (4, 2), (4, 4), (4, 6), (4, 8) \\ (5, 1), (5, 3), (5, 5), (5, 7), (5, 9), (6, 2), (6, 4), (6, 6), (6, 8), \\ (7, 1), (7, 3), (7, 5), (7, 7), (7, 9), (8, 2), (8, 4), (8, 6), (8, 8), \\ (9, 1), (9, 3), (9, 5), (9, 7), (9, 9)\} \end{aligned}$$

Wir erkennen:

- Die Relation ist reflexiv, denn

$$\{(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9)\}$$

sind alle in der Relation enthalten.

- Die Relation ist symmetrisch, denn zu jedem $(x, y) \in \sim$ ist auch das Element $(y, x) \in \sim$ enthalten. (Überzeugen Sie sich davon, indem Sie das für ein paar Elemente überprüfen!)
- Die Relation ist transitiv, denn zu je zwei Elementen in \sim mit (x, y) und (y, z) ist immer auch das Element (x, z) in der Relation. (Auch dies sollten Sie anhand von ein Paar konkreten Fällen überprüfen.)

Damit ist die Relation \sim , welche wir aus der Partitionierung von M erhalten haben, nach Definition §5-13 tatsächlich eine Äquivalenzrelation.

◇

§5-19 Beispiel. (Laufwerk-Partitionierung als Äquivalenzrelation)

Die Partitionierung eines Laufwerks ist eine Äquivalenzrelation in welcher zwei Speichereinheiten genau dann zueinander äquivalent sind, wenn sie derselben Partition angehören.

◇

5.5 Ordnungsrelationen

Wie können wir durch eine Relation auf einer Menge, diese Menge bzgl. einer bestimmten Eigenschaft "ordnen", d.h., eine Art Hierarchie unter den Elementen einführen? Allgemein muss die Relation dazu mindestens folgende Eigenschaften haben, was wir dann eine Halbordnung nennen:

§5-20 Definition. (Halbordnung; halbgeordnete Menge)

Eine Relation R auf einer Menge A heisst **Halbordnung (Halbordnungsrelation)**, genau dann wenn R

- (i) reflexiv,
- (ii) antisymmetrisch,
- (iii) transitiv

ist. Eine Halbordnungsrelation symbolisieren wir auch mit \preceq und das Paar (A, \preceq) nennen wir auch **halbgeordnete Menge**. \diamond

Die Halbordnungsrelation unterscheidet sich also von einer Äquivalenzrelation genau durch Bedingung (ii): Sie ist nicht notwendigerweise symmetrisch, aber stattdessen antisymmetrisch. Zur Erinnerung: Wir hatten früher definiert, dass eine Relation R auf einer Menge A antisymmetrisch ist, wenn für alle $x, y \in A$ mit $x \neq y$ und xRy gilt, dass $(y, x) \notin R$. Zum Testen der Bedingung für Antisymmetrie können wir aber auch die folgende zu §5-9 (iii) äquivalente Bedingung verwenden:

§5-21 Satz. (Äquivalente Bedingung für Antisymmetrie)

Eine Relation R ist antisymmetrisch, genau dann wenn aus xRy und yRx folgt, dass $x = y$ gelten muss. Formal also:

$$\begin{aligned} R \subseteq A \times A \text{ ist antisymmetrisch} \\ \iff \forall x, y \in A : (xRy \wedge yRx \Rightarrow x = y) \end{aligned}$$

\diamond

Beweis. Der Beweis ist nicht Prüfungssto und ist deshalb in den ergänzenden Notizen zu finden^[E.16].

\square

§5-22 Beispiel. (Standard-Beispiel einer Halbordnung)

Die Relation $\preceq := \{(x, y) \in \mathbb{R}^2 \mid x \leq y\}$ ist das Standard-Beispiel einer Halbordnung.

(Wie früher gesehen: \leq ist reflexiv, antisymmetrisch und transitiv, also ist die Definition einer Halbordnung erfüllt.)

\diamond

Wenn wir die Menge der Elemente im vorhergehenden Beispiel endlich machen und in der Anzahl stark einschränken, dann erkennen wir die Halbordnungseigenschaft auch an der aufzählenden Darstellung:

§5-23 Beispiel. (Halbordnung auf der Menge $M := \{0, 1, 2, 3\}$)

Sei M die Menge $M := \{0, 1, 2, 3\}$. Dann ist

$$\preceq := \{(x, y) \in M^2 \mid x \leq y\}$$

eine Halbordnung auf M gegeben durch

$$\begin{aligned} \preceq = & \{(0, 0), (0, 1), (0, 2), (0, 3), \\ & (1, 1), (1, 2), (1, 3), \\ & (2, 2), (2, 3) \\ & (3, 3)\}. \end{aligned}$$

Das heisst, (M, \preceq) ist eine halbgeordnete Menge. (Überlegen Sie sich anhand der Auflistung der Elemente, dass es in der Tat eine reflexive, antisymmetrische und transitive Relation ist.) \diamond

Es gibt ein paar Grundbegriffe im Zusammenhang mit Halbordnungsrelationen, welche immer wieder benutzt werden und die wir uns merken sollten:

§5-24 Definition. (Grundbegriffe zu Halbordnungen)

Sei \preceq eine Halbordnung auf der Menge A .

- (1) Ein Element $x \in A$ heisst dann **Vorgänger von $y \in A$** , gdw. $x \preceq y$ und $x \neq y$ gilt.
In dieser Situation nennen wir y den **Nachfolger von x** .
- (2) Sei $x \preceq y$ und $x \neq y$ (d.h. also, dass x Vorgänger von y bzw. y Nachfolger von x ist). Dann nennen wir x **direkten Vorgänger** und y **direkten Nachfolger**, gdw. $\neg \exists z \in A, z \neq x, z \neq y : (x \preceq z \wedge z \preceq y)$.
(Mit anderen Worten, "es gibt kein Element zwischen x und y ".)
- (3) Ein Element $x \in A$ heisst **minimales Element** (von A bzgl. der Halbordnung \preceq), gdw. x keine direkten Vorgänger hat.
- (4) Ein Element $x \in A$ heisst **kleinstes Element** (von A) bezüglich \preceq , wenn alle anderen Elemente y Nachfolger von x sind, d. h. für alle $y \in A$ mit $y \neq x$ gilt, dass $x \preceq y$.
- (5) Ein Element $x \in A$ heisst **maximales Element** (von A bzgl. der Halbordnung \preceq), gdw. x keine direkten Nachfolger hat.
- (6) Ein Element $x \in A$ heisst **grösstes Element** (von A) bezüglich \preceq , wenn alle anderen Elemente y Vorgänger von x sind, d. h. für alle $y \in A$ mit $y \neq x$ gilt, dass $y \preceq x$.

\diamond

Wir können also auch sagen, dass ein **minimales Element**, das eindeutig ist, **kleinstes Element** genannt wird und dass ein **maximales Element**, das eindeutig ist, **grösstes Element** heisst.

§5-25 Beispiel.

Auf der durch

$$\preceq = \{(0, 0), (0, 1), (0, 2), (0, 3), (1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3)\}$$

definierten Halbordnung auf $M := \{0, 1, 2, 3\}$ des früheren Beispiels §5-23 gilt zum Beispiel:

- 0, 1, 2 sind Vorgänger von 3, wobei 2 sogar direkter Vorgänger ist.

- 3 ist damit auch direkter Nachfolger von 2
- 3 ist Nachfolger von 0, 1, 2.
- 0 ist minimales und sogar kleinstes Element in M .
- 3 ist maximales und sogar grösstes Element in M .

◊

Es ist nicht so, dass nur die "o ensichtliche" Ordnung \leq auf Zahlenmengen, wie in den Beispielen §5-22 und §5-23 studiert, eine Halbordnungsrelation definiert. Ein erstes anderes Beispiel, welches nicht auf einer Zahlenmenge, sondern auf einem Mengensystemen – also einer "Menge mit wieder Mengen als Elemente" – definiert ist, ist folgendes. Die Ordnungsrelation ist dabei gerade die Teilmengenrelation \subseteq , welche wir schon in der Mengenlehre kennengelernt haben:

§5-26 Beispiel. (Halbordnung auf der Potenzmenge von $M := \{1, 2, 3\}$)

Sei die Menge $M := \{1, 2, 3\}$ gegeben und deren Potenzmenge

$$\mathcal{P}(M) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

Dann ist

$$\preceq := \{(S, T) \in \mathcal{P}(M)^2 \mid S \subseteq T\}$$

eine Halbordnung auf dieser Potenzmenge $\mathcal{P}(M)$.

Wir überlegen uns, dass dann die Halbordnung \preceq gegeben ist durch folgende geordnete Paare

$$\begin{aligned} \preceq = & \{(\emptyset, \emptyset), (\emptyset, \{1\}), (\emptyset, \{2\}), (\emptyset, \{3\}), \\ & (\emptyset, \{1, 2\}), (\emptyset, \{1, 3\}), (\emptyset, \{2, 3\}), (\emptyset, \{1, 2, 3\}), \\ & (\{1\}, \{1\}), (\{1\}, \{1, 2\}), (\{1\}, \{1, 3\}), (\{1\}, \{1, 2, 3\}), \\ & (\{2\}, \{2\}), (\{2\}, \{1, 2\}), (\{2\}, \{1, 3\}), (\{2\}, \{1, 2, 3\}), \\ & (\{3\}, \{3\}), (\{3\}, \{1, 3\}), (\{3\}, \{2, 3\}), (\{3\}, \{1, 2, 3\}), \\ & (\{1, 2\}, \{1, 2\}), (\{1, 2\}, \{1, 2, 3\}), \\ & (\{1, 3\}, \{1, 3\}), (\{1, 3\}, \{1, 2, 3\}), \\ & (\{2, 3\}, \{2, 3\}), (\{2, 3\}, \{1, 2, 3\}), \\ & (\{1, 2, 3\}, \{1, 2, 3\})\} \end{aligned}$$

◊

Die Darstellung einer Halbordnung als gerichteter Graph ist nun relativ ine zient und auch schnell mal unübersichtlich. Wir erhalten eine übersichtlichere graphische Darstellung einer Halbordnung, indem wir

- die Reflexivitäten bei jedem Element (Schleifen bei jedem Knoten) weglassen, welche diese bei einer Halbordnung sowieso immer dabei sind,
- die "abkürzenden" Pfeile aufgrund der Transitivitäten auch weglassen (auch die sind per Definition ja immer dabei),
- die Pfeile durch einfache ungerichtete Kanten ersetzen und dafür eine Richtung "von unten nach oben" in der Zeichenebene festlegen.

Die so entstehende, schlankere Visualisierung nennt man auch "transitive Reduktion" des gerichteten Graphen (weil es insbesondere um die transitiven Pfeile reduziert ist) oder meist einfach "Hasse-Diagramm" einer Halbordnung.

§5-27 Definition. (Hasse-Diagramm einer Halbordnung)

Es sei \preceq eine Halbordnung auf einer (endlichen) Menge A . Das Hasse-Diagramm von \preceq ist wie folgt aufgebaut:

- Jene Elemente von A , die keine Vorgänger haben, stehen auf einer untersten Stufe der Zeichenebene.
- Auf jeder höheren Stufe stehen alle Elemente, die direkte Nachfolger von Elementen in der um Eins tiefer liegenden Stufe sind. Direkte Nachfolger werden durch eine Kante mit ihrem direkten Vorgänger verbunden (**und nur die!** vgl. Sie die Definition von "direktem Nachfolger" in §5-24!).
- Dies wird pro "Stufe" in der Zeichenebene solange ausgeführt, bis kein Element mit einem direkten Nachfolger mehr übrigbleibt.

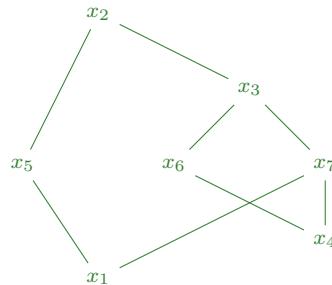
◊

§5-28 Beispiel. (Hasse-Diagramm einer Halbordnung \preceq)

Sei $X := \{x_1, x_2, \dots, x_7\}$ und eine Halbordnung in aufzählender Form auf X gegeben durch

$$\begin{aligned}\preceq = & \{(x_1, x_1), (x_2, x_2), (x_3, x_3), (x_4, x_4), (x_5, x_5), (x_6, x_6), (x_7, x_7), \\ & (x_1, x_5), (x_1, x_6), (x_1, x_7), (x_1, x_3), (x_1, x_2), \\ & (x_4, x_6), (x_4, x_7), (x_4, x_3), (x_4, x_2), (x_5, x_2), \\ & (x_6, x_3), (x_6, x_2), (x_7, x_3), (x_7, x_2), (x_3, x_2)\} \\ & \subseteq X \times X.\end{aligned}$$

Dann erhalten wir folgendes Hasse-Diagramm für (X, \preceq) :



Beachten Sie, dass die gezeigte Richtung im Allgemeinen die schwierigere oder zumindest aufwändigere ist, da die Transitivitäten berücksichtigt werden müssen (Stichwort "Transitive Reduktion").

Die Umkehrung ist einfacher: Aus einem Hasse-Diagramm kann der bipartite Graph bzw. die aufzählende Form relativ einfach ermittelt werden.

Im weiteren ist das obige Hasse-Diagramm absichtlich nicht in der ästhetisch schönsten und regelmässigsten Form gezeichnet, damit Sie daran denken, dass die alleinige Information über die Halbordnung in den Kantenverbindungen steckt und es zum Beispiel egal ist, ob ich Knoten x_4 etwas höher als Knoten x_1 zeichne oder tiefer. x_1 ist ja weder Vorgänger noch Nachfolger von x_4 , da es keinen nach oben gerichteten Kantenpfad zwischen ihnen gibt (s.a. nachfolgend §5-31). ◊

Für das andere frühere Beispiel erhalten wir folgendes Hasse-Diagramm:

§5-29 Beispiel. (Hasse-Diagramm der Halbordnung von §5-26)

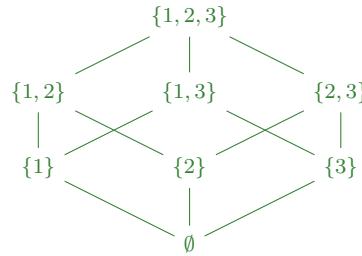
Wie bereits gesehen ist auf

$$\mathcal{P}(M) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

die Halbordnung in aufzählender Form gegeben durch

$$\preceq = \{(\emptyset, \emptyset), (\emptyset, \{1\}), (\emptyset, \{2\}), (\emptyset, \{3\}), (\emptyset, \{1, 2\}), (\emptyset, \{1, 3\}), (\emptyset, \{2, 3\}), (\emptyset, \{1, 2, 3\}), \\ (\{1\}, \{1\}), (\{1\}, \{1, 2\}), (\{1\}, \{1, 3\}), (\{1\}, \{1, 2, 3\}), \\ (\{2\}, \{2\}), (\{2\}, \{1, 2\}), (\{2\}, \{1, 3\}), (\{2\}, \{2, 3\}), (\{2\}, \{1, 2, 3\}), \\ (\{3\}, \{3\}), (\{3\}, \{1, 3\}), (\{3\}, \{2, 3\}), (\{3\}, \{1, 2, 3\}), \\ (\{1, 2\}, \{1, 2\}), (\{1, 2\}, \{1, 2, 3\}), \\ (\{1, 3\}, \{1, 3\}), (\{1, 3\}, \{1, 2, 3\}), \\ (\{2, 3\}, \{2, 3\}), (\{2, 3\}, \{1, 2, 3\}), \\ (\{1, 2, 3\}, \{1, 2, 3\})\} \subseteq \mathcal{P}(M) \times \mathcal{P}(M).$$

In relativ aufwändiger, systematischer Überlegung und dem Streichen der nicht-relevanten Paare in der Aufzählung erhalten wir das folgende Hasse-Diagramm für $(\mathcal{P}(M), \preceq)$:



(Achtung: Dieses Diagramm ist *nicht* etwa "3-dimensional" als Quader oder Ähnliches zu sehen!
Das Diagramm liegt tatsächlich in der Zeichenebene und Kantenüberschneidungen sind
tatsächlich auch Kantenüberschneidungen.)

◊

Ein kleiner Trick, wie man unter Umständen doch schneller zum Hasse-Diagramm einer Halbordnung kommt, liefert der folgende Algorithmus.

§5-30 Algorithmus. (Reduktion am gerichteten Graphen einer Halbordnung zur Ermittlung des Hasse-Diagramms)

Ist für die Halbordnung der ungerichtete Graph gegeben, dann kann ich

1. alle Schleifen (Reflexivitäten) löschen, und
2. alle "abkürzenden" Pfeile, welche es zwischen gleichgerichteten "Zwei-Pfeile-Pfade" gibt (Transitivitäten) löschen,
(so erhalte ich quasi das Gerüst des Hasse-Diagramms und jeder übriggebliebene Pfeil entspricht genau einer ungerichteten Kante im Hasse-Diagramm.)
3. das erhaltene Zwischenergebnis nur noch richtig bzgl. der Halbordnung in der Zeichenebene anordnen und
4. die Pfeilspitzen eliminieren.

◊

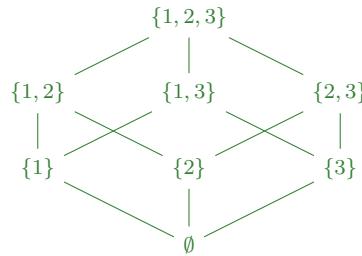
Dieser Algorithmus kann natürlich nur angewendet werden, wenn die halbgeordnete Menge relativ klein und insbesondere endlich ist, und vorzugsweise, wenn eben schon der gerichtete Graph ermittelt wurde. In manchen Fällen kann es sich deshalb auch lohnen, für das Hasse-Diagramm zuerst den gerichteten Graphen der Halbordnung zu ermitteln und dann mit obigem Algorithmus zum Hasse-Diagramm zu gelangen.

§5-31 Definition. (Unvergleichbare Elemente in einer Halbordnung)

Zwei Elemente x und y einer halbgeordneten Menge (M, \preceq) heissen **unvergleichbar**, genau dann wenn $(x, y) \notin \preceq \wedge (y, x) \notin \preceq$. \diamond

§5-32 Beispiel. (Unvergleichbare Elemente in §5-26)

In



sind zum Beispiel $\{1\}$ und $\{2\}$, oder auch $\{1, 3\}$ und $\{2, 3\}$ bzw. $\{3\}$ und $\{1, 2\}$ unvergleichbar. Dies "sieht" man auch in der Auflistung, wenn auch bedeutend weniger gut:

$$\preceq = \{(\emptyset, \emptyset), (\emptyset, \{1\}), (\emptyset, \{2\}), (\emptyset, \{3\}), (\emptyset, \{1, 2\}), (\emptyset, \{1, 3\}), (\emptyset, \{2, 3\}), (\emptyset, \{1, 2, 3\}), (\{1\}, \{1\}), (\{1\}, \{1, 2\}), (\{1\}, \{1, 3\}), (\{1\}, \{2, 3\}), (\{2\}, \{2\}), (\{2\}, \{1, 2\}), (\{2\}, \{1, 3\}), (\{2\}, \{2, 3\}), (\{2\}, \{1, 2, 3\}), (\{3\}, \{3\}), (\{3\}, \{1, 3\}), (\{3\}, \{2, 3\}), (\{3\}, \{1, 2, 3\}), (\{1, 2\}, \{1, 2\}), (\{1, 2\}, \{1, 2, 3\}), (\{1, 3\}, \{1, 3\}), (\{1, 3\}, \{1, 2, 3\}), (\{2, 3\}, \{2, 3\}), (\{2, 3\}, \{1, 2, 3\}), (\{1, 2\}, \{1, 2\})\}$$

\diamond

Gibt es Mengen, deren Elemente immer vergleichbar sind? D.h., für welche es eine Halbordnung gibt, so dass für alle Elemente x, y :

$$x \preceq y \vee y \preceq x \quad ?$$

Ja, natürlich!

§5-33 Beispiel. (Halbgeordnete Menge, in welcher alle Elemente vergleichbar sind)

Bei der durch \leq definierten Halbordnung auf $M := \{0, 1, 2, 3\}$ im früheren Beispiel §5-23 ist jedes Element vergleichbar! \diamond

Allgemeiner gilt sogar, dass alle Elemente der durch \leq auf $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ definierten Halbordnung vergleichbar sind! Halbordnungen in welchen alle Elemente vergleichbar sind heissen übrigens auch **Totalordnungen** oder **lineare Ordnungen**. Wie sieht in einem solchen Fall (einer Totalordnung) das Hasse-Diagramm aus? Zum Beispiel für die ganzen Zahlen (\mathbb{Z}, \leq) sieht das so aus, was auch den Namen "lineare Ordnung" erklärt:

$$\begin{array}{c} \vdots \\ 3 \\ | \\ 2 \\ | \\ 1 \\ | \\ 0 \\ | \\ -1 \\ | \\ -2 \\ | \\ -3 \\ \vdots \end{array}$$

Offensichtlich gibt es damit auf $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ kein minimales und kein maximales Element. Es gilt jedoch folgender wichtiger Satz:

§5-34 Satz.

In jeder endlichen (!) Menge auf der eine Halbordnung \preceq existiert, gibt es ein minimales und ein maximales Element. \diamond

Beachten Sie aber, dass die Eindeutigkeit des minimalen (oder maximalen) Elements trotzdem nicht gelten muss. D.h., ein kleinsten (oder grösstes) Element muss natürlich auch bei endlichen halbgeordneten Mengen nicht notwendigerweise existieren! Ein Beispiel dafür, wo wir in einem solchen Fall kein kleinstes Element haben, haben wir ja im Beispiel §5-28 kennengelernt.

Ein paar Anwendungsbeispiele von Halbordnungen gibt es im mgli-Wiki der Modul-Moodleseite unter dem Eintrag "Halbordnungsrelationen ... wozu?".

5.6 Operationen zwischen und mit Relationen

Aus gegebenen Relationen können neue Relationen konstruiert werden. Zwei solche Beispiele von Operationen wollen wir in diesem Abschnitt kennenlernen. Dabei wollen wir uns wieder auf den Fall binärer Relationen $R \subseteq A \times B$ beschränken.

5.6.1 Verknüpfung zweier binärer Relationen

Zwei Relationen R und S können miteinander verknüpft werden, wenn die "mittleren Mengen" kompatibel sind, d.h., wenn die zweite Menge der ersten Relation gleich der ersten Menge der zweiten Relation ist. [E.17]

§5-35 Beispiel. (Prinzip der Verknüpfung zweier binärer Relationen)

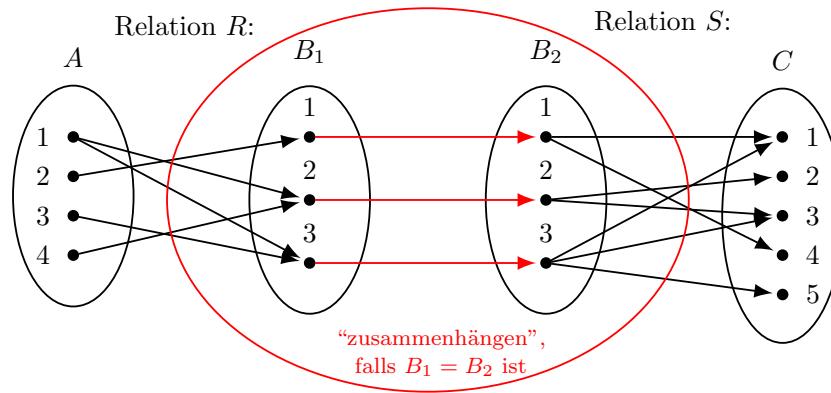


Abbildung 5.3 – Verknüpfung $S \circ R$ zweier kompatibler Relationen.

Beachten Sie unbedingt die Reihenfolge in der Notation $S \circ R$! Diese Verknüpfung liefert so eine Relation auf $A \times C$:

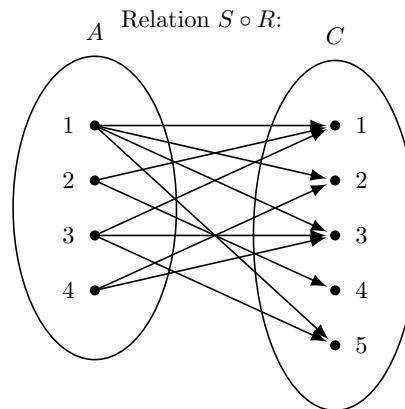


Abbildung 5.4 – Das Ergebnis der Verknüpfung $S \circ R$ von R mit S .

Man erhält die Elemente der neuen Relation, indem man sich von jedem Element von A über jede mögliche Pfeilfolge zum entsprechenden Element in C "durchhangelt". Über In aufzählender Form lautet die Verknüpfung $S \circ R$ damit

$$\begin{aligned} S \circ R &= \{(1, 1), (1, 2), (1, 3), (1, 5), (2, 1), (2, 4), (3, 1), (3, 3), (3, 5), (4, 2), (4, 3)\} \\ &\subseteq A \times C \end{aligned}$$

◊

Formal definieren wir diese Operation so:

§5-36 Definition. (Verknüpfung von binären Relationen)

Seien $R \subseteq A \times B$ und $S \subseteq B \times C$ zwei binäre Relationen.

Die Verknüpfung $S \circ R$ der beiden Relationen ist definiert als die Relation über $A \times C$ mit

$$S \circ R := \{(x, z) \in A \times C \mid \exists y \in B : (xRy \wedge ySz)\}$$

◊

§5-37 Beispiel. (Verknüpfung zweier binären Relationen)

Seien folgende homogene binäre Relationen R und S auf der Menge M von Menschen gegeben:

$$R := \{(x, y) \in M^2 \mid "x \text{ ist Kind von } y." \}$$

$$S := \{(x, y) \in M^2 \mid "x \text{ ist Schwester oder Bruder von } y." \}$$

Dann ist die Verknüpfung $S \circ R$ die Relation

$$S \circ R = \{(x, y) \in M^2 \mid "x \text{ ist Nichte oder Neffe von } y." \}.$$

◊

§5-38 Beispiel.

Wir überlegen uns, wie die Relationsverknüpfung von Beispiel §5-37 für den Fall, dass Franz' Schwester Dora die zwei Kinder Anna und Bernd hat und Franz' andere Schwester Elsbeth einen Sohn Christoph.

Die – in diesem Fall homogenen – Relationen sind also:

$$R = \{(Anna, Dora), (Bernd, Dora), (Christoph, Elsbeth)\},$$

$$S = \{(Franz, Dora), (Dora, Franz), (Franz, Elsbeth), (Elsbeth, Franz), \\ (Dora, Elsbeth), (Elsbeth, Dora), (Anna, Bernd), (Bernd, Anna)\}.$$

Wir schreiben in den Graphen abkürzend

$$A := \text{"Anna"}, B := \text{"Bernd"}, C := \text{"Christoph"}, \dots \text{usw..}$$

Die bipartiten Graphen dieser Verknüpfung sehen dann so aus:

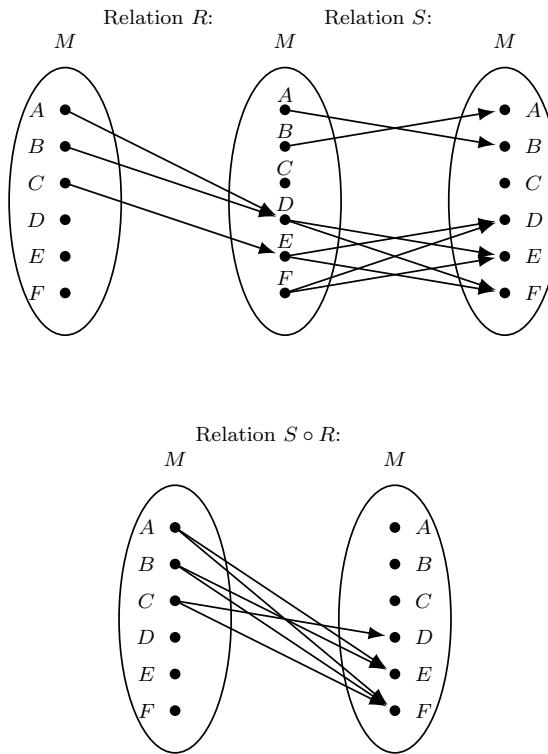


Abbildung 5.5 – Die bipartiten Graphen der Verknüpfung $S \circ R$ für die konkreten R und S .

D.h. in aufzählender Form:

$$S \circ R = \{(Anna, Elsbeth), (Anna, Franz), (Bernd, Elsbeth), \\ (Bernd, Franz), (Christoph, Dora), (Christoph, Franz)\}.$$

◇

5.6.2 Inversion einer binären Relationen

Eine Relation R kann umgedreht (wir sagen auch “invertiert”) werden, indem wir die beiden Mengen vertauschen und die Pfeile einfach umdrehen.

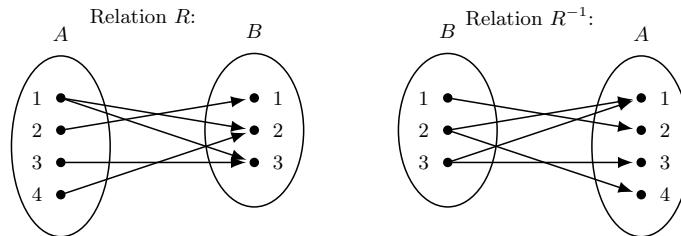


Abbildung 5.6 – Das einfache Prinzip einer inversen Relation R^{-1} .

Die so entstandene Relation bezeichnen wir mit R^{-1} und nennen sie die inverse Relation von R :

§5-39 Definition. (Inverse Relation)

Sei $R \subseteq A \times B$ eine binäre Relation über $A \times B$.

Die inverse Relation R^{-1} ist die Relation definiert über $B \times A$ mit

$$R^{-1} = \{(y, x) \in B \times A \mid (x, y) \in R\}.$$

◊

§5-40 Beispiele. (Inversion von Relationen)

- (1) Die inverse Relation der Relation \leq ist \geq .
- (2) Die inverse Relation der Relation "ist Mutter von" ist die Relation "ist Kind von Mutter".
- (3) Zeichnen Sie den bipartiten Graphen und geben Sie die Auflistung der Elemente von $(S \circ R)^{-1}$ unseres Beispiels der Verknüpfungsoperation §5-35.

Aus

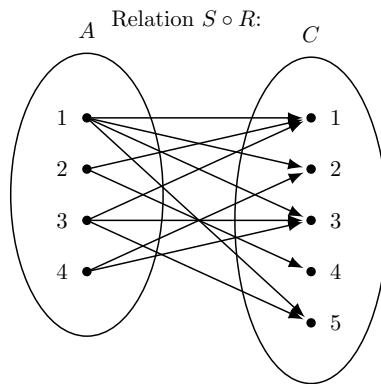


Abbildung 5.7 – Das einfache Prinzip einer inversen Relation R^{-1} .

$$\begin{aligned} S \circ R &= \{(1, 1), (1, 2), (1, 3), (1, 5), (2, 1), (2, 4), \\ &\quad (3, 1), (3, 3), (3, 5), (4, 2), (4, 3)\} \\ &\subseteq A \times C \end{aligned}$$

folgt durch Umkehrung der Mengen und Pfeile, bzw. der geordneten Paare:

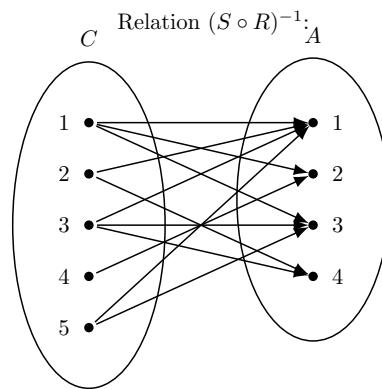


Abbildung 5.8 – Das einfache Prinzip einer inversen Relation R^{-1} .

Also

$$\begin{aligned}
 (S \circ R)^{-1} &= \{(1, 1), (1, 2), (1, 3), (2, 1), (2, 4), (3, 1), \\
 &\quad (3, 3), (3, 4), (4, 2), (5, 1), (5, 3)\} \\
 &\subseteq C \times A
 \end{aligned}$$

◇

Kapitel 6

Graphen

Kapitelinhalt

6.1	Grundlagen	123
6.1.1	Gerichtete Graphen	123
6.1.2	Ungerichtete Graphen	128
6.1.3	(Ungerichtete) Multigraphen	134
6.2	Zusammenhang und Zusammenhangskomponenten	137
6.3	Modellierung von Anwendungsproblemen	140
6.3.1	Königsberger Brückenproblem	141
6.3.1.1	Kurzer Exkurs: Notwendige und hinreichende Bedin- gung	141
6.3.1.2	Das Königsberger Brückenproblem	142
6.3.1.3	Lösung des Königsberger Brückenproblems; Eulerbe- dingungen in (Multi)Graphen	145
6.3.1.4	Eulerwege und Eulerkreise in allgemeinen (Multi)Graphen	146
6.3.2	Bäume, Spannbäume und typische Anwendungen	150
6.3.2.1	Bäume	150
6.3.2.2	Spannbäume	153
6.3.2.3	Spannbäume kantengewichteter Graphen	154
6.3.2.4	Algorithmen von Kruskal und Prim zur Lösung der Anwendungsaufgabe	155
6.4	Matrix-Darstellungen von Graphen	156
6.5	Graph- und Digraphisomorphismus	162

L E R N Z I E L E :

- LZ 6.1** Sie kennen den Begriff eines gerichteten und eines ungerichteten Graphen.
- LZ 6.2** Sie kennen die wichtigsten Eigenschaften von Graphen.
- LZ 6.3** Sie sind in der Lage, Probleme aus der Praxis mit Hilfe von Graphen zu modellieren.
-

6.1 Grundlagen

Wir haben gesehen, dass wir mit Relationen ganz viele Situationen und Probleme aus Anwendungen darstellen und "bearbeiten" können. Wir haben bei Relationen auch schon zwei weitere Darstellungen von binären Relationen $R \subseteq A \times B$ zwischen endlichen Mengen kennengelernt: der bipartite Graph (wenn $A \neq B$ ist) und der gerichtete Graph (wenn $A = B$ ist). Damit haben wir schon gesehen, wie nützlich Graphen bei der Handhabung von Relationen sind. Dieses sehr mächtige Werkzeug, welches übrigens ganz allgemein in der sogenannten "Graphentheorie" untersucht wird, wollen wir jetzt einführend etwas genauer kennenlernen und erste Eindrücke gewinnen, wie Anwendungsprobleme mit Werkzeugen der Graphentheorie gelöst werden können. Dabei bleibt es natürlich bei einem ganz kleinen Einblick, denn das Gebiet der Graphentheorie mit all ihren verschiedenen Facetten ist riesig.

Eine der grundlegendsten Unterscheidungen von Graphen ist diejenige zwischen ungerichteten und gerichteten Graphen. Der ungerichtete Graph besteht aus einer endlichen Menge von Knoten (engl. "vertices", sg. "vertex") und einer ebenfalls endlichen Menge von (ungerichteten) Kanten (engl. "edges"), welche je zwei Knoten verbinden. Das heißt, so eine Verbindung hat keine Richtung, bzw. keiner der beteiligten Knoten ist der "erste Knoten" und der andere der "zweite". Deshalb können wir so eine ungerichtete Kante zwischen zwei Knoten v und w einfach als zweielementige Menge $\{v, w\}$ schreiben. Bekanntlich wird dies dann auch nicht von $\{w, v\}$ unterschieden. Bei einem gerichteten Graphen dagegen, werden zwei Knoten v und w durch eine gerichtete Kante, auch Pfeil (engl. "arrow") genannt, verbunden. Dort wo der Pfeil startet haben wir den Startknoten und dort wo der Pfeil hinzeigt bzw. endet ist der Endknoten. Wir haben jetzt also eine Richtung und auch einen ersten Knoten (Startknoten genannt) und einen zweiten Knoten (Endknoten genannt). Der Pfeil entspricht damit einem geordneten Paar (v, w) von Knoten und (v, w) und (w, v) ist nicht mehr die gleiche gerichtete Kante bzw. der gleiche Pfeil. Wir haben bei den Relationen gesehen, dass solche gerichteten Graphen (ob bipartit oder nicht) in natürlicher Weise eine binäre Relation zwischen den beteiligten Knoten darstellen.

Einen ungerichteten Graphen können wir übrigens auf verschiedene Weise aussen: Wir können ihn zum Beispiel als Darstellung einer Relation aussen, in welcher die Richtung keine Rolle spielt. Oder als eine Relation, welche symmetrisch ist und wo das Paar von entgegengesetzt gerichteten Pfeilen einfach zusammengefasst als Kante gezeichnet wird. Andere Eigenschaften der einen Graphen bestimmenden Relation, wie zum Beispiel die Reflexivität, können zum Beispiel auch ausgeschlossen werden, dann spricht man von einem "einfachen Graphen". Wir wollen das Wesentliche einmal anhand der üblichen grundlegenden Definitionen festhalten. Dabei beginnen wir beim gerichteten Graphen, weil wir diesen ja schon aus der Anwendung mit Relationen kennen:

6.1.1 Gerichtete Graphen

§6-1 Definition. (Gerichteter Graph (Digraph))

Ein gerichteter Graph X (Digraph) ist ein 2-Tupel $X := (V, E)$ mit

- (i) V ist eine endliche Menge von Knoten $\{v_1, v_2, \dots, v_n\}$.
- (ii) E ist eine endliche Menge von Pfeilen (gerichteten Kanten), d.h., eine Teilmenge

$$E \subseteq V \times V$$

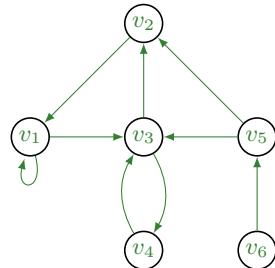
und damit eine homogene binäre Relation auf der Knotenmenge V . Die Pfeile als Knotenpaare werden üblicherweise in einer natürlichen Ordnung (z.B.

lexikografisch, also wie Wörter in einem Lexikon oder Wörterbuch) geordnet und dann mit e_i für aufsteigende $i \in \mathbb{N}$ bezeichnet und nummeriert.

◊

Oft meint man auch mit dem Wort "Graph" genauer einen ungerichteten Graphen (siehe nächster Abschnitt 6.1.2). Mit "Digraph" (aus dem englischen "directed graph") meint man dann zur Unterscheidung einen solchen gerichteten Graphen, wie eben definiert. Zur Ergänzung sei hier gesagt, dass diese grundlegenden Begriffe in verschiedener Hinsicht verallgemeinert werden können (Multigraph, Multidigraph, kanten-/knotengewichtete Graphen und Digraphen, Hypergraphen, ...). Aber wir wollen nichts überstürzen und uns vorerst einmal nur auf diesen eben definierten Typen eines gerichteten Graphen beschränken.

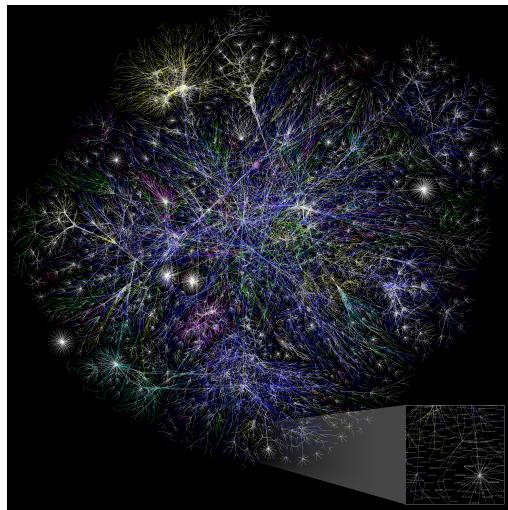
§6-2 Beispiel. (Ein gerichteter Graph)



$X = (V, E)$ mit
 $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ und
 $E = \{(v_1, v_1), (v_1, v_3), (v_2, v_1),$
 $(v_3, v_2), (v_3, v_4), (v_4, v_3),$
 $(v_5, v_2), (v_5, v_3), (v_6, v_5)\}$

◊

§6-3 Beispiel. (Alltagsbeispiel eines gerichteten Graphen)



Eine Karte des Internets. [A1]

Das Internet kann als Menge von Webseiten (Knoten) aufgefasst werden, in welcher jede Webseite eine Anzahl von Hyperlinks (gerichtete Kanten) auf andere Webseiten enthält. Wie "wichtig" ist eine Webseite? Google's berühmter *PageRank Algorithmus* benutzt verschiedenste Mathematik: Random Walks auf Graphen, Lineare Algebra, Wahrscheinlichkeitstheorie, ... etc. (Details: z.B. [3, 17, 5] ... für später :-))

◊

Der eben definierte Typ eines gerichteten Graphen ist natürlich noch sehr allgemein. In Anwendungen und bei der Modellierung und Lösung von Problemen sind viele mögliche spezielle Eigenschaften von entscheidender Bedeutung. Wir wollen hier ein paar grundlegende Begriffe und Eigenschaften zusammenstellen. Wenn wir im Folgenden von einem "Digraphen" sprechen, dann meinen wir immer so einen gerichteten Graphen.

§6-4 Definition. (Grundbegri e)

- (1) Die Anzahl $|V|$ von Knoten und die Anzahl $|E|$ von Kanten bzw. Pfeile sind natürlich ein erstes Charakteristikum eines Digraphen.
- (2) Bei gerichteten Graphen können wir den Grad eines Knotens – d.h., die Anzahl Kanten, welche ihn als Knoten enthalten – je nach Pfeilrichtung unterscheiden: Zählen wir die Pfeile, welche vom Knoten ausgehen, dann sprechen wir vom Ausgangsgrad $\deg_{out}(v)$, zählen wir die Pfeile, welche im Knoten v enden, dann sprechen wir vom Eingangsgrad $\deg_{in}(v)$.
- (3) In einem gerichteten Graphen nennen wir bei einem Pfeil $e := (u, v) \in E$ den Knoten u auch den Vorgänger von v und Knoten v entsprechend den Nachfolger von u .

Wir sagen dann auch, dass u und e positiv inzident (da der Pfeil vom Knoten herausgeht), und dass v und e negativ inzident (da der Pfeil zum Knoten hin gerichtet ist) sind.

◊

Folgende Beispiele verdeutlichen diese Grundbegri e:

§6-5 Beispiel. (Grundbegri e)

Für den folgenden gerichteten Graphen $X = (V, E)$ gilt beispielsweise



Weiter ist z.B. wegen e_6 der Knoten v_3 der Vorgänger von v_2 und v_2 der Nachfolger von v_3 . v_3 und e_6 sind also positiv inzident und v_2 und e_6 negativ inzident. ◊

§6-6 Definition. (Weg (Pfad) in einem gerichteten Graphen; Kreis (Zyklus); Schleife (Loop))

- (1) Ein Weg oder Pfad in einem gerichteten Graphen $X = (V, E)$ ist eine Folge

von Knoten

$v_{i_1}, v_{i_2}, v_{i_3}, \dots, v_{i_{k-1}}, v_{i_k}$,

für ein $k \in \mathbb{N}$, wobei $\forall j \in \mathbb{N}_{k-1}$ gilt, dass $(v_{i_j}, v_{i_{j+1}}) \in E$ (d.h., in der Kantenmenge) ist.^[E,18]

- (2) Ist bei einem Weg der Anfangs- und der Endknoten gleich, also gilt für

$v_{i_1}, v_{i_2}, v_{i_3}, \dots, v_{i_{k-1}}, v_{i_k}$

$$v_{i_1} = v_{i_k},$$

so sprechen wir von einem (gerichteten) Kreis oder einem Zyklus.

- (3) Eine Schleife (Loop) ist eine gerichtete Kante (v_i, v_i) .

◇

§6-7 Beispiele.

Im gerichteten Graphen von Beispiel §6-5 gilt beispielsweise:

v_3, v_4, v_1, v_1, v_5

ist ein Weg (Pfad),

v_1, v_4, v_3, v_4, v_1

ein Kreis und

(v_1, v_1) sowie (v_5, v_5)

Schleifen.

Da auch die Kanten im Graphen §6-5 alle bezeichnet sind, könnten wir natürlich auch diese Kantenbezeichnungen benutzen. Dann gilt entsprechend, dass beispielsweise:

e_7, e_8, e_1, e_4

ein Weg (Pfad) ist,

e_3, e_9, e_7, e_8

ein Kreis und

e_1 sowie e_{11}

Schleifen sind.

◇

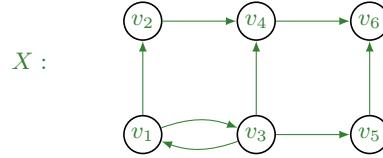
§6-8 Definition. (Teildigraph, Oberdigraph eines gerichteten Graphen)

Ein Digraph $X' = (V', E')$ heisst **Teildigraph** (oder **Subdigraph**, **Unterdigraph**) eines Graphen $X = (V, E)$, falls $V' \subseteq V$ und $E' \subseteq E$ gilt. In einem solchen Fall nennen wir X dann auch den **Oberdigraphen** (oder **Superdigraphen**) von X' .

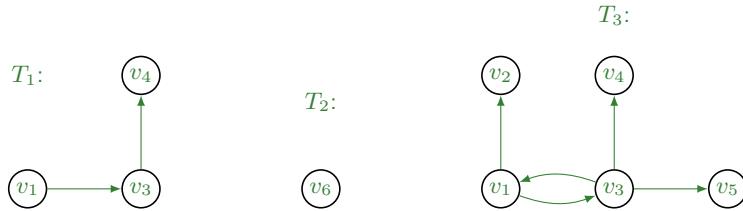
◇

§6-9 Beispiel. (Teildigraphen)

Sei folgender Digraph $X = (V, E)$ gegeben:



Dann sind in X die folgenden drei Digraphen T_1, T_2, T_3 je Beispiele für Teildigraphen:



Keine Teildigraphen in X sind zum Beispiel die Graphen X_1 und X_2 :



◇

Bipartite Digraphen haben wir bei der grafischen Darstellung von binären Relationen schon kennengelernt. Formal sind sie folgendermassen definiert:

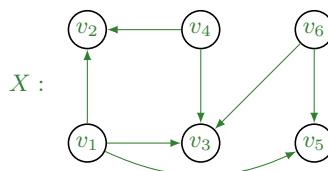
§6-10 Definition. (Bipartiter Digraph)

Ein bipartiter Digraph ist ein Digraph $X = (V, E)$, dessen Knotenmenge V in zwei Knoten-Teilmengen A und B partitioniert werden kann, so dass jede Kante $e \in E$ entweder in A startet und in B endet, oder in B startet und in A endet.

