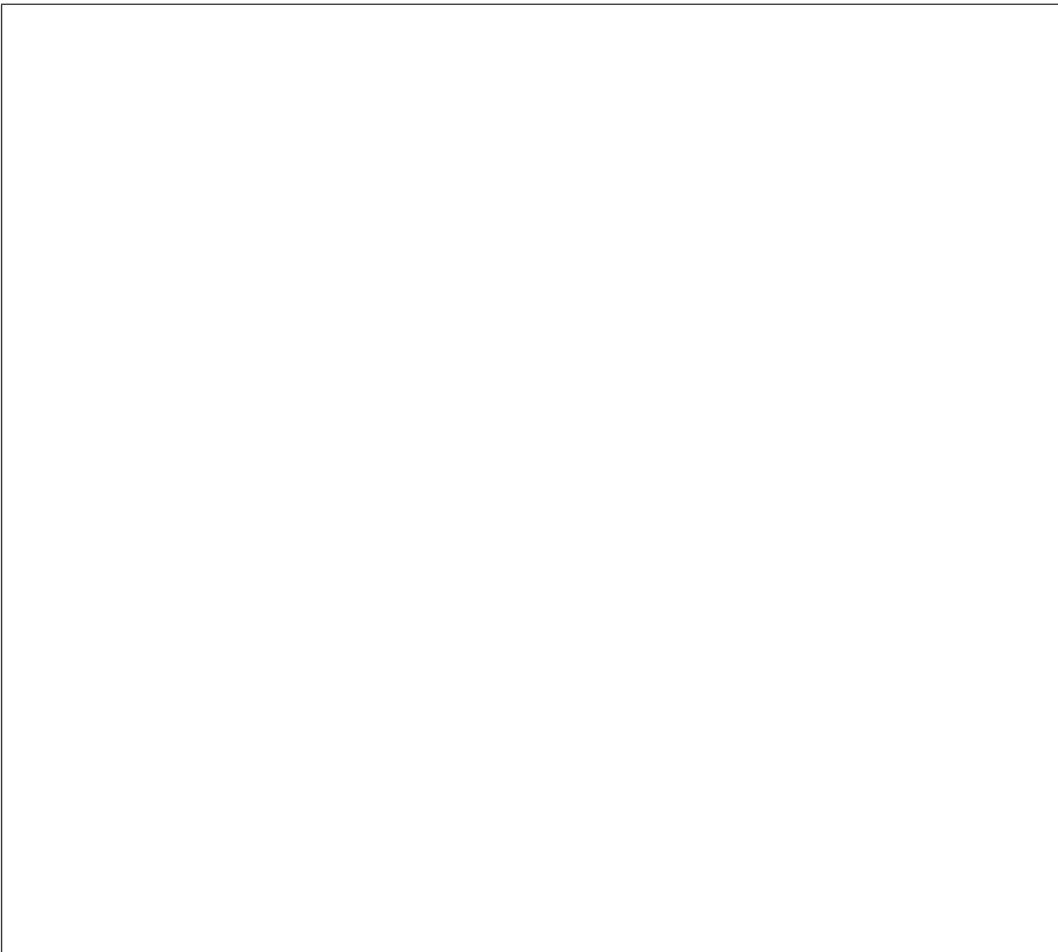


1608: Computersysteme I

Kurseinheit 1

Autor: Jörg Keller



Rückseite

Inhaltsverzeichnis

1	Boole'sche Ausdrücke, Schaltnetze, Zahlendarstellungen	5
1.1	Vorbemerkungen	6
1.1.1	Allgemeines	6
1.1.2	Mengen und Funktionen	6
1.1.3	Gerichtete Graphen	8
1.2	Boole'sche Ausdrücke	11
1.2.1	Vollständig geklammerte Ausdrücke	13
1.2.2	Einsetzungen	15
1.2.3	Identitäten und Ungleichungen	16
1.2.4	Lösen von Gleichungen	20
1.2.5	Der Darstellungssatz	21
1.2.6	Kosten von Ausdrücken	23
1.3	Minimalpolynome	24
1.3.1	Polynome und Primimplikanten	24
1.3.2	Bestimmung von Minimalpolynomen	26
1.4	Schaltkreise	29
1.4.1	Gatter	29
1.4.2	Schaltkreise	30
1.5	Rechnen mit Schaltkreisen	32
1.5.1	Einsetzungen	32
1.5.2	Identitäten und berechnete Funktionen	34
1.5.3	Anfangsschaltkreise	35
1.5.4	Darstellungssatz	36
1.6	Schaltkreiskomplexität	40
1.6.1	Komplexitätsmaße	40
1.6.2	Assoziativität und balancierte Bäume	41
1.6.3	Boole'sche Ausdrücke und korrespondierende Schaltkreise	44
1.7	Darstellungen für ganze Zahlen	45
1.8	Häufig benutzte Schaltkreise	48
1.8.1	Multiplexer und Demultiplexer	49
1.8.2	Decoder und Coder	50
1.9	Arithmetik-Schaltkreise für Ganzzahl-Rechnungen	52
1.9.1	Carry-Chain Addierer	54
1.9.2	Conditional-Sum Addierer	57
1.9.3	Multiplizierer	61
1.10	Darstellungen für rationale Zahlen	65
	Literaturverzeichnis	67

Index	68
Lösungen der Selbsttestaufgaben	71

Kurseinheit 1

Boole'sche Ausdrücke, Schaltnetze, Zahlendarstellungen

Lernziele

Die Lernziele dieser Kurseinheit sind:

- Schaltfunktionen und ihre verschiedenen Darstellungen,
- Eigenschaften von Boole'schen Ausdrücken,
- Definition und Verwendung von Schaltkreisen,
- Grundlegende Zahlendarstellungen,
- Einfache arithmetische Schaltkreise.

1.1 Vorbemerkungen

