Prof. Dr. D. Schweigert
Fachbereich Mathematik
Universität Kaiserslautern

D I S K R E T E M A T H E M A T I K

INHALTSVERZEICHNIS

I Kombinatorik der Mengen

Kapitel		Seite
1	Mengen, Ordnungsrelationen und Potenzmengen	1
2	Funktionen und Äquivalenzrelationen	6
3	Über die Axiomatik der Mengenlehre	13
4	Natürliche Zahlen, Induktion und Dedekind-Tripel	17
5	Primitiv rekursive Funktionen	30
6	Über die Kardinalität von Mengen	36
7	Permutationen und Kombinationen	41
8	Prinzip der Inklusion und Exklusion	50
9	Teilbarkeit	53
10	Primzahlen und Zahldarstellung	63
11	Kongruenzen	67
12	Über die Komplexität von Algorithmen	69
13	Erzeugung von Primzahlen	72
14	Rekurrenzrelationen	78
15	Lineare homogene Rekurrenzrelationen	82 83
16	Lineare inhomogene Rekurrenzrelationen	86 89
17	Rekurrenzrelationen für einige Sortieralgorithmen	96 97
18	Erzeugende Funktionen	103 104

II Kombinatorik der binären symmetrischen Relationen (Graphen)

Kapitel		Seite
19	Graphen	111 112
20	Wege, Zykel und kürzeste Wege	120 121
21	Aufspannende Bäume	126 127
22	Hamiltonsche Graphen	136 134
23	Gerichtete Graphen	141 142
24	Eulersche Graphen	147 148
25	Bäume mit Wurzeln	149 153
26	Gewichtete Bäume	165 167
27	Suchen auf Bäumen und Backtracking	169 171
28	Matchings	175 176
29	Netzwerke	181 182
30	Planare Graphen	190 192
31	Eckenfärbungen	198 200
32	Kantenfärbungen	203 205
33	Ramseyzahlen	209 211 – 217

Kapitel 1 : Mengen, Ordnungsrelationen und Potenzmengen

Unter einer MENGE verstehen wir die Zusammenfassung verschiedener Objekte. Diese Objekte werden als ELEMENTE bezeichnet und durch FORMELN beschrieben. Als Schreibweise benutzen wir:

$$M = \{ a \mid \varphi(a) \},$$

wobei $\varphi(a)$ die Elemente von M charakterisiert.

Häufig geben wir eine Menge auch an, indem wir alle Elemente auflisten, z.B.:

$$\mathbb{N} = \{1, 2, 3, \ldots\}$$

oder

$$\mathbb{N}_0 = \{0, 1, 2, \ldots\}.$$

Ist a ein Element von M, so schreiben wir $a \in M$.

Definition (1.1):

Eine Menge N heißt TEILMENGE der Menge M, $N \subseteq M$, wenn aus $a \in N$ folgt, daß $a \in M$ ist.

Definition (1.2):

Zwei Mengen heißen GLEICH, $N = M$, wenn $N \subseteq M$ und $M \subseteq N$ gilt.

Definition (1.3):

Die Menge $\emptyset = \{ x \mid x \neq x \}$ heißt LEERE MENGE.

Definition (1.4):

Seien A, B zwei Mengen. Die Menge

$$A \times B = \{ (a,b) \mid a \in A \text{ und } b \in B \}$$

heißt KARTESISCHES PRODUKT.

Die Elemente (a,b) von $A \times B$ sind geordnete Paare, wobei die erste Komponente ein Element aus A und die zweite Komponente ein Element aus B ist.

Definition (1.5):

Eine BINÄRE RELATION δ auf einer Menge A ist eine Teilmenge $\delta \in A \times A$. δ heißt ORDNUNGSRELATION, wenn

δ REFLEXIV ist, d.h.: $(a,a) \in \delta$ für alle $a \in A$,

δ ANTISYMMETRISCH ist, d.h.: aus $(a,b) \in \delta$ und $(b,a) \in \delta$ folgt $a=b$,

δ TRANSITIV ist, d.h.: aus $(a,b) \in \delta$ und $(b,c) \in \delta$ folgt $(a,c) \in \delta$.

Das Relationssystem (A,δ) heißt TEILWEISE GEORDNETE MENGE.

Beispiel:
Bezeichne $\delta \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ die Teilmenge aller Punkte (a,b), die oberhalb der Diagonalen oder auf der Diagonalen im kartesischen Koordinatensystem liegen. Dann ist offensichtlich δ eine Ordnungsrelation, wobei man statt $(a,b) \in \delta$ in üblicher Weise $a \leqslant b$ schreibt.

Definition (1.6):

Die Menge P(M) aller Untermengen der Menge M heißt POTENZMENGE von M.

Bemerkung (1.7):

Die binäre Relation $\subseteq$ auf P(M) ist eine Ordnungsrelation, oder anders ausgedrückt: $(P(M);\subseteq)$ ist eine teilweise geordnete Menge.

Beweis:
Die Reflexivität und die Transitivität folgen sofort aus Definition (1.1), die Antisymmetrie aus Definition (1.2).

Man führt folgende Operationen auf Mengen ein:

Definition (1.8):

Die Menge $N \cap M = \{ a \mid a \in N \text{ und } a \in M \}$ heißt der DURCHSCHNITT von N und M.
Die Menge $N \cup M = \{ a \mid a \in N \text{ oder } a \in M \}$ heißt die VEREINIGUNG von N und M.
Die Menge $M - N = \{ a \mid a \in M \text{ und } a \notin N \}$ heißt die DIFFERENZ von N in M.
Ist $M \in N$, so heißt $C_M^N = M - N$ auch das KOMPLEMENT von N in M.

Bemerkung (1.9):

Eine ALGEBRA A = (M, Φ) ist ein geordnetes Paar, bestehend aus einer nicht leeren Menge M und einer Menge Φ von Operationen, so daß gewisse 'Verträglichkeitsbedingungen' erfüllt sind. Erlaubt sind auch nullstellige Operationen, die in der Auszeichnung einzelner Elemente bestehen.
Die Menge B = (P(M); ∩, ∪, ', ∅, M) ist eine BOOLESCHE ALGEBRA, denn B erfüllt folgende, vom Leser nachzuweisende Gesetze:

1. ASSOZIATIVGESETZ:
   $A \cap (B \cap C) = (A \cap B) \cap C$ und $A \cup (B \cup C) = (A \cup B) \cup C$

2. KOMMUTATIVGESETZ:
   $A \cap B = B \cap A$ und $A \cup B = B \cup A$

3. IDEMPOTENZGESETZ:
   $A \cap A = A$ und $A \cup A = A$

4. ADSORPTIONSGESETZ:
   $A \cap (B \cup A) = A$ und $A \cup (B \cap A) = A$

5. DISTRIBUTIVGESETZ:
   $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
   und $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

6. DE-MORGAN-GESETZE:
   $(A \cap B)' = A' \cup B'$ und $(A \cup B)' = A' \cap B'$

7. INVOLUTIONSGESETZ:
   $A'' = A$

8. AUSGEZEICHNETE ELEMENTE: Sei M das größte Element, ∅ das kleinste Element. Dann gilt: $M \cap A = A$ und $\emptyset \cup A = A$

Wir ergänzen nun unser Wissen über Ordnungsrelationen mit folgenden Begriffen:

Definition (1.10):

Sei $(A;\leqslant)$ eine teilweise geordnete Menge. Für eine Teilmenge C von A heißt das Element m ein MAXIMALES ELEMENT von C, wenn gilt:

1. $m \in C$
2. Ist $a \in C$ und $m \leqslant a$, dann gilt $m=a$.

Man definiert dual, was ein MINIMALES ELEMENT von C ist.

Definition (1.11):

Sei $(A;\leqslant)$ eine teilweise geordnete Menge und C eine Teilmenge von A. Ein Element b heißt OBERE SCHRANKE von C, wenn für alle $c \in C$ gilt:

$$c \leqslant b.$$

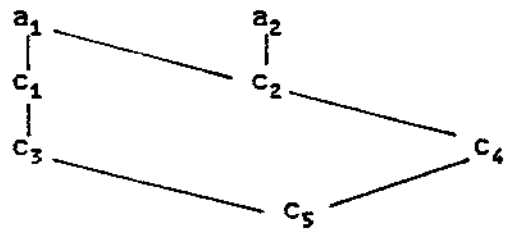
Ein Element s heißt SUPREMUM (KLEINSTE OBERE SCHRANKE) von C, wenn gilt:

1. s ist obere Schranke von C
2. Ist b eine obere Schranke von C, dann ist $s \leqslant b$.

Man definiert dual die Begriffe UNTERE SCHRANKE, INFIMUM, ~~KLEINSTE~~ GRÖßTE UNTERE SCHRANKE.

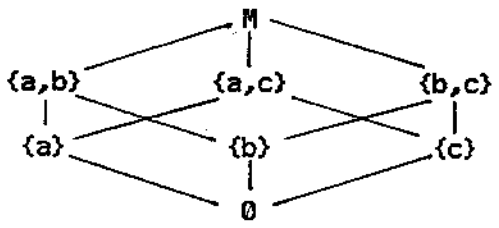
Den Sachverhalt kann man sich leicht mit Hilfe eines HASSEDIAGRAMMS veranschaulichen. Dabei wird eine 'kleine' teilweise geordnete Menge $(A;<)$ veranschaulicht, indem man $a \leqslant b$ mit Hilfe einer von unten nach oben laufenden Linie zwischen den Punkten a, b darstellt.

Beispiel:
$A = \{ a_1, a_2, c_1, \ldots, c_5 \}$
$C = \{ c_1, \ldots, c_5 \}$



C besitzt zwei obere Schranken a_1 und a_2, und zwei maximale Elemente c_1 und c_2. C hat aber kein Supremum (da die oberen Schranken nicht vergleichbar sind). C hat ein Infimum, welches zugleich auch minimales Element ist, nämlich c_5.

Beispiel:
Das Hassediagramm der teilweise geordneten Menge (P(M); $\subseteq$) für M = { a,b,c }:



Lemma (1.12):

Sei (P; $\leqslant$) eine teilweise geordnete Menge und C $\subseteq$ P. Besitzt C ein Supremum (bzw. ein Infimum) in P, dann ist das Supremum (bzw. das Infimum) eindeutig bestimmt.

Beweis:
Seien s,t zwei Suprema von C. Dann sind sowohl s als auch t obere Schranken von C. Da s kleinste obere Schranke ist, gilt s $\leqslant$ t. Da t kleinste obere Schranke ist, gilt t $\leqslant$ s. Es folgt s=t.

Lemma (1.13):

Zu je zwei Elementen A,B der teilweise geordneten Menge (P(M); $\subseteq$) gibt es ein Supremum sup {A,B} und ein Infimum inf {A,B}. Dabei gilt: sup {A,B} = A $\cup$ B und inf {A,B} = A $\cap$ B.

Beweis:
Offensichtlich gilt A $\subseteq$ A $\cup$ B und B $\subseteq$ A $\cup$ B. Also ist A $\cup$ B eine obere Schranke von A und von B. Sei D eine andere obere Schranke von A und von B. Dann gilt A $\subseteq$ D und B $\subseteq$ D, also A $\cup$ B $\subseteq$ D. Also ist A $\cup$ B die kleinste obere Schranke.

Kapitel 2 : Funktionen und Äquivalenzrelationen

Bemerkung (2.1):

Man kann eine Funktion $f: A \longrightarrow B$ als binäre Relation $\delta \subseteq A \times B$ auffassen. Es ist nämlich:

$$\delta = \{ (a,f(a)) \mid a \in A \}.$$

Dabei hat die Relation δ folgende Eigenschaften:

1. Für jedes $a \in A$ gibt es ein $b \in B$ mit $(a,b) \in \delta$.
2. Ist $(a,b) \in \delta$ und $(a,c) \in \delta$, dann ist $b = c$.

Umgekehrt ist jede Relation δ mit diesen Eigenschaften eine Funktion.

Definition (2.2):

Sei $f: A \longrightarrow B$ eine Funktion und seien $M \subseteq A$ und $N \subseteq B$ Teilmengen. Dann heißt

$$f[M] = \{ b \mid b \in B, \text{ es gibt } a \in M \text{ mit } f(a) = b \}$$

das BILD von M, und

$$f^{-1}(N) = \{ a \mid a \in A, f(a) \in N \}$$

das URBILD von N.

Definition (2.3):

Eine Funktion $f: A \longrightarrow B$ heißt SURJEKTIV, wenn $f[A] = B$ gilt, d.h. wenn es zu jedem Element $b \in B$ ein Element $a \in A$ gibt mit $f(a) = b$.

Definition (2.4):

Eine Funktion $f: A \longrightarrow B$ heißt INJEKTIV, wenn für alle $c,d \in A$ gilt:

Aus $f(c) = f(d)$ folgt $c = d$.

Definition (2.5):

Eine Funktion $f: A \longrightarrow B$ heißt BIJEKTIV, wenn f surjektiv und injektiv ist.

Beispiel:
Betrachte die Menge $\mathbb{Q}$ der rationalen Zahlen und die Funktion

$$f: \mathbb{Q} \longrightarrow \mathbb{Q},\ f(x) := 2x.$$

Dann ist f surjektiv, denn für jedes $b \in \mathbb{Q}$ gibt es ein $a \in \mathbb{Q}$ mit $b = 2a$ und damit $f(a) = b$.
Andererseits folgt aus $f(c) = f(d)$, also $2c = 2d$, daß $c = d$ gilt.
Also ist f auch injektiv und damit bijektiv.

Bezeichnung:
Mit $id_A: A \longrightarrow A$ bezeichnen wir die IDENTISCHE FUNKTION (IDENTITÄT) auf A:

$$id_A(x) = x \quad \text{für alle } x \in A.$$

Satz (2.6):

Sei $f: A \longrightarrow B$ eine Funktion.
Dann gilt:

i) f injektiv $\Leftrightarrow$ es gibt eine Funktion $g: B \longrightarrow A$ mit $g\,f = id_A$

ii) f surjektiv $\Leftrightarrow$ es gibt eine Funktion $g: B \longrightarrow A$ mit $f\,g = id_B$

iii) f bijektiv $\Leftrightarrow$ es gibt eine Funktion $g: B \longrightarrow A$ mit

$$g\,f = id_A \text{ und } f\,g = id_B.$$

Beweis:
i) '=>':
Da f injektiv ist, gibt es für $b \in f[A]$ genau ein $a \in A$ mit $f(a) = b$.
Definiere also

$$g: B \longrightarrow A$$
$$g(b) := a \quad \text{für } b \in f[A]$$

und

$$g(b) := a_0 \quad \text{für } b \in B - f[A],$$

wobei a_0 beliebig aus A gewählt ist.
Damit ist g eine Funktion von B nach A mit

$$(g\,f)(a) = g(f(a)) = g(b) = a,$$

also

$$g\,f = id_A.$$

'<=':

Ist g: B ⟶ A mit g f = id_A gegeben, und ist

$$f(c) = f(d) \text{ für } c,d \in A,$$

so ist

$$c = id_A(c) = g(f(c)) = g(f(d)) = id_A(d) = d,$$

also ist f injektiv.

ii) ‚=>':

Sei f injektiv.
Zu jedem b ∈ B wählen wir ein festes a ∈ A mit f(a) = b und setzen

$$g(b) := a.$$

Für die so definierte Abbildung g: B ⟶ A gilt

$$(f\ g)(b) = f(g(b)) = f(a) = b,$$

also

$$f\ g = id_B.$$

'<=':

Ist g: B ⟶ A mit f g = id_A gegeben und ist b ∈ B, so ist

$$b = f(g(b)),\ g(b) \in A,$$

also b ∈ f[A], und f ist surjektiv.

iii):

Vorbemerkung: Ist f: A ⟶ B bijektiv, so besteht für jedes b ∈ B das Urbild $f^{-1}(\{b\})$ aus genau einem Element, das wir mit $f^{-1}(b)$ bezeichnen. So erhält man eine Abbildung

$$f^{-1}: B \longrightarrow A,\ b \longrightarrow f^{-1}(b),$$

die wir UMKEHRABBILDUNG von f nennen.

'=>':

Ist f bijektiv, so erfüllt $g:=f^{-1}$ die beiden Gleichungen.

'<=':

Ist g: B ⟶ A mit g f = id_A und f g = id_B gegeben, so ist f nach i) und ii) injektiv und surjektiv, also bijektiv.

Definition (2.7):

Eine binäre Relation $\delta \subseteq A \times A$ heißt **ÄQUIVALENZRELATION** auf A, wenn gilt:

1. δ ist REFLEXIV : $(a,a) \in \delta$ für alle $a \in A$
2. δ ist SYMMETRISCH : $(a,b) \in \delta \Rightarrow (b,a) \in \delta$
3. δ ist TRANSITIV : $(a,b),(c,d) \in \delta \Rightarrow (a,c) \in \delta$.

Beispiele:

a) '=' ist eine Äquivalenzrelation (bei beliebiger Menge A). Sie heißt auch IDENTITÄT oder DIAGONALE, denn bei der Grundmenge $A = \mathbb{R}$ ist (im Koordinatenkreuz $\mathbb{R} \times \mathbb{R}$)

$(r_1,r_2) \in \delta \iff (r_1,r_2)$ liegt auf der Diagonalen.

b) Die Allrelation $A \times A$ ist eine Äquivalenzrelation.

c) Betrachte die Menge $\mathbb{Z}$ und definiere für ein festes $a \in \mathbb{Z}$

$$\delta_a \subseteq \mathbb{Z} \times \mathbb{Z}$$

durch

$(c,d) \in \delta_a \iff c-d$ ist durch a teilbar ($c-d \mid a$).

δ_a ist eine Äquivalenzrelation, denn:

- $(z,z) \in \delta_a$, denn $z-z=0$ und $0 \mid a$
- $(z_1,z_2) \in \delta_a \Rightarrow (z_1-z_2) \mid a$
  $\Rightarrow (z_2-z_1) \mid a$
  $\Rightarrow (z_2,z_1) \in \delta_a$
- $(z_1,z_2), (z_2,z_3) \in \delta_a$
  $\Rightarrow z_1-z_2 \mid a,\ z_2-z_3 \mid a$
  $\Rightarrow (z_1-z_2) + (z_2-z_3) \mid a$
  $\Rightarrow (z_1-z_3) \mid a$
  $\Rightarrow (z_1,z_3) \in \delta_a$.

Lemma (2.8):

Ist $f: A \longrightarrow B$ eine Funktion, so ist δ_f, definiert durch

$$(c,d) \in \delta_f \iff f(c) = f(d)$$

eine Äquivalenzrelation auf A.
Man nennt sie die (zu f) ZUGEHÖRIGE ÄQUIVALENZRELATION.

Beweis:
i) $(a,a) \in \delta_f$, da $f(a) = f(a)$.
ii) $(a,b) \in \delta_f \Rightarrow f(a) = f(b) \Rightarrow f(b) = f(a) \Rightarrow (b,a) \in \delta_f$.
iii) analog zu ii).

Definition (2.9):

Sei δ eine Äquivalenzrelation auf A, $c \in A$ fest gewählt. Dann heißt die Menge

$$[c] := \{ x \mid (c,x) \in \delta, x \in A \}$$

die ÄQUIVALENZKLASSE von c.

Beispiel:
Für die Äquivalenzrelation δ_3 auf $\mathbb{Z}$ und a = 0, 1, 2, 3 ergeben sich die Äquivalenzklassen

$$[0] = \{ \ldots -8, -4, 0, 4, 8, \ldots \}$$
$$[1] = \{ \ldots -7, -3, 1, 5, 9, \ldots \}$$
$$[2] = \{ \ldots -6, -2, 2, 6, 10, \ldots \}$$
$$[3] = \{ \ldots -5, -1, 3, 7, 11, \ldots \}.$$

Lemma (2.10):

Sei δ eine Äquivalenzrelation auf A. Dann sind zwei Äquivalenzklassen entweder gleich oder DISJUNKT (d.h. $[c] \cap [d] = \emptyset$).

Beweis:
Seien [c], [d] Äquivalenzklassen mit $[c] \cap [d] \neq \emptyset$. Dann gibt es ein $b \in A$ mit

$$b \in [c] \text{ und } b \in [d].$$

Sei $e \in [c]$, d.h. $(c,e) \in \delta \Rightarrow (e,c) \in \delta$. Da $b \in [c]$, ist $(c,b) \in \delta$. Wegen der Transitivität ist damit $(e,b) \in \delta$. Aus $b \in [d]$ folgt $(d,b) \in \delta$, also $(b,d) \in \delta$. Da δ transitiv ist, folgt $(e,d) \in \delta$, also $(d,e) \in \delta$. Also ist $e \in [d]$.

Analog zeigt man $[d] \subseteq [c]$ und hat damit $[c] = [d]$.

Bemerkung (2.11):
Es gilt also: c ∈ [d] <=> [c] = [d].

Im obigen Beispiel waren alle Äquivalenzklassen disjunkt.

Definition (2.12):

Die Menge

$$A/\delta = \{ [a] \mid a \in A \}$$

aller Äquivalenzklassen heißt FAKTORMENGE (QUOTIENTENMENGE).

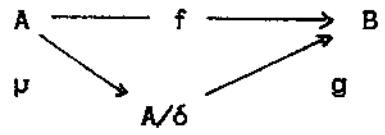
Satz (2.13):

Sei f : A --> B eine Funktion und δ die zugehörige Äquivalenzrelation. Dann gibt es Funktionen

$$\mu: A \longrightarrow A/\delta \quad \text{und} \quad g: A/\delta \longrightarrow B,$$

so daß folgende FAKTORISIERUNG gilt:

$$f = g\,\mu.$$



Beweis:
Definiere μ(x):=[x] und g([x]):=f(x).
μ ist eine Funktion, denn (für a, b ∈ A):

$$a = b \Rightarrow f(a) = f(b) \Rightarrow [a] = [b] \Rightarrow \mu(a) = \mu(b).$$

μ heißt auch die KANONISCHE ABBILDUNG (in die Faktormenge).
g ist eine Funktion, denn (für x, y ∈ A):

$$\begin{aligned} [x] = [y] &\Rightarrow x \in [y] && \text{, nach Bemerkung (2.11)} \\ &\Rightarrow (x,y) \in \delta \\ &\Rightarrow f(x) = f(y) && \text{, nach Definition (2.8).} \end{aligned}$$

Es gilt

$$(g\,\mu)(x) = g(\mu(x)) = g([x]) = f(x),$$

also

$$g\,\mu = f.$$

Satz (2.14):

Sei $f: A \longrightarrow B$ surjektiv und δ zugehörige Äquivalenzrelation.
Dann gibt es eine bijektive Abbildung $g: A/\delta \longrightarrow B$.

Beweis:
Definiere $g([x]) := f(x)$.

i) 'g ist injektiv'

$g([c]) = g([d]) \Rightarrow (g \mu)(c) = (g \mu)(d)$,nach (2.12)
$\Rightarrow f(c) = f(d)$,da $g \mu = f$
$\Rightarrow c \in [d]$,nach (2.8)
$\Rightarrow [c] = [d]$,nach (2.11)

ii) 'g ist surjektiv'
Sei $b \in B$, dann gibt es $a \in A$ mit $f(a) = b$, da f surjektiv.
Also gibt es $[a] \in A/\delta$ mit

$$g([a]) = f(a) = b.$$

g ist also bijektiv.

Beispiel:
Betrachte die Division mit Rest auf $\mathbb{Z}$.
Für $z \in \mathbb{Z}$ sei

$$z : 4 = q + r(x)/4,$$

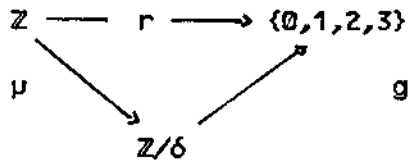
wobei $q \in \mathbb{Z}$, und $r(x) \in \mathbb{Z}$ den Rest bei der Division bedeute, also $r(x) \in \{ 0, 1, 2, 3 \}$. Für die hierdurch gegebene Abbildung

$$r : \mathbb{Z} \longrightarrow \{ 0, 1, 2, 3 \}$$

ist die zugehörige Äquivalenzrelation gegeben durch

$$(a,b) \in \delta \iff r(a) = r(b).$$

Nun gilt aber: $r(a) = r(b) \iff a-b$ durch 4 teilbar, wir können die nach Satz (2.13) mögliche Faktorisierung also wie folgt angeben:



mit $\mathbb{Z}/\delta = \{ [0], [1], [2], [3] \}$.
Da r surjektiv ist, ist g nach (2.13) sogar bijektiv.

Kapitel 3 : Über die Axiomatik der Mengenlehre

Die Frage, wie der Begriff MENGE mathematisch präzise eingeführt werden kann, gehört zu den noch nicht befriedigend gelösten Problemen der mathematischen Grundlagenforschung. In diesem Kapitel wird die ZERMELO-FRAENKELSCHE MENGENLEHRE eingeführt.

Wie wir gesehen haben, können Mengen Elemente von anderen Mengen sein. So sind zum Beispiel bei der Potenzmenge die Elemente Untermengen.
Nehmen wir an, die 'Menge'

$$\{ x \mid x \text{ hat Eigenschaft } E \}$$

sei bildbar. Betrachten wir dann speziell die Eigenschaft '$x \notin x$', so erhalten wir mit der obigen Bildungsvorschrift die 'Menge'

$$A := \{ x \mid x \notin x \}.$$

Insbesondere folgt dann der Widerspruch:

$$a \in A \iff a \notin A.$$

Diese von Bertrand RUSSELL 1901 entdeckte und nach ihm benannte Antinomie muß also bei der Definition des Mengenbegriffs ausgeschlossen werden. Es ergibt sich die Notwendigkeit, die durch eine gemeinsame Eigenschaft E zu einer Menge zusammenzufassenden Elemente aus einer bereits definierten Grundmenge G auszusonderen, also:

$$\{ x \in G \mid x \text{ hat Eigenschaft } E \}$$

ist eine Menge.
Als Menge bildbar ist jedoch ohne Angabe einer Grundmenge:

$$\{ x \mid x \notin x \}.$$

Die Begründung liegt in der Tatsache, daß die Eigenschaft '$x \notin x$' für beliebige Elemente beliebiger Mengen stets nicht erfüllt ist, also stets gilt:

$$\{ x \mid x \notin x \} = \emptyset.$$

Wir geben nun einen axiomatischen Aufbau, der widerspruchsfrei ist, die sogenannte ZERMELO-FRAENKELSCHE MENGENLEHRE.

Wir benutzen dabei nur eine informelle Sprache; wir gebrauchen die nicht weiter definierten Begriffe 'Menge' und 'Klasse'. Alle wichtigen Eigenschaften von Mengen und Klassen werden durch die Axiome festgelegt.

Wir vereinbaren, daß Klassen in der Form

$$\{ x \mid \varphi(x) \}$$

geschrieben werden.
Es sind

$a, b, c, \ldots, A, B, C$ Symbole für Mengen

und

A^*, B^*, C^* Symbole für Klassen.

Es werden folgende Klassen eingeführt:

$\emptyset = \{ x \mid x \neq x \}$, die LEERE KLASSE

$V^* = \{ x \mid x = x \}$, die UNIVERSELLE KLASSE
$\{a,b\} = \{ c \mid c = a \text{ oder } c = b \}$, UNGEORDNETES PAAR von a, b
$\{a\} = \{ a, a \}$, EINELEMENTIGE KLASSE von a.

Ferner wird definiert:

$$\bigcup A^* = \{ a \mid \text{es gibt ein } b \text{ mit } b \in A^* \text{ und } a \in b \}$$

$$\bigcap A^* = \{ a \mid \text{für alle } b \in A^* \text{ folgt } a \in b \}$$

und

$$A^* \subseteq B^* :\Leftrightarrow (a \in A^* \Rightarrow a \in B^*)$$

$$A^* \cup B^* = \{ a \mid a \in A^* \text{ oder } a \in B^* \}$$

$$A^* \cap B^* = \{ a \mid a \in A^* \text{ und } b \in B^* \}$$

$$A^* - B^* = \{ a \mid a \in A^* \text{ und } a \notin B^* \}$$

Definition (3.1):

1. $P(A^*) := \{ a \mid a \subseteq A^* \}$ heißt die POTENZKLASSE von A^*.

2. Die Menge $\{ \{a\}, \{a,b\} \}$ heißt GEORDNETES PAAR (a, b) von Mengen a und b.

3. $A^* \times B^* = \{ (a, b) \mid a \in A^*, b \in B^* \}$ heißt KARTESISCHES PRODUKT.

4. Eine BINÄRE RELATION zwischen A^* und B^* ist eine Unterklasse von $A^* \times B^*$.

5. Eine FUNKTION $f: A^* \longrightarrow B^*$ ist eine binäre Relation

$\delta \subseteq A^* \times B^*$ mit

i) Für alle $a \in A^*$ gibt es ein $b \in B^*$ mit $(a,b) \in \delta$.
ii) Ist $(a, b), (a, c) \in \delta$, so ist $b = c$.

6. Die Klasse $f[A^*]$ wird definiert durch $f[A^*] = \{ f(a) \mid a \in A^* \}$.

Wir legen nun folgende Axiome fest:

1. EXTENSIONALITÄT:

Zwei Mengen sind gleich, wenn sie dieselben Elemente besitzen.

2. EXISTENZ DER LEEREN MENGE:

Die leere Menge ist eine Menge.

3. UNGEORDNETE PAARE:

Sind a, b Mengen, so ist auch { a, b } eine Menge.

4. VEREINIGUNG

Ist a eine Menge, so ist $\bigcup a$ eine Menge.

5. EINSCHRÄNKUNG:

Ist a eine Menge und A^* eine Klasse, so ist $a \cap A^*$ eine Menge.

6. POTENZMENGE:

Ist a eine Menge, so ist P(a) eine Menge.

7. BILDER VON FUNKTIONEN:

Ist f eine Funktion und A eine Menge, so ist f[A] eine Menge.

8. UNENDLICHKEIT:

Die Klasse

$$\{ a \mid b \in a \text{ und für alle } b\text{: } (b \in a \Rightarrow b \cup \{b\} \in a) \}$$

ist nicht leer.

Nur die Axiome 2 und 8 geben die Existenz von Mengen an; nach Axiom 8 gibt es unendliche Mengen.

In der Mathematik wird außerdem noch das Auswahlaxiom sehr oft benutzt. Wir wollen zwei seiner (vielen) Varianten hier darstellen; dazu definieren wir das KARTESISCHE PRODUKT EINER MENGE S VON MENGEN als

$$\prod_{a \in S} a = \{ f: S \longrightarrow \bigcup S \mid f(a) \in a,\ a \in S \}.$$

Satz (3.2): [Auswahlaxiom, 1. Variante]

Das kartesische Produkt von einer nicht leeren Menge von nicht leeren Mengen ist nicht leer.

Satz (3.3): [Auswahlaxiom, 2. Variante]

Ist eine Menge M gegeben, so existiert eine Funktion φ, die jeder nichtleeren Untermenge a von M ein bestimmtes Element $\varphi(a)$ aus a zuordnet. Man kann also aus jeder Untermenge a, $a \neq \emptyset$, einer Menge M ein Element 'auswählen'.

Bevor wir dieses Kapitel abschließen, wollen wir noch angeben, wie man aus der Mengenlehre die natürlichen Zahlen einschließlich der Null einführen kann:

Wir schreiben

$$A+ = A \cup \{ A \}.$$

Für $A = \{ a, b \}$ hat man zum Beispiel

$$A+ = \{ a, b \} \cup \{ \{ a, b \} \} = \{ a, b, \{ a, b \} \}.$$

Wir definieren nun:

$$0 = \emptyset$$
$$1 = 0+ = \emptyset \cup \{\emptyset\} = \{\emptyset\}$$
$$2 = 1+ = \{\emptyset\} \cup \{ \{\emptyset\} \} = \{ \emptyset, \{\emptyset\} \}$$
$$3 = 2+ = \{ \emptyset, \{ \emptyset \}, \{ \emptyset, \{\emptyset\} \} \}$$
usw.

Im folgenden Kapitel werden wir die natürlichen Zahlen durch die Axiome von PEANO einführen. Aus den Axiomen 1 - 8 kann bewiesen werden, daß die oben konstruierten natürlichen Zahlen diese Peano-Axiome erfüllen.

Kapitel 4 : Natürliche Zahlen, Induktion und Dedekind-Tripel

4A : Natürliche Zahlen und Induktion

Wir gehen davon aus, daß uns die natürlichen Zahlen durch Zählvorgänge gegeben sind (' Die natürlichen Zahlen hat der liebe Gott gemacht', Kronecker). Die Menge $\mathbb{N}$ der natürlichen Zahlen kann man durch die folgenden Axiome von PEANO beschreiben:

1. 1 ist eine natürliche Zahl.

2. Jede natürliche Zahl n besitzt genau einen Nachfolger n^+ in der Menge der natürlichen Zahlen.

3. Für alle natürlichen Zahlen n gilt $n^+ \neq 1$, d.h. 1 ist kein Nachfolger.

4. $n^+ = m^+ \Rightarrow n = m$.

5. Jede Menge von natürlichen Zahlen, welche die 1 enthält und welche zu jeder natürlichen Zahl n auch deren Nachfolger n^+ enthält, enthält alle natürlichen Zahlen.

Auf der Eigenschaft 5.) beruht das Beweisprinzip der VOLLSTÄNDIGEN INDUKTION:

Sei $\varphi(n)$ eine Aussage, die die natürliche Zahl n enthält.

Induktionsanfang: $\varphi(n)$ ist wahr für $n = n_0$ (meist $n_0 = 1$).

Induktionsvoraussetzung: $\varphi(n)$ ist wahr für $n = k$, $k \geq n_0$.

Induktionsbehauptung: $\varphi(k^+)$ ist wahr.

Induktionsschritt: Zeige die Induktionsbehauptung mit Hilfe der Induktionsvoraussetzung.

Dann gilt: $\varphi(n)$ ist wahr für alle natürlichen Zahlen n, $n \geq n_0$.

Als Beispiel beweisen wir folgendes Lemma:

Lemma (4.1):

Ist $n \in \mathbb{N}$, $n \neq 1$, so gibt es einen Vorgänger $u \in \mathbb{N}$ mit $u^+ = n$.

Beweis:
Induktionsanfang:

$n = n_0 = 1^+ = 2$. Dann ist $u = 1 \in \mathbb{N}$ mit $u^+ = n$.

Induktionsvoraussetzung:

Die Behauptung gelte für $k \in \mathbb{N}$, $k \geqslant n_0$ d.h. zu k gibt es $u \in \mathbb{N}$ mit $u^+ = k$.

Induktionsbehauptung:

Die Behauptung gilt für k^+, d.h.: es gibt u mit $u^+ = k^+$.

Induktionsschritt:

Setze $u:=k$, damit ist $u^+ = k^+$.

Das obige Schema des Induktionsbeweises kann man noch nach zwei Seiten ergänzen. Dazu benötigen wir, daß man die natürlichen Zahlen linear anordnen kann. Der erste Schritt dazu ist die Einführung einer Addition.

Bemerkung (4.2): [Definition einer Funktion durch REKURSION]

Um jedem $n \in \mathbb{N}$ ein neues Objekt $\varphi(n)$ zuzuordnen, gibt man ein System von Bedingungen vor, die den Funktionswert $\varphi(n)$ jeweils mit den vorangehenden Werten $\varphi(m)$, $m \leqslant n$, verknüpfen.

Beispiel (4.3):
Jedem Paar $(n,m) \in \mathbb{N} \times \mathbb{N}$ wird eine natürliche Zahl $n + m$ zugeordnet, wobei man rekursiv definiert:

$$n + 1 = n^+$$

$$n + m^+ = (n + m)^+.$$

Durch diese Bedingungen ist die Addition vollständig auf $\mathbb{N}$ erklärt und eindeutig bestimmt.

Lemma (4.4):

Für zwei natürliche Zahlen x, y hat man genau einen der drei Fälle:

1) $x = y$.

2) $x = y + u$, $u \in \mathbb{N}$.

3) $y = x + v$, $v \in \mathbb{N}$.

Beweis:
Wir zeigen zunächst durch Induktion, daß $y \neq x+y$ für alle $x, y \in \mathbb{N}$.

Induktionsanfang:

$y = 1$. Es gilt $1 \neq x + 1$ wegen Peanoaxiom 3 .

Induktionsvoraussetzung:

$k \neq x + k$, $k \geq 1$.

Induktionsschritt:

Annahme: $k^+ = x + k^+$

$\Rightarrow k^+ = (x + k)^+$, nach Definition der Addition
$\Rightarrow k = x + k$,nach Peanoaxiom 4,

im Widerspruch zur Induktionsvoraussetzung.
Daraus folgt nun, daß Fall 1 und Fall 2 nicht gleichzeitig auftreten können, denn

$$x = y \text{ und } x = y + u \Rightarrow x = x + u,$$

was nach dem gerade Bewiesenen nicht sein kann.
Mit derselben Argumentation folgt, daß auch Fall 1 und 3 sowie Fall 2 und Fall 3 nicht gleichzeitig auftreten können. Man hat also höchstens einen der drei Fälle.

Wähle nun x fest.

Induktionsanfang:

$y = 1 : x \neq y \Rightarrow x = u^+$, nach (4.1)

$\Rightarrow x = u^+ = 1 + u = y + u,$
also Fall 2.

Im Fall $x = y = 1$ haben wir Fall 1.

Induktionsvoraussetzung:

Mindestens einer der 3 Fälle gilt für y.

Induktionsbehauptung:

Mindestens einer der 3 Fälle gilt für y^+.

Induktionsschritt:

Sei $y^+ = k^+$.
Für k gilt entweder

$x = k \quad \Rightarrow \quad y^+ = x + 1$, Fall 3

oder $k = x + u \Rightarrow y^+ = k^+ = x + (u+1)$, Fall 3
oder $x = k + v \Rightarrow$

für $v \neq 1 : x = k^+ + u$, mit $u^+ = v$, Fall 2

für $v = 1 : x = k + 1 = k^+ = y^+$, $y^+ = x$, Fall 1.

Also gilt mindestens und höchstens ein Fall, d.h. genau einer.

Satz (4.5):

Auf $\mathbb{N}$ gibt es eine Ordnungsrelation '<', so daß für $x, y \in \mathbb{N}$ gilt:

Entweder $x < y$ oder $x = y$ oder $y < x$.

Beweis:
Definiere

$$x < y \quad :\Leftrightarrow \quad y = x + u,\ u \in \mathbb{N}.$$

Der Beweis folgt sofort aus Lemma (4.4).

Das Prinzip der vollständigen Induktion gibt es in verschiedenen Formen. Zwei Varianten sollen noch kurz vorgestellt werden:
Ändert man in Form I die Induktionsvoraussetzung in

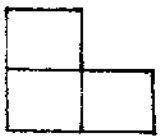
'$\varphi(n)$ ist wahr für alle n mit $n_0 \leqslant n \leqslant k$',

so erhält man Form II.
Ferner überträgt sich das Prinzip natürlich auch auf $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$; man läßt die Induktionsbeweise dann mit $n_0 = 0$ anfangen.

Zur Illustration geben wir ein Beispiel für eine Induktion der Form II auf $\mathbb{N}_0$:

Gegeben sei ein Schachbrett, auf dem ein Feld fehlt. Kann man die 63 Felder (8 x 8 -1 = 63) mit Triminoes ohne Überlappungen

überdecken? Dabei hat ein TRIMINOE folgende Form:



Behauptung: Jedes Schachbrett mit $2^n \times 2^n - 1$ Feldern kann mit Triminoes überdeckt werden.

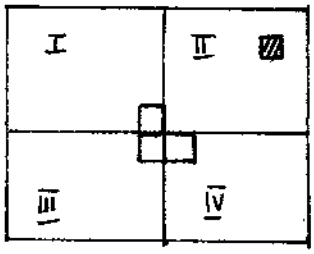
Induktionsbeweis:
Induktionsanfang: $n = 1$.
Ein $2 \times 2 - 1$ Schachbrett kann auf vier verschiedene Weisen mit einem Triminoe überdeckt werden:

Induktionsvoraussetzung: Jedes $(2^n \times 2^n - 1)$ - Schachbrett läßt sich für $1 \leq n \leq k$, k fest, überdecken.

Induktionsbehauptung: Ein $(2^{k+1} \times 2^{k+1} - 1)$ - Schachbrett läßt sich überdecken.

Induktionsschritt:

Sei ein $(2^{k+1} \times 2^{k+1} - 1)$ - Schachbrett gegeben. Teilt man es in 4 Quadranten auf, so fehlt in einem Quadranten, sagen wir in II, ein Feld. II ist ein $(2^{k-1} \times 2^{k-1} - 1)$ - Schachbrett, also können wir die Induktionsvoraussetzung anwenden und II überdecken. Plaziere nun ein Triminoe so im Zentrum, daß auch in I, III und IV ein Feld fehlt. Nach Voraussetzung können auch I, III und IV, und somit das ganze Schachbrett überdeckt werden.



Die Form III des Induktionsbeweises beruht auf folgendem Satz:

Satz (4.6):

Jede nicht-leere Teilmenge der natürlichen Zahlen besitzt ein kleinstes Element.

Beweis:
Sei $A \subseteq \mathbb{N}$ und $A \neq \emptyset$. Dann gibt es ein Element $a \in A$ und wir betrachten $[1,a] \cap A$, wobei

$$[1,a] = \{ n \mid 1 \leq n \leq a,\ n \in \mathbb{N} \}.$$

Diese endliche Teilmenge von $\mathbb{N}$ hat ein kleinstes Element.

Prinzip der vollständigen Induktion, Form III:
Sei $\varphi(n)$ eine Aussage, die n enthält. Man beweist, daß es keine kleinste Zahl k gibt, für die die Aussage $\varphi(k)$ falsch ist.

4B: Dedekind-Tripel

Die wichtigsten Werkzeuge der Kombinatorik sind Induktion und Reduktion. Diese zu modellieren, sind Dedekind-Tripel bestens geeignet. Sie formalisieren unser primitives Verständnis vom 'Zählen': Es gibt ein erstes Element 0, und zu einem gegebenen Element a gibt es ein nächstes, succ(a), und alle zum Zählen benutzten Elemente sind aufgelistet in der Reihe 0, succ(0), succ(succ(0)), usw.

Addition heißt dann einfach:

```
if b = 0  then a+b := a
          else a+b := succ(a) + pred(b),
```

wobei pred definiert ist durch succ(pred(b)) = b. Weiter ist die Multiplikation ein 'Zählen in Sprüngen', d. h.

```
if b = 0  then a*b := 0
          else a*b := a*pred(b) + a.
```

Diese Dinge werden nun formalisiert und präzisiert.

Wir starten mit einer Menge D, einem Element $0 \in D$ und einer Abbildung succ: D --> D. Eine Untermenge $X \subseteq D$ heißt SUBALGEBRA von (D,0,succ) genau dann, wenn $0 \in X$ und $succ(X) \subseteq X$ ist.

Definition (4.7):

(D,0,succ) heißt DEDEKIND-TRIPEL :<=>

1) $0 \notin succ(D)$.
2) succ ist injektive Abbildung.
3) D ist die einzige Subalgebra von (D,0,succ).

Die Existenz von Dedekind-Tripeln folgt aus der Existenz unendlicher Mengen zusammen mit anderen Axiomen der Mengentheorie.

Der wichtigste Satz über Dedekind-Tripel ist Dedekind's Rekursionstheorem. Eine speziellere Fassung dieses Existenzund Eindeutigkeitssatzes für die Auflösung von Rekursionsregeln findet sich in Satz (5.2).

Satz (4.8): [Dedekind's Rekursionstheorem]

Sei (D,∅,succ) ein Dedekind-Tripel. Sei A eine Menge, a ∈ A. Wenn R eine Abbildung von D x A nach A ist (die 'REKURSIONSREGEL'), dann existiert genau eine Abbildung f: D ⟶ A mit

$$f(\emptyset) = a$$
$$\text{Für alle } x \in D: \quad f(\mathrm{succ}(x)) = R(x,f(x)).$$

Beweis:
Sei H die Menge aller Relationen $F \subseteq D \times A$ mit den Eigenschaften

a) $(\emptyset,a) \in F$
b) $(x,s) \in F \Rightarrow (\mathrm{succ}(x), R(x,s)) \in F.$

Offensichtlich ist $D \times A \in H$. Setze nun

$$f := \bigcap F,$$

wobei der Durchschnitt über alle $F \in H$ zu nehmen ist. f erfüllt a) und b), und f ist das kleinste Element von H. Wir zeigen, daß f wirklich eine Abbildung ist.

Sei X die Menge aller $x \in D$, so daß genau ein $s \in A$ mit $(x,s) \in f$ Zuerst zeigen wir $\emptyset \in X$. Sicherlich ist $(\emptyset,a) \in f$. Nehme an: $(\emptyset,b) \in f$ und $a \neq b$.

Setze $f' := f - \{(\emptyset,b)\}$. f' erfüllt a). Nehme an, daß $(y,s) \in f'$. Daraus folgt $(y,s) \in f$ mit $(\mathrm{succ}(y), R(y,s)) \in f$.
Aus der Eigenschaft 1) der Dedekind-Tripeln folgt

$$\emptyset \neq \mathrm{succ}(y)$$
$$\Rightarrow (\mathrm{succ}(y), R(y,s)) \in f'$$
$$\Rightarrow f' \in H,$$

was einen Widerspruch darstellt, da $f' \subsetneq f$ und f als kleinstes Element von H gewählt ist. Also ist $\emptyset \in X$. ⁂

Nehme an: $x \in X \Rightarrow$ es existiert ein eindeutiges $s \in A$ mit $(x,s) \in f$ und auch $(\mathrm{succ}(x), R(x,s)) \in f$. Nehme an, daß ein $t \in A$ mit $t \neq R(x,s)$ und $(\mathrm{succ}(x), R(x,s)) \in f$ existiert und betrachte die Relation

$$f' := f - \{(succ(x),t)\}.$$

Aus $\emptyset \notin succ(x)$ folgt $(\emptyset,a) \in f'$. Sei $(y,u) \in f' \Rightarrow (succ(y),R(y,u)) \in f$. Es gibt zwei Fälle:

$$y \neq x \Rightarrow (succ(y), R(y,u)) \in f'.$$

$$y = x \Rightarrow u = s,\ (succ(y), R(y,u)) = (succ(y), R(x,s))$$
$$\Rightarrow (succ(y), R(y,u)) \in f'.$$

Wir erhalten also $f' \in H$ und wegen $f' \subsetneq f$ wieder einen Widerspruch zur Definition von f.

Wir haben damit gezeigt, daß X Subalgebra von $(D,\emptyset,succ)$ ist, damit also $X = D$ gilt. Dies zeigt, daß f eine Abbildung ist, und mehr noch: f hat die geforderten Eigenschaften

$$f(\emptyset) = a$$
$$f(succ(x)) = R(x,f(x)).$$

Nehme nun an, daß g eine zweite solche Abbildung ist:

$$\Rightarrow g \in H$$
$$\Rightarrow f \subseteq g$$
$$\Rightarrow f = g,$$

da g eine Abbildung ist. Dies zeigt die Eindeutigkeit von f und beendet den Beweis ab.

Man beachte, daß Induktion nicht dasselbe ist wie Rekursion. Ein Tripel $(D,\emptyset,succ)$, welches 3) erfüllt, erlaubt Induktion, aber nur Dedekind-Tripel erlauben auch Rekursion, wie der folgende Satz zeigt.

Satz (4.9):

Sei D eine Menge mit mindestens 2 Elementen, $\emptyset \in D$. Sei $succ: D \longrightarrow D$ eine Abbildung, so daß $(D,\emptyset,succ)$ keine andere Subalgebra als D hat. Dann sind die folgenden Aussagen äquivalent:

a) $(D,\emptyset,succ)$ ist ein Dedekind-Tripel.

b) Sei A eine Menge, $R: D \times A \longrightarrow A$ eine Abbildung, $a \in A$
$\Rightarrow$ es existiert eine Abbildung $f: D \longrightarrow A$ mit
$$f(\emptyset) = a$$
$$\text{Für alle } x \in D : f(succ(x)) = R(x,f(x)).$$

c) Es existiert eine Abbildung $pred: D \longrightarrow D$ mit
$$pred(\emptyset) = \emptyset$$
$$\text{Für alle } x \in D : pred(succ(x)) = x.$$

Beweis:
a) => b) ist der Inhalt von Satz (4.8).

b) => c):
Definiere R: D x D --> D durch R(x,y) := x. Dann existiert pred: D --> D mit

pred(∅) = ∅
pred(succ(x)) = R(x,pred(x)) = x.

c) => a):
succ ist injektiv wegen pred*succ = id.

Annahme: ∅ = succ(x)
=> ∅ = pred(succ(x)) = x
=> {∅} ist Subalgebra von (D,∅,succ)
=> D = {∅}.

Widerspruch, da D mindestens 2 Elemente enthält.

=> ∅ ∉ succ(D)
=> a).

Das einfachste Beispiel einer Algebra (D,∅,succ), die nur Induktion, aber nicht Rekursion erlaubt, ist D := {∅,1} mit

succ(∅) := 1, succ(1) := ∅.

Ein Beispiel, in dem succ nicht injektiv ist, ist D := {∅,1,2} mit

succ(∅) := 1, succ(1) := 2, succ(2) := 1.

Definition (4.10):

Sei (D,∅,succ) ein Dedekind-Tripel. Für x ∈ D setzen wir

$$T(x) := \bigcap (Y \mid Y \subseteq D,\ x \in Y,\ succ(Y) \subseteq Y).$$

T steht für 'Tail', denn T(x) besteht aus x und allen 'späteren' Elementen in D.

Satz (4.11):

Sei (D,∅,succ) ein Dedekind-Tripel. Dann gilt für alle x ∈ D:

a) $x \in T(x)$.
b) $z \in T(x) \Rightarrow T(z) \subseteq T(x)$.
c) $T(\emptyset) = D$.
d) $x \notin T(succ(x))$.
e) $T(x) = T(succ(x)) \cup \{x\}$.

Beweis:
folgt sofort aus der Definition von T(x) und den Eigenschaften eines Dedekind-Tripels.

Definition (4.12):

Sei (D,∅,succ) ein Dedekind-Tripel. Für $x \in D$ setzen wir

$$Dlt(x) := D - T(x).$$

'Dlt' steht für 'D with lost tail'. Es gelten analog zu (4.11) folgende Eigenschaften:

Satz (4.13):

Sei (D,∅,succ) ein Dedekind-Tripel. Dann gilt:
a) $Dlt(\emptyset) = \emptyset$.
b) $x \notin Dlt(x)$ für alle $x \in D$.
c) $Dlt(succ(x)) = Dlt(x) \cup \{x\}$ für alle $x \in D$.
d) Die Abbildung Dlt: D --> {Teilmengen von D} ist injektiv.
e) $y \in D$, $x \in Dlt(y) \Rightarrow Dlt(x) \subseteq Dlt(y)$.
f) $x, y \in D \Rightarrow Dlt(x) \subseteq Dlt(y)$ oder $Dlt(y) \subseteq Dlt(x)$.
g) $\bigcup (Dlt(x) \mid x \in D) = D$.

Wir notieren noch zwei Eigenschaften der Abbildung pred aus (4.9c):

Satz (4.14):

Sei (D,∅,succ) ein Dedekind-Tripel und pred die eindeutige Abbildung pred: D --> D mit pred(∅) = ∅ und pred * succ = id. Dann gilt:

a) $x \in D \Rightarrow x \neq succ(x)$.
b) $x \in D, x \neq \emptyset \Rightarrow pred(x) \neq x$ und $succ(pred(x)) = x$.

Aus (4.8) folgt, daß Dedekind-Tripel bis auf Isomorphie eindeutig bestimmt sind; das Standardbeispiel für diese Tripel bilden die natürlichen Zahlen ℕ.

Satz (4.15):

Seien (D,∅,succ) und (E,∅,succ) Dedekind-Tripel. Dann gibt es genau einen Isomorphismus f: D --> E, d.h. eine bijektive Abbildung f mit den Eigenschaften

$$f(\emptyset) = \emptyset$$
$$f(succ(x)) = succ(f(x)) \quad \text{,für alle } x \in D.$$

In (1.5) wurde der Begriff Ordnungsrelation als binäre Relation auf

einer Menge mit den Eigenschaften der Reflexivität, der Antisymmetrie und der Transitivität. Wir erweitern diese Begriffsbildung nun zur 'Wohlordnung'.

Definition (4.16):

Eine Menge H heißt WOHLGEORDNET, wenn auf H eine Ordnungsrelation definiert ist und zusätzlich jede nichtleere Teilmenge von H ein kleinstes Element hat.

Satz (4.17):

Sei (D,∅,succ) ein Dedekind-Tripel. Definiere die Relation '$\lesssim$' auf D durch

$$x \lesssim y := Dlt(x) \subseteq Dlt(y). \quad \text{('NATÜRLICHE ORDNUNG')}$$

Dann ist '$\lesssim$' eine Wohlordnung von D und es gilt:

$$x \lesssim y \iff succ(x) \lesssim succ(y).$$

Ferner gelten die Beziehungen (für alle $x, y \in D$):

a) $Dlt(x) = \{ y \mid y \in D, y < x \}$
b) $x < succ(x)$
c) $pred(x) < x$ (für $x \neq \emptyset$)
d) $x < y \lesssim succ(x) \Rightarrow y = succ(x)$.

Das nun folgende zweite Rekursionstheorem dient dazu, Addition und Multiplikation auf Dedekind-Tripeln einzuführen.

Satz (4.18):

Sei (D,∅,succ) ein Dedekind-Tripel, A und B Mengen, $R: B \times D \times A \to A$ und $g: B \to A$ Abbildungen. Dann existiert eine eindeutige Abbildung $F: B \times D \to A$ mit (für alle $b \in B$, für alle $x \in D$):

$$F(b,\emptyset) = g(b).$$
$$F(b,succ(x)) = R(b,x,F(b,x)).$$

Definiere die Addition auf dem gegebenen Dedekind-Tripel (D,∅,succ) durch $g := id_D$ und mit der Rekursionsregel

$$R(a,b,c) := succ(c).$$

Dann ist $g: D \to D$ (A=B=D), $R: D \times D \times D \to D$, und nach (4.18) gibt es eine Abbildung '+': $D \times D \to D$ mit

$$+(a,\emptyset) = g(a) = a$$
$$+(a,succ(b)) = R(a,b,+(a,b)) = succ(+(a,b)),$$

für alle a, b ∈ D. Schreibt man nun 'a+b' statt '+(a,b)', so heißt dies:

$$a + 0 = a$$
$$a + succ(b) = succ(a+b),$$

und man erkennt die Definition der Addition aus Beispiel (4.3) wieder.