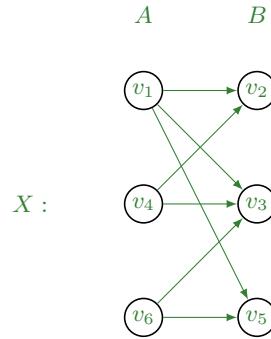
◇

§6-11 Beispiel. (Bipartiter Digraph)

Der Digraph



ist bipartit, weil wir mit $A = \{v_1, v_4, v_6\}$ und $B := \{v_2, v_3, v_5\}$ eine entsprechende Partitionierung der Knotenmenge haben. Dies sieht man besser, wenn wir die Knoten anders anordnen (wie wir das bei den Relationen kennengelernt haben):



Bei Relationen war es dann einfach noch üblich, dass wir die Elemente von A und B je "Venn-Diagramm-mässig" als Mengendiagramm umfasst haben. Dies ist aber nur eine leicht verschiedene Darstellungsvariante. \diamond

6.1.2 Ungerichtete Graphen

In vielen Anwendungen spielt die Richtung eines Pfeils zwischen zwei Knoten keine Rolle. Das heisst, in der Relation, welche durch einen Graphen beschrieben wird spielt es keine Rolle ob (v, w) oder (w, v) in der Kantenmenge ist und die Relation kann als symmetrisch aufgefasst werden. Deshalb sind die zugehörigen Graphen von einem anderen Typ, eben vom Typ eines ungerichteten Graphen. Präzise definieren wir dies folgendermassen:

§6-12 Definition. (Einfacher ungerichteter Graph (schlichter Graph))

Ein einfacher ungerichteter Graph G , auch schlichter Graph oder oft auch nur Graph genannt, ist ein 2-Tupel $G := (V, E)$ mit

- (i) V ist eine endliche Menge von Knoten $\{v_1, v_2, \dots, v_n\}$.
- (ii) E ist eine Menge von zweielementigen Knotenmengen

$$E := \{\{v, w\} \mid v \in V \text{ und } w \in V\}.$$

\diamond

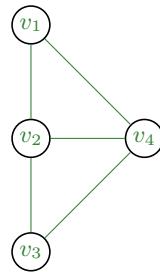
Aufgrund der Definition einer Menge (Reihenfolge der Elemente spielt keine Rolle und kein Element kann mehrfach auftreten), sehen wir unmittelbar, dass ein einfacher ungerichteter Graph keine Mehrfachkanten zwischen zwei gleichen Knotenpaaren und keine Schleifen (Kante zwischen einem Knoten und sich selbst) haben kann. Wir halten fest:

§6-13 Korollar.

Ein einfacher ungerichteter Graph hat keine Mehrfachkanten zwischen zwei Knoten und insbesondere auch keine Schleifen (Kante von einem Knoten zu sich selbst). \diamond

Wie schon erwähnt, meint man oft mit dem Wort "Graph" genauer einen ungerichteten Graphen, genau wie in §6-12 definiert. Deshalb sollte man explizit immer von "Digraphen" oder von "gerichteten Graphen" sprechen, wenn die Konstrukte des vorhergehenden Abschnitts 6.1.1 gemeint sind.

§6-14 Beispiel. (Beispiel eines einfachen ungerichteten Graphen)



$G = (V, E)$ mit
 $V = \{v_1, v_2, v_3, v_4\}$ und
 $E = \{\{v_1, v_2\}, \{v_1, v_4\}, \{v_2, v_3\}, \{v_2, v_4\}, \{v_3, v_4\}\}.$

◊

§6-15 Beispiel. (Alltagsbeispiel eines einfachen ungerichteten Graphen)

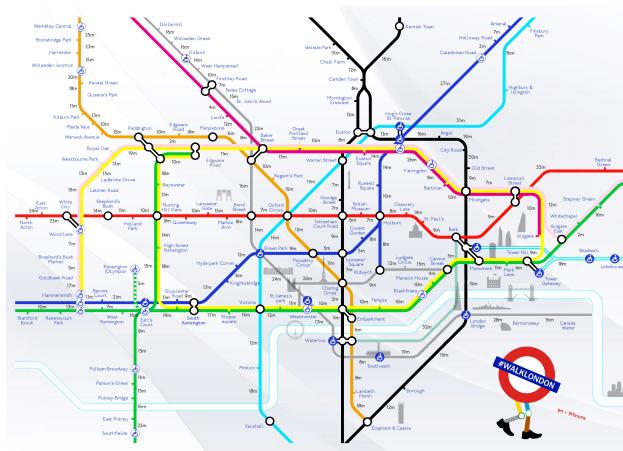
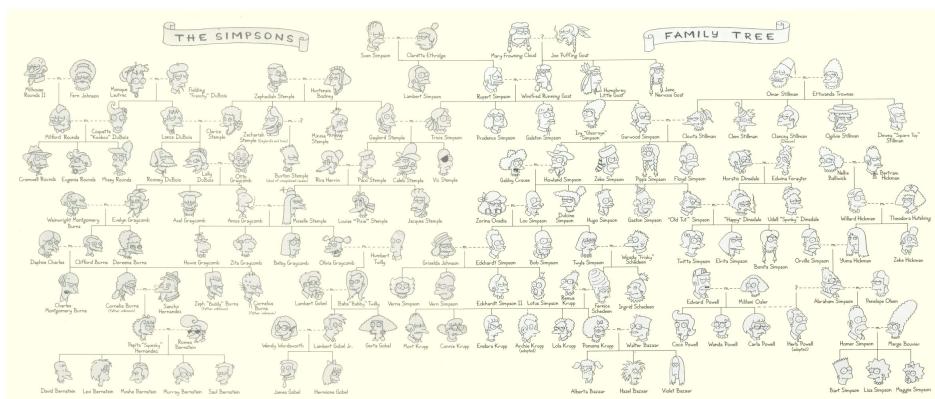


Abbildung 6.1 – aus <http://metrouk2.files.wordpress.com/2014/04/london1.jpg>

◊

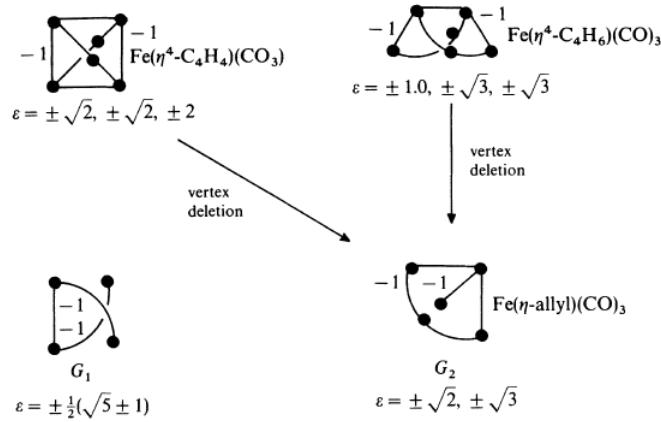
§6-16 Beispiel. (Alltagsbeispiel eines einfachen ungerichteten Graphen)



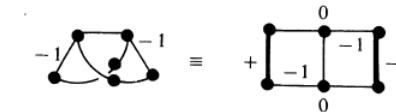
[A2]

◊

§6-17 Beispiel. (Anwendungsbeispiele von ungerichteten Graphen)



a) Embedding fragments.

b) Ethene embedding on $\text{Fe}(\eta^4\text{-C}_4\text{H}_6)(\text{CO})_3$.Fig. 3.7. Embedding fragments for $\text{Fe}(\eta^4\text{-olefin})(\text{CO})_3$ complexes

[7, Fig.3.7]

◊

Auch bei ungerichteten Graphen sind in Anwendungen und bei der Modellierung und Lösung von Problemen wieder viele mögliche spezielle Eigenschaften von entscheidender Bedeutung. Wir stellen deshalb nachfolgend wieder ein paar grundlegende Begriffe und Eigenschaften zusammen.

§6-18 Definition. (Grundbegriffe)

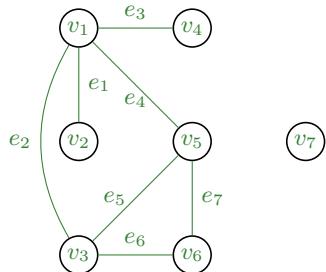
- (1) Die Anzahl $|V|$ von Knoten und die Anzahl $|E|$ von Kanten bzw. Pfeile sind natürlich auch bei ungerichteten Graphen wieder ein erstes Charakteristikum.
- (2) In einem ungerichteten Graphen können wir weiter für jeden Knoten $v \in V$ den Grad $\deg(v)$ (Knotengrad, Valenz) [E.19] definieren: Das ist die Anzahl Kanten die v als Endknoten besitzen.
- (3) Sind in einem ungerichteten Graphen zwei Knoten v und w miteinander durch eine Kante verbunden, d.h., ist $\{u, v\} \in E$, dann sagen wir u und v seien adjazent. Eine Kante $e := \{u, v\} \in E$ heisst inzident zu den Knoten u und v .

◊

Folgende Beispiele verdeutlichen wiederum diese Grundbegri e:

§6-19 Beispiele. (Grundbegri e)

Für den folgenden ungerichteten Graphen $G = (V, E)$ gilt beispielsweise



$$\begin{aligned} |V| &= 7, \\ |E| &= 7, \\ \deg(v_1) &= 4, \\ \deg(v_4) &= 1, \\ \deg(v_7) &= 0, \\ \dots \text{ usw.} \end{aligned}$$

Weiter sind z.B. wegen e_5 die Knoten v_3 und v_5 adjazent. Damit ist die Kante e_5 inzident zu v_3 und v_5 . \diamond

Folgende Begri e sind völlig analog zum Fall der gerichteten Graphen, wobei der Begri einer Schleife nicht definiert wird, da Schleifen in einem ungerichteten (einfachen) Graphen nicht vorkommen können:

§6-20 Definition. (Weg (Pfad) in einem ungerichteten Graph; Kreis (Zyklus))

- (1) Ein Weg (Pfad) in einem ungerichteten Graph $G = (V, E)$ ist eine Folge von Knoten

$$v_{i_1}, v_{i_2}, v_{i_3}, \dots, v_{i_{k-2}}, v_{i_{k-1}}, v_{i_k},$$

für ein $k \in \mathbb{N}$, so dass für alle $j \in \mathbb{N}_{k-1}$ gilt: $\{v_{i_j}, v_{i_{j+1}}\} \in E$. [E.20]

- (2) Ein Weg $v_{i_1}, v_{i_2}, v_{i_3}, \dots, v_{i_{k-2}}, v_{i_{k-1}}, v_{i_k}$ heisst Kreis (Zyklus), wenn der erste und der letzte Knoten des Weges identisch sind. Formal also, wenn

$$v_{i_1} = v_{i_k}$$

gilt. \diamond

§6-21 Beispiele.

Im Graphen von Beispiel §6-19 gilt beispielsweise:

$$v_1, v_5, v_3, v_6, v_5, v_3$$

(oder mit den Kantenlabels geschrieben: e_4, e_5, e_6, e_7, e_5) ist ein Weg (Pfad) und

$$v_4, v_1, v_5, v_1, v_4$$

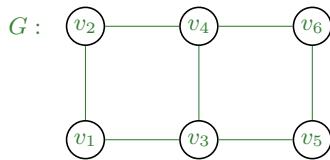
(bzw., e_3, e_4, e_4, e_3) ein Kreis (Zyklus). \diamond

Ähnlich wie im Fall gerichteter Graphen gibt es entsprechende Begri e:

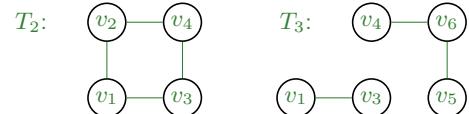
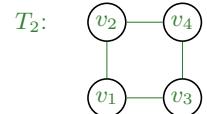
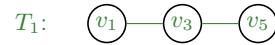
§6-22 Definition. (Teilgraph, Obergraph eines ungerichteten Graphen)

Ein Graph $G' = (V', E')$ heisst **Teilgraph** (oder **Subgraph**, **Untergraph**) eines Graphen $G = (V, E)$, falls $V' \subseteq V$ und $E' \subseteq E$ gilt. In einem solchen Fall nennen wir G dann auch den **Obergraphen** (oder **Supergraphen**) von G' . \diamond

§6-23 Beispiel. (Teilgraphen)



Teilgraphen in G sind je z.B. die folgenden drei Graphen:



Kein Teilgraph in G ist z.B.:



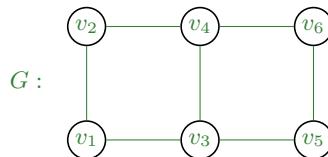
\diamond

§6-24 Definition. (Durch Knotenteilmenge induzierter Teilgraph)

Enthält ein Teilgraph $G' = (V', E')$ von $G = (V, E)$ alle Kanten des Obergraphen G zwischen denjenigen Knoten, die auch in G' liegen (also zwischen den Knoten in V'), so nennen wir den Teilgraphen G' den durch (die Knotenmenge) V' induzierten Teilgraphen von G . \diamond

§6-25 Beispiel. (Durch Knotenteilmenge induzierter Teilgraph)

Sei folgender ungerichteter Graph gegeben:

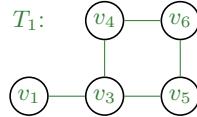


Wir betrachten die Knotenteilmenge

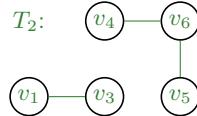
$$V' = \{v_1, v_3, v_4, v_5, v_6\},$$

also ohne den Knoten v_2 .

Ein durch V' induzierter Teilgraphen in G ist z.B.:



Kein durch V' induzierter Teilgraph in G ist z.B.:



weil die Kanten $\{v_3, v_4\}$ und $\{v_3, v_5\}$ fehlen. \diamond

Ungerichtete Graphen helfen manchmal auch bei der Beschreibung von gerichteten Graphen (wir werden beim Begriff des Zusammenhangs darauf zurückkommen, siehe Def. §6-41):

§6-26 Definition. (Ungerichteter Graph eines gerichteter Graphen)

Der ungerichtete Graph eines gerichteten Graphen ist derjenige ungerichtete Graph, welcher entsteht, wenn man die Knotenmenge beibehält und jede gerichtete Kante durch eine ungerichtete ersetzt (und Schleifen und Mehrfachkanten weglässt). \diamond

Auch bei ungerichteten Graphen gibt es viele Eigenschaften und daraus resultierende spezielle Typen von Graphen. Zum Beispiel gibt es auch hier bipartite Graphen:

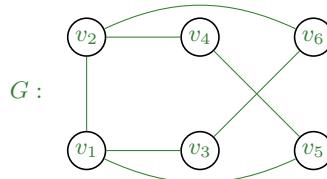
§6-27 Definition. (Bipartiter Graph)

Ein Graph $G = (V, E)$ heisst bipartit, wenn sich seine Knotenmenge V so in zwei Teilmengen A und B partitionieren lässt, dass jede Kante einen Knoten in A und einen Knoten in B hat. \diamond

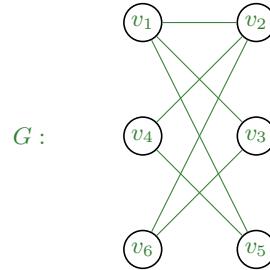
Bipartite Graphen sind also analog der bereits kennengelernten bipartiten Digraphen. Auch hierzu ein Beispiel:

§6-28 Beispiel. (Bipartiter Graph)

Der Graph



ist bipartit, weil wir mit $A = \{1, 4, 6\}$ und $B := \{2, 3, 5\}$ eine entsprechende Partitionierung der Knotenmenge haben. Dies sieht man wieder besser, wenn wir ähnlich wie bei den bipartiten Digraphen die Knoten anders anordnen:



◊

6.1.3 (Ungerichtete) Multigraphen

Ein dritter zentraler Begri ist derjenige eines (ungerichteten) Multigraphen, wofür wir später ein typisches Anwendungsbeispiel besprechen werden. Es gibt dann auch noch gerichtete Multigraphen, also "Multidigraphen", diese werden wir aber nicht näher untersuchen.

Als erstes brauchen wir folgenden zentralen Begri aus der Mengenlehre. Dabei erinnern wir uns, dass in einer Menge ein Element nur einmal vorkommen kann. Das müssen wir nun abändern.

§6-29 Definition. (Multimenge)

Eine Multimenge ist eine Menge, in welcher die Elemente auch mehrmals vorkommen können. Die Reihenfolge spielt dabei weiterhin keine Rolle. ◊

Multimengen werden (meist) wie Mengen in geschweiften Klammern geschrieben. Das mag etwas unglücklich sein, zeigt aber wiederum, dass wir sowieso immer unsere Symbole klar deklarieren und präzise festlegen müssen, wofür sie stehen.

§6-30 Beispiele.

- (1) $M := \{2, 3, 4, 3, 5, 6, 4\}$ ist eine Multimenge.
- (2) $M := \{(v_1, v_2), (v_1, v_2), (v_2, v_5), (v_2, v_6), (v_2, v_6)\}$ ist eine Multimenge von geordneten Paaren.
- (3) $M := \{\{x\}, \{x\}, \{x, y\}, \{x, z\}, \{y, z\}, \{y, z\}\}$ ist eine Multimenge von Mengen.
- (4) $M := \{\{x\}, \{x, x\}, \{x, y\}, \{x, z, z\}, \{y, z, z, z\}, \{y, z, z, z\}\}$ ist eine Multimenge von Multimengen.

◊

Mit Multimengen können wir nun unseren zentralen Begri definieren:

§6-31 Definition. ((Ungerichteter) Multigraph)

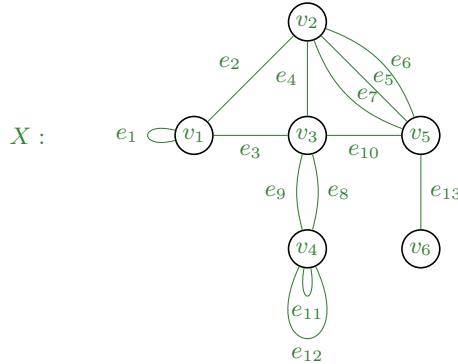
Ein (ungerichteter) Multigraph^[E.21] ist ein Paar $X = (V, E)$ bestehend aus

- einer endlichen Menge V von Knoten und
- einer Multimenge E von 2-elementigen Multi-Teilmengen $\{u, v\} \in E$, wobei $u, v \in V$ Knoten sind.

◊

Hier ist unbedingt zu beachten, dass im Gegensatz zur Definition des (einfachen) Graphen diese Definition eines Multigraphen explizit Schleifen zulässt!

§6-32 Beispiel. (Ein Multigraph)



$X = (V, E)$ mit

$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ und

$E = \{\{v_1, v_1\}, \{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}, \{v_2, v_5\}, \{v_2, v_5\}, \{v_2, v_5\},$

$\{v_3, v_4\}, \{v_3, v_4\}, \{v_3, v_5\}, \{v_4, v_4\}, \{v_4, v_4\}, \{v_5, v_6\}\}$

$= \{e_1, e_2, \dots, e_{13}\}$.

◊

§6-33 Beispiele. (Anwendungsgebiete von Multigraphen)

Es gibt viele mögliche Anwendungen von Multigraphen, da das Konzept sehr flexibel ist. Allgemein können damit (komplexere) Netzwerke modelliert werden, wie sie zum Beispiel im Gebiet der "social network analysis" (SNA), in technischen Netzwerken oder auch in Verkehrsnetzen vorkommen können. Historisch war Leonhard Euler's Anwendung von Multigraphen auf das berühmte "Königsberger Brückenproblem" (siehe Abschnitt 6.3.1) sogar quasi die "Geburtsstunde" der ganzen Graphentheorie.

◊

Viele Grundbegriffe lassen sich "nahtlos" auf Multigraphen übertragen:

§6-34 Definition. (Grundbegriffe)

- (1) Die Anzahl $|V|$ von Knoten und die Anzahl $|E|$ von Kanten bzw. Pfeile sind auch bei Multigraphen ein erstes Charakteristikum.
- (2) In einem Multigraphen können wir ebenso für jeden Knoten $v \in V$ den Grad $\deg(v)$ (Knotengrad, Valenz) definieren: Es ist wieder die Anzahl Kanten die v als Endknoten besitzen.
- (3) Sind in einem Multigraphen zwei Knoten v und w miteinander durch mindestens eine Kante verbunden, d.h., ist $\{u, v\} \in E$, dann sagen wir u und v seien adjazent. Eine Kante $e := \{u, v\} \in E$ heisst inzident zu den Knoten u und v .

◊

§6-35 Definition. (Weg (Pfad) in einem Multigraphen; Kreis (Zyklus); Schleife (Loop))

- (1) In einem Multigraphen nennen wir eine abwechselnde Folge von Knoten und Kanten der Form

$$v_{i_1}, \underbrace{\{v_{i_1}, v_{i_2}\}}_{=:e_1}, v_{i_2}, \underbrace{\{v_{i_2}, v_{i_3}\}}_{=:e_2}, v_{i_3}, \dots, v_{i_{k-1}}, \underbrace{\{v_{i_{k-1}}, v_{i_k}\}}_{=:e_{k-1}}, v_{i_k},$$

$k \in \mathbb{N}$, einen **Weg (Pfad)**. Im Allgemeinen muss man in Multigraphen die Kanten(labels) zur Beschreibung eines Weges verwenden, da die Angabe einer ein- oder zweielementigen Knotenmenge für eine Kante nicht eindeutig ist.

- (2) Ein Weg in einem Multigraphen heisst **Kreis (Zyklus)**, wenn

$$v_{i_1} = v_{i_k}$$

gilt. D.h., wenn der Anfangs- und der Endknoten die gleichen sind.

- (3) Eine **Schleife (Loop)** in einem Multigraphen ist eine Kante $\{v_i, v_i\}$.

◊

§6-36 Definition. (Teilmultigraph, Obermultigraph eines Multigraphen)

Ein Multigraph $X' = (V', E')$ heisst **Teilmultigraph (oder Submultigraph, Untermultigraph)** eines Multigraphen $X = (V, E)$, falls $V' \subseteq V$ und $E' \subseteq E$ gilt. In einem solchen Fall nennen wir X dann auch den **Obermultigraphen (oder Supermultigraphen)** von X' . ◊

§6-37 Beispiele.

Im Multigraphen X von Beispiel §6-32 gilt beispielsweise:

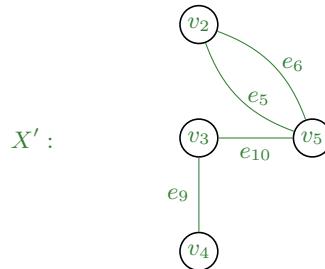
$$v_3, e_9, v_4, e_{11}, v_4, e_{11}, v_4, e_{12}, v_4, e_9, v_3, e_4, v_2$$

ist ein Weg und

$$v_5, e_7, v_2, e_6, v_5$$

ist ein Kreis. Ein Beispiel einer Schleife ist $\{v_1, v_1\}$ also e_1 .

Der Multigraph $X' = (V', E')$



definiert durch

$$\begin{aligned} V' &:= \{v_2, v_3, v_4, v_5\}, \\ E' &:= \{e_5, e_6, e_9, e_{10}\} \end{aligned}$$

ist ein Teilmultigraph von X aus §6-32. ◊

Abschliessend sei noch gesagt, dass Multigraphen manchmal – d.h., wenn es die beabsichtigte Anwendung erlaubt – als sogenannte kantengewichtete (einfache) Graphen dargestellt werden. Dabei werden zwischen zwei Knoten nicht mehrere Kanten erfasst, sondern die Kante erhält ein "Gewicht", welche der Anzahl Kanten zwischen den betre enden Knoten entspricht. Beispielsweise also bei zwei Kanten zwischen den Knoten v_1 und v_2 wird nur eine Kante $\{v_1, v_2\}$ erfasst, aber mit einem Kantengewicht $w(v_1, v_2) = 2$ versehen. Wir werden auf kantengewichtete Graphen bei Anwendungen noch zurückkommen. Jetzt sind wir jedenfalls auch mit dem Begri eines Multigraphen soweit, dass wir dann exemplarisch typische Anwendungen und Lösungsalgorithmen dafür studieren können (siehe Abschnitt 6.3.1).

6.2 Zusammenhang und Zusammenhangskomponenten

Mit Hilfe des eben eingeführten Begri s eines Weges (Pfades) lässt sich eine weitere wichtige Eigenschaft von Graphen, Multigraphen und Digraphen beschreiben. Hier ist es einfacher, zuerst mit dem Begri für den ungerichteten Graphen und den Multigraphen zu starten, weil dann damit relativ einfach der entsprechende Begri für den Digraphen-Fall definiert werden kann.

Im folgenden schreiben wir jeweils "(Multi)Graph", wenn etwas analog sowohl für ungerichtete Graphen als auch für ungerichtete Multigraphen gilt. Ebenso schreiben wir "(Di)Graphen", wenn es sowohl für gerichtete als auch ungerichtete Graphen entsprechend gilt, oder "(Multi,Di)Graphen", wenn es sogar für alle vier Varianten gilt.

§6-38 Definition. (Zusammenhängender (Multi)Graph)

Ein (Multi)Graph heisst **zusammenhängend**, wenn von jedem Knoten aus zu jedem anderen Knoten ein Weg existiert. ◊

§6-39 Definition. (Maximal zusammenhängender Teil(multi)graph, Zusammenhangskomponenten)

- (1) Ein Teil(multi)graph $T = (V_T, E_T)$ eines (Multi)Graphen $X = (V, E)$ heisst **maximal zusammenhängend**

$\iff \forall$ Teilgraphen $T' = (V_{T'}, E_{T'})$ von X , für welche T auch

Teil(multi)graph von T' ist, schon notwendigerweise $T = T'$ gilt,

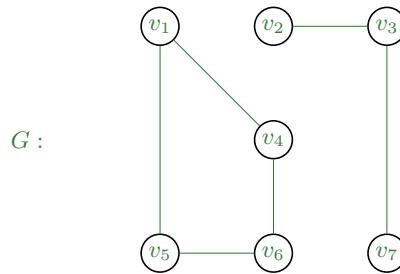
$$\begin{aligned} \iff_{\substack{(\text{ganz} \\ \text{formal})}} & \forall \text{ Teil(multi)graphen } T' = (V_{T'}, E_{T'}) : \\ & (V_{T'} \subseteq V) \wedge (E_{T'} \subseteq E) \wedge (V_T \subseteq V_{T'}) \wedge (E_T \subseteq E_{T'}) \\ \implies & (V_T = V_{T'} \wedge E_T = E_{T'}). \end{aligned}$$

- (2) Einen maximalen zusammenhängenden Teil(multi)graphen eines nicht-notwendig zusammenhängenden (Multi)Graphen X nennen wir auch **Zusammenhangskomponente** von X

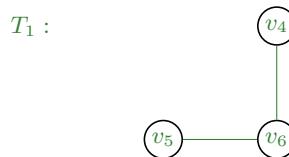
◊

§6-40 Beispiele. (Zusammenhang in (Multi)Graphen)

1. Sei folgender (nicht zusammenhängender) Graph $G = (V, E)$ gegeben:



Dann ist beispielsweise

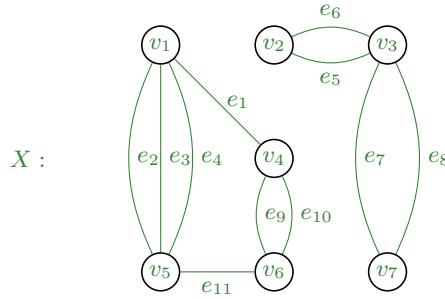


eine zusammenhängende Komponente, welche nicht maximal ist. Hingegen sind

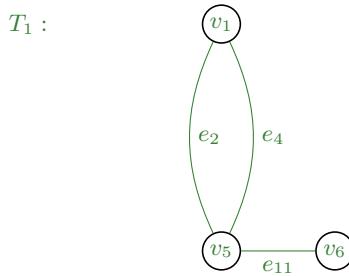


zwei maximal zusammenhängende Komponenten von G , sogar die zwei maximal zusammenhängenden Komponenten von G .

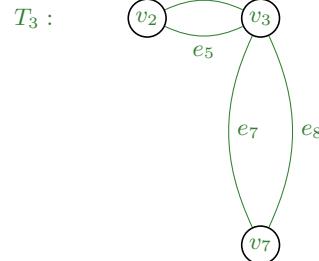
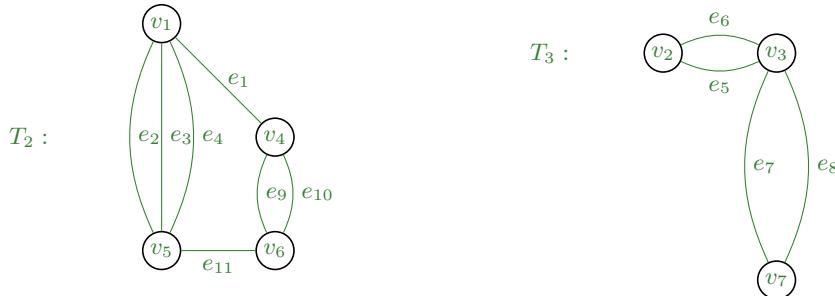
2. Sei folgender (nicht zusammenhängender) Multigraph $X = (V, E)$ gegeben:



Dann ist beispielsweise



eine zusammenhängende Komponente, welche nicht maximal ist. Hingegen sind



zwei maximal zusammenhängende Komponenten von X , sogar die zwei maximal zusammenhängenden Komponenten von X .

◇

§6-41 Definition. (Der einem gerichteten Graphen unterliegende ungerichtete Graph)

Sei $X = (V, E)$ ein Digraph.

Der X unterliegende^[E.22] ungerichtete Graph ist der Graph $G_X := (V, E')$ mit $\forall v, w \in V$:

$$\{v, w\} \in E' \iff (v, w) \in E \vee (w, v) \in E.$$

◇

Anschaulich heisst dies, dass der unterliegende ungerichtete Graph eines Digraphen dieselbe Knotenmenge besitzt und genau dann eine ungerichtete Kante zwischen zwei Knoten v und w hat, wenn es einen Pfeil irgendeiner Richtung zwischen v und w in X gibt. [E.23]

Damit können wir nun den Begri des Zusammenhangs auch für gerichtete Graph leicht definieren, wobei wir nun zwei Versionen des Begri s erhalten:

§6-42 Definition. (Stark und schwach zusammenhängender Digraph, starke und schwache Zusammenhangskomponenten)

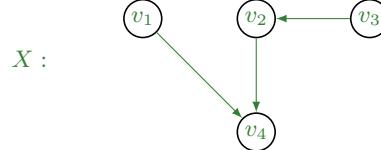
- (1) Ein Digraph $X = (V, E)$ heisst **stark zusammenhängend**, wenn für je zwei beliebige verschiedene Knoten v und w gilt: Es gibt einen Pfad von v nach w .
- (2) Ein Digraph $X = (V, E)$ heisst **schwach zusammenhängend**, wenn sein zugehöriger ungerichteter Graph G_X zusammenhängend ist.

◊

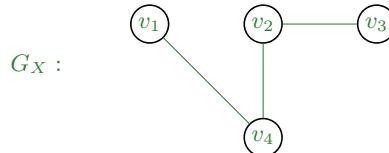
Es ist o ensichtlich, dass ein stark zusammenhängender Digraph immer auch ein schwach zusammenhängender ist. Das Gegenteil muss aber nicht gelten, wie folgendes Beispiel zeigt:

§6-43 Beispiele. (Schwach aber nicht stark zusammenhängender Digraph)

Sei folgender Digraph gegeben:



Dann ist der unterliegende ungerichtete Graph gegeben durch



G_X ist o ensichtlich zusammenhängend und damit ist X schwach zusammenhängend. X ist jedoch *nicht stark zusammenhängend*: Beispielsweise gibt es keinen (gerichteten) Weg von v_1 nach v_2 . ◊

6.3 Modellierung von Anwendungsproblemen mit Graphen

Um die Modellierung von Anwendungsproblemen mittels Graphen zu studieren, halten wir zuerst fest, was wir mit "mathematischer Modellierung eines Anwendungsproblems" genau meinen. Es ist das folgende allgemeine Prinzip der mathematischen Modellierung von Anwendungsproblemen:

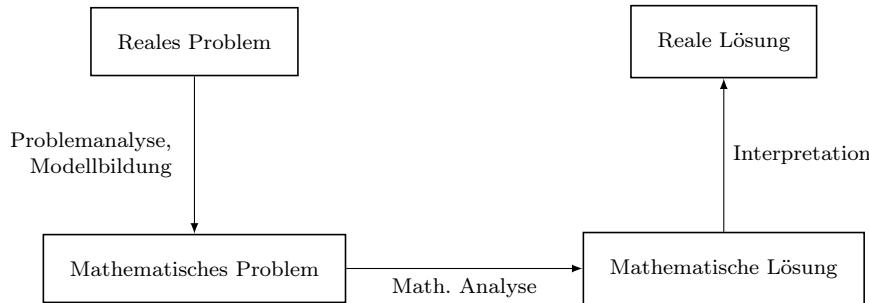


Abbildung 6.2 – Allgemeines Prinzip der mathematischen Modellierung.

Ausgehend von einem realen Problem bauen wir ein mathematisches Modell – d.h., eine rein mathematische Formulierung – des Problems und gelangen so zu einem mathematischen Problem. Bei diesem Schritt der Modellbildung ist der Aspekt der Abstraktion wesentlich: Es sollen nur diejenigen Aspekte modelliert werden, welche für die Lösung des Problems relevant sind. Alles andere soll weggelassen werden. Mit einer (korrekten) mathematischen Modellierung geht also auch eine vertiefte Analyse des realen Problems einher.

Wenn wir nun ein mathematisches Problem vorliegen haben, dann können wir dieses mathematisch analysieren und alle verfügbare und notwendige Mathematik benutzen, um es mathematisch zu lösen. Wenn wir eine mathematische Lösung gefunden haben, dann sind wir aber noch nicht fertig: Wir müssen die mathematische Lösung im Kontext des realen Problems interpretieren und zudem evaluieren, ob bzw. inwiefern sie tatsächlich eine (die) Lösung des realen Problems ist. Dieser letzte Schritt ist sehr wesentlich und darf nicht vernachlässigt werden. Denn er zeigt, ob wir das reale Problem korrekt mathematisch abstrahiert haben ("mathematisiert haben") und auch, ob zum Beispiel alle mathematischen Lösungen einer sinnvollen, realen Lösung entsprechen. (Es ist zum Beispiel durchaus möglich, dass man bei der Modellierung eines thermodynamischen Problems eine negative oder komplexe Zahl als Temperatur in Kelvin erhält. Im Interpretationsschritt muss man also unter Umständen noch die für die Realität relevante Lösung suchen und bestimmen.)

6.3.1 Die Lösung des Königsberger Brückenproblems mittels Multigraphen

Historisch gesehen war das sogenannte "Königsberger Brückenproblem", welches vom Schweizer Mathematiker Leonhard Euler (1707–1783) untersucht wurde, die Geburtsstunde der Graphentheorie. Zu seiner Lösung musste Euler den allgemeineren Begriff eines "Multigraphen", wie wir ihn in §6-31 definiert haben, verwenden. Zum besseren Verständnis der Lösung dieses Problems, brauchen wir noch zwei, auch sonst in der Mathematik sehr zentrale, Begriffe. Diese besprechen wir zuerst als kleiner Exkurs im folgenden Abschnitt.

6.3.1.1 Kurzer Exkurs: Notwendige und hinreichende Bedingung

§6-44 Definition. (Notwendige Bedingung, hinreichende Bedingung)

- (1) Es gelte $A \implies B$ für zwei Aussagen A und B .

Dann sagen wir A ist eine **hinreichende Bedingung** für B und B ist eine **notwendige Bedingung** für A .

- (2) Gilt sogar $A \iff B$, dann sagen wir, dass A eine **hinreichende und notwendige Bedingung** ist für die Aussage B . (Dasselbe dann natürlich auch für A und B vertauscht.)

◊

Ein kleines Beispiel soll die Anwendung dieser Begriffe illustrieren:

§6-45 Beispiele. (Notwendige und hinreichende Bedingung)

- (1) Wenn A die Aussage ist $A := \text{"Es wurden } x > 0 \text{ Franken auf das Konto } y \text{ eingezahlt"}$ ist und B die Aussage $B := \text{"Der Kontostand von Konto } y \text{ hat sich erhöht"}$, dann gilt offensichtlich $A \implies B$.

$\rightsquigarrow A$ ist dann eine hinreichende Bedingung für B . Um den Kontostand zu erhöhen "reicht es" sozusagen aus, einen Betrag $x > 0$ einzuzahlen. Aber A ist nicht notwendige Bedingung, denn der Kontostand kann sich auch durch eine interne Zinsvergütung erhöhen, oder meinetwegen wegen eines Hackerangriffs des Kontoinhabers ;-).

$\rightsquigarrow B$ ist notwendige Bedingung für A . Denn wenn Geld einbezahlt wird, dann erhöht sich notwendigerweise der Kontostand. Aus demselben Grund wie oben, ist aber B nicht hinreichend dafür, dass A eingetreten ist.

- (2) Für eine natürliche Zahl $n > 1$ ist die Aussage

$$A := \text{"n ist Primzahl"}$$

äquivalent zur Aussage

$$B := \text{"n ist nur durch sich selbst und durch 1 teilbar."}.$$

Es gilt also $A \iff B$. (B ist ja sogar gerade die Definition einer Primzahl.)

Deshalb sagen wir dann " B ist hinreichend und notwendig dafür, dass A gilt.

◊

6.3.1.2 Das Königsberger Brückenproblem

Leonhard Euler soll sich im frühen 18. Jahrhundert folgende Fragen anlässlich eines Spaziergangs durch Königsberg (russ. Kaliningrad; siehe historischer Stadtplan unten) gestellt haben:

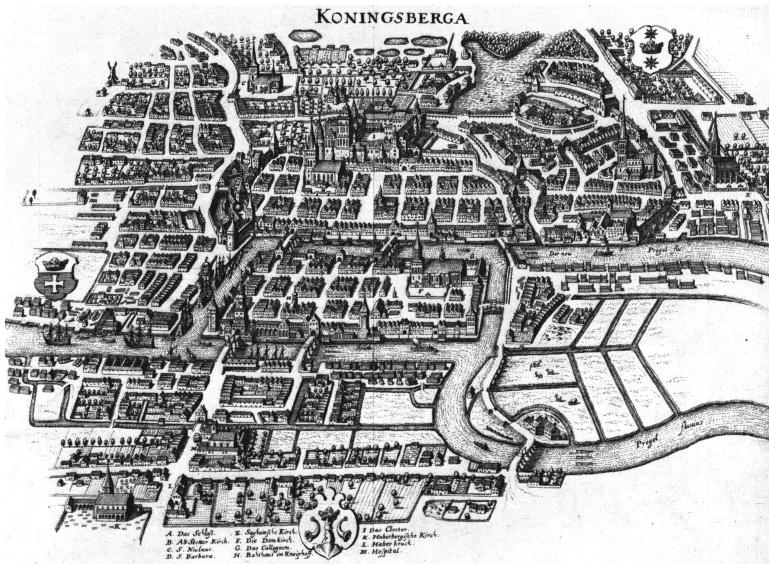
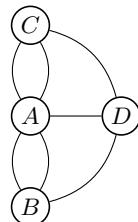


Abbildung 6.3 – Königsberg (russ. Kaliningrad), 1651.

[A3]

- Gibt es einen Weg, bei dem alle sieben Brücken über den Fluss Pregel genau einmal überquert werden?
- Wenn ja, gibt es auch einen Weg, bei dem man am Schluss wieder am Ausgangspunkt landet?

Euler fand im Jahr 1736 die Antwort: Ein solcher Weg ist in Königsberg nicht möglich. Die Graphentheorie zeigt uns, weshalb Euler recht hat! Als erstes stellen wir die Situation von Königsberg mit seinen Brücken über den Pregel als Graph (genauer als "Multigraph") dar (dies ist die Abstrahierung bzw. "Mathematisierung" im Vorgehen der mathematischen Modellierung):



Die Knoten sind die verschiedenen Stadtteile von Königsberg und die Kanten entsprechen den Brücken. Wenn wir die Aufgabenstellung nochmals genau studieren, dann sehen wir, dass noch etwas Wichtiges mehr gefordert ist als nur einen Weg oder einen Kreis zu finden: Eine Brücke, d.h., eine Kante, muss genau einmal durchlaufen werden, damit das Problem als gelöst gilt. Historisch werden solche Wege und Kreise nach Leonhard Euler benannt und wir definieren sie folgendermassen:

§6-46 Definition. (Eulerwege und Eulerkreise in einem (Multi)Graphen; Eulergraph)

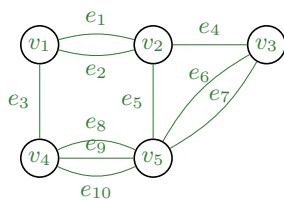
- (1) Wird bei einem Weg jede Kante (!) des (Multi)Graphen genau einmal besucht,

so heisst der Weg Eulerweg.

- (2) Einen Eulerweg in einem (Multi)Graphen, der auch Kreis ist, nennen wir Eulerkreis.
- (3) Ein (Multi)Graph welcher einen Eulerkreis besitzt, heisst Eulergraph.^[E.24]

◇

§6-47 Beispiel. (Wege und Eulerwege in einem Multigraph)



$X = (V, E)$ mit
 $V = \{v_1, v_2, v_3, v_4, v_5\}$ und
 $E = \{e_1, e_2, \dots, e_{10}\}$.

Wege sind z.B.:

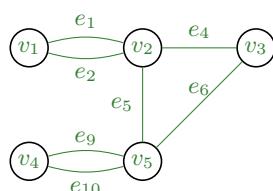
- $v_1, e_1, v_2,$
- $v_5, e_7, v_3, e_6, v_5, e_{10}, v_4, e_3, v_1, e_3, v_4, e_8, v_5, e_6, v_3,$
- aber auch:
 $v_5, e_8, v_4, e_8, v_5, e_8, v_4, e_8, v_5$
 (ein etwas spezieller Weg...)

Ein Eulerweg ist z.B.:

- $v_3, e_6, v_5, e_7, v_3, e_4, v_2, e_5, v_5, e_8, v_4, e_{10}, v_5, e_9, v_4, e_3, v_1, e_1, v_2, e_2, v_1.$

◇

§6-48 Beispiel. (Kreis und Eulerkreis in einem Multigraph)



Sei $X = (V, E)$ wie vorher in §6-47,
jetzt aber ohne die Kanten ^[E.25]

$$\begin{aligned} e_3 &= \{v_1, v_4\}, \\ e_7 &= \{v_3, v_5\}, \\ e_8 &= \{v_2, v_5\}. \end{aligned}$$

Kreise sind dann zum Beispiel:

- $v_2, e_5, v_5, e_6, v_3, e_4, v_2,$
- etwas weniger typisch "kreisförmig":
 $v_1, e_2, v_2, e_4, v_3, e_4, v_2, e_2, v_1.$

Ein Eulerkreis ist z.B.:

- $v_1, e_1, v_2, e_4, v_3, e_6, v_5, e_9, v_4, e_{10}, e_5, v_2, e_2, v_1.$

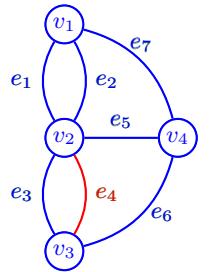
◇

Jetzt haben wir die notwendigen Begriffe exakt beisammen um die Lösung des Problems in Angrip zu nehmen. Das zugehörige zu lösende mathematische Problem ist also die Frage, ob es einen Eulerweg bzw. sogar einen Eulerkreis in diesem Multgraphen gibt. Untersuchen wir dies genauer!

6.3.1.3 Lösung des Königsberger Brückenproblems; Eulerbedingungen in (Multi)Graphen

Am besten lernen wir das Problem noch etwas genauer kennen, indem wir einen Versuch wagen, einen Eulerweg zu finden:

§6-49 Beispiel. (Versuch eines Weges, der jede der sieben Brücken Königbergs genau einmal benutzt)



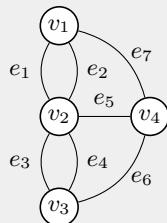
Versuch: $v_1, e_1, v_2, e_3, v_3, e_6, v_4, e_5, v_2, e_2, v_1, e_7, v_4$.

◇

Die Kante (= Brücke) e_4 konnte im Beispiel nicht überquert werden. Jedoch sind wir ja jetzt willkürlich vorgegangen. Wie können wir eine andere mögliche Lösung (alle möglichen anderen Wege) ausschliessen? Wir sehen: das Problem ist also etwas verzwickter als vielleicht zuerst gedacht. Heute kann man folgenden Satz zeigen, welcher die Frage des Königsberger Brückenproblems beantwortet:

§6-50 Theorem. (Königsberger Brückenproblem ist unlösbar)

Im Multigraphen welcher Königsberg repräsentiert, also



gibt es keinen Eulerweg und insbesondere damit auch keinen Eulerkreis.

◇

Beweis.

- Nennen wir einen Knoten eines Weges, welcher nicht End- oder Anfangsknoten ist, einen Zwischenknoten.
- Für jeden Eulerweg gilt nun: Führt ein Weg über eine bisher unbenutzte Kante in einen beliebigen Zwischenknoten, so muss zu dieser hineinführenden Kante immer auch eine bisher unbenutzte herausführende Kante gehören.
- D.h. aber, dass ein Zwischenknoten immer nur eine gerade Anzahl Kanten haben kann.
- Im Königsberger Multigraphen, hat aber jeder Knoten eine ungerade Anzahl von Kanten. Somit können diese alle nicht Zwischenknoten eines Weges sein.