1.1.1 Allgemeines

Wir wünschen eine erfolgreiche Beschäftigung mit dem Kurs 1608 *Computersysteme I*. Der Kurs findet seine Fortsetzung im Kurs 1609 *Computersysteme II*. Arithmetische Schaltkreise werden in Kurs 1726 *Rechnerarithmetik* ausführlich behandelt. Praktische Aspekte von PC-Systemen werden im Kurs 1744 *PC-Technologie* vertieft.

Wir haben darauf verzichtet, lange Listen Lehrbücher anzugeben. Es gibt viele Lehrbücher zu den im Kurs 1608 *Computersysteme I* behandelten Themen, viele heißen „Technische Informatik“, „Rechneraufbau“ oder ähnlich. Eine Liste wäre aber notwendigerweise unvollständig, ein Online-Katalog erfüllt einen solchen Zweck heutzutage besser. Wir haben lediglich auf entsprechende Bücher der Kursautoren [1, 3] verwiesen. Dies geschieht allerdings nicht, um zum Kauf dieser Bücher anzuhalten, sondern um an Stellen, die wir im Kurs nicht vollständig ausführen, genaue Verweise platzieren zu können. Dies ist am einfachsten mit den Büchern die wir am besten kennen.

1.1.2 Mengen und Funktionen

In diesem Abschnitt befassen wir uns mit Funktionen auf endlichen Mengen. Diese Funktionen werden sich durch Schaltkreise berechnen lassen. Wir setzen ein allgemeines Verständnis von Mengen und Funktionen voraus, und werden hier einige speziellere Sachverhalte wiederholen.

Wir bezeichnen mit

$$\mathbf{N} = \{1, 2, 3, \dots\} \text{ bzw. } \mathbf{N}_0 = \mathbf{N} \cup \{0\}$$

natürliche Zahlen
reelle Zahlen

leere Menge

die *Menge der natürlichen Zahlen* (ohne bzw. mit Null) und mit \mathbf{R} die *Menge der reellen Zahlen*. Mit Z_n bezeichnen wir die Menge der Zahlen von 0 bis $n-1$. Weitere Mengen bezeichnen wir mit Großbuchstaben, z.B. $M = \{1, 2, 5\}$. Die leere Menge bezeichnen wir mit \emptyset .