Für die Addition gelten folgende, leicht nachzurechnende Regeln:

Satz (4.19):

a) $succ(a+b) = succ(a) + b.$

b) $$a + b = \begin{cases} a & , \ b = 0 \\ succ(a) + pred(b), & b \neq 0 \end{cases}$$

c) $a + b = b + a$, 'KOMMUTATIVITÄT'.
d) $(a+b)+c = a+(b+c)$, 'ASSOZIATIVITÄT'.
e) $a+c = b+c \Rightarrow a = b$, 'KÜRZUNGSREGEL'.

(D,0,+) ist also ein kommutatives Monoid, welches die Kürzungsregel erfüllt. Dabei ist ein MONOID eine Menge mit assoziativer Verknüpfung und Einselement (hier ist 0 das Einselement: m+0 = 0+m = m).

Die Verknüpfung '+' ist kompatibel mit '$\leqslant$', d.h. $a \leqslant b \Leftrightarrow a + c \leqslant b + c$, für beliebige a, b, c ∈ D. Es gilt sogar weiter, wie in Lemma (4.4) schon für D = ℕ bewiesen:

$$a \leqslant b \Leftrightarrow \text{es gibt } x \in D \text{ mit } a + x = b.$$

Zur Definition der Multiplikation gehen wir von einem Monoid (M,0,+) und einem Dedekind-Tripel (D,0,succ) aus. Ferner seien die Abbildungen g: M --> M mit g(a) = 0 für alle a ∈ M und R: M x D x M --> M mit R(a,x,b) := a + b gegeben.

Nach Satz (4.18) existiert eine Abbildung '*': M x D --> M mit

$$a * 0 = a$$
$$a * succ(x) = a + a * x \qquad \text{(für alle } a \in M, x \in D).$$

Im Falle (M,0,+) = (D,0,+) heißt '*' MULTIPLIKATION und mit der Bezeichnung

$$1 := succ(0)$$
$$\Rightarrow succ(a) = succ(a+0) = a + succ(0) = a + 1$$

gilt:

Satz (4.20):

Sei (D,0,succ) Dedekind-Tripel und (M,0,+) Monoid. Dann gilt für alle x, y ∈ D und a, b ∈ M:

a) 0 * x = 0.
b) a * 1 = a. c) a * (x+y) = a*x + a*y.
d) a * (x*y) = a*x + a*y.
e) Wenn (M,0,+) kommutativ ist: (a+b) * x = a*x + b*x.

(D,0,1,+,*) ist ein kommutativer SEMIRING, d. h. (D,0,+) und (D,1,*) sind kommutative Monoide und beide Distributivgesetze sind gültig.

Eigenschaften der Multiplikation:

Satz (4.21):

Sei (D,0,succ) ein Dedekind-Tripel. Für alle a, b, c ∈ D gilt:

a) für c ≠ 0: a * c = b * c <=> a = b (Kürzungsregel).
b) a * b = 0 => a=0 oder b=0 (Nullteilerfreiheit).
c) für c ≠ 0: a < b <=> a*c < b*c (Kompatibel mit '<').

Ein wirkungsvolles Werkzeug zur Durchführung von Berechnungen in Dedekind-Tripeln ist der in Kapitel 9 noch ausführlich behandelte Euklidische Algorithmus, die 'Division mit Rest':

Satz (4.22):

Seien a, b ∈ Dedekind-Tripel (D,0,succ) mit b > 0. Dann existiert q, r ∈ D mit

$$a = q * b + r, \quad r < b,$$

wobei q und r eindeutig bestimmt sind.

Kapitel 5 : Primitiv rekursive Funktionen

Die vollständige Induktion ermöglicht es, Funktionen durch ein Rekursionsschema zu definieren. Dabei gehen wir von $\mathbb{N}_0$ aus.

Definition (5.1):

Seien $g : \mathbb{N}_0^{\,k} \longrightarrow \mathbb{N}_0$ und $h : \mathbb{N}_0^{\,k+2} \longrightarrow \mathbb{N}_0$ zwei Funktionen, $k \in \mathbb{N}$.
Dann heißt $f : \mathbb{N}_0^{\,k+1} \longrightarrow \mathbb{N}_0$ mit

$$f(x,0) = g(x) \qquad (*)$$
$$f(x,y+1) = h(\, x,\, y,\, f(x,y)\,)$$

für $x \in \mathbb{N}_0^{\,k}$, $y \in \mathbb{N}$, eine durch ein PRIMITIVES REKURSIONSSCHEMA definierte Funktion.

Beispiel:
Sei $k = 1$, $g(x) = x$, $h(x,y,z) = z + 1$. Durch

$$f(x,0) = g(x)$$
$$f(x,y+1) = h(\, x,y,f(x,y)\,) = f(x,y) + 1$$

ist die Addition auf $\mathbb{N}_0$ erklärt, d.h. es gilt:

$$x + y = f(x,y).$$

Kann man g und h berechnen, so kann man die durch ein primitives Rekursionsschema definierte Funktion f berechnen. Man geht dazu folgendermaßen vor:

$$f(x,0) = g(x)$$
$$f(x,1) = f(x,0+1) = h(x,0,f(x,0))$$
$$f(x,2) = f(x,1+1) = h(x,1,f(x,1))$$
$$\text{usw.}$$

Satz (5.2):

Sind $k \in \mathbb{N}$, $g : \mathbb{N}_0^{\,k} \longrightarrow \mathbb{N}$ und $h : \mathbb{N}_0^{\,k+2} \longrightarrow \mathbb{N}_0$ gegeben, so existiert genau eine Funktion $f : \mathbb{N}_0^{\,k+1} \longrightarrow \mathbb{N}$, die obiges Rekursionsschema erfüllt.

Beweis:
Betrachte den Abschnitt $[0,n] = \{ x \mid x \in \mathbb{N},\ 0 \leq x \leq n \}$.
Ind.Beh.:
Für jeden Abschnitt $[0,n]$ gibt es genau eine Funktion

$$\varphi_n : [0,n]^{k+1} \longrightarrow [0,n],$$

die dem Rekursionsschema genügt, für $y + 1 \leq n$.

Ind.Anfang:
$n = 1$.

$$\varphi_1(x,y) := \begin{cases} g(x) & \text{für } y = 0 \\ h(x,0,g(x)) & \text{für } y = 1 \end{cases}$$

erfüllt (*) für $y + 1 \leq n = 1$, d.h. $y = 0$, denn

und
$$\begin{cases} \varphi_1(x,0) = g(x) \\ \varphi_1(x,y+1) = h(x,0,g(x)) = h(x,0,\varphi_1(x,y)). \end{cases}$$

φ_1 kann nicht anders definiert werden, wenn (*) erfüllt sein soll.

Ind.Schritt:
Für $n = k$ definiere

$$\varphi_k(x,y) = \begin{cases} \varphi_{k-1}(x,y) & \text{für } y \in [0,k-1] \\ h(x,k-1,\varphi_{k-1}(y)) & \text{für } y = k \end{cases}$$

φ_k erfüllt (*), da φ_{k-1} (*) erfüllt gemäß Ind.Vor.
φ_k kann nicht anders definiert werden.

Definition (5.3):

Die Menge der primitiv rekursiven GRUNDFUNKTIONEN besteht aus folgenden Funktionen:

1) NACHFOLGERFUNKTION : $s : \mathbb{N}_0 \longrightarrow \mathbb{N}_0$
$s(x) = x^+ = x + 1.$

2) NULLFUNKTION : $c : \mathbb{N}_0 \longrightarrow \mathbb{N}_0$
$c(n) = 0$ für alle $n \in \mathbb{N}_0$.

3) PROJEKTIONEN : $e_i^k : \mathbb{N}_0^k \longrightarrow \mathbb{N}_0$
(auf i. Komp.) $e_i^k(n_1, \ldots, n_k) = n_i.$

Definition (5.4):

Seien $g:\mathbb{N}_0^s \longrightarrow \mathbb{N}_0$ und $h_i : \mathbb{N}_0^r \longrightarrow \mathbb{N}_0$, i=1,...,s, Funktionen.

Dann heißt $f : \mathbb{N}_0^r \longrightarrow \mathbb{N}_0$, definiert durch

$$f(x_1, \ldots, x_r) = g\,(\,h_1(x_1, \ldots, x_r),\ldots,h_s(x_1,\ldots x_r)),$$

die durch SUBSTITUTION aus g und h_i, i= 1,...,s entstandene Funktion.

Definition (5.5):

Eine Funktion $f : \mathbb{N}_0^k \longrightarrow \mathbb{N}_0$ heißt PRIMITIV REKURSIV, wenn f entweder eine primitiv rekursive Grundfunktion ist, oder aus diesen in endlich vielen Schritten durch Substitution und primitive Rekursion erzeugbar ist.

Beispiele:

1)Addition entsteht durch primitive Rekursion aus der Nachfolgerfunktion s und den Projektionen e_1^1, e_3^3.
Rekursionsschema:

$$f(x,0) = g(x) = x = e_1^1(x)$$

$$f(x,y+1) = h(x,y,f(x,y)) = f(x,y)^*,$$

wobei

$$h(x,y,z) = z + 1 = S\,e_3^3\,(x,y,z) \text{ ist.}$$

2)Multiplikation:

Sei g(x) = c(x) die Nullfunktion und $h: \mathbb{N}_0^3 \longrightarrow \mathbb{N}_0$ mit h(x,y,z) = x + z. h ist eine primitiv rekursive Funktion.
Rekursionsschema:

$$f(x,0) = g(x)$$
$$f(x,y+1) = h(x,y,f(x,y)) = x + f(x,y).$$

Es gilt z.B.

$$\begin{aligned} f(2,3) &= 2 + f(2,2) \\ &= 2 + 2 + f(2,1) \\ &= 2 + 2 + 2 + f(2,0) \\ &= 2 + 2 + 2 + 0 \\ &= 6, \end{aligned}$$

allgemein:

$$f(x,y) = x * y.$$

Definition (5.6):

Die Funktion $B_1 : \mathbb{N} \longrightarrow \mathbb{N}$, definiert durch

$$B_1(x) = \begin{cases} 1 & \text{für } x = 0 \\ 2x & \text{für } x \geq 1 \end{cases}$$

heißt 0.TE ACKERMANNSCHE FUNKTION.

Die Funktion $B_{n+1}: \mathbb{N} \longrightarrow \mathbb{N}$, definiert durch

$$B_{n+1}(x) = B_n^{\,x}(1),$$

wobei $B_n^{\,x} = B_n\, B_n^{\,x-1}$, heißt N.TE ACKERMANNSCHE FUNKTION.

Lemma (5.7):

Die i. ten Ackermannschen Funktionen sind primitiv rekursiv.

Beweis:
Man hat für B_1 folgendes Rekursionsschema:

1) $B_1(0) = 1$
2) $B_1(x+1) = B_1(x) * \text{sgn}(x) + 2,$

wobei

$$\text{sgn}(x) = \begin{cases} 0 & \text{für } x = 0 \\ 1 & \text{für } x \geq 1 \end{cases}$$

eine primitive rekursive Funktion ist. Für B_{n+1} hat man folgendes Rekursionsschema:

1) $B_{n+1}(0) = 1$
2) $B_{n+1}(x+1) = B_n(B_{n+1}(x))),$

denn $B_{n+1}(x+1) = B_n^{\,x+1}(1) = B_n(B_n^{\,x}(1)) = B_n(B_{n+1}(x)).$

Satz (5.8):

Für die Ackermannschen Funktionen gilt:

1) $B_2(x) = 2^x$.

2) $B_3(x) = 2^{B_3(x-1)}$ für $x > 0$.
$B_3(0) = 1$.

3) $B_n(x+1) > B_n(x) > x$.

Beweis:

1) Ind.Anf.: $B_2(0) = B_1^0(1) := 1$.

Ind.Vor.: $B_2(0) = 2^x$ für ein $x \in \mathbb{N}$.

Dann folgt: $B_2(x+1) = B_1(B_2(x)) = B_1(2^x) = 2 * 2^x = 2^{x+1}$.
(Dabei folgt das erste '=' nach dem Rekursionsschema 5.7).

2) Ind.Anf.: $B_3(0) = B_2^0(1) := 1$.
Nach Rekursionsschema folgt sofort

$$B_3(x+1) = B_2(B_3(x)) = 2^{B_3(x)}.$$

Es gilt also:

$$B_3(x) = 2^x.$$

3) Induktionsanfang: (n=1)
$B_1(x+1) = 2(x+1) > 2x = B_1(x) > x$.

Induktionsvoraussetzung:
i) $B_n(x+1) > B_n(x)$ und ii) $B_n(x) > x$.

Induktionsschritt:
i) $B_{n+1}(x+1) = B_n(B_{n+1}(x))$, nach (5.7)
$> B_{n+1}(x)$, da $B_n(x) > x$.

ii) $B_{n+1}(x+1) = B_n^x(1) = B_n(B_n^{x-1}(1))$
$> B_n^{x-1}(1)$, da $B_n(x) > x$.

Man hat: $B_{n+1}(x) > B_n^{x-1}(1) > B_n^{x-2}(1) \ldots$
$> B_n^1(1) > 1$,nach Ind.vor.

Daraus ergibt sich, daß $B_{n+1}(x) > x$, da die absteigende Kette x natürliche Zahlen enthält, die streng monoton geordnet sind.

Nicht nur bei einem primitiven Rekursionsschema wird die wesentliche Eigenschaft der natürlichen Zahlen, nämlich das Prinzip der vollständigen Induktion verwendet. Es gibt auch allgemeine Rekursionsschemen, die wie bei ~~einem~~ dem folgenden Beispiel keine primitiv rekursive Funktion ergeben.

Definition (5.9):

Die Funktion $A : \mathbb{N}_0^2 \longrightarrow \mathbb{N}$, definiert durch $A(n,x) := B_n(x)$ heißt (ALLGEMEINE) ACKERMANNSCHE FUNKTION.

Man bemerke, daß die Ackermannsche Funktion durch 'doppelte Rekursion' gegeben ist. Für das Resultat, daß A nicht primitiv rekursiv ist, verweisen wir auf [Noltemeier, Informatik I].

Verwandt mit der Ackermannschen Funktion ist die Funktion $C:\mathbb{N}_0^2 \longrightarrow \mathbb{N}$, definiert durch das nicht primitive Rekursionsschema:

$$\begin{aligned} C(0,m) &= m+1. \\ C(n+1,0) &= C(n,1). \\ C(m+1,n+1) &= C(m,C(n+1,m)). \end{aligned}$$

Kapitel 6 : Über die Kardinalität von Mengen

Zählen von Gegenständen besteht darin, den Gegenständen in einer Reihenfolge Nummern 1, 2, 3, ... zuzuordnen. Zählen einer nicht leeren endlichen Menge M von Elementen besteht also in einer Bijektion f:M ---> [0,n] auf einen Abschnitt der natürlichen Zahlen.

Lemma (6.1):

Seien $n, m \in \mathbb{N}_0$. Ist $m < n$, dann gibt es keine bijektive Abbildung $f : [0,n] \cap \mathbb{N} \longrightarrow [0,m] \cap \mathbb{N}$.

Beweis:
Induktionsanfang: $n = 1$. Dann ist $m = 0$ und es gibt keine bijektive Abbildung $f : [0,1] \cap \mathbb{N} = \{0,1\} \longrightarrow \{ 0 \}$.
Induktionsvoraussetzung: Die Aussage sei für $n \in \mathbb{N}$, bei beliebigem m, $m < n$, richtig.

Induktionsschritt:
Sei nun $m < n + 1$.
Der Fall $m = 0$ scheidet aus, weil auf $\{ 0 \}$ dann die Zahlen n und n+1 abgebildet werden müßten, d.h. die Abbildung ist nicht bijektiv.
Betrachte nun für $m > 0$ eine Abbildung $f : [0,n+1] \longrightarrow [0,m]$. Wenn nun f bijektiv wäre, dann wäre auch die Abbildung $g : [0,n+1] \longrightarrow [0,m]$, definiert durch

$$g(x) = \begin{cases} f(x) & \text{,für } x \neq n+1 \text{ und } f(x) \neq m \\ f(n+1) & \text{,für } f(x) = m \\ m & \text{,für } x = n+1 \end{cases}$$

bijektiv.
Entgegen der Induktionsvoraussetzung ist dann $h : [0,n] \longrightarrow [0,m-1]$ bijektiv, wobei $h(x) = g(x)$ für $x \in [0,n]$ definiert ist.

Satz (6.2):

Sei $n, m \in \mathbb{N}_0$. Dann gilt: Es gibt eine bijektive Abbildung $f : [0,n] \longrightarrow [0,m]$ genau dann, wenn $n = m$.

Beweis:
'<=': Ist $m = n$, dann ist die identische Abbildung $id_{[0,n]}$ eine bijektive Abbildung.
'=>': $f : [0,n] \longrightarrow [0,m]$ bijektiv, so folgt aus (6.1), daß $m < n$ nicht gelten kann. Da es zu f eine Umkehrabbildung $g : [0,m] \longrightarrow [0,n]$ gibt, folgt analog, daß $n < m$ unmöglich ist. Also ist $n = m$.

Definition (6.3):

Unter der ANZAHL |M| der Elemente einer endlichen, nicht leeren Menge M versteht man diejenige Zahl $n \in \mathbb{N}$, für die eine bijektive Abbildung $f : M \longrightarrow [1,n]$ existiert.
Für $M = \emptyset$ ist $|M| = 0$.
Existiert keine bijektive Abbildung der nicht leeren Menge M auf einen Abschnitt von $\mathbb{N}$, so heißt M UNENDLICH und man schreibt $|M| = \infty$

|M| heißt auch die KARDINALITÄT der Menge M.

Bemerkung (6.4):

Seien N, M zwei nicht leere Mengen mit $|N| < |M|$. Ist $f : M \longrightarrow N$ eine surjektive Abbildung, so gibt es mindestens ein $y \in \mathbb{N}$, so daß y mindestens zwei verschiedene Urbilder besitzt.

Daraus ergibt sich ein kombinatorisches Zählprinzip, welches sich oft anwenden läßt:

Lemma (6.5): [SCHUBFACHPRINZIP]

Bei einer Verteilung von m Gegenständen auf n Schubfächer, wobei $m > n$, muß mindestens in einem Schubfach mehr als ein Gegenstand vorkommen.

Beispiel:
Seien 15 Mikrocomputer und 10 Drucker gegeben. Alle 5 Minuten benötigt ein Teil der 15 Mikrocomputer Drucker. Wieviele Verbindungsleitungen sind notwendig, um sicherzustellen, daß von 10 oder weniger Mikrocomputern, die gleichzeitig Bedarf anmelden, jeder einen eigenen Drucker zur Verfügung hat. Dabei soll jeder Drucker mit gleich vielen Mikrocomputern verbunden sein.

Lösung:
Es müssen mindestens 60 Verbindungsleitungen sein. Nimmt man nämlich an, daß ein Drucker nur mit 5 (oder weniger) Mikrocomputern verbunden ist, kann es sein, daß keiner dieser 5 Mikrocomputer unter den 10 Mikrocomputern ist, die gerade einen Drucker suchen. Dieser Drucker würde also nicht zur Verfügung stehen, es wären nur noch 9 Drucker verfügbar.

Satz (6.6):

i) M ist genau dann endlich, wenn es keine bijektive Abbildung von M auf eine echte Teilmenge von M gibt.
ii) M ist genau dann unendlich, wenn es eine echte Teilmenge N von M gibt, so daß M bijektiv zu N ist.

Beweis:
ii) '=>':
Wähle $x_1 \in M$, $x_2 \in M - \{ x_1 \}$ usw. Aus $M - \{ x_i, i = 1, \dots, n \}$ wählen wir x_{n+1} aus, was immer möglich ist, da diese Menge nach Voraussetzung (M unendlich) nicht leer sein kann.
Es gibt also eine Teilmenge $\{ x_i \mid i \in \mathbb{N} \}$ in M. Wir definieren nun $g : M \longrightarrow M$ durch

$$g(x) = \begin{cases} x & , \text{ für } x \in M - \{ x_i \mid i \in \mathbb{N} \} \\ x_{i+1} & , \text{ für } x = x_i. \end{cases}$$

offensichtlich ist g injektiv und $g[M]$ ist eine echte Teilmenge von M, die x_1 nicht enthält.

ii) '<=':
Voraussetzung: Es gibt eine Bijektion von M auf eine echte Teilmenge $N \subsetneq M$.
Annahme: M ist endlich, d.h. nach Definition (6.3), daß eine bijektive Abbildung $f : M \longrightarrow [1,m]$ existiert. Da mit M auch N endlich ist, gibt es auch eine bijektive Abbildung $g : N \longrightarrow [1,n]$, mit $n < m$. Also gibt es auch eine solche von $[0,n]$ nach $[0,m]$, im Widerspruch zu (6.2), da $n < m$.

i) ist nur eine Umformulierung von ii).

Beispiel:
$\mathbb{N}$ ist eine unendliche Menge, denn

$$f:\mathbb{N} \longrightarrow 2\mathbb{N}, \quad f(x) = 2x,$$

ist eine Bijektion auf eine echte Teilmenge von $\mathbb{N}$.

Definition (6.7):

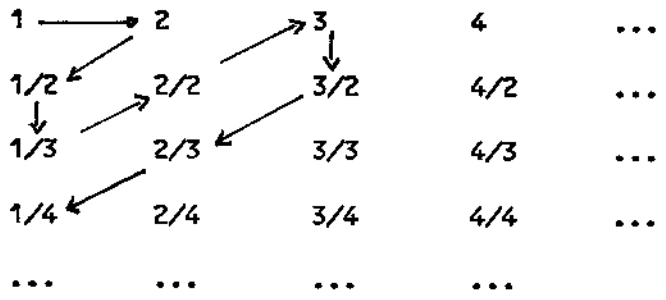
i) Eine Menge M heißt ABZÄHLBAR UNENDLICH, wenn M bijektiv zu $\mathbb{N}$ ist.
ii) Eine unendliche Menge, die nicht abzählbar unendlich ist, heißt ÜBERABZÄHLBAR UNENDLICH.

Beispiel (6.8a):

Die Menge $\mathbb{Q}^+$ aller positiven rationalen Zahlen ist abzählbar unendlich.

Beweis:

Jedes $q \in \mathbb{Q}^+$ läßt sich darstellen als Bruch zweier natürlicher Zahlen. Betrachte nun folgendes Schema:



Es wird über den Linienzug abgezählt, wobei offenkundig viele rationale Zahlen mehrmals gezählt werden. Dies definiert eine surjektive Abbildung $f : \mathbb{N} \longrightarrow \mathbb{Q}^+$. Überspringt man alle Brüche, die bereits in anderer Darstellung erfaßt wurden, so wird die Abbildung bijektiv.

Beispiel (6.8b):
Die Menge $[0,1] \cap \mathbb{R}$ ist überabzählbar unendlich.

Beweis:
Jede reelle Zahl $r \in [0,1]$ läßt sich auf genau eine Weise als ein unendlicher Dezimalbruch $r = 0,x_1x_2x_3 \ldots$ schreiben. Dabei ergeben sich die rationalen Zahlen als periodische Dezimalbrüche, während die irrationalen Zahlen keine Perioden haben.
Wir nehmen nun an, daß die reellen Zahlen aus $[0,1]$ in einer Reihenfolge abgezählt werden könnten:

$$\begin{aligned} r_1 &= 0, a_{11}\ a_{12}\ a_{13} \\ r_2 &= 0, a_{21}\ a_{22}\ a_{23} \\ r_3 &= 0, a_{31}\ a_{32}\ a_{33} \\ &\quad \text{usw.} \end{aligned}$$

Bilde nun einen neuen Dezimalbruch $b = 0, b_1\ b_2\ b_3$ mit

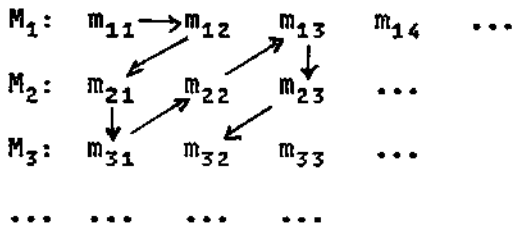
$$b_i \neq a_{ii} \text{ und } b_i \neq 0.$$

Nun ist $b \notin \{ r_i \mid i \in \mathbb{N} \}$, da $b_i \neq a_{ii}$. Also lassen sich die reellen Zahlen (zwischen 0 und 1) nicht abzählen.

Satz (6.9):

Die Vereinigung von abzählbar vielen Mengen $M_1, M_2, M_3, \ldots$, wobei jedes M_i abzählbar ist, ist ebenfalls eine abzählbare Menge.

Beweis:
Benutze wieder das Cantorsche Diagonalverfahren:



Zähle nach dem eingezeichneten Linienzug ab.

Satz (6.10):

Sei A abzählbare Menge. Dann ist die Menge aller endlichen Folgen von A eine abzählbare Menge.

Bemerkung (6.11):
Eine Menge A von Symbolen (Buchstaben) heißt ALPHABET. Ein WORT ist eine endliche Folge von Buchstaben. Aus (6.10) folgt also, daß die Menge aller Wörter bei einem abzählbaren Alphabet abzählbar ist.

Kapitel 7 : Permutationen und Kombinationen

In diesem Kapitel behandeln wir Probleme der klassischen Kombinatorik, deren Ergebnisse von Bedeutung für die endliche Wahrscheinlichkeitstheorie und die Statistik sind.

Definition (7.1):

Sei X eine N-MENGE, d.h. $|X| = n$, sei $k \in \mathbb{N}_0$ fixiert. Dann ist eine K-PROBE $x_1, \ldots, x_k$ aus X eine Hintereinanderreihung von k Elementen $x_1, \ldots, x_k$ aus X. Die Menge der k-Proben aus X bezeichnen wir mit X(k).
Eine k-Probe $x_1, \ldots, x_k \in X(k)$ heißt

1) GEORDNET, wenn $x_1, \ldots, x_k$ mit $(x_1, \ldots, x_k)$ identifiziert wird.

2) UNGEORDNET, wenn $x_1, \ldots, x_k$ mit $x_{\pi(1)}, \ldots, x_{\pi(k)}$ identifiziert wird für alle $\pi \in \{1,\ldots,k\}!$, d.h. für alle Permutationen (bijektive Abbildung) der Menge $\{ 1, \ldots, k \}$.

3) OHNE WIEDERHOLUNG, wenn $|\{x_1, \ldots, x_k\}| = k$, die x_i also paarweise verschieden sind.

4) MIT WIEDERHOLUNG, wenn $|\{ x_1, \ldots, x_k \}| < k$.

Mit den Begriffen 'Auswahl' und 'Tupel' läßt sich dies auch in folgender Weise definieren:

Definition (7.2):

Sei X eine n-Menge, $k \in \mathbb{N}_0$ fixiert.
Eine K-AUSWAHL B aus X ist eine Auswahl von k Elementen aus X, wobei es auf die Reihenfolge nicht ankommt.
Ein k-TUPEL B aus X ist eine Auswahl von k Elementen aus X, wobei es auf die Reihenfolge ankommt.

Die Bezeichnung von B als 'k-Auswahl' weist darauf hin, daß es sich um keine Menge handelt (z.B. ist $\langle 1,1,4,4 \rangle$ eine 4-Auswahl von $\{1,2,3,4\}$). Dennoch läßt sich aber eine Auswahl mit den Begriffen der Mengenlehre exakt definieren.
Eine k-Auswahl wird auch allgemein als MULTIMENGE bezeichnet.

Wir können nun auf vier verschiedene Weisen k Elemente aus X ziehen:

α1) Die Reihenfolge ist von Bedeutung, Wiederholungen nicht zulässig:

Name: - Geordnete k-Probe ohne Wiederholung
- k-Tupel ohne Wiederholung
- K-PERMUTATION

α2) Reihenfolge ist von Bedeutung, Wiederholungen erlaubt:

Name: - Geordnete k-Probe mit Wiederholung
- k-Tupel mit Wiederholung
- K-VARIATION

β1) Reihenfolge unwesentlich, Wiederholungen unzulässig:

Name: - Ungeordnete k-Probe ohne Wiederholung
- k-Auswahl ohne Wiederholung
- K-KOMBINATION

β2) Reihenfolge von Bedeutung, Wiederholungen erlaubt:

Name: - Ungeordnete k-Probe mit Wiederholung
- k-Auswahl mit Wiederholung
- K-REPETITION

Problem (7.3):
Sei $N = \{ 1, 2, \ldots, n \}$ und $0 \leqslant k \leqslant n$. Es sollen Formeln hergeleitet werden für die Anzahlen der k-Proben aus N für die vier oben angegebenen Spezifikationen.

Es gilt folgende Übersicht (dabei ist W=Wiederholung [ja!nein], R=Reihenfolge, n = 4 und k = 3):

	Name	W?	R?	Anzahl	Beispiel
α1)	k-Permutation	n	j	w(n,k)	(1,2,3)
α2)	k-Variation	j	j	v(n,k)	(1,1,4)
β1)	k-Kombination	n	n	c(n,k)	{1,2,3}
β2)	k-Repetition	j	n	r(n,k)	>1,1,4<

Um dieses Problem zu lösen, führen wir einige Bezeichnungen ein:

Definition (7.4):

Seien $n, k \in \mathbb{N}_0$.
Die für alle $n \in \mathbb{N}$ definierte Funktion $f(n) = n!$ mit

$$n! = \begin{cases} 1 & ,n = 0 \\ n*(n-1)! & ,n \in \mathbb{N} - \{ 0 \} \end{cases}$$

heißt 'N-FAKULTÄT'. Es ist $n! = \prod_{i=1..n} n$.

Die für alle $n, k \in \mathbb{N}$ definierte Funktion $f(n,k) = \binom{n}{k}$ (lies: n über k) mit

$$\binom{n}{k} = \begin{cases} n! \,/ (k!\ (n-k)!) & \text{für } 0 < k \leq n \\ 0 & \text{für } 0 < n < k \end{cases}$$

heißt BINOMIALKOEFFIZIENT.

Die für alle $n, k \in \mathbb{N}$, $k \leq n$ definierte Funktion $f(n,k) = (n)_k$ mit

$$(n)_k = n! \,/\, (n-k)! = \binom{n}{k} k!$$

heißt 'K UNTER N'.

Bemerkung (7.5):

Es gilt:

$$\binom{n}{k} = \frac{n!}{k!\ (n-k)!} = \frac{1*2* \ \ldots \ *n}{(1*2 \ldots *k)\ (1*2 \ldots *(n-k))}$$

$$= \frac{n*(n-1)* \ \ldots \ * \ (n-k+1)}{1*2* \ \ldots \ *k}$$

und

$$(n)_k = \frac{n!}{(n-k)!} = n * (n-1) \ \ldots \ * \ (n-k+1).$$

Satz (7.6):

Für $0 \leqslant k \leqslant n$ gelten die Formeln:

$\alpha 1)$ $w(n,k) = (n)_k$

$\alpha 2)$ $v(n,k) = n^k$

$\beta 1)$ $c(n,k) = \binom{n}{k}$

$\beta 2)$ $r(n,k) = \binom{n+k-1}{k}$

Beweis:
$\alpha 1)$: Für die 1. Komponente hat man n Möglichkeiten, für die zweite nur noch (n-1) Möglichkeiten, da es ja keine Wiederholung geben darf. Für die letzte Komponente bleiben schließlich nur noch (n-k+1) Möglichkeiten. Daher gilt:

$$w(n,k) = n*(n-1) \ldots *(n-k+1) = (n)_k.$$

Speziell für k=n hat man $w(n,n) = (n)_n = n!$ Möglichkeiten.

$\alpha 2)$: Da Wiederholungen erlaubt sind, hat man für jede Komponente n Möglichkeiten, also insgesamt $v(n,k) = n^k$.

$\beta 1)$: Wir teilen die Permutationen in disjunkte (=elementfremde) Klassen auf. Zu einer Klasse gehören alle Permutationen, die aus den gleichen k Elementen aus N gebildet werden. Also gehören zu jeder Klasse k! Permutationen, wie in $\alpha 1)$ mit k=n gezeigt wurde.
Jede Klasse läßt sich mit genau einer k-Kombination identifizieren, weil es auf die Reihenfolge bei einer 'Auswahl' nicht ankommt. Also haben wir

$$c(n,k) = \frac{(n)_k}{k!} = \binom{n}{k}.$$

$\beta 2)$: Wir interpretieren jede k-Repetition als k-Kombination über einer Menge mit n+k-1 Elementen, so daß $\beta 1)$ anwendbar wird.
Sei

$$R_{n,k} = \{ 1, \ldots, n, x_1=x_2, \ldots, x_{k-1} = x_k \}$$

eine Menge, die aus n Zahlen und k-1 Gleichungen besteht. Es gilt also: $| R_{n,k} | = n+k-1$.
Beh.: Einer k-Kombination aus $R_{n,k}$ entspricht genau eine k-Repetition aus N und umgekehrt (d.h. es gibt eine bijektive Abbildung).
Bew.: Ist eine k-Kombination aus $R_{n,k}$ gegeben, so ordne man die

darin vorkommenden Zahlen (es kommt mindestens eine Zahl vor) der Größe nach und wiederhole sie gemäß den Gleichungen. Dadurch entsteht eine k-Repetition über N.
Für n=10 und k=8 entspricht die Kombination

$$\{1,\ 3,\ 5,\ 6,\ 9,\ x_2 = x_3,\ x_3 = x_4,\ x_7 = x_8\ \}$$

der Repetition >1,3,3,3,5,6,9,9<.
Umgekehrt läßt sich zu jeder k-Repetition eine k-Kombination von $R_{n,k}$ angeben. Also haben wir:

$$r(n,k) = c(n+k-1,k) = \binom{n+k-1}{k}.$$

Damit ist der Satz vollständig bewiesen.

Wir geben noch einige nützliche Formeln an:

Satz (7.7):

1) $\binom{n}{k} = \binom{n}{n-k}$ für $0 \leq k \leq n$. [SYMMETRIESATZ]

2) $\binom{n}{0} = \binom{n}{n} = 1$

3) $\binom{n}{1} = \binom{n}{n-1} = n$

4) Für $a, b \in \mathbb{R}$, $n \in \mathbb{N}$ gilt der BINOMISCHE SATZ:

$$(a+b)^n = \Sigma_{k=0\ldots n}\ a^{n-k} b^k \binom{n}{k}.$$

Dabei ist $0^0 := 1$ (für a=b=0).

5) $\binom{n}{k} * \binom{k}{i} = \binom{n}{i} * \binom{n-i}{k-i}$ $\quad i,k,n \in \mathbb{N}_0,\ i \leq k \leq n$

6) $\Sigma_{k=0\ldots n} = \binom{n}{k} 2^n$

7) $\Sigma_{k=0\ldots n}\ (-1)^k \binom{n}{k} = 0$

8) $\Sigma_{k=r\ldots n}\ (-1)^k \binom{k}{r}\binom{n}{k} = 0,\ n \geq r$

9) $\Sigma_{i=0\ldots m}\ \binom{r}{i}\binom{s}{m-i} = \binom{r+s}{m}$

Beweis:
1), 2), 3) und 5) folgen sofort aus Def. (7.4), 1) und 2) durch Einsetzen und einfaches Nachrechnen.
4) ist ein längerer, geradezu typischer Induktionsbeweis (über n).

6) ergibt sich aus 4), wenn man a = b = 1 setzt.
7) ergibt sich aus 4), wenn man a = 1, b = -1 setzt.

9) ergibt sich aus $(1+x)^r\ (1+x)^s = (1+x)^{r+s}$ durch Verwendung von 4) und anschließenden Koeffizientenvergleich.

Der binomische Satz (7.6,4.) rechtfertigt auch den Namen 'Binomialkoeffizient'; ordnet man nämlich die Koeffizienten der binomischen Formeln $(a+b)^n$, $n\in\mathbb{N}$, im PASCALSCHEN DREIECK (PD), beginnend mit $(a+b)^1 = a+b$, so ergibt sich

n = 1 1 1
n = 2 1 2 1
n = 3 1 3 3 1
n = 4 1 4 6 4 1

Allgemein gilt für die Zahlem im PD: Die k-te Zahl der n-ten Zeile ist die Summe der (k-1)-ten und der k-ten Zahl der (n-1)-ten Zeile, für $2 \leq k \leq n-1$. Die 0-te und n-te Zahl jeder Zeile ist eine 1.
Es gilt nun: Der Binomialkoeffizient 'n über k' ist die k-te Zahl der n-ten Zeile im PD; dabei beginnt die Numerierung der Zahlen in einer Zeile bei 0 und nicht bei 1. Das Bildungsgesetz des PD entspricht genau dem Additionssatz (7.7,2.) , mit den Randbedingungen aus (7.7,3.).

Beispiele:
Zu den in (7.6) bewiesenen Formeln sei je ein Beispiel genannt:

α1):
Es gibt genau $w(8,3) = (8)_3 = 336$ verschiedene Tips für die

Reihenfolge der drei Erstplazierten bei einem Turnier, an dem sich 8 Mannschaften beteiligen.

α2)
Sei $A = \{a_1, \ldots, a_n\}$ ein Alphabet. Ein Wort heißt von der Länge k, wenn es k Buchstaben enthält. Es gibt $v(26,3) = 26^3$ voneinander verschiedene Wörter mit drei Buchstaben eines Alphabets mit 26 Buchstaben.

β1)
Beim Lotto '6 aus 49' gibt es

$$c(49,6) = \binom{49}{6} = \frac{49*48*47*46*45*44}{1*2*3*4*5*6} = 13\ 983\ 815$$

verschiedene Kombinationen.

Systematische Erzeugungen

Wir betrachten im folgenden n-Tupel ohne Wiederholungen, also Permutationen, auf der Menge $N = \{1, 2, 3, \ldots, n\}$. Wir wollen alle möglichen Permutationen auf N erzeugen. Dazu brauchen wir eine Methode, die Permutationen 'der Reihe nach' aufzulisten.

Definition (7.8): [LEXIKOGRAPHISCHE ANORDNUNG]

Die Permutation $(a_1,\ldots, a_n)$ KOMMT VOR der Permutation $(b_1,\ldots,b_n)$, falls es ein m gibt, $0 \leq m < n$, mit

$$a_i = b_i \quad \text{für } i = 1, \ldots, m-1$$
$$a_m < b_m.$$

Anmerkung: 'm<n', da aus $a_i = b_i$, $1 \leq i \leq n-1$, schon $a_n=b_n$ folgt.

Beispiel:
Sei n=6. Die Permutation (1,2,4,6,5,3) kommt vor (1,2,5,6,4,3), denn $a_1=b_1$, $a_2=b_2$, $a_3=4 < 5=b_3$ (m=3).

Problem:
Gegeben sei eine Permutation $(a_1, \ldots, a_n)$. Wie erhält man alle Permutationen, die nach der lexikographischen Anordnung folgen?

Beispiel:
Der Nachfolger von (1,2,4,6,5,3) ist (1,2,5,3,4,6).

Algorithmus (7.9): [Bestimmen des Nachfolgers einer Permutation]

Gegeben: Permutation $(a_1, \ldots, a_n)$.
Gesucht: Nachfolger $(b_1, \ldots, b_n)$.

Initialisierung: $j := n-1$.

Schritt 1: Falls $a_j < a_{j+1}$, gehe nach 3.

Schritt 2: $j := j-1$; gehe nach 1.

Schritt 3: $m := j$.
$b_i := a_i$ für $1 \leq i \leq m-1$.
$b_m := \min \{ a_{m+i} \mid a_{m+i} > a_m,\ i \in \{1,\ldots,n-m-1\}\}$.

Schritt 4: Ordne $\{a_m, a_{m+1}, \ldots, a_n\} - \{b_m\}$ der Größe nach aufsteigend in Liste $h_{m+1}, \ldots, h_n$.
$b_{m+i} := h_{m+i}$, $i=1,\ldots,n-m$.

Beispiel:
$n=6$, $a = (1,2,4,6,5,3)$.
Schritte 1 und 2 : Erstes Aufsuchen von '$a_j < a_{j+1}$' von rechts nach links ergibt $j=3$, denn $a_3 = 4 < 6 = a_4$.

Schritt 3 : $m := 3$, $b_1 = 1$, $b_2 = 2$, $b_3 = \min \{ 6, 5 \} = 5$.

Schritt 4 : Ordne $\{ 4, 6, 3 \}$ zu $\{ 3, 4, 6 \}$.
$b_4 = 3$, $b_5 = 4$, $b_6 = 6$.

Ergebnis : $b = (1,2,5,3,4,6)$ ist der Nachfolger von $(1,2,4,6,5,3)$.

Satz (7.10):

Der Algorithmus (7.9) ordnet jeder Permutation ihren Nachfolger zu.

Um dies zu beweisen, genügt es natürlich nicht, auf das obige Beispiel zu verweisen; dieses könnte ja gerade günstig gewählt sein. Ein einzelnes Beispiel kann einen Algorithmus nur falsifizieren (wenn es ein Gegenbeispiel ist), nicht aber verifizieren.
Um z.B. die 'Gleichheit' von $g(x,y) = x+y$ und $f(x,y) = x*y$ zu 'beweisen', genügen auch die unendlich vielen Beispiele x beliebig ungleich 1 und $y = x / (x-1)$ nicht; die Aussage ist falsch.

Beweis (von 7.10):
Sei $a = (a_1, \ldots, a_n)$ abgekürzt; b und c analog. b sei aus a nach Algorithmus (7.9) entstanden. Dann kommt a lexikographisch vor b, weil es wegen Schritt 2 ein m gibt, so daß wegen Schritt 3 gilt: $a_1 = b_1, \ldots, a_{m-1} = b_{m-1}$ und $b_m = \min \{ a_{m+i} \mid a_{m+i} > a_m,\ 1 \leq i \leq n-m-1\}$ und daher $b_m > a_m$.
Nun könnte zwischen a und b lexikographisch noch ein Tupel c kommen. Das geht aber deswegen nicht, weil c mit b an den Stellen $i = 1, \ldots$

, m übereinstimmen muß. Weil aber die restlichen Zahlen der Größe nach geordnet sind (Schritt 4), folgt, daß entweder c = b oder b vor c kommt.

Problem (7.11):

Wie erzeugt man systematisch alle k-Kombinationen von N = {1, ..., n}?

Lösung:
Wieder benutzt man eine lexikographische Anordnung. Man definiert:

$$>a_1, \ldots, a_k< \text{ kommt vor } >b_1, \ldots, b_k<,$$

falls es ein m gibt, $0 \leq m < k$, so daß $a_1 = b_1, \ldots, a_{m-1} = b_{m-1}$ und $a_m < b_m$.

Beispiel:
Die 4-Kombinationen von {1,2,3,4,5,6} in lexikographischer Anordnung sind:

{1,2,3,4}	{1,2,3,5}	{1,2,3,6}	{1,2,4,5}	{1,2,4,6}
{1,2,5,6}	{1,3,4,5}	{1,3,4,6}	{1,3,5,6}	{1,4,5,6}
{2,3,4,5}	{2,3,4,6}	{2,3,5,6}	{2,4,5,6}	{3,4,5,6}.

Man gebe einen Algorithmus zu Problem (7.11) an.

Kapitel 8 : Das Prinzip der Inklusion und Exklusion

Wieviele Zahlen mit der Endziffer 1, mit der Endziffer 2, ... , oder mit der Endziffer 9 kommen unter den ersten 100 Zahlen 1, ... , 100 vor? Man überlegt sich sofort, daß es genau 10 Zahlen mit der Endziffer 0 unter den ersten 100 gibt, und die Antwort auf die Frage also 100 - 10 = 90 ist. Beim Zählen ist es also bisweilen vorteilhaft, statt der zu zählenden Menge ihr Komplement in einer gegebenen Gesamtmenge zu betrachten. Wir wollen dies mit der folgenden Aufgabe zeigen.

Aufgabe (8.1):
Sei S = { 0, 1, 2, 3, 4 } ein Alphabet. Wörter der Länge r sind r-Tupel mit Buchstaben aus S. Bestimme die Anzahl der Wörter der Länge r aus S, die mindestens eine 1, mindestens eine 2 und mindestens eine 3 enthalten.

Lösung:
Wir schreiben S_r für die Menge der Wörter aus S der Länge r und betrachten die Mengen:

$$A_1 = \{ w \mid w \in S_r, \text{ 1 kommt in w nicht vor } \}$$
$$A_2 = \{ w \mid w \in S_r, \text{ 2 kommt in w nicht vor } \}$$
$$A_3 = \{ w \mid w \in S_r. \text{ 3 kommt in w nicht vor } \}.$$

Dann ist

$$A_1 \cup A_2 \cup A_3 = \{ w \mid w \in S_r, \text{ 1 oder 2 oder 3 kommt in w nicht vor } \}$$

genau das Komplement der Menge der Wörter, die wir zählen wollen.

Man kann nun, ohne Satz (8.2) formuliert zu haben, intuitiv die Aufgabe leicht lösen wie folgt:

$$| A_1 | = | A_2 | = | A_3 | = 4^r.$$
$$| A_1 \cap A_2 | = | A_2 \cap A_3 | = | A_1 \cap A_3 | = 3^r.$$
$$| A_1 \cap A_2 \cap A_3 | = 2^r.$$

Also folgt:

$$\begin{aligned} | A_1 \cup A_2 \cup A_3 | = {} & | A_1 | + | A_2 | + | A_3 | \\ & - | A_1 \cap A_2 | - | A_1 \cap A_3 | \\ & - | A_2 \cap A_3 | + | A_1 \cap A_2 \cap A_3 | \\ = {} & 3 * 4^r - 3 * 3^r + 2^r. \end{aligned}$$

Als Lösung ergibt sich also:

$$| A | - | A_1 \cup A_2 \cup A_3 | = 5^r - 3\,(4^r - 3^r) - 2^r.$$

Der wesentliche Schritt dieses Zählprinzips besteht in der Zerlegung der Zahl $| A_1 \cup A_2 \cup A_3 |$. Wir wollen dies nun für beliebige Mengen $A_1, \ldots, A_n$ formulieren.

Satz (8.2): [PRINZIP DER INKLUSION UND EXKLUSION]

Sei $\{ A_i \mid i = 1, \ldots, n \}$ eine Familie von Mengen. Dann gilt:

$$\begin{aligned} | \cup_{i=1\ldots n} A_i | = {} & \Sigma_{i=1\ldots n} | A_i | \\ & - \Sigma_{1 \le i < j \le n} | A_i \cap A_j | \\ & + \ldots + (-1)^{n-1} | A_1 \cap \ldots \cap A_n |. \end{aligned}$$

Beweis:
Induktionsanfang (n=2):
$| A_1 \cup A_2 | = | A_1 | + | A_2 | - |A_1 \cap A_2|$.
Dies gilt, wie leicht zu sehen ist. Zählt man nämlich $| A_1 \cup A_2 |$ als $|A_1| + |A_2|$, so werden die Elemente von $|A_1 \cap A_2|$ doppelt gezählt.

Induktionsschritt (n -> n+1):

$$\begin{aligned} | \cup_{i=1\ldots n+1} A_i | &= | \cup_{i=1\ldots n} A_i \cup A_{n+1} | \\ &= | \cup_{i=1\ldots n} A_i | + | A_{n+1} | - | \cup_{i=1\ldots n} (A_i \cap A_{n+1}) | \\ &= \Sigma_{i=1\ldots n} | A_i | - \Sigma_{1 \le i < j \le n} | A_i \cap A_j | \\ &\quad + \Sigma_{1 \le i < j < k \le n} | A_i \cap A_j \cap A_k | - \ldots \\ &\quad + (-1)^{n-2} \Sigma_{i=1\ldots n} | A_1 \cap \ldots \cap A_{i-1} \cap A_{i+1} \cap \ldots \cap A_n | \\ &\quad + (-1)^{n-1} | A_1 \cap \ldots \cap A_n | \\ &\quad + | A_{n+1} | \\ &\quad - \{ \Sigma_{i=1\ldots n} | A_i \cap A_{n+1} | \\ &\qquad - \Sigma_{1 \le i < j \le n} | A_i \cap A_j \cap A_{n+1} | + \ldots \\ &\qquad + (-1)^{n-1} | A_1 \cap \ldots \cap A_n \cap A_{n+1} | \}, \end{aligned}$$

wobei bei dem in Klammern stehenden Ausdruck die Induktionsvoraussetzung angewendet wurde. Zusammenfassen ergibt nun die Behauptung:

$$
\begin{aligned}
| \, U_{i=1\ldots n+1} A_i \, | &= \Sigma_{i=1\ldots n+1} | A_i | \\
&\quad - \Sigma_{1\leq i<j\leq n} | A_i \cap A_j | \\
&\quad + \Sigma_{1\leq i<j<k\leq n} | A_i \cap A_j \cap A_k | - \ldots \\
&\quad + (-1)^n | A_1 \cap \ldots \cap A_{n+1} |.
\end{aligned}
$$

Das Prinzip der Inklusion und Exklusion hat mit anderen Zählprinzipien gemeinsam, daß die Schwierigkeit weniger in der Theorie liegt, sondern darin, wie man es anwenden kann (konkret in der Konstruktion der Mengen A_i zum Beispiel). Dabei helfen Erfahrung und Ideenreichtum. Da Übungsaufgaben jedoch meist im Zusammenhang mit den Paragraphen gestellt werden, ist klar, welches Zählprinzip oder welche kombinatorische Methode angewendet werden soll.
Insofern sind sie also nicht sehr geeignet, bei der Suche nach Lösungen auch einmal neue Wege zu versuchen.

Kapitel 9 : Teilbarkeit

In den folgenden Kapiteln betrachten wir die Menge $\mathbb{Z}$ der ganzen Zahlen.

9A : Der euklidische Algorithmus

Definition (9.1):

Eine Zahl $a \in \mathbb{Z}$ heißt durch die Zahl $b \in \mathbb{Z}$ TEILBAR, wenn es $c \in \mathbb{Z}$ gibt mit $a = b * c$. Wir schreiben dann $b \mid a$ (b teilt a) bzw. $b \nmid a$ (b teilt nicht a).

Definition (9.2): [Division mit Rest]

Seien $a, b \in \mathbb{Z}$ mit $b > 0$. Dann gibt es eindeutig bestimmte $q, r \in \mathbb{Z}$, so daß

$$a = q * b + r, \text{ mit } 0 \leqslant r < b$$

gilt.

Beweis:
Man betrachte die Folge

$$\ldots, a - 2b, a - b, a, a + b, a + 2b, \ldots$$

die nach beiden Seiten unbegrenzt fortgesetzt ist. In dieser Folge kommen negative Glieder (genügend weit links) und positive Glieder (genügend weit rechts) vor. Nun wähle

$$r = \min \{ a + q * b \mid q \in \mathbb{Z}, a + q * b \geqslant 0 \}.$$

Es gilt:

$$a + (q_0 - 1) * b < r = a + q_0 * b < a + (q_0 + 1) * b.$$

Mit diesem $r = a + q_0 * b$ und $q := -q_0$ gilt $a = q * b + r$, und man sieht, daß $0 \leqslant r < b$ erfüllt ist.
Um die Eindeutigkeit zu beweisen, nehmen wir an, daß es zusätzlich q_1 und r_1 gibt mit $a = q_1 * b + r_1$ und $0 \leqslant r_1 < b$. Sei $r_1 \leqslant r$ angenommen. Dann folgt:

$$
\begin{aligned}
& a = q*b + r = q_1*b + r_1 \\
\Rightarrow\quad & r_1 - r_2 = b * (q - q_1) \\
\Rightarrow\quad & b \mid r - r_1 \text{ und zugleich } 0 \leqslant r - r_1 < b \\
\Rightarrow\quad & r - r_1 = 0 \\
\Rightarrow\quad & r = r_1 \\
\Rightarrow\quad & q = q_1.
\end{aligned}
$$

Definition (9.3):

Eine Zahl $g \in \mathbb{Z} - \{ 0 \}$ heißt ein GEMEINSAMER TEILER von $b, c \in \mathbb{Z}$, wenn $g \mid b$ und $g \mid c$.
g heißt GRÖßTER GEMEINSAMER TEILER von b, c (b und c nicht gleichzeitig 0), wenn gilt:

1) $g \mid b$ und $g \mid c$
2) Aus $d \mid b$ und $d \mid c$ folgt $d \leqslant g$.

Wir schreiben: $g = (b,c)$.

Beispiel:
$2 = (4,6)$.
$2 \neq (12,18)$, da $3 \mid 12$ und $3 \nmid 18$, aber nicht $3 < 2$ gilt.

Satz (9.4):

$g = (b,c) \Rightarrow$ es gibt $x_0, y_0 \in \mathbb{Z}$ mit $g = b\,x_0 + c\,y_0$.

Beweis:
Sei $h := \min \{ b\,x + c\,y \mid x, y \in \mathbb{Z},\ b\,x + c\,y > 0 \}$, also $h = b\,x_0 + c\,y_0$ für bestimmte $x_0, y_0 \in \mathbb{Z}$. Wir wollen $h \mid b$ zeigen und nehmen an, daß $h \nmid b$ gilt. Aus Satz (9.2) folgt:

$$b = q\,h + r, \text{ mit } 0 \leqslant r < h.$$

Daher gilt
$$
\begin{aligned}
r = b - h\,q &= b - q\,(b\,x_0 + c\,y_0) \\
&= b\,(1 - q\,x_0) + c\,(-q\,y_0).
\end{aligned}
$$

Man sieht, daß $r \in \{ b\,x + c\,y \mid x, y \in \mathbb{Z} \}$ und hat $0 < r < h$. Dies steht aber im Widerspruch zur Definition von h.
Wir haben also $h \mid b$ gezeigt, und ebenso zeigt man $h \mid c$.