- Deshalb existiert in diesem Multigraphen kein Eulerweg und damit auch kein Eulerkreis.

□

Natürlich will man nach einem solchen Ergebnis mehr über das allgemeine Problem wissen. Das heisst, was lässt sich für allgemeine bzw. für beliebige Multigraphen (also nicht nur für diesen spezifischen, zu Königsberg gehörenden Multigraphen) über das Eulerweg- und das Eulerkreisproblem sagen? Also quasi, wie sieht die Antwort "für alle anderen möglichen Städte" aus? Schauen wir uns diese Antwort im folgenden Abschnitt an.

6.3.1.4 Eulerwege und Eulerkreise in allgemeinen (Multi)Graphen

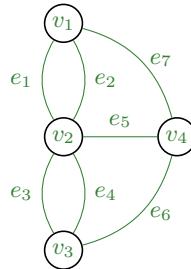
Bei der allgemeinen Problemstellung stellen sich grundsätzlich zwei zentrale Fragen:

- (1) Wann hat das Problem für einen gegebenen Graphen überhaupt eine Lösung bzw. gibt es ein Kriterium mit welchem man eine Lösung garantieren kann?
- (2) Wenn das Problem für einen gegebenen Graphen lösbar ist, wie lösen wir es dann? D.h., mit welchem Verfahren (Algorithmus) finde ich einen Eulerweg bzw. einen Eulerkreis?

Einen ersten Hinweis darauf, was uns für die allgemeine Situation weiterhilft, erhalten wir, wenn wir uns noch einmal das Königsberger Beispiel und darin die Knotengrade (siehe Def. §6-34) anschauen:

§6-51 Beispiel. (Knotengrade im Königsberger Multigraph)

Für Königsberg's Graph ("Multigraph")



gilt offensichtlich:

$$\deg(v_1) = 3$$

$$\deg(v_2) = 5$$

$$\deg(v_3) = 3$$

$$\deg(v_4) = 3$$

◇

Wir können uns bezüglich der Knotengrade überlegen, dass die Anzahl Knoten mit ungeradem Grad nicht beliebig sein kann. Warum? Jede Kante welche "irgendwo" bei einem Knoten "angehängt ist", wir sagen auch inzident zu einem Knoten ist, ist genau noch ein weiteres Mal inzident zu einem Knoten (dies kann ein anderer Knoten oder im Fall einer Schleife derselbe sein). D.h., ein Grad mehr bei einem Knoten durch Hinzufügen einer Kante führt immer zwangsläufig zu einem zweiten, weiteren zusätzlichen Kantengrad, insgesamt also zu zwei zusätzlichen Knotengraden. Dies führt unter anderem auf folgenden wichtigen Satz der Graphentheorie über die mögliche Anzahl von Knoten mit ungeradem Knotengrad:

§6-52 Satz. (Gerade Anzahl Knoten mit ungeradem Knotengrad in einem (Multi)graphen)

In einem Graphen oder Multigraphen $X = (V, E)$ kann es nur eine gerade Anzahl von Knoten mit ungeradem Grad haben. \diamond

Beweis.

- Jede Kante hat offensichtlich zwei Knoten an ihren Enden. (Eine Kante ist ja definiert als geordnetes Paar zweier Knoten.)
- Also ist die Summe aller Grade in einem (Multi)Graphen zweimal die Anzahl Kanten, d.h.,

$$2|E|.$$

$|E|$ bezeichnet ja die Mächtigkeit der Kantenmenge, also die Anzahl Kanten.

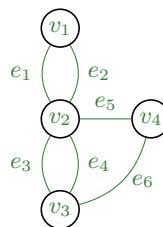
(Überlegen Sie sich: jede Kante trägt ja gerade die Zahl 2 zu dieser Summe bei, weil die beiden Knoten einer bestimmten Kante bzgl. dieser Kante je einmal den Grad 1 erhalten.)

- $2|E|$ ist aber immer eine gerade Zahl (das Doppelte jeder Zahl ist gerade), also ist diese Summe aller Grade auch gerade.
- Wenn die Summe aller Grade gerade ist, heißt das, dass es zu jedem Knoten mit ungeradem Grad einen anderen Knoten geben muss, der ebenfalls ungeraden Grad hat. Sonst könnte ja obige Summe ungerade sein.
- Damit aber gibt es immer eine gerade Anzahl von Knoten mit ungeradem Grad, was die zu zeigende Behauptung ist.

\square

Frage: Existiert im Königsberger Multigraph ohne die Kante e_7 (siehe folgendes Beispiel) ein Eulerweg?

§6-53 Beispiel. (Königsberger Multigraph ohne Kante e_7)



\diamond

Ja! Wir sehen, dass wenn wir in einem Knoten mit ungeradem Grad starten und in einem mit ungeradem Grad enden

$$v_2, e_3, v_3, e_6, v_4, e_5, v_2, e_2, v_1, e_1, v_2, e_4, v_3,$$

wir tatsächlich einen Eulerweg finden! Und, wenn wir genau hinschauen, sehen wir, dass der Multigraph nur zwei Knoten mit ungeradem Grad hat. Dies gilt nun sogar allgemein und man kann zeigen:

§6-54 Theorem. (Notwendige Eulerbedingung für Eulerwege)

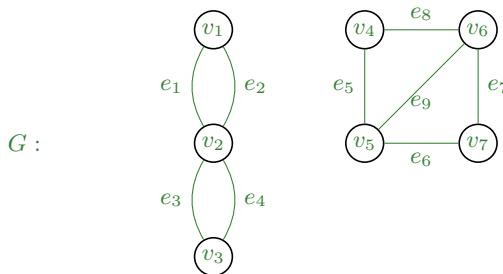
Existiert in einem (Multi)Graph ohne isolierte Knoten^[E.26] ein Eulerweg, dann haben notwendigerweise alle Knoten bis auf höchstens zwei (also bis auf Null oder zwei gemäss vorhergehendem Satz!) einen geraden Grad. Also haben entweder keiner oder 2 Knoten einen ungeraden Grad! \diamond

Weiter überlegen wir uns leicht, dass wenn der (Multi)Graph ohne isolierte Knoten zwei Knoten mit ungeradem Grad besitzt, dann muss der Eulerweg in einem von diesen beiden starten und im anderen enden.

Frage: Wenn die notwendige Eulerbedingung für Eulerwege in einem (Multi)Graphen erfüllt ist, existiert dann ein Eulerweg? D.h., ist diese notwendige Eulerbedingung auch hinreichend? Nein! Das zeigt folgendes Beispiel im Fall eines Multigraphen:

§6-55 Beispiel.

Sei folgender Multigraph $X := (V, E)$ mit $V := \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ und $E := \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9\}$ gegeben (als *einen Graphen zu betrachten!*):



Dann erfüllt X die notwendige Eulerbedingung, hat aber trotzdem keinen Eulerweg. \diamond

Wir haben offensichtlich eine wichtige Eigenschaft vergessen, die auch noch gelten muss: Der (Multi)Graph muss natürlich auch zusammenhängend sein, denn das haben wir ja nicht explizit vorausgesetzt. Damit erhalten wir aber nun:

§6-56 Theorem. (Notwendige und hinreichende Eulerbedingung für Eulerwege)

Ein (Multi)Graph ohne isolierte Knoten hat genau dann einen Eulerweg, wenn er zusammenhängend ist und die notwendige Eulerbedingung erfüllt ist, also entweder keinen oder zwei Knoten mit ungeradem Grad hat. \diamond

Beweis. Einen konstruktiven Beweis liefert der Algorithmus von Hierholzer, welchen wir nun anschliessend besprechen. \square

Mit folgendem Algorithmus von Hierholzer haben wir einen Algorithmus zur Verfügung, welcher in jedem (Multi)Graphen, der die notwendige und hinreichende Eulerbedingung erfüllt, einen Eulerweg findet.

§6-57 Satz. (Algorithmus von Hierholzer zur Berechnung eines Eulerwegs)

Input: Zusammenhängender Graph oder Multigraph ohne isolierte Knoten und mit maximal zwei Knoten von ungeradem Grad.

- (1) Wähle einen beliebigen Startknoten (falls es Knoten von ungeradem Grad gibt, wähle einen solchen als Startknoten)
- (2) Gehe von diesem Knoten aus über noch nicht besuchte Kanten so lange weiter, bis du nicht mehr weiterkommst.
- (3) Sind schon alle Kanten einmal besucht worden?

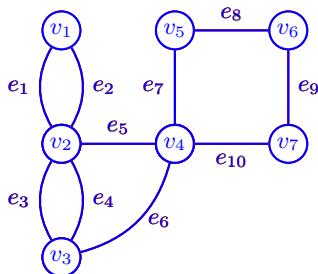
Ja: Wir sind fertig.

Nein: Wähle einen beliebigen Knoten mit noch unbesuchten Kanten auf dem bisherigen Weg als neuen Startknoten und gehe zu Schritt (2).

Output: Alle Wege an den entsprechenden Knoten zusammengefügt ergeben einen Eulerweg.

◊

§6-58 Beispiel. (Berechnung eines Eulerwegs mittels Hierholzer-Algorithmus)



1. Teil des Eulerwegs:

$v_3, e_6, v_4, e_5, v_2, e_2, v_1, e_1, v_2, e_3, v_3, e_4, v_2$

2. Teil des Eulerwegs:

$v_4, e_7, v_5, e_8, v_6, e_9, v_7, e_{10}, v_4$

Gesamter Eulerweg:

$v_3, e_6, v_4, e_7, v_5, e_8, v_6, e_9, v_7, e_{10}, v_4, e_5, v_2, e_2, v_1, e_1, v_2, e_3, v_3, e_4, v_2$

◊

Abschliessend fragen wir noch nach der allgemeinen Lösung für die Existenz eines Eulerkreises in einem (Multi)Graphen. Also der Frage, wann ein gegebener (Multi)Graph sogar ein Euler(multi)graph ist. Auch diese Frage ist in der Praxis oft von Bedeutung:

§6-59 Beispiel. (Motivierung: Optimale Routenplanung der Müllabfuhr)

Idealerweise würde die Müllabfuhr jede Strasse in einer Stadt genau einmal abfahren und wieder an den Anfangspunkt zurückkehren. Das entspricht genau einem Eulerkreis! ◊

Allerdings muss man sich für reale Situationen Folgendes bewusst sein: Meistens haben in der Praxis Graphen, welche z.B. aus Stadtplänen (Kreuzungen (= Knoten), Strassen (= Kanten)) modelliert wurden, keine Eulerkreise. Deshalb ist die Müllabfuhr in obigem Beispiel gezwungen, gewisse Strassen mehrmals abzufahren (z. B. bei Sackgassen). Hier haben wir es dann mit einem sogenannten Optimierungsproblem zu tun (ist aber nicht Thema in diesem Modul!): Die Müllabfuhr möchte nämlich möglichst wenige Strassen doppelt abfahren und möglichst jene Strassen dafür wählen, die "wenig kosten" (also z. B. kurz sind). Das heisst: Man möchte also einen Weg finden, der zumindest "möglichst nahe" an einem Eulerkreis ist und dann auch noch minimale Gesamtlänge hat. Das heisst, man such einen

Weg der "fast" (d.h., "approximativ") ein Eulerkreis ist.

Aber zurück zur Frage nach der Existenz eines "exakten" Eulerkreises. Das zentrale Kriterium für Eulerkreise folgt nämlich relativ leicht aus dem Fall für Eulerwege:

§6-60 Theorem. (Notwendige und hinreichende Eulerbedingung für Eulerkreise (bzw. für einen Euler(multi)graphen))

Ein (Multi)Graph ohne isolierte Knoten hat einen Eulerkreis (d.h., ist ein Euler(multi)graph) genau dann, wenn er zusammenhängend ist und wenn jeder Knoten geraden Grad hat. ◇

Beweis. Das folgt unmittelbar aus §6-52 und §6-56, da im Fall eines Eulerkreises Anfangsknoten und Endknoten, welche als einzige ungeraden Grad haben könnten, zusammenfallen und damit auch geraden Grad haben müssen! □

Mit dem Algorithmus von Hierholzer für Eulerwege lassen sich damit insbesondere "automatisch" auch Eulerkreise bestimmen.

Wegen Satz §6-60 wissen wir jetzt auch, wieso wir die eine Kante im Beispiel §6-53 und die drei Kanten im Beispiel §6-48 rausnehmen mussten. Letzteres Beispiel liefert uns zudem ein Beispiel für einen Eulergraphen:

§6-61 Beispiel. (Beispiel Eulergraph)

Der Multigraph im Beispiel §6-48 ist ein Eulergraph. ◇

6.3.2 Bäume, Spannbäume und typische Anwendungen

6.3.2.1 Bäume

Ein Baum ist ein spezieller Typ eines ungerichteten Graphen, welcher oft in verschiedenen Anwendungen vorkommt. Schauen wir uns ein solches einfaches Anwendungsbeispiel an:

§6-62 Beispiel. (Brückenbau zwischen Inseln und Festland [E.27])

Wir betrachten folgende Inseln und ihr zugehöriges Festland, welche durch eine Anzahl Fährverbindungen miteinander verbunden sind:

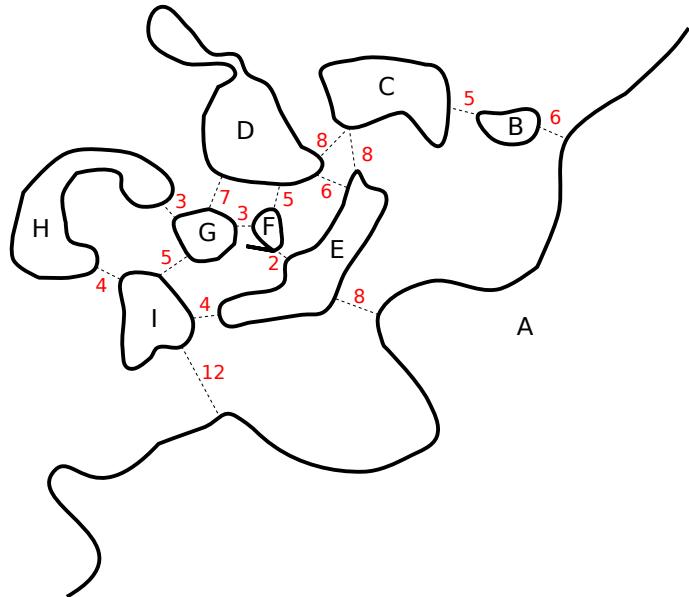


Abbildung 6.4 – Festland, Inseln und Fährverbindungen (in km).

Ein Infrastruktur-Projekt will jetzt die Fährverbindungen durch Brücken ersetzen. Aus Kostengründen soll dabei sowohl die Anzahl der Brücken als auch die Gesamtlänge der Brücken zusammen minimal sein. Wo müssen die Brücken gebaut werden um dieses Ziel zu erreichen?

◊

Für dieses typische Optimierungsproblem liefert die Graphentheorie wiederum eine geeignete Lösungsmethode. Die Aufgabenstellung erfordert aber ein paar neue Begriffe, unter anderem eben den eines Baumes.

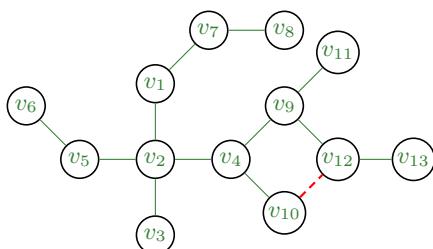
§6-63 Definition. (Baum)

Ein ungerichteter Graph $G = (V, E)$ heisst **Baum**, wenn er

- zusammenhängend ist und
- keine Kreise enthält.

◊

§6-64 Beispiel. (Baum)



Der abgebildete Graph ist ein Baum. Mit der roten Kante würde der Graph zum Beispiel zusätzlich einen Kreis $v_{10}, v_4, v_9, v_{12}, v_{10}$ beinhalten. Damit wäre er dann aber kein Baum mehr.

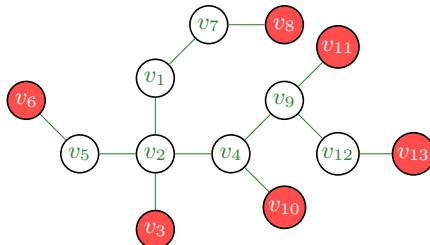
◊

§6-65 Definition. (Blatt, innere Knoten)

In einem Baum heisst ein Knoten vom Grad 1 ein **Blatt**. Ein Knoten, welcher nicht Blatt ist, heisst **innerer Knoten**.

◊

§6-66 Beispiel. (Blätter und innere Knoten in einem Baum)



rote Knoten = Blätter, andere Knoten = innere Knoten.

◊

Ein paar (erste, einfache) Sätze über Bäume:

§6-67 Satz und Definition. (Sätze über Bäume)

- (1) In einem Baum sind je zwei Knoten immer durch genau einen Weg verbunden.
- (2) Ein Baum mit n Knoten hat $n - 1$ Kanten.
- (3) Jeder Baum mit mindestens zwei Knoten besitzt mindestens zwei Blätter.

◊

Schliesslich können wir Bäume mittels der folgenden zwei Begriffe vollständig charakterisieren:

§6-68 Definition. (Minimal zusammenhängende und maximal kreisfreie Bäume)

- (1) Ein zusammenhängender Graph heisst **minimal zusammenhängend**, wenn jedes Entfernen einer Kante ihn unzusammenhängend macht.
- (2) Ein kreisfreier Graph heisst **maximal kreisfrei**, wenn jedes Hinzufügen von Kanten dazu führt, dass er nicht mehr kreisfrei ist.

◊

Die vollständige Charakterisierung von Bäumen lautet dann:

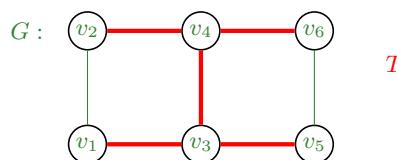
§6-69 Satz. (Vollständige Charakterisierung von Bäumen)

Ein Baum ist genau ein minimal zusammenhängender und maximal kreisfreier Graph. \diamond

Wir sehen, dass wir bei der Aufgabenstellung von Beispiel §6-62 einen Baum suchen, welcher die Knoten (=Inseln und das Festland) verbindet. Denn der Graph soll zusammenhängend sein (jede Insel soll über Brücken erreichbar sein) und die Anzahl Brücken soll minimal sein (d.h., es soll kein Kreis im Graphen bestehen). Das ist aber noch nicht alles: Wir müssen auch noch die Länge der Brücken modellieren und dann eine Konfiguration mit minimaler Gesamtlänge finden. Das führt auf den Begriff des Spannbaums eines kantengewichteten Graphen, welchen wir in den zwei nächsten Abschnitten einführen.

6.3.2.2 Spannbäume**§6-70 Definition. (Spannbaum (aufspannender Baum) eines Graphen)**

Sei $G = (V, E)$ ein zusammenhängender Graph. Ein Teilgraph T heisst Spannbaum (oder aufspannender Baum) von G , falls T ein Baum ist und alle Knoten von G enthält. \diamond

§6-71 Beispiel. (Spannbaum eines (zusammenhängenden) Graphen) \diamond

Ein einfacher Graph heisst vollständig, wenn jeder Knoten mit jedem anderen Knoten durch eine Kante verbunden ist. Die Anzahl Spannbäume in einem solchen vollständigen Graphen können wir unmittelbar ausrechnen:

§6-72 Satz und Definition. (Cayley-Formel für die Anzahl Spannbäume eines vollständigen Graphen)

Die Anzahl Spannbäume eines vollständigen Graphen mit n Knoten beträgt

$$n^{n-2}$$

 \diamond

Die Cayley-Formel gibt uns also eine prinzipielle obere Grenze für die Anzahl möglicher Spannbäume in einem einfachen Graphen.

6.3.2.3 Spannbäume kantengewichteter Graphen

Um aber in Anwendungsbeispiel §6-62 auch die Distanzen zwischen den einzelnen Landmassen modellieren zu können, müssen wir die einzelnen Kanten im Graphen "gewichten". Dies führt auf den Begriff eines kantengewichteten Graphen:

§6-73 Definition. (Kantengewichtsfunktion, Kosten einer Kante, kantengewichteter Graph)

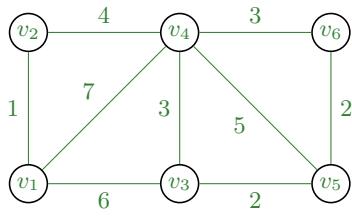
Sei $G = (V, E)$ ein Graph. Eine Abbildung

$$w : E \rightarrow \mathbb{R},$$

welche jeder Kante ein Gewicht (eine reelle Zahl) zuordnet, heißt **Kantengewichtsfunktion**. Das Gewicht einer Kante wird auch **Kosten der Kante** genannt.

Ein Graph $G = (V, E)$ mit einer Gewichtsfunktion $w : E \rightarrow \mathbb{R}$ heißt **ein kantengewichteter Graph**. Wir schreiben dann oft $G = (V, E, w)$. \diamond

§6-74 Beispiel. (Kantengewichteter Graph mit einer Kantengewichtsfunktion)



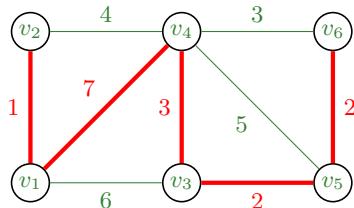
$$\begin{aligned} w : E &\rightarrow \mathbb{R} \text{ mit} \\ w(\{v_1, v_2\}) &= 1 \\ w(\{v_1, v_4\}) &= 7 \\ w(\{v_1, v_3\}) &= 6 \\ w(\{v_2, v_4\}) &= 4 \\ w(\{v_3, v_5\}) &= 2 \\ &\vdots \text{ etc.} \end{aligned}$$

\diamond

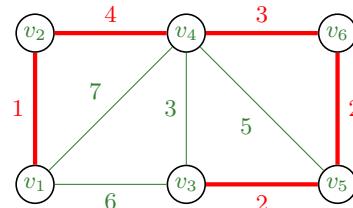
§6-75 Definition. (Minimaler Spannbaum, Kosten eines Spannbaums)

Ein Spannbaum T in einem kantengewichteten Graphen $G = (V, E, w)$ heißt **minimaler Spannbaum** (von G), wenn die Summe der Kantengewichte minimal ist. Wir nennen die Summe der Kantengewichte die **Kosten des Spannbaums**. \diamond

§6-76 Beispiel. (Kosten von Spannbäumen in kantengewichteten Graphen)



Kosten des Spannbaums:
 $1 + 7 + 3 + 2 + 2 = 15$



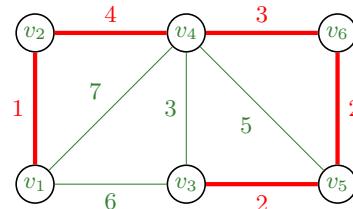
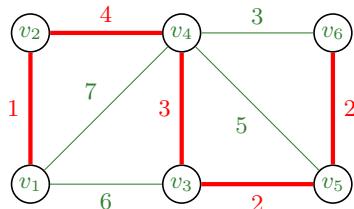
Kosten des Spannbaums:
 $1 + 4 + 3 + 2 + 2 = 12$

◇

Beachten Sie, dass auch der minimale Spannbaum nicht eindeutig sein muss:

§6-77 Beispiel. (Nicht-Eindeutigkeit minimaler Spannbäume eines kantengewichteten Graphen)

Die folgenden Spannbäume haben beide Kosten 12:



◇

6.3.2.4 Algorithmen von Kruskal und Prim zur Lösung der Anwendungsaufgabe

Es lässt sich zeigen, dass sich das Anwendungsproblem in Beispiel §6-62 sowohl mit einem Algorithmus von Joseph Kruskal als auch einem Algorithmus von Robert C. Prim (ursprünglich eigentlich von Vojtěch Jarník entwickelt) lösen lässt.

§6-78 Algorithmus. (Algorithmus von Kruskal)

Eingabe: Kantengewichteter, zusammenhängender Graph G mit n Knoten.

Ausgabe: Minimaler Spannbaum von G .

1. Wähle eine billigste (d.h., tiefstes Kantengewicht), noch nicht markierte Kante, welche nicht in einem vorhergehenden Schritt bereits getestet wurde.
2. Falls die gewählte Kante keinen Kreis mit bereits markierten Kanten bildet, markiere sie.
3. Falls $n - 1$ Kanten markiert sind, breche ab, sonst gehe zu Schritt 1.

◇

§6-79 Algorithmus. (Algorithmus von Prim)

Eingabe: Kantengewichteter, zusammenhängender Graph G mit n Knoten.

Ausgabe: Minimaler Spannbaum von G .

1. Markiere einen beliebigen Startknoten. Betrachte diesen markierten Startknoten alleine bereits als einen trivialen Baum.

2. Markiere eine billigste (d.h., tiefstes Kantengewicht) vom bereits durch Markierung konstruierten Baum ausgehende, unmarkierte Kante, welche keinen Kreis im Baum bildet.
3. Falls $n - 1$ Kanten markiert sind, breche ab, sonst gehe zu Schritt 2.

◊

§6-80 Beispiel. (Lösung von Anwendungsbeispiel §6-62)

Lösen Sie das Anwendungsproblem §6-62 sowohl mit dem *Algorithmus von Kruskal* als auch mit dem *Algorithmus von Prim*. Überlegen Sie sich gut, wie sich die beiden Algorithmen unterscheiden. Überprüfen und vergleichen Sie schliesslich durch Summation der Kantengewichte der minimalen Spannbäume Ihr Ergebnis. ◊

6.4 Matrix-Darstellungen von Graphen

Wir kennen bisher folgende wichtige Darstellungen von Graphen und Digraphen:

- **Aufzählende Darstellung** als Paar $G = (V, E)$ einer Knotenmenge V und einer Menge E von 2-elementigen Teilmengen bzw. als Paar $X = (V, E)$ einer Knotenmenge und von 2-Tupeln von Knoten:

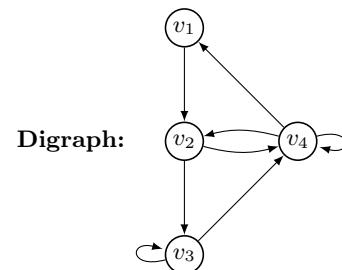
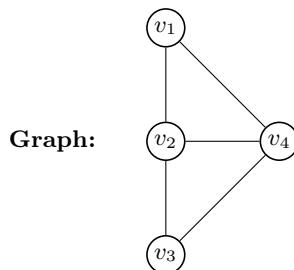
Graph:

$$\begin{aligned} G &= (V, E) \text{ mit} \\ V &= \{v_1, v_2, v_3, v_4\} \text{ und} \\ E &= \{\{v_1, v_2\}, \{v_1, v_4\}, \{v_2, v_3\}, \\ &\quad \{v_2, v_4\}, \{v_3, v_4\}\}. \end{aligned}$$

Digraph:

$$\begin{aligned} X &= (V, E) \text{ mit} \\ V &= \{v_1, v_2, v_3, v_4\} \text{ und} \\ E &= \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_4, v_2), \\ &\quad (v_3, v_3), (v_3, v_4), (v_4, v_1), (v_4, v_4)\}. \end{aligned}$$

- **Graphische Darstellung:**



Es gibt aber noch einen viel wichtigeren Typ: Matrix-Darstellungen! Dank diesen können wir (Di)Graphen mithilfe der Methoden eines der zentralsten und am weitesten entwickelten Gebiete der Mathematik untersuchen: der linearen Algebra. Eine Einführung in die "Lineare Algebra" werden Sie demnächst belegen (oder Sie haben sie schon parallel belegt). Das Ziel hier ist es, ganz kurz darzustellen, was eine "Matrix" in der Linearen Algebra ist und wie sie Graphen und Digraphen darstellen kann. Richtig einführen werden wir Matrizen erst im Modul "Lineare Algebra und Geometrie". Deshalb reicht es für uns im Moment, wenn wir definieren:

§6-81 Definition. (Matrix)

Eine Matrix (plural: Matrizen) ist ein rechteckiges Schema von Zahlen, angeordnet in m Zeilen und n Spalten, wobei $m, n \in \mathbb{N} \setminus \{0\}$ ist.

Wir sprechen dann oft auch von einer $m \times n$ -Matrix, gesprochen "m-kreuz-n-" oder "m-mal-n-Matrix" und im Fall von $m = n$ von einer quadratischen Matrix. Das Schema wird entweder mit einer runden oder einer eckigen Klammer umfasst. (Wir benutzen die Schreibweise mit den runden Klammern.) \diamond

§6-82 Beispiel.

Der Ausdruck

$$A := \begin{pmatrix} 3 & -2 & \frac{2}{3} & 5 \\ -\frac{3}{4} & 0 & -1 & 2 \\ 0 & \pi & 4 & -2 \end{pmatrix}$$

definiert eine 3×4 -Matrix A .

Dieselbe Matrix in eckigen Klammern würden wir so schreiben:

$$A := \left[\begin{array}{cccc} 3 & -2 & \frac{2}{3} & 5 \\ -\frac{3}{4} & 0 & -1 & 2 \\ 0 & \pi & 4 & -2 \end{array} \right].$$

\diamond

§6-83 Definition. (Weitere Matrix-Grundbegriffe)

- (1) Die einzelnen Zahlen in einer Matrix (oder Variablen für solche Zahlen) nennen wir Elemente, (Matrix-) Einträge, Komponenten oder auch Koeffizienten. Diese schreiben wir oft in der Indexschreibweise z.B. als

$a_{ij} :=$ Element der i -ten Zeile und j -ten Spalte der Matrix A .

- (2) Eine $1 \times n$ -Matrix, wobei $n > 1$, nennen wir einen Zeilenvektor.
(3) Eine $m \times 1$ -Matrix, wobei $m > 1$, nennen wir einen Spaltenvektor.
(4) Eine 1×1 -Matrix (a) betrachten wir als dasselbe wie die Zahl a selbst.

\diamond

§6-84 Beispiel.

- (1) In

$$A := \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

sind die a_{ij} für $i, j \in \{1, 2\}$ also die Koeffizienten (Matrix-Elemente / -Einträge / -Komponenten).

(2) $v := (v_1, v_2, v_3, \dots, v_n)$ ist ein Zeilenvektor.

(3) $v := \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \dots \\ v_m \end{pmatrix}$ ist ein Spaltenvektor.

(4) Die Matrix (4) identifizieren wir mit der Zahl 4 (oft auch "Skalar" genannt) selbst.

◊

Wenn die Zahlen für eine Matrix aus einem bestimmten Zahlbereich stammen, schreiben wir das zur Kennzeichnung auch hin. Genauer schreiben wir z.B.

- $\mathbb{N}^{m \times n}$ für die Menge der $m \times n$ -Matrizen mit allesamt Einträgen von natürlichen Zahlen,
- $\mathbb{Z}^{m \times n}$ für die Menge der $m \times n$ -Matrizen mit Einträgen aus den ganzen Zahlen,
- $\mathbb{Q}^{m \times n}$ für die Menge der $m \times n$ -Matrizen mit Einträgen aus den rationalen Zahlen und
- $\mathbb{R}^{m \times n}$ für die Menge der $m \times n$ -Matrizen mit reelle Zahlen als Einträge.

$A \in \mathbb{N}^{m \times n}$ bzw. $(a_{ij}) \in \mathbb{N}^{m \times n}$ meint dann, dass A bzw. (a_{ij}) eine $m \times n$ -Matrix mit Elementen aus den natürlichen Zahlen ist.

In der "Linearen Algebra" werden wir sehen, dass wir mit Matrizen "rechnen" können, weil wir sinnvolle Operationen zwischen ihnen definieren können. Dies macht einen grossen Teil ihrer Nützlichkeit aus. Aber zurück zu unserem Thema:

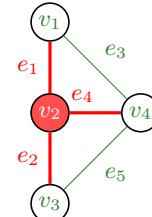
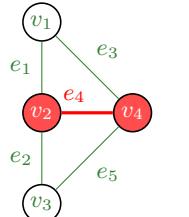
Wie lassen sich nun Graphen oder Digraphen als Matrizen darstellen?

Zuerst erinnern wir bei (Di)Graphen an die beiden wichtigsten Beziehungen, welchen wir schon in §6-4 und §6-18 spezielle Namen gegeben haben. Dies sind die Beziehungen, dass

- ein Knoten mittels Kante oder Pfeil mit einem anderen Knoten verbunden ist (Nachbarschaftsbeziehung zwischen zwei Knoten, wir haben dies Adjazenz genannt),
- und dass ein Knoten ein End- oder Anfangsknoten einer Kante oder eines Pfeils ist, (Nachbarschaftsbeziehung zwischen einem Knoten und einer Kante, wir haben dies Inzidenz bzw. positive / negative Inzidenz genannt).

§6-85 Beispiel. (Adjazenz und Inzidenz in Graphen)

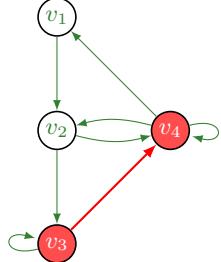
Die Knoten v_2 und v_4 sind adjazent aufgrund Die Kanten e_1 , e_2 und e_4 sind inzident zu v_2 : von e_4 :



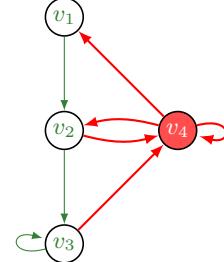
◊

§6-86 Beispiel. (Vorgänger, Nachfolger und Inzidenz in Digraphen)

Knoten v_3 ist Vorgänger von v_4 und damit v_4 Nachfolger von v_3 via der gerichteten Kante (v_3, v_4) :



Die gerichteten Kanten (v_4, v_1) und (v_4, v_2) sind positiv incident zu v_4 . (v_2, v_4) and (v_3, v_4) sind negativ incident zu v_4 . Und die Kante (Schleife) (v_4, v_4) ist sowohl positiv wie auch negativ incident zu v_4 :

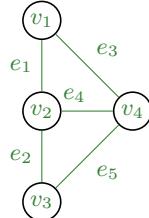


◊

Die Nachbarschaftsbeziehung von Knoten, die Adjazenzbeziehung, können wir nun bequem in einer Matrix dargestellen. Die Idee anhand eines ungerichteten Graphen ist folgende:

§6-87 Beispiel. (Adjazenzmatrix eines Graphen)

Für jeden Knoten schreiben wir eine Zeile und eine Spalte auf. An der Kreuzung der i -ten Zeile und der j -ten Spalte steht eine 1, wenn die Knoten v_i und v_j verbunden sind, und sonst steht eine 0:



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

◊

Deshalb definieren wir formal:

§6-88 Definition. (Adjazenzmatrix von (Di)Graphen)

- (1) Sei $G = (V, E)$ ein schleifenfreier Graph mit Knotenmenge $V = \{v_1, v_2, \dots, v_n\}$.

Die Adjazenzmatrix von G ist eine $(n \times n)$ -Matrix (also eine quadratische Matrix mit n Zeilen und n Spalten), so dass an der Kreuzung a_{ij} der i -ten Zeile und der j -ten Spalte eine 1 steht, wenn v_i und v_j adjazent sind, und sonst eine 0. Formal: $\forall i, j \in \mathbb{N}_n$ gilt

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

- (2) Für die Adjazenzmatrix für einen schleifenfreien gerichteten Gra-

phen $X = (V, E)$ definieren wir entsprechend: $\forall i, j \in \mathbb{N}_n$ gilt

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

(Bitte beachten Sie, dass wir die Adjazenzmatrix hier nur für einen schleifenfreien Digraphen definiert haben!)

◊

Bei gewichteten Graphen, in denen jeder Kante noch ein spezielles “Gewicht” zugeordnet ist, steht anstatt einer 1 bei a_{ij} das Kantengewicht.

Graphen können aufgrund dieser Adjazenzmatrix im Computer als “zweidimensionale Arrays” oder als “verknüpfte Listen” (sogenannte Adjazenzlisten) gespeichert werden (siehe Programmiermodule).

§6-89 Beispiel. (Adjazenzmatrix eines (schleifenfreien) Digraphen)

Eine Adjazenzmatrix eines gerichteten Graphen ist zum Beispiel:

$$X : \quad \begin{array}{c} \text{Diagramm eines gerichteten Graphen mit 6 Knoten } v_1 \text{ bis } v_6 \text{ und folgenden Kanten:} \\ \begin{array}{l} v_1 \rightarrow v_3, v_1 \rightarrow v_2, v_3 \rightarrow v_5, v_3 \rightarrow v_4, v_5 \rightarrow v_3, v_5 \rightarrow v_6, \\ v_2 \rightarrow v_5, v_4 \rightarrow v_5, v_6 \rightarrow v_5 \end{array} \end{array} \quad A_X = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

◊

§6-90 Beispiel. (Adjazenzmatrix eines “kantengewichteten Graphen”)

Ein Beispiel einer Adjazenzmatrix eines “kantengewichteten Graphen” (siehe Abschnitt 6.3.2) erhalten wir durch:

$$G : \quad \begin{array}{c} \text{Diagramm eines kantengewichteten Graphen mit 6 Knoten } v_1 \text{ bis } v_6 \text{ und folgenden Kanten:} \\ \begin{array}{ll} v_2 \xrightarrow{4} v_4, v_4 \xrightarrow{3} v_6, v_1 \xrightarrow{1} v_4, v_1 \xrightarrow{6} v_3, v_2 \xrightarrow{7} v_3, v_3 \xrightarrow{3} v_4, v_3 \xrightarrow{5} v_5, v_4 \xrightarrow{2} v_5, v_5 \xrightarrow{2} v_6 \end{array} \end{array} \quad A_G = \begin{pmatrix} 0 & 1 & 6 & 7 & 0 & 0 \\ 1 & 0 & 0 & 4 & 0 & 0 \\ 6 & 0 & 0 & 3 & 2 & 0 \\ 7 & 4 & 3 & 0 & 5 & 3 \\ 0 & 0 & 2 & 5 & 0 & 2 \\ 0 & 0 & 0 & 3 & 2 & 0 \end{pmatrix}$$

◊

Auch die Nachbarschaftsbeziehung eines Knotens mit einer Kante (gerichtet oder ungerichtet), d.h., die Inzidenzbeziehung, können wir mit einer Matrix darstellen. Als Beispiel die Idee vorerst wiederum anhand eines ungerichteten Graphen:

§6-91 Beispiel. (Inzidenzmatrix eines Graphen)

Für jeden Knoten schreiben wir wieder eine Zeile. Nun schreiben wir aber für jede Kante eine Spalte auf. Beachten Sie, dass damit die Matrix nicht mehr quadratisch sein muss.

An der Kreuzung der i -ten Zeile und der j -ten Spalte steht eine 1, wenn die Kante e_j inzident zum Knoten v_i ist, und sonst steht eine 0:



◊

Deshalb definieren wir formal:

§6-92 Definition. (Inzidenzmatrix eines (Di)Graphen)

- (1) Sei $G = (V, E)$ ein schleifenfreier Graph mit Knotenmenge $V = \{v_1, v_2, \dots, v_n\}$ und Kantenmenge $E = \{e_1, \dots, e_m\}$.

Die Inzidenzmatrix von G ist eine $(n \times m)$ -Matrix (also eine Matrix mit n Zeilen und m Spalten), so dass an der Kreuzung b_{ij} der i -ten Zeile und der j -ten Spalte eine 1 steht, wenn v_i Endknoten der Kante e_j ist (d.h., e_j inzident zu v_i ist), und sonst eine 0. Formal: $\forall i \in \mathbb{N}_n, \forall j \in \mathbb{N}_m$ gilt

$$b_{ij} = \begin{cases} 1 & \text{wenn } \exists w \in V : e_j = \{v_i, w\} \in E, \\ 0 & \text{sonst.} \end{cases}$$

- (2) Für die Inzidenzmatrix für einen *schleifenfreien gerichteten Graphen* $X = (V, E)$ definieren wir analog:

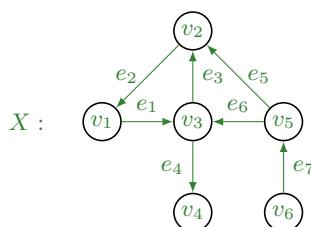
$$b_{ij} = \begin{cases} 1 & \text{wenn } \exists w \in V : e_j = (v_i, w) \in E, \\ -1 & \text{wenn } \exists w \in V : e_j = (w, v_i) \in E \\ 0 & \text{sonst.} \end{cases}$$

◊

Bei gewichteten Graphen steht anstatt einer 1 bei b_{ij} wiederum das Kantengewicht. Graphen können alternativ im Computer auch als Inzidenzmatrix (oder -liste) gespeichert werden.

§6-93 Beispiel. (Inzidenzmatrix eines Digraphen)

Inzidenzmatrix eines gerichteten Graphen:

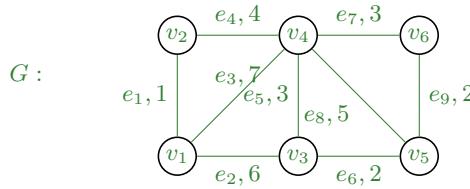


$$B_X = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & -1 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

◊

§6-94 Beispiel. (Inzidenzmatrix eines kantengewichteten Graphen)

Inzidenzmatrix eines “kantengewichteten Graphen” (siehe Abschnitt 6.3.2):



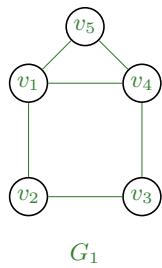
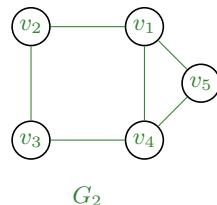
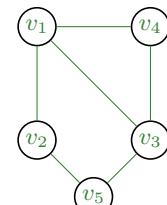
$$B_G = \begin{pmatrix} 1 & 6 & 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 & 3 & 2 & 0 & 0 & 0 \\ 0 & 0 & 7 & 4 & 3 & 0 & 3 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 5 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 2 \end{pmatrix}$$

◊

6.5 Graph- und Digraphisomorphismus

§6-95 Beispiel.

Welche Gemeinsamkeit haben die drei folgenden Graphen?

 G_1  G_2  G_3

- G_1 und G_2 “liegen” nur geometrisch anders in der Zeichenebene.
- G_2 und G_3 lassen sich durch “Umnummerierung der Knoten” ineinander überführen. (Nummeriert man in G_3 die Knoten gemäß $v_4 \mapsto v_5, v_3 \mapsto v_4, v_5 \mapsto v_3$ um, dann erhält man gerade den Graphen G_2 .)

◊

Solche Graphen können wir bezüglich der Beziehungsstruktur ihrer Knoten (d.h., bzgl. der Relation, welche ein Graph beschreibt) als gleich ansehen. Wir sagen auch, sie sind isomorph:

§6-96 Definition. (Isomorphe Graphen)

Zwei Graphen $G = (V, E)$ und $G' = (V', E')$ mit Knotenmengen $V = \{v_1, v_2, \dots, v_n\}$ und $V' = \{v'_1, v'_2, \dots, v'_n\}$ heissen **isomorph**, genau dann wenn eine bijektive Funktion $f : V \rightarrow V'$ existiert, so dass

$$\{u, v\} \in E \text{ genau dann, wenn } \{f(u), f(v)\} \in E'$$

gilt.

Notation: $G \cong G'$

◊

Isomorphe Graphen unterscheiden sich also nur durch die zeichnerische Anordnung und/oder die Benennung der Knoten.

Deshalb fassen wir zwei isomorphe Graphen als identische, nicht unterscheidbare Objekte auf. Wir schreiben auch $f : G \rightarrow G'$ und abgekürzt $f(G) = G'$, wenn die bijektive Funktion $f : V \rightarrow V'$ den Graphisomorphismus von G und G' beschreibt.

Die Bestimmung ob zwei beliebige (sehr grosse) Graphen isomorph sind, ist algorithmisch ein sehr schwieriges Problem, welches unter der Bezeichnung **Graphen-Isomorphieproblem** – oft als **GI-Problem** abgekürzt – bekannt ist.

Auch bei anderen Graph-Varianten lässt sich die Frage nach Isomorphie untersuchen, z.B. durch Einführung des Begriffs des Digraphenismorphismus.

Teil III

Vollständige Induktion, Rekursion

Kapitel 7

Vollständige Induktion

Kapitelinhalt

7.1	Einführung	169
7.1.1	Das Prinzip der vollständigen Induktion	169
7.1.2	Summen- und Produktzeichen	172
7.1.2.1	Summenschreibweise	172
7.1.2.2	Produktschreibweise	174
7.2	Modell-Beispiel	175
7.3	Vollständige Induktion bei Aussagen über Ungleichungen	176
7.4	Vollständige Induktion bei allgemeinen Aussagen	176
7.5	Korrektheit eines Computer-Programmes	177
7.6	Einordnung der Beweismethode der vollst. Induktion	178

L E R N Z I E L E :

LZ 7.1 Sie verstehen das Prinzip der vollständigen Induktion und sind in der Lage, Induktionsbeweise zu führen.

7.1 Einführung

7.1.1 Das Prinzip der vollständigen Induktion

Die Grundidee der vollständigen Induktion in der Mathematik lässt sich mit folgendem kurzen YouTube-Video (<https://youtu.be/uB4DeDMFmJQ?t=24>) zusammenfassen. Dass das Prinzip nicht nur mathematisch sondern auch physikalisch ganz erstaunlich mächtig ist, kann man in diesem YouTube-Video (<https://youtu.be/y97rBdSYbkg>) sehen. Die Idee der vollständigen Induktion ist dieselbe wie am Domino Day: Wenn der erste Stein fällt, fällt der zweite Stein auch um. Durch diesen Impuls fällt auch der dritte, der vierte, ... Stein um. Fassen wir zusammen:

Idee:

Sei

- $U(1)$ die Aussage “Der erste Stein fällt um.”,
- $U(2)$ die Aussage “Der zweite Stein fällt um.”,
- $U(3)$ die Aussage “Der dritte Stein fällt um.”
- usw.

Ziel: Zeigen, dass alle Dominosteine umfallen.

Problem: Wir können nicht für jeden Stein einzeln überprüfen, ob er umgefallen ist.