Wir bezeichnen die Mächtigkeit einer Menge M mit $\#M$. Bei einer endlichen Menge ist $\#M \in \mathbf{N}_0$. Zum Beispiel ist $\#\{1, 2, 5\} = 3$. Zwei Mengen M und N sind gleichmächtig, wenn es eine Bijektion $f : M \rightarrow N$ gibt, d.h. wenn jedem Element von M eineindeutig ein Element aus N zugeordnet werden kann. Während diese Festlegung für endliche Mengen (bei denen man ja lediglich die Anzahlen vergleichen müsste) aufwändig erscheint, ist sie bei unendlichen Mengen angebracht. Eine unendliche Menge, die gleichmächtig wie \mathbf{N} ist, heißt *abzählbar unendlich*, ansonsten *überabzählbar unendlich*. Zum Beispiel ist die Menge der reellen Zahlen überabzählbar unendlich, die Menge M der geraden natürlichen Zahlen ist hingegen abzählbar unendlich, da die Funktion $f : \mathbf{N} \rightarrow M$, $f(x) = 2x$ eine Bijektion zwischen diesen Mengen darstellt.

Das *kartesische Produkt* $M \times N$ zweier Mengen M und N ist definiert als

$$M \times N = \{(a, b) : a \in M, b \in N\} .$$

abzählbar
unendlich
überabzählbar
unendlich
kartesisches
Produkt

Hierbei gilt für endliche Mengen M und N : $\#(M \times N) = \#M \cdot \#N$. Beispielsweise ist

$$Z_2 \times Z_3 = \{0, 1\} \times \{0, 1, 2\} = \{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2)\} .$$

und

$$\#(Z_2 \times Z_3) = 6 = 2 \cdot 3 = \#Z_2 \cdot \#Z_3 .$$

In gleicher Weise kann man das kartesische Produkt aus $n \in \mathbf{N}$ Mengen M_0, \dots, M_{n-1} definieren. Sind alle Mengen $M_i = M$ identisch, dann schreibt man statt $M \times \dots \times M$ auch M^n . Beispielsweise ist

$$\begin{aligned} \{0, 1\}^3 = \{ & (0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), \\ & (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1) \} . \end{aligned}$$

Korreakterweise müsste man das n -fache kartesische Produkt induktiv definieren und hätte dann sehr viele Klammern zu setzen. Wir setzen allerdings nur die äußeren Klammern, d.h. statt

$$a = (\dots (a_0, a_1), a_2) \dots), a_{n-1})$$

schreiben wir

$$a = (a_0, a_1, a_2, \dots, a_{n-1}) .$$

Wir nennen a eine *Folge der Länge n* , in Zeichen $l(a) = n$. Weiterhin vereinbaren wir die Notation

$$A^+ = \bigcup_{i \in \mathbf{N}} A^i .$$

Eine endliche Menge A von Symbolen oder Zeichen nennen wir ein *Alphabet*. Die Folgen über A , d.h. die Elemente aus A^+ nennen wir *Zeichenreihen*.

Die eindeutige Zeichenreihe mit Länge 0 wird *das leere Wort* genannt und häufig mit ϵ abgekürzt. Für ein beliebiges Alphabet A bezeichnet man mit A^0 die Menge, die nur das leere Wort enthält, also

$$A^0 = \{\epsilon\} .$$

Man bezeichnet mit

$$A^* = A^+ \cup \{\epsilon\} = \bigcup_{i \in \mathbf{N}_0} A^i$$

die Menge aller Zeichenreihen mit Zeichen aus A einschließlich des leeren Worts.

Für die Anzahl von Zeichenreihen der Länge höchstens n gilt

$$\# \left(\bigcup_{i=0}^{n-1} A^i \right) = \sum_{i=0}^{n-1} (\#A)^i . \quad (1.1)$$

Um die Summe der rechten Seite zu vereinfachen, benötigen wir das folgende Lemma 1.1.

Folge

Alphabet
Zeichenreihe
leeres Wort

Lemma 1.1 Seien x und n zwei natürliche Zahlen mit $x \neq 1$. Dann gilt

$$\sum_{i=0}^{n-1} x^i = \frac{x^n - 1}{x - 1}.$$

Beweis: Sei $s = \sum_{i=0}^{n-1} x^i$. Es gilt $x \cdot s = \sum_{i=1}^n x^i$. Dann ist $(x - 1) \cdot s = x \cdot s - s = \sum_{i=1}^n x^i - \sum_{i=0}^{n-1} x^i = x^n - 1$. Dividiert man die linke und rechte Seite der Gleichungskette durch $x - 1$, so erhält man die Behauptung. ■

geometrische
Reihe

Den Ausdruck $\sum_{i=0}^{n-1} x^i$ nennt man *geometrische Reihe*.