Da $g = (b,c)$, gilt $b = g\,b_1$ und $c = g\,c_1$ und damit schreiben wir nun

$$
\begin{aligned}
h &= b\,x_0 + c\,y_0 \\
&= g\,b_1\,x_0 + g\,c_1\,y_0 \\
&= g\,(b_1\,x_0 + c_1\,y_0).
\end{aligned}
$$

Also gilt $g \mid h$, daher $g \leqslant h$. Aus $g = (b, c)$ folgt dann $g = h$, d.h. wir haben den größten gemeinsamen Teiler g ($= h$) dargestellt in der Form

$$h = b\,x_0 + c\,y_0, \text{ mit } x_0, y_0 \in \mathbb{Z}.$$

Satz (9.5): [EUKLIDISCHER ALGORITHMUS]

Seien $b, c \in \mathbb{Z}$ mit $c > 0$. Wir führen wiederholt die Division mit Rest nach Satz (9.2) durch, um eine Folge von Gleichungen zu erhalten:

$$\begin{aligned}
b &= c\,q_1 + r_1 && 0 < r_1 < c \\
c &= r_1\,q_2 + r_2 && 0 < r_2 < r_1 \\
r_1 &= r_2\,q_3 + r_3 && 0 < r_3 < r_2 \\
&\cdots \\
r_{j-2} &= r_{j-1}\,q_j + r_j && 0 < r_j < r_{j-1} \\
r_{j-1} &= r_j\,q_{j+1}
\end{aligned}$$

Der größte gemeinsame Teiler von b, c ergibt sich als der letzte von Null verschiedene Rest r_j in diesem Algorithmus.

Beweis:
Der Algorithmus stoppt, wenn die Division ohne Rest aufgeht, d.h. der Rest $r_{j+1} = 0$ ist. Dies muß nach endlich vielen Schritten eintreten, da die Reste $r_j < r_{j-1} < \ldots < r_2 < r_1 < c$ immer kleiner werden und positiv sind.
Wir wollen nun zeigen, daß $r_j = (b,c)$.
Wir zeigen zunächst, daß für $d = (b,c)$ und $e = (b-cq_1,c)$ schon folgt, daß $d = e$.
Gilt nämlich $d \mid b$ und $d \mid c$, so gilt auch $d \mid b-cq_1$, also $d \mid e$, da $e = (b-cq_1,c)$.
Andererseits folgt aus $e \mid c$ und $e \mid b-cq_1$, daß $e \mid b$, also $e \mid d$.
Also gilt $d = e$.
Wir haben also

$$(b,c) = (b-cq_1,c) = (r_1,c).$$

Wir fahren auf diese Weise fort und erhalten:

$$(r_1,c) = (r_1,c-r_1q_2) = (r_1,r_2).$$

Insgesamt erhalten wir:

$$(b,c) = (r_1,c) = (r_1,r_2) = \ldots = (r_{j-1},r_j) = r_j.$$

Nach Satz (9.4) läßt sich r_j darstellen als $r_j = b\,x_0 + c\,y_0$. Man findet x_0, y_0, indem man r_{j-1} aus der vorletzten und der 3.letzten Gleichung eliminiert, dann r_{j-2} aus der 3.letzten und 4.letzten, usw.

4B : Die Lame'sche Schranke

Algorithmische Formulierung des Euklidischen Algorithmus (9.5):

Für $b,c \in \mathbb{Z}$, $c \neq 0$, liefert Division mit Rest Zahlen $q, r \in \mathbb{Z}$ mit

$$b = q * c + r \qquad , 0 \leqslant r < |c|.$$

Wir setzen

$$b \text{ div } c := q$$
$$b \text{ mod } c := r,$$

so daß also gilt:

$$b = (b \text{ div } c) * c + b \text{ mod } c,$$

wobei $0 \leqslant b \text{ mod } c < |c|$.

In PASCAL-Notation formulieren wir den Euklidischen Algorithmus wie folgt:

```
Input : b, c ∈ ℤ
Output: ggT (b,c)

begin
   B  :=  |b|;
   C  :=  |c|;
   while C > 0 do
       begin R := B mod C;
             B := C;
             C := R
       end;
   ggT := B
end.
```

Verifikation:
Wegen ggT(b,0) = |b| und (für $c \neq 0$) ggT(b,c) = ggT(c,b mod c) leistet der Algorithmus das Gewünschte. Weil der Operator mod Zahlen verkleinert (B mod C < C), ist das neue C in der while-Bedingung kleiner als das alte und der Algorithmus terminiert also.

Die Lame'sche Schranke

Wir wollen die Anzahl der arithmetischen Operationen abschätzen, die man zur Berechnung des ggT mit obigem Algorithmus benötigt.
Wegen

$$B = (B \text{ div } C) * C + B \text{ mod } C$$

ist, falls C nur wenig oberhalb von B/2 liegt, B div C = 1 und B mod C relativ groß, $C_{neu} := B \bmod C_{alt}$ sinkt also nur langsam. Ein Beispiel für diesen ungünstigen Fall:

b = 21, c = 13

$$\begin{aligned} 21 &= 1*13 + 8 \\ 13 &= 1* 8 + 5 \\ 8 &= 1* 5 + 3 \\ 5 &= 1* 3 + 2 \\ 3 &= 1* 2 + 1 \\ 2 &= 2* 1 + 0 \end{aligned}$$

Jeder Quotient, der letzte natürlich ausgenommen, ist hier 1, d.h. es ist immer B div C = 1.

Das gleiche Phänomen tritt bei den Zahlen

$$\begin{aligned} b &= 21+13,\ c = 21 \\ b &= 34+21,\ c = 34 \end{aligned}$$

auf. Dies führt uns zur Definition der Fibonaccizahlen durch

$$\begin{aligned} &F_1 := 1,\ F_2 := 2 \\ &F_{n+1} := F_n + F_{n-1},\ n \geqslant 2. \end{aligned}$$

Es ist also $F_3 = 3$, $F_4 = 5$, $F_5 = 8$, $F_6 = 13$, $F_7 = 21$, $F_8 = 34$ ($F_0 = 1$ wird hier nicht benötigt), und die Wahl von $b:=F_n$ und $c:=F_{n-1}$ führte in den obigen Beispielen auf worst-case Abschätzungen für die Anzahl der benötigten arithmetischen Operationen.

Die Frage, wieviele k-stellige Fibonaccizahlen es gibt, beantwortet Lame in seiner Arbeit von 1844 in folgendem Satz.

Satz (9.6):

Es gibt mindestens 4 und höchstens 5 k-stellige Fibonaccizahlen, wobei $k \in \mathbb{N}$ beliebig, aber fest ist.

Beweis:

Der rechentechnisch nicht interessante Beweis mittels Induktion über k benutzt Abschätzungen für die Größe von F_{n+i}, $0 \leqslant i \leqslant 5$, die durch einfache Additionen aus der Definitionsformel der Fibonaccizahlen folgen.

Satz (9.7):

Ist F_n die n-te Fibonaccizahl und ist k die Anzahl der Dezimalstellen von F_n (wir schreiben: dez $(F_n) = k$), so gilt $n \leqslant 5k$.

Beweis:

Induktion über k:

k=1 : $dez(F_n) = 1 \Rightarrow 1 \leq n \leq 5$ (da $F_5 = 8$, $F_6 = 13$)
$\Rightarrow n \leq 5 = 5k.$

Sei $k \geq 1$, der Satz gelte für k:

$dez(F_n) = k+1 \Rightarrow dez(F_{n-5}) \leq k$ (sonst gäbe es ≥ 6 Fibonacci-zahlen mit dez( ) = k+1, Widerspruch zu Satz (9.6)).

$\Rightarrow n-5 \leq 5k$ (nach Ind. Vor.)
$\Rightarrow n \leq 5k + 5 = 5(k+1).$

Satz (9.8):

Seien $b,c \in \mathbb{N}$.
Es gelte:

$b = q*c + r$, mit $0 \leq r < c$
$F_n \geq b > c \geq F_{n-1}$, mit $r \geq F_{n-2}$,

so folgt:

$F_n = b,\ F_{n-1} = c,\ F_{n-2} = r.$

Beweis:

Wegen $b \geq c$ ist $q \geq 1$. Also gilt

$$F_n \geq b = q*c + r \geq c + r \geq F_{n-1} + F_{n-2} = F_n.$$

Also steht überall '=' und die Behauptung ist bewiesen.

Wir bezeichnen nun mit w(b,c) die Zahl, die angibt, wie oft die Anweisung R := B mod C in der while-Schleife des Algorithmus bei Eingabe von b und c ausgeführt wird.

Satz (9.9):

Seien $b, c \in \mathbb{N}$ und $F_n \leq c < F_{n+1} \Rightarrow w(b,c) \leq n.$

Beweis:

Wir schreiben w(c) für w(b,c), da diese Zahl, wie jetzt bewiesen wird, nur von c abhängt.

Induktion über c:

Ind. Anfang:
$c = 1 \Rightarrow n = 1$ und $w(c) = 1.$

Ind. Vor.:
Sei $F_i \leqslant r < F_{i+1}$ => $w(r) \leqslant i$. (mit r := b mod c)

Ind. Beh.:
Sei $F_n \leqslant c < F_{n+1}$ => $w(c) \leqslant n$.

Ind. Schritt:
c > 1. Setze r := b mod c

=> ggT(b,c) = ggT(c,r)
=> w(c) = w(r) + 1 (1)

Sei $F_i \leqslant r < F_{i+1}$
=> $w(r) \leqslant i$, nach Ind.Vor
=> $F_i \leqslant r < c < F_{n+1}$, Wahl von r und c,(2)

1. Fall: i < n.

=> $w(c) = w(r) + 1$
$\leqslant i + 1$, nach Ind. Vor.
$\leqslant n-1 + 1$, da i < n-1
$= n$,
also $w(c) \leqslant n$.

2. Fall: i = n.

=> $F_n \leqslant r < c < F_{n+1}$, nach Ind. Vor. und (2)
Setze s := c mod r.
=> w(r) = w(s) + 1
=> w(c) = w(s) + 2.

i) Wäre $F_{n-1} \leqslant s$
=> $s = F_{n-1}$, $r = F_n$, $c = F_{n+1}$, nach (9.8),
im Widerspruch zu '$c<F_{n+1}$'.

=> ii) $s < F_{n-1}$
=> $w(s) \leqslant n-2$, nach Ind. Vor.
=> $w(c) = w(s) + 2 \leqslant n$.

Um w(c) nach oben abzuschätzen, braucht man aber die Fibonaccizahlen aus (9.9) nicht wirklich. Es gilt nämlich:

Satz (9.10): [Lame'sche Schranke]

Seien b, c ∈ ℤ => $w(b,c) \leqslant 5$ dez(c).

Beweis:

Sei $F_n \leqslant |c| < F_{n+1}$
=> $dez(F_n) \leqslant dez(c)$
=> $w(b,c) \leqslant n$, nach (9.9),

und wegen $n \leqslant 5\ \mathrm{dez}(F_n)$ [Satz (9.7)] folgt insgesamt:

$$w(b,c) \leqslant n \leqslant 5\ \mathrm{dez}(F_n) \leqslant 5\ \mathrm{dez}(c).$$

Bewertung der Abschätzungen:
Die Abschätzung für w(b,c) aus (9.10) ist sehr handlich. Der Preis für diese Handlichkeit ist, daß sie nicht so präzise, nicht so scharf wie die bestmögliche Abschätzung aus Satz (9.9) ist. Die Verschlechterung fällt jedoch nicht sehr ins Gewicht.

Zur Illustration nehmen wir die größte, fünfzigstellige Fibonaccizahl

$$F_{239} = 64\ 202\ 014\ 863\ 723\ 094\ 126\ 901\ 777\ 428\ 873\ 111\ 802\ 307\ 548\ 623\ 680$$

Ist also dez(c) = 50, so ist $w(c) \leqslant 239$ nach (9.9) und $w(c) \leqslant 250$ aufgrund der Lame'schen Schranke aus (9.10).

9C: Verallgemeinerung des Euklidischen Algorithmus

Wir wollen den im Satz (9.5) steckenden Algorithmus vereinfachen und verallgemeinern. Dabei stellen wir uns das Problem, nicht nur von zwei vorgegebenen Zahlen den größten gemeinsamen Teiler zu finden, sondern von n Zahlen $a_1, \ldots, a_n$. Außerdem wollen wir zugleich auch eine Darstellung von $g = (a_1, \ldots, a_n)$ in der Form

$$g = a_1 x_1 + a_2 x_2 + \ldots + a_n x_n$$

haben.

Beispiel:
$g = (3, 4, 6) = 1$ und $1 = (-1)*3 + (1)*4 + 0*6$.

Ohne Beschränkung der Allgemeinheit setzen wir die a_i als nicht negative Zahlen voraus.

Algorithmus (9.11): [Größter gemeinsamer Teiler von n Zahlen]

Gegeben: n Zahlen $a_1, \ldots, a_n$; $a_i > 0$.
Gesucht: $g = (a_1, \ldots, a_n)$.

Betrachte folgendes aus n Zeilen und n+1 Spalten bestehendes Schema:

$$\begin{array}{lcccccc} a_1 & 1 & 0 & 0 & \ldots & 0 \\ a_2 & 0 & 1 & 0 & \ldots & 0 \\ a_3 & 0 & 0 & 1 & \ldots & 0 \end{array}$$

usw.
a_n 0 0 0 ... 1

Die Einträge in der 1. Spalte (a_i) nenne 'Führer'.

Schritt 1 : Suche Zeile mit kleinstem Führer ungleich 0 und nenne sie 'Operator'.

Schritt 2 : Wähle eine beliebige andere Zeile mit einem Führer ungleich 0 und nenne sie 'Operand'.

Schritt 3 : Division ohne (!) Rest: q = Führer(Operand)/Führer(Operator).

Schritt 4 : Operand:=Operand - q * Operator, als neue Zeile.
Streiche den alten Operanden.
Zurück zu Schritt 1.

STOP, wenn als einzige Zeile g x_1 ... x_n übrigbleibt, mit g ungleich 0.

Ergebnis : $g = a_1 x_1 + \ldots + a_n x_n$.
$g = (a_1, \ldots, a_n)$.

Beispiel:
n=3, (a_1, a_2, a_3)= (3, 4, 6).

Schema:

	3	1	0	0	Operator
	4	0	1	0	
	6	0	0	1	Operand

q = 6 div 3 = 2. Neue 3. Zeile: 3. Zeile - 2 * 1. Zeile

	3	1	0	0	Operator
	4	0	1	0	Operand
	0	-2	0	1	

q = 4 div 3 = 1. Neue 2. Zeile: 2. Zeile - 1 * 1. Zeile

	3	1	0	0	Operand
	1	-1	1	0	Operator
	0	-2	0	1	

q = 3 div 1 = 3. Neue 1. Zeile: 1. Zeile - 3 * 2. Zeile

	0	4	-3	0
-->	1	-1	1	0
	0	-2	0	1

STOP. Ergebnis: (3,4,6) = 1 = 3*(-1) + 4*(1) + 6*(0).

Satz (9.12): [Verifikation von (9.11)]

Der Algorithmus (9.11) leistet das angegebene.

Beweis:
Nach Schritt 4 wird jeweils der Führer einer Zeile verkleinert und bleibt dabei ≥ 0. Nach endlich vielen Schritten (weniger als $\Sigma_{i=1..n} a_i$ viele) sind die Führer von n-1 Zeilen 0. Also terminiert der Algorithmus.
Jede Zeilenumwandlung nach Schritt 4 erhält den ggT der Führer, denn es gilt:

$$(b_1,\ldots,b_i,\ldots,b_j,\ldots,b_n) = (b_1,\ldots,b_i-qb_j,\ldots,b_j,\ldots,b_n),$$

wie wir schon im Beweis von (9.5) gesehen haben.
Ist es schließlich erreicht, daß alle Führer bis auf einen gleich 0 sind, ist damit klar, daß dieser der ggT ist.

Bemerkung (9.13):

Dem Leser sei ergänzend mitgeteilt, daß sich dieser Algorithmus in leicht veränderter Form für Polynomringe oder für den Chinesischen Restsatz benutzen läßt.

Kapitel 10 : Primzahlen und Zahldarstellungen

Definition (10.1):

Eine Zahl $p \in \mathbb{N}, p>1,$ heißt PRIMZAHL, wenn es keinen Teiler d von p gibt mit $1 < d < p$.

Hilfssatz (10.2):

Ist p Primzahl und gilt $p \mid a*b$, so ist $p \mid a$ oder $p \mid b$.

Beweis:
Gilt $p \not\mid a$, so ist $(a,p) = 1$ und es folgt $(ab,pb) = (a,p) * b = b$.
Da $p \mid ab$ und $p \mid pb$, folgt $p \mid b$, nach der Definition des ggT.

Satz (10.3):

Jedes $n \in \mathbb{N}$, $n > 1$, läßt sich auf eindeutige Weise darstellen als Produkt von Primzahlpotenzen:

$$n = p_1^{\alpha_1} * \ldots * p_r^{\alpha_r},$$

wobei $p_1, \ldots, p_r$ Primzahlen und $\alpha_i \in \mathbb{N}$.

Beweis:
Ist n Primzahl, so ist nichts zu zeigen.
Sei n keine Primzahl, also $n = n_1 n_2$ mit $1 < n_i < n$, $i = 1, 2$. Nach Induktionsvoraussetzung (Induktionsanfang: $n = 4 = 2 * 2$) kann man n_1 und n_2 in ein Produkt von Primzahlpotenzen zerlegen, also auch n.
Eindeutigkeit: Wir nehmen an, daß wir zwei verschiedene Zerlegungen haben,

$$n = p_1^{\alpha_1} * \ldots * p_r^{\alpha_r} = q_1^{\beta_1} * \ldots * q_s^{\beta_s},$$

wobei die Primzahlen p_i und q_i geordnet seien durch

$$p_1 > \ldots > p_r \text{ und } q_1 > \ldots > q_s.$$

Nun gilt $p_1 \mid n = q_1^{\beta_1} * \ldots * q_s^{\beta_s}$ und nach Hilfssatz (10.2) teilt p_1 mindestens einen Faktor q_i, also $p_1 = q_i$, da q_i ja Primzahl ist.
Andererseits gilt $q_1 \mid n = \Pi_{i=1..r} p_i^{\alpha_i}$ und mit (10.2) folgt $q_1 \mid p_j$, also $q_1 = p_j$ für ein p_j.
Wir haben also $q_1 \geqslant q_i = p_1$ und $q_1 = p_j \leqslant p_1$, woraus folgt, daß $i = j$

= 1 gilt. Wir kürzen nun n durch $p_1 = q_1$, und aus der Induktionsvoraussetzung für

$$n_1 = p_1^{\alpha_1 - 1} * p_2^{\alpha_2} * \ldots * p_r^{\alpha_r} = q_1^{\beta_1 - 1} * q_2^{\beta_2} * \ldots * q_s^{\beta_s}$$

folgt die Behauptung.

Satz (10.4): [Euklid]

Es gibt unendlich viele Primzahlen.

Beweis:
Angenommen, es gibt nur endlich viele Primzahlen $p_1, \ldots, p_r$. Dann bilden wir $n = 1 + \Pi_{i=1..r} p_i$ und sehen, daß n durch keine der Primzahlen p_i teilbar ist. Nach (10.3) ist also n entweder eine Primzahl oder es gibt eine weitere Primzahl p_{r+1} mit $p_{r+1} \mid n$.

Beispiel (10.5):
Sei $A = \{ a_1, \ldots, a_k \}$ ein Alphabet und $h : A \longrightarrow \mathbb{N}$ eine Funktion mit $h(a_i) = i$, $i = 1, \ldots, k$.
Wir betrachten die geordnete Folge $p_1 = 2 < p_2 = 3 < p_3 = 5 \ldots$ der Primzahlen und definieren eine Abbildung $f : A^+ \longrightarrow \mathbb{N}$ für jedes Wort $w \in A^+$ mit $w = y_1 y_2 \ldots y_n$, $y_i \in A$, durch

$$f(w) = p_1^{h(y_1)} * \ldots * p_n^{h(y_n)} .$$

Nach Satz (10.3) ist f eineindeutige Abbildung, d. h. man kann aus den Zahlen (Bilder von f) die Wörter (Urbilder) eindeutig zurückgewinnen, mithin also jeden Text (auch ganze Bücher) durch eine einzige Zahl eindeutig festlegen.

Beispiel: Zur Zahl 2100 gehört das Wort 'BABA', denn

$$2100 = 2^2 \, 3^1 \, 5^2 \, 7^1,$$

und

$$h(B = a_2) = 2, \; h(A = a_1) = 1.$$

Nicht nur die Darstellung der natürlichen Zahlen als Produkte verdient unser Interesse, auch die allgemein übliche Dezimalschreibweise soll hier näher untersucht werden.

So bedeutet ja die Zahl 2052515 dasselbe wie

$$2*10^6 + 0*10^5 + 5*10^4 + 2*10^3 + 5*10^2 + 1*10^1 + 5*10^0.$$

Es ist bekannt, daß man außer dem Dezimalsystem (g=10) auch andere Systeme wählen kann, wie z. B. das dyadische (g=2) oder das hexadezimale (g=16). Für alle g > 1 gilt folgender Satz:

Satz (10.6):

Sei $g > 1$. Jedes $n \in \mathbb{N}$ läßt sich auf eindeutige Weise darstellen durch

$$n = n_0 + n_1 g + \ldots + n_s g_s,$$

mit $0 \leqslant n_i < g$, $i = 0, \ldots, s$, $s \geqslant 0$.

Beweis:

Die Menge $\{ t \mid t \in \mathbb{N}_0, n < g^{t+1} \}$ hat (wie jede Teilmenge von $\mathbb{N}$) ein kleinstes Element s. Daher gilt $g^s \leqslant n < g^{s+1}$.
Nun sind nach Satz (9.2) (Division mit Rest) die Zahlen $n_0, \ldots, n_r$ eindeutig bestimmt:

$$\begin{aligned} n &= g^s n_s + r_s. \\ r_s &= g^{s-1} n_{s-1} + r_{s-1}. \\ &\text{usw.} \\ r_{s-s+2} &= g\, n_1 + n_0. \end{aligned}$$

Daraus ergibt sich sofort die Darstellung

$$n = g^s n_s + r_s = g^s n_s + g^{s-1} n_{s-1} = \ldots$$
$$= n_s g^s + \ldots + n_1 g + n_0.$$

Die Darstellung ist eindeutig, denn für eine andere Darstellung

$$n = m_t g^t + \ldots + m_1 g + m_0, \quad 0 \leqslant m_i < g, \; i = 1,..,t$$

folgt wegen $g^s \leqslant n < g^{s+1}$, daß s = t sein muß. Ferner folgt aus $(n_s - m_s)g^s + \ldots + (n_1 - m_1)g + (n_0 - m_0) = 0$, daß $n_0 = m_0$ und nacheinander $n_i = m_i$ gilt für i= 1,..., s.

Für g=2 ergibt sich:

Folgerung (10.7):

Jedes $n \in \mathbb{N}$ läßt sich auf genau eine Weise darstellen durch

$$n = n_s 2^s + n_{s-1} 2^{s-1} + \ldots + n_1 2^1 + n_0,$$

mit $n_i \in \{ 0, 1 \}$, $i = 0, \ldots, s$; $s \geq 0$.

Beispiel:

Es gilt $6 = 1*2^2 + 1*2^1 + 0$, man schreibt: $6_{10} = 110_2$.

Kapitel 11 : Kongruenzen

Definition (11.1):

Sei $m \in \mathbb{N}$, $a, b \in \mathbb{Z}$. Gilt $m \mid a-b$, so schreiben wir

$$a = b \bmod m$$

und lesen: a ist KONGRUENT zu b MODULO M.

Beispiel [vgl. Beispiel zu Def. (2.6)]:

$4 = 0 \bmod 4$ $\quad$ $7 = 3 \bmod 4$
$5 = 1 \bmod 4$ $\quad$ $8 = 0 \bmod 4$
$6 = 2 \bmod 4$ $\quad$ $-1 = 3 \bmod 4$

Definition (11.2):

Ist $a = b \bmod m$, so heißt b ein REST von a MODULO M. Eine Menge von Zahlen $r_1, \ldots, r_m$ heißt ein VOLLSTÄNDIGES RESTSYSTEM MODULO M, wenn für jedes $a \in \mathbb{Z}$ genau ein r_i existiert mit $z = r_i$ modulo m.

Beispiel:
$\{ r_1 = 0, r_2 = 1, r_3 = 2, r_4 = 3 \}$ ist ein vollständiges Restsystem modulo 4, denn für $a \in \mathbb{Z}$ gilt nach der Division mit Rest, daß $z = r \bmod 4$ mit $0 \leq r < 4$, also $r \in \{ 1, 2, 3, 4 \}$.

Definition (11.3):

Eine Zahl a heißt RELATIV PRIM zur Zahl b, wenn $(a,b) = 1$.

Statt 'relativ prim' sagt man auch TEILERFREMD.

Hilfssatz (11.4):

Ist $ax = ay \bmod m$ und $(a,m) = 1$, so folgt $x = y \bmod m$.

Beweis:
Aus $ax = ay \bmod m$ folgt nach Definition $m \mid ax-ay = a(x-y)$. Da $(a,m) = 1$, folgt also $m \mid x-y$.

Definition (11.5):

Ein PRIMES RESTSYSTEM MODULO M ist eine Menge von ganzen Zahlen r_i mit $(r_i,m) = 1$ und $r_i \not\equiv r_j \bmod m$ für $i \neq j$, so daß jedes $z \in \mathbb{Z}$ mit $(z,m) = 1$ genau zu einem r_i modulo m kongruent ist.

Beispiel:
{ 1, 3 } ist ein primes Restsystem modulo 4.

Definition (11.6):

$\varphi(m)$ ist die Anzahl der natürlichen Zahlen n mit $n < m$, die zu m relativ prim sind. Die Funktion φ heißt EULERSCHE φ-FUNKTION.

Beispiel (11.7):
Ist $m = p$ eine Primzahl, so ist $\varphi(p) = p-1$, da ja 1, 2, ..., p-1 relativ prim zu p sind.

Satz (11.8):

Ist $(a,m) = 1$ so gilt $a^{\varphi(m)} = 1 \bmod m$.

Beweis:
Sei $r_1, \ldots, r_{\varphi(m)}$ ein primes Restsystem modulo m [$\varphi(m)$ Elemente nach Def (11.5) und (11.6)]. Dann ist auch $ar_1, \ldots, ar_{\varphi(m)}$ ein primes Restsystem modulo m, denn zu jedem r_i gibt es genau ein ar_j mit $r_i = ar_j \bmod m$. Das bedeutet, daß die Zahlen $ar_1, \ldots, ar_{\varphi(m)}$ genau die Reste modulo m von $r_1, \ldots, r_{\varphi(m)}$ darstellen, nur nicht notwendig in derselben Reihenfolge.
Bilden wir nun das Produkt, so gilt:

$$\prod_{j=1..\varphi(m)} (ar_j) = \prod_{i=1..\varphi(m)} r_i \bmod m.$$

Daraus folgt:

$$a^{\varphi(m)} \prod_{j=1..\varphi(m)} r_j \equiv \prod_{i=1..\varphi(m)} r_i \bmod m.$$

Nun gilt $(r_j,m) = 1$, so daß wir Hilfssatz (11.4) anwenden können, um die r_j zu kürzen. Also folgt $a^{\varphi(m)} = 1 \bmod m$.

Satz (11.9): [KLEINER FERMAT]

Ist p Primzahl und $p \nmid a$, so folgt $a^{p-1} = 1 \bmod p$ bzw. $a^p = a \bmod p$.

Beweis:
Folgt sofort aus (11.8) und Bemerkung (11.7).

Kapitel 12 : Über die Komplexität von Algorithmen

Jeder hat schon die Erfahrung gemacht, daß man durch geschicktes Rechnen Zeit sparen kann. Bekannt ist die Geschichte des Grundschullehrers, der seiner Klasse die Aufgabe stellte, alle Zahlen von 1 bis 100 zu addieren. Er glaubte, eine gute halbe Stunde in Ruhe seine Zeitung lesen zu können. Nun soll sich aber Gauß unter seinen Schülern befunden haben, der nicht etwa 1 + 2 + ... + 100 addierte, sondern (0+100) + (1+99) ... + (49+51) + 50 zu 50*100 + 50 zusammenfaßte; er hatte also nur 1 Multiplikation und 1 Addition im Vergleich zu 99 Additionen durchzuführen.

Der Zeitaufwand eines Algorithmus hängt natürlich von den Eingabedaten und der Anzahl der Rechenschritte ab. So ist es offensichtlich, daß der Unterschied zwischen den beiden Verfahren immer deutlicher geworden wäre, wenn die Aufgabe gelautet hätte, die ersten 1000 oder gar 10000 Zahlen zu addieren.

Wir gehen in unserem idealisierten Modell von der Annahme aus, daß bei einem gegebenen Algorithmus A mit einer Eingabe einer Zahl n der

Zeitaufwand durch eine Funktion $f:\mathbb{N} \longrightarrow \mathbb{R}^+$, $n \longrightarrow f(n)$ beschrieben wird.
Bei derzeitigem Stand der Computertechnik gibt folgende Tabelle einen Eindruck von der zu erwartenden Rechenzeit:

	n=10	n=20	n=30
$f(n) = n$	0.00001	0.00002	0.00003
$f(n) = n^2$	0.0001	0.0004	0.0009
$f(n) = n^3$	0.001	0.008	0.027
$f(n) = 2^n$	0.001	1.0	17.9 min
$f(n) = 3^n$	0.059	58 min	6.5 Jahre

Zahlen ohne Zeiteinheit: Sekunden.

Man beobachtet, daß die Rechenzeit bei einer exponentiellen Funktion wesentlich schneller steigt.

Um die Beurteilung der Komplexität eines Algorithmus mathematisch faßbar zu machen, benutzt man folgende Definition.

Definition (12.1):

Seien $f, g : \mathbb{N}_0 \longrightarrow \mathbb{R}^+$ gegeben. Die Funktion $f(n)$ ist $O(G(N))$, d. h. 'ist in der Größenordnung von $g(n)$', wenn es eine Konstante $c \in \mathbb{R}$ gibt, so daß $f(n) \leqslant c\, g(n)$ für alle $n \geqslant n_0$.

Beispiel (12.2):

Ist f ein Polynom, $f(x) = a_k x^k + \ldots + a_1 x + a_0$, so läßt sich f stets mit der höchsten auftretenden Potenz abschätzen:
Es ist

$$\begin{aligned} f(n) &= a_k x^k + \ldots + a_1 n + a_0 \\ &\leqslant \Sigma_{i=0..k}\, a_i n^k \\ &= c\, n^k \text{ mit } c = \Sigma_{i=0..k}\, a_i. \end{aligned}$$

Also gilt: $f(n)$ ist $O(n^k)$.

Natürlich könnte man $f(n)$ auch durch eine höhere Potenz abschätzen und sagen $f(n) = O(n^{k+1})$; es ist aber von Interesse, die Abschätzung möglichst genau zu treffen.

Definition (12.3):

Seien $f, g: \mathbb{N}_0 \longrightarrow \mathbb{R}^+$ monoton wachsende Funktionen gegeben. Die Funktion f WÄCHST ASYMPTOTISCH nicht SCHNELLER als die Funktion g, wenn

$$\lim_{x \to \infty} f(x)/g(x) = c \quad \text{für eine Konstante } c > 0$$

gilt.

Beispiel (12.4):

Das Polynom $f(n) = a_k n^k + \ldots + a_0$ wächst asymptotisch nicht schneller als $g(n) = n^k$, denn

$$\lim_{n \to \infty} (a_k n^k + \ldots + a_0)/n^k$$

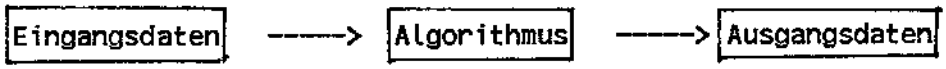
$$= \lim_{n \to \infty} (a_k + a_{k-1}/n + \ldots + a_0/n^k) = a_k.$$

Definition (12.5):

Ein Algorithmus heißt POLYNOMIAL, wenn seine Rechenzeit durch ein Polynom abgeschätzt werden kann. Andernfalls heißt der Algorithmus EXPONENTIELL.

Bemerkung (12.6):
Algorithmen von exponentieller Komplexität gelten als sehr schlechte, undurchführbare Algorithmen. Eine Rechenzeit beispielsweise von Jahren ist unrealistisch. Die obige Tabelle zeigt, daß man in diesem Sinne einen Trennungsstrich zwischen Algorithmen von polynomialer und exponentieller Komplexität ziehen sollte.

Bemerkung (12.7):
Wir werden im folgenden häufig den Zeitaufwand von Algorithmen nach ganz verschiedenen Eingabedaten wie zum Beispiel Größe von Zahlen, Elemente einer Menge und nach ganz verschiedenen Operationen wie Anzahl der Additionen, Multiplikationen oder Vergleichen berechnen. Dem liegt die Vorstellung zugrunde, einen Algorithmus als 'BLACK BOX' anzusehen:



Dabei sind Eingangsdaten und Ausgangsdaten in Wörtern des Alphabets {0,1} gegeben. Der Algorithmus kann als eine Folge von Bitoperationen wie 'oder', 'und', 'nicht' usw. aufgefaßt werden. Die Komplexität eines Algorithmus ist nur auf die Länge dieser Eingangsdaten und auf die Anzahl dieser Operationen bezogen.
Unsere Ergebnisse über die Komplexität von Algorithmen lassen sich auf dieses Modell übertragen, wobei Aussagen der Form O(g(n)) unverändert erhalten bleiben.

Kapitel 13 : Erzeugung von Primzahlen

13A : Primzahltests

Seit dem Altertum schon ist uns eine Methode bekannt, mit deren Hilfe alle Primzahlen unterhalb einer beliebig vorgegebenen Schranke gefunden werden, das SIEB DES ERATOSTHENES. Mit dieser Methode wollen wir nun alle Primzahlen ≤ 20 finden. Dazu schreiben wir zunachst alle Zahlen von 1 bis 20 in eine Liste:

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Streiche nun alle Zahlen n mit 2 | n und 2 < n:

1 2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~ 19 ~~20~~

p:=nächstgrößere nicht-gestrichene Zahl; streiche alle n mit p | n und p < n (hier: p = 3):

1 2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~

Setze diesen Prozeß so lange fort, bis die Liste abgearbeitet ist, in unserem Fall geht man die Liste mit p = 5, 7, 11, 13, 17 und 19 durch und erhält, daß alle nicht-gestrichenen Zahlen >1 Primzahlen sind.

Mit diesem Verfahren läßt sich natürlich auch feststellen, ob eine gegebene Zahl eine Primzahl ist oder nicht.
Eine Verbesserung des Verfahrens bedeutet folgender Satz.

Satz (13.1):

Ist $n \in \mathbb{N}$ keine Primzahl, so gibt es eine Primzahl p mit p | n und $p \leq \sqrt{n}$.

Beweis:
n keine Primzahl => n = a*b (Primzahlzerlegung). Wäre a > sqrt(n) und b > sqrt(n), so würde folgen a*b > sqrt(n)*sqrt(n) = n.

Der Satz besagt, daß man im Sieb des Eratosthenes nur die Vielfachen von Primzahlen $\leq$ sqrt(n) herausstreichen muß, also die Zahlen unterhalb n nur bis sqrt(n) durchzugehen braucht.
Dennoch wird man feststellen, daß der Algorithmus einen großen Aufwand an Rechenschritten verlangt. Wir wollen deshalb einen Algorithmus vorstellen, der von sehr guter Komplexität ist, aber den Nachteil hat, daß er nicht mit Sicherheit entscheidet, ob eine

vorgelegte Zahl n eine Primzahl ist, sondern nur mit hoher Wahrscheinlichkeit.

Definition (13.2):

Sei p' ungerade und $p'-1 = 2^s t$ mit $(2,t) = 1$ (immer mögliche Darstellung). p' heißt STARKE PSEUDOPRIMZAHL zur Basis b, wenn entweder

$$b^t = 1 \bmod p'$$

oder ein r mit $0 \leq r < s$ existiert, so daß

$$b^{t*2^r} = -1 \bmod p'.$$

Lemma (13.3):

Jede Primzahl p ist eine starte Pseudoprimzahl zur Basis b für b = 1,..., p-1.

Beweis:
Da p Primzahl ist und $p \nmid b$ (wegen $b < p$), gilt nach dem 'Kleinen Fermat' [Satz 11.9], daß $b^{p-1} = 1 \bmod p$.

Da p ungerade ist, ist p-1 gerade und also in der Form $p-1 = 2^s t$ für ein s mit $(t,2) = 1$ darstellbar. Also haben wir eingesetzt:

$$b^{t*2^s} = 1 \bmod p.$$

Angenommen, daß $b^t \not\equiv 1 \bmod p$, dann hat das letzte von 1 verschiedene Element der Folge

$$b^t \bmod p,\ b^{2t} \bmod p,\ \ldots,\ b^{2^{s-1}t} \bmod p$$

den Wert -1. Also ist ein r mit $0 \leq r \leq s$ und der geforderten Eigenschaft gefunden und p ist damit eine starke Pseudoprimzahl.

Algorithmus (13.4): [Probabilistischer Primzahltest]

Sei p' gegeben. Wähle zufällig k Zahlen aus { 1, 2, ... , p'-1 }. Für jede ausgewählte Zahl b prüfe, ob p' starke Pseudoprimzahl zur Basis b ist. Ist dies der Fall, so kann man aufgrund eines Ergebnisses aus der Wahrscheinlichkeitstheorie sagen, daß p' eine Primzahl ist, wobei die Fehlerwahrscheinlichkeit dieser Aussage $\leq 4^{-k}$ ist.

Dieser Test ist sehr schnell, da es für das Potenzieren sehr schnelle Algorithmen gibt.

Bei einem Probelauf wurde die Zahl $2^{400}-593$ mit mehr als 100 Werten für b getestet. $2^{400}-593$ ist mit der Fehlerwahrscheinlichkeit von $\leq 4^{-100} < 10^{-60}$ eine Primzahl.

13B : Kryptographie

Eine Menge M von Personen will vertrauliche Nachrichten über einen öffentlichen Kanal austauschen. Dabei ergibt sich folgendes Problem: Das Alphabet a, b, c, ..., x, y, z wird mit den Zahlen 01, 02, ..., 26 identifiziert. Wörter werden durch 00 getrennt. Die Nachricht (der Text) ergibt sich also als Zahl M. Diese Zahl muß verschlüsselt werden (Satzzeichen werden hier nicht berücksichtigt).

Eine als 'RSA-Chiffre' bekanntgewordene Chiffrierung, zu deren Entschlüsselung probabilistische Primzahltests (siehe 13.4) herangezogen werden können, wird im folgenden beschrieben. Zunächst sollen einige Bezeichnungen festgelegt werden:

Bezeichnungen (13.5):

Eine CHIFFRE ist eine vom Sender vor der Übermittlung durchgeführte injektive Abbildung f der 'verständlichen' Daten auf 'unverständliche'. Der Empfänger dechiffriert den übermittelten CHIFFRE-TEXT durch Anwendung der invertierten Abbildung f^{-1} in den zugehörigen KLARTEXT.

Zur Chiffrierung wird eine reichhaltige Familie $(f_s)_{s \in S}$ von Chiffren gewählt, die mit dem SCHLÜSSEL s parametrisiert ist (wobei die Vereinbarung des Schlüssels auf besonders sicherem Wege, etwa per Kurier, geschehen muß).

Definition (13.6):

Sei M die Menge der zu übertragenden Klartexte, C die Menge der möglichen Chiffretexte und S die Menge der möglichen Schlüssel. Ein KRYPTOSYSTEM K = (M, C, S, F) ist eine Menge F von injektiven Abbildungen

$$F = \{ E_s : M \longrightarrow C \mid s \in S \}.$$

Die Frage nach möglicherweise verborgenen Eigenschaften von f, die die Bildung der Umkehrfunktion (d. h. die Dechiffrierung) mit geringem Rechenaufwand ermöglichen, führt zur Untersuchung von Abbildungen, die diese Möglichkeit mit hoher Sicherheit ausschliessen:

Definition (13.7):

Eine injektive Abbildung f: X --> Y heißt EINWEGFUNKTION, wenn

a) für beliebige $x \in X$ der Wert $y = f(x)$ 'leicht' zu berechnen ist.
b) es zu gegebenem $y \in Y$ 'praktisch unmöglich' festzustellen ist, welches x die Gleichung $f(x) = y$ löst.

Die Beschreibung von $f^{-1}_{|\,im\ f}$ ist aus der gewählten Darstellung von f mit realistischem Rechenaufwand nicht herzuleiten.

Definition (13.8):

Solche Einwegfunktionen f, deren praktisch undurchführbar erscheinende Umkehrung durch Mitteilung zusätzlicher 'Geheim'-Parameter praktisch leicht wird, heißen FALLTÜR-FUNKTIONEN.

Die bekannteste Falltürfunktion tritt in der 1977 vorgeschlagenen RSA-Chiffre auf, die nun beschrieben wird.

Das RSA-System benutzt als Klar- und Chiffretextmenge

$$M = C = \{ n \in \mathbb{N}_0 \mid n \leqslant m_0 \},\ m_0 \text{ vorgegeben.}$$

Die Menge S der Schlüssel besteht aus den Paaren (N,r) mit $N > m_0$ und $ggT(r,\varphi(n)) = 1$, d.h.

$$S = \{ (N,r) \mid N > m_0,\ ggT(r,\varphi(n)) = 1 \} \subseteq \mathbb{N}^2,$$

wobei $\varphi(n)$ die in (11.6) eingeführte Eulersche φ-Funktion ist. Jeder Benutzer wählt nun zwei verschieden Primzahlen p, q mit jeweils mindestens 40 Stellen (!) mit

$$n = p * q > m_0$$

und dazu eine sehr große Zahl d, die teilerfremd zu $\varphi(n)$ ist, wobei ja nach (11.7) $\varphi(n) = (p-1)\,(q-1)$ gilt:

$$\max(p,q) < d < \varphi(n)$$
$$\text{ggT}\,(d,\varphi(n)) = 1. \qquad (*)$$

Sein privater (Falltür-)Schlüssel ist das Tripel (p,q,d). Der öffentlichen Schlüsselbibliothek teilt er den öffentlichen Schlüssel a(T) = (n,e) mit. Dabei ist eine SCHLÜSSELBIBLIOTHEK ein Kryptosystem mit einer Adressierabbildung a: Teilnehmermenge --> S, die jedem Teilnehmer T den (öffentlichen) Schlüssel a(T) zuordnet.

In unserem Fall wird e als multiplikatives Inverses von d mod $\varphi(n)$, d.h. aus

$$e * d = 1 \bmod \varphi(n) \qquad (**)$$

mit dem Euklidischen Algorithmus (9.5) bestimt.

Erläuterung des Verfahrens an einem Beispiel:

N Streifenwagen der Polizei ('Teilnehmer') kommunizieren über Polizeifunk ('Kanal') und tauschen dabei sensitive Personendaten aus, die vor unbefugtem Zugriff geschützt werden müssen. Da es technisch möglich ist, Polizeifunk abzuhören ('prinzipiell öffentlicher Kanal'), müssen die Texte kryptographisch verschlüsselt werden.

T_1 will an T_2 einen Klartext m (z. B. eine Personenbeschreibung) schicken. T_1 berechnet mit der Chiffre $E_{s=1}$ und $s = a(T_1) = (n,e)$ den Chiffretext

$$c = E_s(m) = m^e \bmod n$$

aus dem Klartext m in O(log n) Schritten.

Dieser Text c wird nun zu T_2 gesendet, der ihn mit der nur ihm bekannten Falltür-Information 'n = p*q' wie folgt entschlüsseln kann (wobei $D_{s=2}$ die Umkehrfunktion zu $E_{s=1}$ ist):

$$
\begin{aligned}
D_{s=2}(c) &= (E_{s=1}(m))^{d} \\
&= c^{d} \\
&= (m^{e})^{d} \mod n \\
&= m^{ed} \mod n \\
&= m^{1+k(p-1)(q-1)} \mod n \quad \text{, nach Wahl von e} \\
&= m^{1+k\varphi(n)} \mod n \quad \text{, nach Def. von } \varphi \\
&= m \bmod n \quad \text{, nach Satz (11.8).}
\end{aligned}
$$

T_2 muß also nur den Chiffretext c mit d potenzieren und modulo n rechnen und hat den Klartext vorliegen. T_2 könnte ihn nun ändern und mit dem öffentlichen Schlüssel $E_{s=2}$ chiffrieren und zurück an T_1 schicken usw.

Bewertung:

Bei einem Entschlüsselungsversuch müßte der Angreifer i.a. zuerst einmal versuchen, die Zahl n in ihre Primfaktoren zu zerlegen und dann aus e die Zahl d zu bestimmen. Die Primfaktorzerlegung ist zwar eindeutig, aber sie (bei ca. 80-stelligem n) herauszufinden, ist sehr schwer. Beachte, daß T_1 diese Primzahlzerlegung ja nicht durchführt, sondern Zahlen p und q wählt und daraus n berechnet und veröffentlicht.

Der schnellste bisher bekannte Algoritmus zur Primfaktorzerlegung von Schroeppel (1978) benötigt $O(e^{(\log n)(\log\log n)})$ Schritte. Dieser Algorithmus würde bei einer 80-stelligen Zahl n, die Produkt zweier etwa 40-stelliger Primzahlen p, q ist, auf einem Rechner mit 1 µs Schaltzeit pro Division mehrere Jahre Rechenzeit benötigen.

Dagegen ist der Aufwand für das Finden großer Primzahlen, die ja für die Angabe des Schlüssels benötigt werden, etwa nach dem Solovay-Strassen-Test (nach k Schritten einer Berechnung von Jacobi-Symbolen: Irrtumswahrscheinlichkeit 2^{-k}) oder nach dem oben beschriebenen probabilistischen Primzahltest (13.4) sehr klein.

Kapitel 14 : Rekurrenzrelationen

Die nachfolgenden Kapitel geben uns ein Werkzeug in die Hand, um die Komplexität einiger Klassen von Algorithmen abzuschätzen.

Definition (14.1):

Eine Funktion $f : \mathbb{N}_0 \longrightarrow \mathbb{R}$ heißt DISKRETE, NUMERISCHE FUNKTION oder FOLGE. Die Funktionswerte von f werden als Folge $f(i)=a_i$ geschrieben.

Beispiel:
Auf einem Sparbuch befinden sich 100 DM, die jährlich mit 5 % verzinst werden. Welcher Betrag steht im r. Jahr auf dem Sparbuch?

Jahr:	0	1	2	3	...
Betrag:	100	105	110.25	115.7625	...

Als Lösung ergibt sich folgende Gleichung ('Rekurrenzrelation' genannt):

$$a_{r+1} = a_r + (5/100)\, a_r.$$

Die Lösung dieser Rekurrenzrelation ist:

$$a_r = 100 * 1.05^r.$$

Bemerkung (14.2):
Jede diskrete numerische Funktion läßt sich als Folge auffassen, daher kann man folgende Operationen auf der Menge der diskreten, numerischen Funktionen einführen:
Seien f, g zwei diskrete, numerische Funktionen, gegeben durch die Folgen $(a)_r = a_0, a_1, \ldots, a_r, \ldots$ und $(b)_r = b_0, \ldots, b_r, \ldots$
Dann heißt

$(a+b)_r = (a_0+b_0, \ldots, a_r+b_r, \ldots)$ SUMME

$(a\ b)_r = (a_0\ b_0, \ldots, a_r\ b_r, \ldots)$ PRODUKT

$(a*b)_r = (0, a_0b_1+a_1b_0, \ldots, \Sigma_{i=0..r}\ a_ib_{r-i}, \ldots)$ KONVOLUTION (FALTUNG)

$(S^1 a)_r = (0, \ldots, 0, a_{r-1}, a_{r-1+1}, \ldots)$ SHIFT mit $r = i+1$.

Beispiel (14.3):
Gesucht ist die Anzahl A der Wörter der Länge r über dem Alphabet {x,y,z,α,β}, wobei der 'erste Teil' (Länge: i, von 0 bis r variabel) des Wortes aus Buchstaben x, y, z und der zweite Teil aus Buchstaben α, β besteht.

Es gilt $a_r = 3^r$ und $b_r = 2^r$ und

$$A = (a*b)_r = \Sigma_{i=0..r} 3^i 2^{r-i}.$$

Definition (14.4):

Ist $(a)_r$ eine diskrete, numerische Funktion, dann heißt die Potenzreihe

$$A(z) = a_0 + a_1 z + \ldots + a_r z^r + \ldots = \Sigma_{i=0..\infty} a_i z^i$$

die ERZEUGENDE FUNKTION von $(a)_r$.

Beispiel (14.5):

Für $f : \mathbb{N}_0 \longrightarrow \mathbb{R}$, $f(r) = 3^r$ ist die erzeugende Funktion

$$A(z) = \Sigma_{i=0..\infty} 3^i z^i = 1/(1-3z)$$

innerhalb des Konvergenzradius; $\Sigma_{i=0..\infty} q^i$ konvergiert für $|q| < 1$ gegen $1/(1-q)$ (geometrische Reihe).

Wir wollen anmerken, daß sich alle Operationen für diskrete numerische Funktionen auch für Potenzreihen einführen lassen. Welche Rolle die erzeugenden Funktionen spielen, soll erst in einem der nachfolgenden Kapitel diskutiert werden.

Wir haben nun 4 verschiedene Darstellungen für diskrete numerische Funktionen, am Beispiel $(a_r) = (3^r)$:

1) $f : \mathbb{N} \longrightarrow \mathbb{R}$, $f(r) = 3^r$

2) f als Folge, $(a^r) = (3^0, 3^1, 3^2, \ldots)$

3) f als Koeffizient der erzeugenden Funktion $A(z)=1/(1-3z)$

4) f als Rekurrenzrelation

Wir befassen uns im folgenden mit Rekurrenzrelationen.

Beispiel (14.6): [FIBONACCIFOLGE]
Aufgabe: Wieviele Kaninchenpaare existieren nach einem Jahr, wenn zu Jahresbeginn ein neugeborenes Paar vorhanden ist und jedes geschlechtsreife Paar jeden Monat ein neues Paar erzeugt? Die Geschlechtsreife tritt jeweils zwei Monate nach der Geburt ein.

Lösung:
Sei a_r die Anzahl der Kaninchenpaare im r. ten Monat. Dann gilt: $a_0 = 0$, $a_1 = 1$, $a_2 = 1$, $a_3 = 1+1$, $a_4 = 2+1$, $a_5 = 5$, $a_6 = 8$,... Die gesuchte Rekurrenzrelation lautet:

$$a_r = a_{r-1} + a_{r-2},\ a_0 = 0,\ a_1 = 1.$$

Definition (14.7):

Eine Gleichung in den Unbekannten $a_r, a_{r-1}, \dots, a_{r-k}$, $k \leq r$ von der Form

$$a_r = f(a_{r-1}, \dots, a_{r-k})$$

heißt REKURRENZRELATION.

Um eine Rekurrenzrelation aufzulösen, sind einige Startwerte (Anfangswerte) nötig, s.o. Werte a_0 und a_1.

Definition (14.8):

Sei eine Folge $a_0, a_1, \dots, a_r, \dots$ gegeben, dann definiert man als ERSTE DIFFERENZ

$$\delta a_r = a_r - a_{r-1},$$

und als ZWEITE DIFFERENZ

$$\delta^2 a_r = \delta a_r - \delta a_{r-1}, \text{ usw.}$$

Eine DIFFERENZENGLEICHUNG ist eine Gleichung, die a_r mit ihren Differenzen in Beziehung setzt, z. B. $\delta^2 a_r - 3\delta a_r + a_r = 0$.
Jede Rekurrenzrelation läßt sich als Differenzengleichung ausdrücken und umgekehrt. Also lassen sich die Methoden für Differenzengleichungen und bis zu einem gewissen Grad für Differentialgleichungen zur Lösung von Rekurrenzrelationen anwenden.

Wir werden uns im folgenden mit linearen Rekurrenzrelationen befassen, deren Auflösung im allgemeinen nicht schwierig ist.

Definition (14.9):

Einen Rekurrenzrelation der Form

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = g(r)$$

heißt LINEARE REKURRENZRELATION MIT KONSTANTEN KOEFFIZIENTEN, wobei die $c_i \in \mathbb{R}$ sind. Ist $g(r) = 0$, so heißt die Rekurrenzrelation HOMOGEN, andernfalls heißt sie INHOMOGEN.

Beispiel:
$a_r - a_{r-1} + a_{r-2} = 0$ ist eine homogene lineare Rekurrenzrelation.

Definition (14.10):

Eine Lösung $a^{(p)} = (a_0^{(p)}, \ldots, a_r^{(p)}, \ldots)$ der inhomogenen linearen Rekurrenzrelation

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = g(r)$$

heißt eine PARTIKULÄRE LÖSUNG. Eine Lösung $a^{(h)} = (a_0^{(h)}, \ldots, a_r^{(h)}, \ldots)$ der zugehörigen homogenen linearen Rekurrenzrelation

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = 0$$

heißt eine HOMOGENE LÖSUNG.

Satz (14.11):

Sei $c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = g(r)$ eine lineare Rekurrenzrelation (*). Jede Lösung $a^{(l)}$ setzt sich dann aus einer homogenen Lösung $a^{(h)}$ und einer partikulären Lösung $a^{(p)}$ zusammen:

$$a^{(l)} = a^{(h)} + a^{(p)}.$$

Beweis:

Zunächst sieht man, daß $a^{(h)} + a^{(p)}$ eine Lösung von (*) ist, denn

$$c_0(a_r^{(h)} + a_r^{(p)}) + \ldots + c_k(a_{r-k}^{(h)} + a_{r-k}^{(p)})$$
$$= c_0 a_r^{(h)} + \ldots + c_k a_{r-k}^{(h)}$$
$$+ c_0 a_r^{(p)} + \ldots + c_k a_{r-k}^{(p)}$$
$$= 0 + g(r) = g(r).$$

Sind $a^{(1)}$ und $a^{(p)}$ zwei Lösungen von (*), so ist $a^{(1)}-a^{(p)}$ eine Lösung des homogenen Teils von (*). Wir setzen $a^{(h)} = a^{(1)}-a^{(p)}$ und haben

$$a^{(1)} = a^{(p)} + a^{(h)}.$$

Kapitel 15 : Lineare homogene Rekurrenzrelationen

Definition (15.1):

Sei die lineare Rekurrenzrelation

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = 0 \quad (*)$$

gegeben. Dann heißt die Gleichung k. ten Grades

$$c_0 x^k + c_1 x^{k-1} + \ldots + c_k = 0$$

in der Unbekannten x die zu (*) gehörende CHARAKTERISTISCHE GLEICHUNG.