Vorgehen:

1. Wir beobachten, dass $U(1)$ gilt (Induktionsverankerung).
2. Wir zeigen, dass unter der Annahme, dass für ein beliebiges $n \geq 1$ der n -te Stein fällt (also wenn $U(n)$ gilt) der $(n+1)$ -te Stein auch umfällt (also auch $U(n+1)$ gilt) (Induktionsschritt).

Ziel erreicht: \sim Daraus folgt, dass wenn der erste Stein umfällt, alle anderen Dominosteine nacheinander auch umfallen.

Und: Das gilt sogar, wenn wir abzählbar unendlich viele Dominosteine (d.h. soviele, wie es natürliche Zahlen gibt) aneinanderreihen würden!

Auf dieser Idee basierend können wir das folgende Prinzip festhalten:

§7-1 Definition. (Das Prinzip der vollständigen Induktion)

Basierend auf dieser Idee erhalten wir die Beweistechnik der vollständigen Induktion, mit welcher wir Aussagen der Form

$$\forall n \in \mathbb{N} \text{ mit } n \geq n_0 : A(n)$$

für ein Prädikat $A(n)$ beweisen können. ◊

§7-2 Beispiel. (Gauss'sche Summenformel (“Kleiner Gauss”))

Für alle Zahlen $n \in \mathbb{N} \setminus \{0\}$ gilt

$$1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}.$$

◊

Bemerkung.

- In diesem Beispiel ist $n_0 = 1$ und somit soll für alle Zahlen $n \in \mathbb{N} \setminus \{0\}$ das Prädikat $A(n) := "1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}"$ bewiesen werden.
- Wenn wir für den Beweis dieser Aussage die Methode der vollständigen Induktion verwenden, ersparen wir es uns, das Prädikat $A(n)$ für alle unendlich vielen Zahlen einzeln zu zeigen (was natürlich unmöglich wäre).

Ein Beweis mittels vollständiger Induktion folgt immer demselben Schema:

§7-3 Definition. (Schema eines Beweises mittels vollständiger Induktion)

Zu zeigen: Für ein Prädikat $A(n)$ gilt die folgende Aussage:

$$\forall n \in \mathbb{N} \text{ mit } n \geq n_0 : A(n)$$

Methode:

1. Induktionsverankerung (Induktionsbasis):

Zeigen Sie, dass $A(n_0)$ gilt.

2. Induktionsschritt:

Zeigen Sie für ein allgemeines $n \in \mathbb{N}$ mit $n \geq n_0$, dass unter der Annahme, dass $A(n)$ gilt, auch $A(n+1)$ gilt.

Formal: Zu zeigen ist

$$\forall n \in \mathbb{N} \text{ mit } n \geq n_0 : \left(\underbrace{A(n)}_{\text{Induktionsannahme}} \implies A(n+1) \right).$$

◊

§7-4 Beispiel. (Vollständige Induktion – Beispiel: Beweis der Gauss'schen Summenformel ("Kleiner Gauss"))

Für alle Zahlen $n \in \mathbb{N}$ gilt

$$1 + 2 + 3 + \cdots + n = \sum_{i=1}^n i = \frac{n(n+1)}{2}.$$

Beweis.

- 1. Induktionsverankerung:** Wir überprüfen die Aussage für $n_0 := 1$, d.h., ob $A(n_0)$ eine korrekte Aussage ist. Die linke Seite der Gleichung ist in diesem Fall offensichtlich 1 und wir erhalten weiter auf der rechten Seite:

$$1 = \frac{1(1+1)}{2} = \frac{2}{2} = 1.$$

Also eine korrekte Aussage.

□

- 2. Induktionsschritt:** Jetzt zeigen wir, dass für ein beliebiges $n \in \mathbb{N}$ der Induktionsschritt

$$A(n) \implies A(n+1)$$

gilt.

Induktionsannahme: Wir nehmen also an, dass $A(n)$ gilt. D.h., in unserem Fall, dass die Gleichung

$$1 + 2 + \dots + n = \frac{n(n+1)}{2} \quad (*)$$

gilt.

Zu zeigende Induktionsbehauptung $A(n+1)$: Wir müssen zeigen, dass unter der Voraussetzung $(*)$ auch die Gleichung

$$1 + 2 + \dots + n + (n+1) = \frac{(n+1)(n+2)}{2} \quad (**).$$

gilt.

Beweis (von $(*) \Rightarrow ()$):**

Wir starten mit der linken Seite der zu zeigenden Induktionsbehauptung $(**)$:

$$\begin{aligned} 1 + 2 + \dots + n + (n+1) &= \frac{n(n+1)}{2} + (n+1) = \frac{n(n+1)}{2} + \frac{2(n+1)}{2} \\ &\stackrel{\substack{(\text{Ind.-Annahme } A(n)}{(*) \text{ eingesetzt}})}{=} \frac{n(n+1) + 2(n+1)}{2} \\ &= \underbrace{\frac{(n+1)(n+2)}{2}}_{\text{rechte Seite von } (**)}. \end{aligned}$$

Beachten Sie: Mit dieser Gleichungskette haben wir die zu zeigende Behauptung $(**)$ gezeigt, wobei wir in der ersten Gleichheit dieser Kette die Induktionsvoraussetzung $(*)$ benutzt haben. Damit ist die Behauptung wie gewünscht bewiesen und der Beweis mittels vollständiger Induktion komplett. □ ◇

Zwei Fragen stellen sich in diesem Zusammenhang. Dafür nehmen wir an, dass der Induktionsschritt schon bewiesen worden ist, dass also

$$\forall n \in \mathbb{N} \text{ mit } n \geq n_0 : (A(n) \implies A(n+1))$$

gilt.

1. Darf die Induktionsverankerung bei der Methode der vollständigen Induktion weglassen werden?

2. Darf die Induktionsverankerung für eine Zahl $m \geq n_0$ durchgeführt werden?

Die Antworten zu diesen beiden Fragen sollten wir uns unbedingt merken:

1. Nein, weil sonst der erste Schritt in der Implikationskette nicht erfolgen kann: $A(n_0+1)$ wäre dann nicht gezeigt. Damit könnten wir dann aber auch nicht auf $A(n_0 + 2)$ schliessen, und ebenso wenig auf $A(n_0 + 3)$ usw.. Die Dominosteine würde somit quasi nicht fallen.
2. Ja, aber über die Korrektheit der Aussage für $n < m$ kann dann keine Aussage gemacht werden. Eine bewiesene Korrektheit gilt dann nur für $n \geq m$.

7.1.2 Summen- und Produktzeichen

In unseren Beispielsätzen, welche wir zur Übung mit vollständiger Induktion beweisen wollen, kommen öfters zwei fundamentale Schreibweisen der Mathematik vor. Diese wollen wir deshalb hier genauer besprechen, so dass wir uns danach wieder auf die Induktionsbeweise konzentrieren können.

7.1.2.1 Summenschreibweise

§7-5 Beispiel.

Die Summe $1 + 2 + 3 + \dots + n$ lässt sich abgekürzt schreiben als

$$\sum_{i=1}^n i .$$

◊

§7-6 Definition. (Summenschreibweise für die n -fache Summe, $n \in \mathbb{N}$ beliebig)

Seien a_m, a_{m+1}, \dots, a_n beliebige Zahlen. Die Summe dieser Zahlen lässt sich abgekürzt mit dem Summenzeichen \sum schreiben als

$$a_m + a_{m+1} + \dots + a_n = \sum_{i=m}^n a_i .$$

◊

Wir sehen: Das Ziel muss also sein, die Summanden als einen allgemeinen Ausdruck a_i in Funktion des Index i darzustellen. Dies muss natürlich nicht immer so einfach sein, wie in den noch relativ einfachen Beispielen.

§7-7 Beispiel.

Im Beispiel oben sind die Summanden $a_i = i$ und der Start ist bei $m = 1$.

◊

§7-8 Beispiel. (Aufgabe)

Schreiben Sie die folgenden Summen in der Summenschreibweise:

$$(a) 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = \sum_{i=1}^{10} i$$

$$(b) 2 + 4 + 6 + \dots + 100 = \sum_{i=1}^{50} 2i$$

$$(c) 1 - 4 + 9 - 16 + 25 - 36 + 49 - 64 = \sum_{i=1}^8 (-1)^{i+1} i^2$$

$$(d) 5 + 5 + 5 + 5 + 5 + 5 + 5 = \sum_{i=1}^7 5$$

$$(e) -9 - 8 - 7 - 6 - 5 - 4 - 3 - 2 - 1 + 0 + 1 + 2 + 3 = \sum_{i=-9}^3 i$$

◊

§7-9 Satz. (Rechenregeln für die Summenschreibweise)

1. Für eine konstante Zahl c gilt:

$$\begin{aligned} \sum_{i=m}^n (c \cdot a_i) &= (c \cdot a_m) + (c \cdot a_{m+1}) + \dots + (c \cdot a_{n-1}) + (c \cdot a_n) \\ &= c \cdot (a_m + a_{m+1} + \dots + a_{n-1} + a_n) = c \cdot \sum_{i=m}^n a_i. \end{aligned}$$

2. Summen lassen sich aufteilen:

$$\begin{aligned} \sum_{i=m}^n a_i &= a_m + a_{m+1} + \dots + a_k + a_{k+1} + \dots + a_{n-1} + a_n \\ &= \sum_{i=m}^k a_i + \sum_{i=k+1}^n a_i. \end{aligned}$$

3. Für die Summe von zwei Summen gilt:

$$\begin{aligned} \sum_{i=m}^n (a_i + b_i) &= (a_m + b_m) + (a_{m+1} + b_{m+1}) + \dots + (a_{n-1} + b_{n-1}) + (a_n + b_n) \\ &= (a_m + a_{m+1} + \dots + a_{n-1} + a_n) + (b_m + b_{m+1} + \dots + b_{n-1} + b_n) \\ &= \sum_{i=m}^n a_i + \sum_{i=m}^n b_i. \end{aligned}$$

4. Für eine konstante Zahl c gilt:

$$\sum_{i=m}^n c = \underbrace{c + c + \dots + c}_{(n-m+1)\text{-Mal}} = (n-m+1) \cdot c.$$

Insbesondere gilt also (wenn $m := 1$): $\sum_{i=1}^n c = n \cdot c$.

◊

§7-10 Beispiele.

$$(1) \sum_{i=1}^5 (2i + 3^i) = \sum_{i=1}^5 2i + \sum_{i=1}^5 3^i.$$

$$(2) \sum_{i=1}^{10} 3 = 10 \cdot 3 = 30.$$

◊

7.1.2.2 Produktschreibweise

Analog zum Summenzeichen ist das Produktzeichen definiert:

§7-11 Definition. (Produktschreibweise für das n -fache Produkt, $n \in \mathbb{N}$ beliebig)

Seien a_m, a_{m+1}, \dots, a_n beliebige Zahlen. Das Produkt dieser Zahlen lässt sich abgekürzt mit dem Produktzeichen \prod schreiben als

$$a_m \cdot a_{m+1} \cdot \dots \cdot a_n = \prod_{i=m}^n a_i.$$

◊

§7-12 Beispiele.

$$(1) 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7 \cdot 8 \cdot 9 \cdot 10 = \prod_{i=1}^{10} i$$

$$(2) 2 \cdot 4 \cdot 6 \cdot \dots \cdot 100 = \prod_{i=1}^{50} 2i$$

$$(3) 1 \cdot (-4) \cdot 9 \cdot (-16) \cdot 25 \cdot (-36) \cdot 49 \cdot (-64) = \prod_{i=1}^8 (-1)^{i+1} i^2$$

$$(4) 5 \cdot 5 \cdot 5 \cdot 5 \cdot 5 \cdot 5 \cdot 5 = \prod_{i=1}^7 5 \quad (= 5^7)$$

◊

§7-13 Satz. (Rechenregeln für die Produktschreibweise)

1. Für eine konstante Zahl c gilt:

$$\prod_{i=m}^n (c \cdot a_i) = (c \cdot a_m) \cdot \dots \cdot (c \cdot a_n) = c^{n-m+1} \cdot \prod_{i=m}^n a_i$$

2. Produkte lassen sich aufteilen:

$$\prod_{i=m}^n a_i = a_m \cdot \dots \cdot a_k \cdot a_{k+1} \cdot \dots \cdot a_n = \prod_{i=m}^k a_i \cdot \prod_{i=k+1}^n a_i.$$

3. Für das Produkt von zwei Produkten gilt:

$$\begin{aligned} \prod_{i=m}^n (a_i \cdot b_i) &= (a_m \cdot b_m) \cdot \dots \cdot (a_n \cdot b_n) \\ &= (a_m \cdot \dots \cdot a_n) \cdot (b_m \cdot \dots \cdot b_n) = \prod_{i=m}^n a_i \cdot \prod_{i=m}^n b_i. \end{aligned}$$

4. Für eine konstante Zahl c gilt:

$$\prod_{i=m}^n c = \underbrace{c \cdot c \cdot \dots \cdot c}_{(n-m+1)\text{-Mal}} = c^{n-m+1}$$

$$\text{Insbesondere gilt also (wenn } m := 1\text{): } \prod_{i=1}^n c = c^n.$$

◊

7.2 Einfaches Modell-Beispiel einer vollständigen Induktion

S7-14 Beispiel. (Vollständige Induktion: Ein einfaches Modell-Beispiel)

Für alle $n \in \mathbb{N}_0$ gilt

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

Beweis.

1. **Verankerung:** Sei $n_0 = 0$. Dann gilt für $A(0)$:

$$\begin{array}{ll} \sum_{i=0}^0 2^i = 2^0 = 1 & \text{(linke Seite)} \\ 2^{0+1} - 1 = 2 - 1 = 1 & \text{(rechte Seite)} \end{array}$$

Also ist die Aussage für $n_0 = 0$ korrekt.

2. **Induktionsschritt:**

$$\text{Induktions-Annahme } A(n): \quad \sum_{i=0}^n 2^i = 2^{n+1} - 1.$$

$$\text{Induktions-Beh. } A(n+1): \quad \sum_{i=0}^{n+1} 2^i = 2^{n+2} - 1.$$

Beweis:

$$\sum_{i=0}^{n+1} 2^i = \sum_{i=0}^n 2^i + 2^{n+1} = 2^{n+1} - 1 + 2^{n+1} \underset{\text{(hier } A(n) \text{ eingesetzt)}}{=} 2 \cdot 2^{n+1} - 1 = 2^{n+2} - 1.$$

Mit dieser Gleichheitskette haben wir gerade die zu zeigende Induktions-Behauptung gezeigt und der Beweis mit vollständiger Induktion ist komplett.

□

◊

7.3 Vollständige Induktion bei Aussagen über Ungleichungen

Auch Aussagen über Ungleichungen bezüglich natürlicher Zahlen n können durch vollständige Induktion bewiesen werden. Dazu ein Beispiel zur Demonstration:

§7-15 Beispiel. (Ungleichung von Bernoulli)

Für beliebige Zahlen $x > -1$ und alle natürlichen Zahlen $n \in \mathbb{N} \cup \{0\}$ gilt die folgende Ungleichung von Bernoulli:

$$(1+x)^n \geq 1 + n \cdot x$$

Beweis.

1. Verankerung: Sei $n_0 = 0$. Dann gilt für $A(0)$:

$$\begin{array}{ll} (1+x)^0 = 1 & \text{(linke Seite)} \\ 1 + 0 \cdot x = 1 & \text{(rechte Seite)} \end{array}$$

Also ist die Aussage für $n_0 = 0$ korrekt, da $1 \geq 1$.

2. Induktionsschritt:

Ind.-Annahme $A(n)$: $(1+x)^n \geq 1 + n \cdot x, \quad \forall x > -1$.

Ind.-Beh. $A(n+1)$: $(1+x)^{n+1} \geq 1 + (n+1) \cdot x, \quad \forall x > -1$.

Beweis: $\forall x > -1$:

$$\begin{aligned} (1+x)^{n+1} &= (1+x)^n \cdot (1+x) \geq (1+n \cdot x) \cdot (1+x) = 1 + x + nx + nx^2 \\ &\quad (\text{mit Ind.-annahme } A(n)) \\ &= 1 + (n+1)x + \underbrace{nx^2}_{\geq 0} \geq 1 + (n+1)x. \end{aligned}$$

Mit dieser Ungleichungskette haben wir insbesondere (Term ganz links und Term ganz rechts) gezeigt, dass

$$(1+x)^{n+1} \geq 1 + (n+1)x$$

gilt, was gerade unsere zu zeigende Induktionsbehauptung ist. Damit ist der Beweis komplett.

□
◇

7.4 Vollständige Induktion bei allgemeinen Aussagen

Die vollständige Induktion kann auch zum Beweis von allgemeiner formulierten Aussagen über natürliche Zahlen verwendet werden. Dazu wieder ein Beispiel:

§7-16 Beispiel.

Behauptung: Für alle $n \in \mathbb{N}$ ist

$$8^n - 1$$

durch 7 teilbar.

Beweis.

1. Verankerung: Sei $n_0 = 1$. Dann gilt für $A(1)$:

$$8^1 - 1 = 7$$

Also ist die Aussage für $n_0 = 1$ korrekt, da 7 sich selbst teilt.

2. Induktionsschritt:

Ind.-Annahme $A(n)$: $8^n - 1$ ist durch 7 teilbar.

Ind.-Beh. $A(n + 1)$: $8^{n+1} - 1$ ist durch 7 teilbar.

Beweis:

$$\begin{aligned} 8^{n+1} - 1 &= 8 \cdot 8^n - 1 \quad \underbrace{-7 + 7}_{(\text{kleiner Trick})} = 8 \cdot 8^n - 8 + 7 = 8(8^n - 1) + 7 \\ &= 8 \cdot \underbrace{(8^n - 1)}_{\substack{\text{wegen } A(n) \\ \text{durch 7 teilbar}}} + \underbrace{7}_{\text{durch 7 teilb.}}, \\ &\quad \underbrace{\text{durch 7 teilbar}}_{\text{durch 7 teilbar}} \end{aligned}$$

also ist $8^{n+1} - 1$ durch 7 teilbar und damit die Behauptung des Induktionsschrittes gezeigt.

□

◇

7.5 Vollständige Induktion zum Beweis der Korrektheit eines Computer-Programmes

In sicherheitskritischen Applikationen will man Programme oder Programmteile mathematisch als korrekt beweisen. Dazu kann oft die vollständige Induktion verwendet werden. Ein einfaches Beispiel zur Demonstration:

§7-17 Beispiel.

Sei folgender Code gegeben:

input: Zahl $n \in \mathbb{N}$

```
z = 1
for i = 1 to n do
    z ← 5z
```

output: z

Beh.: Dieses Programm berechnet $z = 5^n$ für alle $n \in \mathbb{N}$.

Beweis. ↗ Übungsblatt.

□



7.6 Einordnung der Beweismethode der vollständigen Induktion in die allgemeinen Beweistechniken

Zum Abschluss noch einmal die Frage, die wir schon im Kapitel zu den Beweistechniken angeschnitten hatten, wieso die “vollständige Induktion” trotzdem ein deduktives Argument, basierend rein auf der Logik, ist. Wie dort schon angedeutet, lautet die Antwort: Weil sie eben vollständig ist! Wie können wir dies etwas genauer verstehen? Wenn wir vollständig, d.h., für jeden Einzelfall auf den allgemeinen Fall schliessen können, dann können wir daraus deduktiv die allgemeine Richtigkeit ableiten und haben einen logisch-mathematischen Beweis der Aussage. Zusammengefasst also:

Sei $Z := \forall n \in \mathbb{N} \cup \{0\}, n \geq n_0, n_0 \in N_0 : A(n)$ eine typische Aussage mit einem Prädikat $A(n)$, bei welcher wir vollständige Induktion anwenden.

Dann liefert das Induktionsprinzip gerade einen direkten (und insbesondere deduktiven!) Beweis, denn es gilt die Implikation

$$(A(n_0) \wedge (\forall n \geq n_0 : A(n) \implies A(n+1))) \implies Z.$$

Kapitel 8

Rekursion

Kapitelinhalt

8.1	Was bedeutet "Rekursion"?	181
8.2	Zahlenfolgen – rekursiv und explizit	184
8.3	Explizite Darstellung von rekursiven Zahlenfolgen bestimmen	187
8.4	Was sind Rekursionsgleichungen?	190
8.5	Homogene lineare Rek.-gleichungen 1. Ordnung	194
8.5.1	Iterationsmethode	194
8.5.2	Lösungsformel für den Fall eines konstanten Koeffizienten $c(n) = c$	195
8.6	Inhomogene lineare Rek.-gleichungen 1. Ordnung	196
8.6.1	Iterationsmethode	196
8.6.2	Lösung einer (inhomogenen) linearen Rekursionsgleichung 1. Ordnung mit konstanten Koeffizienten und Störterm	197
8.6.3	Verallgemeinerung: Nicht konstanter Störterm	199
8.7	Homogene lineare Rek.-gleichungen 2. Ordnung	202
8.8	Inhomogene lineare Rek.-gleichungen 2. Ordnung	205
8.9	Variablentransformation	206
8.9.1	Einleitung	206
8.9.2	Variablentransformationen in Funktionen auf abzählbaren Mengen	206
8.9.3	Variablentransformation zur Analyse von Algorithmen	208
8.9.4	Ausblick: Erzeugendenfunktion, Master-Theorem	211
8.10	Übersicht Lösungsmethoden Rekursionsgleichungen	212

L E R N Z I E L E :

LZ 8.1 Sie können einfachere lineare homogene sowie inhomogene Rekursionsgleichungen (bis zum Grad 2) lösen.

LZ 8.2 Sie sind fähig, durch Variablentransformation Rekursionsgleichungen zu erhalten und zu lösen, in der Art, wie sie häufig bei der Analyse von Algorithmen auftreten.

8.1 Was bedeutet "Rekursion"?

Bei der Induktion sind wir von einem Startelement einer Menge ausgegangen und haben danach für ein beliebiges Element auch ihr Nachfolge-Element betrachtet. Wir haben uns also vom ersten Element quasi Schritt-für-Schritt, sogar eine abzählbar unendliche Anzahl Schritte, von einem mathematischen Objekt zum anderen vorwärts "gehängelt". So konnten wir zum Beispiel Aussagen über Objekte die aus einer geeignet [E.28] geordneten abzählbaren Menge stammen (in unserem Fall waren das z.B. die natürlichen Zahlen) beweisen.

In vieler Hinsicht ist die Umkehrung dieser Idee im Begri einer "Rekursion", bzw., in der Eigenschaft "rekursiv" zu sein, enthalten:

- Wir starten nun bei irgendeinem Element einer geeignet geordneten Menge (z.B. weil wir den Wert dieses Elements berechnen wollen) und gehen dann "rekursiv" (von lat. "recurrere" = zurücklaufen) von Vorgänger zu Vorgänger zurück bis wir am Anfang, beim ersten Element, angekommen sind.
- Oft kann man den Wert dieses ersten Elements ganz einfach berechnen. Und mehr noch: Kennen wir allgemein den Wert eines n -ten Elements, dann können wir oft auch ganz einfach daraus den Wert des $(n + 1)$ -ten Elements berechnen.
- Das heisst, einmal am Anfang angelangt können wir wieder schrittweise vorwärts gehen und in jedem Schritt aus dem vorhergehenden Wert den nächsten Wert berechnen. Und zwar bis wir wieder bei unserem Ausgangselement angelangt sind und auch dessen Wert erhalten.
- Allgemein: Während diesem ganzen Prozess "schauen wir in jedem Schritt was passiert" oder wir stellen in jedem Schritt irgend etwas Gescheites mit den Elementen der Menge an. Es muss nicht immer die Berechnung eines Wertes sein. Vielleicht wollen wir die Elemente auch nach einem bestimmten Kriterium sortieren oder eine Teilmenge bilden, ... usw.. Auch wird die Berechnung des $(n + 1)$ -ten Elements vielleicht nicht nur auf diejenige des n -ten, sondern vielleicht auch des $(n - 1)$ -ten, und $(n - 2)$ -ten ... usw. zurückgeführt.

Mit derselben Idee können wir weiter auch Dinge, welche durch die natürlichen Zahlen geordnet (indiziert) sind, definieren. Wir sagen zum Beispiel, wie das erste Element definiert ist, und sagen dann wie für ein beliebiges n das $(n + 1)$ -te Element in Abhängigkeit des n -ten Elements definiert ist. Dann haben wir alle Elemente der abzählbar unendlichen Menge definiert. Dieses Vorgehen bezeichnet man als rekursive Definition. Wir halten fest:

§8-1 Definition. (Rekursiv, rekursives Vorgehen)

Sei M eine geeignet geordnete Menge. (Zur Vereinfachung werden wir im Folgenden immer die natürlichen Zahlen, also $M := \mathbb{N} \cup \{0\}$ oder $M := \mathbb{N} \setminus \{0\}$ betrachten [E.29].)

Wir bezeichnen ein Vorgehen in M als **rekursiv**, wenn es definiert oder beschrieben wird durch

- (1) einen **Basisfall** (für ein Basiselement, d.h., das erste Element der Menge),
- (2) eine Anzahl **Rückführungregeln**, wie jeder beliebige andere Fall, d.h., jedes andere Element, auf den Basisfall zurückgeführt wird.



Definition 8.1 (Rekursive Struktur; rekursive Definition)

Ist irgendeine mathematische oder andere “Struktur” durch ein rekursives Vorgehen beschrieben, so nennen wir dies eine **rekursive Struktur**.

Entsprechend nennen wir eine **Definition**, welche dieses Vorgehen benutzt, eine **rekursive Definition**.

Rekursive Strukturen treten in vielfältigster Gestalt in der Natur, in der Informatik, in der Mathematik, in der Technik, usw. auf. Ein paar Beispiele:

§8-2 Beispiel. (Rekursive Definition der Fakultät einer natürlichen Zahl)

Die Fakultät $n!$ für eine natürliche Zahl $n \in \mathbb{N} \cup \{0\}$ ist das Produkt aller Faktoren von 1 bis n :

$$\begin{aligned} 0! &:= 1, \\ 1! &:= 1, \\ n! &:= 1 \cdot 2 \cdot \dots \cdot n, \quad n > 1. \end{aligned}$$

Dasselbe können wir auch als *rekursive Definition* so formulieren:

$$\begin{aligned} 0! &:= 1 && (\text{Basisfall für } n = 0) \\ (n+1)! &:= n! \cdot (n+1), \quad \forall n > 0 && (\text{Rückführungsregel für } n > 0) \end{aligned}$$

**§8-3 Beispiel. (Rekursive Prozedur / Funktion in der Informatik)**

In der Informatik gibt es die Möglichkeiten Prozeduren bzw. Programm-Funktionen rekursiv zu implementieren. Im Fall unserer Fakultätsfunktion gerade zum Beispiel durch folgende Prozedur (in Pseudocode):

```

1   function fakultaet(n:integer): integer
2   begin
3       if n < 0 then
4           return ERROR
5       else
6           if n=0 then # Basisfall
7               return 1
8           else
9               return n*fakultaet(n-1) # Rueckfuehrungsregel
10      end

```

Beachten Sie, wie in der Definition der Funktion selbst die zu definierende Funktion (rekursiv) aufgerufen wird!

**§8-4 Beispiel. (Rekursive Strukturen in der Natur)**

In der Natur treten in lebenden Organismen zum Beispiel aufgrund des Phänomens der “Selbstähnlichkeit” rekursive Strukturen und Formen auf:

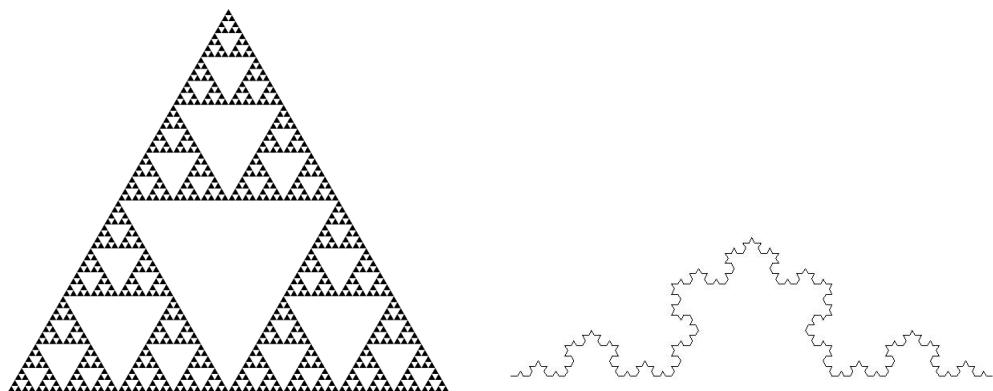


aus: <http://de.wikipedia.org/wiki/Blumenkohl>

Wenn man "reinzoomed", d.h., immer kleinere Teilbereiche anschaut, erkennt man wieder Formen ähnlich der gesamten Struktur. ◇

§8-5 Beispiel. (Rekursive Strukturen als geometrisch-mathematische Objekte)

Im Bereich von "Fraktalen", treten geometrisch-mathematische Objekte mit rekursiven Strukturen auf:



aus: <http://de.wikipedia.org/wiki/Sierpinski-Dreieck> und <http://de.wikipedia.org/wiki/Kochkurve>

◇

§8-6 Beispiel. (Molekulare rekursive Strukturen)

Molekulare Strukturen können oft auch rekursive Formen annehmen, wie man sehr schön an Eiskristallen unter dem Mikroskop erkennt:



By Schnobby - Own work, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=19055302>.

◇

Weiter treten in der Kunst und in zahlreichen anderen Gebieten ebenfalls rekursive Strukturen auf. Sie sehen, das Phänomen ist sehr reichhaltig. Wir wollen uns im Folgenden aber auf den mathematischen Begri der Rekursionsgleichung (Rekurrenzgleichung, Di erenzen- gleichung) beschränken, welcher einer mathematische Darstellung von Rekursionen in Form von Gleichungen entspricht.

8.2 Zahlenfolgen – rekursiv und explizit

Eine erste Möglichkeit auf rekursive mathematische Strukturen zu kommen, ist die Betrachtung von sogenannten Zahlenfolgen. Deshalb führen wir zuerst den Begri einer Zahlenfolge ein und zeigen dann, wie diese auch rekursiv definiert bzw. dargestellt werden können. Zahlenfolgen werden Ihnen in späteren Mathematik-Module (z.B. Analysis, Stochastik, Numerik, ...) und Informatik-Module (Algorithmen, Analyse von Algorithmen, ...) wieder begegnen und dort wird das Thema weiter vertieft werden. Beachten Sie, dass es üblich ist, Zahlenfolgen von 1 an zu indizieren, weswegen wir meist $\mathbb{N} \setminus \{0\}$ betrachten:

§8-7 Definition. (Grundbegri e zu Zahlenfolgen)

- (1) **Funktionale Definition (oder Darstellung):** Eine reelle Zahlenfolge (oft auch nur Zahlenfolge genannt) ist eine Funktion

$$f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}, \quad n \mapsto f(n).$$

Weitere Schreibweisen: $(f(n))_{n \in \mathbb{N} \setminus \{0\}}$, $(f(n))_{n=1}^{\infty}$, $\{f(n)\}_{n=1}^{\infty}$, $(x_n)_{n \in \mathbb{N} \setminus \{0\}}$, für $x_n := f(n)$, ... usw.

$f(n)$ alleine bezeichnen wir auch als **explizite Definition (Darstellung)**.

- (2) Bei der **rekursiven Darstellung (Definition)** k -ter Ordnung, $k \in \mathbb{N} \setminus \{0\}$, einer Zahlenfolge $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$ werden
- die Folgenglieder $f(1), f(2), \dots, f(k)$ angegeben und
 - für $n \geq k + 1$ eine Vorschrift, wie das n -te Folgenglied $f(n)$ aus den Folgengliedern $f(n-1), f(n-2), \dots, f(n-k)$ berechnet wird.
- (3) In der **aufzählenden Darstellung** der Zahlenfolge f werden alle Folgenglieder – durch jeweils ein Komma getrennt – aufgelistet:

$$f(1), f(2), f(3), f(4), \dots$$

(Dies kann aber i.A. nicht eine Definition sein, weil unvollständig und mehrdeutig.)



§8-8 Beispiele.

(1) Fakultät:

- $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}, n \mapsto f(n) := n! := n \cdot (n-1) \cdot (n-2) \cdots 3 \cdot 2 \cdot 1$. (Funktionale/explizite Definition)^[E.30]
- $f(1) := 1, f(n) := n \cdot f(n-1)$ für $n \in \mathbb{N}$ mit $n \geq 2$. (Rekursive Definition 1. Ordnung).
- Aufzählende Darstellung: 1, 2, 6, 24, 120, ...

(2) Fibonacci-Folge^[E.31]:

- Rekursive Definition 2. Ordnung:

$$\begin{aligned} f(1) &:= 1, & f(2) &:= 1, \\ f(n) &:= f(n-1) + f(n-2) \text{ für } n \in \mathbb{N} \text{ mit } n \geq 3. \end{aligned}$$

- Aufzählende Darstellung: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...
- Explizite Definition:

$$f(n) = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right)$$



In Anwendungen kommt es öfters vor, dass wir eine aufzählende Zahlenfolge haben (aufgrund von empirischen Daten, Messungen, etc.) und wir dann eine rekursive und explizite Form finden möchten.

§8-9 Beispiele. (Rekursive und explizite Darstellungen für aufzählend gegebene Zahlenfolgen)

Wie lauten für die folgenden Zahlenfolgen in aufzählender Darstellung jeweils eine rekursive und eine explizite Darstellung?

- (1) 1, 4, 7, 10, 13, 16, ...
- (2) 2, 4, 8, 16, 32, 64, ...
- (3) 3, 33, 333, 3333, 33333, ...

Lösung:

- (1) rekursiv: $f(1) = 1, f(n) = f(n-1) + 3, \forall n > 1$.
explizit: $f(n) = 3n - 2, \forall n \in \mathbb{N} \setminus \{0\}$.
Wie? Werden wir gleich lernen \leadsto Induktives Schliessen!
- (2) rekursiv: $f(1) = 2, f(n) = 2f(n-1), \forall n > 1$.
explizit: $f(n) = 2^n, \forall n \in \mathbb{N} \setminus \{0\}$.
Wie? \leadsto das folgt!
- (3) rekursiv: $f(1) = 3, f(n) = 10f(n-1) + 3, \forall n > 1$.
explizit: $f(n) = \sum_{i=1}^n (3 \cdot 10^{i-1}) = 3 \sum_{i=1}^n 10^{i-1} (= \frac{1}{3}(10^n - 1)), \forall n \in \mathbb{N} \setminus \{0\}$.
Wie? \leadsto das folgt!



Fazit: Die rekursive Darstellung ist oft einfacher aus der aufzählenden Darstellung zu bestimmen als die explizite Darstellung. Die explizite Darstellung ist aber einfacher anzuwenden, um für eine beliebige natürliche Zahl n das entsprechende Folgenglied $f(n)$ zu bestimmen. Um mit der rekursiven Darstellung ein beliebiges Folgenglied $f(n)$ zu berechnen, muss ja der Wert von $f(n-1)$ gegeben sein und allenfalls (rekursiv) berechnet werden. Haben wir aber eine explizite Darstellung, dann müssen wir "nur" einen algebraischen Ausdruck für n auswerten (welcher aber natürlich auch relativ kompliziert sein kann und zudem numerische Problemen unterworfen sein kann!).

Man will also oft aus einer rekursiven Darstellung (Definition) ihre explizite Darstellung bestimmen.

~ Wie, das werden wir gleich lernen!

§8-10 Definition. (Berechnung des Folgengliedes $f(n)$ einer rekursiven Zahlenfolge mit Rekursionsbaum)

Die Berechnung oder Darstellung des n -ten Folgengliedes $f(n)$ einer rekursiven Zahlenfolge erfolgt an ihrem Rekursionsbaum (= gerichteter Baum mit Wurzel $f(n)$) dadurch, dass wir:

- (1) die Anfangswerte zu den entsprechenden Blättern hinzufügen,
- (2) im Baum aufsteigend für jeden Vorgänger gemäss f den Funktionswert ausrechnen und hinschreiben, bis wir an der Wurzel $f(n)$ angekommen sind und so den Funktionswert $f(n)$ (die Auswertung) für das vorgegebene n erhalten haben.

Wir sagen auch, dass wir $f(n)$ für ein solches spezielle n auswerten.

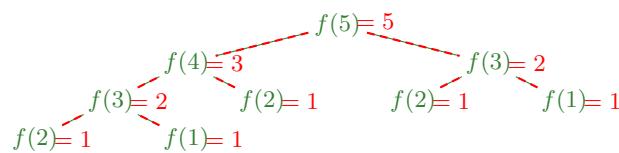


Für eine Rekursion der Ordnung k ist der jeweilige Rekursionsbaum gerade ein gerichteter Baum mit Grad k . (D.h., jeder innerer Knoten ist Wurzel eines (Teil-) Baumes mit k Nachfolgerknoten) Es ist üblich, Rekursionsbäume nur mit Kanten zu zeichnen, obwohl sie gerichtete Bäume sind. Die Kantenrichtung ist dann implizit von oben nach unten.

§8-11 Beispiel. (Rekursionsbaum und Auswertung für $f(5)$ der Fibonacci-Folge)

Wir bestimmen $f(5)$ der Rekursion für die Fibonacci-Folge gegeben durch:

$$f(1) = 1, \quad f(2) = 1, \quad f(n) = f(n-1) + f(n-2), \forall n \geq 2.$$



8.3 Explizite Darstellung von rekursiven Zahlenfolgen bestimmen

Wie kann man nun effektiv eine explizite (und damit auch eine funktionale) Darstellung aus einer rekursiven Darstellung für eine Zahlenfolge gewinnen? Die Idee ist die Folgende: Mit Hilfe von Zahlenbeispielen versuchen wir von der rekursiven Darstellung der Zahlenfolge auf eine explizite Darstellung zu schliessen. Dieses Vorgehen heisst **induktives Schliessen**. Danach können wir eine allfällige explizite Darstellung mittels vollständiger Induktion beweisen.

§8-12 Beispiel.

Wir bestimmen eine explizite Darstellung der rekursiven Zahlenfolge 1. Ordnung, welche definiert ist durch

$$\begin{aligned} f(1) &= 1, \\ f(n) &= 4 \cdot f(n-1) + 4^{n-1} \text{ für } n \in \mathbb{N} \text{ mit } n \geq 2. \end{aligned}$$

Wie gehen wir vor? Zuerst stellen wir gezielt eine *Vermutung* auf:

$$\begin{aligned} f(1) &= 1. (= \underline{4^0 \cdot 1}) \\ f(2) &= 4f(1) + 4^1 = 4 \cdot 1 + 4 = 4(1+1) = \underline{4 \cdot 2} = 8. \\ f(3) &= 4f(2) + 4^2 = 4 \cdot 4 \cdot 2 + 4^2 = 4^2 \cdot 2 + 4^2 = 4^2(2+1) = \underline{4^2 \cdot 3} = 48. \\ f(4) &= 4f(3) + 4^3 = 4 \cdot 4^2 \cdot 3 + 4^3 = 4^3 \cdot 3 + 4^3 = 4^3(3+1) = \underline{4^3 \cdot 4} = 256. \end{aligned}$$

Die unterstrichenen Ausdrücke und ihre augenfällige Regelmässigkeit $4^{n-1} \cdot n$ für die jeweiligen n lassen uns folgende Vermutung aufstellen:

$$\text{Vermutung: } f(n) = n \cdot 4^{n-1}.$$

Die Vermutung, die aus diesem induktiven Schliessen resultiert, muss wie gesagt noch bewiesen werden. Wie ist natürlich klar: Dafür eignet sich die Beweismethode der *vollständigen Induktion*, da wir es ja mit einer Aussage über die abzählbare Menge \mathbb{N} zu tun haben.

Wir beweisen durch vollständige Induktion, dass $f(n) = n \cdot 4^{n-1}$ für $n \in \mathbb{N} \setminus \{0\}$ die explizite Darstellung der rekursiv definierten Zahlenfolge

$$\begin{aligned} f(1) &= 1, \\ f(n) &= 4 \cdot f(n-1) + 4^{n-1}, \text{ für } n \in \mathbb{N} \text{ mit } n \geq 2 \end{aligned}$$

aus dem vorherigen Beispiel ist.

Lösung: Beweis durch vollständige Induktion

Verankerung: Für $n = 1$ gilt: $f(1) = 1$ nach rekursiver Definition.

In der zu beweisenden Formel eingesetzt (d.h. $n \cdot 4^{n-1}$) erhalten wir: $1 \cdot 4^0 = 1 \cdot 1 = 1$. Also was gelten muss.

Induktionsschritt:

Beachten Sie: Anstatt $A(n) \Rightarrow A(n+1)$ für beliebiges $n \geq 1$ können wir auch $A(n-1) \Rightarrow A(n)$ für beliebiges $n \geq 2$ zeigen! Das ist gleichwertig.

Annahme: Es gelte $f(n-1) = (n-1) \cdot 4^{n-2}$ für beliebiges $n \geq 2$ und die genannte rekursive Formel.

Behauptung: Es gilt dann $f(n) = n \cdot 4^{n-1}$ für beliebiges $n \geq 2$ und für die genannte rekursive Formel.

Beweis (der Induktionsbehauptung):

$$\begin{aligned}
 f(n) &= 4 \cdot f(n-1) + 4^{n-1} && \text{(nach rekursiver Definition)} \\
 &= 4 \cdot (n-1) \cdot 4^{n-2} + 4^{n-1} && \text{(Induktionsannahme eingesetzt)} \\
 &= 4^{n-1}(n-1) + 4^{n-1} && \text{(Potenzregel } 4^1 \cdot 4^{n-2} = 4^{1+n-2} = 4^{n-1}) \\
 &= 4^{n-1}((n-1) + 1) && \text{(\(4^{n-1}\) ausgeklammert)} \\
 &= 4^{n-1} \cdot n. && \square
 \end{aligned}$$

◊

Eine Bemerkung zum Beweis einer vermuteten expliziten Darstellung mittels vollständiger Induktion:

- Die Verankerung ist bei einer rekursiv definierten Zahlenfolge durch den Anfangswert $f(1)$ der rekursiven Definition bestimmt.
- Im Induktionsschritt $(f(n-1) = \dots) \implies (f(n) = \dots)$

wird zuerst $f(n)$ rekursiv formuliert. In diesen Term wird dann die Annahme – also dass die entsprechende explizite Formel für $f(n-1)$ gilt – eingesetzt und damit muss dann die Induktionsbehauptung bewiesen werden.

§8-13 Satz und Definition. (Beweisen durch Einsetzen)

Die Korrektheit des Induktionsschrittes im Beweis einer expliziten Darstellung einer Zahlenfolge kann auch erfolgen, indem überprüft wird,

1. ob die explizite Darstellung der Zahlenfolge für $n = 1$ gilt und
2. ob die explizite Darstellung, eingesetzt in die rekursive Definition, eine korrekte Gleichung ergibt.

Diese Art des Nachweises der Gültigkeit des Induktionsschrittes nennt man auch Beweisen durch Einsetzen.

◊

§8-14 Beispiel. (Beweisen durch Einsetzen)

Wir beweisen durch Einsetzen, dass $f(n) = n \cdot 4^{n-1}$ für $n \in \mathbb{N} \setminus \{0\}$ die explizite Darstellung der rekursiv definierten Zahlenfolge

$$\begin{aligned}
 f(1) &= 1 \\
 (*) \quad f(n) &= 4 \cdot f(n-1) + 4^{n-1} \text{ für } n \in \mathbb{N} \text{ mit } n \geq 2
 \end{aligned}$$

ist.

Lösung:

Wir behaupten also dass für $n \in \mathbb{N} \setminus \{0\}$ die Formel

$$(**) \quad f(n) = n \cdot 4^{n-1}$$

die obige rekursive Definition darstellt und wollen das durch Einsetzen zeigen.

- $f(1) \underset{(*)}{=} 1 \cdot 4^{1-1} = 1 \cdot 1 = 1 \checkmark.$

- Behauptung (**) rechts und links in die rekursiven Definition (*) einsetzen und prüfen, ob dies eine wahre Gleichheit ergibt:

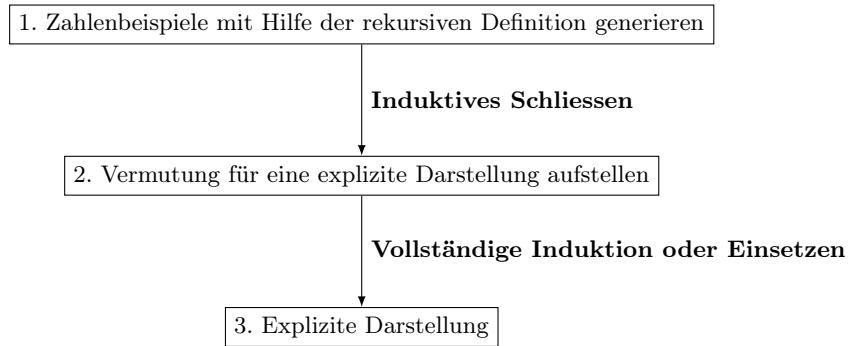
(**) links in (*) ergibt: $n \cdot 4^{n-1}$.

$$\begin{aligned} (\text{**}) \text{ rechts in } (*) \text{ ergibt: } & 4 \cdot (n-1) \cdot 4^{n-2} + 4^{n-1} = (n-1) \cdot 4^{n-1} + 4^{n-1} = \\ & n \cdot 4^{n-1} - 4^{n-1} + 4^{n-1} = n \cdot 4^{n-1}. \end{aligned}$$

Also auf beiden rechten Seiten dasselbe ✓.

◊

Zum Überblick noch einmal das Vorgehen beim Bestimmen einer expliziten Darstellung von rekursiv definierten Zahlenfolgen:



Die Erweiterung dieses Vorgehens auf rekursiv definierte Zahlenfolgen k -ter Ordnung für ein $k \in \mathbb{N}, k > 1$, erfolgt in naheliegender Weise. Wir erhalten dann folgende allgemeine Definition:

§8-15 Definition. (Explizite Darstellung rekursiver Zahlenfolgen k -ter Ordnung)

Sei $k \in \mathbb{N} \setminus \{0\}$.