Eine Folgerung aus Lemma 1.1 ist

$$\sum_{i=m}^{n-1} x^i = \frac{x^n - x^m}{x - 1}. \quad (1.2)$$

wobei m eine natürliche Zahl kleiner als n sein soll. Zum Beispiel ist

$$\sum_{i=0}^{n-1} 2^i = 2^n - 1 \quad \text{und} \quad \sum_{i=m}^{n-1} 2^i = 2^n - 2^m.$$

Selbsttestaufgabe 1.1 Beweisen Sie Gleichung (1.2).

Eine *Funktion* kann nun ebenfalls über ein kartesisches Produkt definiert werden.

Funktion

Definition 1.1 Es seien X und Y Mengen. Eine Funktion f von X nach Y ist eine Teilmenge f von $X \times Y$, für die gilt: für jedes $x \in X$ gibt es genau ein $y \in Y$ mit $(x, y) \in f$. Dieses y heißt der Funktionswert von f an der Stelle x . Die Menge X heißt Definitionsbereich von f , die Menge Y heißt Wertebereich von f .

Funktionswert
Definitionsbereich
Wertebereich

Statt $(x, y) \in f$ schreibt man gewöhnlich $f(x) = y$. Statt „ $f \subseteq X \times Y$ ist eine Funktion“ schreibt man gewöhnlich $f : X \rightarrow Y$.

1.1.3 Gerichtete Graphen

Zusammenhänge zwischen Elementen einer Menge werden in der Informatik häufig durch Graphen dargestellt. Wir werden die wichtigsten Begriffe hier einführen.

gerichteter Graph

Definition 1.2 Ein endlicher gerichteter Graph wird spezifiziert durch ein Paar $G = (V, E)$. Hierbei gilt

- V ist eine endliche Menge. Die Elemente von V heißen die Knoten des Graphen.

- $$E \subseteq V \times V = \{(u, v) \mid u, v \in V\}.$$

Kante,
Vorgänger,
Nachfolger

Ein Element $(u, v) \in E$ heißt eine gerichtete Kante von u nach v . Ist $(u, v) \in E$, so heißt u direkter Vorgänger von v und v heißt direkter Nachfolger von u .

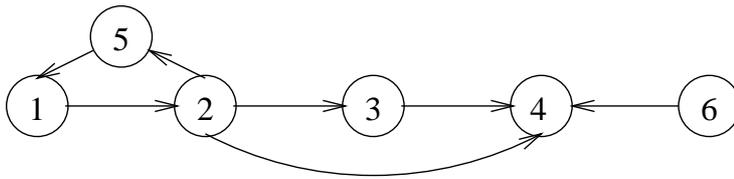


Abbildung 1.1: Beispiel eines Graphen

Da V endlich ist, ist notwendig auch E endlich. Wir betrachten bis auf weiteres weder unendliche Graphen noch ungerichtete Graphen. Wir schreiben deshalb statt „endlicher gerichteter Graph“ meistens einfach „gerichteter Graph“ oder „Graph“. Man zeichnet gerichtete Graphen, indem man die Knoten $v \in V$ als Kreise oder Punkte mit Beschriftung v und gerichtete Kanten (u, v) als Pfeile von u nach v malt.

Definition 1.3 Sei $G = (V, E)$ ein gerichteter Graph. Eine Folge von Kanten

$$e_i = (v_i, w_i) \in E, i = 1, \dots, l,$$

heißt Pfad, falls $v_{i+1} = w_i$ für $i = 1, \dots, l - 1$. Wir nennen l die Länge des Pfades. Gilt $w_l = v_0$, so heißt der Pfad Zyklus. Gerichtete Graphen, in denen es keine Zyklen gibt, heißen zykelfrei.