Beispiel:
Sei $a_r = a_{r-1} + a_{r-2}$ die Fibonacci-Rekurrenzrelation. Dann ist für $a_r - a_{r-1} - a_{r-2} = 0$ die charakteristische Gleichung gegeben durch

$$x^2 - x - 1 = 0.$$

Die Wurzeln sind: $\alpha_1 = (1+5^{1/2})/2$ und $\alpha_2 = (1-5^{1/2})/2$.

Wir bemerken, daß nach dem Hauptsatz der Algebra eine Gleichung k. ter Ordnung $c_0 x^k + c_1 x^{k-1} + \ldots + c_k = 0$ k Wurzeln (Nullstellen) hat. Diese können alle voneinander verschieden sein.

Satz (15.2):

Seien $\alpha_1, \ldots, \alpha_k$ k verschiedene Lösungen der charakteristischen Gleichung der Rekurrenzrelation (*), dann ist

$$a_r^{(h)} = A_1 \alpha_1^r + \ldots + A_k \alpha_k^r$$

eine allgemeine homogene Lösung von (*). $A_1, \ldots, A_k$ werden aus den Startwerten ermittelt.

Beweis:
Wir setzen in (*) ein und erhalten:

$$c_0(A_1\alpha_1^r + \ldots + A_k\alpha_k^r)$$

$$+ \ldots + c_k(A_1\alpha_1^{r-k} + \ldots + A_k\alpha_k^{r-k})$$

$$= A_1\alpha_1^{r-k}(c_0\alpha_1^k + c_1\alpha_1^{k-1} + \ldots + c_k)$$

$$+ \ldots + A_k\alpha_k^{r-k}(c_0\alpha_1^k + c_1\alpha_1^{k-1} + \ldots + c_k)$$

$$= 0.$$

Damit ist also $a_r^{(h)} = A_1\alpha_1^r + \ldots + A_k\alpha_k^r$ eine Lösung von (*) und es bleibt zu zeigen, daß auch jede andere Lösung eine solche Form hat.

Sei also $b_r^{(h)}$ eine beliebige Lösung von (*). Daraus folgt sofort, daß $b_0^{(h)} = a_0^{(h)}$, $b_1^{(h)} = a_1^{(h)}$, ... ist. Wir müssen also zeigen, daß die Konstanten $A_1, \ldots, A_k$ eindeutig bestimmt sind, so daß gilt:

$$b_0^{(h)} = A_1 + \ldots A_k$$

$$b_1^{(h)} = A_1\alpha_1 + \ldots + A_k\alpha_k$$

usw.

$$b_{k-1}^{(h)} = A_1\alpha_1^{(k-1)} + \ldots A_k\alpha_k^{(k-1)}$$

Wir haben also ein System von k linearen Gleichungen in den Unbekannten $A_1, \ldots, A_k$. Als die Koeffizientenmatrix ergibt sich:

$$M = \begin{pmatrix} 1 & 1 & 1 \\ \alpha_1 & \alpha_2 & \alpha_k \\ & & \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \alpha_k^{k-1} \end{pmatrix}$$

Diese Matrix heißt auch VANDERMONDSCHE MATRIX und ihre Determinante ist $\det(M) = \prod_{1\leq i<j\leq k}(\alpha_j-\alpha_i)$. Da die Wurzeln alle verschieden sind, ist $\det(M) \neq 0$ und $A_1, \ldots, A_k$ sind eindeutig bestimmbar. Also ist die obige Lösung eine allgemeine Lösung.

Beispiel:

Sei $a_r = a_{r-1} + a_{r-2}$ mit $\alpha_1 = (1+5^{1/2})/2$ und $\alpha_2 = (1-5^{1/2})/2$ als Wurzeln der charakteristischen Gleichung. Nach obigem Satz ist

$$a_r^{(h)} = A_1 ((1+5^{1/2})/2)^r + A_2 ((1-5^{1/2})/2)^r$$

eine homogene Lösung. A_1 und A_2 sind aus den Startwerten $a_0 = 0$ und $a_1 = 1$ zu ermitteln:

$$0 = A_1 + A_2$$

$$1 = A_1 (1+5^{1/2})/2 + A_2 (1-5^{1/2})/2.$$

Lösungen: $A_1 = 1/\mathrm{sqrt}(5)$, $A_2 = -1/\mathrm{sqrt}(5)$.

Also haben wir als Lösung der Rekurrenzrelation:

$$a_r^{(h)} = 1/\mathrm{sqrt}(5)\ (1+\mathrm{sqrt}(5)/2)^r - 1/\mathrm{sqrt}(5)\ (1-\mathrm{sqrt}(5)/2)^r.$$

Es ist klar, daß wir auch den Fall betrachten müssen, daß nicht alle k Wurzeln $\alpha_1, \ldots, \alpha_k$ der charakteristischen Gleichung verschieden sind. Gilt $\alpha_1 = \alpha_2 = \ldots = \alpha_m$ für $2 \leqslant m \leqslant k$, so sagt man : die Wurzel α_1 hat die VIELFACHHEIT m.

Es ist bekannt, daß $p(x) = c_0x^k + c_1x^{k-1} + \ldots + c_k$ teilbar durch $(x - \alpha_1)$ ist, weil α_1 Wurzel ist. Nun ist α_1 eine Wurzel der Vielfachheit m und daher ist

$$p(x) = (x - \alpha_1)^m q(x),$$

wobei q(x) ein Polynom (k-m)-ten Grades ist.
Wenn man nun die 1. Ableitung von p(x) bildet, erhält man

$$p'(x) = m (x - \alpha_1)^{m-1} q(x) + (x - \alpha_1)^m q'(x).$$

Offensichtlich ist α_1 auch Wurzel der 1. Ableitung von p(x), also von $p'(x) = 0$. Man sieht auf diese Weise, daß α_1 auch Wurzel der i. Ableitung (i=2, ... , m-1) von p(x) ist.

Lemma (15.3):

Hat die charakteristische Gleichung eine Wurzel α_1 der Vielfachheit m, so ist

$$a_r^{(h)} = (A_1 r^{m-1} + \ldots + A_{m-1} r + A_m)\, \alpha_1^r$$

eine homogene Lösung.

Beweis:
Da α_1 Wurzel der Gleichung

$$c_0 x^r + \ldots + c_k x^{r-k} = 0 \quad (*)$$

ist, folgt, daß $A_m \alpha_1^r$ eine Lösung ist. Außerdem ist α_1 auch Wurzel der 1. Ableitung

$$c_0 r x^{r-1} + \ldots + c_r (r-k) x^{r-k-1} = 0,$$

da α_1 ja eine m-fache Wurzel ist. Multipliziert man die Gleichung nun mit $A_{m-1} x$, so erhält man:

$$c_0 A_{m-1} r x^r + \ldots + c_r (r-k) A_{m-1} x^{r-k} = 0.$$

Man sieht, daß $A_{m-1} r \alpha_1^r$ eine Lösung ist.
Die Tatsache, daß α_1 die 2. te, 3. te, ... und (m-1)te Ableitung von (*) erfüllt, führt dazu, daß $A_{m-2} r^2 \alpha_1^r, \ldots, A_1 r^{m-1} \alpha_1^r$ Lösungen sind.

Wir fassen nun zusammen:

Satz (15.4):

Seien $\alpha_1, \ldots, \alpha_t$, $1 \leq t \leq k$ die verschiedenen Wurzeln mit den Vielfachheiten $e_1, \ldots, e_t$ der charakteristischen Gleichung der Rekurrenzrelation

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = 0.$$

Dann ist die allgemeine Lösung $a_r^{(h)} = a_{r,1}^{(h)} + \ldots + a_{r,t}^{(h)}$, wobei

$$a_{r,i}^{(h)} = (A_{1,i} r^{e_i - 1} + \ldots + A_{e_i - 1, i} r + A_{e_i, i}) \alpha_i^r$$

ist.

Beweis:
Aus den vorangegangenen Überlegungen (Lemma 15.3) ergibt sich, daß $a_r^{(h)}$ eine Lösung ist.
Mit derselben Überlegung wie im Beweis von (15.2) kann gezeigt werden, daß diese Lösung allgemein ist.

Beispiel:
Sei $a_r + 6\, a_{r-1} + 12\, a_{r-2} + 8\, a_{r-3} = 0$, $a_0 = 1$, $a_1 = -6$, $a_2 = 28$. Die charakteristische Gleichung

$$\alpha^3 + 6\alpha^2 + 12\alpha + 8 = 0$$

hat $\alpha_1 = -2$ als 3-fache Nullstelle. Nach (15.4) gilt also

$$a_r^{(h)} = (A_1 r^2 + A_2 r + A_3)\ (-2)^r,$$

wobei die A_1, A_2 und A_3 noch aus den Startwerten zu ermitteln sind:

$$\begin{array}{lrrrr} r = 0: & & & A_3 & = 1 \\ r = 1: & A_1 & + A_2 & - 2A_3 & = -6 \\ r = 2: & 4A_1 & + 2A_2 & + 4A_3 & = 28 \end{array}$$

Die Lösung dieses linearen Gleichungssystem ist $A_1 = A_2 = A_3 = 1$ und damit lautet die Lösung der Rekurrenzrelation

$$a_r^{(h)} = (r^2 + r + 1)\ (-2^r).$$

Zur Beantwortung der Frage, wieviele Startwerte man braucht, um bei

einer linearen homogenen Rekurrenzrelation $A_1, \ldots, A_k$ zu bestimmen, dient folgender Satz:

Lemma (15.5):

Für eine lineare homogene Rekurrenzrelation

$$c_0 a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k}$$

können k Koeffizienten $A_1, \ldots, A_k$ der homogenen Lösung durch Startwerte

$$a_s, a_{s+1}, \ldots, a_{s+k-1}$$

für jedes s, also insbesondere auch für $s = 0$, bestimmt werden.

Beweis:
OBdA seien alle Wurzeln des charakteristischen Polynoms verschieden. Die allgemeine Lösung ist von der Form

$$a_r = A_1\alpha_1^r + \ldots + A_k\alpha_k^r + p(r),$$

wobei p(r) eine partikuläre Lösung ist. Wir haben also folgendes System von k linearen Gleichungen:

$$\begin{aligned} a_s &= A_1\alpha_1^s + \ldots + A_k\alpha_k^s + p(s) \\ &\ldots \\ a_{s+k-1} &= A_1\alpha_1^{s+k-1} + \ldots + A_k\alpha_k^{s+k-1} + p(s+k-1). \end{aligned}$$

Aus diesen k Gleichungen können die $A_1, \ldots, A_k$ bestimmt werden.

Kapitel 16 : Lineare inhomogene Rekurrenzrelationen

Es gibt kein allgemeines Verfahren, die partikuläre Lösung einer linearen, inhomogenen Rekurrenzrelation zu bestimmen. Der Leser ist also auf sein eigenes Geschick angewiesen. Wir wollen jedoch hier anhand von Beispielen einige Lösungsvorschläge geben.

16A: Lösung durch Iteration und Induktion

Betrachte die Rekurrenzrelation $a_r = a_{r-1} + r^3$, $a_0 = 0$. Durch Iteration gewinnt man:

$$\begin{aligned} a_r &= a_{r-1} + r^3 \\ &= a_{r-2} + (r-1)^3 + r^3 \\ &= 0 + 1^3 + 2^3 + \ldots + (r-1)^3 + r^3. \end{aligned}$$

Wir sehen, daß gilt:

$$\begin{aligned} a_0 &= 0^2 \\ a_1 &= 0 + 1^2 = 1^2 \\ a_2 &= 1 + 2^3 = (1+2)^2 \\ a_3 &= 9 + 3^3 = (1+2+3)^3 \end{aligned}$$

Da wir wissen, daß $\Sigma_{i=1\ldots r}\ i = r(r+1)/2$ gilt, formulieren wir nun die Behauptung: $a_r = r^2(r+1)^2/4$.

Beweis:
Induktionsanfang: $r=0$ ergibt $a_0 = 0$, den Startwert.

Induktionsschritt: $a_{r+1} = a_r + (r+1)^3 = r^2(r+1)^2/4 + (r+1)^3$

$$= (r+1)^2(r+2)^2/4.$$

Diese Methode (16A) kann auch bei nicht-linearen Rekurrenzrelationen mit Erfolg angewendet werden.

16B : Lösung mit einer Differenzentafel

Für eine Funktion φ ist die Differenz $\delta\varphi$ definiert durch

$$\delta\varphi(x) = \varphi(x+1) - \varphi(x).$$

Allgemein ist die Differenz k-ter Ordnung definiert durch

$$\delta^k\varphi(x) = \delta^{k-1}\varphi(x+1) - \delta^{k-1}\varphi(x),\ k \geqslant 1.$$

Satz (16.1):

Ist $\varphi(x) = b_n x^n + \ldots b_1 x + b_0$ ein Polynom n-ten Grades, dann sind die Differenzen (n+1)-ter Ordnung alle Null.

Beweis:
Induktionsanfang:
n=0, $\varphi(x) = a_0$. Es gilt $\delta\varphi(x) = a_0 - a_0 = 0$.

Induktionsvoraussetzung:
Die Behauptung sei richtig für alle Polynome vom Grad k, $k < n$, $n \geqslant 1$.

Sei nun $\varphi(x)$ ein Polynom n-ten Grades. Betrachte das Polynom

$$\delta\varphi(x) = \varphi(x+1) - \varphi(x)$$

und beachte, daß $\delta\varphi(x)$ vom Grad k, $k < n$ ist (die höchste Potenz fällt gerade weg durch die Subtraktion) und daß die Differenzen (k+1)-ter Ordnung nach Induktionsvoraussetzung sämtlich Null sind. Also sind für $\varphi(x)$ die Differenzen n-ter Ordnung alle Null.

Eine Anwendung dieses Satzes für Rekurrenzrelationen gibt folgendes Beispiel: Sei $a_r = a_{r-1} + 2r$ mit $a_0 = 1$ gegeben.

Bilde folgende Differenzentafel:

a_0	a_1	a_2	a_3	...
	a_1-a_0	a_2-a_1	a_3-a_2	...
	$a_2-a_1)-(a_1-a_0)$	$(a_3-a_2)-(a_2-a_1)$	...	

In unserem Beispiel ergibt dies:

```
1     3     7     13     21 ...
   2     4     6     8 ...
      2     2     2 ...
         0     0  ...
```

Aus Satz (16.1) folgt, daß ein Polynom 2-ten Grades eine solche Differenzentafel erfüllt. Wir setzen also an: $\varphi(x) = b_2x^2 + b_1x + b_0$ und bestimmen b_0, b_1 und b_2 aus den ersten drei Werten:

$$\varphi(0) = b_2 0^2 + b_1 0 + b_0 = 1$$

$$\varphi(1) = b_2 1^2 + b_1 1 + b_0 = 3$$

$$\varphi(2) = b_2 2^2 + b_1 2 + b_0 = 7$$

und erhalten als Lösung dieses Gleichungssystems $b_0 = b_1 = b_2 = 1$, und somit $\varphi(x) = x^2 + x + 1$.

Als Lösung der Rekurrenzrelation ergibt sich daher $a_r = r^2 + r + 1$, was sich mit Hilfe eines Induktionsbeweises leicht nachprüfen läßt.

16C : Rekurrenzrelation der Form $a_r = c\, a_{r/d} + g(r)$

Viele Algorithmen arbeiten nach der Strategie 'DIVIDE AND CONQUER' (d.h. 'Teile und herrsche'). Dabei wird ein Problem mit r Objekten in d Unterprobleme mit r/d Objekten geteilt. Dadurch ergeben sich Rekurrenzrelationen obiger Form. Die Art ihrer Lösung wollen wir für einige Spezialfälle in einer Tabelle angeben.

Dabei ist int(u) für $u \in \mathbb{R}$ definiert als ~~kleinste~~ größte ganze Zahl w mit $u \geq w$, also z. B. $\text{int}(\pi) = 3$, $\text{int}(-\pi) = -4$.
h stehe für eine Konstante.

c, d	g(r)	a_r
c = 1	g(r) = h	$\text{int}(\log_d r)$
c = d	g(r) = h	$A_1 r + A_2$
c < d	g(r) = h r	$A_1 r$
c = d	g(r) = h r	$O(r \log_d r)$
c > d	g(r) = h r	$O(r \log_d c)$

Ein Lösungsansatz bei solchen Rekurrenzrelationen besteht häufig darin, nur solche $r \in \mathbb{N}$ zu betrachten, die sich als Potenz von d schreiben lassen.

Beispiel (16.2):

Mit der 'normalen' Multiplikationsmethode braucht man n^2 Stelle-mal-Stelle Multiplikationen, um 2 n-stellige Zahlen zu multiplizieren. Es soll nun eine Methode vorgestellt werden, die mit weniger Multiplikationen auskommt.

Verfahren:

Sei $n = 2^k$ und seine α, β zwei n-stellige Zahlen. Wir zerlegen

$$\alpha = \alpha_1 \, 10^{n/2} + \alpha_2, \quad \beta = \beta_1 \, 10^{n/2} + \beta_2,$$

so daß $\alpha \, \beta = \alpha_1 \, \beta_1 \, 10^n + \alpha_1\beta_2 \, 10^{n/2} + \alpha_2\beta_1 \, 10^{n/2} + \alpha_2\beta_2$.

α_1 besteht dabei aus den ersten $n/2 = 2^{k-1}$ Stellen von α, und α_2 besteht aus den letzten n/2 Stellen von α.
Man braucht also 4 Multiplikationen von jeweils 2 n/2-stelligen Zahlen. Wegen

$$\alpha_1\beta_2 + \alpha_2\beta_1 = (\alpha_1-\alpha_2)(\beta_2-\beta_1) + \alpha_1\beta_1 + \alpha_2\beta_2$$

genügen sogar 3 Multiplikationen.
Wir haben also die Rekurrenzrelation $a_n = 3 \, a_{n/2}$, mit $a_1 = 1$.

Sei nun b_k die Anzahl der Stelle-mal-Stelle Multiplikationen bei 2^k-stelligen Zahlen. Dann gilt $b_k = 3 \, b_{k-1}$ und $b_0 = 1$. Als Lösung ergibt sich $b_k = 3^k$.

Für $n = 2^k$ gilt nun:

$$b_k = 3^{\log_2 n} = (2^{\log_2 3})^{\log_2 n}$$

$$= (2^{\log_2 n})^{\log_2 3}$$

$$= n^{\log_2 3} = \text{ca. } n^{1.59},$$

im Vergleich zu n^2 Operationen bei der herkömmlichen Methode.

Für den Fall $n \neq 2^k$ überlegt man sich, daß man vorne stets Nullen hinzusetzen kann. Für $2^k \leqslant n \leqslant 2^{k+1}$ braucht man also 3^{k+1} Multiplikationen. Beim herkömmlichen Verfahren benötigt man $n^2 \geqslant 4^k$ Multiplikationen. Ab $k > 4$ (also 16-stelligen Zahlen) ist obiges Verfahren also immer günstiger.

16D : Rekurrenzrelationen der Form $a_r = c\, a_{r-1} + g(r)$

In diesem Fall sollte man zunächst die zugehörige homogene Rekurrenzrelation $a_r = c\, a_{r-1}$ lösen, mit der allgemeinen Lösung

$$a_r^{(h)} = A\, c^r.$$

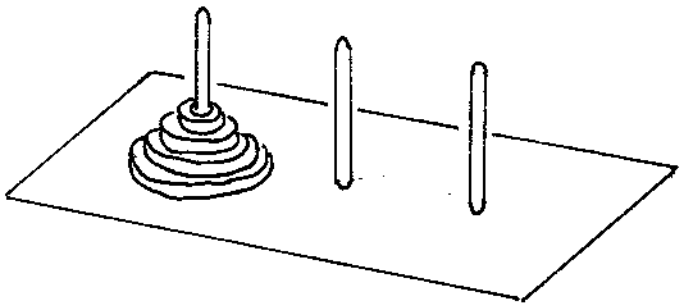
Ist $a_r^{(p)}$ eine partikuläre Lösung, so ergibt sich

$$a_r = a_r^{(h)} + a_r^{(p)} = A\, c^r + a_r^{(p)}.$$

Der Koeffizient A kann erst dann mit Hilfe eines Startwertes bestimmt werden wenn man $a_r^{(p)}$ kennt. Für $c \neq 0$ kann man $a_r^{(p)}$ mit Hilfe von Ansätzen und mit Koeffizientenvergleich sehr häufig lösen. Die folgende Tabelle gibt eine Hilfe:

g(r)	Ansatz für $a_r^{(p)}$
$g(r) = h$ (const)	P
$g(r) = h\,r$	$P_1 r + P_0$
$g(r) = h\,r^2$	$P_2 r^2 + P_1 r + P_0$
$g(r) = h\,r^3$	$P_3 r^3 + P_2 r^2 + P_1 r + P_0$
usw.	
$g(r) = h^r$	$P\,h^r$

Beispiel (16.3): [Türme von Hanoi]



Die n Ringe auf dem 1. Stab sollen (ein Ring nach dem anderen) auf den 3. Stab gebracht werden, wobei man den mittleren Stab zu Hilfe nimmt. Bei diesem Verfahren darf kein größerer Ring auf einem kleineren liegen. In wieviel Schritten kann man das Problem lösen?

Der Schlüssel zur Lösung ist folgende Beobachtung: Um den r-ten Ring vom Stab 1 zum Stab 3 zu bringen, müssen zunächst die r-1 kleineren Ringe zum Stab 2 gebracht werden und später, wenn der r-te Ring auf Stab 3 liegt, ebenfalls auf Stab 3 gebracht werden. Sei also a_r die Anzahl der Schritte, um r Ringe von Stab 1 zu Stab 3 zu bringen. Dann gilt $a_r = a_{r-1} + 1 + a_{r-1}$, wie wir das beschrieben haben.
Wir erhalten also die Rekurrenzrelation

$$a_r = 2\,a_{r-1} + 1, \text{ mit den Startwerten } a_1 = 1,\ a_2 = 3,\ a_3 = 7.$$

Lösung der Rekurrenzrelation:
Die allgemeine Lösung der homogenen Rekurrenzrelation $a_r = 2\,a_{r-1}$ ist $a_r^{(h)} = A\,2^r$, wie man leicht herausfindet.

Die partikuläre Lösung findet man, indem man den Ansatz $a_r^{(p)} = p$ macht, wie das in obiger Tabelle vorgeschlagen wird. Setzt man den Ansatz in $a_r = 2\,a_{r-1} + 1$ ein, so erhält man:

$$p = a_r^{(p)} = 2\,a_{r-1}^{(p)} + 1 = 2\,P + 1.$$

Also ist $P = 2\,P + 1$, daher $P = -1$. $a_r^{(p)} = -1$ ist also eine partikuläre Lösung.
Nun betrachten wir:

$$a_r = a_r^{(h)} + a_r^{(p)} = A\,2^r - 1.$$

Aus der Startbedingung $a_1 = 1$ erhalten wir $1 = A\,2^1 - 1$, also $A = 1$.
Die gesuchte Lösung ist also:

$$a_r = 2^r - 1.$$

Wir geben noch ein weiteres Beispiel, wo man die Lösung nicht gleich erraten kann:

$$a_r + 5a_{r-1} + 6a_{r-2} = 3r^2,\ a_0 = a_1 = a_2 = 1. \qquad (*)$$

Wir nehmen an, daß die allgemeine Form einer partikulären Lösung ist:

$$P_1r^2 + P_2r + P_3,\ P_i \text{ Konstanten.}$$

Setze dies in (*) ein:

$$P_1r^2 + P_2r + P_3 + 5P_1(r-1)^2 + 5P_2(r-1) + 5P_3$$
$$+ 6P_1(r-2)^2 + 6P_2(r-2) + 6P_3 = 3r^2.$$

Man hat:

$$12P_1r^2 + (34P_1-12P_2)r + (29P_1-17P_2+12P_3) = 3r^2.$$

Daraus folgt:

1) $12\,P_1 = 3$
2) $34\,P_1 - 12\,P_2 = 0$
3) $29\,P_1 - 17\,P_2 + 12\,P_3 = 0.$

Also hat man: $P_1 = 1/4$, $P_2 = 17/24$, $P_3 = 115/288$.
Eine partikuläre Lösung ist:

$$a_r^{(p)} = r^2/4 + 17r/24 + 115/288.$$

Für die verbleibenden Schritte kann man nun die bereits bekannten Methoden für lineare homogene Rekurrenzrelationen einsetzen.

Kapitel 17 : Rekurrenzrelationen für einige Sortieralgorithmen

Viele Algorithmen sind rekursiv aufgebaut. Um ihre Komplexität zu beurteilen, muß man die Anzahl von Rechenschritten bezüglich einer Größe der Eingabe bestimmen. In solchen Fällen treten Rekurrenzrelationen auf. Dies haben wir bereits am Beispiel der Türme von Hanoi gesehen. Im folgenden wollen wir das darüberhinaus an drei verschiedenen Sortierverfahren zeigen. In zweien dieser Fälle werden wir die in 15D beschriebene Methode zur Lösung verwenden.

Problem: Gegeben sind n Register $x_1, \ldots, x_n$, in denen n Zahlen gespeichert sind (Inhalt von Register x_i). Alle Zahlen seien verschieden.

Gesucht sind Verfahren, die die Inhalte der Größe nach (aufsteigend) ordnen; d.h. bei der Ausgabe enthält das Register x_1 die kleinste und x_n die größte Zahl.

Ein Sortierverfahren besteht aus einer Reihe von Vergleichen der Inhalte der Register x_i, x_j. Der Fundamentalbefehl $A(x_i, x_j)$ vergleicht die Inhalte und speichert die kleinere Zahl in x_i, die größere in x_j.

$$A(x_i, x_j) \quad \hat{=} \quad \begin{cases} x_i \leftarrow\!\!- \min(x_i, x_j) \\ x_j \leftarrow\!\!- \max(x_i, x_j) \end{cases}$$

17.1: Ein einfaches Sortierverfahren

Beschreibung: Finde die kleinste Zahl in $\{x_1, \ldots, x_n\}$. Diese wird in x_1 gespeichert. Das Problem reduziert sich auf das Sortieren der n-1 Zahlen in $x_2, \ldots, x_n$.

Bezeichne $S(x_1, \ldots, x_n)$ die Prozedur, die die Zahlen sortiert und $M(x_1, \ldots, x_n)$ die Prozedur, die die kleinste der Zahlen in x_1 speichert.
Es gilt:

$$S(x_1, \ldots, x_n) \hat{=} M(x_1, \ldots, x_n)\ S(x_2, \ldots, x_n)$$

(erst M, dann S) und

$$M(x_1, \ldots, x_n) = A(x_{n-1}, x_{n-2}) \ldots A(x_2, x_3)\ A(x_1, x_2).$$

Nach dem Schritt $A(x_{n-1}, x_{n-2})$ ist die kleinere Zahl in x_{n-1} gespeichert usw.
Wir haben also:

$S(x_1, \ldots, x_n) = A(x_{n-1}, x_n) \ldots A(x_1, x_2)\, S(x_2, \ldots, x_n)$ mit der Randbedingung: $S(x_{n-1}, x_n) \hat{=} A(x_{n-1}, x_n)$.

Sei a_n die Anzahl der Vergleiche, die notwendig sind bei der Prozedur $S(x_1, \ldots, x_n)$ für n Register, so haben wir die Rekurrenzrelation

$$a_n = n - 1 + a_{n-1},\ a_1 = 0.$$

Dies ist eine lineare, inhomogene Rekurrenzrelation und kann nach den uns bekannten Methoden gelöst werden. In diesem Fall bietet sich jedoch eine einfache Lösung an, indem man beobachtet:

$$\begin{aligned} a_n &= (n-1) + a_{n-1} = (n-1) + (n-2) + a_{n-2} \\ &= (n-1) + (n-2) + \ldots + 2 + 1 \\ &= \Sigma_{i=1\ldots n-1}\ i \\ &= n(n-1)/2. \end{aligned}$$

Offensichtlich ist der Algorithmus von polynomialer Komplexität, denn mit n = Anzahl der Register ist $n(n-1)/2 = n^2/2 - 1/2$ ein Polynom 2. ten Grades. Die Komplexität ist also in der Größenordnung $O(n^2)$.

Von dem hier dargestellten Sortierverfahren gibt es mehrere Versionen, die alle mit derselben Anzahl von Rechenschritten auskommen. Die bekannteste Version ist wohl der BUBBLE SORT, der so heißt, weil kleine Zahlen in den Registern gleichsam wie Blasen aufsteigen. Kurz beschrieben, sieht der Bubble Sort so aus:

```
for m:=2 to n do
    for i:=n step -1 to m do
        if x_i < x_{i-1} then interchange x_i, x_{i-1}.
```

17.2: SORTIERVERFAHREN VON BOSE-NELSON

Beschreibung:

Wir bezeichnen die Prozedur mit $T(x_1, \ldots, x_m)$, wobei $m = 2^k$ vorausgesetzt ist. Sei $P[(x_1, \ldots, x_n), (x_{n+1}, \ldots, x_{2n})]$ eine Prozedur, die die Zahlen in den Registern $x_1, \ldots, x_{2n}$ aufsteigend ordnet, unter der Voraussetzung, daß $(x_1, \ldots, x_n)$ und $(x_{n+1}, \ldots, x_{2n})$ schon geordnet sind.

Es gilt:

$$T(x_1, \ldots, x_{2n}) \hat{=} T(x_1, \ldots, x_n)\ T(x_1, \ldots, x_{2n})\ P[(x_1, \ldots, x_n), (x_{n+1}, \ldots, x_{2n})].$$

Sortiere also erst n Zahlen in $x_1, \ldots, x_n$ und $x_{n+1}, \ldots, x_{2n}$ und bilde die Vereinigung der sortierten Listen. P kann rekursiv definiert werden:

$$P[(x_1, \ldots, x_n), (x_{n+1}, \ldots, x_{2n})]$$

$$\hat{=} P[(x_1, \ldots, x_{n/2})\ (x_{n+1}, \ldots, x_{3n/2})] \qquad (*)$$

$$P[(x_{n/2+1}, \ldots, x_n)\ (x_{3n/2+1}, \ldots, x_{2n})]$$

$$P[(x_{n/2+1}, \ldots, x_n)\ (x_{n+1}, \ldots, x_{3n/2})].$$

1) Die n/2 kleinsten Zahlen werden in $x_1, \ldots, x_{n/2}$ gespeichert.
2) Die n/2 größten Zahlen werden in $x_{3n/2+1}, \ldots, x_{2n}$ gespeichert.
3) Die n restlichen Zahlen werden in $x_{n/2}, \ldots, x_{3n/2}$ gespeichert.

Es gilt:

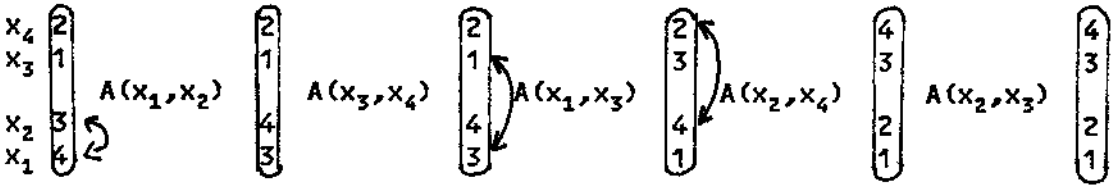
$$T(x_i, x_j) \hat{=} P[(x_i), (x_j)] \hat{=} A(x_i, x_j).$$

T läßt sich also als Hintereinanderausführung von Fundamentalbefehlen darstellen.

Beispiel:

$$\begin{aligned} &T(x_1, x_2, x_3, x_4) \\ &\quad \hat{=} T(x_1, x_2)\ T(x_3, x_4)\ P[(x_1, x_2)(x_3, x_4)] \\ &\quad \hat{=} A(x_1, x_2)\ A(x_3, x_4)\ P[(x_1), (x_3)]\ P[(x_2), (x_4)]\ P[(x_2), (x_3)] \\ &\quad \hat{=} A(x_1, x_2)\ A(x_3, x_4)\ A(x_1, x_3)\ A(x_2, x_4)\ A(x_2, x_3). \end{aligned}$$

Sei in (x_1, x_2, x_3, x_4) zu Anfang (4,3,1,2) gespeichert.



Sei c_r die Anzahl der Vergleiche der Prozedur

$$P[(x_1, \ldots, x_{2^r}), (x_{2^r+1}, \ldots, x_{2^{r+1}})].$$

Es gilt $c_r = 3\, c_{r-1}$ wegen (*).
Wir sehen, daß man für $P[(x_1, x_2), (x_3, x_4)]$ 3 Vergleiche braucht, also ist der Startwert $c_1 = 3$.

Als Lösung von (*) ergibt sich also $c_r = 3^r$.

Sei b_r die Anzahl der Vergleiche der Prozedur $T(x_1, \ldots, x_{2^{r+1}})$, so gilt $b_r = 2\, b_{r-1} + c_{r-1}$, also

$$b_r = 2\, b_{r-1} + 3^{r-1}.$$

Die homogene Rekurrenzrelation liefert den Ansatz

$$b_r^{(h)} = A\, 2^r.$$

Für die partikuläre Lösung setzen wir nach Tabelle von 15D

$$b_r^{(p)} = P\, 3^{r-1}.$$

Wir setzen ein und erhalten:

$$P\, 3^{r-1} = 2\, P\, 3^{r-2} + 3^{r-1}$$

und daraus $3\, P = 2\, P + 3$, also $P = 3$.

Also gilt: $b_r = b_r^{(h)} + b_r^{(p)} = A\, 2^r + 3^r$.

Für $b_1 = 5$ ergibt sich $A = 1$, somit als Lösung:

$$b_r = 2^r + 3^r.$$

Für eine Komplexitätsaussage genügt es, 3^r zu betrachten unter der Voraussetzung $n = 2^r$. Man hat dann $r = \log_2 n$ und

$$O(3^{\log_2 n}) = O(2^{\log_2 3 \log_2 n})$$
$$= O(n^{\log_2 3}) = \text{ungefähr } O(n^{1.6}).$$

17.3: Das Sortierverfahren ODD-EVEN-MERGE

Wir bezeichnen das Verfahren mit $U(x_1, \ldots, x_n)$ und setzen wiederum voraus, daß $n = 2^r$ eine Potenz von 2 ist.
Wir definieren rekursiv:

$$U(x_1, \ldots, x_{2n}) \hat{=} U(x_1, \ldots, x_n)\ U(x_{n+1}, \ldots, x_{2n})\ B[(x_1, \ldots, x_n)\ (x_{n+1}, \ldots, x_{2n})].$$

Dabei wird die Liste $x_1, \ldots, x_n$ sortiert, danach die Liste $x_{n+1}, \ldots, x_{2n}$, und dann werden mit der Prozedur B die Listen so vereinigt, daß alles sortiert ist.
Bezeichnet d_r die Anzahl der Vergleiche für die Prozedur $U(x_1, \ldots, x_{2n})$ und c_{r-1} die Anzahl der Vergleiche für $B[(x_1, \ldots, x_n)\ (x_{n+1}, \ldots, x_{2n})]$, so haben wir die Rekurrenzrelationen

$$d_r = 2\,d_{r-1} + c_{r-1} \qquad (*)$$

und

$$B[(x_1, \ldots, x_{2n}), (y_1, \ldots, y_{2n})] \hat{=} B[(x_1, \ldots, x_{2k+1}, \ldots, x_{2n-1}), (y_1, \ldots, y_{2k+1}, \ldots, y_{2n-1})] * B[(x_2, \ldots, x_{2k}, \ldots, x_{2n}), (y_2, \ldots, y_{2k}, \ldots, y_{2n})] * A(x_2, x_3)\ A(x_4, x_5)\ \ldots\ A(x_{2n}, y_1)\ \ldots\ A(y_{2n-2}, y_{2n-1}).$$

Die Listen $x_1, \ldots, x_{2n}$ und $y_1, \ldots y_{2n}$ werden halbiert in zwei Listen von Registern mit geraden bzw. ungeraden Nummern. Für die Rekurrenzrelation sieht man, daß es n Vergleiche $A(x_2, x_3)$, $A(x_4, x_5)$, $\ldots$, $A(x_{2n}, y_1)$ gibt, sowie n-1 Vergleiche $A(y_2, y_3), \ldots, A(y_{2n-2}, y_{2n-1})$, also $n+n-1 = 2^{r+1}-1$ Vergleiche $(n=2^r)$. Also ergibt sich als Rekurrenzrelation $c_{r+1} = 2c_r + 2^{r+1}-1$, d.h.

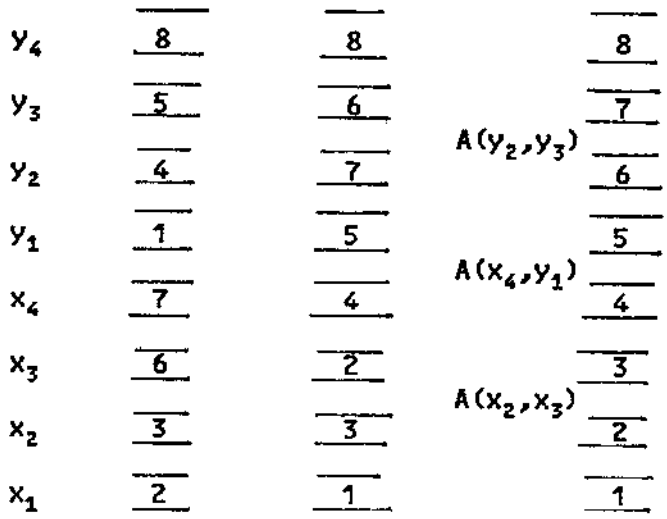
$$c_r = 2c_{r-1} + 2^r - 1. \qquad (**)$$

Bevor wir die beiden Rekurrenzrelationen lösen, möchten wir die Vereinigung der beiden Listen an folgendem Beispiel illustrieren:

$$x = (x_1, x_2, x_3, x_4) = (2, 3, 6, 7)$$
$$y = (y_1, y_2, y_3, y_4) = (1, 4, 5, 8).$$

Wir haben

$$B[(x_1, x_2, x_3, x_4), (y_1, y_2, y_3, y_4) \hat{=} B[(x_1, x_3), (y_1, y_3)]\ B[(x_2, x_4), (y_2, y_4)]\ A(x_2, x_3)\ A(x_4, y_1)\ A(y_2, y_3)$$



odd-even-merge

Man sieht, daß wir 9 Vergleiche benötigen, daß also $c_2 = 9$ ist. Zunächst lösen wir nun (**) auf. Der Ansatz für die homogene Gleichung ist

$$c_r^{(h)} = A\, 2^r.$$

Für die inhomogene Gleichung machen wir den Ansatz

$$c_r^{(p)} = p_1\, r\, 2^r + p_0$$

und erhalten durch Einsetzen in (**)

$$p_1\, r\, 2^r + p_0 = 2\,[\,p_1(r-1)2^{r-1} + p_0] + 2^r - 1$$
$$= p_1\, r\, 2^r - p_1\, 2^r + 2p_0 + 2^r - 1,$$

also

$$p_1\, 2^r - p_0 = 2^r - 1.$$

Aus dem Koeffizientenvergleich ergibt sich $p_1 = p_0 = 1$.
Für den Startwert $c_2 = 9$ erhält man aus

$$c_r = c_r^{(h)} + c_r^{(p)} = A\, 2^r + r\, 2^r + 1$$
$$9 = A\, 2^2 + 2*2^2 + 1$$

$A = 9$. Wir haben also $c_r = r\, 2^r + 1$.

Für die Rekurrenzrelation (*) ist der Ansatz für die homogene Lösung $d_r^{(h)} = D\ 2^r$. Für die partikuläre Lösung machen wir den Ansatz $d_{r+1}^{(p)} = (Ar^2 + Br + C)\ 2^r + E$ und setzen in $d_{r+1} = 2\ d_r + c_r$ ein:

$$(Ar^2 + Br + C)2^r + E = 2(A(r-1)^2 + B(r-1) + C)2^{r-1} + 2E + r2^r + E$$

$$\Leftrightarrow Ar^2 2^r + Br2^r + C2^r + E = (Ar^2 - 2Ar + A + Br - B + C)2^r + 2E + r2^r + 1$$

$$\Leftrightarrow Ar^2 2^r + Br2^r + C2^r + E = Ar^2 2^r - 2Ar2^r + A2^r + Br2^r - B2^r + C2^r + r2^r + 1$$

$$\Leftrightarrow A(2r2^r - 2^r) + B2^r - E = r2^r + 1 \Leftrightarrow A(2r-1) + B = r,\ E = 1.$$

Setze $B = A$, dann gilt $A(2r-1) + A = r \Rightarrow A(2r-1+1) = r \Rightarrow A = B = 1/2$ und $C = 0$.
Wir erhalten also

$$d_{r+1}^{(p)} = (1/2\ r^2 + 1/2\ r)\ 2^r + 1$$

und für den Startwert $d_3 = 4 + 9 = 13$ ($d_3 = d_{2+1}$), woraus für D folgt:

$$13 = D\ 2^3 + (1/2 * 4 + 1/2 * 2)\ 2^2 + 1$$

$$\Rightarrow 12 = D*8 + 3*4 \Rightarrow D = 0.$$

Also ergibt sich insgesamt

$$d_{r+1} = (r^2 + r)\ 2^{r-1} + 1.$$

Um die Komplexität grob abzuschätzen, genügt es, $r^2 2^r$ zu betrachten unter der Voraussetzung $n = 2^r$, d.h. $r = \log_2 n$. Wir erhalten also eine Komplexität von $O(n(\log_2 n)^2)$.

Kapitel 18 : Erzeugende Funktionen

Erzeugende Funktionen kann man außer bei vielen anderen kombinatorischen Problemen auch zur Auflösung von Rekurrenzrelationen benutzen. Die meisten Rekurrenzrelationen von a_r können nämlich in eine Gleichung unter Benutzung der erzeugenden Funktion

$$f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_rx^r + \ldots$$

verwandelt werden.
Eine solche Funktionalgleichung für $f(x)$ kann häufig algebraisch gelöst und die Lösung als Potenzreihe entwickelt werden, wobei man a_r als Koeffizient von x^r erhält. Für lineare homogene Rekurrenzrelationen läßt sich das stets anwenden, wie folgendes Beispiel zeigt:

Sei $(a_r) = a_0, a_1, \ldots, a_r, \ldots$ eine Folge von Zahlen, die der Rekurrenzrelation $a_r + a_{r-1} - 16a_{r-2} + 20a_{r-3} = 0$ genügt, wobei $a_0 = 0$, $a_1 = 1$, $a_2 = -1$ die Startwerte sind.

Sei nun $f(x) = a_0 + a_1x + \ldots + a_rx^r + \ldots$ die erzeugende Funktion für (a_r). Wir addieren folgende vier Gleichungen:

$$\begin{aligned}
f(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_rx^r + \ldots \\
xf(x) &= a_0x + a_1x^2 + a_2x^3 + \ldots + a_{r-1}x^r + \ldots \\
-16x^2f(x) &= -16a_0x^2 - 16a_1x^3 - \ldots - 16a_{r-2}x^r - \ldots \\
20x^3f(x) &= 20a_0x^3 + \ldots + 20a_{r-3}x^r + \ldots
\end{aligned}$$

und erhalten:

$$\begin{aligned}
(1+x-16x^2+20x^3)f(x) &= a_0 + (a_1+a_0)x + (a_2+a_1-16a_0)x^2 \\
&\quad + (a_3+a_2-16a_1+20a_0)x^3 + \ldots + \\
&\quad + (a_r+a_{r-1}-16a_{r-2}+20a_{r-3})x^r + \ldots
\end{aligned}$$

Nun ist aber $a_r + a_{r-1} - 16a_{r-2} + 20a_{r-3} = 0$ für $r=3, 4, 5, \ldots$

Daher erhalten wir

$$(1+x-16x^2+20x^3)\ f(x) = a_0 + (a_1+a_0)x + (a_2+a_1-16a_0)x^2.$$

Wir setzen die Startwerte ein: $(1 + x - 16x^2 + 20x^3)\ f(x) = x$, d. h. es gilt

$$f(x) = x/(1+x-16x^2+20x^3).$$

Nun hat man die Zerlegung $(1+x-16x^2+20x^3) = (1-2x)^2\ (1+5x)$, wir setzen daher für Konstante A, B, C:

$$\frac{x}{1+x-16x^2+20x^3} = \frac{A}{1-2x} + \frac{B}{(1-2x)^2} + \frac{C}{1+5x}$$

Um A, B und C zu bestimmen, multipliziert man beide Seiten mit dem Nenner $(1+x-16x^2+20x^3)$ und erhält $x = (A+B+C) + (3A+5B-4C)x + (-10A+4C)x^2$,

woraus sich das Gleichungssystem

$$\begin{aligned} A + B + C &= 0 \\ 3A + 5B - 4C &= 1 \\ -10A \quad + 4C &= 0 \end{aligned}$$

mit den Lösungen $A = 2/49$, $B = 7/49$, $C = -5/49$ ergibt.

Ferner entwickeln wir die linke Seite in Potenzreihen. Dabei ist:

$$1/(1-2x) = \Sigma_{k=0..\infty}\ 2^k x^k.$$

$$1/(1-2x)^2 = \Sigma_{k=0..\infty}\ (k+1)2^k x^k.$$

$$1/(1+5x) = \Sigma_{k=0..\infty}\ (-5)^k x^k.$$

Daraus ergibt sich als erzeugende Funktion:

$$f(x) = -2/49\ (\Sigma_{k=0..\infty} 2^k x^k)$$
$$+7/49\ (\Sigma_{k=0..\infty}(k+1)2^k x^k)\ -5/49\ (\Sigma_{k=0..\infty}(-5)^k x^k)$$
$$= \Sigma_{k=0..\infty}\ [(-2/49)2^k + (7/49)(k+1)2^k - (5/49)(-5)^k]\ x^k.$$

Daraus erhält man sofort die Lösung der Rekurrenzrelation, nämlich

$$a_r = -(2/49)2^r + (7/49)(r+1)2^r - (5/49)(-5)^r.$$

Das obige Verfahren erinnert uns etwas an die Methode, die Lösung mit der charakteristischen Gleichung zu finden. Offensichtlich sind

2, 2, -5 die Wurzeln der charakteristischen Gleichung $x^3 + x^2 - 16x + 20 = 0$. Wenn wir x durch 1/x ersetzen, erhalten wir

$$r(1/x) = 1/x^3 + 1/x^2 - 16/x + 20 = 0.$$

Multiplizieren wir mit x^3, so ergibt sich

$$q(x) = 1 + x - 16x^2 + 20x^3.$$

Damit zeigt es sich, daß die erzeugende Funktion f

$$f(x) = p(x)/q(x)$$

als Quotient zweier Polynome p(x) = x und q(x) dargestellt worden ist.

Satz (18.1):

Erfüllt die Folge (a_r) die lineare homogene Rekurrenzrelation

$$a_r + c_1 a_{r-1} + \ldots + c_k a_{r-k} = 0,$$

dann ist die erzeugende Funktion f(x) von (a_r) von der Form

$$f(x) = p(x)/q(x),$$

wobei $q(x) = x^k\ r(1/x)$ (r(x) das charakteristische Polynom) und

$p(x) = a_0 + (a_1+c_1a_0)x + \ldots + (a_{k-1}+c_1a_{k-2}+ \ldots + c_{k-1}a_0)\ x^{k-1}$ ist.

Der Beweis zu diesem Satz ergibt sich aus der oben angegebenen Methode.

Dieses Verfahren läßt sich (sogar noch besser) für nicht-lineare Rekurrenzrelationen einsetzen.

Beispiel (18.2):
Auf wieviel Arten kann man das Produkt $k_1 * \ldots * k_n$ beklammern?

Wir haben $a_2 = 1$ für (k_1k_2) und $a_3 = 2$ für $((k_1k_2)k_3)$ und $(k_1(k_2k_3))$. Außerdem benutzen wir als Startwerte $a_0 = 2$ und $a_1 = 1$, um eine einfache Form für die Rekurrenzrelation zu erhalten.

Um die Rekurrenzrelation zu finden betrachte

$$((k_1 \ldots k_i)*(k_{i+1} \ldots k_n), \text{ für } i=1, \ldots, n-1.$$

Die Anzahl der Arten, die Teilprodukte zu beklammern, ist a_i und a_{n-i}, insgesamt also $a_i * a_{n-i}$. Damit ergibt sich die Rekurrenzrelation

$$a_n = a_1a_{n-1} + a_2a_{n-2} + \ldots + a_{n-1}a_1.$$

Betrachtet man a_n als Koeffizient der erzeugenden Funktion

$$f(x) = 0 + x + \Sigma_{n=2..\infty}(a_1a_{n-1} + \ldots + a_{n-1}a_1)x^n,$$

so ergibt sich:

$$f(x)\,f(x) = (\,0 + a_1x + \ldots + a_nx^n + \ldots\,)^2$$

$$= 0 + \Sigma_{n=2..\infty}(a_1a_{n-1} + \ldots + a_{n-1}a_1)\,x^n$$

$$= f(x) - x.$$

Daraus folgt $f(x)^2 - f(x) + x = 0$.
Diese Funktionalgleichung wird wie eine Gleichung 2. ten Grades behandelt, wobei man $y = f(x)$ setzt und $y^2 - y + x = 0$ löst.
Als Lösung erhält man $f(x) = 1/2\,(1 +- \mathrm{sqrt}(1-4x))$.

sqrt(1-4x) entwickelt man nun wieder in eine erzeugende Funktion:

$$\text{sqrt}(1-4x) = \sum_{n=0..\infty} \binom{1/2}{n} (-4x)^n = \sum_{n=0..\infty} b_n x^n .$$

Dies ist eine Entwicklung nach verallgemeinerten Binomialkoeffizienten. Es gilt:

$$b_n = \binom{1/2}{n} (-4)^n = \frac{1/2\ (1/2-1)\ (1/2-2)\ \ldots\ (1/2-n+1)}{n!} (-4)^n$$

$$= - \frac{1\ 3\ 5\ \ldots\ (2n-3)}{n!} 2^n = - \frac{2}{n} \binom{2n-2}{n-1}$$

Damit erhalten wir für a_n als Koeffizient von f(x)= 1/2 (1-sqrt(1-4x))

$$a_n = \frac{1}{n} \binom{2n-2}{n-1}$$

Die Zahlen dieser Form heißen CATALANISCHE ZAHLEN.

Im folgenden geben wir für einige Folgen jeweils die erzeugende Funktion an:

Folgen	Erzeugende Funktion
$a_r = 1$	$1/(1-x)$
$a_r = r+1$	$1/(1-x)^2$
$a_r = 1/r,\ a_0 = 0$	$\log(1/(1-x))$
$a_r = \binom{n}{r}$	$(1+x)^r$
$a_r = 1/r!$	e^x
$a_r = n^r/r!$	e^{nx}

Das Sortierverfahren ODD-EVEN-MERGE

Für $n = 2^r$, $r \geq 0$, seien folgende Prozeduren definiert:

$U(x_1, \ldots, x_n)$ sortiert $x_1, \ldots, x_n$ bzgl. '<'.

$B((x_1, \ldots, x_n)(y_1, \ldots, y_n))$ mischt die sortierten Listen $(x_1, \ldots, x_n)$, $(y_1, \ldots, y_n)$ zu einer sortierten Liste $(z_1, \ldots, z_{2n})$.

$$A(x,y) = \begin{cases} (x,y), & \text{falls } x \leq y \\ (y,x), & \text{falls } y < x \end{cases}$$ Vergleichsoperator

u_r bzw. b_r sei die Anzahl der Vergleichsoperationen in U bzw. B.

a) B wird rekursiv definiert durch:

i) $B((x),(y)) := A(x,y)$

ii) $B((x_1, \ldots, x_n)(y_1, \ldots, y_n) :=$

$$B((x_1, x_3, \ldots, x_{n-1})(y_1, y_3, \ldots, y_{n-1})) * B((x_2, x_4, \ldots, x_n)(y_2, y_4, \ldots, y_n)) * \ldots * A(x_2, x_3)\, A(x_2, x_5) * \ldots * A(x_n, y_1) * A(y_2, y_3) * \ldots * A(y_{n-2}, y_1)$$

für $n = 2^r$, $r \geq 1$.

Damit gilt: $b_0 = 1$, $b_1 = 3$, $b_2 = 9$ und allgemein: $b_r = 2\, b_{r-1} + 2^r - 1$, für $r \geq 1$.

b) U wird rekursiv definiert durch:

i) $U(x_1) := (x_1)$

ii) $U(x_1, \ldots, x_n) :=$

$$U(x_1, \ldots, x_{n/2})\; U(x_{(n/2)+1}, \ldots, x_n) * B((x_1, \ldots, x_{n/2})(x_{(n/2)+1}, \ldots, x_n)),$$

für $n = 2^r$, $r \geq 1$.

Damit ist $u_0 = 0$, $u_1 = 1$, $u_2 = 5$, $u_3 = 19$, $u_4 = 2*19+25 = 63$, und allgemein: $u_r = 2\, u_{r-1} + b_{r-1}$, $r \geq 1$.

c) Lösung der Rekurrenzrelation für b_r: $b_r = r\, 2^r + 1$, für $r \geq 0$, denn:

i) $b_0 = 1 = 0\, 2^0 + 1$.

ii) $(r>0)$: $b_r = 2\, b_{r-1} + 2^r - 1 = 2((r-1)2^{r-1} + 1) + 2^r - 1$

$$= 2r\, 2^{r-1} - 2\, 2^{r-1} + 2 + 2^r - 1 = r\, 2^r + 1.$$

d) Lösung der Rekurrenzrelation für u_r:

Homogen: $u_r^{(h)} = E\, 2^r$.

Partiell: $u_r^{(p)} = (Ar^2+Br+C)\, 2^r + D$, $r \geq 0$.

Einsetzen in $u_r = 2\, u_{r-1} + b_{r-1}$ liefert

$$(Ar^2+Br+C)\, 2^r + D = 2\,(A(r-1)^2 + B(r-1) + C)\, 2^{r-1} + D) + ((r-1)2^{r-1} + 1)$$

<=>

$$2^r(A(r^2-(r-1)^2) + B(r-(r-1)) + (C-C)) = D + 1 + (r-1)\,2^{r-1}$$

$\Rightarrow D = -1$. $r=1$ ergibt: $2\,(A(2r-1) + B) = r-1$

$\Leftrightarrow r(4A-1) + (-2A+2B+1) = 0$

und daraus folgt $A = 1/4$ und daraus $B = -1/4$.