Die explizite Darstellung einer gegebenen rekursiven Zahlenfolgen k -ter Ordnung ist eine Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$, für welche durch vollständige Induktion gezeigt werden kann, dass sie die Rekursion löst. Dabei gilt:

- Bei rekursiven Zahlenfolgen k -ter Ordnung ist die Verankerung durch die ersten k Folgenglieder bestimmt.
- Im Induktionsschritt muss für die Konklusion ($f(n) = \dots$) die Korrektheit von den k vorangehenden Folgengliedern angenommen werden, also soll bewiesen werden, dass die Implikation

$$\begin{aligned} (f(n-1) = \dots) \wedge (f(n-2) = \dots) \wedge \dots \wedge (f(n-k) = \dots) \\ \implies (f(n) = \dots) \end{aligned}$$

wahr ist.

◊

Diese Definition soll als Hinweis verstanden werden, wie sich dies verallgemeinern lässt. Näher diskutieren wollen wir dies hier aber in dieser Allgemeinheit nicht.

8.4 Was sind Rekursionsgleichungen?

Wenn wir uns die rekursive Definition einer Zahlenfolge anschauen, zum Beispiel

$$\begin{aligned} f(1) &= 4, \\ f(n) &= 3f(n-1), \quad n > 1, \end{aligned}$$

dann sehen wir, dass es sich im Wesentlichen um eine Anzahl Gleichungen handelt, bei welcher wir die explizite Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$ als die Unbekannte aussuchen können und diese Funktion f erstmal "nur" in rekursiver Form vorliegt. Damit sehen wir auch, dass die Aufgabe für eine solche rekursiven Definition einer Funktion f eine explizite Darstellung zu finden dem Lösen einer Gleichung, hier eben einer Rekursionsgleichung entspricht. Ganz allgemein sind Rekursionsgleichungen und ihre Lösungen folgendermassen definiert:

§8-16 Definition. (Rekursionsgleichung k -ter Ordnung; Lösung; Anfangswerte (Anfangsbedingungen))

- (1) Eine (reelle) Rekursionsgleichung k -ter Ordnung, $k \in \mathbb{N} \setminus \{0\}$, ist eine Gleichung der Form

$$f(n) = g(n, f(n-1), \dots, f(n-k)) \quad (8.1)$$

für eine unbekannte Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$ und $n > k$.

$g(n, f(n-1), \dots, f(n-k))$ bedeutet hier, dass g (die rechte Seite) einfach "irgendein Ausdruck" bestehend aus den k Vorgänger-Funktionswerte $f(n-1), \dots, f(n-k)$ und Termen in n ist, also selber als eine Funktion g in den Variablen n und $f(n-1)$ bis $f(n-k)$ gesehen werden kann.

- (2) Eine Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$ ist eine Lösung der Rekursionsgleichung, wenn sie eingesetzt in (8.1) diese Gleichung erfüllt.
 (3) Sind für eine (allgemeine) Rekursionsgleichung in der allgemeinen Funktion $f(n)$, $n > k$, noch zusätzliche Bedingungen für $f(1), f(2), \dots, f(k)$ gegeben, so nennen wir diese zusätzlichen Bedingungen **Anfangswerte** oder **Anfangsbedingungen** der (allgemeinen) Rekursionsgleichung.

◇

Auch bei Rekursionsgleichungen müssen wir flexibel sein, ob jetzt die 0 in \mathbb{N} und damit im Definitionsbereich der gesuchten Funktion f drin ist oder nicht. Entsprechend müssen die Indizes um eins nach unten verschoben gelesen werden. Zur Verdeutlichung noch diese Variante:

§8-17 Definition. (Def. §8-16 für $\mathbb{N} \cup \{0\}$)

- (1) Eine (reelle) Rekursionsgleichung k -ter Ordnung, $k \in \mathbb{N} \setminus \{0\}$, ist eine Gleichung der Form

$$f(n) = g(n, f(n-1), \dots, f(n-k)) \quad (8.2)$$

für eine unbekannte Funktion $f : \mathbb{N} \cup \{0\} \rightarrow \mathbb{R}$ und $n > k-1$.

$g(n, f(n-1), \dots, f(n-k))$ bedeutet hier, dass g (die rechte Seite) einfach

“irgendein Ausdruck” bestehend aus den k Vorgänger-Funktionswerte $f(n-1), \dots, f(n-k)$ und Termen in n ist, also selber als eine Funktion g in den Variablen n und $f(n-1)$ bis $f(n-k)$ gesehen werden kann.

- (2) Eine Funktion $f : \mathbb{N} \cup \{0\} \rightarrow \mathbb{R}$ ist eine Lösung der Rekursionsgleichung, wenn sie eingesetzt in (8.2) diese Gleichung erfüllt.
- (3) Sind für eine (allgemeine) Rekursionsgleichung in der allgemeinen Funktion $f(n)$, $n > k-1$, noch zusätzliche Bedingungen für $f(0), f(1), \dots, f(k-1)$ gegeben, so nennen wir diese zusätzlichen Bedingungen **Anfangswerte** oder **Anfangsbedingungen** der (allgemeinen) Rekursionsgleichung.

◊

Ein Beispiel:

§8-18 Beispiel. (Reelle nichtlineare Rekursionsgleichung 2. Ordnung)

Die Gleichungen

$$\begin{aligned} f(0) &:= 1, \\ f(1) &:= 2, \\ f(n) &:= 3n^2 f(n-1) + 2nf(n-2)^2, \quad n > 1 \end{aligned}$$

definieren eine nichtlineare Rekursionsgleichung 2. Ordnung mit zwei Anfangsbedingungen. Diese beginnt bei 0 (und die k 's und n 's in der Definition müssen an der entsprechenden Stelle um eins nach unten verschoben gedacht werden. Wie in Def. §8-17 formuliert. ◊

In der Analysis z.B. (siehe eana-Modul) schreibt man übrigens meist $x_n = g(n, x_{n-1}, \dots, x_{n-k})$, d.h., man schreibt $x_n := f(n)$ für die einzelnen Werte der Rekursion. Zwei prominente Beispiele:

§8-19 Beispiel. (Newton-Iteration)

Die Gleichung

$$x_{n+1} := x_n - \frac{h(x_n)}{h'(x_n)},$$

ist die berühmte “Newton-Iteration” für eine “stetig differenzierbare reelle Funktion” $h : \mathbb{R} \rightarrow \mathbb{R}$ (Details im eana-Modul). Sie ist eine nichtlineare Rekursionsgleichung 1. Ordnung. (Auch sie wird oft beim Startwert x_0 gestartet, hat also einen Anfangswert bei 0.) ◊

§8-20 Beispiel. (Logistische Gleichung)

Durch

$$x_{n+1} := rx_n(1 - x_n), \quad \text{für } x_n \in [0, 1], r > 0,$$

ist die berühmte “logistische Gleichung” (Verhulst-Gleichung) definiert. Auch sie ist eine nichtlineare Rekursionsgleichung 1. Ordnung. ◊

Die eben gegebenen beiden Definitionen einer Rekursionsgleichung sind sehr allgemein. Obwohl es nicht schwierig ist, eine allgemeine Rekursionsgleichung dieser Art hinzuschreiben, so sind für derart allgemeine Fälle praktisch nie eine “Lösungstheorie” oder gar eine einfache Lösungsformel möglich. Daran sollten Sie sich auf alle Fälle erinnern, wenn Sie in der Praxis “einfach so” auf eine Rekursionsgleichung stoßen. Für den Fall einfacherer “linearer”

Rekursionsgleichungen (Definition folgt gleich) werden wir allerdings in den anschliessenden Abschnitten relativ allgemeine Verfahren kennenlernen. Und anschliessend werden wir auch sehen, wie in ausgewählten Fällen zum Beispiel durch Variablentransformation doch noch nicht "ganz alles hōnungslos" ist. Aber zuerst einmal: Was ist also eine "lineare" Rekursionsgleichung?

§8-21 Definition. (Lineare Rekursionsgleichung k -ter Ordnung)

Eine lineare (reelle) Rekursionsgleichung k -ter Ordnung, $k \in \mathbb{N} \setminus \{0\}$, ist eine Gleichung der Form

$$f(n) = s(n) + \sum_{i=1}^k c_{n-i}(n) \cdot f(n-i)$$

für alle $n \in \mathbb{N} \setminus \{0\}$ mit $n > k$. Darin nennen wir $s(n)$ den Störterm und die $c_{n-i}(n)$ die Koeffizienten bzw. die Koeffizienten-Funktionen. Beachten Sie, dass diese alle von n abhängig sind. ◇

Der Störterm bei linearen Rekursionsgleichungen wird übrigens auch sehr oft hinter die Summe der rekursiven Terme gestellt, also so geschrieben:

$$f(n) = \left(\sum_{i=1}^k c_{n-i}(n) \cdot f(n-i) \right) + s(n)$$

Dies sollte Sie nicht verwirren, denn das ist ja nur eine Sache der Schreibweise einer Summe.

§8-22 Beispiele. (Lineare Rekursionsgleichungen)

- (1) $f(n) = n \cdot 3^n + 2nf(n-2) + \cos(n)f(n-1)$.
(linear, 2.Ordnung, $s(n) = n \cdot 3^n$, $c_{n-2}(n) = 2n$, $c_{n-1}(n) = \cos(n)$.)
- (2) $f(n) = 1 + 2 \cdot f(n-1)$.
(linear, 1.Ordnung, konstanter Störterm $s(n) = 1$ und konstante Koeffizientenfunktion $c_{n-1}(n) = 2$). ◇

Wir haben gesehen, dass eine Lösung einer Rekursionsgleichung in unserem Fall eine ganze "Folge" von reellen Zahlen ist. Und eine Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$ (also mit Definitionsbereich $\mathbb{N} \setminus \{0\}$) ist genau das, was wir auch eine (reelle) Zahlenfolge genannt haben (siehe Def. §8-7). Ein wichtiger Untertyp von linearen Rekursionsgleichungen, sind die "homogenen". Das sind diejenigen, in welchen der Störterm 0 ist. Wir definieren also:

§8-23 Definition. (Homogene lineare Rekursionsgleichung k -ter Ordnung)

Eine lineare Rekursionsgleichung (k -ter Ordnung) heisst homogen, wenn der Störterm $s(n) = 0$ ist:

$$f(n) = \sum_{i=1}^k c_{n-i}(n) \cdot f(n-i)$$



§8-24 Beispiele. (Weitere Beispiele aus dem Zoo der Rekursionsgleichungen)

- (1) **Fakultät:** $f(n) = n \cdot f(n - 1)$ ist eine homogene lineare Rekursionsgleichung 1. Ordnung.
- (2) $f(n) = 2 \cdot f(n - 1)$ ist eine homogene lineare Rekursionsgleichung 1. Ordnung.
- (3) **Türme von Hanoi:** $f(n) = 1 + 2 \cdot f(n - 1)$ ist eine (inhomogene) lineare Rekursionsgleichung 1. Ordnung.

Dies ist eine sehr schöne Motivierung zum Thema Rekursionsgleichungen. Bei Interesse finden Sie in https://de.wikipedia.org/wiki/T%C3%BCrme_von_Hanoi eine Beschreibung des Problems.

- (4) $f(n) = 3 + 10 \cdot f(n - 1)$ ist eine (inhomogene) lineare Rekursionsgleichung 1. Ordnung.
- (5) **Fibonacci-Folge:** $f(n) = f(n - 1) + f(n - 2)$ ist eine homogene lineare Rekursionsgleichung 2. Ordnung.
- (6) $f(n) = 4^{n-1} + 4 \cdot f(n - 1)$ ist eine (inhomogene) lineare Rekursionsgleichung 1. Ordnung.
- (7) $f(n) = 2 - \frac{1}{f(n-1)}$ ist keine lineare Rekursionsgleichung.
- (8) $f(n) = 5n - 2(f(n - 1))^2$ ist keine lineare Rekursionsgleichung.



Da das Lösen einer Rekursionsgleichung wie bei allen Gleichungen eine der Hauptaufgaben ist, legen wir auch dies präzise fest:

§8-25 Definition. (Lösung einer linearen Rekursionsgleichung k -ter Ordnung)

Eine lineare Rekursionsgleichung k -ter Ordnung, $k \in \mathbb{N} \setminus \{0\}$ der Form

$$f(n) = s(n) + \sum_{i=1}^k c_{n-i}(n) \cdot f(n - i)$$

für $n > k$, zu lösen bedeutet, die sogenannte allgemeine Lösung $f(n)$ zu finden, d.h., eine Funktion $f : \mathbb{N} \setminus \{0\} \rightarrow \mathbb{R}$, welche diese Gleichung erfüllt. Diese Funktion f ist dann eine explizite Darstellung der Folge, mit anderen Worten, wir suchen durch das Lösen eine explizite Darstellung.



Beachten Sie zum Beispiel: Die Lösung der Gleichung $f(n) = f(n - 1) + f(n - 2)$ (welche die Fibonacci-Folge definiert) ist erst durch die Angabe der beiden Anfangsbedingungen $f(1)$ und $f(2)$ eindeutig bestimmt. Es gilt allgemein:

§8-26 Satz. (Eindeutige Lösbarkeit einer linearen Rekursionsgleichung k -ter Ordnung)

Die Lösung einer linearen Rekursionsgleichung k -ter Ordnung der Form

$$f(n) = s(n) + \sum_{i=1}^k c_{n-i}(n) \cdot f(n - i)$$

ist erst durch Angabe von k Anfangsbedingungen

$$f(1), f(2), \dots, f(k)$$

eindeutig bestimmt.

◊

8.5 Homogene lineare Rekursionsgleichungen 1. Ordnung lösen

Nun kommen wir zu unserem ersten Typ Rekursionsgleichung, für welchen es ausnahmsweise eine Lösungstheorie gibt. Im allgemeinen können wir die sogenannte "Iterationsmethode", wie in Abschnitt 8.3 bereits gemacht, anwenden, um eine sinnvolle Vermutung der Lösung zu erhalten. Diese Vermutung können wir dann wie auch gesehen, sowohl mit der üblichen vollständigen Induktion als auch mit "Beweisen durch Einsetzen" prüfen und sehen, ob die Vermutung tatsächlich eine Lösung ist. Diese "Iterationsmethode" studieren wir im nächsten Abschnitt. Anschliessend werden wir dann noch sehen, dass wir im Fall, dass die Rekursionsgleichung einen konstanten Koeffizienten $c(n) = c$ hat (also gar nicht von n abhängt), wir sogar eine einfache explizite Lösungsformel angeben können.

8.5.1 Iterationsmethode

Wir studieren die Idee der Iterationsmethode nochmals anhand zweier einfacher Beispiele (die Idee ist aber ganz im Sinn, wie wir schon in Abschnitt 8.3 vorgegangen sind):

§8-27 Beispiel. (Lösung des Problems der Türme von Hanoi)

Wir lösen die folgende *homogene* lineare Rekursionsgleichung, welche wegen $f(n) = 2^n \iff f(n) - 1 = 2^n - 1$ auch gerade ein Weg ist, zur Lösung des Problems der Türme von Hanoi zu kommen (hier nochmals die [Problembeschreibung](#)):

$$f(n) = 2 \cdot f(n - 1)$$

mit $f(1) = 2$.

Lösung:

- Wir überlegen uns:

$$\begin{aligned} f(n) &= 2 \cdot f(n - 1) = 2 \cdot 2 \cdot f(n - 2) = 2 \cdot 2 \cdot 2 \cdot f(n - 3) \\ &= \dots = \underbrace{2 \cdot 2 \cdot \dots \cdot 2}_{n-1\text{-Mal}} \cdot \underbrace{f(1)}_{=2} = \underbrace{2 \cdot 2 \cdot \dots \cdot 2}_{n\text{-Mal}} = 2^n. \end{aligned}$$

D.h., die explizite Formel scheint

$$f(n) = 2^n$$

zu sein. ($\iff f(n) - 1 = 2^n - 1$, und letzteres wird damit die Lösung im Hanoi-Problem sein!)

- Das dem so ist, verifizieren wir leicht: $2^1 = 2 = f(1)$ und $2 \cdot 2^{n-1} = 2^n = f(n)$.

◊

§8-28 Beispiel. (Fakultät)

Wir lösen die *homogen* lineare Rekursionsgleichung, welche die Fakultät rekursiv definiert:

$$\text{Fakultät :} \quad f(n) = n \cdot f(n-1) \\ f(1) = 1$$

Lösung: Hier gilt

$$\begin{aligned} f(n) &= n \cdot f(n-1) = n \cdot (n-1) \cdot f(n-2) \\ &= \dots = n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot 2 \cdot \underbrace{f(1)}_{=1} \\ &= n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot 2 \cdot 1 =: n! \quad (\text{nach Def. der Fakultät}) \end{aligned}$$

Also ergibt obige rekursive Definition tatsächlich die Fakultät. (Dass dies stimmt, sieht man hier schon unmittelbar. Im Detail: $f(1) = 1$ und $n \cdot f(n-1) = n \cdot (n-1)! = n! = f(n)$.) \diamond

Allgemein können wir dieses iterative (d.h., schrittweise) Vorgehen so als Methode zusammenfassen:

§8-29 Satz. (Homogene Rekursionsgleichungen 1. Ordnung lösen mit der Iterationsmethode)

- **Gegeben:** $f(n) = c(n) \cdot f(n-1)$ mit $f(1) = a$ ($a \in \mathbb{R}$).
- **Vorgehen:** $f(n-1), f(n-2), \dots, f(1)$ nach und nach einsetzen und versuchen, das Ergebnis zu vereinfachen. Daraus ergibt sich oft eine sinnvolle Vermutung, welche dann noch als korrekt bewiesen werden muss.

\diamond

Beachten Sie, dass die Iterationsmethode der Methode des induktiven Schliessens entspricht: Die explizite Darstellung wird aus einem Term bestehend aus Summanden oder Faktoren, die irgendeine Gesetzmässigkeit aufweisen, eruiert. Und zur Erinnerung: Um zu überprüfen, dass die vereinfachte, explizite Form tatsächlich die Lösung der Rekursionsgleichung ist, kann mit zwei möglichen Methoden überprüft werden, ob diese explizite Darstellung stimmt:

- entweder über vollständige Induktion,
- oder durch die Einsetzungsmethode:
 1. Testen, ob die Anfangsbedingung(en) im expliziten Term die geforderten Werte ergeben, und
 2. Einsetzen des expliziten Terms (also der Lösung) für f links und rechts der Rekursionsgleichung und testen, ob beide Seiten gleich sind.

8.5.2 Lösungsformel für den Fall eines konstanten Koeffizienten $c(n) = c$

Für den Spezialfall, dass der konstante Koeffizient in der Rekursionsgleichung gar nicht von n abhängt, sondern eine Konstante ist, können wir sogar eine einfache explizite Lösungsformel angeben:

§8-30 Satz. (Homogene lineare Rekursionsgleichungen 1. Ordnung mit $c(n) = c$ konstant lösen)

Falls $c(n) = c$ eine konstante Funktion ist, lautet die Lösung der homogenen linearen Rekursionsgleichung mit Anfangswert $a \in \mathbb{R}$:

$$f(n) = a \cdot c^{n-1}.$$

◊

Als Begründung für diese Formel betrachten wir: $f(n) = c \cdot f(n-1) = c \cdot c \cdot f(n-2) = \underbrace{c \cdot c \cdot \dots \cdot c}_{n-1\text{-Mal}} \cdot f(1) = c^{n-1} f(1) = a \cdot c^{n-1}$. □

§8-31 Beispiel. (Homogene lineare Rekursionsgleichung 1. Ordnung mit konstantem Koeffizienten lösen)

Sei die Rekursionsgleichung

$$\begin{aligned} f(1) &= 3 \\ f(n) &= 4f(n-1), \quad n > 1 \end{aligned}$$

gegeben. Mit der Lösungsformel erhalten wir also sofort die explizite Form

$$f(n) = 3 \cdot 4^{n-1} \quad (= \frac{3}{4} 4^n).$$

◊

8.6 Inhomogene lineare Rekursionsgleichungen 1. Ordnung lösen

8.6.1 Iterationsmethode

Auch inhomogene lineare Rekursionsgleichungen lassen sich natürlich prinzipiell mittels der Iterationsmethode lösen und es führt oft auch tatsächlich zum Ziel:

§8-32 Satz. (Inhomogene lineare Rekursionsgleichungen 1. Ordnung lösen)

- **Gegeben:** $f(n) = s(n) + c(n) \cdot f(n-1)$ mit $f(1) = a$ ($a \in \mathbb{R}$)
- **Vorgehen:** $f(n-1), f(n-2), \dots, f(1)$ nach und nach einsetzen und versuchen, das Ergebnis zu vereinfachen.

◊

§8-33 Beispiel. (Türme von Hanoi)

Lösen Sie die Rekursionsgleichung

$$f(n) = 1 + 2 \cdot f(n - 1)$$

mit der Anfangsbedingung $f(1) = 1$ durch die Iterationsmethode.

Lösung:

$$\begin{aligned} f(n) &= 1 + 2f(n - 1) = 1 + 2 \cdot (1 + 2f(n - 2)) = 1 + 2 + 4f(n - 2) \\ &= 1 + 2 + 4 \cdot (1 + 2f(n - 3)) = 1 + 2 + 4 + 8f(n - 3) \\ &= 1 + 2 + 2^2 + 2^3 f(n - 3) = \dots = 1 + 2 + 2^2 + 2^3 + \dots + 2^{n-1} f(1) \\ &= \sum_{i=0}^{n-1} 2^i \\ &= 2^n - 1 \quad (\text{siehe Beispiel §7-14}). \end{aligned}$$

Das entspricht wieder der Formel, welche wir in Beispiel §8-27 gefunden haben. Wir überprüfen diese gefundene explizite Formel noch durch Einsetzen:

$$f(1) = 2^1 - 1 = 2 - 1 \stackrel{?}{=} 1. \quad \checkmark$$

$$f(n) = 2^n - 1 \stackrel{?}{=} 1 + 2(f(n - 1)) = 1 + 2(2^{n-1} - 1) = 1 + 2^n - 2 = 2^n - 1. \quad \checkmark$$

□

◊

Wir haben im Beweis eine schon bewiesene Formel $\sum_{i=0}^n 2^i = 2^{n+1} - 1$ gebraucht. Wir können aber nicht immer davon ausgehen, dass sich die vereinfachte Form so einfach schliessen lässt. Deshalb brauchen wir eine Methode, die unabhängig davon ist, ob wir schon eine Gesetzmässigkeit früher bewiesen haben.

Für (inhomogene) lineare Rekursionsgleichungen 1. Ordnung mit konstanten Koeffizienten und konstantem Störterm gibt es wieder eine Lösungsformel, mit welcher direkt die Lösung einer solchen Rekursionsgleichung bestimmt werden kann:

8.6.2 Lösung einer (inhomogenen) linearen Rekursionsgleichung 1. Ordnung mit konstanten Koeffizienten und Störterm

Wir betrachten nun also folgenden allgemeinen Typ von Rekursionsgleichung:

§8-34 Definition. ((Inhomogene) lineare Rekursionsgleichung k -ter Ordnung mit konstanten Koeffizienten und konstantem Störterm)

Eine Rekursionsgleichung k -ter Ordnung der Form

$$f(n) = s + \sum_{i=1}^k c_{n-i} \cdot f(n - i)$$

für Konstanten $s \in \mathbb{R}$ und $c_{n-i} \in \mathbb{R}$ für alle $i \in \{1, 2, \dots, k\}$ heisst eine (inhomogene) lineare Rekursionsgleichung k -ter Ordnung mit konstanten Koeffizienten. ◊

§8-35 Beispiel.

Somit ist $f(n) = c \cdot f(n-1) + s$ eine (inhomogene) lineare Rekursionsgleichung 1. Ordnung mit konstantem Koeffizienten und konstantem Störterm. \diamond

Im Fall 1. Ordnung haben wir:

§8-36 Satz. ((Inhomogene) lineare Rekursionsgleichungen 1. Ordnung mit konstanten Koeffizienten und Störterm lösen)

Die Lösung einer (inhomogenen) linearen Rekursionsgleichung der Form

$$f(n) = c \cdot f(n-1) + s$$

mit $s, c \in \mathbb{R}$ (also konstant) und der Anfangsbedingung $f(1) = a$ ($a \in \mathbb{R}$) ist

$$f(n) = \begin{cases} a \cdot c^{n-1} + \frac{1-c^{n-1}}{1-c} \cdot s & \text{für } c \neq 1, \\ a + (n-1) \cdot s & \text{für } c = 1. \end{cases}$$

\diamond

§8-37 Beispiel. (Türme von Hanoi)

Für $f(n) = 1 + 2 \cdot f(n-1)$ mit $f(1) = 1$ ist $c = 2$ und $s = 1$, d. h.

$$f(n) = 1 \cdot 2^{n-1} + \frac{1 - 2^{n-1}}{1 - 2} \cdot 1 = 2^{n-1} - (1 - 2^{n-1}) = 2^{n-1} - 1 + 2^{n-1} = 2^n - 1.$$

\diamond

§8-38 Beispiel. (Inhomogene lineare Rekursionsgleichung erster Ordnung mit konstantem Koeffizienten und Störterm lösen)

- (1) Bestimmen Sie die Lösung der Rekursionsgleichung $f(n) = 3 + 10 \cdot f(n-1)$ für die Anfangsbedingung $f(1) = 3$.
- (2) Testen Sie Ihre Lösung.

Lösung: In unserem Beispiel ist also $c = 10$ und $s = 3$.

(1)

$$\begin{aligned} f(n) &= 3 \cdot 10^{n-1} + \frac{1 - 10^{n-1}}{1 - 10} \cdot 3 && (\text{Satz §8-36}) \\ &= 3 \cdot 10^{n-1} - \frac{1}{3}(1 - 10^{n-1}) \\ &= 3 \cdot 10^{n-1} - \frac{1}{3} + \frac{1}{3}10^{n-1} \\ &= \frac{10}{3}10^{n-1} - \frac{1}{3} = \frac{1}{3}10^n - \frac{1}{3} = \frac{1}{3}(10^n - 1). \end{aligned}$$

D.h.,

$$f(n) = \frac{1}{3}(10^n - 1).$$

(2) Test durch Einsetzen:

$$\text{Anfangsbedingung: } f(1) = \frac{1}{3}(10^1 - 1) = 3 \stackrel{?}{=} \underbrace{3}_{\substack{\text{(mit rek.Def.)}}} \quad \checkmark$$

$$\begin{aligned} f(n) &= \frac{1}{3}(10^n - 1) = \frac{1}{3}10^n - \frac{1}{3} \stackrel{?}{=} \underbrace{3 + 10 \cdot f(n-1)}_{\substack{\text{(rek.Def.)}}} \\ &= 3 + 10 \cdot \frac{1}{3}(10^{n-1} - 1) \\ &= \frac{1}{3} \cdot 10^n - \frac{1}{3}. \end{aligned} \quad \checkmark \quad \square$$

◇

8.6.3 Verallgemeinerung: Nicht konstanter Störterm

Nun muss der Störterm $s(n)$ natürlich nicht immer konstant sein, sondern kann eine von n abhängige Funktion sein. Für die Lösung dieses Typs von Gleichungen brauchen wir noch einen zentralen Begriff:

§8-39 Definition. (Zugehörige homogene Rekursionsgleichung)

In einer linearen Rekursionsgleichung k -ter Ordnung mit dem Störterm $s(n)$ und den Funktionen $c_{n-i}(n)$, also

$$f(n) = s(n) + \underbrace{\sum_{i=1}^k c_{n-i}(n) \cdot f(n-i)}_{=:h(n)} \quad (*),$$

nennen wir den Teil $h(n)$ ohne den Störterm $s(n)$, d.h., die Rekursionsgleichung

$$h(n) = \sum_{i=1}^k c_{n-i}(n) \cdot h(n-i)$$

mit den Koeffizienten $c_{n-i}(n)$ wie in (*), die zur inhomogenen Rekursionsgleichung (*) zugehörige homogene Rekursionsgleichung. ◇

§8-40 Beispiele. (Zu inhomogenen Rekursionsgleichungen zugehörige homogene Gleichungen)

(1) Die zur inhomogenen Rekursionsgleichung

$$f(n) = 3 + 2f(n-1)$$

zugehörigen homogenen Rekursionsgleichung ist die Gleichung

$$h(n) = 2h(n-1).$$

(2) Die homogene Rekursionsgleichung, die zu

$$f(n) = 4^{n-1} + 3n^2 f(n-1)$$

gehört, ist gegeben durch

$$h(n) = 3n^2 h(n-1).$$

(3) Zu

$$f(n) = \cos\left(\frac{2\pi}{n+1}\right) + e^{2n^3} f(n-1) + 3f(n-2) - (n^5 - 3n)f(n-3)$$

gehört die homogene Rekursionsgleichung

$$h(n) = e^{2n^3} h(n-1) + 3h(n-2) - (n^5 - 3n)h(n-3).$$

◇

§8-41 Satz. ((Inhomogene) lineare Rekursionsgleichung 1. Ordnung mit c konstant und allgemeinem Störterm $s(n)$ lösen)

Eine (inhomogene) lineare Rekursionsgleichung der Form

$$f(n) = s(n) + c \cdot f(n-1)$$

mit $c \in \mathbb{R}$ (also 1. Ordnung und konstantem Koeffizienten) und mit der Anfangsbedingung $f(1) = a \in \mathbb{R}$ hat die Lösung

$$\begin{aligned} f(n) &= \text{"allg. Lösung der dazugehörigen homogenen Gleichung"} + p(n) \\ &= d \cdot c^{n-1} + p(n) \end{aligned}$$

für ein $d \in \mathbb{R}$, wobei

$$p(n)$$

irgendeine spezielle Lösung (die sogenannte partikuläre Lösung) der **inhomogenen** Rekursion ist, welche zudem **nicht** Lösung der zugehörigen homogenen Gleichung ist.

◇

Wie erhalten wir eine partikuläre Lösung $p(n)$ der inhomogenen Gleichung? Zum Beispiel durch folgende Ansätze^[E.32]:

§8-42 Definition. (Ansätze für partikuläre Lösungen $p(n)$)

Seien $u, t \in \mathbb{R}, n, m \in \mathbb{N} \setminus \{0\}$.

Störterm $s(n)$	Zu wählender Ansatz für $p(n)$
Polynom in n vom Grad m	allgemeines Polynom in n vom Grad m
u^n	$t \cdot u^n$
(Polynom in n vom Grad m) $\cdot u^n$	(allgem. Polynom in n vom Grad m) $\cdot u^n$

◇

Wichtig: Die Polynomkoeffizienten bzw. t in den jeweiligen Ansätzen sind noch zu bestimmen! Diese noch freien Polynomkoeffizienten bzw. Parameter werden bestimmt, indem der gewählte Ansatz $p(n)$ in die Rekursionsgleichung eingesetzt wird. Ein paar Beispiele von konkreten Störtermen $s(n)$ und damit von gewählten Ansätzen sind:

§8-43 Beispiele. (Auswahl der Ansätze für konkrete Störterme)

$s(n)$	Ansatz für $p(n)$	(x, y, z, t, \dots sind noch zu bestimmen)
6	x	
n	$xn + y$	
n^2	$xn^2 + yn + z$	
5^n	$t \cdot 5^n$	
$n \cdot 7^n$	$(xn + y) \cdot 7^n$	
$n^2 \cdot 7^n$	$(xn^2 + yn + z) \cdot 7^n$	

◊

Was ist zu tun, wenn ein Ansatz oder ein Summand des Ansatzes bereits Lösung der homogenen Rekursionsgleichung ist (dann darf man ihn nicht als $p(n)$ benutzen!):

Achtung!

Ist der Ansatz oder ein Summand des Ansatzes bereits Lösung der homogenen Rekursionsgleichung:

Dann muss der Ansatz so oft mit n multipliziert werden, bis dies nicht mehr der Fall ist, also bis der neue Ansatz und auch keiner seiner Summanden Lösung der homogenen Rekursionsgleichung ist.

Zusammenfassend erhalten wir also:

§8-44 Definition. (Schema zur Lösung von $f(n) = s(n) + c \cdot f(n - 1)$)

1. c in $f(n) = d \cdot c^{n-1} + p(n)$ einsetzen.
2. Spezielle Lösung $p(n)$ bestimmen (siehe S. 200 und obige Warnung) und mögliche Parameter x, y, z, t, \dots durch Einsetzen von $p(n)$ in die Rekursionsgleichung bestimmen .
3. Danach d durch Einsetzen der Anfangsbedingung in die Gleichung bestimmen.

◊

§8-45 Beispiel.

Lösen Sie folgende Rekursionsgleichung:

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

mit der Anfangsbedingung $f(1) = 3$. (Nicht verwirren lassen, wenn der Störterm auch mal am Ende steht!)

Lösung:

Wir gehen gemäss Satz §8-41 bzw. dem Schema S. §8-44 vor:

1. $f(n) = d \cdot 3^{n-1} + p(n).$
2. Für den Störterm $s(n)$ wählen wir als Ansatz aus der Tabelle Def. §8-42 ein Polynom 1.

Grades: $p(n) = xn + y$.

Test: Ansatz bzw. keiner seiner Summanden ist Lösung der homogenen Gleichung ✓.

x und y bestimmen ($p(n)$ in die rekursive Definition der Gleichung einsetzen):

$$\begin{aligned} & xn + y = 3(x(n-1) + y) + 2n \\ \Leftrightarrow & xn + y = 3x(n-1) + 3y + 2n \\ \\ \Leftrightarrow & xn + y = 3xn - 3x + 3y + 2n \\ \Leftrightarrow & 0 = 2xn + 2n - 3x + 2y \\ \Leftrightarrow & 0 = \underbrace{(2x+2)n}_{(*)} + \underbrace{(2y-3x)}_{(**)}. \end{aligned}$$

Die rechte Seite der letzten Gleichung kann nur für beliebige n Null sein, wenn (*) und (**) beide Null sind. Diese beiden einfachen Gleichungen nach x und y auflösen (bitte tun Sie das!) liefert $x = -1$ und $y = -\frac{3}{2}$ und damit ist unsere partikuläre Lösung:

$$p(n) = -n - \frac{3}{2}.$$

3. Jetzt bestimmen wir noch d durch Einsetzen der Anfangsbedingung. D.h., wir setzen $f(1) = 3$ in $f(n) = d3^{n-1} - n - \frac{3}{2}$ ein:

$$f(1) = d \cdot 3^0 - 1 - \frac{3}{2} = 3$$

also $d = \frac{11}{2}$.

Damit erhalten wir insgesamt die explizite Lösung für unser Beispiel:

$$f(n) = \frac{11}{2}3^{n-1} - n - \frac{3}{2}.$$

◇

8.7 Homogene lineare Rekursionsgleichungen 2. Ordnung lösen

Wir betrachten nun in diesem Abschnitt homogene lineare Rekursionsgleichungen 2. Ordnung mit konstanten Koeffizienten c_1 und c_2 . Allerdings werden wir in diesem Modul den Fall nicht-konstanter Koeffizienten bei Rekursionsgleichungen 2. Ordnung nicht behandeln. Die Koeffizienten sind also alle immer konstant. Dafür schauen wir uns im anschliessenden Abschnitt noch den inhomogenen Fall mit allgemeinem Störterm $s(n)$ an.

§8-46 Definition. (Homogene lineare Rekursionsgleichungen 2. Ordnung mit konstanten Koeffizienten)

Eine homogene lineare Rekursionsgleichung 2. Ordnung mit konstanten Koeffizienten ist eine Rekursionsgleichung der Form

$$f(n) = c_1 \cdot f(n-1) + c_2 \cdot f(n-2)$$

und zwei Anfangsbedingungen

$$f(1) = a_1,$$

$$f(2) = a_2.$$

◊

§8-47 Beispiel.

Für $c_1 = c_2 = 1$ ist die **Fibonacci-Folge** $f(n) = f(n-1) + f(n-2)$ eine lineare Rekursionsgleichung 2. Ordnung mit konstanten Koeffizienten und Anfangsbedingungen $f(1) = f(2) = 1$, also $a_1 = a_2 = 1$. ◊

Zur Erinnerung: Homogene lineare Rekursionsgleichungen 1. Ordnung mit konstanten Koeffizienten der Form

$$f(n) = c \cdot f(n-1)$$

hatten als Lösung eine Exponentialfunktion $f(n) = a \cdot c^{n-1}$ mit $a = f(1)$ (siehe Satz §8-30). Auch für homogene lineare Rekursionsgleichungen 2. Ordnung

$$f(n) = c_1 \cdot f(n-1) + c_2 \cdot f(n-2) \quad (*)$$

mit konstanten Koeffizienten versuchen wir nun, diesen Ansatz zu verwenden:

Ansatz $f(n) = \lambda^n$ mit $\lambda \neq 0$ in $(*)$ einsetzen:

$$\begin{aligned} \lambda^n &= c_1 \lambda^{n-1} + c_2 \lambda^{n-2} && |: \lambda^{n-2} \\ \lambda^2 &= c_1 \lambda + c_2 \\ \lambda^2 - c_1 \lambda - c_2 &= 0. \end{aligned}$$

Dies ist eine quadratische Gleichung in der Variablen λ und wir können sie mit der bekannten Formel dafür ("Mitternachtsformel") lösen.

Die beiden Lösungen lauten:

$$\lambda_{1,2} = \frac{c_1}{2} \pm \frac{\sqrt{c_1^2 + 4c_2}}{2}.$$

Weil dies quadratische Gleichung von zentraler Bedeutung ist, erhält sie einen eigenen Namen:

§8-48 Definition. (Charakteristische Gleichung)

Die Gleichung

$$\lambda^2 - c_1 \cdot \lambda - c_2 = 0$$

heisst **charakteristische Gleichung** für die Rekursionsgleichung $f(n) = c_1 \cdot f(n-1) + c_2 \cdot f(n-2)$. ◊

- Zusammenfassend also, Lösung der charakteristischen Gleichung:

$$\lambda_1 = \frac{c_1}{2} + \frac{\sqrt{c_1^2 + 4c_2}}{2} \quad \text{und} \quad \lambda_2 = \frac{c_1}{2} - \frac{\sqrt{c_1^2 + 4c_2}}{2}$$

- Fallunterscheidung:

- $c_1^2 + 4c_2 > 0 \rightsquigarrow$ zwei reelle Lösungen λ_1 und λ_2
- $c_1^2 + 4c_2 = 0 \rightsquigarrow$ eine reelle Lösung $\lambda (= \lambda_1 = \lambda_2)$
- $c_1^2 + 4c_2 < 0 \rightsquigarrow$ keine reelle Lösung

Damit lässt sich zeigen:

§8-49 Satz. (Homogene lineare Rekursionsgleichungen 2. Ordnung mit konstanten Koeffizienten lösen)

Die Lösung einer homogenen linearen Rekursionsgleichung der Form

$$f(n) = c_1 \cdot f(n-1) + c_2 \cdot f(n-2)$$

mit $c_1, c_2 \in \mathbb{R}$ und den Anfangsbedingungen $f(1) = a_1$ und $f(2) = a_2$ ($a_1, a_2 \in \mathbb{R}$) ist bestimmt durch die Lösung der charakteristischen Gleichung $\lambda^2 - c_1 \cdot \lambda - c_2 = 0$:

- für $c_1^2 + 4c_2 > 0$:

$$f(n) = d_1 \cdot \lambda_1^n + d_2 \cdot \lambda_2^n$$

- für $c_1^2 + 4c_2 = 0$:

$$f(n) = (d_1 + d_2 \cdot n) \lambda^n$$

d_1 und d_2 lassen sich dann noch durch Einsetzen der Anfangsbedingungen in die Lösung bestimmen. \diamond

§8-50 Beispiel. (Fibonacci-Folge)

Finden Sie die explizite Darstellung der Fibonacci-Folge [E.33]

$$f(n) = f(n-1) + f(n-2)$$

$$f(1) = 1$$

$$f(2) = 1$$

mit Hilfe der charakteristischen Gleichung (d.h., mit Hilfe von Satz §8-49).

Lösung:

Die charakteristische Gleichung der Fibonacci-Rekursionsgleichung lautet:

$$\lambda^2 - \lambda - 1 = 0.$$

Die Lösungen dafür lauten

$$\lambda_{1,2} = \frac{1}{2} \pm \frac{\sqrt{1+4}}{2} = \frac{1}{2} \pm \frac{\sqrt{5}}{2} = \frac{1 \pm \sqrt{5}}{2}.$$

Also hat die explizite Darstellung die Form

$$f(n) = d_1 \cdot \left(\frac{1 + \sqrt{5}}{2} \right)^n + d_2 \cdot \left(\frac{1 - \sqrt{5}}{2} \right)^n$$

wobei wir die Konstanten d_1 und d_2 noch durch Einsetzen der Anfangsbedingungen bestimmen:

$$f(1) = d_1 \frac{1+\sqrt{5}}{2} + d_2 \frac{1-\sqrt{5}}{2} \stackrel{\text{nach Anf.bed.}}{=} 1$$

$$f(2) = d_1 \left(\frac{1+\sqrt{5}}{2}\right)^2 + d_2 \left(\frac{1-\sqrt{5}}{2}\right)^2 \underset{\text{nach Anf. bed.}}{\stackrel{=}{\underbrace{1}}}$$

Also (nach einer Detailrechnung):

$$\begin{aligned} d_1 &= \frac{1}{\sqrt{5}} \\ d_2 &= -\frac{1}{\sqrt{5}}. \end{aligned}$$

Damit erhalten wir die explizite Formel zu:

$$f(n) = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2}\right)^n,$$

welche identisch ist mit derjenigen, die wir schon früher bewiesen haben. Jetzt kennen wir also eine Methode, wie man dort auf diese Vermutung gekommen ist! ◇

8.8 Inhomogene lineare Rekursionsgleichungen 2. Ordnung lösen

Es lässt sich ebenso eine Lösungsformel herleiten für den inhomogenen Fall:

§8-51 Satz. ((Inhomogene) lineare Rekursionsgleichung 2. Ordnung mit c_1, c_2 konstant und allgemeinem Störterm $s(n)$ lösen)

Die Lösung einer (inhomogenen) linearen Rekursionsgleichung der Form

$$f(n) = s(n) + c_1 \cdot f(n-1) + c_2 \cdot f(n-2)$$

mit $c_1, c_2 \in \mathbb{R}$ und den Anfangsbedingungen $f(1) = a_1$ und $f(2) = a_2$ mit $a_1, a_2 \in \mathbb{R}$ ist

$$\begin{aligned} f(n) &= \text{"allg. Lösung der dazugehörigen homogenen Gleichung"} + p(n) \\ &= h(n) + p(n), \end{aligned}$$

wobei $p(n)$ irgendeine partikuläre Lösung der inhomogenen Rekursion ist. Zur Bestimmung der partikulären Lösung kann der gleiche Ansatz wie für Gleichungen 1. Ordnung verwendet werden (siehe S. 200). ◇

§8-52 Beispiel.

Lösen Sie die folgende inhomogene lineare Rekursionsgleichung 2. Ordnung:

$$\begin{aligned} f(n) &= 4 \cdot f(n-1) - 4 \cdot f(n-2) + 2^n \\ f(1) &= 1 \\ f(2) &= 8 \end{aligned}$$

Lösung: → Übungsblatt. ◇

8.9 Variablentransformation

8.9.1 Einleitung

Die Analyse von Algorithmen – also die Bestimmung der Größenordnungen von Laufzeit, benötigtem Speicherplatz und anderen Ressourcen von Algorithmen – ist für Informatiker eine sehr wichtige Anwendung von Rekursionsgleichungen. Hierbei hat man nicht immer “so schöne” Gleichungen wie eben kennengelernt und man braucht noch andere Methoden. Eine wichtige Methode – die Variablentransformation – wollen wir jetzt anhand einer solchen Anwendung noch kurz kennenlernen. (Dies ist aber nur im Sinn einer kleinen Einführung zu verstehen.)

Einordnung dieser Methode:

- Achtung: Schon lineare Rekursionsgleichungen können schnell ziemlich schwierig zu lösen sein, auch mit Variablentransformationen (diese zu finden kann kniffig sein!).
- Deshalb begnügt man sich in der Praxis bei der Analyse von Algorithmen meist mit asymptotischen Größenordnungsschätzungen (Stichwort: Landau-Symbole, “O-Notation”) → siehe nachfolgende Module, z.B. algd1.
- Eine wichtige Methode ist dabei das Master Theorem, welches auf der Idee von Variablentransformationen basiert → in späteren Modulen.

Zur Einführung werden wir den rekursiven “Mergesort”-Algorithmus, ein sogenannter “divide-and-conquer”-Algorithmus, betrachten.

8.9.2 Variablentransformationen in Funktionen auf abzählbaren Mengen

Wir betrachten im folgenden Funktionen $\phi : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$. Als kurze Repetition halten wir folgende Begriffe fest:

§8-53 Definition. (Surjektive, injektive, bijektive Funktion $\phi : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$ auf abzählbarem M)

Sei $\phi : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$ eine Funktion, also auf einer abzählbaren Menge M .

(1) ϕ heisst **surjektiv**

$$\begin{aligned} &\iff \forall y \in \mathbb{R} \exists x \in M : \phi(x) = y \\ &\iff \phi(M) = \mathbb{R}. \end{aligned}$$

(2) ϕ heisst **injektiv**

$$\begin{aligned} &\iff \forall x_1, x_2 \in M : (x_1 \neq x_2 \implies \phi(x_1) \neq \phi(x_2)) \\ &\iff \forall x_1, x_2 \in M : (\phi(x_1) = \phi(x_2) \implies x_1 = x_2). \end{aligned}$$

(3) ϕ heisst **bijektiv**

$$\begin{aligned} &\iff \phi \text{ ist surjektiv und injektiv} \\ &\iff \forall y \in \mathbb{R} \exists !x \in M : y = \phi(x). \end{aligned}$$

◇

Beachten Sie: Wenn wir Funktionen $\phi : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$ betrachten, dann sehen wir sofort, dass diese nie surjektiv und damit auch nie bijektiv sein können! (\mathbb{N} ist ja abzählbar und

\mathbb{R} überabzählbar.) Die Einschränkung des Wertebereichs einer solchen Funktion auf den Bildbereich, d.h., die Funktion

$$\begin{aligned}\phi |^{\phi(M)} : M \subseteq \mathbb{N} &\rightarrow \phi(M) \\ x &\mapsto \phi |^{\phi(M)}(x) := \phi(x)\end{aligned}$$

ist aber immer (trivialerweise) surjektiv! Damit ist die Einschränkung $\phi |^{\phi(M)}$ einer injektiven Funktion $\phi : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$ natürlich immer automatisch schon bijektiv!