Pfad
Zyklus

Definition 1.4 Sei $G = (V, E)$ ein gerichteter Graph. Für einen Knoten $v \in V$ ist der Ingrad $\text{indeg}(v)$ und der Outgrad $\text{outdeg}(v)$ definiert als

$$\begin{aligned} \text{indeg}(v) &= \#\{u \mid (u, v) \in E\} \\ \text{outdeg}(v) &= \#\{u \mid (v, u) \in E\}. \end{aligned}$$

Ingrad
Outgrad

Knoten v mit $\text{indeg}(v) = 0$ heißen Quellen des Graphen. Knoten v mit $\text{outdeg}(v) = 0$ heißen Senken des Graphen. Für den gesamten Graphen definiert man

Quelle
Senke

$$\begin{aligned} \text{indeg}(G) &= \max\{\text{indeg}(v) \mid v \in V\} \\ \text{outdeg}(G) &= \max\{\text{outdeg}(v) \mid v \in V\} \end{aligned}$$

Beispiel 1.1 Abbildung 1.1 zeigt einen gerichteten Graphen. Es gibt einen Pfad der Länge 3 und einen Pfad der Länge 2 von Knoten 1 nach Knoten 4, es gibt einen Zyklus der Länge 3 aus den Kanten $(1, 2)$, $(2, 5)$, $(5, 1)$. Der Ingrad von Knoten 1 ist 1, der Outgrad von Knoten 2 ist 3. Knoten 4 ist eine Senke, Knoten 6 eine Quelle.

Definition 1.5 Sei $G = (V, E)$ ein gerichteter Graph und $v \in V$. Die Tiefe $T(v)$ von v wird definiert als die Länge eines längsten Pfades von einer Quelle zu v , falls ein solcher längster Pfad existiert. Andernfalls ist die Tiefe von v nicht definiert.

Tiefe

1.2 Boole'sche Ausdrücke

Sei $n \in \mathbb{N}$. Wir interessieren uns für Schaltungen mit n Eingängen X_1, \dots, X_n und einem Ausgang. An jedem Eingang sowie am Ausgang sollen nur zwei Signale vorkommen können, die wir der Einfachheit halber mit 0 und 1 bezeichnen. Das Signal am Ausgang soll durch die Signale an den Eingängen eindeutig festgelegt sein. Das Ein-/Ausgabeverhalten einer solchen Schaltung lässt sich dann offensichtlich beschreiben durch eine Funktion $f : \{0, 1\}^n \rightarrow \{0, 1\}$, die jeder Kombination von Eingangssignalen das hierdurch festgelegte Ausgangssignal zuordnet. Dies motiviert die folgende Definition 1.7.

Definition 1.7 Sei $n \in \mathbb{N}$. Eine n -stellige Schaltfunktion ist eine Abbildung $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

Schaltfunktion

Lemma 1.2 Es gibt 2^{2^n} n -stellige Schaltfunktionen.

Beweis: Die Mächtigkeit des Definitionsbereichs $Z_2^n = \{0, 1\}^n$ ist $m = 2^n$. An jeder Stelle des Definitionsbereichs kann einer von zwei möglichen Funktionswerten genommen werden. Damit gibt es insgesamt, $2^m = 2^{2^n}$ Möglichkeiten, die Funktion zu definieren. ■

Zum Beispiel ist die Anzahl der 5-stelligen Schaltfunktionen $2^{2^5} = 2^{32} \approx 4$ Milliarden. Zum Schätzen der Größenordnung von Zweierpotenzen siehe Abschnitt 1.7.

Um eine Schaltfunktion jemandem anderen mitteilen zu können, muss man sie auf irgendeine Art und Weise darstellen. Hierzu sind die folgenden Methoden gebräuchlich, wobei die Liste keinen Anspruch auf Vollständigkeit erhebt:

- Wertetabellen,
- Karnaugh-Diagramme,
- Boole'sche Ausdrücke,
- Schaltnetz-Zeichnungen,
- geordnete binäre Entscheidungsbäume (OBDDs).