Also haben wir: $u_r = (r^2/4 - r/4 + F)\,2^r - 1$ mit $F:=C+E$.

Startwert $u_0 = 0$ eingesetzt ergibt: $0 = (0/4 - 0/4 + F) - 1$
$\Rightarrow F = 1$. Die Lösung der Rekurrenzrelation lautet also:

$$u_r = (r^2/4 - r/4 + 1)\,2^r - 1,\ r \geq 0.$$

Für die Anzahl der Vergleichsoperationen des Sortierverfahrens Odd-even Merge ergibt sich damit nun:

$$d_{r+1} = (r^2/2 + r/2 + 2)\,2^r - 1,$$

bzw.

$$d_r = (\,(r-1)^2/2 + (r-1)/2 + 2\,)\,2^{r-1} - 1$$

$$= 1/4\,((r-1)^2 + r + 4)\,2^r - 1$$

$$= 1/4\,(r^2 - 2r + 1 + r + 4)\,2^r - 1$$

$$= (r^2/4 - r/4 + 1)\,2^r - 1.$$

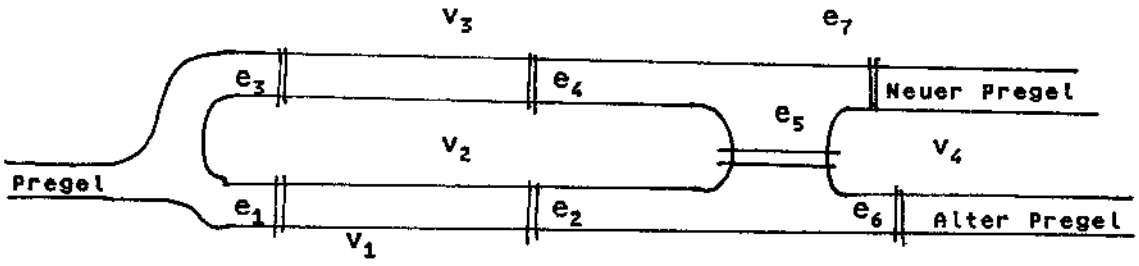
Um die Komplexität grob abzuschätzen, genügt es, $r^2 2^r$ zu betrachten unter der Voraussetzung $n = 2^r$, d. h. $r = \log_2 n$.

Wir erhalten: $O(n(\log_2 n)^2)$.

II GRUNDBEGRIFFE UND ALGORITHMEN DER GRAPHENTHEORIE

Kapitel 19 : Graphen

Auf folgender Skizze wird Königsberg mit dem Fluß Pregel und sieben Brücken dargestellt.



Im Jahre 1736 befaßte sich Euler mit der Frage, ob man genau einmal über jede dieser Brücken gehen und dabei zum Ausgangspunkt zurückkommen könne.
Zunächst ist es schwierig zu erkennen, daß in diesem Zusammenhang Mathematik überhaupt eine Rolle spielt. Häufig ist es eine der wichtigsten Leistungen, für Probleme die adäquate mathematische Form zu finden. Der Leser wird in diesem Buch zumeist vollständig dieser Aufgabe enthoben, denn im allgemeinen werden erst die mathematischen Begriffsbildungen und Theorien und danach die Probleme präsentiert.

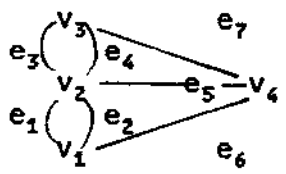
19A : Grundbegriffe

Definition (19.1):

Ein GRAPH G ist ein geordnetes Tripel $G = (V, E, \Phi)$, das aus einer nicht leeren Menge $V = V(G)$ von ECKEN (KNOTEN), einer davon disjunkten Menge $E = E(G)$ von KANTEN und einer INZIDENZFUNKTION $\Phi = \Phi_G$ besteht. Die Inzidenzfunktion Φ ordnet jeder Kante e ein ungeordnetes Paar a b nicht unbedingt verschiedener Ecken zu. Wir schreiben $\Phi(e) = a\ b$ und nennen a und b die Ecken der Kante e.

Beispiel: [KÖNIGSBERGER BRÜCKENPROBLEM]
In einem ersten Schritt mathematisieren wir das obige Problem folgendermaßen. Die Brücken werden als Kanten und die durch den Fluß

getrennten Gebiete als Ecken dargestellt. Wir haben also:



$$V = \{v_1, v_2, v_3, v_4, \}$$
$$E = \{e_1, \ldots, e_7\}$$
$$\Phi(e_1) = \Phi(e_2) = v_1\ v_2$$
$$\Phi(e_3) = \Phi(e_4) = v_2\ v_3$$
$$\Phi(e_5) = v_2\ v_4$$
$$\Phi(e_6) = v_1\ v_4$$
$$\Phi(e_7) = v_3\ v_4$$

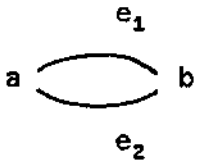
Die graphische Darstellung gibt jede Information über den obigen Graphen anschaulich, knapp und präzise wieder.

Bezeichnungen (19.2):

Seien a, b zwei Ecken des Graphen G, die durch die Kante e verbunden sind, d. h. $\Phi(e) = a\ b$. Die Kante e heißt INZIDENT mit den Ecken a und b, umgekehrt heißen die Ecken a und b INZIDENT mit der Kante e. Zwei Ecken a, b, die inzident mit einer gemeinsamen Kante sind, heißen BENACHBART oder ADJAZENT.
Eine Kante e mit $\Phi(e) = a\ b$ und $a = b$ heißt SCHLINGE (SCHLEIFE).
Zwei Kanten e_1, e_2 mit $\Phi(e_1) = \Phi(e_2) = a\ b$ heißen MEHRFACHKANTEN.

Schlinge

Mehrfachkanten



Definition (19.3):

Ein Graph G heißt ENDLICH, wenn die Eckenmenge V und die Kantenmenge E endlich sind. Dann wird die Anzahl der Ecken mit $\nu = \nu(G)$ und die Anzahl der Kanten mit $\epsilon = \epsilon(G)$ bezeichnet.

Wir treffen die Vereinbarung, daß alle Graphen in den folgenden Kapiteln als endlich vorausgesetzt sind, ohne daß dies jedesmal erwähnt wird. Dagegen werden wir angeben, wenn auch unendliche Graphen zugelassen sind.
Im übrigen kann der Leser selbst untersuchen, welche der Behauptungen auch für unendliche Graphen zutreffen.

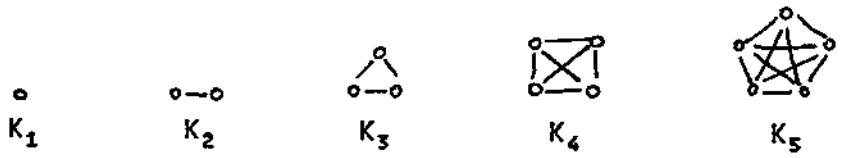
Definition (19.4):

Ein Graph G heißt SCHLICHT (EINFACH), wenn G keine Schlingen und keine Mehrfachkanten hat.

Definition (19.5):

Ein schlichter Graph G, bei dem jedes Paar verschiedener Ecken durch eine Kante verbunden ist, heißt VOLLSTÄNDIG.

Beispiel (19.6):
Alle vollständigen Graphen bis zu 5 Ecken sind:

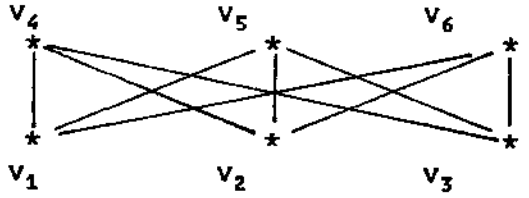


Der Graph K_1 heißt auch der TRIVIALE GRAPH.

Definition (19.7):

Ein Graph G heißt BIPARTIT, wenn sich die Eckenmenge V(G) in zwei disjunkte Teilmengen X und Y so zerlegen läßt, daß jede Kante eine Ecke in X und die andere Ecke in Y hat. (X, Y) heißt BIPARTITION des Graphen.
Ein bipartiter Graph G heißt VOLLSTÄNDIGER BIPARTITER GRAPH, wenn jede Ecke in X mit jeder Ecke in Y verbunden ist. Ist $|X| = l$ und $|Y| = m$, so wird der Graph mit $K_{l,m}$ bezeichnet.

Beispiel (19.8):
$K_{3,3}$ ist ein vollständiger, bipartiter Graph mit $|X| = |Y| = 3$.



19B : Isomorphismen

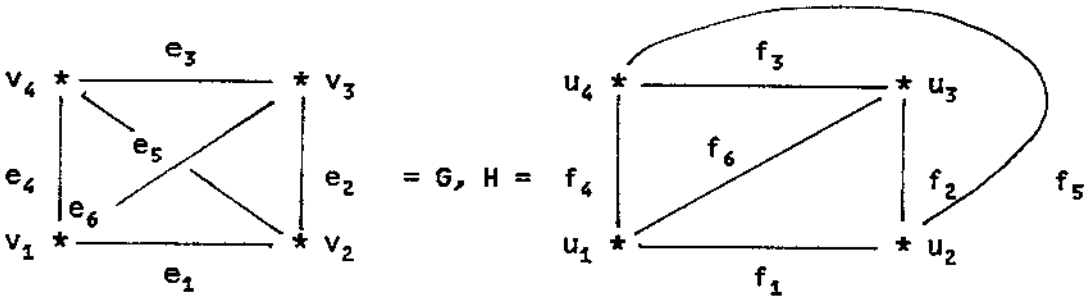
Definition (19.9):

Zwei Graphen G und H heißen ISOMORPH, wenn es bijektive Abbildungen $\alpha : V(G) \longrightarrow V(H)$ und $\beta : E(G) \longrightarrow E(H)$ gibt, so daß

$$\Phi_G(e) = u\,v \quad \Leftrightarrow \quad \Phi_H(\beta(e)) = \alpha(u)\,\alpha(v).$$

Das Paar (α, β) von Abbildungen heißt dann ISOMORPHISMUS zwischen den Graphen G und H.

Beispiele:
a) Der Graph G ist isomorph zu dem Graphen H:

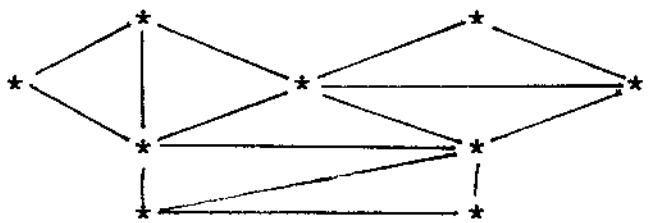


b) Der Graph G ist nicht zum Graphen H isomorph:

Definition (19.10):

Ein Graphisomorphismus (α, β) mit $\alpha : V(G) \longrightarrow V(G)$ und $\beta : E(G) \longrightarrow E(G)$ eines Graphen G heißt ein AUTOMORPHISMUS von G.
Die Automorphismen eines Graphen G bilden eine Gruppe.

Beispiele:
a) Der vollständige Graph K_n hat als Automorphismengruppe die symmetrische Gruppe S_n, d.h. die Gruppe aller Permutationen der Menge $\{1, \ldots, n\}$.
b) Ein Graph mit so wenig Ecken als möglich, der die zyklische Gruppe C_3 der Ordnung $|C_3| = 3$ als Automorphismengruppe hat, ist folgender:



c) Zu jeder endlichen Gruppe H gibt es einen endlichen Graphen G, der H als Automorphismengruppe hat.

19C : Untergraphen

Definition (19.11):

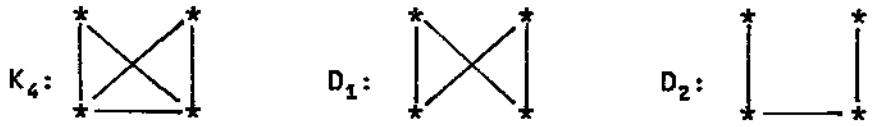
Ein Graph H ist ein UNTERGRAPH von G, wenn gilt:

1) $V(H) \subseteq V(G)$
2) $E(H) \subseteq \subseteq (G)$
3) $\Phi_H(e) = \Phi_G(e)$ für alle $e \in E(H)$.

Definition (19.12):

Ein Untergraph H heißt AUFSPANNENDER UNTERGRAPH von G, wenn $V(H) = V(G)$.

Beispiel:
Für den Graphen K_4 sind D_1 und D_2 zwei aufspannende Untergraphen:



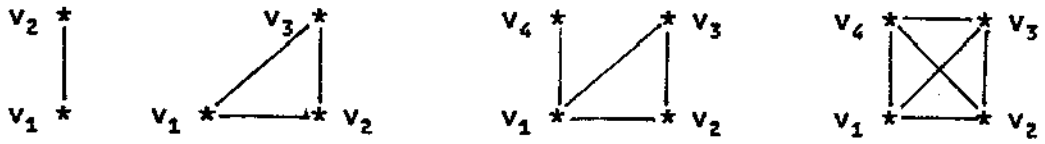
Definition (19.13):

Seien H_1 und H_2 Untergraphen von G. H_1 und H_2 heißen DISJUNKT, wenn $V(H_1) \cap V(H_2) = \emptyset$.
$H_1 \cup H_2$ ist der Untergraph von G mit der Eckenmenge $V(H_1) \cup V(H_2)$, der Kantenmenge $E(H_1) \cup E(H_2)$ und der Inzidenzfunktion

$$\Phi_{H_1 \cup H_2}(e) = \Phi_G(e)$$

für alle $e \in E(H_1) \cup E(H_2)$. Wenn H_1, H_2 disjunkt sind, schreiben wir für $H_1 \cup H_2$ auch $H_1 + H_2$.

Beispiel:
Für die Untergraphen H_1, H_2 des K_4 erhält man $H_1 \cup H_2$:

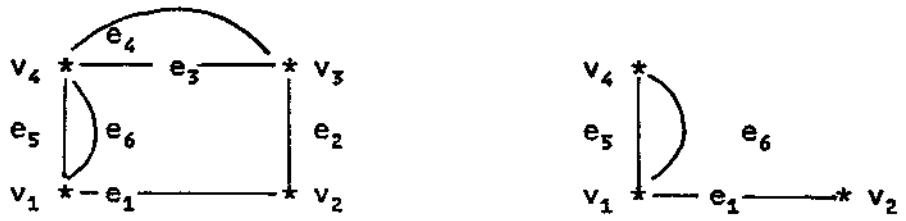


H_1 H_2 $H_1 \cup H_2$ K_4

Definition (19.14):

Sei G = (V,E,Φ) ein Graph und V' eine nicht leere Teilmenge von V. Der Untergraph von G, der aus allen Ecken von V' und allen Kanten von G besteht, deren Endecken in V' sind, wird mit G[V'] bezeichnet; er heißt der durch V' INDUZIERTE UNTERGRAPH von G.

Beispiel:
Für den Graphen G und V' = { v_1, v_2, v_4 } erhält man G[V']:



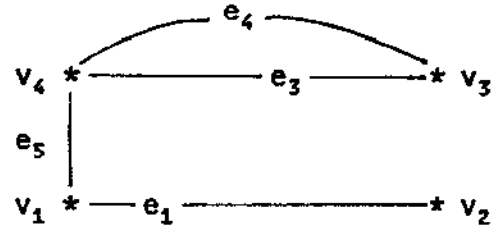
G G[V']

Bezeichnungen:
G[V-V'] wird mit G-V' bezeichnet.
G - v wird für G - { v} geschrieben.

Definition (19.15):

Sei G = (V, E, Φ) ein Graph und E' eine nicht leere Teilmenge von V. Der Untergraph von G, der aus allen Ecken, die mit den Kanten aus E' inzident sind, und aus allen Kanten aus E' besteht, wird mit G[E']bezeichnet; er heißt der durch E' KANTENINDUZIERTE UNTERGRAPH von G.

Beispiel:
Für den Graphen des vorhergehenden Beispiels und E' = { e_1, e_3, e_4, e_5} erhält man G[E']:

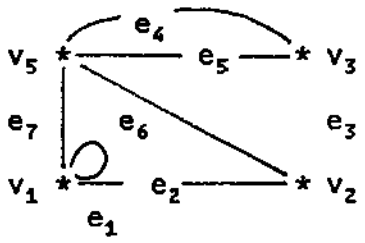


19D : Inzidenz- und Adjazenzmatrizen

Definition (19.16):

Sei G gegeben mit $V(G) = \{ v_1, \ldots, v_\nu \}$ und $E(G) = \{ e_1, \ldots, e_\varepsilon \}$. Die Inzidenzmatrix von G ist $M(G) = m_{ij}$, wobei die Zahl m_{ij} die Zahl angibt, wie oft v_i und e_j inzident sind. Bei einer Schleife mit einem Eckpunkt wird 2 gezählt.

Beispiel:



$\nu \times \varepsilon$ - Matrix:

	e_1	e_2	e_3	e_4	e_5	e_6	e_7
ν_1	2	1	0	0	0	0	1
ν_2	0	1	1	0	0	1	0
ν_3	0	0	1	1	1	0	0
ν_4	0	0	0	1	1	1	1

Man beachte, daß die Summe jeder Spalte $\Sigma_{i=1\ldots4}\, m_{ij} = 2$ beträgt.

Definition (19.17):

Sei G gegeben mit $V(G) = \{v_1, \ldots, v_\nu\}$. Die NACHBARSCHAFTS-MATRIX oder ADJAZENZMATRIX ist $A(G) = (a_{ij})$, wobei die Zahl a_{ij} angibt, mit wieviel Kanten die Ecken v_i und v_j inzident sind.

Für obiges Beispiel ergibt sich folgende $\nu \times \nu$ - Matrix A(G):

	ν_1	ν_2	ν_3	ν_4
ν_1	1	1	0	1
ν_2	1	0	1	1
ν_3	0	1	0	2
ν_4	1	1	2	0

Man beachte, daß die Adjazenzmatrix symmetrisch ist, und man daher nur eine Dreicksmatrix zu speichern braucht. Es ist klar, daß man aus dieser Matrix den Graphen eindeutig zurückgewinnt.

Es gibt einige Varianten zu diesen Darstellungen von Graphen, die wir auf später verschieben. Für die Entwicklung von Datenstrukturen für Graphen genügen bereits Inzidenz- oder Adjazenzmatrizen.

19E : Eckengrade

Definition (19.18):

Der ECKENGRAD $\delta(v) = \delta_G(v)$ einer Ecke v von G ist die Anzahl der Kanten, mit denen v inzident ist (jede Schleife zählt als 2 Kanten). $\delta(G)$ bezeichnet den kleinsten Eckengrad: $\delta(G) = \min \{\delta(v) \mid v \in V(G)\}$. $d(G)$ bezeichnet den größten Eckengrad: $d(G) = \max \{\delta(v) \mid v \in V(G)\}$.

Satz (19.19):

Es gilt in jedem Graphen G: $\Sigma_{v \in U} \delta(v) = 2\epsilon$, d.h. die Summe der Eckengrade eines Graphen ist gleich der doppelten Kantenanzahl.

Beweis:
Betrachte die Inzidenzmatrix M(G). Es gilt:

$$\Sigma_{j=1\ldots\epsilon} m_{ij} = \delta(v_i),$$

und daraus folgt: $\Sigma_{v \in U} \delta(v) = \Sigma_{i=1\ldots\nu, j=1\ldots\epsilon} m_{ij}$, also die Summe aller Zahlen in M(G).
Da die Summe der Zahlen in einer Spalte 2 ist, ist die Summe aller Zahlen in M(G) also gleich 2ϵ, womit die Behauptung gezeigt ist.

Korollar (19.20):

In jedem Graphen ist die Anzahl der Ecken mit ungeradem Eckengrad eine gerade Zahl.

Beweis:
Sei V_1 die Menge der Ecken mit ungeradem Eckengrad, $V_2 = V - V_1$.

Weil

$$\Sigma_{v \in U_1} \delta(v) + \Sigma_{v \in U_2} \delta(v) = 2\,\epsilon$$

und $\Sigma_{v \in U_2} \delta(v)$ gerade ist, folgt die Behauptung. da

Kapitel 20 : Wege, Zykel und kürzeste Wege

20A : Wege

Definition (20.1):

Eine KANTENFOLGE K von G ist eine Folge $K=v_0e_1v_1\ldots e_kv_k$, deren Ausdrücke abwechselnd Ecken und Kanten sind, so daß für $1 \leq i \leq k$ die Endecken von e_i genau v_{i-1} und v_i sind.

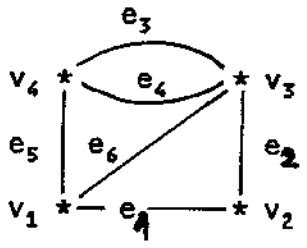
Definition (20.2):

Eine Kantenfolge L heißt eine LINIE, wenn die Kanten alle voneinander verschieden sind.

Definition (20.3):

Eine Linie W heißt ein WEG, wenn die Ecken alle voneinander verschieden sind.

Beispiel:



Kantenfolge : $v_2e_2v_3e_4v_4e_4v_3e_6v_1$

Linie : $v_2e_2v_3e_4v_4e_3v_3e_6v_1$

Weg : $v_2e_2v_3e_4v_4e_5v_1$

Bemerkungen (20.4):

1) In einem schlichten Graphen ist eine Kantenfolge (eine Linie, ein Weg) bereits durch die Angabe der Ecken bestimmt, $K = v_0v_1\ldots v_k$.
2) v_0 heißt der ANFANG von K, v_k heißt das ENDE von K, k heißt die LÄNGE von K.
3) Unter einem (u,v)-WEG versteht man einen Weg mit dem Anfang u und dem Ende v.

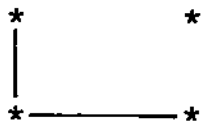
Definition (20.5):

Die Ecken u und v von G heißen ZUSAMMENHÄNGEND, wenn es einen (u,v)-Weg in G gibt. Sind alle Eckenpaare eines Graphen G zusammenhängend, so heißt G ZUSAMMENHÄNGEND.

Bemerkung (20.6):

Der Zusammenhang ist eine reflexive, symmetrische und transitive Relation auf der Menge der Ecken, d. h. eine Äquivalenzrelation. Die Äquivalenzklassen heißen ZUSAMMENHANGSKOMPONENTEN.

Beispiel:
G hat 2 Zusammenhangskomponenten:



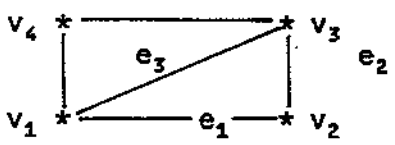
Bezeichnung:
w(G) ist die Anzahl der Zusammenhangskomponenten von G.

20B : Zykel

Definition (20.7):

Eine Kantenfolge, deren Anfangsecke und Endecke identisch sind, heißt GESCHLOSSEN. Jede Ecke außer Anfangsecke und Endecke wird als INNERE ECKE bezeichnet.
Eine geschlossene Linie, deren innere Ecken alle verschieden sind, heißt ZYKEL (KREIS). Ein Zykel der Länge k heißt auch K-ZYKEL. Ist k gerade, so sagt man auch GERADER ZYKEL, ansonsten UNGERADER ZYKEL.

Beispiel:
Ein 3-Zykel $v_1v_2v_3v_1$:



Satz(20.8):

Ein Graph ist genau dann bipartit, wenn G keine ungeraden Zykel hat.

Beweis:
'=>':
Sei G bipartit mit der Bipartition (X,Y) und sei $C = v_0v_1 \ldots v_kv_0$ ein (k+1)-Zykel von G. OBdA können wir annehmen, daß $v_0 \in X$. Da G bipartit ist, muß $v_1 \in Y$ sein, analog wieder $v_2 \in X$ usw. Wir haben also $v_{2i} \in X$, $v_{2i+1} \in Y$. Daher muß $k = 2i+1$ sein für ein i und damit ist der Zykel C gerade.

'<=':
Wir dürfen annehmen, daß G zusammenhängend ist (ansonsten beweise diese Richtung für jede Zusammenhangskomponente von G). G habe keine ungeraden Zykel.
Für zwei Ecken u und x bezeichne d(u,x) die Länge eines kürzesten (u,x)-Weges. Wir wählen u fest aus G und definieren:

$$X = \{ x \in V \mid d(u,x) \text{ gerade} \}$$
$$Y = \{ y \in V \mid d(u,x) \text{ ungerade} \}.$$

Nun müssen wir zeigen, daß je 2 Ecken $v, w \in X$ (bzw. $\in Y$) nicht benachbart sind (denn dann ist (X,Y) eine Bipartition).

Sei P ein kürzester (u,v)-Weg und Q ein kürzester (u,w)-Weg. Sei u_1 die letzte Ecke, die P und Q gemeinsam haben (sie starten ja an derselben Anfangsecke). Der (u,u_1)-Abschnitt von Q hat dieselbe Länge wie der (u,u_1)-Abschnitt von P, da P und Q je kürzeste Wege sind.

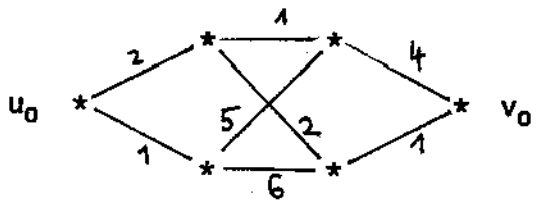
Da P und Q gerade Länge haben ($v,w \in X$), müssen der (u_1,v)-Abschnitt P_1 von P und der (u_1,w)-Abschnitt Q_1 von Q gleiche Parität haben, d. h. entweder beide gerade oder beide ungerade sein. Es folgt, daß der zusammengesetzte (v,w)-Weg $P_1^{-1}Q_1$ (erst P_1^{-1}, dann Q_1) gerade Länge hat. Wenn nun v und w benachbart wären, hätte der Zykel $P_1^{-1}Q_1v$ eine ungerade Länge, was nach Voraussetzung nicht sein kann. Also sind v und w nicht benachbart und (X,Y) ist die gesuchte Bipartition.

20C : Kürzeste Wege

Definition (20.9):

Sei G ein Graph. Eine Funktion w:E(G) ---> ℝ, die jeder Kante von E eine reelle Zahl zuordnet, heißt GEWICHTSFUNKTION. Ein Graph G mit einer Gewichtsfunktion heißt GEWICHTETER GRAPH.

Beispiel:



Definition (20.10):

In einem gewichteten Graphen versteht man unter der LÄNGE EINES (U,V)- WEGES die Summe der Gewichte über den Kanten des Weges.

Problem (20.11): [Das Kürzeste-Wege-Problem]

Es sei ein zusammenhängender, schlichter Graph G gegeben mit einer Gewichtsfunktion $w:E \longrightarrow \mathbb{R}^+$; $u_0, v_0 \in E(G)$. Bestimme den kürzesten Weg von u_0 nach v_0.

Der Algorithmus von Dijkstra löst dieses Problem mit folgenden Ideen:

1) Man sucht von u_0 aus kürzeste Wege zu allen Ecken des Graphen.

2) Sei $S_i = \{u_0, \ldots, u_i\} \subseteq V$ und $S_i^* = V - S_i$. Ist $P = u_0 \ldots u^* v^*$ ein kürzester Weg von u_0 nach S_i^*, dann ist $u^* \in S_i$ und der (u_0,u^*)-Abschnitt von P ein kürzester (u_0,u^*)-Weg.

   Bezeichnet $d(u_0,w)$ das Gewicht eines kürzesten (u_0,w)-Weges, so gilt:

$$d(u_0,v^*) = d(u_0,u^*) + w(u^*,v^*)$$

   und für das Gewicht $d(u_0,S_i^*)$ ergibt sich:

$$d(u_0,S_i^*) = \min \{ d(u_0,u) + w(u,v) \}, \quad (*)$$

   wobei das Minimum über alle $u \in S_i$ und $v \in S_i^*$ genommen wird.

3) Für jede Ecke von G führt man eine Markierungsfunktion (LABEL-FUNKTION) $l:V \longrightarrow \mathbb{R}^+$ ein. Dabei gilt zunächst für S_0, daß $l(u_0) = 0$ und $l(u_i) = \infty$ für $i \neq 0$ ist. Für S_i hat man

$$l(u) = d(u_0,u) \text{ für } u \in S_i.$$

Für die Ecke v^* von S_i^* bestimmt man $l(v^*)$ nun nach (*), wobei man einige Vergleiche noch einsparen kann.

(20.12): [ALGORITHMUS VON DIJKSTRA]

1) Setze $l(u_0) = 0$, $l(v) = \infty$ für $v \neq u_0$, $S = \{ u_0 \}$, $i = 0$.

2) Für jedes $v \in S^*$ ersetze $l(v)$ durch

$$\min \{ l(v), l(u_i) + w(u_i,v) \},$$

wobei das Minimum über alle $v \in S^*$ genommen wird.

Danach berechne $\min \{ l(v) \mid v \in S^* \}$ und bezeichne die Ecke, bei der das Minimum angenommen wird, mit u_{i+1}. Setze $S := S \cup \{u_{i+1}\}$.

3) Wenn $i = \nu - 1$ —> STOP.
Wenn $i < \nu - 1$ —> $i:=i+1$, gehe nach 2).

Komplexität des Algorithmus von Dijkstra (20.13):
Man braucht im i. ten Schritt, um das 1. Minimum in Schritt 2 zu bestimmen, ν-i-1 Vergleiche und ν-i-1 Additionen. Für das 2. Minimum braucht man ebenso ν-i-1 Vergleiche. Die Anzahl der benötigten Grundoperationen beträgt also $3 \sum_{i=0\ldots\nu-1} \nu\text{-}i\text{-}1$.

Es ist

$$3 \sum_{i=0\ldots\nu-1} \nu\text{-}i\text{-}1 = 3 \sum_{i=1\ldots\nu-1} i$$
$$= 3\, \nu(\nu-1)/2 = (3/2)\nu^2 - 3\nu/2.$$

Der Algorithmus ist also polynomial und in der Größenordnung $O(n^2)$ bei einem vorgegebenen Graphen mit n Ecken.

Das Prinzip der Optimalität (20.14):
Eine Lösung des obigen Problems ist es sicherlich auch, alle möglichen Wege aufzulisten und sie dann zu vergleichen (vollständige Enumeration). Anstatt sich dermaßen ungeschickt zu verhalten, überlegt man sich, daß jeder Teilweg eines kürzesten Weges von u_0 nach v_0 wiederum ein kürzester Weg sein muß. Daraus ergibt sich folgendes Prinzip:

Jede Teillösung einer optimalen Lösung muß selbst in Bezug auf ihren Anfangs- und Endzustand optimal sein.

Das Argument dafür ist einfach: Wenn eine Teillösung nicht optimal ist, ersetzen wir sie durch eine bessere Teillösung. Die Gesamtlösung kann dadurch nur besser werden.

Für eine große Klasse von Problemen ist dieses Prinzip anwendbar. Natürlich ist es nicht Aufgabe dieses Buches, alle diese Probleme aufzuführen, sondern nur exemplarisch davon eines herauszuheben, wie wir dies hier mit dem Problem der kürzesten Wege getan haben.

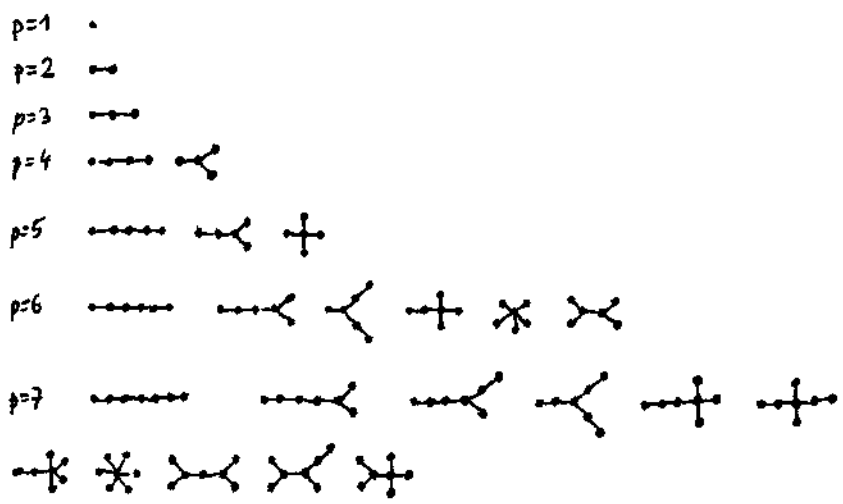
Kapitel 21 : Aufspannende Bäume

21A Bäume

Definition (21.1):

Ein Graph G heißt AZYKLISCH (oder WALD), wenn G keine Zykel hat.
Ist G azyklisch und zusammenhängend, so heißt G BAUM.

Beispiel:
Alle Bäume mit $|E(G)| \leq 7$:



Bemerkung (21.2):
Je zwei Ecken eines Baumes sind durch genau einen Weg verbunden
(bei zwei verschiedenen Wegen würde ein Zykel auftreten).

Satz (21.3):

Ist G ein Baum, so gilt $e = \nu - 1$.

Beweis:
Induktionsanfang:
$\nu = 1$. Der Baum besteht nur aus einer Ecke: $0 = 1 - 1$.
Induktionsvoraussetzung:
Für alle Bäume mit weniger als ν Ecken, $\nu > 1$, gelte die Behauptung.

Induktionsschritt:

Ist e eine Kante von G, dann besteht der Graph G - e aus 2 Zusammenhangskomponenten B_1 und B_2, die Bäume sind. Es gilt $\epsilon(B_1) = \nu(B_1) - 1$ und $\epsilon(B_2) = \nu(B_2) - 1$ nach Induktionsvoraussetzung.
Also haben wir:

$$\begin{aligned}\epsilon &= \epsilon(B_1) + \epsilon(B_2) - 1 \\ &= \nu(B_1) - 1 + \nu(B_2) - 1 + 1 = \nu - 1.\end{aligned}$$

Korollar (21.4):

Jeder Baum hat mindestens 2 Ecken vom Eckengrad 1.

Beweis:
Die Ecken von G haben den Eckengrad $\delta(v) \geq 1$. Nach (19.19) und (21.3) gilt:

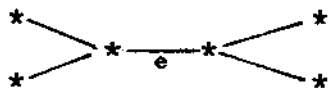
$$\Sigma_{v \in V} \delta(v) = 2\epsilon = 2(\nu-1) = 2\nu - 2.$$

Hätte jede Ecke einen Eckengrad $\delta(v) \geq 2$, dann wäre $\Sigma_{v \in V} d(v) \geq 2\nu$, also gibt es mindestens 2 Ecken vom Eckengrad 1.

Definition (21.5):

Eine Kante e des Graphen G heißt SCHNITTKANTE von G, wenn $\omega(G-e) > \omega(G)$, wobei $\omega(G)$ die Anzahl der Zusammenhangskomponenten von G bezeichnet.

Beispiel:
Jede Kante eines Baumes ist Schnittkante:

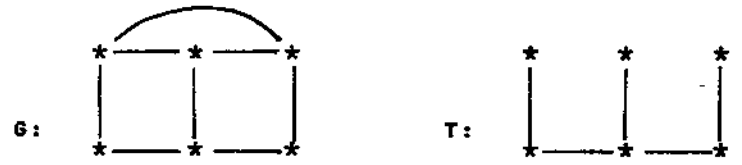


Ein zusammenhängender Graph G ist genau dann ein Baum, wenn jede Kante von G Schnittkante ist.

Definition (21.6):

Ein aufspannender Untergraph T von G, der ein Baum ist, heißt ein AUFSPANNENDER BAUM von G.

Beispiel:



Satz (21.7):

Jeder zusammenhängende Graph besitzt einen aufspannenden Baum.

Beweis:
Sei T ein minimaler, zusammenhängender aufspannende Untergraph von G. Dann gilt $\omega(T) = 1$ und $\omega(T-e) > 1$ für jede Kante e von T wegen der Minimalität. T ist daher ein Baum.

Korollar (21.8):

Ist G zusammenhängend, so ist $\epsilon \geqslant \nu - 1$.

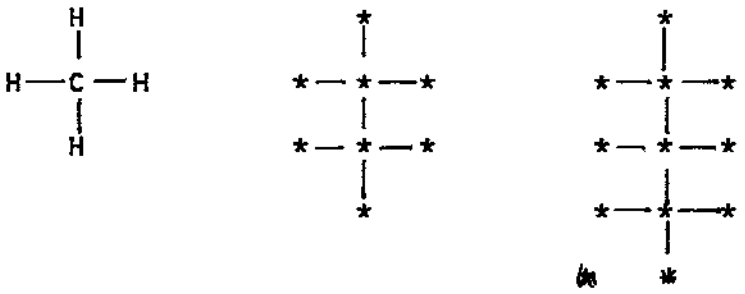
Beweis:
Nach (21.7) enthält G einen aufspannenden Baum T. Es gilt:

$$\epsilon(G) \geqslant \epsilon(T) = \nu(T) - 1 = \nu - 1.$$

21B: Anzahl der aufspannenden Bäume eines vollständigen Graphen

Die Anzahl von Bäumen mit vorgegebener Eigenschaft spielt in vielen Zusammenhängen eine Rolle.

Beispiel: Anzahl der Isomere von gesättigten Kohlewasserstoffen



Methan Äthan Propan

(Die Eckengrade entsprechen der Valenz der chemischen Elemente)

Im folgenden interessiert uns die Anzahl t(G) der aufspannenden Bäume eines Graphen G.

Definition (21.9):

Der Graph G*e heißt durch KONTRAKTION der Kante e aus G entstanden, wenn man e aus G herausnimmt und die Endecken von e identifiziert.

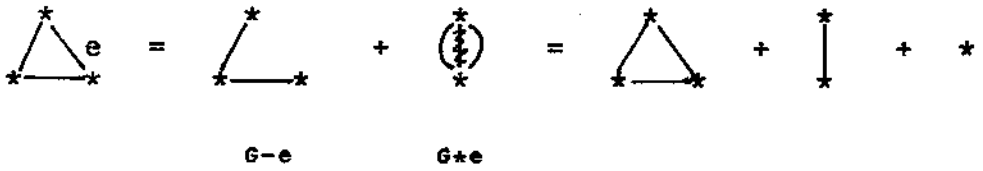
Beispiel:

Satz (21.10):

Ist e eine Kante mit zwei verschiedenen Ecken von G, dann ist die Anzahl der aufspannenden Bäume $t(G) = t(G-e) + t(G*e)$.

Beweis:
Jeder aufspannende Baum von G, der die Kante e nicht enthält, ist ein aufspannender Baum von G-e. t(G-e) ist also die Anzahl der aufspannenden Bäume von G, die e nicht enthalten.
Jedem aufspannenden Baum T von G, der e enthält, entspricht ein aufspannender Baum T*e von G*e. Deshalb ist t(G*e) genau die Anzahl der aufspannenden Bäume, die die Kante e enthalten.

Beispiel:



G-e G*e

Also erhält man t(G) = 3, durch rekursive Anwendung obiger Formel; man sieht jedoch schon, daß diese Methode für große Graphen wenig brauchbar ist.

Satz (21.11):

Ein vollständiger Graph K_n mit n Ecken hat $t(K_n) = n^{n-2}$ aufspannende Bäume.

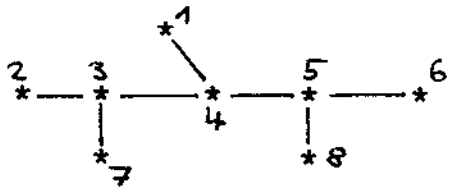
Beweis:
Die Eckenmenge von K_n sei $E(K_n) = \{1, \ldots, n\}$. Die Anzahl der (n-2)- Tupel auf dieser Menge ist n^{n-2} (siehe 7.5).
Wir zeigen, daß jeder aufspannende Baum genau einem solchen (n-2)-Tupel entspricht (A) und umgekehrt (B).

(A): Jedem aufspannenden Baum T von K_n wird auf folgende Weise ein Tupel $(t_1, \ldots, t_{n-2})$ zugeordnet: Man betrachte N als geordnete Menge und suche die 'erste' Ecke s_1 vom Eckengrad 1.
Die zu s_1 benachbarte Ecke ist t_1.
Nun entfernen wir die Ecke s_1 aus T und suchen die 'erste' Ecke s_2 vom Eckengrad 1 in $T-s_1$. Die zu s_2 benachbarte Ecke setzen wir als

t_2.

Wir wiederholen diese Prozedur solange,bis wir t_{n-2} gefunden haben.

Illustration: (für K_8)



——> (4, 3, 5, 3, 4, 5)

(B): Wir bemerken, daß jede Ecke v von T im Tupel $(t_1,\ldots,t_{n-2})$ $\delta_T(v)-1$ mal vorkommt, d.h. die Ecken vom Eckengrad 1 kommen in dieser Folge nicht vor. Um T aus $(t_1, \ldots, t_{n-2})$ zurückzugewinnen, gehen wir folgendermaßen vor:

Sei s_1 die erste Ecke in $N = \{ 1, \ldots, n \}$, die in $(t_1, \ldots, t_{n-2})$ nicht vorkommt. Dann verbinde s_1 mit t_1. Sei s_2 die erste Ecke in $N - \{s_1\}$, die in $(t_2, \ldots, t_{n-2})$ nicht vorkommt, dann verbinde s_2 mit t_2.
Man setzt dies fort, bis die (n-2) Kanten $s_1t_1, \ldots, s_{n-2}t_{n-2}$ bestimmt sind. T erhält man, indem man die zwei restlichen Kanten aus $N- \{s_1, \ldots, s_{n-2}\}$ verbindet.

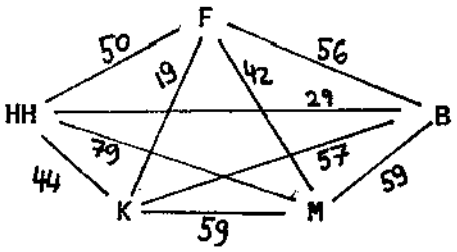
Verschiedene Tupel geben offensichtlich verschiedene Bäume. Die Zuordnung ist eine Bijektion.

21C : Der Algorithmus von Kruskal

Problem (21.12):

Gegeben ist ein zusammenhängender Graph G mit einer Gewichtsfunktion w von E nach $\mathbb{R}^+$. Gesucht ist ein aufspannender Baum von G mit minimalem Gewicht.

Beispiel:
Wie kann man diese 5 Städte miteinander verbinden, so daß das Verbindungsnetz so kurz wie möglich wird?



(21.13):[ALGORITHMUS VON KRUSKAL]

1) Ordne die Kanten nach aufsteigendem Gewicht : $E = \{a_1, \ldots, a_\varepsilon\}$, setze $e_1 = a_1$.

2) Sind die Kanten $e_1, \ldots, e_i$ bereits gewählt, so wähle $e_{i+1} = a_j$ aus $E-\{e_1, \ldots, e_i\}$ derart, daß

   i) $G[\{e_1, \ldots, e_{i+1}\}]$ azyklisch ist
   ii) $w(e_{i+1})$ so klein wie möglich ist.

3) STOP, wenn Schritt 2) nicht mehr weiter durchgeführt werden kann.

Für Schritt 2) braucht man eine geeignete Unterprozedur, die die Abfrage '$G[S \cup \{a_j\}]$ azyklisch ?' für $S = \{e_1, \ldots, e_i\}$ durchführt. Dazu muß man feststellen, ob die Endecken von a_j in verschiedenen Komponenten des Waldes $G[S]$ sind oder nicht. Man führt eine Markierung l ein, mit der bei jedem Schritt festgestellt werden kann, ob zwei Ecken zur selben Komponente von $G[S]$ gehören:

$$l(v) = k, \text{ falls } v \text{ zur } k\text{-ten Komponente gehört.}$$

Wird nun a_j mit den Endecken v_i und v_{i+1} hinzugenommen,dann gilt:

$$\begin{aligned} l(v_i) &\longleftarrow \min \{ l(v_i), l(v_{i+1}) \} \\ l(v_{i+1}) &\longleftarrow \min \{ l(v_i), l(v_{i+1}) \}. \end{aligned}$$

Nun muß man auch die anderen Ecken für die neuen Werte $l(v_i)$ und $l(v_{i+1})$ prüfen. Dabei braucht man höchstens ν Vergleiche. Da insgesamt höchstens $\nu-1$ Kanten zu betrachten sind, genügen $\nu(\nu-1)$ Vergleiche.

Komplexität des Algorithmus von Kruskal (21.14):
Setzt man beispielsweise den Bubble Sort für Schritt 1) ein, so hat man dafür $O(\varepsilon^2)$. Für Schritt 3) hat man $O(\nu^2)$ und daher insgesamt $O(n^2)$ für einen Graphen mit n Kanten.

Satz (21.15): [Verifikation des Algorithmus]

Jeder aufspannende Baum $T_1 = G[\{e_1, \ldots, e_s\}]$, der durch den Algorithmus von Kruskal konstruiert wird, ist von minimalem Gewicht.

Beweis:
Bezeichne für jeden aufspannenden Baum T von G, der von T_1 verschieden ist, $f(T) = i$ die kleinste Zahl, für die e_i nicht in T ist. Wir nehmen an, daß T_1 nicht von minimalem Gewicht ist. Sei T ein Baum minimalen Gewichts, wobei $f(T)$ so groß wie möglich ist. Sei $f(T) = k$, d. h. $e_1, \ldots, e_{k-1} \in T_1 \cap T$, aber $e_k \notin T$.

Dann enthält $T+e_k$ genau einen Zykel C. Sei e_k^0 eine Kante, die in C liegt, aber nicht in T_1. Dann ist $(T+e_k)-e_k^0 = T^0$ ein Baum, der G aufspannt. Wir haben $w(T^0) = w(T) + w(e_k) - w(e_k^0)$.

Nach dem Algorithmus von Kruskal war e_k so gewählt, daß e_k von minimalem Gewicht ist, wobei $G[\{e_1, \ldots, e_k\}]$ azyklisch ist.

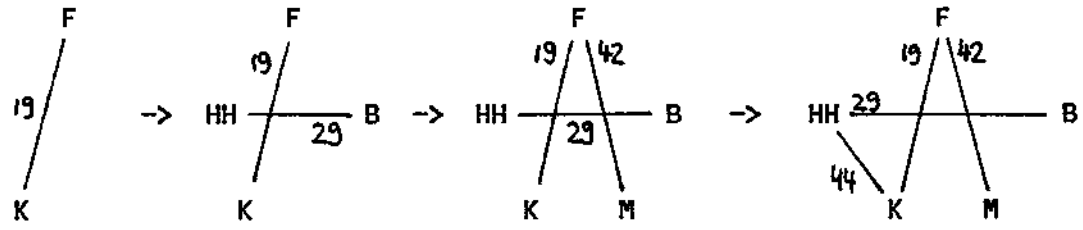
Da $G[\{e_1, \ldots, e_{k-1}, e_k^0\}]$ ebenfalls azyklisch ist, muß gelten:

$$w(e_k^0) \geqslant w(e_k).$$

Wir haben also $w(T^0) \leqslant w(T)$ und daher ist T^0 ebenfalls von minimalem Gewicht. Aber $f(T^0) > k = f(T)$ ist ein Widerspruch zu unserer Annahme.

Beispiel:

$$19 \leqslant 29 \leqslant 42 \leqslant 44 \leqslant 50 \leqslant 56 \leqslant 57 \leqslant 59 \leqslant 59 \leqslant 79$$



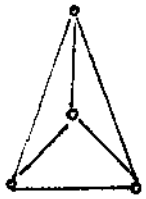
Kapitel 22 : Hamiltonsche Graphen

22A : Theorie der Hamiltonschen Graphen

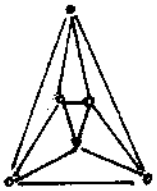
Definition (22.1):

Ein Weg, der jede Ecke von G enthält, heißt ein HAMILTONSCHER WEG. Ein HAMILTONZYKEL ist ein Zykel, der alle Ecken von G enthält. Ein Graph G heißt HAMILTONSCH, wenn er einen Hamiltonzykel enthält.

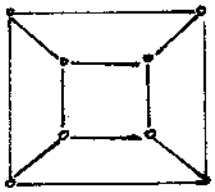
Beispiel:
Die platonischen Graphen sind hamiltonsch: (siehe auch 30.10)



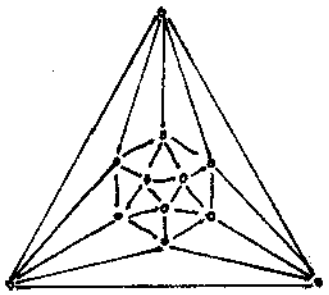
Tetraeder



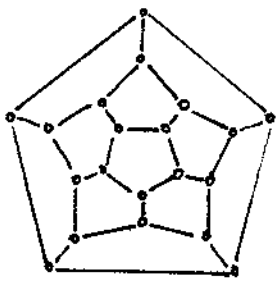
Oktaeder



Würfel



Ikosaeder



Dodekaeder

Bemerkung (22.2):
Eine Charakterisierung der hamiltonschen Graphen durch Eckengrade oder ähnliche Eigenschaften von Graphen ist bisher nicht gelungen und stellt ein ungelöstes Problem dar. Dagegen gibt es eine Reihe

hinreichender Bedingungen sowie eine Reihe notwendiger Bedingungen.

Satz (22.3):

Ist G hamiltonsch, dann gilt für jede nichtleere, echte Untermenge S von V für die Zahl der Zusammenhangskomponenten $\omega(G-S) \leq |S|$.

Beweis:
Sei C eine Hamiltonzykel von G. Dann gilt für jede nichtleere echte Untermenge S von V

$$\omega(C-S) \leq |S|.$$

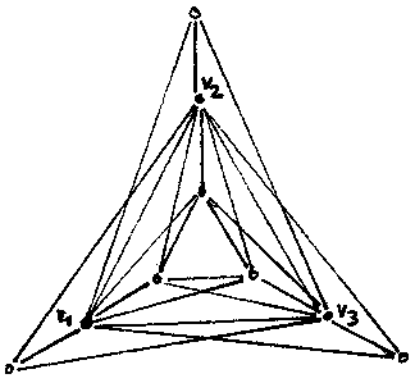
Da nun C-S ein aufspannender Untergraph von G-S ist, gilt:

$$\omega(G-S) \leq \omega(C-S).$$

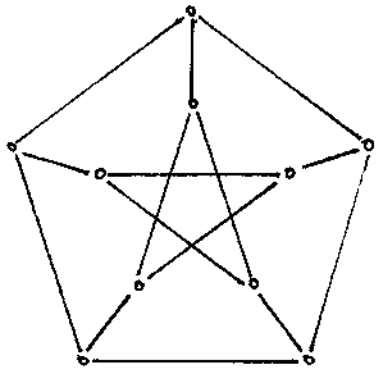
Beispiel:
Folgender Graph ist nicht hamiltonsch:

$S = \{ v_1, v_2, v_3 \}$

$\omega(G-S) = 4.$



Der PETERSENGRAPH ist nicht hamiltonsch, aber Satz (22.3) ist nicht anwendbar:



Wir bemerken, daß ein Graph genau dann hamiltonsch ist, wenn der zugrunde liegende schlichte Graph hamiltonsch ist. Daher betrachten wir nur schlichte Graphen. Satz (22.3) vermittelt den Eindruck, daß ein Graph hamiltonsch ist, wenn der kleinste Eckengrad groß ist.

Dies bestätigt folgender Satz.

Satz (22.4):

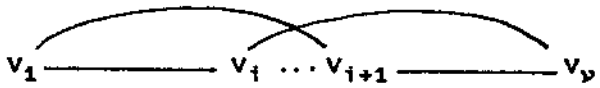
Ist G schlicht, $\nu \geq 3$ und δ [kleinster Eckengrad] $\geq \nu/2$, dann ist G hamiltonsch.

Beweis:
Angenommen, der Satz ist falsch. Sei G ein maximaler, nicht hamiltonscher schlichter Graph mit $\nu \geq 3$ und $\delta \geq \nu/2$. G kann nicht vollständig sein, denn vollständige Graphen mit mindestens 3 Ecken sind hamiltonsch. Daher gibt es zwei Ecken u, v , die nicht benachbart sind in G. Wegen der Wahl von G ist G + uv hamiltonsch und jeder Hamiltonzykel von G + uv muß die Kante uv enthalten.

Daher gibt es einen Hamiltonweg $v_1v_2 \ldots v_\nu$ in G mit dem Anfang $u=v_1$ und dem Ende $v=v_\nu$.
Sei $S = \{ v_i \mid uv_{i+1} \in E \}$ und $T = \{ v_i \mid v_iv \in E \}$.
Da $v_\nu \notin S \cup T$ ist gilt

$$|S \cup T| < \nu \quad (*) \text{ und weiter } S \cap T = \emptyset \quad (**),$$

denn, falls $S \cap T$ eine Ecke v_i enthält, würde G den Hamiltonzykel $v_1v_2 \ldots v_iv_\nu v_{\nu-1} \ldots v_{i+1}v_1$ haben, Widerspruch.



Zykel

Wegen (*) und (**) gilt:

$$d(u) + d(v) = |S| + |T| = |S \cup T| + |S \cap T| = |S \cup T| < \nu,$$

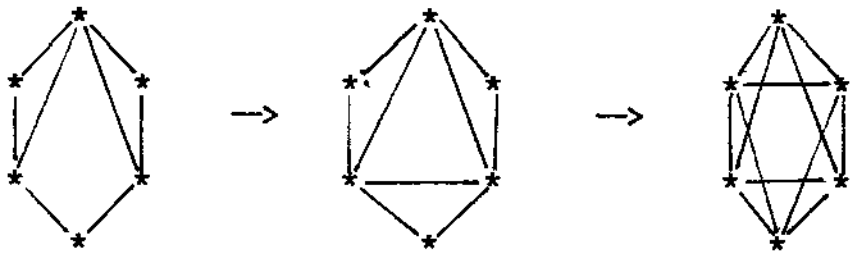
also $d(u) + d(v) < \nu/2 + \nu/2$, im Widerspruch zu $\delta \geq \nu/2$.

Satz (22.5):

Sei G ein schlichter Graph und u, v zwei nicht benachbarte Ecken in G mit $d(u) + d(v) \geq \nu$. G ist genau dann hamiltonsch, wenn G + uv hamiltonsch ist.