Zentral ist auch der folgende Begriff, den Sie wahrscheinlich schon kennen:

§8-54 Definition. (Inverse (Umkehrfunktion) einer bijektiven Funktion)

Ganz allgemein lässt sich einer beliebigen bijektiven Funktion $f : A \rightarrow B$ eindeutig ihre Inverse (Umkehrfunktion) zuordnen, d.h., die Funktion

$$f^{-1} : B \rightarrow A, \quad y \mapsto f^{-1}(y) := x$$

wobei $x \in A$ genau so, dass $f(x) = y$. ◊

Damit können wir nun den zentralen Begriff definieren, welcher der Methode zugrunde liegt:

§8-55 Definition. (Variablentransformation (Variablensubstitution))

Sei $\phi : M \subseteq \mathbb{N} \rightarrow \phi(M)$ eine bijektive Funktion auf einer abzählbaren Menge M und $f : \phi(M) \subseteq \mathbb{N} \rightarrow \mathbb{R}$ eine andere Funktion definiert auf dem Bild von ϕ . Dann nennen wir die Verknüpfung

$$\tilde{f} := f \circ \phi,$$

eine Variablentransformation (Variablensubstitution)^[E.34] in / von f (bzgl. ϕ) und die Funktion

$$\begin{aligned}\tilde{f} : M \subseteq \mathbb{N} &\rightarrow \tilde{f}(M) \quad (= f(\phi(M))) \\ x &\mapsto \tilde{f}(x) := f(\phi(x))\end{aligned}$$

die bzgl. ϕ transformierte Funktion. Da \tilde{f} wieder bijektiv ist, existiert ihre Umkehrfunktion

$$\begin{aligned}\tilde{f}^{-1} : \tilde{f}(M) &\rightarrow M \\ x &\mapsto \tilde{f}^{-1}(x),\end{aligned}$$

welche wir die Rücktransformation (Rücksubstitution) nennen. ◊

Bei der Analyse von Algorithmen werden meist Variablen symbole k, l, m, n oder auch i, j benutzt, da es sich um natürliche Zahlen handelt. Wir übernehmen im folgenden diese Schreibweise. Zur Verdeutlichung, dass es bei der Variablentransformation um eine Variablen substitution bzw. Variablenersetzung ("Substitution" = "Ersetzung") geht, verwenden wir oft für das Argument von ϕ eine andere Variable, z.B. k , wenn wir für f die Variable n verwendet haben. Dann erkennt man gut, dass das n in f eben durch das k in ϕ ersetzt wird.

Und insbesondere verwenden wir dann $n = \phi(k)$ als die Variable für den Funktionswert der Transformation ϕ . (Siehe auch die folgenden Beispiele und das Beispiel zur Laufzeit-Analyse von MergeSort.)

Schauen wir uns zuerst einmal ein konkretes Beispiel einer Variablentransformation an:

§8-56 Beispiele. (Variablentransformationen für Funktionen $f : M \subseteq \mathbb{N} \rightarrow \mathbb{R}$)

- (1) Sei $G := \{2k \mid k \in \mathbb{N}\}$ die Menge aller geraden natürlichen Zahlen. Die Funktion

$$\phi : \mathbb{N} \rightarrow G, \quad k \mapsto n = \phi(k) := 2k$$

ist bijektiv mit Umkehrfunktion

$$\phi^{-1} : G \rightarrow \mathbb{N}, \quad n \mapsto k = \phi^{-1}(n) := \frac{1}{2}n.$$

ϕ kann also als Variablentransformation benutzt werden, welche die Variable n durch der Variablen k ($:= \frac{1}{2}n$) substituiert.

- (2) Sei $G := \{2^k \mid k \in \mathbb{N}\}$ die Menge aller Zweierpotenzen mit positiver, ganzzahliger Potenz. Die Funktion

$$\phi : \mathbb{N} \rightarrow G, \quad k \mapsto n = \phi(k) := 2^k$$

ist bijektiv mit Umkehrfunktion

$$\phi^{-1} : G \rightarrow \mathbb{N}, \quad n \mapsto k = \phi^{-1}(n) := \log_2(n).$$

ϕ kann damit wieder als Variablentransformation benutzt werden, welche die Variable n durch der Variablen k ($:= \log_2(n)$) substituiert.

◊

Im folgenden Abschnitt wollen wir nun studieren, wie wir diese Methode zur Analyse von Algorithmen anwenden.

8.9.3 Variablentransformation zur Analyse von Algorithmen

Zur Illustration der Methode betrachten wir folgenden – wahrscheinlich wohlbekannten – Algorithmus:

§8-57 Definition. (Mergesort)

- Sei $\mathbf{a} := (a_1, a_2, \dots, a_n)$ eine Liste von Objekten, auf welchen eine Totalordnung definiert ist. Sie können sich der Einfachheit halber vorstellen, es handle sich um natürliche Zahlen.
- Sei **split(d, a)** eine einfache Funktion mit vernachlässigbarem Aufwand, welche für **d=1** (**l** = “links”) die linke Hälfte (bis auf höchstens ein Element Unterschied) einer Liste **a** liefert und für **d=r** (**r** = “rechts”) den rechten Teil.
- Sei ferner **merge(l₁, l₂)** eine Funktion, welche mit vernachlässigbarem Aufwand zwei geordnete (Teil)-Listen im Reissverschlussverfahren zu einer geordneten Liste zusammenfügt und zurückliefert.

Algo.:

1. **mergesort(a)**
2. **wenn** $|a| \leq 1$ **dann**
3. **return a**
4. **sonst**

```

5.      b1 = mergesort(split(l, a))
6.      b2 = mergesort(split(r, a))
7.      return merge(b1, b2)

```

◊

Es lässt sich nun folgender Satz zeigen:

§8-58 Satz. (Laufzeit von Mergesort)

Für eine Liste der Länge n benötigt der Mergesort-Algorithmus eine Laufzeit $t(n)$ von

$$t(n) = 2t\left(\frac{n}{2}\right) + n.$$

◊

Sie müssen sich oben jetzt das t als das bisherige f in den Rekursionsgleichungen denken. Bei Algorithmen braucht man meist T oder t wegen dem englischen "time". (Entsprechend s für Speicherplatz, engl. "space" oder "storage".)

Beachten Sie: Obige Rekursionsgleichung lässt sich mit den bisherigen Methoden für Rekursionsgleichungen nicht lösen, da wir einen Bruch im Argument von t auf der rechten Seite haben!

(Dies ist noch ein sehr einfacher Fall zur Demonstration. In der Praxis können kompliziertere "Komplikationen" auftreten, welche teilweise auch dann durch Variablentransformation gelöst werden können.)

Die Idee der Variablentransformation ("Variablensubstitution") erkennen wir nun anhand des folgenden Schemas:

§8-59 Satz. (Rekursionsgleichung lösen mittels Variablentransformation)

- (1) Wir "transformieren" mittels $k \mapsto \phi(k)$ (der Variablentransformation) die Variable n der Funktion $n \mapsto t(n)$ in eine Variable k der transformierten Funktion $\tilde{t}(k) := (t \circ \phi)(k)$.
(Oder anders betrachtet, wir "substituieren" / "ersetzen" n mit k und betrachten stattdessen \tilde{t}).
- (2) Wir lösen die so erhaltene neue Rekursionsgleichung in \tilde{t} und mit Argument k mit uns bekannten Methoden auf.
- (3) Wir "rücktransformieren" ("rücksubstituieren") das Ergebnis so, dass wir eine Lösung des ursprünglichen Problems erhalten. Das können wir immer erreichen, da ϕ ja eine Bijektion ist.

◊

Die Lösung der Rekursionsformel für die Laufzeit des Mergesort-Algorithmus mithilfe dieser Idee wollen wir nun schrittweise studieren. Zur Vereinfachung schränken wir unsere Listenlänge n so ein, dass sie immer eine Zweierpotenz sei, also

$$n = 2^k$$

für ein $k \in \mathbb{N}$. So müssen wir beim rekursiven Aufteilen der Listen nicht die $\lfloor \dots \rfloor$ -Funktion für die Anzahl Elemente einführen, welche die Analyse unnötig erschwert.

§8-60 Beispiel. (Lösung der Mergesort-Rekursionsgleichung durch Variablentransformation)

Gegeben: Rekursionsgleichung: (*) $t(n) = 2t(\frac{n}{2}) + n$.

Gesucht: Lösung bzw. Ermittlung der expliziten Darstellung mittels Variablentransformation und bekannter Methoden.

1. Variablentransformation (Substitution) festlegen:

$$k \mapsto \phi(k) = 2^k \quad (= n), \quad \text{d.h., } k = \log_2(n).$$

Damit ist (sehr ausführlich dargestellt)

$$\begin{aligned} (\text{**}) \quad \tilde{t}(k) &= (t \circ \phi)(k) = t(\phi(k)) = t(n) = t(2^k) \\ &\stackrel{(*)}{=} 2t\left(\frac{2^k}{2}\right) + 2^k = 2t(2^{k-1}) + 2^k \end{aligned}$$

nach Def. der Variablentransformation und der Rekursionsgleichung. Damit erhalten wir die transformierte Rekursionsgleichung in der substituierten Variable k als

$$\begin{aligned} \tilde{t}(k) &\stackrel{(**)}{=} 2t(2^{k-1}) + 2^k \\ &\stackrel{(**)}{=} 2\tilde{t}(k-1) + 2^k, \end{aligned}$$

d.h., wir lösen die transformierte Rekursionsgleichung

$$(\text{***}) \quad \tilde{t}(k) = 2\tilde{t}(k-1) + 2^k.$$

2. Die transformierte Rekursionsgleichung (***)) lässt sich mittels Iterationsmethode Satz L.18-1 und vollständiger Induktion oder mittels Satz L.18-10 bzw. dem Schema auf S.11 in L.18 lösen.

(Anfangsbedingung ist aufgrund des Mergesort-Algorithmus' $t(1) = 1$ (bzw. konstant, wir setzen $c := 1$) und damit ist die transformierte Anfangsbedingung $\tilde{t}(1) = t(2^1) = t(2) = 2t(\frac{2}{2}) + 2 = 2t(1) + 2 = 4$.)

Die Lösung lautet:

$$(\text{****}) \quad \tilde{t}(k) = 2^k + k2^k.$$

3. Rücktransformation: Wegen $n = 2^k \iff k = \log_2(n)$ erhalten wir aus (****) durch "Rückersetzen" der Variablen die Lösung der Original-Rekursionsgleichung

$$\begin{aligned} t(n) &= 2^{\log_2(n)} + \log_2(n)2^{\log_2(n)} \\ &= n + \log_2(n) \cdot n, \end{aligned}$$

d.h., die Laufzeit von Mergesort ist:

$$t(n) = n + \log_2(n) \cdot n. \quad (\in O(n \cdot \log_2(n))).$$

◇

Wir fassen dieses Ergebnis noch als Satz zusammen:

§8-61 Satz. (Explizite Formel für die Laufzeit des MergeSort-Algorithmus')

Für eine ungeordnete Liste von 2^n , $n \in \mathbb{N}$, Elementen ist die Laufzeit des MergeSort-Algorithmus' (unter Vernachlässigung konstanter Terme) gegeben durch

$$t(n) = n + \log_2(n) \cdot n. \quad (\in O(n \cdot \log_2(n))).$$

◊

8.9.4 Ausblick: Erzeugendenfunktion, Master-Theorem

Die Idee der Variablentransformation kann ausgedehnt werden, so dass sie in einer ganzen Anzahl von Fällen, welche bei der Analyse von Algorithmen wichtig sind, zur Auflösung von Rekursionen angewendet werden kann. Ein Hauptergebnis hierzu ist dann das bekannte Master Theorem, welches Sie in späteren Modulen kennenlernen werden. Eine weitere sehr hilfreiche Idee zur Auflösung von Rekursionen stammt aus den Gebieten der sogenannten analytischen und abzählenden Kombinatorik: Wir können jeder Folge^[E.35] $f : \mathbb{N} \cup \{0\} \rightarrow \mathbb{R}$ nämlich eine sogenannte Erzeugenden“funktion” (genauer: eine “Potenzreihe”)

$$G(f; x) := \sum_{n=0}^{\infty} f(n)x^n$$

zuordnen^[E.36]. Und mit dieser haben wir dann einige mächtige Werkzeuge zur Verfügung um Rekursionen zu lösen, d.h., aus der rekursiven Darstellung die explizite (funktionale) zu bestimmen.

Eine “kleine” alphabetisch geordnete Liste mit weiterführender Lit. dazu (ACHTUNG: Geht teilweise sehr weit!) sei hier noch angeführt: Z.B. [1], [2], [6], [8], [9], [10], [12], [13], [14], [16], [18], [19], [20]. Und natürlich auch Donald Knuth's TAoCP (“The Art of Computer Programming”; mehrere Bände).

In der Mathematik werden übrigens “Rekursionsgleichungen” auch “Dierenzengleichungen” (im weiteren Sinn) genannt. Sie werden also auch unter diesem Stichwort viele weiterführende Literatur finden. Der englische Begriff für Rekursionsgleichungen lautet übrigens “recurrence relation” bzw. “difference equation”.

8.10 Übersicht über die kennengelernten Lösungsmethoden für Rekursionsgleichungen

- Zur Erinnerung: Alle folgende Rekursionsgleichungen sind linear.

Typ der Rekursionsgl.:	Form der Rekursionsgl.:	Lösungsmethode(n):
Homogene Rekursionsgl. 1. Ordnung	(\sim siehe Satz §8-29)	Iterationsmethode (\sim siehe Satz §8-29) und vollst. Ind./Einsetzungsmethode
Homogene Rekursionsgl. 1. Ordnung mit $c = c(n)$ konst.	(\sim siehe Satz §8-30)	Lösungsformel (basierend auf Iterationsmethode \sim siehe Satz §8-30)
Inhomogene Rekursionsgl. 1. Ordnung	(\sim siehe Satz §8-32)	Iterationsmethode (\sim siehe Satz §8-32) und vollst. Ind./Einsetzungsmethode
Inhomogene Rekursionsgl. 1. Ordnung mit konstanten $c = c(n)$ und $s = s(n)$	(\sim siehe Satz §8-36)	Lösungsformel (\sim siehe Satz §8-36)
Inhomogene Rekursionsgl. 1. Ordnung mit konstanten $c = c(n)$ und allgemeinem $s(n)$	(\sim siehe Satz §8-41)	Lösungsformel (\sim siehe Satz §8-41 und Schema §8-44) und Ansätze für $p(n)$ (\sim siehe Def. §8-42) Anm. S.229 beachten!
Homogene Rekursionsgl. 2. Ordnung mit konstanten $c_1 = c_1(n)$ und $c_2 = c_2(n)$	(\sim siehe Def §8-46)	Lösungsformel (\sim siehe Satz §8-49 inkl. Def. §8-48)
Inhomogene Rekursionsgl. 2. Ordnung konstanten $c_1 = c_1(n)$ und $c_2 = c_2(n)$ und allgemeinem $s(n)$	(\sim siehe Satz §8-51)	Lösungsformel (\sim siehe Satz §8-51) (basierend auf Lösungsformel für $h(n)$ \sim siehe Satz §8-49 und Ansätze für $p(n)$ \sim siehe Def. §8-42) Anm. S.229 beachten!

- Für Rekursionsgleichungen beliebiger Form (§8-16/§8-17, §8-21, §8-23, ... usw.):
 - **Iterationsmethode und vollst. Induktion / Einsetzungsmethode:** Vorgehen S.214.
 - **Variablentransformation:** §8-58.

Teil IV

Formale Sprachen, endliche Automaten, Grammatiken

Kapitel 9

Einführung: Formale Sprachen

Kapitelinhalt

9.1	Motivierung	217
9.2	Alphabete und Wörter	217
9.3	Wortpotenzen	221
9.4	Sprachen	222
9.5	Entscheidungsproblem einer Sprache (Wortproblem)	223

L E R N Z I E L E :

LZ 9.1 Nach Durcharbeiten dieses Kapitels sollen Sie den Begriff einer formalen Sprache verstehen und damit zusammenhängend die wichtigsten Grundkonzepte kennen.

LZ 9.2 Sie sollen wissen, was das Entscheidungsproblem (Wortproblem) einer formalen Sprache ist.

Im folgenden Teil IV beinhaltet die Menge der natürlichen Zahlen wiederum die Null. Damit kein Missverständnis auftritt, notieren einige dies oft zur Hervorhebung auch mit \mathbb{N}_0 (diese Schreibweise wollen wir vermeiden) oder $\mathbb{N} \cup \{0\}$. Also:

$$\mathbb{N} := \mathbb{N}_0 := \mathbb{N} \cup \{0\} := \{0, 1, 2, 3, 4, \dots\}.$$

Auch wenn sie etwas länger ist, bevorzuge ich dabei die Notation $\mathbb{N} \cup \{0\}$, da wir ja oft \mathbb{N}_n als die Menge der natürlichen Zahlen bis und mit einer natürlichen Zahl n bezeichnen. Dies kann zu Verwirrung führen.

9.1 Motivierung

Stellen Sie sich vor, Sie haben ein Computer-Programm in der Programmiersprache Ihrer Wahl geschrieben. Und stellen Sie sich vor, es ist ein ganz schönes Stück Code geworden (hunderte, tausende, oder gar mehr Zeilen). Können Sie automatisch überprüfen, ob Ihr Code syntaktisch korrekt ist und ein Computer tatsächlich jede einzelne Zeile "versteht" [E.37] ?

Die Antwort ist: Ja, das geht! Jedoch brauchen wir dafür eine formale Definition Ihrer Programmiersprache, beispielsweise als "formale Grammatik".

Die nächste Frage wird nun sein, wie wir einen solchen "Syntax-Test-Algorithmus" selber wieder beschreiben und implementieren können. Die Antwort hierfür wird uns zum nächsten fundamentalen Begri dieses Kapitels führen, demjenigen eines "erkennenden" oder "akzeptierenden Automaten" für die Sprache.

Die eben skizzierte Fragestellung ist nur eine von vielen interessanten Problemen, welche wir mit den Begri en und Methoden dieses Kapitels lösen können. Noch tiefer führt es uns sogar auf die sehr grundlegende Frage, was den eigentlich "rechnen" bzw. "berechnen" bedeutet und was ganz prinzipiell eine "Rechenmaschine" – egal wie technisch oder biologisch (Gehirn) realisiert – ist.

Eine Hauptinspiration zur Lösung solcher Probleme stammt aus dem Studium natürlicher Sprachen, welche ja ebenso ein Medium zur Darstellung, Übertragung, Speicherung und Verarbeitung von Information sind. Jedoch haben wir die Zusatzbedingung zu erfüllen, dass alle Ideen, Begri e und Methoden vollständig formal vorliegen müssen. (Sie sollen ja präzise, unzweideutig und "maschinenverarbeitbar" sein.) Deshalb studieren wir "formale Sprachen", "formale Grammatiken" und die Verarbeitungsmaschinen sind (formale) "Automaten", wie zum Beispiel Computer verschiedenster technischer Realisierungen [E.38].

Im Folgenden werden wir diese Begri e und Methoden studieren, und zwar schwerpunktmässig für eher begrenzte Fälle. In weiterführenden Veranstaltungen werden wir dann zu zunehmend allgemeineren Fällen fortschreiten, was nicht mehr Thema dieses Moduls sein kann. Aber wir werden diesen Weg zumindest skizzieren und die berühmte "Church-Turing-Theorie" noch kurz ansprechen. (Diese liefert eine Antwort auf die allgemeinste Frage, was ganz prinzipiell "berechnen" bedeutet und was eine ganz allgemeine "Rechenmaschine" ist.)

Jede Sprache, sei sie eine natürliche oder eine "formale" wie zum Beispiel eine Programmiersprache, benutzt ein Alphabet von Symbolen, welche die fundamentalsten Bausteine der Sprache und der "Informationsträger" darstellen. Damit wollen wir nun beginnen.

9.2 Alphabete und Wörter

§9-1 Definition. (Alphabet)

Ein Alphabet Σ ist eine endliche, nichtleere Menge von Symbolen.



§9-2 Beispiele. (Alphabet)

- (1) $\Sigma := \{a, b, c\}$ ist die Menge der drei Symbole a , b und c .
- (2) $\Sigma := \{-, +, \cdot, :\}$ ist die Menge der Symbole für die Grundrechenarten.
- (3) $\Sigma_{\text{Bool}} := \{0, 1\}$ ist das Boolesche Alphabet.
- (4) $\Sigma_{\text{lat}} := \{a, b, c, \dots, z\}$ ist die Menge der lateinischen Kleinbuchstaben.
- (5) $\Sigma_{\text{Lat}} := \{A, B, C, \dots, Z\}$ ist die Menge der lateinischen Grossbuchstaben.
- (6) \mathbb{N} ist *kein* Alphabet (da eine unendliche Menge).



§9-3 Definition. (Wort über einem Alphabet)

Ein Wort w (Zeichenfolge, Zeichenreihe, String) über einem Alphabet Σ ist eine endliche Folge von Symbolen aus diesem Alphabet.



§9-4 Beispiele. (Wort über einem Alphabet)

- (1) $w := abc$ ist ein Wort über dem Alphabet Σ_{lat} (oder über $\Sigma = \{a, b, c\}$).
- (2) $w := 100111$ ist ein Wort über dem Alphabet $\{0, 1\}$.



Bitte beachten Sie: Man sagt Wort w über dem Alphabet Σ , wenn w aus Symbolen des Alphabets Σ besteht. In einer Folge werden weiter normalerweise die einzelnen Symbole durch Kommata getrennt. Wenn klar ist, was die einzelnen Elemente sind und keine Missverständnisse entstehen können, lassen wir die Kommata der Einfachheit halber weg. Wenn wir also beispielsweise ein Alphabet $\Sigma := \{0, 1\}$ haben, können wir Wörter wie 10110 durchaus ohne Kommata schreiben (also anstatt $w := (1, 0, 1, 1, 0)$). Wenn aber das Alphabet zum Beispiel $\Sigma := \{0, 1, 00, 11\}$ ist, dann können wir dies nicht tun, weil das Wort 11 kann über diesem Alphabet dem Wort bzw. der Folge (11) oder $(1, 1)$ entsprechen. Im ersten Fall handelt es sich um das einsymbolische Wort mit dem Symbol 11 des Alphabets und im zweiten Fall um das zweisymbolische Wort mit zweimal demselben Symbol 1 des Alphabets. Dies müssen wir unterscheiden können! Natürlich will man solche Zweideutigkeiten meist vermeiden und Wörter tatsächlich ohne Kommata schreiben können. Deshalb wird man eine entsprechende Symbolmenge wählen. Dies ist etwa beim hexadezimalen Zahlensystem der Fall: Anstatt die Symbole $0, 1, 2, \dots, 9, 10, 11, 12, 13, 14, 15$ wählt man $0, 1, 2, \dots, 9, A, B, C, D, E, F$.

§9-5 Definition. (leeres Wort)

Das leere Wort ε ((klein) "epsilon") ist ein Wort, das keine Symbole enthält.
 (Selten auch mit λ ((klein) "lambda") bezeichnet.) \diamond

Beachten Sie: Das leere Wort ε ist ein Wort über jedem Alphabet.

§9-6 Definition. (Länge eines Wortes)

Die Länge $|w|$ eines Wortes w ist die Länge des Wortes als Folge, also die Anzahl der Folgenglieder in der Folge (d. h. die Anzahl aller Vorkommnisse von Symbolen des Alphabets). \diamond

§9-7 Beispiele. (Wortlängen)

- (1) Über $\Sigma := \{a, b, c\}$ ist $|abc| = 3$.
- (2) Über $\Sigma := \{0, 1\}$ ist $|100111| = 6$.
- (3) Über einem beliebigen Alphabet Σ ist $|\varepsilon| = 0$.
- (4) Über dem Alphabet des ASCII-Zeichensatzes ist

$$|\text{Informatik ist spannend}| = 23$$

(Leerzeichen sind auch Symbole!). \diamond

§9-8 Definition. (Konkatenation (Verkettung) von Wörtern)

Seien x und y zwei beliebige Wörter. Dann steht $x \cdot y = xy$ für die Konkatenation (Verkettung) von x und y . \diamond

§9-9 Beispiel. (Konkatenation (Verkettung) von Wörtern)

Seien $x := 01001$ und $y := 110$ zwei Wörter, dann ist $xy = 01001110$ die Konkatenation dieser beider Wörter. \diamond

Beachten Sie, dass also in diesem Zusammenhang das " \cdot "-Zeichen nicht etwa eine algebraische Multiplikation, sondern die Aneinanderhängung zweier Wörter bedeutet [E.39].

Sei $x := a_1a_2 \dots a_i$ ein aus i Symbolen bestehendes Wort und sei $y := b_1b_2 \dots b_j$ ein aus j Symbolen bestehendes Wort. Dann hat das Wort

$$xy = a_1a_2 \dots a_i b_1 b_2 \dots b_j$$

die Länge $i + j$. Für jedes beliebige Wort w gilt weiter, dass $w\varepsilon = \varepsilon w = w$.

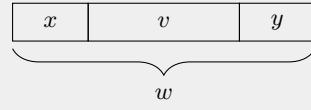
Im Zusammenhang mit Wörtern über einem Alphabet, haben wir auch folgende zentrale Grundbegriffe:

§9-10 Definition. (Teilwort (Infix), Präfix, Suffix)

Sei im folgenden w irgendein Wort über irgendeinem Alphabet Σ .

- (1) Wir sagen, dass v ein **Teilwort (Infix)** von w ist, wenn man w als

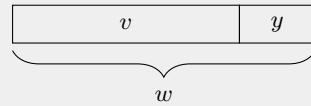
$$w = xvy$$



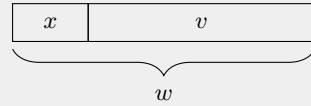
für beliebige Wörter x und y über Σ schreiben kann.

- (2) Ein **echtes Teilwort** von w ist jedes Teilwort von w , das kürzer als w ist.

- (3) Ein Wort v ist ein **Präfix** von w , wenn $w = vy$ gilt für irgendein Wort y .



- (4) Ein Wort v ist ein **Suffix** von w , wenn $w = xv$ gilt für irgendein Wort x .



◊

§9-11 Beispiele. (Teilwort (Infix), Präfix, Suffix)

- (1) *aba* ist ein echtes Teilwort von *babaab*.
- (2) "Prä" ist ein Präfix des Wortes "Präfix". :-)
- (3) "u x" ist ein Suffix des Wortes "Su x". :-)

◊

§9-12 Definition. (Mengen von Wörtern über einem Alphabet; Binärwörter)

- (1) Die Menge aller Wörter der Länge $k \in \mathbb{N} \cup \{0\}$ über einem Alphabet Σ wird mit Σ^k bezeichnet.
- (2) Die Menge aller Wörter beliebiger, aber endlicher Länge über einem Alphabet Σ heisst die Kleenesche Hülle von Σ und wird mit Σ^* bezeichnet.
- (3) $\Sigma^+ := \Sigma^* \setminus \{\varepsilon\}$ ist die Menge aller nichtleeren Zeichenreihen (Wörter) über einem Alphabet Σ .
- (4) Wörter aus $\{0, 1\}^*$ nennt man **Binärwörter** und $\{0, 1\}^*$ ist damit die Menge der Binärwörter.

◊

Bitte beachten Sie unbedingt, dass ein Alphabet immer eine endliche Menge von Symbolen ist und ein Wort immer aus einer endlichen Folge von Symbolen besteht. Hingegen ist die Menge aller möglicher (endlicher) Wörter über einem (endlichen, nicht-leeren) Alphabet, d.h., die Kleenesche Hülle, immer eine unendliche Menge, da jedes Wort zwar endlicher, aber beliebiger Länge ist! In der Kleeneschen Hülle gibt es für jede natürliche Zahl n ein Wort mit $|w| = n$, sofern das Alphabet nicht leer ist!

§9-13 Beispiele. (Mengen von Wörtern über einem Alphabet)

- (1) Über $\Sigma := \{a, b, c\}$ ist $\Sigma^2 = \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$ die Menge aller Wörter der Länge 2.
- (2) Über $\Sigma := \{0, 1\}$ ist $\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$ die Menge aller Wörter der Länge 3.
- (3) Für $\Sigma := \{0, 1\}$ ist $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$ die Kleenesche Hülle.

◊

§9-14 Korollar. (Einfache Eigenschaften der Mengen von Wörtern)

- (1) Es ist immer $\Sigma^0 = \{\varepsilon\}$ unabhängig von Σ .
- (2) $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots$
- (3) $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \dots$
- (4) $\Sigma^* = \Sigma^+ \cup \Sigma^0 = \Sigma^+ \cup \{\varepsilon\}$

◊

9.3 Wortpotenzen

Ein wichtiges grundlegendes Tool sind die sogenannten Wortpotenzen:

§9-15 Definition. (Wortpotenzen)

Sei x ein Wort über einem Alphabet Σ . Für alle $i \in \mathbb{N} \cup \{0\}$ sind Wortpotenzen wie folgt definiert:

$$\begin{aligned} x^0 &= \varepsilon \\ x^1 &= x \\ x^i &= x \cdot x^{i-1}, \quad i > 1. \end{aligned}$$

(Zur Erinnerung: Mit \cdot ist hier die Konkatenation von Wörtern gemeint!) ◊

Mit Wortpotenzen kann man also Wörter kürzer darstellen:

§9-16 Beispiele. (Wortpotenzen)

- (1) $aaaaaaaa = a^9$.
- (2) $aaabb = a^3b^2$.
- (3) $bbababababaaaabab = b^2(ab)^4ba^4bab = b(ba)^4b^2a^3(ab)^2$.

◇

Das letzte Beispiel zeigt allerdings, dass die Schreibweise als Wortpotenz nicht eindeutig ist:
Für ein Wort kann es unter Umständen mehrere verschiedene Darstellungen als Wortpotenz geben!

9.4 Sprachen

Jetzt sind wir soweit, den zentralen Begriffe dieses Kapitels einzuführen:

§9-17 Definition. (Sprache)

Eine Teilmenge $L \subseteq \Sigma^*$ von Wörtern über einem Alphabet Σ wird als Sprache über Σ bezeichnet.

◇

§9-18 Beispiele. (Sprachen)

- (1) Deutsch ist eine Sprache über dem Alphabet der lateinischen Buchstaben, Leerzeichen, Kommata, Punkte, ...
- (2) Programmiersprachen (wie C) sind Sprachen über dem Alphabet ihrer Schlüsselwörter und aller erlaubten Zeichen (z.B. eine Teilmenge des ASCII-Zeichensatzes).

Bemerkung: Es ist gut, sich hier bewusst zu sein, dass damit gewisse *Symbole* des Alphabets einer Programmiersprache im Allgemeinen bestimmte *Wörter der englischen Sprache* sind. Ein paar der Schlüsselwörter (= *Symbole*) beispielsweise des Alphabets $\Sigma_{\text{ANSI-C}}$ sind

{break, case, char, const, do, else, float, if, while, ...}.

Dazu kommen dann noch einzelne Zeichen aus dem ASCII-Zeichensatz, wie

{0, ..., 9, a, ..., z, A, ..., Z, +, -, /, *, %, #, (,), ... usw.}.

- (3) $L := \{\varepsilon, 10, 01, 1100, 1010, 1001, 0110, 0101, 0011, \dots\} \subseteq \{0, 1\}^*$ ist die Sprache der Wörter über $\{0, 1\}$ mit der gleichen Anzahl von Nullen und Einsen.
- (4) $\{10, 100, 110, 1000, 1010, 1100, 1110, \dots\}$ ist die Sprache der positiven geraden Binärzahlen.

◇

Ein paar speziellere Sprachen:

§9-19 Beispiele. (Sprachen)

- (1) Wenn $\Sigma_1 \subseteq \Sigma_2$ gilt und L eine Sprache über Σ_1 ist, dann ist L auch eine Sprache über Σ_2 .

- (2) $L := \Sigma^*$ ist eine Sprache über jedem Alphabet Σ .
- (3) $L := \{\} = \emptyset$ ist die *leere Sprache* für jedes Alphabet.
- (4) $L := \{\varepsilon\}$ ist über jedem Alphabet Σ die Sprache, die nur aus dem leeren Wort ε besteht.

◊

Beachten Sie bei den vorangehenden Beispielen insbesondere, dass $\emptyset \neq \{\varepsilon\}$ gilt. Die leere Sprache und die Sprache nur mit dem leeren Wort sind natürlich nicht dieselben Sprachen! Wichtig ist zudem auch hier: Sprachen können aus unendlich vielen Wörtern bestehen. Diese müssen aber immer aus einem festen, endlichen Alphabet gebildet werden und die Wörter selber haben auch immer eine endliche Länge.

Die vier grundlegendsten Möglichkeiten formale Sprachen zu beschreiben (wir werden aber gleich noch weitere, sehr zentrale kennenlernen^[E.40] !) sind die Folgenden:

§9-20 Definition. (Mögliche Beschreibungen von Sprachen)

Erste Möglichkeiten eine formale Sprache zu beschreiben:

- (1) **Aufzählend** (und damit bei unendlichen Sprachen nicht komplett und unpräzise): Zum Beispiel

$$L := \{\varepsilon, 10, 1100, 111000, \dots\}$$

- (2) **Umgangssprachlich:** Zum Beispiel

L ist die Menge der Wörter über dem Alphabet $\{0, 1\}$, die aus n Einsen gefolgt von n Nullen besteht für eine Zahl $n \in \mathbb{N} \cup \{0\}$.

- (3) **Informal mathematisch bzw. mengentheoretisch:** Zum Beispiel

$$L := \{w \mid w \text{ enthält } n \text{ Einsen gefolgt von } n \text{ Nullen für } n \in \mathbb{N} \cup \{0\}\}$$

- (4) **Mathematisch-formal mit explizitem oder implizitem Prädikat:** Zum Beispiel

$$\begin{aligned} L &:= \{w \in \{0, 1\}^* \mid \underbrace{w = 1^n 0^n}_{=: P(w) \text{ Prädikat}} \wedge n \in \mathbb{N} \cup \{0\}\} \\ &= \underbrace{\{1^n 0^n \mid n \in \mathbb{N} \cup \{0\}\}}_{=: P(w) \text{ Prädikat}} \end{aligned}$$

◊

All diese Notationen definieren dieselbe Sprache. (Bitte überlegen Sie sich das!) In diesem Modul und in der theoretischen Informatik allgemein bevorzugen wir die zwei letzten Notationen.

9.5 Entscheidungsproblem einer Sprache (Wortproblem)

§9-21 Definition. (Entscheidungsproblem einer Sprache (Wortproblem))

Das Entscheidungsproblem (Σ, L) für eine Sprache L über einem Alphabet Σ

(auch Wortproblem für (Σ, L) genannt) ist die folgende Berechnungsaufgabe:

- **Input:** Sprache L über Σ und ein Wort $x \in \Sigma^*$.
- **Output:** JA, falls $x \in L$, und
NEIN, falls $x \notin L$

◊

Die Bedeutung dieser Definition ist folgende: Ihr liegt die Modellierung und Behandlung von vielen alltäglichen Berechnungsproblemen im Formalismus der formalen Sprachen zugrunde und ist deshalb von grossem Interesse. Das Wortproblem kann dabei allerdings von sehr einfach, bis sehr schwierig, bis nicht entscheidbar sein! [E.41]

§9-22 Beispiele. (Primzahltest als Entscheidungsproblem einer Sprache)

Gegeben:

- Alphabet $\Sigma_{\text{Bool}} = \{0, 1\}$.
- Sprache $L_p = \{w \mid w \text{ ist eine Primzahl in Binärdarstellung}\}$.
- Wörter x aus Σ_{Bool}^* .

Der Test, ob x eine Primzahl darstellt, ist äquivalent zur Entscheidung, ob $x \in L_p$ gilt.

Die Entscheidung, ob ein Wort $x \in \Sigma_{\text{Bool}}^*$ eine Primzahl ist oder nicht, ist für einige Wörter einfach: Alle Wörter $x = \dots 0$ stellen zum Beispiel gerade Zahlen dar, können also nicht prim sein (ausser die Zahl 2). Wie wir aber auch wissen, ist für viele andere Wörter (d.h., natürliche Zahlen) diese Entscheidung erheblich aufwändiger.

◊

Kapitel 10

Reguläre Sprachen und endliche Automaten

Kapitelinhalt

10.1 Motivation	227
10.2 Strukturelles Modell / Darstellung endlicher Automaten	228
10.3 Berechnungsmodell eines endlichen Automaten	231
10.4 Sprache eines endlichen Automaten	235

L E R N Z I E L E :

LZ 10.1 Sie verstehen, was reguläre Sprachen und endliche Automaten sind und wie diese beiden Dinge zusammenhängen.

LZ 10.2 Sie verstehen das strukturelle Modell sowie das Berechnungsmodell regulärer Sprachen bzw. endlicher Automaten.

10.1 Motivation

Zwei der fundamentalsten Fragen der Informatik (und auch der Mathematik und sogar vieler wissenschaftlich-technischer Aspekte) sind:

- Was ist ein “Computer” (eine “computing machine”) im weitesten Sinn?
- Was ist “rechnen”, was ist eine “Berechnung” genau?

Wir wollen dies nun mathematisch formalisieren und ein abstraktes Modell dafür erstellen, was genau eine “Berechnung” ist. Wir nennen dies dann ein Berechnungsmodell. Dabei werden wir die sogenannten “endlichen Automaten (EA)” – genauer die “deterministischen endlichen Automaten (DEA)” – kennenlernen. Sie liefern ein einfaches Berechnungsmodell, welches die Modellierung von Berechnungen sehr anschaulich macht. Zudem können wir so weitere fundamentale Konzepte der theoretischen Informatik einführen und etwas vertiefen. Wir werden auch sehen, dass endliche Automaten (EA) allgegenwärtig sind.

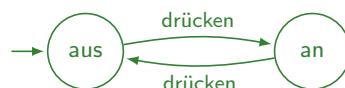
Schauen wir uns zuerst zwei Beispiele an, bevor wir alles exakt und formal definieren:

§10-1 Beispiel. (Lichtschalter)

Ein ganz einfacher Lichtschalter ist bzw. macht im Wesentlichen Folgendes:

- Er kann “an” oder “aus” sein \rightsquigarrow Er hat gewisse “Zustände” (an, aus).
- Er ist zu Beginn immer “aus” \rightsquigarrow Er hat also einen “Startzustand”.
- Man kann den “Zustand” des Lichtschalters durch “drücken” verändern \rightsquigarrow Es gibt “Zustandsübergang” von “an” zu “aus” und umgekehrt.

Schematisch:

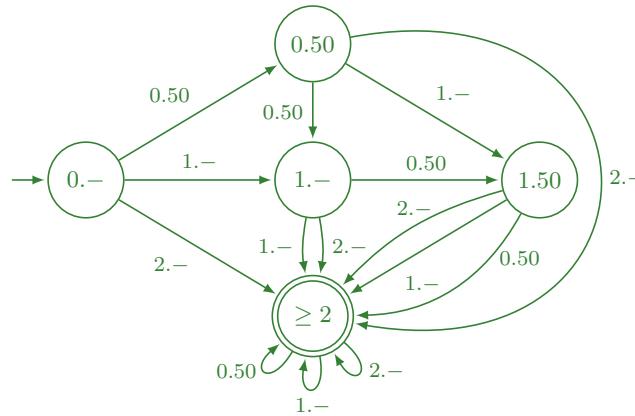


Aufgrund seines aktuellen “Zustands” und einer “Eingabe” (hier “drücken”) “berechnet” er also seinen Nachfolgezustand. ◇

§10-2 Beispiel. (Einstiegsbeispiel: Eintrittskarte Schwimmbad)

Ein etwas interessanteres Beispiel ist ein Münzautomat für die Eintrittskarte bei einem Schwimmbad. Wir nehmen an, der Eintritt betrage 2.-. Der Einfachheit halber nehmen wir an, dass der Automat kein Rückgeld gibt. D.h., die Kosten sind damit *mindestens* 2.- und der Automat gibt die Eintrittskarte aus sobald mind. 2.- eingeworfen wurden. Der Automat nimmt 0.50, 1.- und 2.- Münzen.

Analog zum Beispiel des Lichschalters können wir dies folgendermassen schematisch visualisieren:



Die Zahlen in den Knoten entsprechen der momentan eingeworfenen Summe von Münzen und jede Zahl bei einem Pfeil entspricht einer zusätzlich eingeworfenen (gerade "zu verarbeitenden") Münze. Der doppelt eingekreist Knoten entspricht der Situation, dass mind. für 2.- Münzen eingeworfen wurden und der Knoten mit dem kurzen Pfeil links ist quasi der Startpunkt der ganzen Verarbeitung.

Auch hier "berechnet" der Automat aufgrund seines aktuellen Zustands und der Eingabe seinen Nachfolgezustand. ◇

Wir erkennen bereits an diesen beiden Beispielen zwei wichtige Charakteristika eines daraus abgeleiteten Modells: es ist eine Art "speziellen Maschine", die jeweils ein konkretes Entscheidungsproblem löst und dabei keine Variablen benutzt. Zudem arbeitet das Modell in Echtzeit.

Nun wollen wir uns die Struktur einer solchen "Maschine", welche wir dann eben "endlichen Automaten" nennen werden, einmal näher ansehen:

10.2 Strukturelles Modell / Darstellung endlicher Automaten

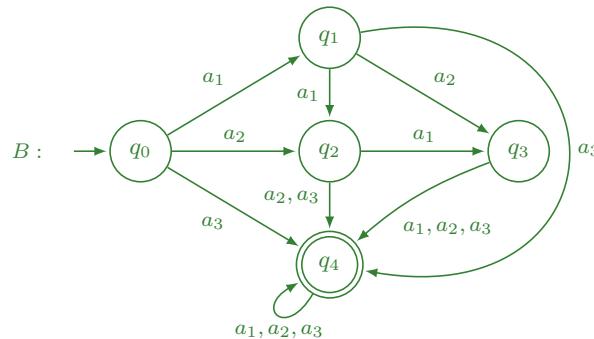
Wenn wir das Vorhergehende analysieren, erkennen wir charakteristische Strukturen für unser Modell:

Charakteristika:	Element im Modell:
Elementare Bausteine:	<ul style="list-style-type: none"> ~ Zustände: ~ Zustandsübergänge: ~ D.h., eine Übergangsfunktion δ mit $q = \delta(p, a)$ für jeden Pfeil (p, q) mit Zeichen a. Zum Beispiel im Beispiel §10-2.
Eingabe:	<ul style="list-style-type: none"> ~ Eingabealphabet Σ von zu verarbeitenden Symbolen. ($\Sigma = \{0.50, 1.-, 2.-\}$ im Beispiel §10-2.) ~ Der Automat liest vom Startzustand beginnend, ein Wort (Eingabewort) w über dem Eingabealphabet Σ von links nach rechts und verarbeitet es zeichenweise.
Speicher:	<ul style="list-style-type: none"> ~ Das Modell benutzt <i>keinen</i> Speicher (ausser dem Automaten selbst, d.h. ausser den möglichen Zuständen und der Übergangsfunktion). ~ Variablen dürfen <i>nicht</i> benutzt werden. ~ Die einzige Information ist der aktuelle Zustand.
Berechnung:	<ul style="list-style-type: none"> ~ Folge von Zustände (auf einem Eingabewort; siehe nächster Abschnitt!)
Ausgabe:	<ul style="list-style-type: none"> ~ Akzeptierende Zustände

Tabelle 10.2 – Strukturelle Charakteristika eines endlichen Automaten.

§10-3 Beispiel. (Einstiegsbeispiel: Zustände und Zustandsübergänge)

Im Schwimmbadautomaten unseres Einstiegsbeispiels sind die Elemente also die Folgenden:

Zustände: $q_0 = \text{Start}$, $q_1 = 0.50$, $q_2 = 1.-$, $q_3 = 1.50$, $q_4 = \geq 2$ Eingabealphabet: $a_1 = 0.50$, $a_2 = 1.-$, $a_3 = 2.-$

◊

Aufgrund der in vorgehender Tabelle gemachten Abstrahierung [E.42] definieren wir unser Modell folgendermassen:

§10-4 Definition. ((Deterministischer) Endlicher Automat)

Ein (deterministischer) endlicher Automat (EA, DEA) ist ein Quintupel

$$M = (Q, \Sigma, \delta, q_0, F)$$

mit

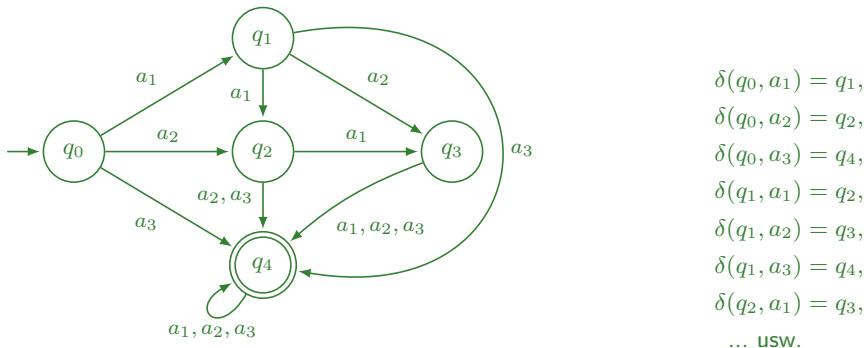
- (1) endliche Menge von Zuständen $Q = \{q_0, q_1, \dots, q_n\}$ ($n \in \mathbb{N} \cup \{0\}$)
- (2) Eingabealphabet $\Sigma = \{a_1, a_2, \dots, a_m\}$ ($m \in \mathbb{N} \setminus \{0\}$)
- (3) Übergangsfunktion [E.43] $\delta : Q \times \Sigma \rightarrow Q$
- (4) Startzustand $q_0 \in Q$
- (5) Menge der akzeptierenden Zustände $F \subseteq Q$

◊

Wieso der erwähnte Zusatz "deterministisch"? Der Grund ist, dass es auch nicht-deterministische endliche Automaten gibt: Bei diesen ist die Übergangsfunktion auch vom Zufall abhängig, d.h., nur mit einer gewissen Wahrscheinlichkeit wird einem q_i beim Verarbeiten des Zeichens a ein bestimmter Folgezustand $q_j := \delta(q_i, a)$ zugeordnet. Mit einer gewissen Wahrscheinlichkeit wird aber eine andere Zuordnung z.B. $q_k := \delta(q_i, a)$, wobei $q_k \neq q_j$ ist, vorgenommen, ... usw.. Wir betrachten in diesem Modul hier aber nur deterministische EA wie oben definiert.