Eine sehr allgemeine Art der Darstellung ist die *Wertetabelle*. Man schreibt in der linken Spalte alle Elemente des Definitionsbereichs untereinander, und in der rechten Spalte schreibt man neben jedes Element des Definitionsbereichs den zugehörigen Funktionswert. Während Wertetabellen für beliebige Funktionen anwendbar sind, sind die weiteren Darstellungen auf Schaltfunktionen spezialisiert. Ein *Karnaugh-Diagramm* ist anwendbar für $n \leq 4$ Variablen, es ist ein Rechteck mit 2^n Feldern, und einer Markierung der Seiten mit Variablenwerten, so dass jedes Feld eineindeutig einem Wert des Definitionsbereichs zugeordnet werden kann. In jedes Feld schreibt man den zugehörigen Funktionswert. Abbildung 1.3 zeigt ein Beispiel eines Karnaugh-Diagramms für eine 4-stellige Schaltfunktion. Mit Boole'schen Ausdrücken wollen wir uns im Folgenden beschäftigen, so dass wir hier auf eine Erklärung verzichten. Gleiches

Wertetabelle

Karnaugh-Diagramm

	$X_1 = 1$	$X_1 = 1$	$X_1 = 0$	$X_1 = 0$	
$X_2 = 1$	1	1	1	0	$X_4 = 0$
$X_2 = 1$	1	1	1	1	$X_4 = 1$
$X_2 = 0$	1	1	1	0	$X_4 = 1$
$X_2 = 0$	0	1	0	0	$X_4 = 0$
	$X_3 = 0$	$X_3 = 1$	$X_3 = 1$	$X_3 = 0$	

$$f(X_1, X_2, X_3, X_4) = \begin{cases} 1 & \text{falls mindestens zwei der } X_i = 1, \\ 0 & \text{sonst} \end{cases}$$

Abbildung 1.3: Karnaugh-Diagramm für $n = 4$

x_1	x_2	$x_1 \wedge x_2$	$x_1 \vee x_2$	$\sim x_1$
0	0	0	0	1
0	1	0	1	
1	0	0	1	0
1	1	1	1	

Tabelle 1.1: Wertetabellen von Konjunktion, Disjunktion und Negation

gilt für Schaltnetz-Zeichnungen, da wir Schaltnetze (auch Schaltkreise genannt) in Abschnitt 1.4 besprechen werden.

Die letzte Darstellung die wir kurz ansprechen wollen ist eine Darstellung als spezieller Graph, nämlich als *geordneter binärer Entscheidungsbaum* (ordered binary decision diagram, OBDD). Hierbei handelt es sich um einen balancierten binären Baum, bei dem alle inneren Knoten der gleichen Tiefe mit einer Variable, und die je zwei Kanten die einen Knoten verlassen mit 0 und 1 markiert sind. Die Blätter sind mit den Funktionswerten markiert. Möchte man den Funktionswert an der Stelle (a_1, \dots, a_n) herausfinden, so startet man an der Wurzel, und bei jedem inneren Knoten folgt man der linken, mit 0 markierten Kante, falls die Variable X_i , mit der Knoten markiert ist, den Wert $a_i = 0$ hat, sonst folgt man der rechten Kante.

Konjunktion

Disjunktion

Negation

Gatter

Inverter

Spezielle Schaltfunktionen sind die *Konjunktion* $\wedge : \{0, 1\}^2 \rightarrow \{0, 1\}$, die *Disjunktion* $\vee : \{0, 1\}^2 \rightarrow \{0, 1\}$ und die *Negation* $\sim : \{0, 1\} \rightarrow \{0, 1\}$. Ihre Wertetabellen sind in Tabelle 1.1 angegeben.

Offensichtlich gilt $\wedge(X_1, X_2) = X_1 \wedge X_2 = 1 \Leftrightarrow X_1 = 1$ und $X_2 = 1$. Eine Schaltung mit diesem Ein-Ausgabeverhalten heißt deshalb *AND-Gatter*. Weiter gilt $\vee(X_1, X_2) = X_1 \vee X_2 = 1 \Leftrightarrow X_1 = 1$ oder $X_2 = 1$. Eine Schaltung mit diesem Ein-Ausgabeverhalten heißt deshalb *OR-Gatter*. Schließlich gilt $\sim X_1 = 1 \Leftrightarrow X_1 \neq 1$. Eine Schaltung mit diesem Ein-Ausgabeverhalten heißt deshalb *NOT-Gatter* oder auch *Inverter*. Diese Gatter kann man technisch leicht realisieren. Man benutzt sie als Bausteine zur Konstruktion von komplizierteren Schaltungen.