Beweis:
Sei G + uv hamiltonsch. Angenommen, G ist nicht hamiltonsch. Dann ergibt sich wie im Beweis von (22.4), daß $d(u) + d(v) < \nu$ ist, Widerspruch.

Dieser Satz legt es nahe, zu untersuchende Graphen um Kanten zu ergänzen. Ein triviales Beispiel liefert folgender Graph:



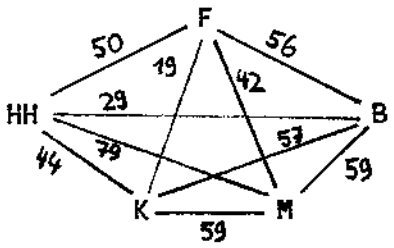
Dieser läßt sich zu einem vollständigen Graphen G ergänzen und ist daher hamiltonsch.

22B : Das 'Travelling Salesman Problem'

Travelling-Salesman-Problem (22.6):
Ein Handelsvertreter wünscht eine Reihe von Städten zu besuchen und danach wieder zu seinem Ausgangspunkt zurückzukehren. Wenn die Entfernungen zwischen den Städten gegeben sind, soll ein Reiseweg ausgesucht werden, der jede Stadt genau einmal enthält und eine minimale Gesamtlänge hat.

Graphentheoretische Version (22.7):
Gegeben sei ein gewichteter vollständiger Graph. Gesucht ist ein Hamiltonzykel von kleinstem Gewicht.
Ein solcher Zykel wird im folgenden OPTIMALER ZYKEL genannt.

Beispiel:
Wie läßt sich eine optimale Rundreise durch diese 5 Städte finden?



Bemerkung (22.8):
Es ist ein ungelöstes Problem, ob es einen Algorithmus polynomialer Komplexität gibt, der einen optimalen Zykel findet. Natürlich gibt

es Algorithmen, die exakt einen optimalen Zykel bestimmen. Man kann beispielsweise alle Hamiltonzykeln auflisten, etwa als Permutationen der Ecken, und die Gewichte dann vergleichen.
Ein solcher Algorithmus hat, von einem Ausgangsort gezählt, (n-1)! Hamiltonzyklen bei vorgegebenen n Ecken zu vergleichen. Hat man etwa 101 Städte vorgegeben, dann sind 100! = ca. $9{,}3 * 10^{157}$ Hamiltonzyklen zu untersuchen.

Offenkundig ist die Komplexität eines solchen Algorithmus exponentiell, denn n! läßt sich abschätzen durch $\mathrm{sqrt}(2\pi n)\ (n/e)^n$. Selbst für kleine n führt das auf einer modernen Rechenanlage zu utopischen Rechenzeiten.

Bezeichnung (22.9):

Ein Algorithmus heißt HEURISTISCH, wenn er nicht notwendigerweise eine formale Garantie für die Güte der gefundenen Lösung bietet.

Dieser Begriff ist in der Allgemeinheit, wie wir ihn hier benutzen, nicht exakter zu definieren.

Wir wollen im folgenden einen heuristischen Näherungsalgorithmus für das Travelling-Salesman Problem vorstellen, der zwar nicht unbedingt einen optimalen Zykel findet, aber von polynomialer Komplexität ist.

22.10: Greedy-Algorithmus

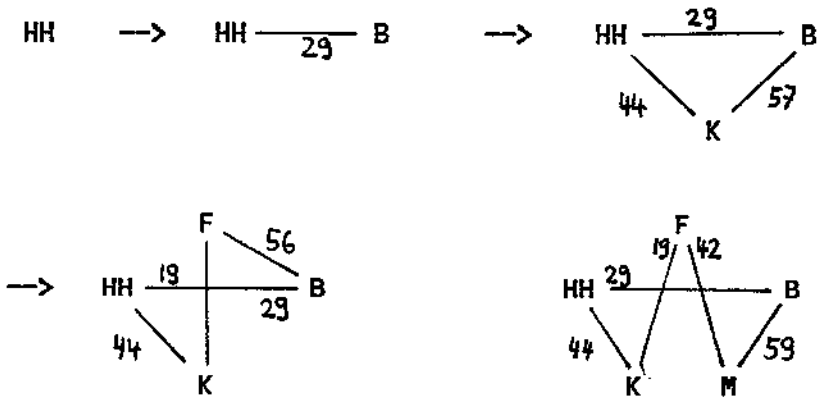
1) Beginne mit dem Zykel C_1, der aus einer beliebigen Ecke v_1 und keiner Kante besteht.

2) Ist der k-Zykel C_k bereits gefunden, suche eine Ecke z_k mit $z_k \in V - C_k$, die die kürzeste Entfernung zu einer Ecke von C_k, sagen wir zu y_k, hat.

3) Sei C_{k+1} der (k+1)-Zykel, der aus C_k entsteht, indem wir nach y_k die Ecke z_k einfügen.

4) Wiederhole Schritt 2) und 3), solange bis ein Hamiltonzykel entsteht.

Beispiel:



Hamiltonzykel

Der Hamiltonzykel hat die Gesamtlänge 193.

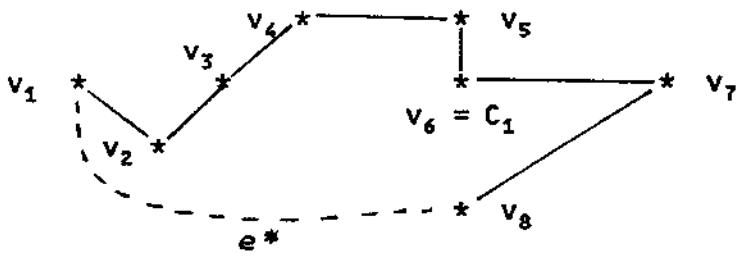
Satz (22.11):

Das Gewicht des Zykels, der vom Algorithmus (22.10) bestimmt wird, ist höchstens doppelt so groß wie das Gewicht eines optimalen Zykels.

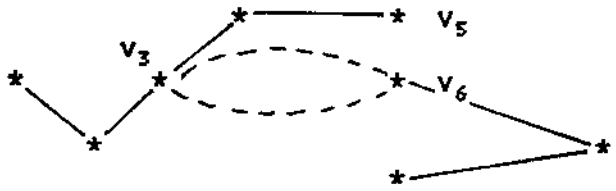
Beweis:
Sei C_k ein k-Zykel, der im Verlaufe des Näherungsalgorithmus (22.10) gefunden wurde. Sei ferner C ein optimaler Zykel und S_k die Menge der Kanten von C, durch die die Ecken aus $V(C_k)$ mit den Ecken aus $V-V(C_k)$ verbunden sind.

Für k=1 bestehe S_1 aus $C-e^*$, wobei e^* die Kante mit dem größten Gewicht in C ist. In diesem Fall ist C_1 eine Ecke und $C-e^*$ ein Hamiltonweg.

Beispiel für $C-e^*$ und C_1:

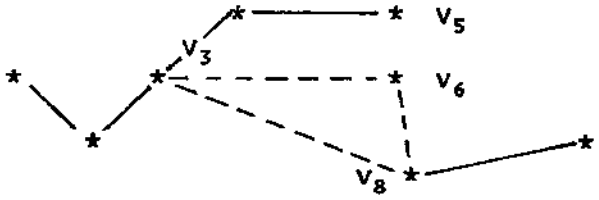


Es schränkt die Allgemeinheit des Beweises nicht ein, wenn wir das vorstehende Beispiel betrachten. Wir gehen davon aus, daß sich nach Anwendung des Greedyalgorithmus C_2 als Zykel $v_6v_3v_6$ ergibt:



Dann ist $S_2 = S_1 - v_5v_6$.

Gehen wir davon aus, daß sich nach dem Greedyalgorithmus C_3 als der Zykel $C_3 = v_6v_8v_3v_6$ ergibt, dann haben wir $S_3=S_2-v_6v_7$:



Im allgemeinen Fall haben wir also, daß S_{k+1} aus S_k dadurch entsteht, daß die erste Kante des Weges in S_k zur neuen Ecke z_k entfernt wird.

Wenn nun c_{ij} das Gewicht der Kante v_iv_j ist, dann ergibt sich im vorstehenden Beispiel, daß c_{68} das kleinste Gewicht ist für Kanten von den Ecken v_2, v_6 von C_2 zu den restlichen Ecken von G.

Daher ist $c_{68} \leqslant c_{67}$, wobei v_6v_7 die Kante war, die entfernt wurde. Benützt man die Dreiecksungleichung, so kann man, wie folgt, beweisen, daß sich das Gewicht von C_2 auf C_3 höchstens um $2c_{67}$ erhöht.

Der Zuwachs von C_2 auf C_3 ist $c_{68}+c_{38}-c_{36}$, da die Kanten v_6v_8 und v_3v_8 die Kante v_3v_6 ersetzen. Nach der Dreiecksungleichung ergibt sich $c_{38} \leqslant c_{36}+c_{68}$ und daraus $c_{38}-c_{36} \leqslant c_{68}$.
Also ist $c_{68}+(c_{38}-c_{36}) \leqslant c_{68}+c_{68} \leqslant 2c_{67}$.

Dasselbe Argument kann nun benutzt werden, um zu zeigen, daß der Gewichtszuwachs von C_k auf C_{k+1} höchstens doppelt so groß ist wie das Gewicht der entfernten Kante.

(22.12):Komplexität des Greedyalgorithmus

Man muß bei jedem Schritt für die neue Ecke v_k die Summe $c_{ih} + c_{hj} - c_{ij}$ aus dem Zykel entfernen, da v_i und v_j nicht mehr auf C_k benachbart sind.

Danach muß man $c_{ih}+c_{hk}-c_{ik}$ und $c_{kh}+c_{hj}-c_{kj}$ in den Zykel einfügen.

Um v_k zu finden, muß man n-k Durchläufe vollziehen. Setzen wir die Bildung der Summen mit log n an und die Gesamtdurchläufe für alle h

= 1, ... , n mit $n^2/2 - n/2$, ergibt sich als Abschätzung $O(n^2 \log n)$.

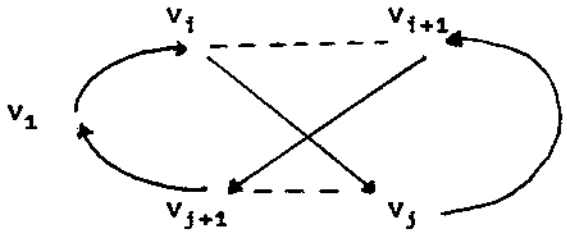
Bemerkung:
Man kann aufgrund von Beispielen für n=6 Ecken zeigen, daß tatsächlich Zykel beim Greedyalgorithmus auftreten, die nahezu das doppelte Gewicht eines optimalen Zykels haben.
Eine Möglichkeit, das Ergebnis zu verbessern, liefert der folgende Algorithmus:

(22.13): Zykelverbesserungsalgorithmus

1) Es sei ein Hamiltonzykel $C=v_1 \ldots v_\nu v_1$ gegeben.

2) Durch Modifikation wird C folgendermaßen verbessert:
Für alle i und j mit $1 < i+1 < j < \nu$ erhält man einen neuen Hamiltonzykel

$$C_{ij} = v_1 v_2 \ldots v_i v_j v_{j-1} \ldots v_{i+1} v_{j+1} \ldots v_\nu v_1$$



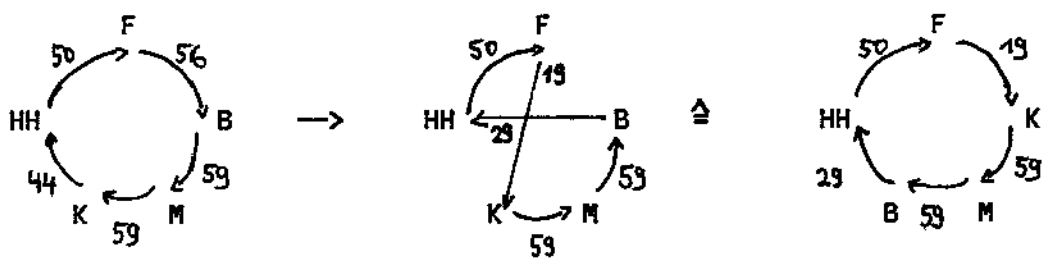
Wenn nun für i, j gilt:

$$w(v_i v_j) + w(v_{i+1} v_{j+1}) < w(v_i v_{i+1}) + w(v_j v_{j+1}),$$

dann ist C_{ij} eine Verbesserung von C, setze $C := C_{ij}$.

3) Nach einer vorher festgesetzten Anzahl von Durchläufen: STOP.

Beispiel:



100 > 48

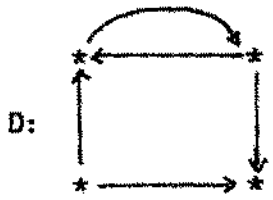
Kapitel 23 : Gerichtete Graphen

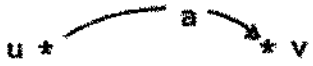
23A : Grundbegriffe

Definition (23.1):

Ein GERICHTETER GRAPH (DIGRAPH) ist ein Tripel $D = (V(D), A(D), \Phi_D)$, wobei $V(D)$ eine nichtleere Menge von Ecken, $A(D)$ eine von $V(D)$ disjunkte Menge von Bögen und $\Phi(D)$: $A(D) \longrightarrow V(D) \times V(D)$ eine Inzidenzfunktion ist, die jedem Bogen ein geordnetes Paar, bestehend aus Anfangsecke und Endecke, zuordnet.

Beispiel:

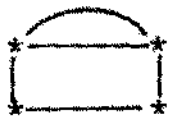




u heißt Anfangsecke des Bogens a
v heißt Endecke des Bogens a
$\Phi(a) = (u,v)$.

Bemerkung:
Zu jedem Digraphen D gibt es den zugrunde liegenden Graphen G. Deswegen heißt D auch eine ORIENTIERUNG von G.

Beispiel:
Der zugrunde liegende (ungerichtete) Graph für obiges D ist:



Die meisten Begriffe für ungerichte Graphen lassen sich ohne Änderung auf Digraphen übertragen. Darüberhinaus wollen wir jedoch einige neue Begriffe einführen.

Definition (23.2):

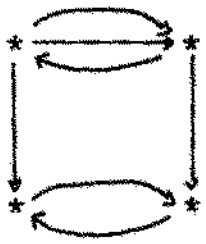
Eine GERICHTETE KANTENFOLGE K ist eine Folge $v_0a_1v_1\ldots a_kv_k$ von abwechselnd Ecken (v_i) und Bögen (a_i), so daß v_i die Anfangsecke und v_{i+1} die Endecke des Bogens a_{i+1} ist, für $i = 0,\ldots,k-1$.
Eine GERICHTETE LINIE L ist eine gerichtete Kantenfolge, die auf dem zugrundeliegenden Graphen G eine Linie ist.
Ein GERICHTETER WEG W ist eine gerichtete Kantenfolge, die auf dem zugrundeliegenden Graph G ein Weg ist.

Definition (23.3):

In einem Digraphen D heißt eine Ecke v ERREICHBAR von einer Ecke u, wenn es einen gerichteten Weg von u nach v gibt.
Zwei Ecken heißen DIZUSAMMENHÄNGEND, wenn jede der beiden Ecken von der anderen aus erreichbar ist.

Bemerkung:
Der Dizusammenhang ist eine Äquivalenzrelation (jede Ecke ist von sich selbst aus erreichbar). Die Äquivalenzklassen heißen DIKOMPONENTEN.

Beispiel:



Dikomponenten:

Definition (23.4):

Der EINGANGSGRAD $\delta_D^-(v)$ einer Ecke v des Digraphen D ist die Anzahl der Bögen, deren Endecke v ist. Der AUSGANGSGRAD $\delta_D^+(v)$ einer Ecke v ist die Anzahl der Bögen, deren Anfangsecke v ist.

Es gilt: $\Sigma_{v\in V}\delta_D^-(v) = \Sigma_{v\in V}\delta_D^+(v) = e$ = Anzahl Bögen.

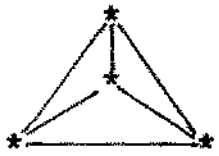
23B : Turniere

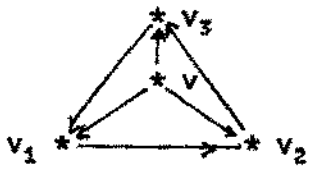
Definition (23.5):

Die Orientierung eines vollständigen Graphen heißt TURNIER.

Beispiele:
a)

K_4: Turnier:

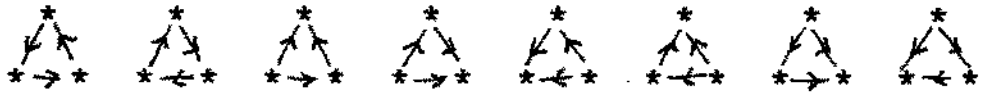




v 'siegt' über v_1, v_2 und v_3

b)

Alle Turniere auf K_3:



c)

Alle Turniere auf K_3 bis auf Isomorphie:

Satz (23.6):

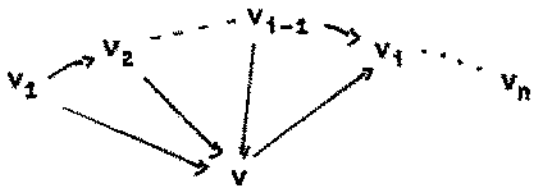
Jedes Turnier hat einen gerichteten Hamiltonweg.

Beweis:
Die Behauptung ist für jedes Turnier mit 3 Ecken richtig (Ind. Anfang). Sei nun D ein Turnier mit n+1 Ecken, dann betrachten wir

$\bar{D} = D - v$, wobei eine Ecke v samt aller mit v inzidenten Bögen

entfernt ist. $\bar{D}$ ist ein Turnier mit n Ecken.
Nach Induktion gibt es einen gerichteten Hamiltonweg $v_1v_2 \ldots v_n$ von

v_1 nach v_n in $\bar{D}$. Ist nun v v_1 ein Bogen in D, so sind wir fertig.
Ist v v_1 kein Bogen, und gibt es einen Bogen v v_i für ein i, dann wähle man das kleinste i mit dieser Eigenschaft und betrachte den Hamiltonweg $v_1v_2 \ldots v_{i-1}$ v $v_i \ldots v_n$:



Gibt es in T keinen Bogen v v_1, so ist $v_1 v_2 \ldots v_n$ v der gesuchte gerichtete Hamiltonweg.

Satz (23.7):

Jede Ecke eines dizusammenhängenden Turniers D mit $\nu \geqslant 3$ ist in einem gerichteten k-Zykel, $3 \leqslant k \leqslant \nu$, enthalten.

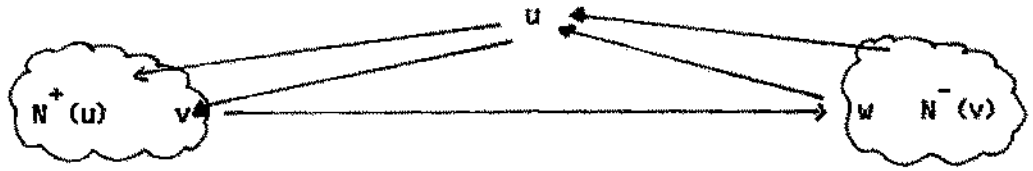
Beweis:
Sei D ein dizusammenhängendes Turnier mit $\nu \geqslant 3$. Für die Ecke u von D setzen wir

$$N^-(u) = \{ v \mid vu \in A(D) \}$$

und

$$N^+(u) = \{ v \mid uv \in A(D) \}.$$

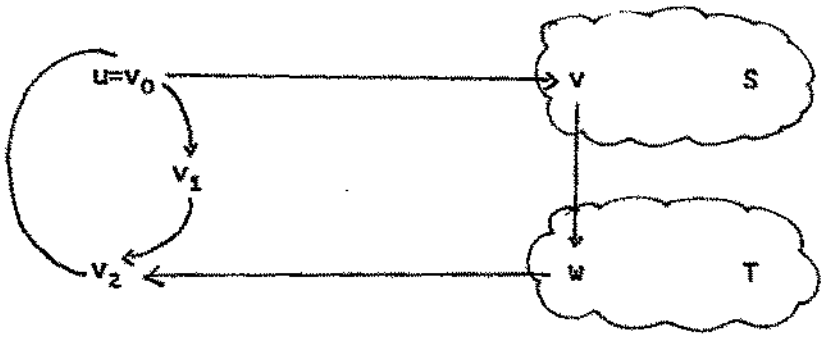
Da D zusammenhängend ist, ist weder $N^-(u)$ noch $N^+(u)$ leer, und aus dem gleichen Grund muß es ein $v \in N^+(u)$ und ein $w \in N^-(u)$ geben, die durch einen Bogen verbunden sind:



Also gibt es einen gerichteten 3-Zykel u v w u.

Wir nehmen an, daß u in gerichteten k-Zykeln für $3 \leqslant k < \nu$ liegt, und zeigen durch Induktion die Behauptung für k+1.
Sei $C = v_0 \ldots v_k$ ein gerichteter k-Zykel mit $u = v_0 = v_k$. Wenn v eine Ecke in $V(D) - V(C)$ ist mit einem Bogen von einer Ecke in C nach v und mit einem Bogen von v nach einer Ecke in C, dann gibt es benachbarte Ecken v_i und v_{i+1} auf C, so daß $v_i v$ und $v v_{i+1}$ Bögen in D sind, denn D ist ein Turnier.
Ist dies nicht der Fall, dann sei S die Menge aller Ecken in $V(D)$ -

V(C), die Endecken für von C ausgehende Bögen sind und sei T die Menge aller Ecken, die Anfangsecken von Bögen sind, die ihre Endecken in C haben;



Da D dizusammenhängend ist, gibt es einen Bogen v w in D mit v ∈ S und w ∈ T. Da D ein Turnier ist, liegt u in einem k+1-Zykel $v_0vwv_2\ldots v_k$.

Aus Satz (23.7) folgt unmittelbar:

Satz (23.8):

Jedes dizusammenhängende Turnier besitzt einen gerichteten Hamiltonzykel.

23C : Optimale gerichtete Hamiltonwege

Problem:
Eine Anzahl von Jobs $J_1, J_2, \ldots, J_n$ müssen von einer einzigen Maschine abgearbeitet werden. Nach jedem Job muß die Maschine umgestellt werden auf die Erfordernisse des neuen Jobs. Die Umstellungszeit von Job J_i zum Job J_j sei t_{ij}.
Man finde eine Folge von Jobs, die die Gesamtzeit für Umstellungen minimiert.

Bemerkung:
Dieses Problem ist offensichtlich mit dem Travelling-Salesman Problem verwandt; es ist also ebenfalls unbekannt, ob dafür ein Algorithmus polynomialer Komplexität existiert.
Auf der Grundlage der folgenden Mathematisierung wollen wir einen Algorithmus vorstellen, der 'passable Lösungen' findet.

Mathematisierung:
Man konstruiert einen Digraphen D mit den Ecken $v_1, \ldots, v_n$, so daß v_iv_j genau dann ein Bogen von D ist, wenn $t_{ij} \leqslant t_{ji}$ ist.

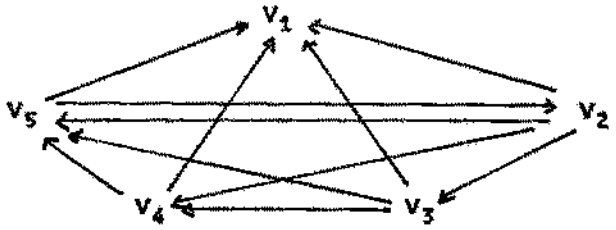
Durch die Konstruktion ist es klar, daß D ein aufspannendes Turnier enthält. Die Aufgabe lösen wir, indem wir einen gerichteten Hamiltonweg finden (der natürlich im allgemeinen kein optimaler Hamilton-

weg sein wird).

Wir wollen dieses Problem durch ein Beispiel illustrieren.
Seien 5 Jobs durch folgende Matrix gegeben, die die Umrüstzeiten als Koeffizienten hat:

	J_1	J_2	J_3	J_4	J_5
J_1	0	5	3	4	2
J_2	1	0	1	2	3
J_3	2	5	0	1	2
J_4	1	4	4	0	1
J_5	1	3	4	5	0

Der zugehörige Digraph ist dann:



Man entwickelt leicht einen guten Algorithmus, um für einen solchen Digraphen einen gerichteten Hamiltonweg zu finden.
In unserem Beispiel erkennt man bereits aus der Zeichnung, daß der Hamiltonweg $J_2J_3J_4J_5J_1$ ein Gewicht von 4 hat. Im Gegensatz dazu ist die Lösung $J_1J_2J_3J_4J_5$, die man als erste Annäherung finden mag, vom Gewicht 8.

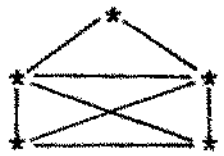
Kapitel 24 : Eulersche Graphen

24A : Grundbegriffe

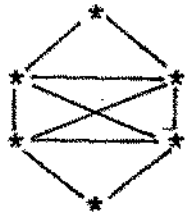
Definition (24.1):

Eine EULERTOUR von G ist eine geschlossene Linie, die jede Kante von G enthält. Ein Graph G heißt EULERSCH, wenn G eine Eulertour besitzt.

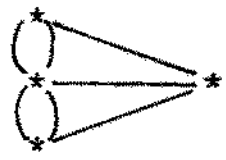
Beispiel:
a)



Dieser Graph ist nicht eulersch, auch wenn man den Graphen ohne Absetzen zeichnen kann (der Anfangspunkt ist nie der Endpunkt, also keine geschlossene Linie).
Dagegen sieht man sofort, daß folgender Graph eulersch ist:



b) Der historische Anlaß für Euler war die Behandlung des Königsberger Brückenproblems (siehe Kapitel 19). Der zugehörige Graph ist nicht eulersch:



Satz (24.2):

Ein nicht leerer, zusammenhängender Graph G ist genau dann eulersch, wenn G keine Ecke von ungeradem Eckengrad hat.

Beweis:
'=>'
Sei G eulersch, dann besitzt G eine Eulertour mit Anfangs- und Endecke u. Läuft man von u aus die Eulertour ab, müssen für jede Ecke v zwei Kanten gezählt werden. Da die Eulertour alle Kanten enthält, ist $\delta(v)$ gerade. Ebenso sieht man, daß $\delta(u)$ gerade ist.

'<='
Sei G zusammenhängend mit wenigstens einer Kante und keiner Ecke von ungeradem Eckengrad. Angenommen, die Behauptung ist falsch, dann gibt es ein Gegenbeispiel mit so wenig Kanten wie möglich.
Da jede Ecke von G den Eckengrad mindestens 2 hat, gibt es irgendeine geschlossene Linie. Sei C eine geschlossene Linie von maximaler Länge in G.
Wegen unserer Annahme enthält C nicht alle Kanten von G. Also besitzt $G - E(C)$ eine nicht triviale Zusammenhangskomponente G^* mit $|E(G^*)|>0$. Da C selbst ein eulerscher Graph ist, besitzt C keine Ecke von ungeradem Eckengrad. Daher hat auch G^* keine Ecke von ungeradem Eckengrad. Wegen $|E(G^*)| < |E(G)|$ ist G^* eulersch und besitzt eine Eulertour C^*.

Da G zusammenhängend ist, ist $V(C) \cap V(C^*) \neq \emptyset$. Daher haben C und C^* eine Ecke u gemeinsam, die man als Anfangsecke und Endecke der Tour $C\ C^*$ ansehen kann. Da $C\ C^*$ größer als C ist, ergibt sich ein Widerspruch.

(24.3): Das Chinesische Briefträger-Problem

Wie kann man eine Route finden, die alle Straßen eines Bezirks enthält und eine minimale Länge hat (dabei heißt eine geschlossene Kantenfolge ROUTE) ?

Mathematisches Modell:
In einem gewichteten Graphen definiert man für eine Route $v_0e_1v_1 \ldots c_nv_n$ das Gewicht der Tour als $\Sigma_{i=1\ldots n}w(e_i)$.
Das Problem lautet nun: Finde eine Route minimalen Gewichts in einem gewichteten zusammenhängenden Graphen, die alle Kanten mindestens einmal enthält. Eine solche Route heiße eine OPTIMALE ROUTE.

(24.4): ALGORITHMUS VON FLEURY

1) Wähle eine Ecke v_0 und setze $L_0 = v_0$.
2) Sei die Linie $L_i = v_0e_1v_1 \ldots e_iv_i$ bereits gefunden. Dann wähle Kante e_{i+1} aus $E - \{e_1, \ldots, e_i\}$, so daß

   a) e_{i+1} inzident mit v_i
   b) e_{i+1} keine Schnittkante von $G_i = G - \{e_1, \ldots, e_i\}$, außer es gibt keine andere Alternative.

3) STOP, wenn Schritt 2 nicht weiter durchgeführt werden kann.

Bemerkung (24.5):

Der Algorithmus von Fleury konstruiert eine Linie. Wenn der Graph G eulersch ist, konstruiert er eine Eulertour. In diesem Fall löst er das Briefträgerproblem.

Ist G nicht eulersch, dann muß eine optimale Route mindestens eine Kante mehrmals enthalten. Dabei ist es vorteilhaft, statt G einen neuen Graphen D zu betrachten, bei dem Kanten dupliziert sind.

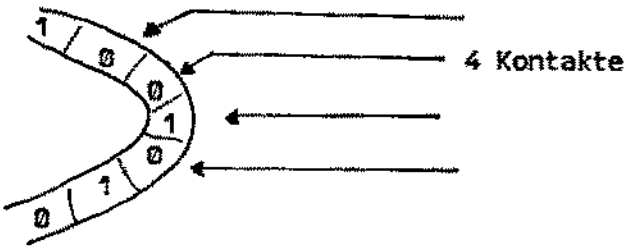
Für eine Kante e wird eine Kante e^* hinzugefügt, die dieselben Endecken und dasselbe Gewicht $w(e^*) = w(e)$ hat.

Nun muß man durch Duplizieren von Kanten D so finden, daß D eulersch ist und $\Sigma w(e^*)$ minimal ist, wobei die Summe über alle Kanten e^* aus $E(D) - E(G)$ läuft. Danach wendet man den Algorithmus von Fleury an.

Bemerkung (24.6):
Der Algorithmus von Fleury ist polynomial. Außerdem gibt es polynomiale Algorithmen, um den Graphen D für einen nicht eulerschen Graphen G zu finden.

24C : Konstruktion einer Computertrommel

Auf einer rotierenden Trommel seien 2^n Segmente aufgetragen, wobei jedes Segment entweder ein Leiter oder Nichtleiter ist.



Bei 4 Kontakten ergibt sich die Position der Trommel durch das Wort 0010. Wenn die Trommel sich um ein Segment weiter dreht, haben wir das Wort 1001.

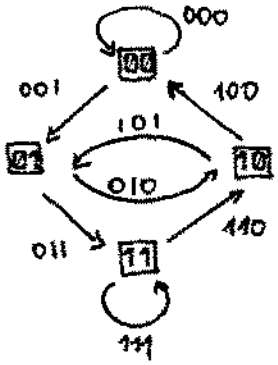
Problem:

Kann man die Trommel so konstuieren, daß die 2^n verschiedenen Positionen der Trommel durch k Kontakte bestimmt werden können?

Vorbemerkung: Man sieht, daß k Kontakte ein k-Tupel aus { 0, 1 } abgreifen. Es gibt 2^k derartige k-Tupel. Wenn also alle 2^n Positioen durch verschiedene k-Tupel bestimmt werden sollen, muß $2^k \geq 2^n$, also $k \geq n$, sein. Im folgenden werden wir sehen, daß k = n ausreicht.

Mathematisierung:
Definiere einen Digraphen D_n folgendermaßen: Die Ecken von D_n sind (n-1)-Tupel $p_1 p_2 \ldots p_{n-1}$ mit $p_i \in \{ 0, 1 \}$. Es gibt einen Bogen mit dem Anfang $p_1 \ldots p_{n-1}$ und dem Ende $q_1 \ldots q_{n-1}$ genau dann, wenn $p_{i+1} = q_i$, $i = 1, \ldots, n-2$.
Ferner werden die Bögen mit dem Gewicht $p_1 \ldots p_{n-1} q_{n-1}$ versehen.

Illustration am Beispiel D_3:



Eulertour:

000
001
011
111
110
101
010
100

D_n ist zusammenhängend und jede Ecke hat den Eingangsgrad 2 und den Ausgangsgrad 2. Man kann beweisen, daß D_n eine gerichtete Eulertour besitzt. Diese Eulertour liefert eine Folge der Länge 2^n, die das Problem löst.
Dabei betrachtet man die Kanten der gerichteten Eulertour von D_n. Im Fall d_4 ist 00011101 eine Lösung, wobei die ersten Ziffern der Gewichte der Kanten einer festgewählten Eulertour zugrunde liegen.

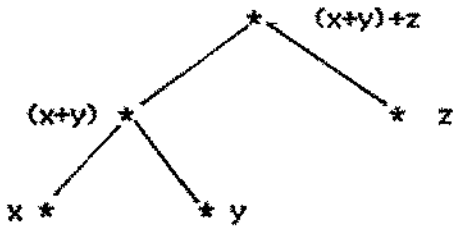
Kapitel 25 : Bäume mit Wurzeln

25A : Grundbegriffe

Definition (25.1):

Ein gerichteter Baum T heißt ein **BAUM MIT WURZEL**, wenn T genau eine Ecke mit dem Eingangsgrad 0 und alle anderen Ecken mit dem Eingangsgrad 1 besitzt.

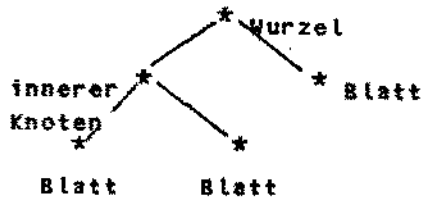
Beispiel:
Berechnungsbaum des Terms (x+y)*z

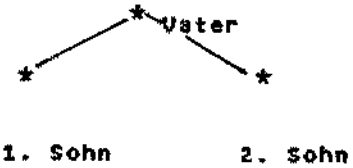


Vereinbarung:
Wenn aus der Zeichnung oder dem Kontext die Orientierung eines Baumes mit Wurzel hervorgeht, lassen wir die Orientierung weg.
Wir sprechen von einem Baum, wenn aus dem Zusammenhang klar ist, daß es sich um einen Baum mit Wurzel handelt.

Bezeichnungen (25.2):
Eine Ecke mit dem Ausgangsgrad 0 heißt ein **BLATT**.
Eine Ecke mit einem Ausgangsgrad > 0 heißt ein **INNERER KNOTEN**.
Ist a ein innerer Knoten mit einem Bogen zu dem Knoten b, so heißt a VATER von b und b SOHN von a. Die Söhne werden häufig durchnumeriert. Daher werden folgende 2 Bäume als verschieden angesehen:

Beispiele zu den Bezeichnungen:



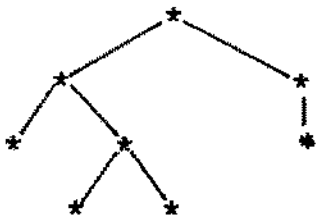


Gleichbedeutend mit dem Begriff 'Ecke' ist der Begriff 'KNOTEN'.

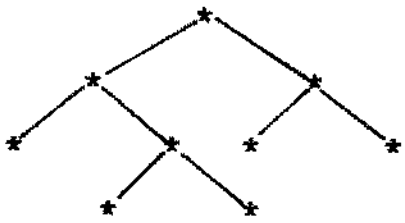
Definition (25.3):

Ein M-ÄRER BAUM ist ein Baum, bei dem jeder innere Knoten höchstens m Söhne hat.
Ein STRIKTER M-ÄRER Baum (oder: M-BAUM) ist ein Baum, bei dem jeder innere Knoten genau m Söhne hat.

Beispiele:
Binärer Baum (m=2):



Strikter binärer Baum (Binärbaum):

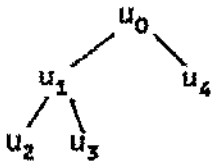


Definition (25.4):

Das NIVEAU l(v) eines Knotens v ist rekursiv definiert durch

$l(v) = 0$ für die Wurzel v des Baumes
$l(u) = l(v)+1$, wenn v der Vater von u ist.

Beispiel:



$l(u_0) = 0.$

$l(u_1) = l(u_4) = 1.$

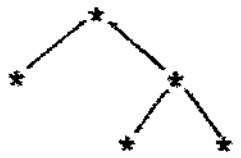
$l(u_2) = l(u_3) = 2.$

Definition (25.5):

Die HÖHE h(T) eines Baumes T ist definiert durch

$$h(T) = \max \{ l(v) \mid v \in V(T) \}.$$

Beispiel:



h(T) = 2.

Bemerkung (25.6):
Ein m-Baum mit i inneren Knoten hat insgesamt m*i+1 Knoten, denn jeder innere Knoten hat m Söhne und außerdem muß die Wurzel mitgezählt werden.

Satz (25.7):

Sei T ein m-Baum.
a) Wenn T i innere Knoten hat, dann ist die Anzahl b der Blätter:

$$b = (m-1)i + 1.$$

b) Wenn T b Blätter hat, dann ist $i=(b-1)/(m-1)$ und die Anzahl n aller Knoten ist $n = (mb-1)/(m-1)$.

c) Wenn T n Knoten hat, dann ist $i = (n-1)/m$ und $b = [(m-1)n + 1]/m$.

Beweis:
Sei i(T) die Anzahl der inneren Knoten von T, B(T) die Anzahl der Blätter von T und K(T) die Anzahl der Knoten von T.

a)
Induktionsanfang:
Aus i(T)=1 folgt B(T) = b = m, also $b = (b-1)1 + 1$.

Induktionsschritt:
Für alle Bäume mit i(T) = k, $k \geq 1$, gelte die Behauptung.
Sei nun T ein Baum mit i(T) = k+1.
Wir nehmen den Vater eines Blattes samt seiner Söhne heraus.
Dann gilt für den verbleibenden Baum T_1 : $B(T_1) = (m-1)k + 1$.
Nimmt man den herausgenommenen inneren Knoten wieder hinzu, gibt es m zusätzliche Blätter und ein Blatt geht verloren.
Also haben wir:

$$b = B(T_1)+m-1 = (m-1)k + (m-1) + 1 = (m-1)(k+1) + 1.$$

b) Aus a) folgt sofort, daß $i=(b-1)/(m-1)$.
Aus (25.6) folgt $n = m\,i + 1$; setzt man i ein, so ergibt sich:

$$n = m[(b-1)/(m-1)] + 1 = (mb-1)/(m-1).$$

c) Aus (25.6) folgt $i(T) = (n-1)/m$ und wegen b) folgt

Satz (25.9):

Für die Höhe eines m-Baumes mit b Blättern gilt:

$$h \geq \log_m b.$$

Beweis:

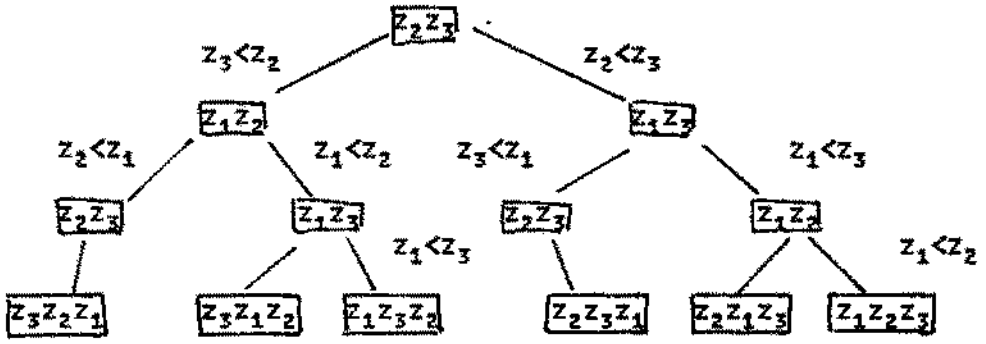
Aus $b \leq m^h$ folgt $\log_m b \leq \log_m m^h = h$.

Satz (25.10):

Die maximale Anzahl von Vergleichen, die man benötigt, um n Zahlen zu sortieren, ist $O(n \log_2 n)$.

Beweis:
Jedem Sortierverfahren, das Paare von Zahlen vergleicht, liegt ein binärer Entscheidungsbaum zugrunde. Ein solcher Baum hat mindestens n! Blätter bei n Zahlen.

Für Zahlen z_1, z_2 und z_3, die paarweise verschieden sind, hat man zum Beispiel folgenden Binärbaum:



Gleichgültig, wie geschickt oder ungeschickt ein solcher Entscheidungsbaum konstruiert ist, nach Satz (25.9) folgt, daß seine Höhe bei n! Blättern mindestens $\log_2 n!$ beträgt.

Dies läßt sich wegen $n! < n^n$ zu $O(n \log_2 n)$ abschätzen.

Bemerkung:
Es gibt Sortieralgorithmen, die diese Schranke 'erreichen'.

$$n = \frac{mb-1}{m-1} \quad \Rightarrow b = \frac{(m-1)n + 1}{m} .$$

Satz (25.8):

Für die Anzahl b der Blätter eines m-Baumes der Höhe h gilt:

$$h - 1 < b \leqslant m^h .$$

Beweis:

a) $b \leqslant m^h$,

Induktion über h:

Für einen m-Baum der Höhe 1 gilt $b \leqslant m^1 = m$, da die Wurzel höchstens m Söhne hat.
Ein m-Baum der Höhe h, h > 1, hat höchstens m Unterbäume, die von den m Söhnen der Wurzel ausgehen. Diese Unterbäume haben höchstens die Höhe h-1 und nach Induktionsvoraussetzung höchstens m^{h-1} Blätter. Also hat der m-Baum der Höhe h höchstens $m*m^{h-1} = m^h$ Blätter.

b) $h - 1 < b$.

i) h = 1 => 1 - 1 = 0 < b; richtig, da mindestens ein Blatt, nämlich die Wurzel vorhanden ist, bei einem m-Baum sogar m > 0 Blätter: die Söhne der Wurzel.

ii) h > 1:
Der Baum hat mindestens einen Unterbaum, der von einem Wurzelsohn ausgeht (sonst wäre h = 1). Dieser Unterbaum hat nach Induktionsvoraussetzung mindestens h-2 Blätter.

Bei einem strikten m-ären Baum (m-Baum) hat der Baum auch einen zweiten Unterbaum (mit mindestens einem Blatt) oder ein Blatt als Wurzelsohn, da die Wurzel genau m > 1 Söhne haben muß.

Bei einem nicht-strikten m-ären Baum (z.B. bzw. einem m-ären Baum mit m=1, also einer 'Liste') hat jeder innere Knoten 1 Sohn. Der Baum hat b=1 Blätter und h-1 < b gilt nicht.

Aussage b) gilt also nur für strikte m-äre Bäume, Aussage a) auch noch für allgemeine m-äre Bäume.

258 : Weglängen auf vollständigen binären Bäumen

Definition (25.11):

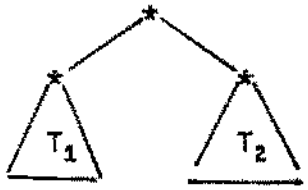
Die ÄUßERE WEGLÄNGE W(T) eines vollständigen, binären Baumes ist die Summe der Niveaus aller Blätter, also $W(T) = \Sigma_{v \in B} l(v)$, B Menge der Blätter.
Die INNERE WEGLÄNGE I(T) eines vollständigen, binären Baumes ist die Summe der Niveaus aller inneren Knoten, also $I(T) = \Sigma_{v \in I} l(v)$, I Menge der inneren Knoten.

Lemma (25.12):

Besitzt der vollständige binäre Baum T n innere Knoten, so gilt

$$W(T) = I(T) + 2n.$$

Beweis:
Wir zerlegen T in zwei Teilbäume T_1 und T_2, deren Wurzeln gerade die zwei Söhne der Wurzel von T sind:

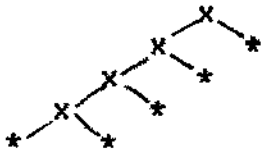


Dann gilt $I(T) = I(T_1) + I(T_2) + n-1$, denn T_1 und T_2 haben $n-1$ innere Knoten. Durch Hinzufügen der Wurzel von T vergrößert sich das Niveau dieser n-1 Knoten um 1.

Genauso überlegt man sich, daß $W(T) = W(T_1) + W(T_2) + n+1$ ist. Man kann dann die Differenz $D(T) = W(T) - I(T)$ bilden und findet, daß für einen Baum T mit n inneren Knoten $D(T) = D(T_1) + D(T_2) + 2$ gilt.

$D(T) = 2n$ ist die Lösung dieser Rekurrenzrelation mit dem Startwert $D(0) = 0$. Daraus ergibt sich nun, daß $W(T) - I(T) = 2n$ ist.

Beispiel:
Ein vollständiger, binärer Baum mit n = 4 inneren Knoten von größtmöglicher innerer und äußerer Weglänge:



x = innere Knoten

Bemerkung (25.13):

Für einen vollständigen binären Baum T mit n inneren Knoten von größtmöglicher äußerer und innerer Weglänge gilt:

$$I(T) = \Sigma_{i=0\ldots n-1}\ i = (n^2-n)/2$$

$$W(T) = (n^2+3n)/2.$$

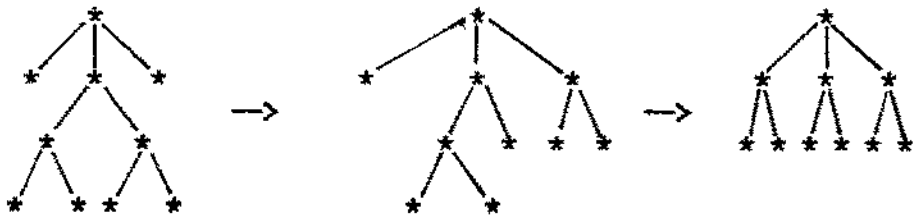
Definition (25.14):

Ein vollständiger binärer Baum T der Höhe h heißt BALANZIERT, wenn alle Blätter von T auf dem Niveau h-1 oder h sind.

Satz (25.15):

Ein vollständiger binärer Baum T mit n inneren Knoten, der eine minimale äußere Weglänge besitzt, hat alle Blätter auf den Niveaus h-1 und h für ein festes h (T ist also balanziert).

Beweis:
Sei T ein vollständiger binärer Baum mit minimaler äußerer Weglänge. Sei L das größte und l das kleinste Niveau, auf denen Blätter existieren.
Nimmt man an, daß $L \geq l+2$ ist (die zu zeigende Behauptung also nicht gilt), so entfernt man zwei Blätter vom Niveau L, die Brüder sind und macht den Vater zu einem Blatt. Die beiden Blätter werden nun an ein Blatt des Niveaus l eingehängt:



Eine solche Modifikation des Baumes ändert die Anzahl der inneren

Knoten und die Anzahl der Blätter nicht. Die äußere Weglänge ändert sich bei einem solchen Schritt um L-(l+1), wobei L-(l+1) > 0 ist.

Dies widerspricht der Annahme, womit die Behauptung gezeigt ist.

Lemma (25.16):

Für die Menge B(T) der Blätter eines vollständigen binären Baumes T gilt:

$$\Sigma_{b \in B(T)} 2^{-l(b)} = 1.$$

Beweis:
Für einen Baum der Höhe h = 1 gilt 1/2 + 1/2 = 1.
Sei die Behauptung für alle Bäume der Höhe h ⩾ 1 bewiesen.

Sei T ein vollständiger Baum der Höhe h+1. Seien T_1, T_2 die beiden Teilbäume, die als Wurzeln die beiden Söhne der Wurzel von T haben. Dann gilt nach Induktionsvoraussetzung:

$$\Sigma_{v \in B(T_i)} 2^{-l(v)} = 1 \quad \text{für } i = 1, 2.$$

Also haben wir für T:

$$\begin{aligned}
&\Sigma_{v \in B(T)} 2^{-l(v)} \\
&= \Sigma_{v \in B(T_1)} 2^{-(l(v)+1)} + \Sigma_{v \in B(T_2)} 2^{-(l(v)+1)} \\
&= 1/2 \left(\Sigma_{v \in B(T_1)} 2^{-l(v)} + \Sigma_{v \in B(T_2)} 2^{-l(v)} \right) \\
&= 1.
\end{aligned}$$

Satz (25.17):

Die minimale äußere Weglänge eines vollständigen binären Baumes T mit n inneren Knoten ist:

$$W(T_n) = (n+1)\log_2(n+1) + (n+1)(2-\varphi-2^{1-\varphi}),$$

wobei $\varphi = \log_2(n+1) - \lfloor \log_2(n+1) \rfloor$ mit $0 \leqslant \varphi < 1$ ist.

Beweis:
Ein vollständiger binärer Baum T_n mit minimaler äußerer Weglänge und n inneren Knoten hat alle seine Blätter auf den Niveaus l und l+1, wobei l+1 die Höhe von T_n ist.

Sei k die Anzahl der Blätter auf dem Niveau l, dann ist n+1-k die Anzahl der Blätter auf dem Niveau l+1, da T_n n+1 Blätter hat (25.7a). Es gilt $1 \leq k \leq n+1$.

Nach (25.16) gilt: $k\,2^{-l} + (n+1-k)\,2^{-l-1} = 1$ und daher $k = 2^{l+1} - n - 1$.

Wegen $k \geq 1$, $2^{l+1} > n+1$ und $k \leq n+1$ folgt $2^{l} \leq n+1$ und daraus dann $l = \lfloor \log_2 n+1 \rfloor$.

Wir haben also $k = 2^{l+1} - n - 1$ und $l = \lfloor \log_2 (n+1) \rfloor$, woraus folgt:

$$k = 2^{\log_2 (n+1)} - n - 1.$$

Die minimale äußere Weglänge ist nun

$$W(T_n) = l\,k + (l+1)\,(n+1-k)$$

$$= (n+1)\,\log_2 (n+1) + (n+1)\,(2-\varphi-2^{1-\varphi}),$$

mit $\varphi = \log_2 (n+1) - \lfloor \log_2 (n+1) \rfloor$.

Bemerkung (25.18):
Es ist nicht schwierig, das obige Ergebnis auch für vollständige m-äre Bäume mit n inneren Knoten herzuleiten. Man muß dabei eine Analogie von Lemma (25.12) benutzen: $W(T) = (m-1)\,I(T) + mn$.

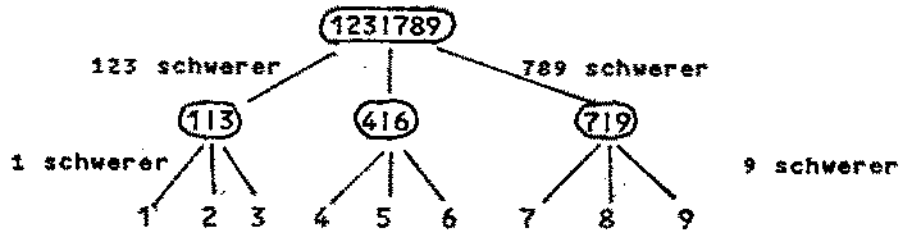
Dabei ergibt sich dann: $W(T) = l\,(n(m-1)+1) - (m^{l+1}-m)/(m-1) + mn$, wobei $l = \lfloor \log_m ((m-1)n+1) \rfloor$ ist.

Definition (25.19):

Die DURCHSCHNITTLICHE ENTFERNUNG E(T) zu einem Blatt ist definiert durch $E(T) = W(T)/b$, wobei b die Anzahl der Blätter ist. Die DURCHSCHNITTLICHE ENTFERNUNG $E^*(T)$ zu einem inneren Knoten ist definiert durch $E^*(T) = I(T)/i$, wobei i die Anzahl der inneren Knoten ist.

Beispiel (25.20):
Es seien n Münzen gegeben, wobei n-1 Münzen vom gleichem Gewicht sind, eine Münze jedoch schwerer ist. Es stehe eine Waage zur Verfügung, die zwei beliebige Mengen von Münzen vergleichen kann. Die Waage kann nach rechts oder nach links ausschlagen oder die Gleichheit anzeigen.
Wieviele Gewichtsvergleiche sind notwendig, um die schwerere Münze herauszufinden?

Unsere Testprozedur besteht aus einem Entscheidungsbaum, den wir für 9 Münzen, die mit 1, 2, ... , 9 durchnumeriert sind, erklären:



Nach Satz (25.9) folgt $h \geq \log_m b$, also $h \geq \log_3 9 = 2$. In unserem Beispiel sieht man, daß man mit zwei Wägungen die schwerere Münze finden kann.

Für die Weglänge W(T) findet man bei einem ternären Baum mit n inneren Knoten und daher 2n+1 Blättern:

$$W(T) = (2n+1)\log_3(2n+1) + 1/2\ (2n+1)(3-2\varphi-3^{1-\varphi}),$$

wobei $\varphi = \log_3(2n+1) - \mathrm{int}(\log_3(2n+1))$ ist.

Die kleinstmögliche durchschnittliche Weglänge zu einem Blatt ist:

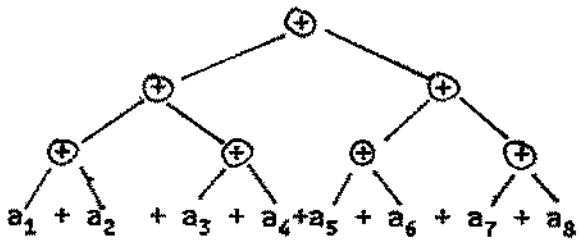
$$E(T) = W(T)/2n+1 = \log_3(2n+1) + 1/2\ (3-2\varphi-3^{1-\varphi})$$

für $\varphi = 0$, was bestimmt dann eintritt, wenn die Anzahl der Blätter 2n+1 eine Potenz von 3 ist.

25C : Bäume und parallele Algorithmen

Unter einem ARITHMETISCHEN AUSDRUCK wollen wir im folgenden ein Wort A(n) verstehen, das aus einer Anzahl von n reellen Zahlen und Variablen mit Hilfe von Operationen aus (+, -, *, /) aufgebaut ist, also zum Beispiel $a_1 + a_2 + \ldots a_8$.