§10-5 Beispiel. (Einstiegsaufgabe: Übergangsfunktion)

$\delta(q_i, a) = q_j$ bedeutet im folgenden: Der EA wechselt zu q_j , falls in q_i das Symbol a gelesen wird. Beispiele für die Übergangsfunktion in unserem Schwimmbadautomaten sind:



◊

Das gesamte Modell für unser Beispiel können wir also formal insgesamt folgendermassen darstellen:

§10-6 Beispiel. (Einstiegsaufgabe: Gesamte formale Darstellung)

Der Münzautomat für die Eintrittskarte zum Schwimmbad in §10-2 ist ein DEA

$$M = (Q, \Sigma, \delta, q_0, F)$$

mit

- Zustandsmenge $Q = \{q_0, q_1, q_2, q_3, q_4\}$
- Eingabealphabet $\Sigma = \{a_1, a_2, a_3\}$
- Startzustand q_0

- akzeptierende Zustände $F = \{q_4\}$
- Übergangsfunktion δ :

Zustand	Eingabe		
	a_1	a_2	a_3
q_0	q_1	q_2	q_4
q_1	q_2	q_3	q_4
q_2	q_3	q_4	q_4
q_3	q_4	q_4	q_4
q_4	q_4	q_4	q_4

◊

10.3 Das Berechnungsmodell eines endlichen Automaten

Damit ist die Struktur unseres Modells nun klar. Wir wollen jetzt noch ein Modell erhalten, was auch die Frage der "Berechnung" klärt.

Um allgemein ein Berechnungsmodell zu erstellen, muss man folgende Fragen beantworten (Zitat [4, p.39]):

- (1) Welche elementaren Operationen stehen zur Verfügung, aus denen man die "Programme" (Operationsabfolgen) zusammenstellen kann?
- (2) Wieviel Speicher steht zur Verfügung und wie geht man mit dem Speicher um?
- (3) Wie wird die Eingabe eingegeben?
- (4) Wie wird die Ausgabe bestimmt und ausgegeben?

Dies führt zu folgendem Schema, mit welchem wir allgemein Berechnungsmodelle bauen können:

§10-7 Definition. (Schema für den Aufbau von Berechnungsmodellen)

- (1) Definiere die Struktur, welche die exakte Beschreibung jedes Objekts der Modellklasse (z.B. EA) ermöglicht.
- (2) Beschreibung der Bedeutung (Semantik) der Struktur:
 - (a) Konfiguration: Vollständige Beschreibung einer Situation, in der sich das Modell befindet.
 - (b) Berechnungsschritt: Übergang aus einer Konfiguration in eine andere durch die Ausführung eines Befehls des Modells.
- (3) Berechnung: Folge solcher elementaren Berechnungsschritte
- (4) Nun kann man jeder Eingabe das Arbeitsresultat als Ausgabe zuordnen.

◊

In unserem Fall haben wir den Schritt (1) mit unserem strukturellen Modell eines EA im vorhergehenden Abschnitt erledigt. Die Schritte (3) und (4) und teilweise auch (2)(b) haben wir dort auch schon angedeutet. Jetzt wollen wir diese aber auch noch genau formalisieren, so dass wir vom strukturellen Modell (einer strukturellen Beschreibung) eines EA zu einem exakten Berechnungsmodell eines EA kommen.

Es zeigt sich, dass die entscheidenden Begriffe hierzu diejenigen der Konfiguration und des (elementaren) Berechnungsschritt eines EA's sind. Aus der exakten Definition eines Berechnungsschrittes können wir dann sehr einfach den Begriff Berechnung in einem EA ableiten, was unser Berechnungsmodell dann komplettiert.

Wir starten mit dem ersten Begriff einer "Konfiguration": Diese liefert die vollständige Beschreibung der Situation, in der sich der Automat befindet und muss alle Informationen enthalten, die noch Einfluss auf spätere Berechnungen haben könnten.

§10-8 Definition. (Konfiguration)

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein EA. Eine Konfiguration von M ist ein Element aus $Q \times \Sigma^*$. \diamond

Wenn sich also ein EA M in einer Konfiguration $(q, w) \in Q \times \Sigma^*$ befindet, bedeutet dies, dass M im Zustand q ist und noch das Subwort w des Eingabewortes lesen (bzw. verarbeiten) soll. Zwei solche Konfigurationen sind von besonderer Bedeutung, denn die Berechnung von M auf einem Eingabewort w muss in einer eindeutig festgelegten Konfiguration beginnen und in einer Konfiguration enden, bei der alle Symbole von w schon gelesen worden sind:

§10-9 Definition. (Startkonfiguration)

Eine Konfiguration $(q_0, w) \in \{q_0\} \times \Sigma^*$ nennen wir eine Startkonfiguration von M auf w . \diamond

§10-10 Definition. (Endkonfiguration)

Jede Konfiguration aus $Q \times \{\epsilon\}$ nennt man eine Endkonfiguration. \diamond

Wie sieht ein Schritt von einer Situation (einer Konfiguration) zur nächsten aus? Dies führt uns auf den zweiten zentralen Begriff, welchen wir zuerst an einem Beispiel demonstrieren und dann formal definieren:

§10-11 Beispiel. (Einstiegsaufgabe: Berechnungsschritt)

Den elementarsten Schritt bzw. die elementarste "Veränderung" in unserem Beispiel §10-2 können wir als das Folgende identifizieren:

Konfiguration vor dem Berechnungsschritt: $(q_2, a_3a_2a_1a_2)$

- Wir sind z.B. im Zustand q_2 .
- Das Wort $a_3a_2a_1a_2$ ist noch zu lesen (zu verarbeiten).



Konfiguration nach dem Berechnungsschritt: $(q_4, a_2a_1a_2)$

- Zustand q_4
- $a_2a_1a_2$ noch zu lesen

Mit anderen Worten: Das elementarste, was geschehen kann, ist ein "Sprung" entlang eines solchen Pfeiles. \diamond

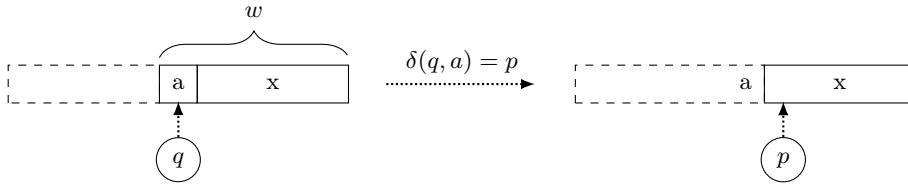
Das heisst, wir können einen Berechnungsschritt folgendermassen exakt und formal definieren:

§10-12 Definition. (Berechnungsschritt)

Ein Berechnungsschritt \vdash_M von M ist die Anwendung der Übergangsfunktion auf die aktuelle Konfiguration und liefert eine Nachfolge-Konfiguration. Formal ist also

$$(q, w) \vdash_M (p, x)$$

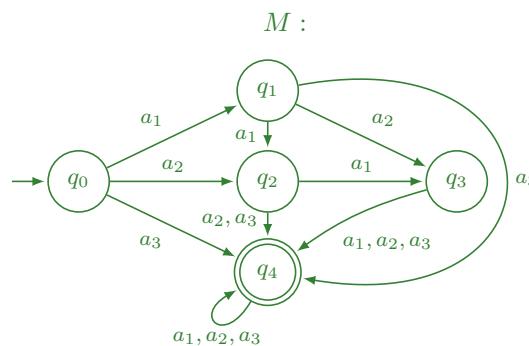
genau dann ein Berechnungsschritt, wenn $w = ax$, $a \in \Sigma$, $x \in \Sigma^*$, $q \in Q$ ist und $\delta(q, a) = p$ mit $p \in Q$, gilt. \diamond



Bei der Schreibweise für einen Berechnungsschritt hat sich folgende Konvention eingebürgert: Wenn aus dem Kontext klar ist, um welchen Automaten M es sich handelt, dann schreiben wir auch nur \vdash und lassen den Subskript M weg.

Nun können wir unsere letzte Frage beantworten. Wie sieht damit eine "Berechnung" aus? Dies ist natürlich eine Folge von Schritten folgender Art, wieder zuerst am Beispiel:

§10-13 Beispiel. (Einstiegsaufgabe: Berechnung)



Berechnung:

$$(q_2, a_3a_2a_1a_2) \vdash_M (q_4, a_2a_1a_2) \vdash_M (q_4, a_1a_2) \vdash_M (q_4, a_2).$$

\diamond

§10-14 Definition. (Berechnung)

Eine endliche Folge von Berechnungsschritten nennt man eine **Berechnung**. Eine Berechnung

$$(q_{i_1}, a_1 a_2 \dots a_n) \vdash_M (q_{i_2}, a_2 \dots a_n) \vdash_M \dots \vdash_M (q_{i_j}, a_j \dots a_n)$$

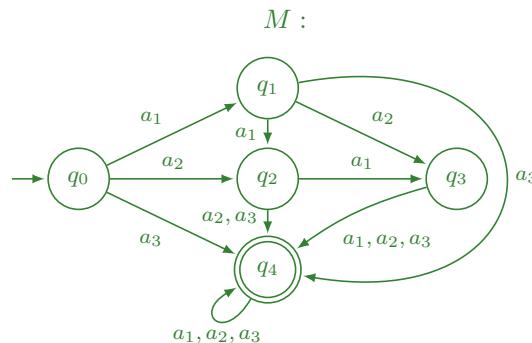
wird abgekürzt als $(q_{i_1}, a_1 a_2 \dots a_n) \vdash_M^* (q_{i_j}, a_j \dots a_n)$ geschrieben. \diamond

§10-15 Definition. (Berechnung von M auf w ; akzeptierende und verwerfende Berechnung)

Die **Berechnung von M auf einer Eingabe $w \in \Sigma^*$** ist eine Berechnung, die in der Startkonfiguration (q_0, w) startet und in einer Endkonfiguration (q_e, ε) für irgend ein $q_e \in Q$ endet.

Diese Berechnung ist **akzeptierend**, wenn $q_e \in F$ gilt, und **verwerfend**, falls $q_e \notin F$. \diamond

§10-16 Beispiel. (Einstiegsaufgabe: akzeptierende und verwerfende Berechnungen)



Dann sind beispielsweise die Berechnungen

- $(q_0, a_2 a_1 a_2) \vdash_M (q_2, a_1 a_2) \vdash_M (q_3, a_2) \vdash_M (q_4, \varepsilon)$ **akzeptierend**,
- $(q_0, a_1 a_2) \vdash_M (q_1, a_2) \vdash_M (q_3, \varepsilon)$ **verwerfend**.

\diamond

§10-17 Definition. (Zusammenfassung: Das Berechnungsmodell eines EA)

Das **Berechnungsmodell eines endlichen Automaten (EA)** M (welcher selber definiert ist gemäß §10-4), ist definiert durch den Begriff der **Berechnung von M auf einem Eingabewort w** gemäß §10-15.

Jede konkrete Instanz in diesem Berechnungsmodell wird **Berechnung** genannt.

\diamond

10.4 Sprache eines endlichen Automaten

§10-18 Definition. (Sprache eines endlichen Automaten)

Die Sprache $L(M)$ eines endlichen Automaten M ist die Menge aller von M akzeptierten Wörter:

$$L(M) = \{w \in \Sigma^* \mid \text{Berechnung von } M \text{ auf } w \text{ endet in einem akzeptierenden Zustand}\}$$

$L(M)$ nennen wir dann auch die von M akzeptierte Sprache. ◊

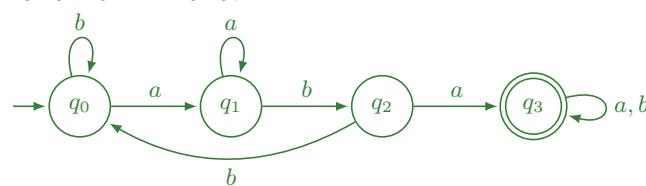
§10-19 Definition. (Klasse der regulären Sprachen (Typ-3 Sprachen))

Die Klasse der regulären Sprachen (oder Typ-3 Sprachen) beinhaltet alle Sprachen, die von einem endlichen Automaten akzeptiert werden.

Jede dieser Sprachen wird regulär bzw. vom Typ-3 genannt. ◊

§10-20 Beispiel.

$$M_1 = (\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_3\})$$



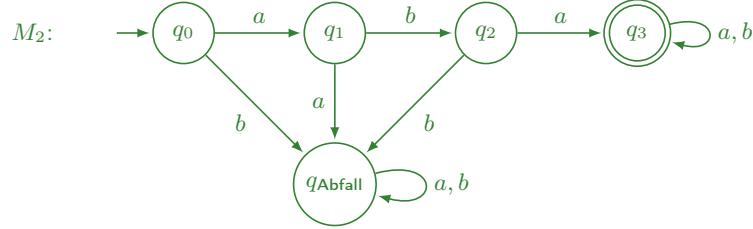
Von M_1 akzeptierte Sprache:

$$\begin{aligned} L(M_1) &= \{xabay \mid x, y \in \{a, b\}^*\} \\ &= \{w \in \{a, b\}^* \mid w \text{ enthält das Teilwort } aba\} \end{aligned}$$

◊

§10-21 Beispiel.

$$\begin{aligned} L(M_2) &= \{abay \mid y \in \{a, b\}^*\} \\ &= \{w \in \{a, b\}^* \mid w \text{ hat als Präfix } aba\} \end{aligned}$$



◇

Das letzte Beispiel zeigt, dass zur Vereinfachung folgende Konvention sinnvoll ist:

Konvention:

Einen Abfall-Zustand – also einen Zustand von welchem aus der EA nicht mehr in einen akzeptierenden Zustand gelangen kann – darf man weglassen und deshalb gilt: Wenn in einem deterministischen endlichen Automaten Übergänge fehlen, dann führen diese in einen Zustand, von dem kein Übergang mehr wegführt ("Abfall"-Zustand).

§10-22 Beispiel.

Wir können den EA der die Sprache M_2 aus Beispiel §10-21 akzeptiert also auch darstellen als:



◇

Zum Schluss noch eine sehr häufig verwendete Schreibweise für eine wichtige Eigenschaft, deren Gebrauch wir im anschliessenden Beispiel gleich demonstrieren werden:

§10-23 Definition. (Anzahl Symbole in einem Wort)

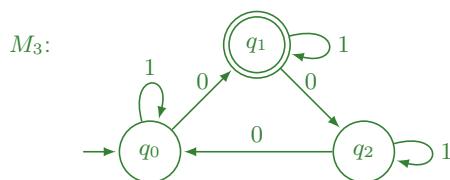
$|w|_a$ ist die Anzahl der Symbole a im Wort w .

◇

§10-24 Beispiel.

Bezeichne $x \bmod n$ den Rest der ganzzahligen Teilung von x durch n (siehe Modul "mada").

Sei $L(M_3) = \{w \in \{0,1\}^* \mid |w|_0 \bmod 3 = 1\}$, d.h., $L(M_3) = \{w \in \{0,1\}^* \mid |w|_0 \in \{1, 4, 7, 10, 13, \dots\}\}$. Dann ist der EA gegeben als



Die akzeptierte Sprache, ist also die Menge aller Binärwörter, in denen die Anzahl Nullen durch 3 geteilt einen Rest von 1 ergeben. D.h., die Anzahl Nullen sind $\{1, 4, 7, 10, 13, \dots\}$. ◇

Kapitel 11

Formale Grammatiken

Kapitelinhalt

11.1 Gibt es nicht-reguläre Sprachen?	239
11.2 Kontextfreie Grammatiken	239
11.3 Mehr- und Eindeutigkeit bei kontextfreien Grammatiken	244
11.4 Sprach-Hierarchie zwischen regulären und kontextfreien Sprachen	245
11.5 Kontextfreie Grammatik für eine reguläre Sprache	246
11.6 Techniken für den Entwurf von kontextfreien Grammatiken	247
11.7 Zentrale Anwendungen in der Informatik	247

L E R N Z I E L E :

- LZ 11.1** Sie verstehen die Motivierung für formale Grammatiken.
 - LZ 11.2** Sie wissen, wie eine formale Grammatik definiert wird.
 - LZ 11.3** Sie können einfache kontextfreie formale Grammatiken der vorgestellten Art entwerfen.
 - LZ 11.4** Sie kennen die vorgestellten Anwendungen von kontextfreien formalen Grammatiken.
 - LZ 11.5** Sie verstehen das Problem der Mehr- und Eindeutigkeit von kontextfreien Grammatiken und wissen, was inhärent mehrdeutige Sprachen sind.
-

11.1 Gibt es nicht-reguläre Sprachen?

Wir haben gesehen, dass sich gewisse Sprachen mittels eines (deterministischen) endlichen Automaten (DEA, EA) beschreiben bzw. definieren lassen. Diese Sprachen haben wir reguläre oder Typ-3 Sprachen genannt. Gibt es auch nicht-reguläre Sprachen? D.h., gibt es Sprachen, welche sich nicht mit einem DEA beschreiben lassen?

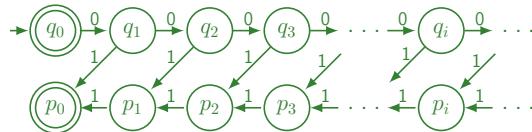
Bereits ein einfaches Beispiel zeigt uns, dass dies der Fall ist:

§11-1 Beispiel. (Einfache nicht-reguläre Sprache)

Wir betrachten die Sprache über dem Booleschen Alphabet $\{0, 1\}$, welche aus allen Wörtern der Form "n Nullen gefolgt von n Einsen" für irgendeine natürliche Zahl n besteht. Formal:

$$L := \{0^n 1^n \mid n \in \mathbb{N} \cup \{0\}\}.$$

Wie muss ein akzeptierender Automat für diese Sprache aussehen? Wenn wir uns dies genauer überlegen, dann ist es offensichtlich so:



Wir sehen: Der Automat muss also *unendlich* viele Zustände haben! Damit ist ein solcher akzeptierender Automat sicher *kein* deterministischer endlicher Automat und die Sprache L dann auch *nicht*-regulär. \diamond

Auch wenn es keinen endlichen Automaten gibt, welcher obige Sprache akzeptiert, lässt sich diese Sprache bzw. Vorschrift zur Erzeugung leicht beschreiben. Der entscheidende Grundbegriff dazu ist der einer "kontextfreien Grammatik" und dies ist Thema des nun folgenden Abschnitts 11.2.

11.2 Kontextfreie Grammatiken

§11-2 Beispiel. (Einführendes Beispiel einer kontextfreien, nicht-regulären Sprache und Grammatik)

Wir betrachten wieder die Sprache des Beispiels §11-1 von vorhin, welche wie gesehen also nicht regulär sein kann:

$$L := \{0^n 1^n \mid n \in \mathbb{N} \cup \{0\}\}.$$

Damit ist es also nicht möglich einen endlichen Automaten zur Beschreibung dieser Sprache anzugeben. Gibt es aber einen anderen Weg?

Ja! Denn wir können folgende zwei Regeln formulieren:

- 1. Regel: $S \rightarrow 0S1$
- 2. Regel: $S \rightarrow \varepsilon$

wobei S ein Symbol ist, mit welchem wir starten (linke Seite einer der beiden Regeln) und welches wir jeweils durch den Ausdruck auf der rechten Seite des Pfeils \rightarrow ersetzen. Dies tun wir solange durch Anwendung einer der beiden Regeln, bis wir rechts kein S mehr im Ausdruck haben. Auf diese Weise können wir *genau* jedes Wort der Sprache L durch eine endliche Sequenz dieser Regeln produzieren. Beispielsweise das Wort 000111:

$$S \xrightarrow{1.\text{Regel}} 0S1 \xrightarrow{1.\text{Regel}} 00S11 \xrightarrow{1.\text{Regel}} 000S111 \xrightarrow{2.\text{Regel}} 000111$$

\diamond

Die Idee im Beispiel zeigt: Alle Wörter, die wir in dieser "Grammatik" ableiten können, bilden dann die "Sprache der Grammatik". Ganz allgemein kann man so "formale Grammatiken" definieren. Wir wollen uns aber in diesem Modul auf höchstens die sogenannten "kontextfreien Grammatiken" konzentrieren, welche folgendermassen exakt definiert sind:

§11-3 Definition. (Kontextfreie Grammatik (KFG; Typ-2 Grammatik))

Eine kontextfreie Grammatik G (KFG; auch Typ-2 Grammatik) ist ein 4-Tupel (N, Σ, P, S) mit

- N ist das Alphabet der Nichtterminale (Variablen).
- Σ ist das Alphabet der Terminalen.
- P ist eine endliche Menge von Produktionen (Regeln). Jede Produktion hat die Form

$$X \rightarrow \beta$$

mit Kopf $X \in N$ und Rumpf $\beta \in (N \cup \Sigma)^*$.

- S ist das Startsymbol, wobei $S \in N$.

◇

Was meinen wir mit diesem Zusatz "kontextfrei" genauer? Es geht darum, dass wir in dieser Definition (erstmals) noch eine wichtige Einschränkung an die erlaubte Form der Produktionen machen: Auf der linken Seite (d.h., im Kopf der Produktion) darf immer nur genau ein Nichtterminal $X \in N$ stehen. Insbesondere hängt also eine Regel nicht von einem "Kontext" um das zu ersetzende Nichtterminal X herum ab, wie wenn wir zum Beispiel einen Kopf aXb für $X \in N$ und $a, b \in \Sigma$ erlauben würden. [E.44]

Als weiterer Punkt ist zu beachten, dass eine Ableitung mittels Produktionen beginnend mit dem Startsymbol nicht zu terminieren braucht. Es ist also möglich, dass wir eine Satzform erreichen, welche noch Nichtterminale enthält, aber keine passenden Produktionen zur Fortsetzung der Ableitung zur Verfügung stehen. Das ist nicht "schlimm", wird aber in diesem Fall zu keiner Erzeugung eines Wortes der Sprache führen.

§11-4 Definition. (Satzform)

Ein Wort $\beta \in (N \cup \Sigma)^*$ nennen wir eine **Satzform**.

◇

§11-5 Beispiel. (Kontextfreie Grammatik)

Eine KFG G_1 für die Sprache $\{0^n 1^n \mid n \in \mathbb{N} \cup \{0\}\}$:

$$G_1 = (\{S\}, \{0, 1\}, P, S)$$

mit

$$P = \{S \rightarrow 0S1, S \rightarrow \varepsilon\}$$

◇

Zur Vereinfachung der Notation vereinbaren wir folgende Schreibweise: Mehrere Regeln mit dem gleichen Nichtterminal im Kopf notieren wir kompakt in einer Zeile und mit einem vertikalen Strich | getrennt. Also beispielsweise

$$S \rightarrow 0S1 \mid \varepsilon$$

für die beiden Regeln $S \rightarrow 0S1$ und $S \rightarrow \varepsilon$ im vorhergehenden Beispiel.

§11-6 Beispiel. (Kontextfreie Grammatik G für die Sprache aller Palindrome^[E.45] über $\{a, b\}$)

$$G = (\{S\}, \{a, b\}, P, S)$$

mit den folgenden Produktionen:

$$\begin{aligned} S &\rightarrow aSa \\ S &\rightarrow bSb \\ S &\rightarrow a \\ S &\rightarrow b \\ S &\rightarrow \varepsilon \end{aligned}$$

◊

§11-7 Beispiel. (Grammatik für balancierte Klammerausdrücke)

Eine KFG G_2 für die Sprache der balancierten Klammerausdrücke wie zum Beispiel $((())()$:

$$G_2 = (\{S\}, \{((),)\}, P, S)$$

Die Menge der Produktionen P ist

$$S \rightarrow (S) \mid SS \mid \varepsilon$$

◊

§11-8 Beispiel. (Ableitung des Wortes $((())()$ in G_2)

$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow (S)(S) \Rightarrow ((S))(S) \Rightarrow ((())(S) \Rightarrow ((())()$$

◊

§11-9 Definition. (Ableitungsschritt, Ableitung eines Wortes $w \in \Sigma^*$)

Seien α, β und γ Satzformen und $A \rightarrow \gamma$ eine Produktion.

- Durch einen **Ableitungsschritt** wird eine Satzform $\alpha A \beta$ durch die Anwendung der Produktion $A \rightarrow \gamma$ in die Satzform $\alpha \gamma \beta$ abgeleitet. Das notieren wir mit

$$\alpha A \beta \Rightarrow \alpha \gamma \beta$$

- Eine **Ableitung eines Wortes** $w \in \Sigma^*$ ist eine Folge von Ableitungsschritten, so dass aus einer Satzform α das Wort w abgeleitet wird:

$$\alpha \Rightarrow \dots \Rightarrow w$$



§11-10 Definition. (Linksseitige- und rechtsseitige Ableitung)

- Eine **linksseitige Ableitung** ersetzt bei jedem Ableitungsschritt das Nicht-terminal, welches am weitesten links in der Satzform auftritt.
- Eine **rechtsseitige Ableitung** ersetzt bei jedem Ableitungsschritt das Nicht-terminal, welches am weitesten rechts in der Satzform auftritt.



Jedes Wort einer kontextfreien Sprache hat mindestens eine links- und eine rechtsseitige Ableitung. Enthalten alle Produktionen nur Satzformen mit einem Nichtterminal, dann sind linksseitige und rechtsseitige Ableitungen *o.ensichtlich* immer identisch in dem Sinn, dass die Folge der angewendeten Regeln exakt die gleiche ist. Sonst muss dies aber nicht notwendigerweise der Fall sein (siehe Beispiele).

§11-11 Beispiel. (Links- und rechtsseitige Ableitung in G_2)

Linksseitige Ableitung von $(())()$: $S \Rightarrow SS \Rightarrow (S)S \Rightarrow ((S))S \Rightarrow (())S \Rightarrow (())(S) \Rightarrow (())()$.
 Rechtseitige Ableitung von $(())()$: $S \Rightarrow SS \Rightarrow S(S) \Rightarrow S() \Rightarrow (S)() \Rightarrow ((S))() \Rightarrow (())()$. ◇

§11-12 Definition. (Ableitbarkeit)

Ein Wort $w \in \Sigma^*$ ist in einer kontextfreien Grammatik $G = (N, \Sigma, P, S)$ **ableitbar**, falls es eine Ableitung in G gibt, die mit dem Startsymbol S beginnt und mit dem Wort w endet. Dafür schreiben wir

$$S \Rightarrow^* w$$

Wir sagen auch, dass w von S **erzeugt** oder **generiert** wird. ◇

§11-13 Beispiel. (Ableitbar in G_2)

Das Wort $(())()$ ist in G_2 ableitbar. ◇

Es ist üblich, wenn man sich auf eine (rein) linksseitige oder rechtsseitige Ableitung bezieht, dies symbolisch zu notieren. Z.B. durch $\xrightarrow{\text{ls}}^*$ oder $\xrightarrow{\text{rs}}^*$ o.ä... Die Mehrdeutigkeit von Grammatiken kann problematisch sein, wenn man sie nicht beachtet. Ein typisches Beispiel:

[Das Dangling Else Problem \(\[https://en.wikipedia.org/wiki/Dangling_else\]\(https://en.wikipedia.org/wiki/Dangling_else\)\)](https://en.wikipedia.org/wiki/Dangling_else)

Das sollten Sie kurz studieren!

§11-14 Definition. (Sprache einer kontextfreien Grammatik)

Die von der Grammatik G erzeugte Sprache $L(G)$ beinhaltet alle Wörter, die in G aus dem Startsymbol S ableitbar sind:

$$L(G) = \{ w \in \Sigma^* \mid S \Rightarrow^* w \}$$

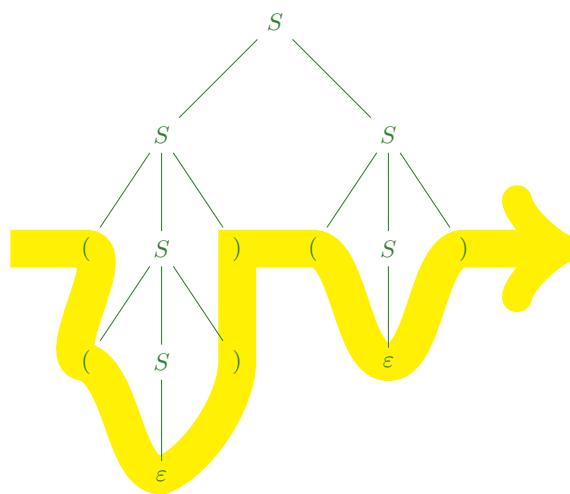
◊

§11-15 Definition. (Kontextfreie Sprache (Typ-2 Sprache))

Wenn es für eine Sprache L eine kontextfreie Grammatik G gibt mit $L = L(G)$, dann nennen wir L eine kontextfreie Sprache oder auch eine **Typ-2 Sprache**.

◊

Wenn wir eine Ableitung eines Wortes schrittweise von oben nach unten als Baum aufbauen, so haben wir eine graphische Darstellung dieser Ableitung. Wir nennen diese den **Ableitungsbau** des Wortes für die gegebene Grammatik [E.46].

§11-16 Beispiel. (Ableitungsbau für das Wort $((())()$ in G_2)

◊

Ableitungsbäume werden typischerweise von Compilern als Datenstruktur für die interne Repräsentation von Quellprogrammen erzeugt. Sie werden deshalb auch als Parsebäume bezeichnet. Sie verdeutlichen also grafisch, wie Symbole der terminalen Wörter bzw. Satzformen einer KFG in Teilwörtern strukturiert werden. Wir betrachten ein weiteres Beispiel, welches diese mehrdeutigen Möglichkeiten illustriert:

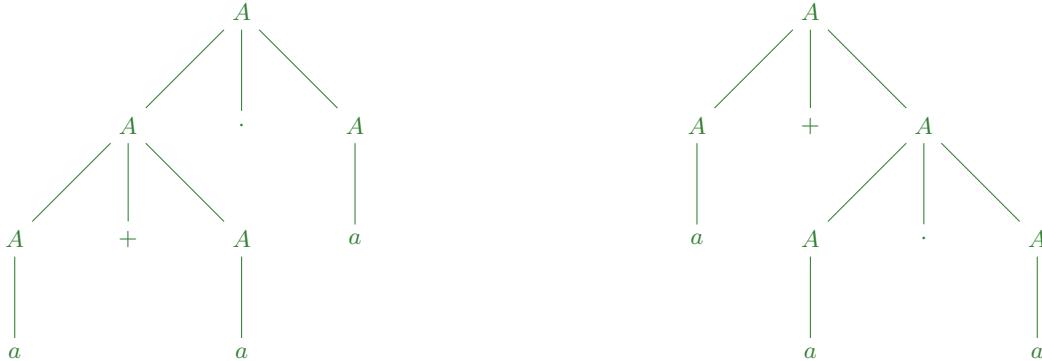
§11-17 Beispiel. (Kontextfreie Grammatik für arithmetische Ausdrücke)

$$G_3 = (\{A\}, \{a, +, \cdot, (,)\}, P, A)$$

mit der Menge der Produktionen

$$A \rightarrow A + A, \quad A \rightarrow A \cdot A, \quad A \rightarrow (A), \quad A \rightarrow a.$$

Das Wort $a + a \cdot a$ kann durch zwei verschiedene Ableitungsbäume dargestellt werden:



◊

11.3 Mehr- und Eindeutigkeit bei kontextfreien Grammatiken

Ein Wort welches bzgl. einer kontextfreien Grammatik mehrere verschiedene Ableitungsbäume besitzt, hat auch mehrere linksseitige (rechtsseitige) Ableitungen.

§11-18 Beispiel.

Zwei linksseitige Ableitungen für das Wort $w := a + a \cdot a$ der Grammatik G_3 aus Beispiel §11-17:

- $A \Rightarrow A \cdot A \Rightarrow A + a \cdot a$.
- $A \Rightarrow A + A \Rightarrow a + A \Rightarrow a + A \cdot A \Rightarrow a + a \cdot A \Rightarrow a + a \cdot a$.

◊

Dies führt auf eine wichtige Eigenschaft von Grammatiken und wir definieren deshalb:

§11-19 Definition. (Mehrdeutige und eindeutige kontextfreie Grammatik)

Eine kontextfreie Grammatik nennen wir **mehrdeutig**, wenn es in ihr ein Wort gibt, welches mehrere Ableitungsbäume besitzt. Ansonsten heisst sie **eindeutige Grammatik**.

◊

Wie können wir die Mehrdeutigkeit von KFG eliminieren? Zum Beispiel:

- Korrekte Klammerung vom Benutzer erzwingen:
z. B. $a + (a \cdot a)$ statt $a + a \cdot a$
- Den Produktionen einen Vorrang vergeben:
z. B. $A \rightarrow A + A$ den Vorrang über $A \rightarrow A \cdot A$ geben. (Beachten Sie: umgekehrt, damit die arithmetische "Punkt- vor Strichoperation"-Regel gilt!)

Für Compiler/Parser sind beispielsweise eindeutige Grammatiken wichtig, da die Darstellung und die Verarbeitung eindeutig und vorhersehbar sein soll! Es gibt aber Sprachen, für die dies prinzipiell nicht möglich ist:

§11-20 Definition. (Inhärent mehrdeutige Sprachen)

Eine kontextfreie Sprache, für die alle Grammatiken mehrdeutig sind, heisst **inhärent mehrdeutig**.

Ansonsten nennen wir die Sprache **(inhärent) eindeutig**. ◇

Beachten Sie: Eine bestimmte Sprache kann also sowohl mehrdeutige Grammatiken, als auch nicht-mehrdeutige (d.h. eindeutige) Grammatiken besitzen! Und wir sprechen nur von einer inhärent mehrdeutigen Sprache, wenn alle Grammatiken mehrdeutig sind. Umgekehrt fassen wir eine Sprache als eindeutig auf, wenn es nur schon eine einzige Grammatik gibt, die eindeutig ist.

§11-21 Beispiel. (Inhärent mehrdeutige kontextfreie Sprache)

$$\{ a^i b^j c^k \mid i = j \text{ oder } j = k \}$$

für $i, j, k \in \mathbb{N} \cup \{0\}$ ist inhärent mehrdeutig.

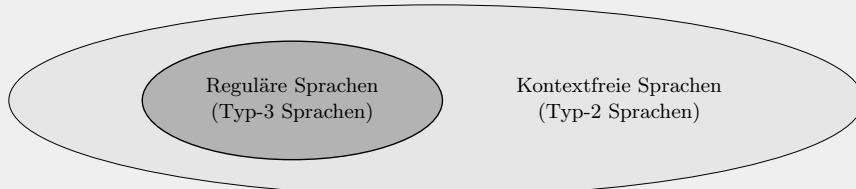
Dass dies eine inhärent mehrdeutige kontextfreie Sprache ist, lässt sich dadurch zeigen, dass sie als Vereinigung zweier kontextfreier Sprachen geschrieben wird, welche beide inhärent mehrdeutig sind und für die der Nachweis der inhärenten Mehrdeutigkeit einfacher ist. Ein Beweis hierzu findet sich zum Beispiel bei [11]. ◇

11.4 Sprach-Hierarchie zwischen regulären und kontextfreien Sprachen

Wie hängen die kontextfreien Sprachen mit den früher kennengelernten regulären Sprachen zusammen? Es gilt folgende grundlegende Aussage:

§11-22 Theorem.

Die kontextfreien Sprachen enthalten die regulären Sprachen.



◇

Beweis. Beweisidee: Dies lässt sich beweisen, indem man zeigt, dass jede reguläre Sprache durch eine kontextfreie Grammatik beschrieben werden kann (siehe nächster Abschnitt 11.5). Dass die Umkehrung nicht gilt, haben wir schon an einem einfachen Beispiel in §11-1 gesehen. \square

11.5 Kontextfreie Grammatik für eine reguläre Sprache

Da reguläre Sprachen also insbesondere kontextfrei sind und damit eine kontextfreie Grammatik haben, stellt sich die Frage, wie wir bei gegebenem deterministischen endlichen Automaten für eine reguläre Sprache zu einer solchen kontextfreien Grammatik kommen. Dafür gibt es einen einfachen Algorithmus:

§11-23 Theorem. (Von einem DEA zu einer zugehörigen KFG)

Sei L eine reguläre Sprache über einem Alphabet Σ . Dann gibt es einen DEA $M = (Q, \Sigma, \delta, q_0, F)$ mit $L(M) = L$.

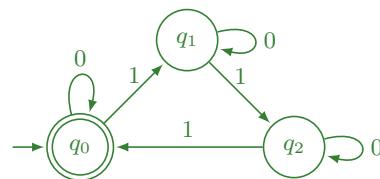
Eine KFG für L können wir wie folgt bauen:

1. Für jeden Zustand q_i gibt es ein Nichtterminal Q_i .
2. Für jede Transition $\delta(q_i, a) = q_j$ erstellen wir die Produktion $Q_i \rightarrow aQ_j$.
3. Für jeden akzeptierenden Zustand $q_i \in F$ erstellen wir die Produktion $Q_i \rightarrow \varepsilon$.
4. Das Nichtterminal Q_0 wird zum Startsymbol.

\diamond

§11-24 Beispiel.

Sei $L = \{w \in \{0, 1\}^* \mid |w|_1 \bmod 3 = 0\}$ die Sprache der binären Wörter in denen die Anzahl der Einsen durch 3 teilbar ist.



Nichtterminale: Q_0 (Startsymbol), Q_1 , Q_2

Terminale: 0, 1

Produktionen:

$$Q_0 \rightarrow 0Q_0 \mid 1Q_1 \mid \varepsilon$$

$$Q_1 \rightarrow 0Q_1 \mid 1Q_2$$

$$Q_2 \rightarrow 0Q_2 \mid 1Q_0$$

Beispiel für eine Ableitung von $w = 10011$:

$$Q_0 \Rightarrow 1Q_1 \Rightarrow 10Q_1 \Rightarrow 100Q_1 \Rightarrow 1001Q_2 \Rightarrow 10011Q_0 \Rightarrow 10011$$

\diamond

11.6 Techniken für den Entwurf von kontextfreien Grammatiken

Komplexere KFGs können oft in mehrere einfachere KFGs mit den Startsymbolen S_1, \dots, S_k aufgeteilt und danach mit den Regeln

$$S \rightarrow S_1 \mid S_2 \mid \dots \mid S_k$$

kombiniert werden. Oder um eine KFG für eine reguläre Sprache zu erstellen, kann zuerst ein DEA erstellt werden und dieser dann in eine KFG umgewandelt werden (siehe vorhergehender Abschnitt.)

Kontextfreie Sprachen enthalten manchmal auch Teilwörter, die voneinander "abhängig" sind. Eine KFG für diese Situation kann dann mit einer Regel in der Art

$$R \rightarrow uRv$$

behandelt werden. Ein Beispiel für eine solche Abhängigkeit von Teilwörtern:

§11-25 Beispiel.

$$R \rightarrow \text{if } Q \text{ then } R \text{ else } R$$

für die Abhängigkeit von "if", "then" und "else". ◊

Wir haben auch schon gesehen, dass komplexere Sprachen meist rekursiv aufgebaut sind: Steht zum Beispiel das Nichtterminal A für einen Ausdruck, kann A wiederum überall dort verwendet werden, wo dieser Ausdruck erlaubt ist. Dieser rekursive Aufbau ist ein sehr starkes Mittel und ermöglicht bereits recht komplexe Sprachen mittels kontextfreier Grammatiken zu formulieren.

Noch eine kleine Warnung: Ein häufiger Fehler beim Entwurf einer KFG ist, dass neben den Wörtern der Sprache auch noch (unbemerkt) Wörter erzeugt werden, die nicht zur Sprache gehören. Das darf natürlich nicht sein!

11.7 Zentrale Anwendungen in der Informatik

Zum Schluss noch einmal ein paar zentrale Anwendungen von kontextfreien Grammatiken in der Informatik zusammengefasst:

- **Compiler / Parsergeneratoren:** Kontextfreie Grammatiken sind wichtig als Teil der Syntaxanalyse: zum Beispiel zum Erkennen von Programmstrukturen wie Klammern, Bedingungen (if – else), Blöcke (begin – end) ...

§11-26 Beispiel.

Frage: Wie lautet eine KFG für die Erkennung von if – else Blöcken, wenn else optional ist?

Mögliche Lösung:

$$G = \{\{A\}, \{if, else\}, P, A\}$$

mit $P = A \rightarrow if A else A \mid if A \mid \epsilon$ ◊

- **Interpretation von Markup-Sprachen:** z. B. HTML, XML, ...
- **Reguläre Ausdrücke (Regex)** \sim ist sogar reguläre Sprache.
- ...

Kapitel 12

Anwendung: Reguläre Ausdrücke

Kapitelinhalt

12.1	Einführung und Motivierung	251
12.2	Syntax regulärer Ausdrücke	252
12.2.1	Die Standard-Syntax regulärer Ausdrücke	252
12.2.2	Erweiterte Syntax regulärer Ausdrücke	253
12.3	Semantik regulärer Ausdrücke	254
12.3.1	Sprache eines regulären Ausdrucks	254
12.3.2	Eigenschaften regulärer Ausdrücke	255
12.3.3	Äquivalenz DEA und RA	255

L E R N Z I E L E :

LZ 12.1 Sie verstehen den Aufbau von regulären Ausdrücken und ihre Bedeutung in der Informatik.

12.1 Einführung und Motivierung

Viele Probleme in der Informatik beinhalten die Prüfung, ob gewisse Wörter zu einer gegebenen Sprache gehören:

- Ist eine Benutzereingabe sinnvoll?
- Gehört ein Wort zu einem Wörterbuch (Rechtschreibprüfung)?
- Bestimmte Folgen von Eingaben (sogenannte "Events", welche als Eingabewörter aufgefasst werden können) sollen bestimmte Reaktionen auslösen (z.B. ein Getränkeautomat).
- Mustersuche in Texten.
- Lexikalische Analyse (in Compilern).
- Erkennung von Schlüsselwörtern ("Tokens").
- Syntax Test (für einfache Syntax)
- ... usw.

Da IT-Systeme nur über endliche Speicherressourcen verfügen, ist es wichtig, Sprachen endlich repräsentieren zu können. Unendlich grosse Sprachen werden so einer maschinellen Bearbeitung überhaupt erst zugänglich gemacht^[E.47].

Reguläre Ausdrücke sind Wörter über einem bestimmten Alphabet, welche Sprachen beschreiben. Sie sind also eine weitere Möglichkeit (gewisse) Sprachen endlich zu repräsentieren. Als solches ist dies selbst eine Sprache und wird als die "Sprache der regulären Ausdrücke über diesem Alphabet" bezeichnet.

Die Syntax der regulären Ausdrücke befasst sich dabei mit der Frage, welche Form (strukturelle Gestalt) diese regulären Ausdrücke haben. Dies wird Thema des nächsten Abschnitts sein. In der Semantik der regulären Ausdrücke wird hingegen erklärt, wie man reguläre Ausdrücke als Sprachen interpretiert. Dies ist ja generell das Wesen von Semantik, der Bedeutungslehre einer bestimmten Sache. Dazu kommen wir dann im übernächsten Abschnitt.

Zuerst noch zwei einfache Beispiele zur Demonstration der Idee:

§12-1 Beispiel. (Sprache der regulären Ausdrücke aller Binärwörter der Länge 4)

Ein regulärer Ausdruck, der die hier noch endliche Sprache aller Binärwörter der Länge 4 beschreibt:

$$(0|1) \quad (0|1) \quad (0|1) \quad (0|1)$$

steht für "0 oder 1" nochmals ein drittes Mal und ein vierstes Mal

Ein passender regulärer Ausdruck ist also:

$$(0|1)(0|1)(0|1)(0|1)$$



§12-2 Beispiel. (Sprache der Binärwörter, die das Teilwort 00 enthalten)

Ein regulärer Ausdruck für die jetzt unendliche Sprache der Binärwörter, die das Teilwort 00 enthalten:

$$(0|1)^*$$

bedeutet beliebig oft 0 oder 1
oder auch nie

00

das Teilwort genau so

(0|1)*

bedeutet beliebig oft 0 oder 1
oder auch nie

Ein passender regulärer Ausdruck ist also:

$$(0|1)^*00(0|1)^*$$

◇

12.2 Syntax regulärer Ausdrücke

12.2.1 Die Standard-Syntax regulärer Ausdrücke

Die Syntax der regulären Ausdrücke über einem Alphabet ist folgendermassen rekursiv definiert:

§12-3 Definition. (Reguläre Ausdrücke über einem Alphabet Σ ; die Sprache Regex)

Es sei Σ ein beliebiges Alphabet. Die Sprache RA_Σ der regulären Ausdrücke über dem Alphabet Σ ist wie folgt definiert:

(1) $\emptyset, \varepsilon \in \text{RA}_\Sigma$

Die Sonderzeichen \emptyset und ε sind also selbst reguläre Ausdrücke.

(2) $a \in \text{RA}_\Sigma$ für ein $a \in \Sigma$

Jedes Symbol aus dem Alphabet Σ ist auch ein regulärer Ausdruck über Σ .

(3) $R \in \text{RA}_\Sigma \Rightarrow (R^*) \in \text{RA}_\Sigma$

Ist R ein regulärer Ausdruck über Σ , dann ist auch (R^*) ein regulärer Ausdruck über Σ .

(4) $R, S \in \text{RA}_\Sigma \Rightarrow (RS) := (R \cdot S) \in \text{RA}_\Sigma$

Sind R und S reguläre Ausdrücke über Σ , dann ist auch (RS) regulärer Ausdrücke über Σ .