Benutzt man einen Prozessor, so erfordert die Berechnung eines Ausdrucks A(n) auf sequentielle Weise n-1 Zeiteinheiten. Wird eine beliebige Anzahl von Prozessoren benutzt, kann ein Ausdruck A(n) günstigstenfalls in $\lceil \log_2 n \rceil$ Zeiteinheiten ausgewertet werden.



Bezeichnet $T_k(A(n))$ die Anzahl der Zeiteinheiten für den arithmetischen Ausdruck A(n) bei k Prozessoren, dann gilt nach (25.9):

$$T_k(A(n)) \geq \lceil \log_2 n \rceil .$$

Im obigen Beispiel kann man also mit k=4 Prozessoren in 3 Zeiteinheiten den Ausdruck $a_1 + \ldots + a_8$ auswerten.

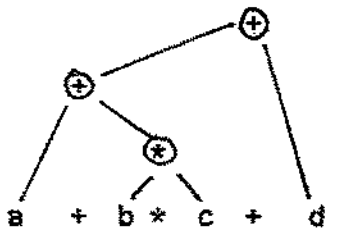
Dabei haben wir das Assoziativgesetzfür die natürlichen Zahlen ausgenutzt, um von einer sequentiellen Verarbeitung, etwa von

$$(((\ldots(a_1+a_2)+a_3)+\ldots)+a_8,$$

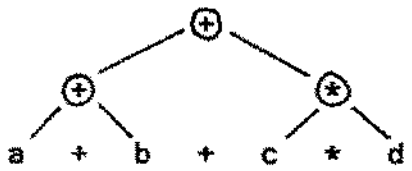
zu einer parallelen Verarbeitung zu kommen.

In ähnlicher Weise kann die Anwendung des Kommutativgesetzes zur Reduktion der Höhe des Baumes führen:

Sei beispielsweise A(4) = (a+b*c)+d zu berechnen.



Bei einem Prozessor hat man $T_1(A(4)) = 3$ Zeiteinheiten. Wenngleich man durch die Anwendung des Assoziativgesetzes in diesem Fall nicht die Höhe des Baumes reduzieren kann, führt doch die Anwendung des Kommutativgesetzes zu B(4):= (a+d) + b*c und folgendem Baum:



Bei zwei Prozessoren hat man $T_2(B(4)) = 2$ Zeiteinheiten.

Um diese Überlegungen auszunutzen, muß man von einem sequentiellen Rechner zu einem parallelen Rechner übergehen.
Das klassische Modell eines Rechners nach John von Neumann (1946) ist auf folgendem Prinzip aufgebaut:

Ein Prozessor P verarbeitet seriell die Daten aus einem Speicher S mit der Kontrolleinheit K. Dieses Rechnermodell erfordert natürlich auch einen seriellen Algorithmus.
Abweichend davon kann man sich nun auf verschiedenen Ebenen Parallelität vorstellen:

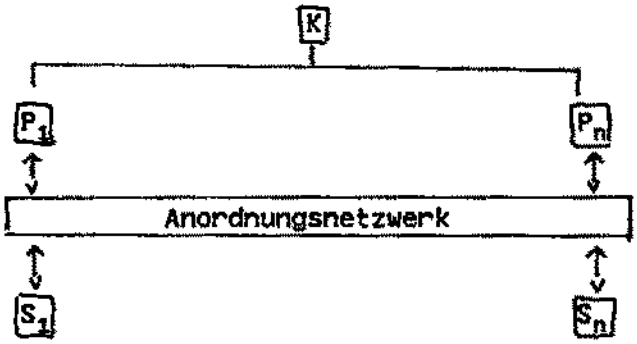
(1) innerhalb der Kontrolleinheit

(2) unter den Prozessoren

(3) unter den Speichern

(4) im Steuernetzwerk für die Daten.

Unter den vielen Möglichkeiten wollen wir nur eine einzige herausgreifen:

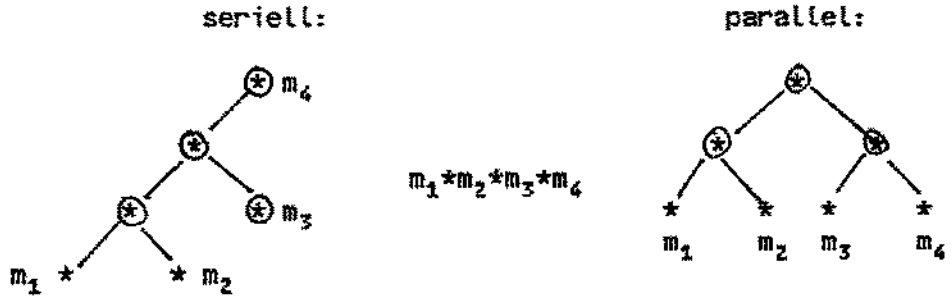


Alle Prozessoren $P_1, \ldots, P_n$ interpretieren die gleichen Befehle und führen sie mit verschiedenen Daten aus den Speichern $S_1, \ldots, S_n$ aus (Feldrechner, Pipelinerechner).

Für parallele Rechner sind in erster Linie auch parallele Algorithmen von Bedeutung. Im folgenden geben wir zwei wichtige Prinzipien an, um aus seriellen Algorithmen parallele Algorithmen zu konstruieren.

I) Vektorisierung serieller Algorithmen:
Dieses Prinzip beruht auf der Beobachtung, daß Vektoroperationen wie etwa die Addition von n-Tupeln komponentenweise, also parallel, ausführbar sind.

II) Prinzip des rekursiven Doppelns:
Das Prinzip läßt sich am besten mit Hilfe einer Menge $K = \{m_1, \ldots, m_d\}$, $d = 2^k$, und einer assoziativen Operation * beschreiben. Dann ergibt sich folgendes Schema:



Als Anwendungsbeispiel betrachten wir die Rekurrenzrelation für die Fibonaccifolge:

$$a_r = a_{r-1} + a_{r-2},\ a_0 = 0\ ,\ a_1 = 1$$

innerhalb des Bereichs $2 \leq r \leq 2^n$.
Betrachte die Spalte F_r und die Matrix A:

$$F_r = \begin{pmatrix} a_r \\ a_{r-1} \end{pmatrix} \quad A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

Dann erhält man für obige Rekurrenzrelation

$$F_r = A\, F_{r-1} = A^{r-1} F_1.$$

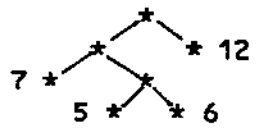
Nun ist die Matrizenmultiplikation assoziativ und laßt sich nach obigem Schema doppeln. Daher kann F_r in $O(\log_2 r)$ Schritten berechnet werden gegenüber $O(r)$ Schritten bei serieller Rechnung.

Kapitel 26 : Gewichtete Bäume

Definition (26.1):

Sei $W = \{ w_1, \ldots, w_n \} \subseteq \mathbb{R}^+$ eine Menge von GEWICHTEN, die aufsteigend sortiert seien: $0 < w_1 \ldots \leq w_n$. Ein vollständiger binärer BAUM T MIT DER GEWICHTUNG W ist ein vollständiger binärer Baum mit n Blättern, die mit $w_1, \ldots, w_n$ markiert sind.
Das GEWICHT G(T) ist definiert durch $G(T) = \Sigma_{i=1\ldots n} w_i l(w_i)$, wobei $l(w_i)$ das Niveau des Blattes w_i ist.

Beispiel:
W = { 5, 6, 7, 12 }.



G(T) = 7*2 + 5*3 + 6*3 + 12 = 59.

Definition (26.2):

Ein Baum T heißt OPTIMAL, wenn G(T) so klein wie möglich ist unter allen vollständigen binären Bäumen mit der Gewichtung W.

Bemerkung:
Es existiert zu vorgegebener Gewichtung W stets ein optimaler Baum. Es gibt ja nur endlich viele vollständige binäre Bäume mit n Blättern.
Einen solchen optimalen Baum kann man mit Hilfe des folgenden Algorithmus finden:

Algorithmus (26.3) [ALGORITHMUS VON HUFFMANN]:

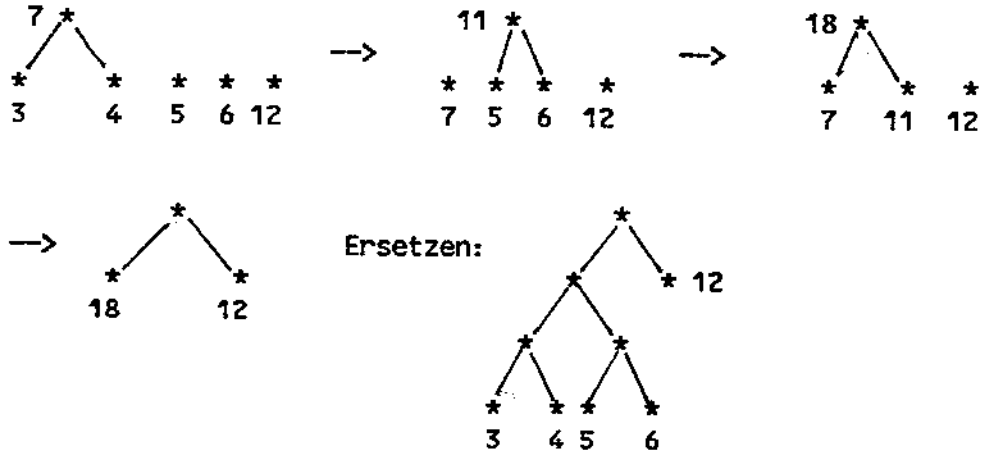
1) Suche die beiden kleinsten Werte von $w_1, \ldots, w_n$. OBdA seien dies w_1 und w_2.

2) Setze $w = w_1 + w_2$.

3) Suche einen optimalen Baum T^* mit n-1 Blättern $w, w_3, \ldots, w_n$.

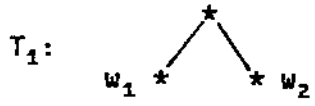
4) Ersetze in T^* das Blatt w durch

Beispiel: $W = \{ 3, 4, 5, 6, 12 \}$



Lemma (26.4):

Sei $W = \{ w_1, \ldots, w_n \} \subseteq \mathbb{R}^+$ eine Menge von Gewichten. Dann gibt es einen optimalen Baum T, der T_1 als Teilbaum enthält, wobei T_1 die folgende Form hat:



Beweis:
Sei T ein optimaler Baum mit der Gewichtung $W = \{ w_1, \ldots, w_n \}$. Sei s ein innerer Knoten von T mit $l(s) = \max \{ l(v) \mid v \text{ innerer Knoten von } T \}$. Die Söhne von s sind dann zwei Blätter w_i und w_j.

Ist $\{ w_1, w_2 \} = \{ w_i, w_j \}$, dann gilt die Behauptung.

Ist $\{ w_1, w_2 \} \cap \{ w_i, w_j \} = \emptyset$, dann vertausche man die Markierung w_1 mit w_i und w_2 mit w_j. Der dadurch entstehende Baum T^* enthält einen Teilbaum T_1 und es gilt:

$$G(T) = w_1 k + w_2 m + w_i (l(s)+1) + w_j (l(s)+1) + \text{Rest}$$

$$G(T^*) = w_i k + w_j m + w_1 (l(s)+1) + w_2 (l(s)+1) + \text{Rest}.$$

Wegen der Definition von s gilt, daß $k, m \leq l(s)+1$ ist und wegen $w_1 \leq w_2 \leq w_i, w_j$ gilt:

$$w_1 k + w_i (l(s)+1) \geq w_i k + w_1 (l(s)+1)$$

und das Analoge für w_2 und w_j. Also ist $G(T^*) \leq G(T)$ und also ist T^* ein optimaler Baum.

Bei den verbleibenden Fällen argumentiert man in derselben Weise.

Lemma (26.5):

Sei T ein Baum mit den Blättern $w_1, \ldots, w_n$. Jedem inneren Knoten s ordne man die Summe w(s) der beiden Söhne von s zu. Dann gilt für die Menge I der inneren Knoten:

$$\Sigma_{s \in I} w(s) = \Sigma_{i=1 \ldots n} w_i l(w_i).$$

Beweis:
Induktionsanfang:
Für n=2 sieht man, daß $w(s) = w_1 + w_2 = w_1 l(w_1) + w_2 l(w_2)$ ist.

Induktionsschritt:
Gelte nun die Behauptung für alle Bäume mit weniger als $n = k > 2$ Blätter. Sei T ein Baum mit den Blättern $w_1, \ldots, w_n$ und seien oBdA w_1 und w_2 Söhne eines Vaters q. Es gilt $q = w_1 + w_2$.

Sei T^* der Baum, der aus T entsteht, wenn man die Blätter w_1, w_2 wegläßt. Nach Induktionsvoraussetzung gilt:

$$\Sigma_{s \in I} w(s) = \Sigma_{s \in I - \{q\}} w(s) + (w_1 + w_2)$$

$$= \Sigma_{i=3 \ldots n} w_i l(w_i) + (w_1 + w_2) l(q) + (w_1 + w_2).$$

Nun ist $l(q) = l(w_1) - 1$ und $l(q) = l(w_2) - 1$, woraus die Behauptung folgt.

Nach diesen Vorbereitungen gelangen wir zur Verifikation des Algorithmus.

Satz (26.6):

Der Algorithmus von Huffmann liefert einen optimalen Baum.

Beweis:
Für n=2 gilt die Behauptung trivialerweise und wir nehmen an, daß sie auch für alle Bäume mit weniger als n Blättern, $n > 2$, richtig sei. Sei nun $W = \{ w_1, \ldots, w_n \}$ gegeben, setze $w = w_1 + w_2$ und betrachte $W' = \{ w, w_3, \ldots, w_n \}$.

Nach Induktionsvoraussetzung liefert der Algorithmus von Huffmann einen optimalen Baum T'. Ersetze in T' w durch den Teilbaum

$T_1 =$

Der so entstandene Baum ist optimal, wie jetzt gezeigt wird.

Angenommen, dies ist nicht der Fall. Dann gibt es einen Baum T^* mit $G(T^*) < G(T)$. T^* darf nach Lemma (26.4) so gewählt werden, daß T^* den Teilbaum T_1 enthält.

Für den Baum $T^* - \{ w_1, w_2 \}$ gilt nach Lemma (26.5), daß

$$G(T^* - \{ w_1, w_2 \}) = G(T^*) - w, \; w = w_1 + w_2.$$

Wir haben also:

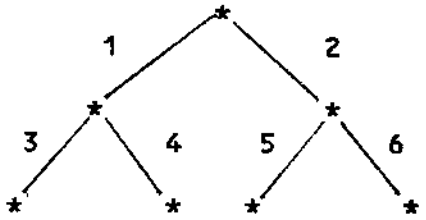
$$\begin{aligned} G(T^* - \{ w_1, w_2 \}) &= G(T^*) - w \\ &< G(T) - w = G(T - \{ w_1, w_2 \}). \end{aligned}$$

Also ist $T - \{ w_1, w_2 \} = T'$ nicht optimal. Widerspruch.

Kapitel 27 : Suchen auf Bäumen und Backtracking

Definition (27.1):

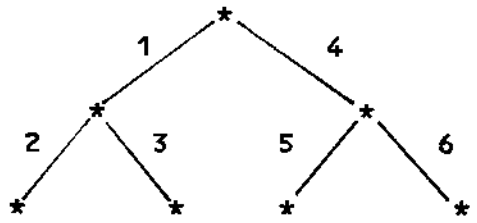
Sei ein Baum mit Wurzel gegeben. Ein Suchverfahren heißt BREATH-FIRST (BREITE-ZUERST-VERFAHREN), wenn zunächst alle Söhne der Wurzel, dann alle Söhne dieser Söhne usw. besucht (aufgesucht) werden.



Breath-first findet beispielsweise beim Dijkstra-Algorithmus Anwendung. Dort beginnt man mit einer Ecke und sucht im Breath-first-Verfahren die nächste Ecke.

Definition (27.2):

Sei ein Baum mit Wurzel gegeben. Ein Suchverfahren heißt DEPTH-FIRST oder TIEFE-ZUERST-VERFAHREN, wenn von der Wurzel ausgehend alle Blätter von links bis rechts besucht werden.



Das Depth-first-Verfahren findet wesentlich öfter Anwendung als das Breath-first-Verfahren. Natürlich gibt es viele Verfahren, die zwischen diesen beiden Suchverfahren liegen. Wir geben hier ein kleines Beispiel, das Vorzüge und Nachteile des Depth-first Verfahrens beleuchtet.

Beispiel:
Wie findet man im folgenden Labyrinth systematisch einen Weg vom Start zum Ziel?

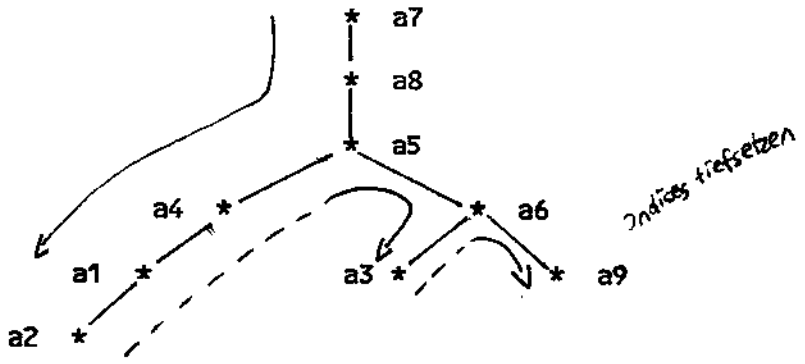
Start: a_7
Ziel : a_9

a1	a2	a3
a4	a5	a6
a7	a8	a9

Depth-first-Verfahren:
Ist man in einem Quadrat angelangt, so nimmt man jeden noch nicht begangenen Weg.

Ist man in einem Quadrat angelangt, von dem aus es keine unbegangenen Wege gibt, so geht man auf dem zuletzt begangenen Weg um ein Quadrat zurück (Backtracking).

Als Baum zu diesem Labyrinth ergibt sich:



mit der Lösung $(a_7, a_8, a_5, a_4, a_1, a_2, a_6, a_3, a_9)$.

Natürlich ist $(a_7, a_8, a_5, a_6, a_9)$ eine elegantere Lösung.

27.3 : Backtracking

Das Backtracking-Verfahren ist ein Tiefe-zuerst-Verfahren. Aufgrund seiner Allgemeinheit ist es für viele Probleme anwendbar und muß als ein Prinzip wie das 'Prinzip der Optimalität' oder 'Divide and conquer' verstanden werden.
Wir fassen das Problem und den Algorithmus sehr allgemein; der Leser wird zum besseren Verständnis auf das Beispiel verwiesen.

Problem:
Aus einer Menge A werden alle n-Tupel $(x_1, \ldots, x_n)$ gesucht, die eine Bedingung $P_n(x_1, \ldots, x_n)$ erfüllen.

Lösungsidee:
Die n-Tupel werden im Verlaufe des Backtracking-Algorithmus konstruiert. Dabei entsteht $(a_1, \ldots, a_{i+1}, x_{i+2}, \ldots, x_n)$ aus $(a_1, \ldots, a_i, x_{i+1}, \ldots, x_n)$, indem man das kleinste Element a_{i+1} aus einer Menge A_{i+1} hinzunimmt, so daß die Eigenschaft $P_{i+1}(a_1, \ldots, a_{i+1})$ gilt.

27.4 Backtracking-Algorithmus

0) Man stellt zwischen n-Tupeln $(a_1, \ldots, a_n)$ und möglichen Konfigurationen eine bijektive Beziehung her. Die Auswahl für a_i wird aus einer Menge A getroffen. Wenn $a_1, \ldots, a_{k-1}$ bereits bestimmt sind, dann sei A_k die Teilmenge von A, die die zulässigen Elemente von A enthält, aus denen a_k ausgewählt werden kann.

1) Ist A_k nicht leer, dann ist a_k das kleinste Element von A_k, das noch nicht ausgewählt wurde.

   Ist $k < n$, setze $k := k+1$ und wiederhole Schritt 1).

   Ist $k = n$, gebe $(a_1, \ldots, a_n)$ als ein Lösungstupel aus.
   Setze $k := k-1$ und wiederhole Schritt 1).

   Ist A_k leer, STOP.

2) Ist A_k leer und $k=1$, existieren keine weiteren zulässigen Tupel, daher STOP.

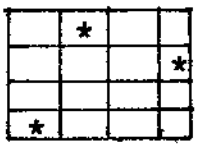
   Ist A_k leer und $k > 1$, setze $k := k-1$, wiederhole Schritt 1).

Bemerkung:
Der Backtrackingalgorithmus ist gewöhnlich von exponentieller Komplexität, wenn man ungünstigste Fälle betrachtet.
Für viele Fälle ist er durchaus anwendbar; im folgenden wollen wir dafür ein Beispiel geben.

Beispiel:
Es sollen 4 Königinnen (Damen) auf einem 4x4-Schachbrett derart aufgestellt werden, daß sie sich gegenseitig nicht schlagen können. Man bestimme alle solche Konfigurationen (Stellungen).

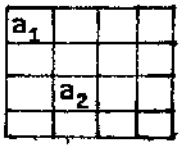
Lösung:
Die erste Aufgabe beim Backtrackingalgorithmus ist es, eine bijek-

tive Entsprechung zwischen Tupeln und Konfigurationen aufzubauen:



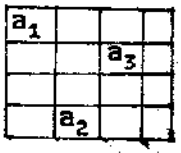
Für die obige Konfiguration von 3 Damen auf dem Schachbrett schreiben wir das 4-Tupel (4, 1, a_3, 2).

Der Algorithmus geht nun folgendermaßen vor:
Wir haben $a_1 = 1$ und das Tupel (1, a_2, a_3, a_4) als Anfang. Nun muß a_2 aus {1, 2, 3, 4} gewählt werden. a_2 kann nicht 1 oder 2 sein, da die Eigenschaft $P_2(a_1, a_2)$ erfüllt sein muß (d. h.: a_1 und a_2 dürfen sich nicht schlagen können). Der kleinste Wert für a_2 ist also 3.



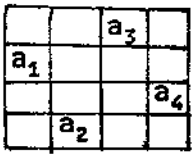
Haben wir (1, 3, a_3, a_4), suchen wir für a_3 einen möglichst kleinen Wert. Jedoch ist für a_3 jeder Wert $\in$ { 1, 2, 3, 4} unmöglich.

Daher müssen wir gemäß Backtracking einen Schritt zurück und a_2 auf 4 erhöhen:



Haben wir (1, 4, a_3, a_4), so ergibt sich $a_3 = 2$. Für das Tupel (1, 4, 2, a_4) läßt sich a_4 nicht $\in$ {1, 2, 3, 4} setzen. Wir müssen also zu $a_3 = 2$ zurück, was sich aber nicht erhöhen läßt. Also müssen wir gemäß Backtracking weiter zu $a_2 = 4$ zurück und schließlich zu a_1.

Nun wird $a_1 = 2$ gesetzt und wir bekommen schrittweise: (2, a_2, a_3, a_4) --> (2, 4, a_3, a_4) --> (2, 4, 1, a_4) --> (2, 4, 1, 3) als erste Lösung:



Nachdem wir das erste Lösungstupel bestimmt haben, gehen wir von (2,4,1,3) ausgehend auf (2, 4, 1, a_4) zurück, aus der Menge der

zulässigen Elemente wird 3 gestrichen und wir wiederholen Schritt 2 solange, bis sich das zweite Lösungstupel (3, 1, 4, 2) ergibt.

Kapitel 28 : Matchings

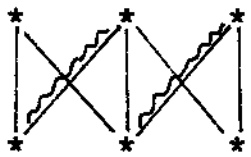
28A : Grundbegriffe

Definition (28.1):

Eine Untermenge M der Kantenmenge E eines Graphen G heißt MATCHING oder PAARUNG, wenn jede Kante aus M zwei verschiedene Ecken hat und keine zwei Kanten benachbart sind, d.h. zwei Kanten haben keine gemeinsame Ecke.

Beispiele:

M_1:

M_2: 

Definition (28.2):

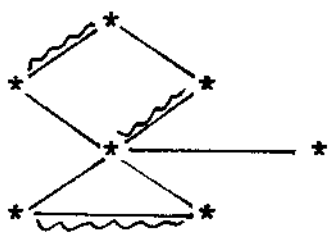
Ein Matching M SÄTTIGT (SATURIERT) eine Ecke v (bzw.: v heißt M-GESÄTTIGT), wenn eine Kante aus M mit v inzidiert.
Ein Matching M heißt PERFEKT, wenn jede Ecke von G M-gesättigt ist.

Beispiel (siehe oben):
M_1 ist ein perfektes Matching, M_2 nicht.

Definition (28.3):

M heißt ein MAXIMALES MATCHING von G, wenn es auf G kein Matching M_* gibt mit $| M | < | M_* |$.

Beispiel:



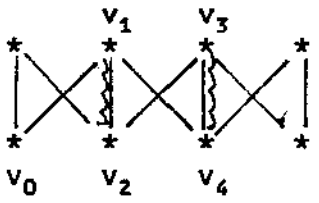
M ist ein maximales Matching, das nicht perfekt ist.

Natürlich ist jedes perfekte Matching auch maximal.

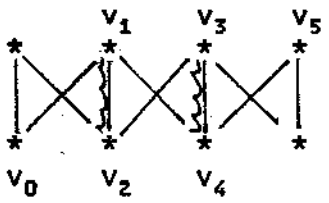
Definition (28.4):

Sei M ein Matching von G. Ein M-ALTERNIERENDER WEG W ist ein Weg in G, dessen Kanten abwechselnd in M und E - M sind.
Ein M-AUGMENTIERTER WEG ist ein M-alternierender Weg, dessen Anfangsecke und Endecke M-ungesättigt sind.

Beispiele:



W_1 ist M-alternierend
$W_1 = v_0v_1v_2v_3v_4$



W_2 ist M-augmentiert
$W_2 = v_0v_1 \ldots v_5$

Satz (28.5):

Ein Matching M von G ist genau dann maximal, wenn G keinen M-augmentierten Weg besitzt.

Beweis:
'=>'
Sei M ein maximales Matching von G. Angenommen, G besitzt einen M-augmentierten Weg $W = v_0v_1 \ldots v_{2m+1}$. (Der Weg W hat 2m Ecken, weil die Kanten aus M nicht benachbart sind und nur Anfangs- und Endecke von W nicht gesättigt sind).
Definiere ein neues Matching M_* durch:

$$M_* = \{ M - \{v_1v_2, v_3v_4, \ldots, v_{2m-1}v_{2m}\} \cup \{v_0v_1, v_1v_2, \ldots, v_{2m}v_{2m+1}\}.$$

Dann gilt $|M_*| = |M| + 1$, im Widerspruch zur Maximalität von M.

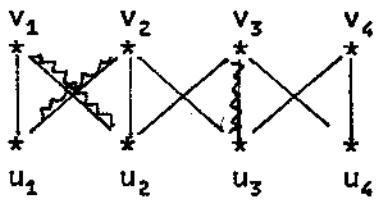
W_2 aus obigem Beispiel liefert ein Matching M_* mit $|M| < |M_*|$.

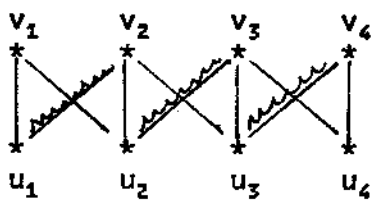
'<='
Sei M ein Matching von G und G besitze keinen M-augmentierten Weg. Wir haben zu zeigen, daß dann M maximal ist. Wir nehmen an, daß M nicht maximal ist, d. h. es gibt es Matching M_* mit $|M| < |M_*|$.

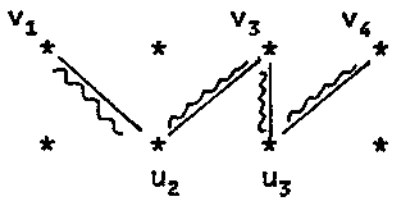
Wir betrachten den durch die symmetrische Differenz

$$Diff(M,M_*) = (M \cup M_*) - (M \cap M_*)$$

induzierten Untergraphen H von G.
Beispiel:

M: 

M_*: 

Diff(M,M_*): 

Jede Ecke von H hat den Eckengrad entweder 1 oder 2 in H, da sie höchstens mit einer Kante von M und mit einer Kante von M_* inzident ist. Daher ist jede Zusammenhangskomponente von H entweder ein gerader Zykel mit Kanten abwechselnd in M und M_* oder ein Weg mit Kanten abwechselnd in M und M_*.

Wegen $|M| < |M_*|$ enthält H mehr Kanten von M_* als von M. Daher gibt es eine Wegkomponente P in H, die mit einer Kante von M_* anfängt und mit einer Kante von M_* aufhört.

Der Anfang und das Ende dieses Weges sind M_*-gesättigt, also M-ungesättigt. Daher ist P ein M-augmentierter Weg. Widerspruch.

28B : Heiratssatz

Bezeichnung:
Sei $S \subseteq V$ eine Menge von Ecken in G. Die **NACHBARSCHAFTSMENGE N(S)** von S besteht aus allen Ecken, die zu den Ecken von S benachbart sind.

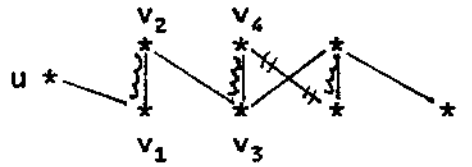
Satz (28.6):

Sei G ein bipartiter Graph mit Bipartition (X,Y). G besitzt genau dann ein Matching, das jede Ecke von X saturiert, wenn gilt:

$$|N(S)| \geq |S| \quad \text{für alle } S \subseteq X. \quad (*)$$

Beweis:
'=>'
Sei M ein Matching, das alle Ecken in X saturiert und sei $S \subseteq X$. Da die Ecken von S durch die Kanten von M mit |S| verschiedenen Ecken verbunden werden, gilt $|N(S)| \geq |S|$.

'<='
Sei G bipartit und gelte (*). Angenommen, G besitzt kein Matching, das alle Ecken von X sättigt. Sei M_* ein maximales Matching von G und sei u eine Ecke, die durch M_* nicht gesättigt ist. Sei Z die Menge aller Ecken, die mit u durch M_*-alternierende Wege verbunden sind.
Nach (28.5) folgt, daß u die einzige Ecke in Z ist, die nicht M_*-gesättigt ist.



$Z = \{ u, v_1, v_2, v_3, v_4 \}$
$S = \{ u, v_2, v_4 \}$
$T = \{ v_1, v_3 \}$

Setze $S = Z \cap X$ und $T = Z \cap Y$. Die Ecken in S-{u} werden unter M_* den Ecken in T zugeordnet. Daher gilt $|T| = |S|-1$ und außerdem $T = N(S)$, weil jede Ecke in N(S) mit u durch einen M_*-alternierenden Weg verbunden ist.
Daraus folgt aber

$$|N(S)| = |T| = |S| - 1 < |S|,$$

ein Widerspruch.

Definition (28.7):

Ein Graph G heißt K-REGULÄR, wenn jede Ecke von G den Eckengrad k hat.

Satz (28.8):

Ist G ein k-regulärer, bipartiter Graph mit k>0, so besitzt G ein perfektes Matching.

Beweis:
Sei G k-regulär mit der Bipartition (X,Y). Also gilt $k\,|X| = k\,|Y| = |E|$. Wegen $k > 0$ haben wir also $|X| = |Y|$.
Sei nun $S \subseteq X$, sei E_1 die Menge aller Kanten, die mit den Ecken in S inzident sind und E_2 die Menge aller Kanten, die mit den Ecken in N(S) inzident sind.
Wegen der Definition von N(S) haben wir $E_1 \subseteq E_2$ und daher gilt:

$$k\,|N(S)| = |E_2| \geq |E_1| = k\,|S|,$$

woraus $|N(S)| \geq |S|$ folgt.

Nach (28.6) folgt, daß G ein Matching besitzt, das jede Ecke in X sättigt. Da |X| = |Y|, ist das Matching perfekt.

Bemerkung:
Kennt jede Frau eines Dorfes k Männer und jeder Mann k Frauen, dann können die Frauen und die Männer Paare bilden, wobei sich die Partner jeweils kennen. In diesem Zusammenhang wird der Satz (28.8) manchmal als Heiratssatz bexeichnet.

28C : Ungarische Methode

Problem:
In einer Fabrik gibt es n Arbeiter $x_1, \ldots, x_n$, die für n Arbeiten $y_1, \ldots, y_n$ eingesetzt werden können, wobei jeder Arbeiter für eine oder mehrere Arbeiten geeignet ist. Kann man die Arbeiter so einsetzen, daß jeder eine Arbeit bekommt, für die er qualifiziert ist?

Mathematisierung:
Gegeben ist ein Graph G mit einer Bipartition (X,Y), wobei $X=\{x_1, \ldots, x_n\}$ und $Y=\{y_1, \ldots, y_n\}$ ist; eine Ecke x_i ist mit einer Ecke y_i verbunden genau dann, wenn der Arbeiter x_i für die Arbeit y_i qualifiziert ist.

(28.9): Algorithmus UNGARISCHE METHODE

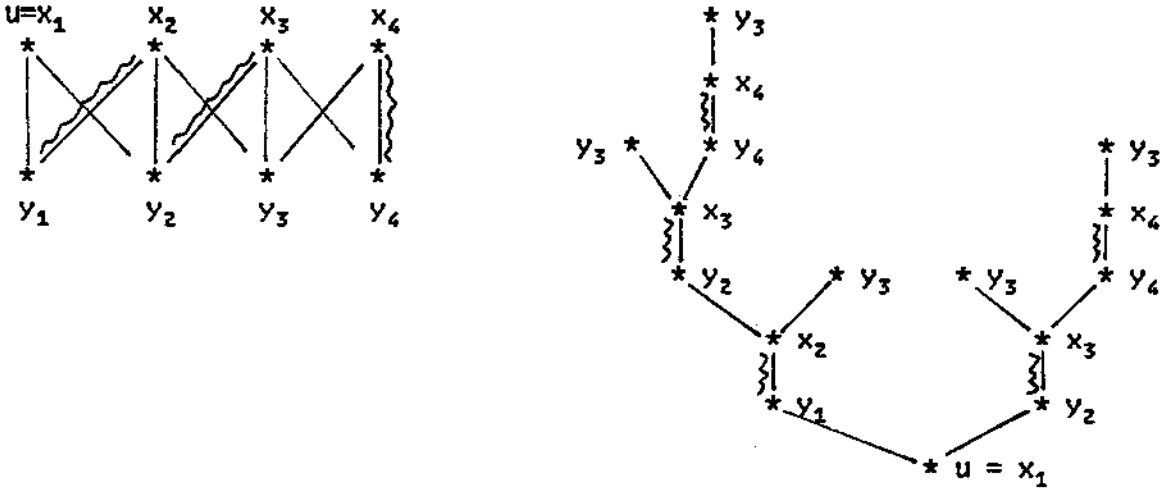
1) Beginne mit einem beliebigen Matching M (evtl. nur eine Kante)

2) Sättigt M alle Ecken von X, dann STOP. Wenn nicht, wähle eine M-ungesättigte Ecke $u \in X$.

3) Suche einen M-augmentierten Weg P mit Anfang u.
Existiert kein solches P, STOP - es gibt kein perfektes Matching.
Andernfalls ergibt Diff(M,E(P)) ein vergrößertes Matching M_*.
Setze $M = M_*$ und gehe nach 2).

Bemerkung:
Für den Schritt 2) wird ein Baum entwickelt, der auch UNGARISCHER BAUM heißt.

Beispiel:



Ungarischer Baum

Ein Ungarischer Baum von u ist ein Baum H mit:

1) $u \in V(H)$,

2) Für jede Ecke v von H ist der (u,v)-Weg ein M-alternierender Weg.

Bei dieser Prozedur läßt man den Baum von der Wurzel u ausgehend wachsen, so daß bei jedem Niveau entweder alle Ecken M-gesättigt sind oder H eine M-ungesättigte Ecke v verschieden von u enthält. Im letzteren Fall kann das Wachsen gestoppt werden und man hat den M-augmentierten (u,v)-Weg P gefunden.

Bemerkung:
Für einen gewichteten bipartiten Graphen gibt es eine Variante dieses Algorithmus. Ein gewichteter Graph tritt etwa dann auf, wenn beim Heiratsproblem jede Frau die ihr bekannten Männer nach einer Rangordnung einteilt.
Obiger Algorithmus und seine Varianten sind von polynomialer Komplexität.

Kapitel 29 : Netzwerke

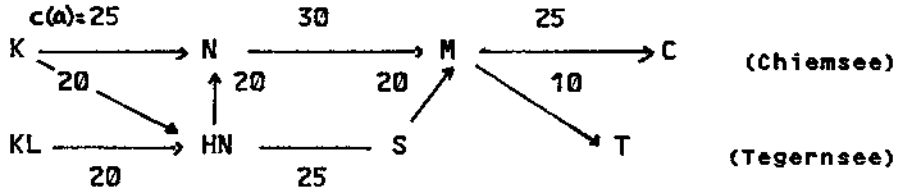
29A : Maximumflüsse und Minimumschnitte

Definition (29.1):

Ein NETZWERK N ist ein Digraph D mit zwei nichtleeren disjunkten Eckenmengen X, Y und einer Funktion c : A ---> $\mathbb{N}$, wobei A = A(D) die Menge der Bögen des Digraphen D ist.
Die Ecken mit Eingangsgrad 0 heißen QUELLEN (X), die mit Ausgangsgrad 0 heißen SENKEN (Y) von N. Die Ecken in I = V - (XuY) heißen INNERE ECKEN.
Die Funktion c heißt KAPAZITÄTSFUNKTION; für a ∈ A ist c(a) die KAPAZITÄT von a.

Beispiel:

Ferienverkehr nach Bayern:



Kapazität: Anzahl der Autos in 100 pro Stunde.
Quellen : K, KL.
Senken : Chiemsee (C), Tegernsee (T).

Bezeichnungen:

Ist f : A ---> $\mathbb{R}^+$ eine Funktion und ist K $\subseteq$ A, dann sei

$$f(K) := \Sigma_{a \in K} f(a).$$

Für S $\subseteq$ V sei C(S)=V-S. (S, C(S)) ist dann die Menge der Bögen mit Anfangsecke in S und Endecke in C(S). Setze $f^+(S) := f(S,C(S))$ und $f^-(S) := f(C(S),S)$.

Definition (29.2):

Ein FLUß f in einem Netzwerk N ist eine Funktion $f : A \longrightarrow \mathbb{N}$, so daß gilt:

(1) $0 \leqslant f(a) \leqslant c(a)$ für alle $a \in A$.

(2) $f^-(v) = f^+(v)$ für alle $v \in I$ (Erhaltungsbedingung).

Der Fluß f mit $f(a)=0$ für alle $a \in A$ heißt der NULLFLUß. Ist S eine Eckenmenge in N und f ein Fluß in N, dann heißt $f^+(S)-f^-(S)$ der RESULTIERENDE FLUß AUS S und $f^-(S) - f^+(S)$ der RESULTIERENDE FLUß NACH S.

$val\ f := f^+(x) - f^-(x)$ heißt der WERT von f.

Bemerkung (29.3):

Der resultierende Fluß aus X ist gleich dem resultierenden Fluß nach Y.

Beweis:

Für alle Ecken in V gilt $\Sigma_{v \in V}(f^+(v)-f^-(v)) = 0$ und daraus folgt:

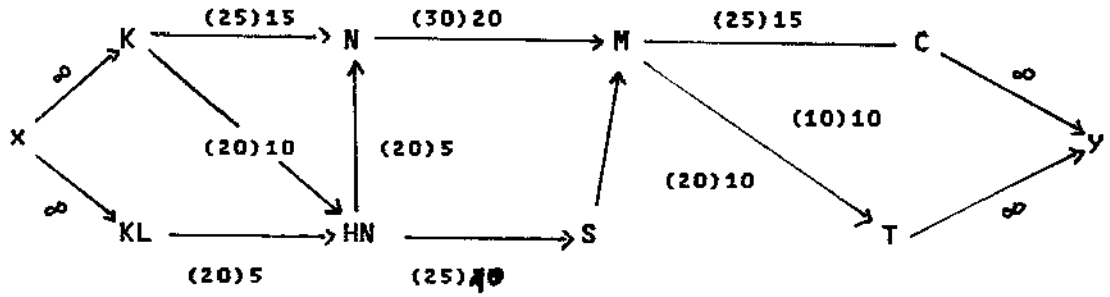
$$\Sigma_{x \in X}(f^+(x)-f^-(x)) + \Sigma_{y \in Y}(f^+(y)-f^-(y))$$

$$+ \Sigma_{v \in V}(f^+(v)-f^-(v)) = 0.$$

Aus (2) folgt:

$$\Sigma_{x \in X}(f^+(x)-f^-(x)) = \Sigma_{y \in Y}(f^-(y)-f^+(y)).$$

Beispiel:



'(x)y' bedeutet: Kapazität c(a) = x
Fluß f(a) = y

Für den Fluß f des Netzwerkes N sind die Bedingungen (1) und (2) der Definition (29.2) erfüllt.

Definition (29.4):

Ein Fluß f in einem Netzwerk N heißt ein MAXIMUMFLUß in N, falls es keinen Fluß f^* in N gibt mit val f < val f^*.

Problem (29.5):

Bestimme in einem Netzwerk N einen Maximumfluß.

Bemerkung (29.6):
Wir wollen das Problem darauf reduzieren, einen Maximumfluß in einem Netzwerk N^* mit nur einer Quelle x^* und nur einer Senke y^* zu bestimmen.

Dabei fügen wir x^* und y^* als neue Ecken ein, verbinden x^* mit allen $x \in X$ durch Bögen der Kapazität ω und jedes y mit y^* durch Bögen der Kapazität ω. Die Flüsse in N und N^* entsprechen sich dabei wie folgt:

Ist f ein Fluß in N, dann ist f^* definiert durch:

$$f^*(a) = \begin{cases} f(a) & \text{für } a \text{ in } N \\ f^+(v)-f^-(v) & \text{für } a = (x^*,v) \\ f^-(v)-f^+(v) & \text{für } a = (v,y^*) \end{cases}$$

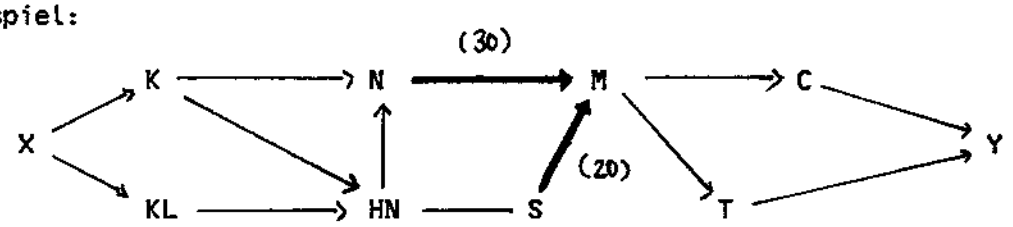
Offensichtlich ist f^* ein Fluß in N^* mit val f^* = val f.

Ist umgekehrt f^* ein Fluß in N^*, dann ist die Einschränkung von f auf N ein Fluß in N mit val f^* = val f. Also können wir uns auf Netzwerke mit nur einer Quelle und nur einer Senke beschränken.

Definition (29.7):

Sei N ein Netzwerk mit nur einer Quelle x und einer Senke y. Ein SCHNITT K in N ist eine Menge von Bögen der Form (S,C(S)) mit $x \in S$ und $y \in C(S)$.
Die Kapazität eines Schnittes K ist die Summe der Kapazität seiner Bögen: cap K = $\Sigma_{a \in K} c(a)$.

Beispiel:



S = {x,K,KL,N,HN,S} C(S) = {M,C,T,y} K = (S,C(S))
cap K = 50.

Im folgenden betrachten wir nur noch Netzwerke mit nur einer Quelle x und einer Senke y.

Lemma (29.8):

Für jeden Fluß f und jeden Schnitt K = (S,C(S)) in N ist val f $\leq$ cap K.

Beweis:
Zunächst sieht man, daß wegen der Erhaltensbedingung (2)

$$f^+(v)-f^-(v) = 0$$

ist für alle Ecken $v \in S - \{x\}$.

Für x gilt $f^+(x) - f^-(x) = f^+(x) = \text{val } f$.
Daraus erhält man sofort

$$\text{val } f = \Sigma_{v \in S}(f^+(v) - f^-(v)) = f^+(S) - f^-(S).$$

Die Bedingung (1) $0 \leqslant f(a) \leqslant c(a)$ gilt für jeden Bogen $a \in A$. Also gilt für den resultierenden Fluß aus S: $f^+(S) \leqslant \text{cap } K$. Wegen $f^-(S) \geqslant 0$ folgt $\text{val } f = f^+(S) - f^-(S) \leqslant \text{cap } K$.

Definition (29.9):

Ein Bogen a heißt ein

F-NULLBOGEN , wenn $f(a) = 0$.

F-POSITIVER Bogen, wenn $f(a) > 0$.

F-GESÄTTIGT , wenn $f(a) = c(a)$.

F-UNGESÄTTIGT , wenn $f(a) < c(a)$.

Satz (29.10):

Für einen Fluß f und einen Schnitt K = (S,C(S)) ist val f = cap K genau dann, wenn jeder Bogen in (S,C(S)) f-gesättigt ist und jeder Bogen in (C(S),S) f-Nullbogen ist.

Beweis:

Es gilt $f^+(S) = \text{cap } K$ genau dann, wenn $f(a) = c(a)$ für jeden Bogen in (s,C(S)) ist. Ferner ist $f^-(S) = 0$ genau dann, wenn jeder Bogen in (C(S),S) f-Nullbogen ist.

Definition (29.11):

Ein Schnitt K in N heißt ein MINIMUMSCHNITT, wenn es keinen Schnitt K^* gibt mit $\text{cap } K^* < \text{cap } K$.

Satz (29.12):

Sei f ein Fluß und K ein Schnitt. Ist val f = cap K, so ist f ein Maximumfluß und K ein Minimumschnitt.

Beweis:

Sei f^* ein Maximumfluß und K^* ein Minimumschnitt. Nach (29.8) gilt val $f^* \leqslant$ cap K^*. Außerdem gilt val $f \leqslant$ val f^* und cap $K^* \leqslant$ cap K. Wegen val f = cap K folgt val f = val f^* = cap K^* = cap K.

29B : Satz von Ford und Fulkerson

Es stellt sich die Frage, ob auch die Umkehrung von Satz (29.12) gilt. Die Umkehrung wird durch einen konstruktiven Beweis geliefert, der anschließend zu einem Algorithmus ausgebaut wird. Dieser Algorithmus liefert dann zu einem gegebenen Netzwerk einen Maximumfluß.

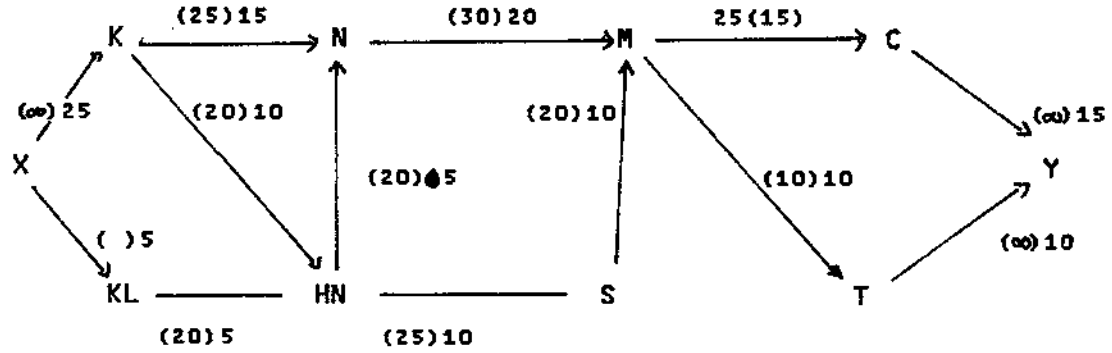
Definition (29.13):

Sei f ein Fluß in einem Netzerk N. Jedem (ungerichteten) Weg P wird eine Zahl $\mu(P)$ zugeordnet. Dabei ist $\mu(P) = \min_{a \in P} \mu(a)$, wobei gilt:

$$\mu(a) = \begin{cases} c(a)-f(a) & \text{, falls a vorwärts gerichtet.} \\ f(a) & \text{, falls a rückwärts gerichtet.} \end{cases}$$

P heißt F-GESÄTTIGT, falls $\mu(P) = 0$. P heißt P-UNGESÄTTIGT, falls $\mu(P) > 0$.
P heißt F-AUGMENTIERT, falls P ein f-ungesättigter Weg von der Quelle bis zur Senke ist.

Beispiel:



Es gilt $\mu(P) = \min \{ \infty, 15, 10, 5, 10, 10, \infty \} = 5$.

P ist f-ungesättigt und auch f-augmentiert.

Bemerkung (29.14):

Aus der Existenz eines f-augmentierten Weges folgt sofort, daß der Fluß f nicht maximal ist.

Begründung:
Man kann folgendermaßen einen neuen Fluß definieren:

$$f^*(a) = \begin{cases} f(a) + \mu(P) & \text{, falls a vorwärts gerichtet in P.} \\ f(a) - \mu(P) & \text{, falls a rückwärts gerichtet in P.} \\ f(a) & \text{, sonst.} \end{cases}$$

Insgesamt ergibt sich $\text{val } f^* = \text{val } f + \mu(P)$.

Satz (29.15):

Ein Fluß f in N ist genau dann ein Maximumfluß, wenn N keinen f-augmentierten Weg besitzt.

Beweis:
'=>':
Wenn N einen f-augmentierten Weg besitzt, dann ist f kein Maximumfluß. Nach Bemerkung (29.13) existiert nämlich ein Fluß mit größerem Wert.

'<=':
Besitzt N keinen f-augmentierten Weg, dann ist zu zeigen, daß f ein Maximumfluß ist. Dazu betrachtet man die Menge S aller Ecken, mit denen x durch f-ungesättigte Wege verbunden ist. Es ist $x \in S$ und $y \in C(S)$, da es keinen f-augmentierten Weg nach y gibt. Daher ist K =

(S,C(S)) ein Schnitt in N.
Wir zeigen: Jeder Bogen von (S,C(S)) ist f-gesättigt.
Sei also a = (u,v) ein Bogen mit u ∈ S und v ∈ C(S). Da u ∈ S ist, gibt es einen f-ungesättigten Weg von x nach u. Wäre (u,v) ebenfalls f-ungesättigt, so könnte man den Weg verlängern und hätte v ∈ S, Widerspruch. Ähnlich zeigt man, daß jeder Bogen von (C(S),S) f-Null sein muß.

Nach Satz (29.10) ergibt sich nun, daß val f = cap K ist und daraus nach (29.12), daß f ein Maximumfluß ist.

Aus dem 2. Beweisteil ('<=') ergibt sich folgender Satz.

Satz (29.16) [Satz von Ford und Fulkerson]:

In jedem Netzwerk ist der Wert eines Maximumflusses gleich der Kapazität eines Minimumschnittes.

Der Satz wird auch als 'min-cut-max-flow-Theorem' bezeichnet.

29C : Algorithmus von Ford und Fulkerson

Im folgenden wird schrittweise ein Algorithmus entwickelt, der einen Maximumfluß findet.

Grundzüge des Algorithmus:
1. Man beginnt mit einem bekannten Fluß (evtl. Nullfluß).

2. Nach jeder Konstruktion eines neuen Flusses f sucht man mit einem Unterprogramm nach einem f-augmentierten Weg.

3. Existiert ein f-augmentierter Weg P, dann wird f^* mit Hilfe von P konstruiert und man wiederholt 2).
   Existiert kein f-augmentierter Weg, dann STOP.

Die Schwierigkeit besteht im wesentlichen nur darin, das Unterprogramm zu entwickeln. Auch dies werden wir nur in groben Umrissen tun und dabei folgende Begriffe einführen.

Definition (29.17):

Ein Baum T in N heißt f-UNGESÄTTIGTER BAUM, wenn

i) $x \in V(T)$
ii) für jede Ecke v von T ist der (x,v)-Weg f-ungesättigt.

In jedem Stadium gibt es 2 Möglichkeiten, durch die der Baum weiter wachsen kann:

1) Es gibt einen f-ungesättigten Bogen a in (S,C(S)) mit $S = V(T)$. Dann wird der Bogen und sein Endpunkt hinzugefügt.

2) Es gibt einen f-positiven Bogen in (C(S),S). Dann wird der Bogen mit seinem Anfangspunkt hinzugefügt.

Falls T die Senke y erreicht, sprechen wir von einem DURCHBRUCH. Im Falle eines Durchbruchs ist der (eindeutig bestimmte) (x,y)-Weg von T ein f-augmentierter Weg.

Falls T zu wachsen aufhört, bevor T die Senke y erreicht, folgt aus Satz (29.15), daß f ein Maximumfluß ist.

Diese Überlegungen führen zu folgender Prozedur:

(29.18): Markierungsprozedur

Beim Wachsen des Baumes T wird jede Ecke v von T mit einer Marke versehen: $l(v) = \mu(P_v)$, wobei P_v der (eindeutig bestimmte) (x,y)-Weg in T ist. Wir beginnen bei der Quelle x und definieren $l(x) = \infty$.

1) Ist a ein f-ungesättigter Bogen, dessen Anfangsecke u bereits ~~nicht markiert ist~~, dann setzen wir $l(v) = \min \{ l(u), c(a) - f(a) \}$.

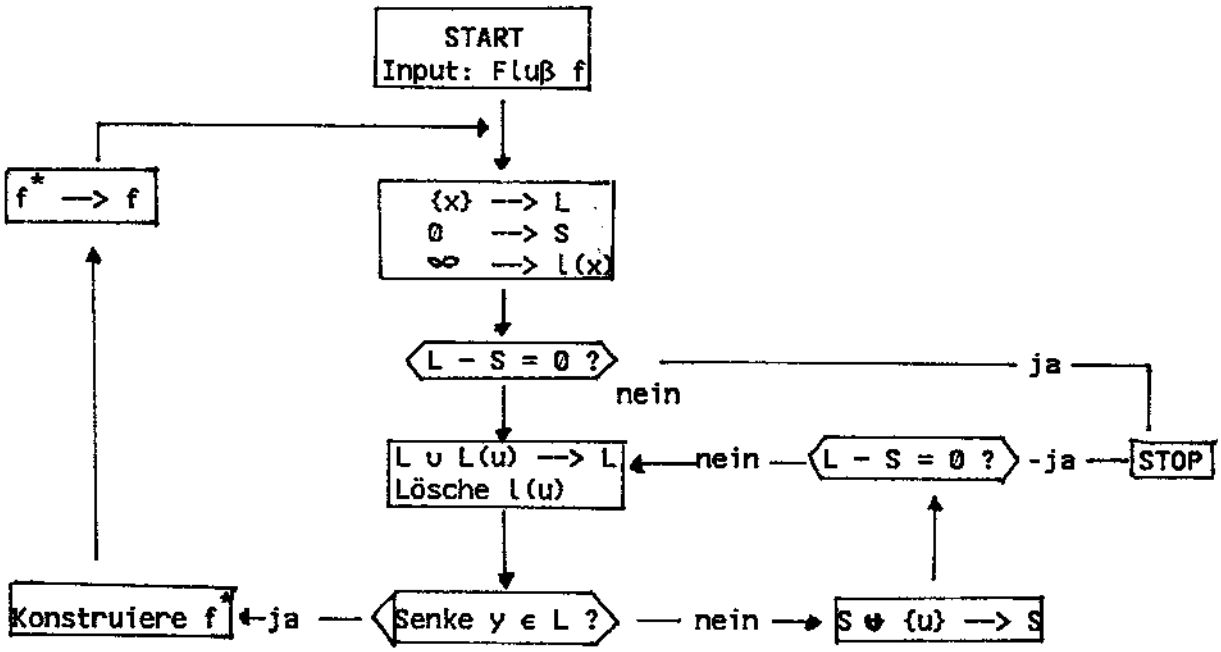
2) Ist a ein f-positiver Bogen, dessen Endecke u bereits eine Marke l(u) trägt, aber dessen Anfangsecke v nicht markiert ~~nicht~~ ist, dann setzen wir $l(v) = \min \{ l(u), f(a) \}$.

3) Man löscht die Marke von u, falls alle Ecken, die von u aus markiert werden können, eine Marke bekommen haben.

Mit den Bezeichnungen

L : Menge der markierten Ecken
S : Menge der Ecken mit ausgelöschten Marken
L(u): Menge der von u aus markierten Ecken

ergibt sich folgendes Flußdiagramm (29.19):



Komplexität (29.20):
In der vorliegenden Form ist dieser Algorithmus nicht von polynomialer Komplexität. In der Literatur gibt es aber Varianten dieses Algorithmus von polynomialer Komplexität.

Kapitel 30 : Planare Graphen

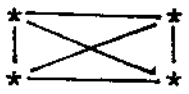
30A : Grundbegriffe

Vorbemerkung:
Unter 'Ebene' werden wir im folgenden unsere Anschauungsebene E verstehen, die auch durch $\mathbb{R}$-Vektorraum $\mathbb{R} \times \mathbb{R}$ koordinatisiert werden kann.
Unter einem EBENEN GRAPHEN G verstehen wir einen in der Ebene eingezeichneten Graphen, bei dem sich die Kanten nur in den Ecken schneiden.

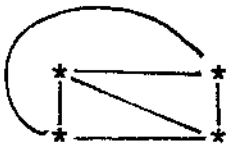
Definition (30.1):

Ein Graph heißt PLANAR, wenn G isomorph zu einem ebenen Graphen ist. Die Darstellung von G als ebenem Graphen nennt man auch PLANARE EINBETTUNG.

Beispiel:



K_4



Planare Einbettung des K_4

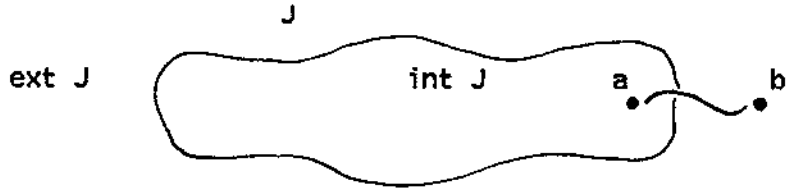
Wir benutzen ohne Beweis folgendes Ergebnis aus der Analysis:

Eine JORDANKURVE J ist eine stetige, sich selbst nicht schneidende Kurve, bei der Anfangs- und Endpunkt identisch sind.

Die Jordankurve J zerlegt E in 2 disjunkte offene Teilmengen, die mit int J und ext J bezeichnet werden. Der (topologische) Abschluß dieser Mengen wird mit Int J und Ext J bezeichnet. Dabei gelten:

$$\text{int } J \cap \text{ext } J = \emptyset$$
$$\text{Int } J \cap \text{Ext } J = J$$

Beispiel:
$E = \mathbb{R} \times \mathbb{R}$



Satz (30.2): [Jordanscher Kurvensatz]

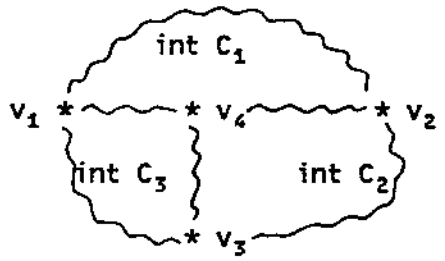
Jede Kurve, die einen Punkt a im Inneren (int J) mit einem Punkt b im Äußeren (ext J) verbindet, schneidet die Jordankurve J.

Dieser Satz gilt von der Anschauung her ganz offensichtlich, aber ein formaler Beweis ist nicht leicht.

Satz (30.3):

Der vollständige Graph K_5 ist nicht planar.

Beweis:
Angenommen, es gäbe eine planare Einbettung G mit den Ecken v_1, ..,v_5. Der Zykel $C = v_1v_2v_3v_1$ ist eine Jordankurve in der Ebene. Die Ecke v_4 muß entweder im Inneren int C oder im Äußeren ext C liegen. Ohne Beschränkung der Allgemeinheit (OBdA) sei v_4 in int C. Die Kanten v_1v_4, v_2v_4 und v_3v_4 teilen int C in 3 Gebiete ein:



Nun muß v_5 in einem der 4 Gebiete liegen. Liegt v_5 in ext C, dann schneidet die Kante v_4v_5 die Jordankurve C.
Auch für die anderen 3 Fälle ergibt sich dieser Widerspruch.

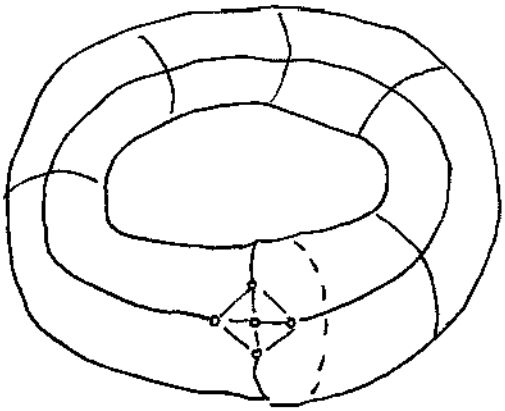
Bemerkung (30.4):
Der vollständige bipartite Graph $K_{3,3}$ ist nicht planar. Der Beweis kann ähnlich leicht geführt werden.

Definition (30.5):

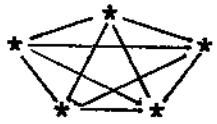
Ein Graph G heißt EINBETTBAR in eine Fläche S, wenn G isomorph ist zu einem Graphen K auf S, dessen Kanten sich nur in den Endecken schneiden.

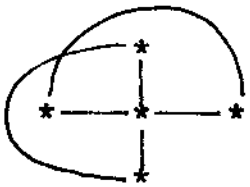
Die folgenden Zeichnungen zeigen, daß sich der K_5 auf dem Torus und der $K_{3,3}$ auf dem Möbiusband einbetten lassen.

Einbettung des K_5 auf die Fläche des Torus:

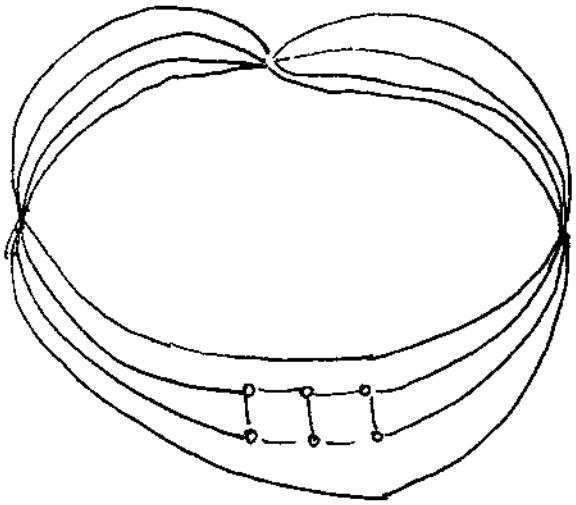


K_5:

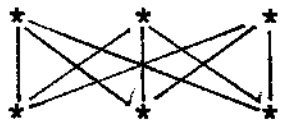


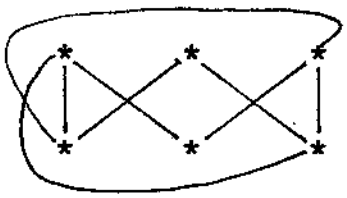


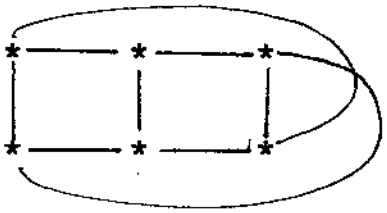
Einbettung des $K_{3,3}$ auf das Möbiusband:



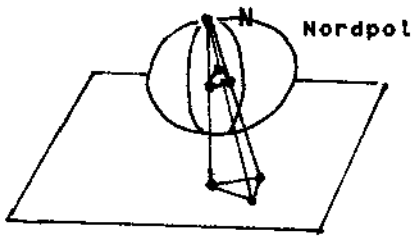
$K_{3,3}$:







Bemerkung (30.6):
Ein planarer Graph G teilt die Ebene in Gebiete ein. Dabei erscheint es so, daß das äußere Gebiet ausgezeichnet sei, weil es sich unbeschränkt ausdehnt. Dies erkennt man mit Hilfe der stereographischen Projektion:



Mit Hilfe der stereographischen Projektion läßt sich jeder planare Graph in die Sphäre (= Kugeloberfläche) einbetten. Umgekehrt läßt sich jeder in die Späre einbettbare Graph mit Hilfe der stereographischen Projektion in die Ebene einbetten.

Um dies einzusehen, bilden wir G also auf die Sphäre ab. Danach drehen wir die Kugel so, daß der Nordpol (Projektionszentrum) in das Gebiet zu liegen kommt, welches wir unbeschränkt ausgedehnt haben wollen, und projizieren den Graphen auf die Ebene.

30 B : Eulersche Formel

Konstruktion des dualen Graphen (30.7):

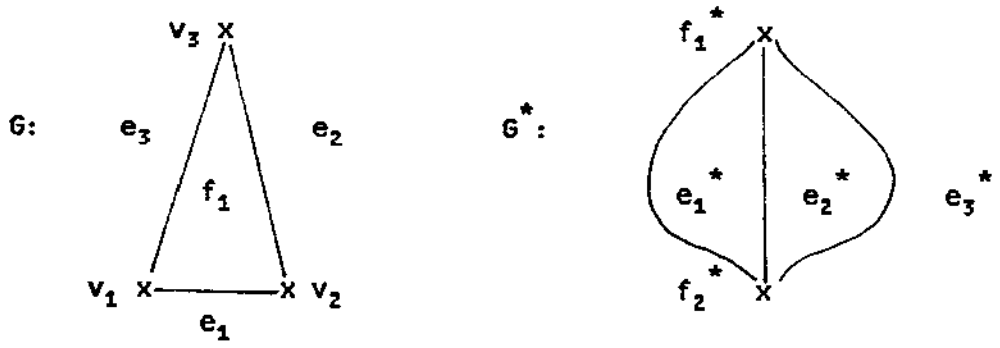
Ist ein planarer Graph G gegeben, so wird der DUALE GRAPH G^* wie folgt konstruiert:

1) Jedem Gebiet f von G entspricht eine Ecke f^* von G^*.

2) Jeder Kante e von G entspricht eine Kante e^* von G^*.

3) 2 Ecken f^* und g^* von G^* sind durch die Kante e^* genau dann verbunden, wenn die Gebiete f und g von G durch die Kante e getrennt werden.

Beispiel:



Folgerungen:

1) $\nu(G^*) = \varphi(G)$, wobei $\varphi(G)$ die Anzahl der Gebiete von G ist.

2) $\epsilon(G^*) = \epsilon(G)$, d. h. gleiche Kantenanzahl.

3) $d_{G^*}(f^*) = d_G(f)$ für alle Gebiete f von G, wobei $d_G(f)$ die Anzahl der Kanten ist, mit denen f inzident ist.

Satz (30.8):

Ist G ein planarer Graph, dann gilt $\Sigma_{f \in F} d(f) = 2\epsilon$, wobei F die Menge der Gebiete von G bezeichnet.

Beweis:

Sei G^* der duale Graph von G. Unter Verwendung von Satz (19.19) gilt dann:

$$\Sigma_{f \in F} d(f) = \Sigma_{f^* \in V(G^*)} d(f^*) = 2\epsilon(G^*) = 2\epsilon.$$

Satz (30.9): [Eulersche Formel]

Ist G ein zusammenhängender, planarer Graph, dann gilt $\nu - \epsilon + \varphi = 2$, wobei φ die Anzahl der Gebiete ist.

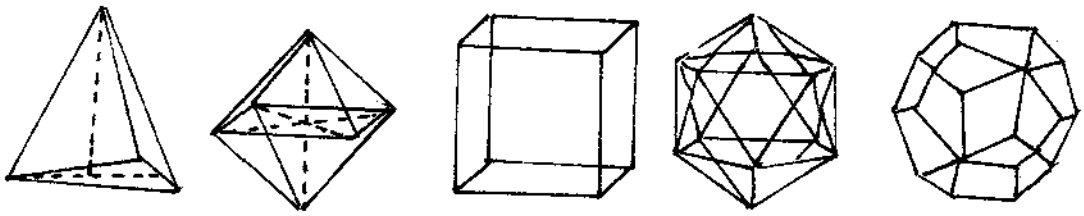
Beweis:
Induktionsanfang:
$\varphi = 1$. Dann ist G ein Baum und nach (21.3) gilt $\epsilon = \nu - 1$.

Induktionsvoraussetzung: Für alle $\varphi \leq n$, $n \geq 2$, gelte die Behauptung des Satzes.

Induktionsschritt:
Sei G ein Graph mit n Gebieten und sei e eine Kante, die keine Schnittkante ist. Dann ist G - e ein zusammenhängender Graph mit n-1 Gebieten und nach Induktionsvoraussetzung gilt dann $\nu(G-e)-\epsilon(G-e)+\varphi(G-e) = 2$.
Benutzt man $\nu(G-e) = \nu$, $\epsilon(G-e) = \epsilon-1$ und $\varphi(G-e) = \varphi-1$, so erhält man $\nu - (\epsilon-1) + (\varphi-1) = 2$, woraus die Induktionsbehauptung folgt.

Bemerkung (30.10):
Obige Formel hängt unmittelbar mit Eulers Polyederformel $\nu-\epsilon+\varphi = 2$ zusammen. Dabei bedeuten ν die Eckenanzahl, ϵ die Kantenanzahl und φ die Seitenflächenanzahl eines Polyeders.

Beispiel:
Die platonischen Körper (siehe auch ~~Beispiel 22.2~~ nach Definition 22.1):



Man kann die Formel für jeden obigen Polyeder nachprüfen, ebenso leicht kann man aber folgende Überlegung anstellen:

Projiziert man eine Polyeder vom Mittelpunkt seiner umschreibenden Kugel auf die Kugeloberfläche, so erhält man einen Graphen, der offenkundig planar ist; dann kann man Satz (30.9) anwenden.

Korollar (30.11):

Ist G ein schlichter, planarer Graph mit mindestens 3 Ecken, dann ist $\epsilon \leqslant 3\nu-6$.

Beweis:
Es genügt, die Behauptung für zusammenhängende Graphen zu zeigen. Wegen $\nu \geqslant 3$, und weil G schlicht ist, folgt, daß jedes Gebiet mit mindestens 3 Kanten inzident ist. Also haben wir $d(f) \geqslant 3$ und deswegen

$$\Sigma_{f\in F} d(f) \geqslant 3\varphi.$$

Nach (30.8) folgt $2\epsilon \geqslant 3\varphi$ und nach (30.9) $\nu-\epsilon+(2\epsilon)/3 \geqslant 2$, woraus sich die Behauptung ergibt.

Korollar (30.12):

Ist G schlicht und planar, so gilt für den kleinsten Eckengrad δ von G, daß $\delta \leqslant 5$.

Beweis:
Für $\nu < 3$ gilt die Behauptung trivialerweise. Für $\nu \geqslant 3$ gilt, daß $\delta * \nu \leqslant \Sigma_{v \in V} d(v) = 2\epsilon$ und $2\epsilon \leqslant 6\nu - 12$ nach (30.11). Also haben wir $\delta \leqslant 6 - 12/\nu$, woraus die Behauptung folgt.

Satz (30.13):

Der vollständige Graph K_5 und der vollständige bipartite Graph $K_{3,3}$ sind nicht planar.

Beweis:
Wäre K_5 planar, so würde nach (30.11) gelten:

$$10 = \epsilon(K_5) \leqslant 3\nu(K_5) - 6 = 9,$$

ein Widerspruch.
Da der $K_{3,3}$ keinen Zykel kleinerer Länge als 4 hat, muß jedes Gebiet f den Grad $d(f) \geqslant 4$ haben. Nach (30.8) gilt:

$$4\varphi \leqslant \Sigma_{f \in F} d(f) = 2\epsilon = 18.$$

Daraus ergibt sich $\varphi \leqslant 4$ und nach (30.9) $2 = \nu - \epsilon + \varphi \leqslant 6 - 9 + 4 = 1$, ein Widerspruch. Also ist (30.8) nicht anwendbar, d.h. $K_{3,3}$ nicht planar.

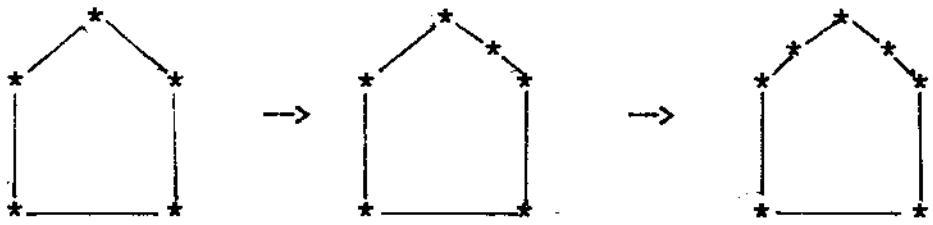
Der Beweis von Satz (30.13) ist unabhängig von (30.3) und (30.4). Die beiden Graphen spielen eine entscheidende Rolle in der Theorie der planaren Graphen, wie der folgende Satz von Kurakowski zeigt. Wir werden den Satz nicht beweisen, aber um ihn zu verstehen, brauchen wir den Begriff der Unterteilung eines Graphen.

Definition (30.14):

Wird in einem Graphen G eine Kante uv ersetzt durch 2 Kanten ua und av, wobei die Ecke a zu G neu hinzugefügt wird, dann heißt der dadurch entstandene Graph G^* eine UNTERTEILUNG von G.

Ferner heißt jeder Graph G^* eine Unterteilung von G, der durch endlich viele Unterteilungen aus G entstanden ist.

Beispiel:
Unterteilungen von C_5:



Satz (30.15): [Satz von Kurakowski]

Ein Graph G ist genau dann planar, wenn G keine Unterteilung des K_5 oder des $K_{3,3}$ enthält.

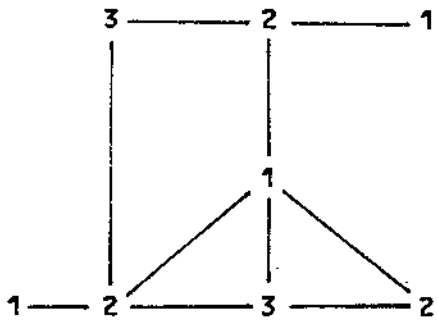
Bemerkung (30.16):
Für planare Einbettungen gibt es einige Anwendungen. So zum Beispiel ist es bei elektonischen Netzwerken interessant, ob sie auf einer Platine gedruckt werden können. Dabei dürfen sich die Leitungen nicht kreuzen.
In der Literatur findet man Algorithmen polynomialer Komplexität, die für einen Graphen G entscheiden, ob G planar ist und dabei auch eine planare Einbettung von G liefern, falls G planar ist.

Kapitel 31 : Eckenfärbungen

31A : Die chromatische Zahl

Problem (31.1) [Vierfarbenproblem]:
Kann man eine Landkarte mit vier Farben so färben, daß benachbarte Länder verschiedene Farben haben?

Beispiel:
Flächenländer der Bundesrepublik Deutschland:



Die erste graphentheoretische Formulierung lautet:
Kann man die Ecken eines planaren Graphen mit vier Farben so färben, daß benachbarte Ecken verschiedene Farben haben?

Das Problem wurde in der Mitte des 19. Jahrhunderts aufgestellt. Im Jahre 1977 konnte gezeigt werden, daß 4 Farben ausreichen.

Definition (31.2):

Eine k-ECKENFÄRBUNG eines Graphen G ist eine Zuordnung

$$f : V(G) \longrightarrow \{ 1, 2, \ldots , k \}$$

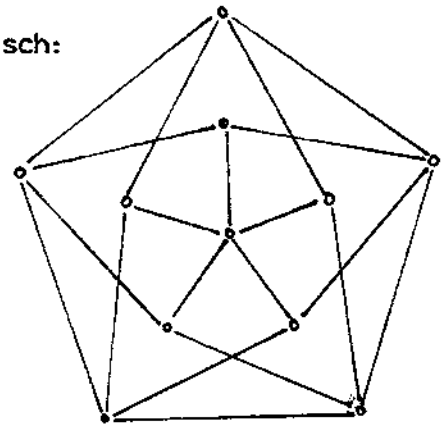
der Ecken von G auf k Farben derart, daß benachbarte Ecken verschiedene Farben haben.
Ein Graph heißt k-FÄRBBAR, wenn G eine k-Eckenfärbung besitzt.
Die CHROMATISCHE ZAHL X(G) ist die kleinste Zahl k, für die G k-färbbar ist. G heißt dann auch k-CHROMATISCH.

Bemerkungen:
G ist genau dann 1-färbbar, wenn die Kantenmenge E(G) leer ist.
G ist genau dann 2-färbbar, wenn G bipartit ist.
Der Graph 'BRD' ist 3-chromatisch (er ist nicht bipartit).

Definition (31.3):

G heißt ein KRITISCHER GRAPH, wenn für alle echten Untergraphen H von G gilt: $X(H) < X(G)$.
G heißt k-KRITISCH, wenn G k-chromatisch und kritisch ist.

Beispiel (31.4):
Der GRÖTZSCHGRAPH ist 4-kritisch:



Sei G k-färbbar. Wir fassen die gleichgefärbten Ecken in Äquivalenzklassen $V_1, \ldots , V_k$ zusammen und sprechen von der K-FÄRBUNG $(V_1, \ldots , V_k)$.

Satz (31.5):

Ist G k-kritisch, dann gilt für den kleinsten Eckengrad δ: $\delta \geq k-1$.

Beweis:
Angenommen, G ist k-kritisch und $\delta < k-1$. Sei v eine Ecke vom Eckengrad δ. Dann existiert für G - v eine (k-1)-Färbung $(V_1, \ldots , V_{k-1})$.

Nach unserer Annahme ist v zu weniger als k-1 Ecken benachbart, also ist v nicht benachbart zu allen Ecken einer Menge V_i.
Aber dann haben wir mit $(V_1, \ldots, V_i \cup \{v\}, \ldots, V_{k-1})$ eine (k-1)-Färbung von G, Widerspruch.

Korollar (31.6):

Jeder k-chromatische Graph G hat mindestens k Ecken vom Eckengrad $\geq$ k-1.

Beweis:
Sei G k-chromatisch und sei H ein k-kritischer Untergraph von G. Nach (31.5) hat jede Ecke von H mindestens den Eckengrad k-1. Also hat H mindestens k Ecken v vom Eckengrd $d(v) \geq k-1$.

Korollar (31.7):

Für jeden Graph G mit dem größten Eckengrad D gilt: $\chi(G) \leq D + 1$.

Beweis:
Folgt unmittelbar aus (31.6).

Das Ergebnis des Korollars (31.7), daß jeder Graph G (D+1)-färbbar ist, liefert nur einen groben Anhaltspunkt. Beispielsweise ist jeder bipartite Graph 2-färbbar. Man kann für jedes $n \in \mathbb{N}$ einen Graphen finden, so daß D = n ist.
Die Abschätzung $2 \leq D+1 = n+1$ ist eben äußerst schlecht.

31B : Der Fünffarbensatz

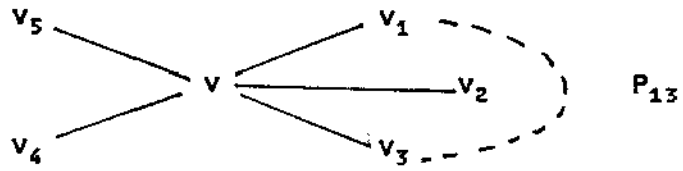
Im Jahre 1890 konnte Heawood zeigen, daß man eine Landkarte mit 5 Farben derart färben kann, daß benachbarte Länder verschiedene Farben haben. Dieses Resultat ist als Fünffarbensatz bekannt. Man beachte, daß man für den K_4 bereits 4 Farben braucht.

Satz (31.8):

Jeder planare Graph ist 5-färbbar.

Beweis:
Angenommen, es gibt einen k-kritischen planaren Graphen G mit $k > 5$. Da G planar und schlicht ist, gilt $d \leq 5$ und nach (32.5) $d \geq k-1$. Daraus folgt, daß G 6-kritisch ist und d = 5.

Sei also v eine Ecke von G vom Eckengrad d(v) = 5. Sei ferner eine 5- Färbung von G-v gegeben durch $(V_1, \ldots, V_5)$. Eine solche 5-Färbung existiert, da G 6-kritisch ist. Da G nicht 5-färbbar ist, muß v zu einer Ecke jeder dieser Farben benachbart sein.



Seien die benachbarten Ecken so angeordnet, wie dies die obige Zeichnung zeigt. Bezeichne G_{ij} den Untergraph $G[V_i \cup V_j]$, den durch $V_i \cup V_j$ induzierten Untergraphen. V_i und V_j müssen in derselben Komponente von G_{ij} liegen. Wäre dies nämlich nicht der Fall, brauchte man nur eine Farbe für $V_i \cup V_j$ und wir bekämen eine 4-Färbung für G-v. Dann könnten wir noch eine Farbe hinzunehmen und hätten eine 5-Färbung für G, Widerspruch.

Für i=3 und j=1 gibt es also einen Weg P_{13} in der Komponente G_{13}. Sei C der Zykel $v\, v_1\, P_{13}\, v_3\, v$. Der Zykel C ist eine Jordankurve in der Ebene, die die Ecken v_2 und v_4 trennt. Nach obiger Überlegung muß es einen Weg P_{24} in G_{24} geben, der v_2 und v_4 verbindet.

Nach dem Jordanschen Kurvensatz folgern wir, daß sich P_{13} und P_{24} schneiden. Sie können sich aber in keiner gemeinsamen Ecke schneiden, weil diese Ecke entweder die Farbe zu V_1V_3 oder zu V_2V_4 haben müßte.

31C : Über Algorithmen für Eckenfärbungen

Problem (31.9):
Es soll ein Stundenplan für alle Vorlesungen an der Universität erstellt werden. Dabei dürfen 2 Vorlesungen nicht zur gleichen Zeit stattfinden, wenn es eine Studentengruppe gibt, die beide Vorlesungen besuchen muß.

Graphentheoretische Version:
Es wird ein Graph konstruiert, dessen Ecken den verschiedenen Vorlesungen entsprechen. Dürfen 2 Vorlesungen nicht zugleich stattfinden, werden die jeweiligen Ecken durch Kanten verbunden. Wenn wir jede Zeit, die für die Vorlesungen zur Verfügung steht, mit einer Farbe versehen, so entspricht eine Eckenfärbung des Graphen einer Lösung des Problems.

Bemerkung:
Die Lösung vieler praktischer Probleme erfordert es, die chromatische Zahl eines Graphen zu finden. Dafür ist kein Algorithmus polynomialer Komplexität bekannt.

Natürlich läßt sich leicht ein polynomialer Algorithmus angeben, der

eine k-Färbung für G herstellt. Dabei wird aber k in aller Regel nicht die chromatische Zahl von G erreichen.

Beispiel (31.10) [Greedy-Algorithmus]:

1) Ordne die Ecken des Graphen: $u_1, u_2, \ldots, u_n$

2) Färbe Ecke u_1 mit der Farbe v_1

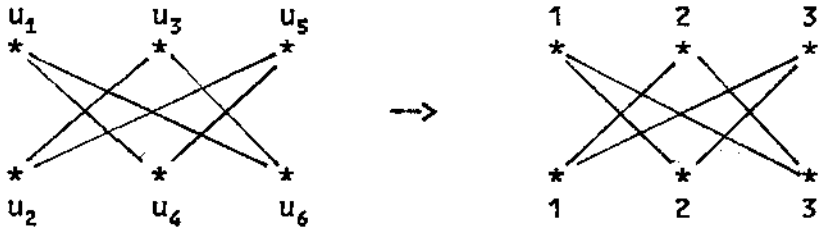
3) Seien die Ecken $u_1, \ldots, u_{i-1}$ schon gefärbt.

   Ist u_i mit keiner Ecke verbunden, dann färbe u_i mit v_1

   Ist u_i mit den Ecken $v_1, \ldots, v_r \in \{u_1, \ldots, u_{i-1}\}$ verbunden, färbe u_i mit der Farbe v_j, falls j die kleinste Zahl ist, so daß v_j bei $v_1, \ldots, v_r$ nicht vorkommt.

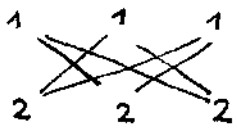
4) STOP, wenn alle Ecken gefärbt sind.

Dieser Algorithmus tritt sehr häufig in verschiedenen Verkleidungen bei graphentheoretischen Problemen auf. Am folgenden Beispiel sieht man, daß man damit nicht immer die chromatische Zahl findet:



(k=3)

Die chromatische Zahl ist aber $\chi(G)=2$. Man färbt wie folgt:



Kapitel 32 : Kantenfärbungen

32A : Kantenchromatische Zahl

In Kapitel 32 betrachten wir nur schlingenlose Graphen.

Definition (32.1):

Eine k-FARBVERTEILUNG für die Kanten eines Graphen ist eine Zuordnung $f : E(G) \longrightarrow \{ 1, 2, \ldots , k \}$ auf k Farben.

Definition (32.2):

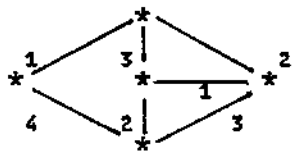
Eine k-KANTENFÄRBUNG eines Graphen G ist eine k-Farbverteilung für die Kanten von G, so daß benachbarte Kanten verschiedene Farben haben. Dabei heißen 2 Kanten benachbart, wenn sie eine Ecke gemeinsam haben.

Die KANTENCHROMATISCHE ZAHL $X^* = X^*(G)$ eines (schlingenlosen) Graphen ist die kleinste Zahl k, für die G k-kantenfärbbar ist.

G heißt k-KANTENCHROMATISCH, wenn $X^*(G) = k$ ist.

Beispiel:

G:



G ist 4-kantenchromatisch, wie man sich leicht überlegt.

Bei jeder Kantenfärbung müssen alle Kanten, die in einer gemeinsamen Ecke inzident sind, verschiedene Farben tragen. Daraus folgt, daß

$$X^* \geqslant D$$

für den größten Eckengrad D ist. Wie das obige Beispiel zeigt, kann auch $X^* > D$ vorkommen.

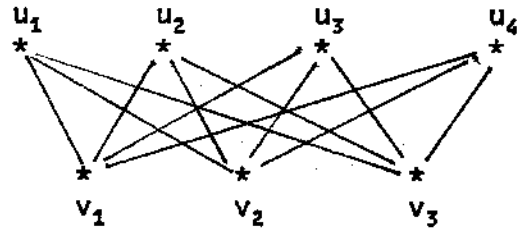
Bemerkung (32.3):
Eine k-Kantenfärbung kann man auch als Partition $(E_1, \ldots , E_k)$ der

Kantenmenge E(G) auffassen. Dabei ist E_i die Menge aller Kanten der Farbe i, i = 1, ... , k. Da die Kanten gleicher Farbe nicht benachbart sind, stellt jede Menge E_i ein Matching von G dar.

Lemma (32.4):

Der vollständige bipartite Graph $K_{m,n}$ mit $m \geq n$ hat die kantenchromatische Zahl $X^*(K_{m,n}) = D$.

Beweis:
Man färbt die Kanten der n Ecken $v_1, \ldots, v_n$ des $K_{m,n}$ folgendermaßen:



Die mit v_1 inzidenten Kanten $v_1u_1, \ldots, v_1u_m$ werden mit den Farben { 1, ... , m } gefärbt. Die mit v_2 inzidenten Kanten $v_2u_1, \ldots, v_2u_m$ werden mit den Farben { 2, 3, ... , m, 1 } gefärbt. Schließlich werden die mit v_n inzidenten Kanten $v_nu_1, \ldots, v_nu_m$ mit { n, n+1, ... , m, 1, ..., n-1 } gefärbt.

Satz (32.5):

Ein bipartiter Graph hat die kantenchromatische Zahl $X^* = D$.

Beweis:
Wir nehmen zunächst an, daß G D-regulär ist. Dann läßt sich ohne weiteres der Algorithmus aus dem Beweis von (32.4) anwenden.
Ist G nicht D-regulär, dann betrachten wir G als Untergraphen eines D-regulären bipartiten Graphen.

Bemerkung:
Für schlichte Graphen zeigt (32.4), daß es eine große Klasse von Graphen mit $X^*(G) = D$ gibt.
Der schlichte vollständige Graph K_3 hat wie jeder ungerade Zykel die kantenchromatische Zahl $X^*(K_3) = D+1$.

Wir verzichten auf den Beweis des folgenden, wichtigen Satzes.

Satz (32.6): [Vizing 1964]

Ist G schlicht, so ist entweder $X^*(G) = D$ oder $X^*(G) = D+1$.

32B : Stundenplanprobleme

Problem:
In einer Schule seien m Lehrer $x_1, \ldots, x_m$ und n Schulklassen y_j gegeben. Jeder Lehrer x_i soll der Klasse y_j p_{ij} Stunden Unterricht erteilen. Man stelle einen Stundenplan auf, so daß eine minimale Anzahl von Stunden auftritt (d.h. der gesamte Unterricht möglichst früh beendet ist).

Mathematisches Modell:
Wir wählen einen bipartiten Graphen G mit der Bipartition (X,Y), wobei $X = \{ x_1, \ldots, x_m \}$ und $Y = \{ y_1, \ldots, y_n \}$. Die Ecken x_i und y_j werden durch p_{ij} Kanten verbunden.

Nun kann ein Lehrer höchstens eine Klasse während einer Schulstunde unterrichten und eine Klasse wird während einer Schulstunde von höchstens einem Lehrer betreut. Unter diesen Voraussetzungen entspricht eine Stunde einem Matching in diesem Graphen.
Die Aufgabe ist es, die Kanten von G in so wenige Matchings wie möglich einzuteilen. Das ist nach (32.3) äquivalent dazu, die Kanten von G mit möglichst wenig Farben zu färben.

Graphentheoretisches Problem:
Sei ein bipartiter Graph G gegeben (auch mit Mehrfachkanten). Wie findet man eine Kantenfärbung mit möglichst wenig Farben?

Wir ergänzen unser Problem noch dadurch, daß wir verlangen, daß jede Klasse nicht mehr als p Stunden haben darf und daß der Unterricht für alle Klassen in maximal p Stunden abgewickelt werden soll; ferner sollen so wenig Klassenzimmer wie möglich gebraucht werden.

Lemma (32.7):

Seien M und N disjunkte Matchings von G mit $|M| > |N|$. Dann gibt es disjunkte Matchings M^*, N^* von G derart, daß $|M^*| = |M| - 1$, $|N^*| = |N| + 1$ und $M^* \cup N^* = M \cup N$.

Beweis:

Betrachte den Graphen $H = G[M \cup N]$. Wie im Beweis zu Satz (28.5) ist jede Komponente von H entweder ein gerader Zykel mit Kanten abwechselnd in M und N oder ein Weg mit Kanten abwechselnd in M und N.
Wegen $|M| > |N|$ gibt es eine Wegkomponente P in H, die mit einer Kante von M beginnt und mit einer Kante von M aufhört. Sei $P = v_0 e_1 v_1 \dots e_{2n+1} v_{2n+1}$ und setze

$$M^* = \{ M - \{e_1, e_3, \dots, e_{2n+1}\} \} \cup \{ e_2, e_4, \dots, e_{2n} \}$$

und

$$N^* = \{ N - \{ e_2, e_4, \dots, e_{2n} \} \cup \{ e_1, e_3, \dots, e_{2n+1} \}.$$

Satz (32.8):

Ist G bipartit und $p \geqslant D$, dann gibt es p disjunkte Matchings $M_1, M_2, \dots M_p$ von G, so daß $E = M_1 \cup M_2 \cup \dots \cup M_p$ ist und ferner $| |M_i| - |M_j| | \leqslant 1$ ist.

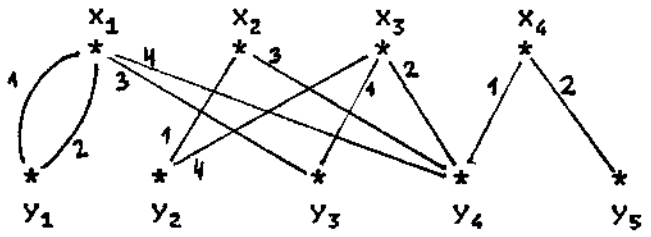
Beweis:

Nach Satz (32.5) hat G die kantenchromatische Zahl $X^*(G) = D$ und nach (32.3) können die Kanten von G in D Matchings $M_1^*, \dots, M_D^*$ eingeteilt werden. Für jedes $p \geqslant D$ gibt es also p Matchings $M_1^*, \dots M_p^*$ mit $M_i^* = \emptyset$ für $D < i \leqslant p$, falls $p > D$.
Durch wiederholte Anwendung von Lemma (32.7) können wir erreichen, daß jedes beliebige Paar M_i, M_j sich um nicht mehr als um eine Kante unterscheidet.

Beispiel:
Es seien 4 Lehrer $x_1, \dots, x_4$ und 5 Klassen $y_1, \dots, y_5$ gegeben. Folgende Matrix P gibt die Lehrverpflichtungen wieder:

	y_1	y_2	y_3	y_4	y_5
x_1	2	0	1	1	0
x_2	0	1	0	1	0
x_3	0	1	1	1	0
x_4	0	0	0	1	1

Der zugehörige bipartite Graph ist folgender:



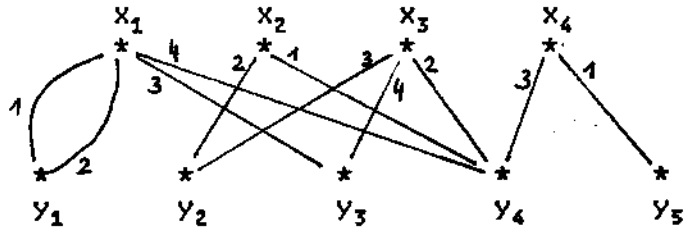
Die Zahlen geben die Kantenfärbung an, die zu folgendem Stundenplan führen:

	1	2	3	4
x_1	y_1	y_2	y_3	y_4
x_2	y_2	–	y_4	–
x_3	y_3	y_4	–	y_2
x_4	y_4	y_5	–	–

Betrachten wir nun das Ergebnis unter dem Gesichtspunkt von Satz (32.8), so sehen wir:

$$
\begin{aligned}
| M_1 | &= | \{ y_1, y_2, y_3, y_4 \} | = 4 \\
| M_2 | &= | \{ y_1, y_4, y_5 \} | = 3 \\
| M_3 | &= | \{ y_3, y_4 \} | = 2 \\
| M_4 | &= | \{ y_2, y_4 \} | = 2
\end{aligned}
$$

Satz (32.8) besagt, daß man Matchings M_i^*, $i = 1, 2, 3, 4$, finden kann, so daß $| \, |M_i^*| - |M_j^*| \, | \leqslant 1$ ist. Wir wollen dies hier durchführen und erhalten:



Daraus ergibt sich ein Stundenplan, bei dem ein Klassenzimmer weniger gebraucht wird:

	1	2	3	4
x_1	y_1	y_1	y_3	y_4
x_2	y_4	y_2	–	–
x_3	–	y_4	y_2	y_3
x_4	y_5	–	y_4	–

Kapitel 33 : Ramseyzahlen

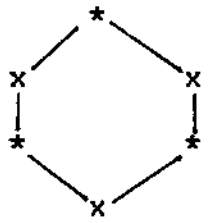
33A : Einführung der Ramseyzahlen

Wir betrachten in diesem Kapitel nur schlichte Graphen.

Definition (33.1):

Eine Untermenge $U \subseteq V(G)$ heißt UNABHÄNGIGE MENGE VON G, wenn keine zwei verschiedenen Ecken in U benachbart sind. Eine unabhängige Menge U heißt MAXIMAL, wenn es keine unbhängige Menge U^* mit $|U^*| > |U|$ gibt.

Beispiel:
$S = \{ v_1, v_2, v_3 \}$ ist maximal unabhängige Menge.

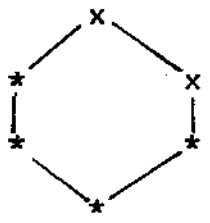


S: mit 'x' markierte Punkte

Definition (33.2):

Eine Untermenge S eines (schlichten) Graphen G heißt CLIQUE, wenn der durch S induzierte Untergraph G(S) vollständig ist.

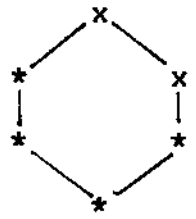
Beispiel:

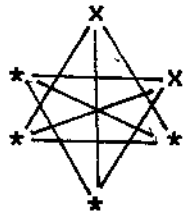


S: mit 'x' markierte Punkte
S ist Clique

Bemerkung:
S ist eine Clique von G genau dann wenn S eine unabhängige Menge des komplementären Graphen C(G) ist. Dabei besteht der KOMPLEMENTÄRE GRAPH C(G) von G aus den Ecken von G, wobei zwei Ecken u, v in C(G) durch eine Kante verbunden sind, wenn u, v in G nicht benachbart sind.

Beispiel:

G : 

C(G) : 

G ist Clique C(G) ist unabhängige Menge

Wenn G keine großen Cliquen (Cliquen mit großer Knotenanzahl) hat, hat G große unabhängige Mengen und umgekehrt. Dieser Sachverhalt wurde von Ramsey (1930) in mehreren Sätzen exakt formuliert. Er zeigte dabei, daß es eine kleinste Zahl r(k,l) gibt, so daß jeder Graph mit r(k,l) Ecken entweder eine Clique mit k Ecken oder eine unabhängige Menge mit l Ecken enthält.
Offensichtlich gelten folgende Beziehungen:

$$r(1,l) = r(k,1) = 1,$$
$$r(2,l) = l,$$
$$r(k,2) = 2.$$

Satz (33.3):

Ist $k \geq 2$ und $l \geq 2$, so gilt $r(k,l) \leq r(k,l-1) + r(k-1,l)$.

Beweis:
Sei G ein Graph mit $r(k,l-1) + r(k-1,l)$ Ecken und sei $v \in V$. Wir treffen eine vollständige Fallunterscheidung:

1) v ist zu keiner Menge S von Ecken mit $|S| \geq r(k,l-1)$ benachbart.
2) v ist zu einer Menge T von Ecken mit $|T| \geq r(k-1,l)$ benachbart.

Die Fallunterscheidung ist vollständig, weil die Mengen S und T die Eigenschaft haben: $|S| + |T| = r(k,l-1) + r(k-1,l) - 1$.
Im Fall 1) enthält der von S induzierte Untergraph $G[S]$ entweder eine Clique von k Ecken oder eine unabhängige Menge von $(l-1)$ Ecken. Also folgt, daß $G[S \cup \{v\}]$ entweder eine Clique von k Ecken oder eine unabhängige Menge von l Ecken enthält (v ist unabhängig von S).

Analog argumentiert man im Fall 2). Aus beiden Fällen zusammen folgt die Behauptung.

Bemerkung (33.4):

Sei $k \geq 2$, $l \geq 2$ und seien $r(k,l-1)$ und $r(k-1,l)$ gerade Zahlen. Dann gilt:

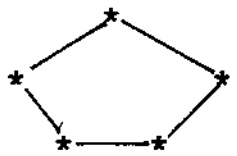
$$r(k,l) < r(k,l-1) + r(k-1,l).$$

Beweis:
Sei G ein Graph mit $r(k,l-1) + r(k-1,l) - 1$ Ecken. Da G eine ungerade Anzahl von Ecken besitzt, folgt nach (19.20), daß es eine Ecke v mit einem geradzahligen Eckengrad d(v) geben muß. Die Ecke v kann nicht mit genau $r(k-1,l) - 1$ Ecken benachbart sein.
Daraus folgt, daß entweder Fall 1) oder Fall 2) gilt, und daraus folgt, daß G entweder eine Clique mit k Ecken oder eine unabhängige Menge mit l Ecken besitzen muß. Also gilt:

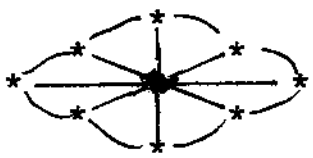
$$r(k,l) < r(k,l-1) + r(k-1,l) - 1.$$

Bestimmung einiger Ramseyzahlen (33.5):
1) $r(3,3) = 6$.
2) $r(3,4) = 9$.

Beweis:
ad 1:
Aus (33.3) folgt $r(3,3) \leq r(3,2) + r(2,3) = 3 + 3 = 6$. Außerdem hat folgender Graph mit 5 Ecken weder eine Clique mit 3 Ecken noch eine unabhängige Menge mit 3 Ecken. Also $5 < r(3,3)$.



ad 2:
Aus (33.3) folgt $r(3,4) \leq r(3,3) + r(2,4) = 6 + 4 = 10$.
Aus (33.4) folgt $r(3,4) < 6 + 4 = 10$. Außerdem hat folgender Graph mit 8 Ecken weder eine Clique von 3 Ecken noch eine unabhängige Menge mit 4 Ecken, also $8 < r(3,4)$.



Obige Betrachtung sollte dem Leser nicht die Vorstellung vermitteln, daß die Bestimmung von Ramseyzahlen eine einfache Aufgabe ist. Im Gegenteil, man kennt nur sehr wenige Ramseyzahlen. Es handelt sich

um ein ungelöstes Problem, die Ramseyzahlen alle zu bestimmen.

Folgende Tabelle zeigt die bisher bekannten Ramseyzahlen:

	l=1	2	3	4	5	6	7
k=1	1	1	1	1	1	1	1
2	1	2	3	4	5	6	7
3	1	3	6	9	14	18	23
4	1	4	9	18			

Satz (33.6):

$$r(k,l) \leq \binom{k+l-2}{k-1}$$

Beweis: Induktion über k+l.
Für $k+l \leq 5$ folgt die Behauptung aus $r(1,l) = r(k,1) = 1$ und $r(2,l)=l$, $r(k,2)=k$.

Induktionsannahme:
Die Behauptung gelte für $5 \leq k+l < m+n$.

Induktionsschritt:
Nach (33.3) gilt:

$$r(m,n) \leq r(m,n-1) + r(m-1,n)$$

$$\leq \binom{m+n-3}{m-1} + \binom{m+n-3}{m-2} = \binom{m+n-2}{m-1}$$

Satz (33.7): [Erdös 1947]

$$r(k,k) \geq 2^{k/2} \quad (k \geq 2).$$

Beweis:

Für k=2 haben wir $r(2,2) = 2 \geq 2^1 = 2$.
Sei $k \geq 3$.
Sei G_n die Menge der schlichten Graphen mit der Eckenmenge $\{v_1, \ldots, v_n\}$ und sei C_n^k die Menge der Graphen in G_n, die eine Clique mit k Ecken haben.

Für die Eckenmenge $\{v_1, \ldots, v_n\}$ sind $n(n-1) = \binom{n}{2}$ Kanten v_iv_j möglich. Jede Teilmenge dieser Kanten bestimmt genau einen Graph in

G_n. Also ist:

$$| G_n | = 2^{\binom{n}{2}} \qquad (*)$$

Für die Eckenmenge $\{v_1, \ldots, v_k\}$ gibt es $\binom{n}{k}$ verschiedene Teilmengen mit k Elementen. Sei eine solche Teilmenge mit k Ecken festgewählt (samt allen Kanten zwischen den k Ecken). Die Anzahl der Möglichkeiten für Kanten ist höchstens $m := \binom{n}{2} - 1/2 \binom{k}{2}$. Es gibt davon also höchstens 2^m Teilmengen und man erhält somit:

$$|C_n^k| \leq \binom{n}{k} 2^m .$$

Zusammen mit (*) ergibt sich (mit $p := -1/2 \binom{k}{2}$)

$$\frac{|C_n^k|}{|G_n|} \leq \binom{n}{k} 2^p < \frac{n^k}{k!} 2^p .$$

Wir nehmen nun an, daß $n < 2^{k/2}$ ist. Dann ergibt sich

$$\frac{|C_n^k|}{|G_n|} < \frac{2^{(k^2/2)}}{k!} 2^p < \frac{2^{(k/2)}}{k!} < \frac{1}{2} .$$

Daraus folgt, daß weniger als die Hälfte aller Graphen in G_n eine Clique mit k Ecken enthalten. Da nun $G_n = \{ G \mid C(G) \in G_n \}$ folgt, daß weniger als die Hälfte aller Graphen in G_n eine unabhängige Menge mit k Ecken haben.
Also gibt es Graphen in G_n, die weder eine Clique mit k Ecken noch eine unabhängige Menge mit k Ecken besitzen. Dieser Wiederspruch gilt für alle $n < 2^{k/2}$. Also gilt $r(k,k) \geq 2^{k/2}$.

Korollar (33.8):

Sei $m = \min \{ k, l \}$, dann ist $r(k,l) \geqslant 2^{m/2}$.

Bemerkung (33.9):
Nun werden wir eine andere Interpretation der Zahl $r(k,l)$ gebrauchen. Danach ist $r(k,l)$ die kleinste Zahl n, so daß jede 2-Kantenfärbung (E_1, E_2) des vollständigen Graphen K_n entweder einen vollständigen Untergraphen mit k Ecken mit der Farbe 1 färbt oder einen vollständigen Untergraphen mit l Ecken mit der Farbe 2 färbt.

In dieser Form kommt man zu einer ganz natürlichen Verallgemeinerung der Ramseyzahlen. Wir definieren $r(k_1, \ldots, k_m)$ als kleinste Zahl n, so daß jede m-Kantenfärbung $(E_1, \ldots, E_m)$ von K_n für ein i mit $1 \leqslant i \leqslant m$ einen vollständigen Untergraphen mit k_i Ecken mit der Farbe i färbt. Damit gelangt man zu folgender Verallgemeinerung der obigen Sätze:

Satz (33.10):

$$r(k_1, \ldots, k_m) \leqslant r(k_1-1, k_2, \ldots, k_m) + \ldots + r(k_1, \ldots, k_m-1) - m + 2.$$

Korollar (33.11):

$$r(k_1+1, \ldots, k_m+1) \leqslant \frac{(k_1 + k_2 + \ldots + k_m)!}{k_1!\, k_2! \ldots k_m!}$$

33B: Satz von Schur

Sei $P_n = (E_1, \ldots, E_n)$ eine Partition der Menge $\{ 1, \ldots, m \}$ von natürlichen Zahlen in n Teilmengen $E_1, \ldots, E_n$ für $m \geqslant n$.

Frage: Ist dann die Bedingung (*) erfüllt ?

(*) Es gibt eine Teilmenge E_i, die 3 Zahlen x, y, z enthält mit $x + y = z$.

Man beobachtet, daß die Partition

$$(E_1,E_2,E_3) = (\{1,4,10,13\},\{2,3,11,12\},\{5,6,7,8,9\})$$

der Menge $\{ 1, \ldots, 13 \}$ die Bedingung (*) nicht erfüllt.
Andererseits kann man zeigen, daß (*) von jeder Partition der Menge $\{1,..14\}$ in drei Teilmengen erfüllt wird.

Schur hat 1916 bewiesen, daß es für jedes $n \in \mathbb{N}$ eine Zahl f_n gibt, so daß jede Partition von $\{ 1, 2, \ldots, f_n \}$ in n Teilmengen die Bedingung (*) erfüllt. Dieser Satz von Schur kann auch aus der Existenz der Ramseyzahlen $r_n = r(k_1, \ldots, k_n)$ mit $k_1 = \ldots = k_n = 3$ hergeleitet werden.

Satz (33.12):

Sei $(S_1, \ldots, S_n)$ eine Partition der Menge $\{ 1, \ldots, r_n \}$ von natürlichen Zahlen. Dann gibt es eine Teilmenge S_i, die drei Zahlen x, y, z enthält mit $x + y = z$.

Beweis:
Betrachte den vollständigen Graphen K_{r_n} mit der Eckenmenge $\{ 1, \ldots, r_n \}$. Färbe die Kanten dieses Graphen in den Farben 1, 2, ... , n unter der Bedingung, daß die Kante u v die Farbe j genau dann trägt, wenn $|u-v| \in S_j$ ist.

Nach Satz (33.5) [Existenz der Ramseyzahlen $r_n(3,\ldots,3)$] existiert ein gleichgefärbtes Dreieck a, b, c einer Farbe i. Nehme für das Dreieck a, b, c an, so daß $a > b > c$ ist und setze $x=a-b$, $y=b-c$ und $z=a-c$. Dann ist $x, y, z \in S_i$ und $x+y = z$.

Es gibt viele interessante und schwierige Anwendungen der Ramseyzahlen in der Kombinatorik. Die meisten überschreiten jedoch den Rahmen dieses Buches.