(5) $R, S \in \text{RA}_\Sigma \Rightarrow (R|S) \in \text{RA}_\Sigma$

Sind R und S reguläre Ausdrücke über Σ , dann ist auch $(R|S)$ regulärer Ausdrücke über Σ .

Die Symbole $^*, \cdot, |$ werden reguläre Operatoren genannt^[E.48]. Da $(R \cdot S)$ gerade die Bedeutung der Aneinanderhängung (Konkatenation) von R und S hat, wird \cdot meist weggelassen.

Die Sprache RA_Σ wird manchmal auch als die Sprache **Regex** oder **RegExp** bezeichnet und ihre Wörter als reguläre Ausdrücke (englisch **regular expressions**).

◇

An dieser Stelle bietet es sich an, zu fragen, was diese Syntax-Definition tun soll. Dies ist aber schon eine Frage der Interpretation und wird auf den nächsten Abschnitt verschoben. Die Bedeutung (Interpretation) obiger Definition ist also bis auf ihre Bedeutung für die Syntax noch offen und wir haben erst einmal einfach definiert, wann ein Wort alleine von

der Struktur her zu dieser Sprache gehört. Man sagt auch, wann ein Wort eine sogenannte wohlgeformte Formel (englisch: well-formed formula, w) in dieser Sprache ist. [E.49]

§12-4 Beispiel. (Reguläre Ausdrücke über dem Alphabet $\{a, b\}$)

Ein paar reguläre Ausdrücke – also wohlgeformte Formeln (w) der Sprache RA_{Σ} – über dem Alphabet $\Sigma := \{a, b\}$ sind:

- $((((aa)^*)(b^*))(a(ba)))$
- $((a|(ab))^*)$
- $((ab)|(ba))$
- $(a(b(ba)))$

◊

Die Menge RA_{Σ} der regulären Ausdrücke über dem Alphabet Σ ist, wie gesagt, selbst eine Sprache und zwar über dem Alphabet

$$\{\emptyset, \varepsilon, ^*, (,), |\} \cup \Sigma$$

(Wenn wir das Konkatenationssymbol · gleich für immer weglassen.) Bei Wörtern werden zudem aus Gründen der Lesbarkeit "überflüssige" Klammern oft weggelassen. Ist ein regulärer Ausdruck aufgrund von weggelassenen Klammern nicht mehr eindeutig lesbar, so gilt folgende Rangfolge der Operatoren:

§12-5 Definition. (Rangfolge der Operatoren in RA_{Σ})

- * vor Konkatenation ·.
- Konkatenation · vor |.

◊

§12-6 Beispiel.

Der Ausdruck $ab^*|c$ wird beispielsweise als $((a(b^*))|c)$ verstanden.

◊

12.2.2 Erweiterte Syntax regulärer Ausdrücke

Es gibt unzählige Erweiterungen der Sprache der regulären Ausdrücke. Einige weit verbreitete abkürzende Schreibweisen sind:

§12-7 Definition. (Erweiterte Syntax von RA_{Σ})

- (1) Ist R ein regulärer Ausdruck, dann steht

$$(R^+)$$

für $R(R^*)$. D.h., R kommt also mindestens einmal vor.

- (2) Ist R ein regulärer Ausdruck, dann steht

$$(R?)$$

für $(R|\varepsilon)$. D.h., R kommt also höchstens einmal vor.

- (3) Sind R_1, \dots, R_k reguläre Ausdrücke, dann steht

$$[R_1, \dots, R_k]$$

für $R_1|R_2| \dots |R_k$. Das heisst, genau einer der Ausdrücke R_i , $i \in \{1, \dots, k\}$, kommt vor^[E.50].

◇

12.3 Semantik regulärer Ausdrücke

12.3.1 Sprache eines regulären Ausdrucks

Ein Wort R der Sprache RA_Σ , also ein regulärer Ausdruck über Σ , kann nun selber wieder eine Sprachen definieren. Dies ist die Sprache $L(R)$ des entsprechenden regulären Ausdrucks R :

§12-8 Definition. (Die Sprache $L(R)$ eines regulären Ausdrucks $R \in \text{RA}_\Sigma$)

Es sei Σ ein beliebiges Alphabet. Für jeden regulären Ausdruck $R \in \text{RA}_\Sigma$ definieren wir die Sprache $L(R)$ von R wie folgt:

- $L(\emptyset) = \emptyset$

Die Sprache des regulären Ausdrucks $R = \emptyset$ ist die leere Sprache. D.h., der reguläre Ausdruck \emptyset beschreibt die leere Sprache \emptyset .

- $L(\varepsilon) = \{\varepsilon\}$

Die Sprache des regulären Ausdrucks ε ist die Sprache, welche nur das leere Wort enthält. D.h., der reguläre Ausdruck ε beschreibt die Sprache, welche gerade nur das leere Wort enthält.

- $L(a) = \{a\}$ für $a \in \Sigma$

Jedes Symbol a des Alphabets Σ beschreibt gerade die Sprache, welche nur dieses Symbol selbst als Wort enthält.

- $L(R^*) = L(R)^*$

Der reguläre Ausdruck R^* beschreibt die Sprache aller endlicher Wörter der Sprache, welche durch R beschrieben ist. (Der Stern rechts ist der Kleene-Stern.)

- $L(R|S) = L(R) \cup L(S)$

Der reguläre Ausdruck $(R|S)$ beschreibt also die Sprache der Wörter, welche entweder von R oder von S beschrieben sind.

- $L(RS) = L(R)L(S)$

Der reguläre Ausdruck (RS) beschreibt also die Sprache der Wörter, welche durch Konkatenation von Wörtern der von R und der von S beschriebenen Sprachen entstehen.



12.3.2 Eigenschaften regulärer Ausdrücke

§12-9 Satz. (Rechenregeln für reguläre Ausdrücke)

Für jedes Alphabet und alle regulären Ausdrücke $R, S, T \in RA_{\Sigma}$ gelten folgende Identitäten:

- $L(R|S) = L(S|R)$ (Kommutativgesetz)
- $L(R(ST)) = L((RS)T)$ (Assoziativgesetz für Konkatenation)
- $L(R|(S|T)) = L((R|S)|T)$ (Assoziativgesetz für |)
- $L(R(S|T)) = L(RS|RT)$ (Distributivgesetz)
- $L((R^*)^*) = L(R^*)$ (Idempotenzgesetz für *)
- $L(R|R) = L(R)$ (Idempotenzgesetz für |)



Beweis. Elementare Mengenumformungen. □

12.3.3 Äquivalenz DEA und RA

§12-10 Definition.

Eine Sprache A über dem Alphabet Σ heisst **regulär**, falls $A = L(R)$ für einen regulären Ausdruck $R \in RA_{\Sigma}$ gilt. ◊

§12-11 Beispiele.

(1) Für $R = 0|(-?)[1, \dots, 9][0, \dots, 9]^*$ gilt

$L(R) =$ Menge der ganzen Zahlen in Dezimaldarstellung

(2) Für $R = (0?)(10)^*(1?)$ oder $R = (10)^*|(01)^*|(10)^*1|(01)^*0$ gilt

$L(R) =$ Menge der Binärwörter mit abwechselnd Nullen und Einsen



Die Sprachen welche das Konzept eines deterministischen Automaten (DEA) beschreiben kann, sind genau diejenigen, welche auch das Konzept der Sprache eines regulären Ausdrucks $L(R)$ beschreiben kann. Endliche Automaten und reguläre Ausdrücke sind also "gleich stark":

§12-12 Satz. (Gleichmächtigkeit von RA und DEA (Hauptsatz von Kleene))

Es gibt einen DEA, der die Sprache L akzeptiert. \iff Es gibt einen RA, der die Sprache L beschreibt. \diamond

Beweis. [E.51]

□

Kapitel 13

Reguläre Sprachen, 2.Teil

Kapitelinhalt

13.1 Übersicht über Darstellungsarten für reguläre Sprachen	259
13.2 Abschlusseigenschaften regulärer Sprachen	259

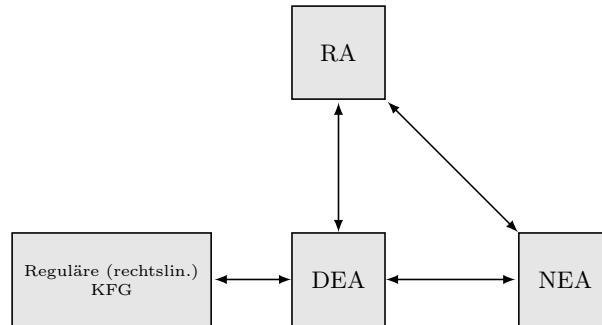
L E R N Z I E L E :

- LZ 13.1** Sie kennen die verschiedenen Darstellungsarten regulärer Sprachen.
LZ 13.2 Sie kennen die vorgestellten Abschlusseigenschaften regulärer Sprachen.
-

13.1 Übersicht über Darstellungsarten für reguläre Sprachen

Damit können wir zusammenfassen, dass reguläre Sprachen also durch verschiedene äquivalente Mechanismen darstellbar sind:

- **Akzeptierender Mechanismus:** DEA, NEA
- **Beschreibender Mechanismus:** RA
- **Generierender Mechanismus:** Reguläre (rechtslineare) kontextfreie Grammatiken



13.2 Abschlusseigenschaften regulärer Sprachen

Reguläre Sprachen haben schliesslich auch wichtige Abschlusseigenschaften bzgl. den zentralen Operationen zwischen Sprachen. Zuerst, welches sind allgemein die wichtigsten Operationen zwischen zwei (beliebigen) formalen Sprachen? Es sind dies teilweise die "natürlichen" Operationen, welche von den Mengenoperationen "induziert" werden (da ja formale Sprachen letztlich einfach Mengen von Wörter sind):

§13-1 Definition. (Operationen zwischen zwei formalen Sprachen)

Seien L, L_1 und L_2 formale Sprachen über einem Alphabet Σ .

- Die Vereinigung der beiden Sprachen L_1 und L_2 ist definiert durch

$$L_1 \cup L_2 := \{w \mid w \in L_1 \text{ oder } w \in L_2\}.$$

- Der Schnitt bzw. Durchschnitt der beiden Sprachen L_1 und L_2 ist definiert durch

$$L_1 \cap L_2 := \{w \mid w \in L_1 \text{ und } w \in L_2\}.$$

- Die Differenz der Sprache L_1 und L_2 ist definiert durch

$$L_1 - L_2 := L_1 \setminus L_2 = \{w \mid w \in L_1 \text{ und } w \notin L_2\}.$$

- Die Konkatenation (Verkettung) der Sprache L_1 mit L_2 ist definiert durch

$$L_1 \cdot L_2 = L_1 L_2 := \{w = w_1 w_2 \mid w_1 \in L_1 \text{ und } w_2 \in L_2\}.$$

- Die Kleenesche Hülle der Sprache L ist definiert durch

$$L^* := \{w = w_1 w_2 \dots w_n \mid w_i \in L \text{ für alle } i \in \{1, 2, \dots, n\}\}.$$

- Das Komplement von L ist definiert durch

$$\overline{L} := \Sigma^* - L = \{w \in \Sigma^* \mid w \notin L\}.$$

◊

Man beachte hier, dass Vereinigung, Durchschnitt und Differenz völlig der Mengenoperation entsprechen (im Zusammenhang mit Sprachen symbolisiert man die Differenz aber meist wie angegeben mit $L_1 - L_2$). Bei der Konkatenation ist es ähnlich, wenn es auch ziemlich anders notiert wird: Sie kann als eine analoge Operation wie das (binäre) kartesische Produkt von Mengen aufgefasst werden. Die Kleeneschen Hülle entspricht hingegen keiner der elementaren Mengenoperationen mehr. Auch mengetheoretisch gelesen ist dies eine neue Operation^[E.52]. Von der Bedeutung her, spielt die Kleenesche Hülle aber die Rolle der Grundmenge in der Mengenlehre und deshalb ist das Komplement einer Sprache auch wieder analog dem Komplement einer Menge bzgl. einer Grundmenge.

Für den Fall regulärer Sprachen haben wir nun die folgenden wichtigen Abschlusseigenschaften^[E.53]:

§13-2 Satz und Definition. (Abschluss bzgl. Vereinigung)

Seien L_1 und L_2 zwei reguläre Sprachen über Σ .

Dann ist auch die Vereinigung $L_1 \cup L_2$ regulär. ◊

Beweis. Weil L_1 und L_2 regulär sind, existieren reguläre Ausdrücke α_1 und α_2 für L_1 und L_2 , das heisst $L(\alpha_1) = L_1$ und $L(\alpha_2) = L_2$.

Aus der Definition für reguläre Ausdrücke folgt, dass $\alpha_1|\alpha_2$ ein regulärer Ausdruck ist, also ist

$$L(\alpha_1|\alpha_2) = L_1 \cup L_2$$

regulär. □

§13-3 Satz und Definition. (Abschluss bzgl. Komplement)

Sei L eine reguläre Sprache über Σ .

Dann ist auch das Komplement $\overline{L} := \Sigma^* - L$ regulär. ◊

Beweis. Sei A ein EA für L . Dann gibt es für jedes Wort $w \in \Sigma^*$ einen eindeutigen Zustand, in dem der EA endet, wenn er w liest. Alle Wörter aus $L(A) = L$ führen in akzeptierende Zustände und alle Wörter aus $\Sigma^* - L = \overline{L}$ in nichtakzeptierende Zustände. Vertauschen von akzeptierenden und nichtakzeptierenden Zuständen ergibt einen EA für \overline{L} . □

Auch bezüglich der anderen Operationen sind die regulären Sprachen abgeschlossen:

§13-4 Satz und Definition. (Weitere Abschlusseigenschaften)

Seien L, L_1 und L_2 beliebige reguläre Sprachen über Σ .

Dann sind auch der Durchschnitt $L_1 \cap L_2$, die Differenz $L_1 - L_2$, die Konkatenation $L_1 \cdot L_2 = L_1 L_2$ und die Kleenesche Hülle L^* reguläre Sprachen. \diamond

Kapitel 14

Die Chomsky-Hierarchie

Kapitelinhalt

14.1 Rückblick: Reguläre (oder Typ-3) und kontextfreie (oder Typ-2) Grammatiken und Sprachen	265
14.2 Kontextsensitive (oder Typ-1) Grammatiken und Sprachen	265
14.3 Unbeschränkte Grammatiken und rekursiv aufzählbare Sprachen (Typ-0)	266
14.4 Chomsky-Hierarchie	267

L E R N Z I E L E :

LZ 14.1 Sie kenne die Chomsky-Hierarchie, ihre Bedeutung und die Unterscheidung der einzelnen Klassen der Hierarchie.

14.1 Rückblick: Reguläre (oder Typ-3) und kontextfreie (oder Typ-2) Grammatiken und Sprachen

Wir haben bisher gesehen, dass jeder endlicher Automat durch die Grammatik mit Produktionen der Form $Q_i \rightarrow \varepsilon$ oder $Q_i \rightarrow aQ_j$ umgesetzt werden kann. Umgekehrt gilt auch, dass alle Grammatiken mit den einzigen Produktionen der Form $Q_i \rightarrow \varepsilon$ und $Q_i \rightarrow aQ_j$ nur reguläre Sprachen (Typ-3 Sprachen) erzeugen. Wir haben deshalb solche Grammatiken regulär oder Typ-3 Grammatiken genannt:

§14-1 Definition. (Reguläre Grammatik (Typ-3 Grammatik))

Eine reguläre Grammatik G (Typ-3 Grammatik) ist ein 4-Tupel (N, Σ, P, S) , wobei die Produktionen die Form

$$X \rightarrow aY$$

oder

$$X \rightarrow \varepsilon$$

mit einem Terminal $a \in \Sigma$ und Nichtterminalen $X, Y \in N$ haben. \diamond

Erlauben wir im Rumpf der Produktionen beliebige Satzformen, bekommen wir die kontextfreien Grammatiken, die wir auch schon kennen:

§14-2 Definition. (Kontextfreie Grammatik (Typ-2 Grammatik))

Eine kontextfreie Grammatik G (Typ-2 Grammatik) ist ein 4-Tupel (N, Σ, P, S) , wobei die Produktionen die Form

$$X \rightarrow \beta$$

mit einem Nichtterminal $X \in N$ und einem Rumpf $\beta \in (N \cup \Sigma)^*$ haben. \diamond

14.2 Kontextsensitive (oder Typ-1) Grammatiken und Sprachen

Bei den regulären und den kontextfreien Grammatiken ist im Kopf der Produktion $X \rightarrow \beta$ nur ein Nichtterminal $X \in N$ erlaubt. Jedes Nichtterminal wird dementsprechend unabhängig vom Kontext ersetzt.

Lässt man nun bei den Produktionen eine Kontextinformation im Kopf zu, aber schränkt die Produktionen noch so ein, dass der Rumpf mindestens so lang wie der Kopf ist, spricht man von kontextsensitiven Grammatiken oder Typ-1 Grammatiken:

§14-3 Definition. (Kontextsensitive Grammatik (Typ-1 Grammatiken))

Eine kontextsensitive Grammatik G (Typ-1 Grammatiken) ist ein 4-Tupel

(N, Σ, P, S) , wobei die Produktionen die Form

$$\alpha X \beta \rightarrow \alpha \gamma \beta \quad \text{oder} \quad S \rightarrow \varepsilon$$

mit Satzformen $\alpha, \beta \in (N \cup \Sigma)^*$, der nichtleeren Satzform $\gamma \in (N \cup \Sigma)^+$, einem Nichtterminal $X \in N$ haben und das Startsymbol in keinem Rumpf vorkommt. \diamond

§14-4 Beispiel. (Kontextsensitive Grammatik und ihre Sprache)

Grammatik $G = (\{S, B, C\}, \{a, b, c\}, P, S)$ mit den Produktionen

$$\begin{aligned} S &\rightarrow aSBC \mid aBC \\ CB &\rightarrow BC \\ aB &\rightarrow ab \\ bB &\rightarrow bb \\ C &\rightarrow c \end{aligned}$$

Diese Grammatik erzeugt die Sprache $\{a^n b^n c^n \mid n \geq 1\}$. \diamond

- Mögliche Ableitung des Wortes $aabbcc$:

$$\begin{aligned} S &\Rightarrow aSBC \Rightarrow aaBCBC \Rightarrow aaBBCC \Rightarrow aabBCC \\ &\Rightarrow aabbCC \Rightarrow aabbC \Rightarrow aabbcc \end{aligned}$$

14.3 Unbeschränkte Grammatiken und rekursiv aufzählbare Sprachen (Typ-0)

Es ist naheliegend, den Weg weiterzugehen: Lassen wir bei den Produktionen alle Einschränkungen weg, bekommen wir die Typ-0-Grammatiken (unbeschränkten Grammatiken):

§14-5 Definition. (Typ-0-Grammatik)

Eine Typ-0-Grammatik G (unbeschränkte Grammatik) ist ein 4-Tupel

$$(N, \Sigma, P, S),$$

wobei die Produktionen die allgemeine Form $\alpha \rightarrow \beta$ mit Satzformen $\alpha, \beta \in (N \cup \Sigma)^*$ haben. \diamond

Die Typ-0-Grammatiken erzeugen sogenannte rekursiv aufzählbare Sprachen, auch Typ-0 Sprachen genannt. Dieser Sprachtyp entspricht all jenen Sprachen, die man automatisch erzeugen kann (\rightsquigarrow nach momentanem Wissenstand: das hängt mit der berühmten “Church-Turing These” zusammen).

Später werden Sie im Modul “Theoretische Informatik” sehen, dass es auch ein Automatenmodell (die berühmten “Turingmaschinen”) gibt, welches rekursiv aufzählbare Sprachen erkennt. Im Gegensatz zu endlichen Automaten ist bei rekursiv aufzählbaren Sprachen aber nicht gefordert, dass die “Turingmaschine” für Wörter, die nicht in der Sprache sind, halten muss.

14.4 Chomsky-Hierarchie

In der Chomsky-Hierarchie (1956) vom Linguisten Noam Chomsky werden die verschiedenen, vorgehend skizzierten Klassen formaler Sprachen zusammengefasst. In folgender Tabelle bezeichnen dabei $a \in \Sigma$ ein Terminal, $X, Y \in N$ Nichtterminale und $\alpha, \beta \in (N \cup \Sigma)^*$, $\gamma \in (N \cup \Sigma)^+$ Satzformen.

Typ	Sprache	Produktionen der Form
Typ-3-Grammatiken	Reguläre Sprachen	$X \rightarrow aY$ oder $X \rightarrow \varepsilon$
Typ-2-Grammatiken	Kontextfreie Sprachen	$X \rightarrow \beta$
Typ-1-Grammatiken	Kontextsensitive Sprachen	$\alpha X \beta \rightarrow \alpha \gamma \beta$ oder $S \rightarrow \varepsilon$ und S kommt in keinem Rumpf vor
Typ-0-Grammatiken	Rekursiv aufzählbare Sprachen	keine Einschränkung

Tabelle 14.1 – Zusammenfassung der Sprachklassen mittels Grammatiken.

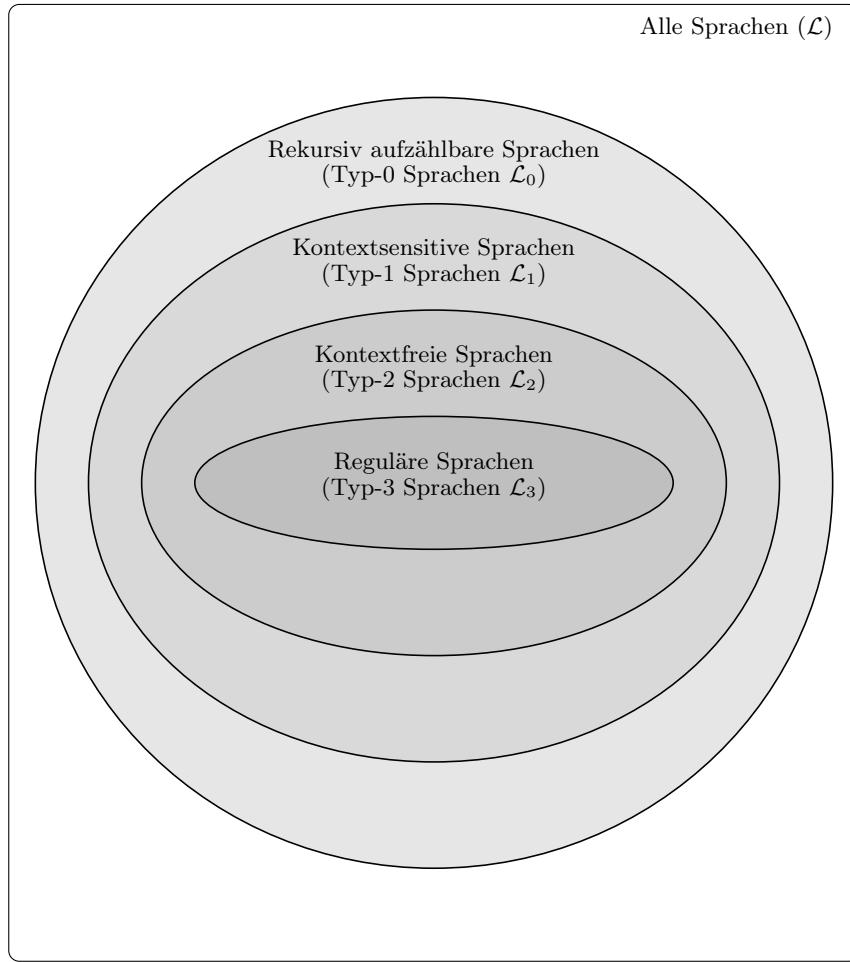


Abbildung 14.1 – Die Chomsky-Hierarchie der Sprachklassen.

Appendix

Ergänzende Notizen

- [E.1] Auf die tieferen und grundlegenden Probleme und Fragen betreffend diesem Fundament können wir in diesem Modul leider nicht näher eingehen.
- [E.2] Eine **Primzahl** ist eine natürliche Zahl grösser 1, welche nur durch sich selbst und 1 ganzzahlig teilbar ist. Eine natürliche Zahl grösser 1, welche nicht Primzahl ist, nennen wir eine **zusammengesetzte Zahl**, weil sie sich dann eindeutig in Primfaktoren zerlegen lässt.
- [E.3] Oder wenn der gedrehte Implikationspfeil irritiert, kann man sich auch
$$(\forall x : P(x)) \vee (\forall x : Q(x)) \implies \forall x : (P(x) \vee Q(x))$$
merken.
- [E.4] Dies ist eine sogenannte "naive" Definition einer Menge. In der *axiomatischen Mengenlehre* hingegen, präzisiert man den Begriff "Menge" noch weiter aufgrund einer möglichst minimalen Anzahl von "Axiomen".
- [E.5] Aber nicht nur: In der höheren Mathematik treten Mengen von unterschiedlichsten mathematischen Objekten in den Fokus!
- [E.6] Die Bezeichnung \mathbb{N}_0 wollen wir nicht zulassen, da sie unterschiedlich aufgefasst werden kann.
- [E.7] Ein "Korollar" ist in der Mathematik eine einfache unmittelbar aus einer Definition oder einem Satz folgende Tatsache.
- [E.8] Eine andere Schreibweise für die Potenzmenge $\mathcal{P}(A)$ einer Menge A ist auch 2^A .
- [E.9] Die Potenzmenge selbst ist ja gemäss Definition gerade ein spezielles Mengensystem.
- [E.10] Jede Platte einer HDD wird von beiden Seiten je mit einem Schreib-/Lesekopf (Head) beschrieben bzw. gelesen. Deshalb hat es bei vier Platten acht Heads.
- [E.11] Wir merken hier zur Ergänzung noch an, dass die Massenspeicher-Technologien der neuesten Generation so genannte SSD (solid-state drives) sind. Diese beruhen auf einem völlig anderen Prinzip ohne die beweglichen elektromechanischen Teile wie bei den HDD's.
- [E.12] Dies sind die Axiome der sogenannten "Zermelo-Fraenkelschen Mengenlehre mit Auswahlaxiom" (ZFC).
- [E.13] Und manchmal ist es sogar so, dass man erst beim Suchen nach Gegenbeispielen in einem ersten Schritt erkennt, wie man die Aussage allenfalls positiv beweisen könnte. Jedenfalls, wenn die Aussage entgegen den anfänglichen Erwartungen vielleicht nicht falsch, sondern richtig ist.
- [E.14] Bei *partiellen* Funktionen reicht es, die *Injektivität* zu fordern.
- [E.15] Ein Korollar ist in der Mathematik ein Satz der sofort und unmittelbar, ohne einen weiteren langen Beweis, aus einem vorhergehenden Satz folgt.
- [E.16] Wir wollen also zeigen, dass §5-9, (iii), und §5-21 semantisch äquivalent sind. Dies wollen wir anhand einer Wahrheitstabelle tun. D.h., wir wollen zeigen, dass bei einer Relation $R \subseteq A \times A$ für alle $x, y \in A$ gilt:

Behauptung:

$$(x \neq y \wedge xRy \Rightarrow (y, x) \notin R) \text{ ist semantisch äquivalent zu } (xRy \wedge yRx \Rightarrow x = y)$$

(Wir benutzen hier zur besseren Leserlichkeit im Zusammenhang mit der Verneinung die Schreibweise $(x, y) \in R$ anstatt xRy . Das $(x, y) \notin R$ würde ja einem " xRy " entsprechen. Diese Notation ist aber sehr unüblich.)

Beweis:

In beiden Bedingungen sind die drei wesentlichen atomaren Aussagen die Folgenden (andere Teile sind dann Negationen und Verknüpfungen hiervon):

$$f : (x, y) \in R,$$

$$\begin{aligned} g : \quad & (y, x) \in R, \\ h : \quad & x = y \end{aligned}$$

Mit diesen Abkürzungen lautet die zu zeigende **Behauptung** also:

$$(\neg h \wedge f \implies \neg g) \text{ ist semantisch äquivalent zu } (f \wedge g \implies h)$$

Für alle möglichen Belegungen dieser drei Aussagen f, g, h bauen wir unsere Wahrheitstabelle auf und erhalten:

f	g	h	$\neg h$	$\neg h \wedge f$	$\neg g$	$(\neg h \wedge f) \implies \neg g$	$f \wedge g$	$(f \wedge g) \implies h$
0	0	0	1	0	1	1	0	1
0	0	1	0	0	1	1	0	1
0	1	0	1	0	0	1	0	1
0	1	1	0	0	0	1	0	1
1	0	0	1	1	1	1	0	1
1	0	1	0	0	1	1	0	1
1	1	0	1	1	0	0	1	0
1	1	1	0	0	0	1	1	1

Wir sehen, die zwei betreffenden Spalten (die Spalten 7 und 9) sind identisch, was die Äquivalenz unserer zwei Bedingungen zeigt. \square

Noch eine kleine ergänzende **Anmerkung**: Eine weitere äquivalente Bedingung für die Antisymmetrie erhält man durch *Kontraposition* der Bedingung $(x \neq y \wedge xRy \Rightarrow (y, x) \notin R)$ der Definition §5-9, (iii), wobei dann meistens die Namen von x und y wieder vertauscht benutzt sind (was natürlich egal ist):

$$R \subseteq A \times A \text{ ist antisymmetrisch, genau dann wenn gilt:}$$

$$\forall x, y \in A : (xRy \implies ((y, x) \notin R \vee x = y)).$$

Dies ist also eine weitere oft benutzte Charakterisierung der Antisymmetrie.

- [E.17] Möglicherweise kennen Sie das Prinzip schon von der Verknüpfung von Funktionen.
- [E.18] Wenn Ihnen diese Definition zu formal ist, dann merken Sie sich einfach, dass ein Pfad in einem gerichteten Graphen eine "zusammenhängende gerichtete Kantenfolge" von Kanten des Graphen ist.
- [E.19] deg für english "degree" = Grad.
- [E.20] Für eine weniger formale Definition, können Sie sich auch einfach merken, dass ein Weg (Pfad) in einem ungerichteten Graphen eine "zusammenhängende ungerichtete Kantenfolge" im Graphen ist.
- [E.21] Es gibt verschiedene Arten "Multigraphen" zu definieren. Eine andere Möglichkeit ist, einen Multigraphen als Tripel $X = (V, E, f : E \rightarrow V \times V)$ darzustellen, wobei E erstmals nur eine Menge von Kantenlabels ist (z.B., e_1, e_2, \dots, e_n), und f dann eine Funktion ist, welche jedem Kantenlabel e_i das Paar seiner Knoten zuordnen, also $f(e_i) := (v_j, v_k)$.
- [E.22] Manchmal auch "zugrunde liegende" ungerichtete Graph [...].
- [E.23] Man beachte, dass aufgrund dieser Definition, Schleifen im Digraphen X keine Entsprechung im unterliegenden ungerichteten Graphen haben, dort also verschwinden und diese Information "verloren" geht.
- [E.24] Beachten Sie, dass man aus historischen Gründen auch einen *Multigraphen*, welcher einen Eulerkreis besitzt, nur "Eulergraph" nennt, obwohl man hier eigentlich genauer von einem Eulermultigraphen sprechen müsste.
- [E.25] Wir werden sehen (Stichwort: Eulerbedingungen), dass dies einen bestimmten, wichtigen Grund hat!
- [E.26] Isolierte Knoten spielen für einen Eulerweg ja gar keine Rolle.
- [E.27] Die Idee ist dem Text "[Minimale aufspannende Bäume](#)" von Katharina Langkau und Martin Skutella entlehnt.
- [E.28] Was "geeignet" ist, können wir hier nicht vertiefen. Dies führt im Wesentlichen auf den Begriff einer "Wohlordnung" und die sogenannte Ordnungstheorie in der Mathematik.
- [E.29] In vielen Texten zu Rekursionen wird die 0 nicht zur Menge \mathbb{N} dazu gerechnet.
- [E.30] Man definiert dann zusätzlich noch: $0! := 1$ und hat so sogar eine Funktion auf $\mathbb{N} \cup \{0\}$.
- [E.31] Sie ist nach Leonardo Fibonacci benannt, der damit im Jahr 1202 das Wachstum einer Kaninchenpopulation beschrieb.
- [E.32] Woher man diese Ansätze hat, können wir leider hier nicht besprechen. Man erhält sie im wesentlichen basierend auf der "Theorie linearer Differentialgleichungen". (Und dort erklärt sich dann z.B. auch der Begriff "Störterm".)
- [E.33] Über den faszinierenden Hintergrund der weitverbreiteten Fibonacci-Folge empfehle ich als Einstieg zum Beispiel den Wikipedia-Artikel <https://de.wikipedia.org/wiki/Fibonacci-Folge>.

- [E.34] In der linearen Algebra und der Geometrie spricht man übrigens oft auch von "Koordinatentransformation".
- [E.35] In diesem Teilgebiet wird die Null meistens wieder zu den natürlichen Zahlen hinzugerechnet und Potenzreihen von 0 an indiziert...
- [E.36] Wenn wir die Folge als $a := (a_n)_{n \in \mathbb{N}}$ schreiben, ist das natürlich dasselbe wie $G(a; x) := \sum_{n=0}^{\infty} a_n x^n$.
- [E.37] Ob Ihr Code auch *semantisch* korrekt ist, d.h., ob das Programm auch tut, was es tun soll und ob es damit seine Spezifikation erfüllt, ist eine andere, ebenso wichtige Frage, die wir aber hier im Moment ausser acht lassen wollen.
- [E.38] Und nicht etwa unser Gehirn, welches ein anderes, verwandtes und spannendes Forschungsthema ist.
- [E.39] Deswegen wird die Konkatenation von Wörtern manchmal auch mit dem "Kringel" \circ bezeichnet: $x \circ y$.
- [E.40] Zum Beispiel: die späteren endlichen Automaten und formalen Grammatiken.
- [E.41] Mehr dazu im Modul "Theoretische Informatik"!
- [E.42] Es ist wichtig, hier noch einmal in Erinnerung zu rufen, was "Abstraktion" heisst: Nämlich eine *Reduktion auf das Wesentliche* (*für einen betrachteten Aspekt*). Leider hat es sich eingebürgert, aus völliger Unwissenheit und mangelndem Verständnis der Sache die Wörter "Abstraktion" und "abstrakt" schon fast als Schimpwort für alles Mögliche und Unmögliche zu benutzen.
- [E.43] Zur Erinnerung: Das **kartesische Produkt von zwei Mengen A und B** ist definiert als $A \times B = \{(a, b) \mid a \in A \wedge b \in B\}$.
- [E.44] Im Modul "eti" werden wir dann sehen, dass die Aufhebung dieser Einschränkung zu noch bedeutend allgemeineren Grammatiken und Sprachen führt, bis hin zu den allgemeinst möglichen "Berechnungsmaschinen", den berühmten Turing-Maschinen.
- [E.45] Ein Palindrom entspricht rückwärts gelesen dem Wort selbst.
- [E.46] Ableitungsbäume kennen wir bereits aus der Aussagenlogik!
- [E.47] Beachten Sie, dass beispielsweise mit endlichen Automaten oder kontextfreien Grammatiken auch *unendliche* Sprachen mit endlichen Mitteln exakt definiert werden können!
- [E.48] Für $|$ wird manchmal auch $+$ verwendet. Dies ist aber etwas verwirrend, da $+$ teilweise auch für die Konkatenation von Strings verwendet wird, was aber gerade \cdot entspricht!
- [E.49] Das ist genau so wie in der Logik, wo beispielsweise " $A \wedge \neg B$ " eine syntaktisch korrekte logische Formel, also eine wohlgeformte Formel, ist, jedoch " $A \wedge \wedge B \neg$ " nicht syntaktisch korrekt und damit keine wff ist.
- [E.50] Streng genommen müsste man hier $R_1|(R_2|(\dots|R_k)\dots)$ schreiben.
- [E.51] Der Beweis liefert auch gleich eine Vorgehensweise, wie man von der einen Beschreibung zur anderen kommt.
 - \implies Dynamische Programmierung: Für jedes Paar von Zuständen p, q regulären Ausdruck finden, der alle Wörter beschreibt, die von p nach q führen.
 - \Leftarrow RA in einen speziellen NEA umwandeln, der auch spontane Übergänge (ohne ein Eingabesymbol zu lesen) zulässt. Diese sogenannten ε -NEAs können in NEAs und somit durch Teilmengenkonstruktion in DEAs umgewandelt werden.
- [E.52] In der Algebra wird dies dann das "freie Monoid über der Menge L " genannt.
- [E.53] Mit "Abschlusseigenschaft" meinen wir, dass die Operation auf reguläre Sprachen angewendet wieder eine *reguläre Sprache* ergibt. Die Operationen führen also nicht über die Klasse der regulären Sprachen hinaus.

Symbolverzeichnis

e	Eulersche Konstante (Euler Zahl) $e \approx 2.71828\dots$
id	Die Identitätsfunktion $\text{id} : \mathbb{R} \rightarrow \mathbb{R}, x \mapsto \text{id}(x) := x$. (D.h., jedem x wird einfach das x selbst zuordnet.)
$\mathbb{N} \setminus \{0\}$	Menge der natürlichen Zahlen ohne Null: $\{1, 2, 3, 4, \dots\}$
$\mathbb{N} = \mathbb{N} \cup \{0\}$	Menge der natürlichen Zahlen inkl. Null: $\{0, 1, 2, 3, 4, \dots\}$
\mathbb{N}_n	Menge der natürlichen Zahlen bis und mit n , für ein $n \in \mathbb{N} \setminus \{0\}$: $\{0, 1, 2, \dots, n\}$ (Wir wollen das Symbol \mathbb{N}_0 nicht verwenden, da es falsch verstanden werden könnte.)
π	Kreiszahl Pi $\pi \approx 3.14159\dots$
\mathbb{Q}	Menge der rationalen Zahlen, d.h., $\mathbb{Q} = \{q \mid \exists m \in \mathbb{Z} \wedge \exists n \in \mathbb{N} \setminus \{0\} : q = \frac{m}{n}\}$
\mathbb{R}	Menge der reellen Zahlen
$\mathbb{R}_{>0}$	Menge der positiven reellen Zahlen, d.h., $\mathbb{R}_{>0} := \{x \in \mathbb{R} \mid x > 0\}$
$\mathbb{R}_{\geq 0}$	Menge der nicht-negativen reellen Zahlen, d.h., $\mathbb{R}_{\geq 0} := \{x \in \mathbb{R} \mid x \geq 0\}$
$\mathbb{R}_{<0}$	Menge der negativen reellen Zahlen, d.h., $\mathbb{R}_{<0} := \{x \in \mathbb{R} \mid x < 0\}$
$\mathbb{R}_{\leq 0}$	Menge der nicht-positiven reellen Zahlen, d.h., $\mathbb{R}_{\leq 0} := \{x \in \mathbb{R} \mid x \leq 0\}$
\mathbb{Z}	Menge der ganzen Zahlen, d.h., $\mathbb{Z} = \{a \mid a \in \mathbb{N} \vee -a \in \mathbb{N} \vee a = 0\}$
$\mathbb{Z}_{>0}$	Menge der positiven, negativen ganzen Zahlen, d.h., $\mathbb{Z}_{>0} = \{a \in \mathbb{Z} \mid a > 0\}$
$\mathbb{Z}_{\geq 0}$	Menge der nicht-negativen, negativen ganzen Zahlen, d.h., $\mathbb{Z}_{\geq 0} = \{a \in \mathbb{Z} \mid a \geq 0\}$
$\mathbb{Z}_{<0}$	Menge der negativen, ganzen Zahlen, d.h., $\mathbb{Z}_{<0} = \{a \in \mathbb{Z} \mid a < 0\}$
$\mathbb{Z}_{\leq 0}$	Menge der nicht-positiven, ganzen Zahlen, d.h., $\mathbb{Z}_{\leq 0} = \{a \in \mathbb{Z} \mid a \leq 0\}$

Tabellenverzeichnis

2.3 Konventionen zur Bezeichnung von Gleichheiten / Äquivalenzen in aussage- und prädikatenlogischen Ausdrücken.	62
2.4 Konventionen zur Bezeichnung von Gleichheiten bei mengentheoretischen und algebraischen Ausdrücken.	63
4.1 Wie können die zweiten Elemente $y \in Y$ in den Paaren (x, y) einer Funktion $f : X \rightarrow Y$ prinzipiell auftreten? Der Fall von Funktionen zwischen endlichen Mengen.	87
4.2 Wie können die zweiten Elemente $y \in Y$ in den Paaren (x, y) einer Funktion $f : X \rightarrow Y$ prinzipiell auftreten? Der Fall von Funktionen zwischen unendlichen Mengen (hier: Teilmenge der reellen Zahlen).	88
10.2 Strukturelle Charakteristika eines endlichen Automaten.	229
14.1 Zusammenfassung der Sprachklassen mittels Grammatiken.	267

Abbildungsverzeichnis

2.1	Partitionierungen spielen in mehrerer Hinsicht bei (älteren) Laufwerken eine Rolle. (By User:Omegatron - Created by User:Omegatron using the GIMP, GPL, ByEvan-Amos-Ownwork, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=27940250)	56
2.2	Partitionierung der physischen Laufwerke in logische Laufwerke. (By User:Omegatron - Created by User:Omegatron using the GIMP, GPL, https://commons.wikimedia.org/w/index.php?curid=1930356)	57
2.3	Zwei verschiedene Partitionierungen der Menge aller Speichereinheiten eines Laufwerks. (By LionKimb - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=331092)	57
4.1	Grundbegriffe im Zusammenhang mit Funktionen.	82
4.2	Funktionsgraphen (Plot) von $f(x) := x^2$ im Bereich zwischen -3 und 3 des Arguments x	85
4.3	Funktionsgraphen (Plot) von $g(x) := \sin(x)$ im Bereich zwischen -9 und 9 des Arguments x	85
5.1	Homogene binäre Relation äquivalent als "gerichteter Graph" dargestellt.	102
5.2	Visualisierung der vier Grundeigenschaften homogener binärer Relationen.	104
5.3	Verknüpfung $S \circ R$ zweier kompatibler Relationen.	115
5.4	Das Ergebnis der Verknüpfung $S \circ R$ von R mit S	115
5.5	Die bipartiten Graphen der Verknüpfung $S \circ R$ für die konkreten R und S	117
5.6	Das einfache Prinzip einer inversen Relation R^{-1}	117
5.7	Das einfache Prinzip einer inversen Relation R^{-1}	118
5.8	Das einfache Prinzip einer inversen Relation R^{-1}	119
6.1	aus http://metrouk2.files.wordpress.com/2014/04/london1.jpg	129
6.2	Allgemeines Prinzip der mathematischen Modellierung.	141
6.3	Königsberg (russ. Kaliningrad), 1651.	143
6.4	Festland, Inseln und Fährverbindungen (in km).	151
14.1	Die Chomsky-Hierarchie der Sprachklassen.	267

Listings

Literaturverzeichnis

- [1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data structures and algorithms*. Addison-Wesley, [reprinted with corrections] edition, 1987.
- [2] M. Bóna, editor. *Handbook of Enumerative Combinatorics*. CRC Press, 2015.
- [3] S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine (The Long Version). Seventh International World-Wide Web Conference (WWW 1998), April 14-18, 1998, Brisbane, Australia., 1998. Online, <http://ilpubs.stanford.edu:8090/361/1/1998-8.pdf>.
- [4] H.-J. Böckenhauer and J. Hromkovic. *Formale Sprachen : endliche Automaten, Grammatiken, lexikalische und syntaktische Analyse*. Springer, 2013.
- [5] F. R. Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, 2007.
- [6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
- [7] J. R. Dias. *Molecular orbital calculations using chemical graph theory*. Springer textbook. Springer, 1993.
- [8] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.
- [9] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics*. Addison-Wesley, 2nd edition, 1994.
- [10] D. H. Greene and D. E. Knuth. *Mathematics for the analysis of algorithms*. Modern Birkhäuser classics. Birkhäuser, 3rd ed. rev. edition, 2007.
- [11] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2nd edition, 2001.
- [12] J. E. Hopcroft, J. D. Ullman, and A. V. Aho. *The design and analysis of computer algorithms*. Addison-Wesley series in computer science and information processing. Addison-Wesley, 1974.
- [13] D. C. Kozen. *The Design and Analysis of Algorithms*. Springer, 1992.
- [14] D. Krob, A. A. Mikhalev, and A. V. Mikhalev, editors. *Formal Power Series and Algebraic Combinatorics*. 12th International Conference, FPSAC'00, Moscow, Russia, June 2000, Proceedings. Springer, 2000.
- [15] A. Leiser. Von ungenauer Sprache in der Mathematik und nur vermeintlich erstaunlichen Phänomenen. Zusatzmaterial und Hintergründe zum FHNW-Modul "Mathematische Grundlagen der Informatik" (mgli), 2020. https://www.andreasleiser.ch/publicdocs/Von_ungenauer_Sprache_in_der_Mathematik_und_nur_vermeintlich_erstaunlichen_Phaenomenen.pdf.

- [16] T. Ottmann and P. Widmayer. *Algorithmen und Datenstrukturen*. Spektrum, 3rd edition, 1996.
- [17] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford InfoLab, 1999. Online, <http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf>.
- [18] R. Sedgewick. *Algorithmen*. Addison-Wesley, 1992.
- [19] D. Vrajitoru and W. Knight. *Practical Analysis of Algorithms*. Springer, 2014.
- [20] H. S. Wilf. *Generatingfunctionology*. Academic Press, 1994.

Bildverzeichnis

- [A1] Matt Britt. Internet map, 2006. By Matt Britt [CC BY 2.5 (<http://creativecommons.org/licenses/by/2.5>)], via Wikimedia Commons - https://upload.wikimedia.org/wikipedia/commons/c/c3/Internet_map_4096.png.
- [A2] Matt Groening. Simpsons family tree, 2017. By Matt Groening - The Simpsons Uncensored Family Album, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=56847200>.
- [A3] Merian-Erben. Königsberg (russ. kaliningrad), 1651, 1952. By Merian-Erben - http://www.preussen-chronik.de/_/bild_jsp/key=bild_kathe2.html, Public Domain, <https://commons.wikimedia.org/w/index.php?curid=1447648>.