Sei nun $V = \{X_1, \dots, X_n\}$ eine im folgenden feste Menge von Variablen. Gelegentlich werden wir diese Variablen zu einem Vektor $X = (X_1, \dots, X_n)$

zusammenfassen. Wir werden im Rest dieses Abschnitts folgendes tun:

1. Wir definieren mit Hilfe von \wedge, \vee und \sim die Menge der Boole'schen Ausdrücke mit Variablen in V . Die Menge $\{\wedge, \vee, \sim\}$ heißt auch *Operatorensystem*.
2. Wir ordnen jedem solchen Ausdruck e eine Schaltfunktion $\|e\| : \{0, 1\}^n \rightarrow \{0, 1\}$ zu. Die Funktion $\|e\|$ heißt die durch Ausdruck e *berechnete Funktion*.
3. Wir geben Regeln für das Rechnen und Gleichungenlösen mit Boole'schen Ausdrücken an.
4. Wir zeigen, daß es zu jeder n -stelligen Schaltfunktion f einen Boole'schen Ausdruck e gibt mit $f = \|e\|$. Das gewählte Operatorensystem ist also vollständig.

Operatorensystem

Damit haben wir zweierlei erreicht. Zum einen besitzen wir Schaltfunktionen als *Spezifikation* des Ein-/Ausgabeverhaltens von Schaltkreisen. Zum anderen besitzen wir Boole'sche Ausdrücke, die als Beschreibungen oder Realisierungen einer Schaltfunktion dienen können, da sie alle Schaltfunktionen beschreiben können. Da es zu jeder Schaltfunktion mehrere Boole'sche Ausdrücke gibt, wird man je nach Einsatzzweck den einen oder anderen nehmen. Dies entspricht der Vorgehensweise in der Software-Entwicklung, wo man zunächst die Aufruf-Semantik einer Prozedur festlegt, und erst später entscheidet, wie die Prozedur intern realisiert wird, d.h. ob ein langsamer aber speicherplatzsparender Algorithmus verwendet wird oder ein schnellerer Algorithmus, der aber mehr Speicherplatz verbraucht.

1.2.1 Vollständig geklammerte Ausdrücke

Sei

$$A = \{0, 1, \wedge, \vee, \sim, X_1, \dots, X_n, (,)\}.$$

Die Menge B der *vollständig geklammerten Boole'schen Ausdrücke* ist eine Menge von Zeichenreihen in A^* . Sie wird auf folgende Weise induktiv definiert:

Boole'scher
Ausdruck

Definition 1.8 *Es ist $B_1 = \{0, 1, X_1, \dots, X_n\}$. Sei $i \in \mathbf{N}$, und seien $a, b \in B_i$. Dann liegen folgende Zeichenreihen in B_{i+1} :*

1. a
2. $(\sim a)$
3. $(a \vee b)$
4. $(a \wedge b)$

Eine Zeichenreihe $z \in A^+$ liegt genau dann in B , wenn z in einer der Mengen B_i liegt.

Beispiel 1.3 Der Ausdruck $((0 \vee (\sim 1)) \wedge (X_2 \vee (\sim(\sim X_{52}))))$ ist ein vollständig geklammerter Boole'scher Ausdruck, denn es gilt

$$\begin{aligned} 0, 1, X_2, X_{52} &\in B_1, \\ (\sim 1), (\sim X_{52}) &\in B_2, \\ (0 \vee (\sim 1)), (\sim(\sim X_{52})) &\in B_3, \\ (X_2 \vee (\sim(\sim X_{52}))) &\in B_4 \quad \text{und} \\ ((0 \vee (\sim 1)) \wedge (X_2 \vee (\sim(\sim X_{52})))) &\in B_5. \end{aligned}$$

Die obige Definition findet man praktisch in jedem einschlägigen Lehrbuch. Für das praktische Rechnen hat sie jedoch einen entscheidenden Schönheitsfehler: wir rechnen nicht allein mit Boole'schen Ausdrücken, die nur mit den drei Funktionen \wedge , \vee und \sim gebildet sind. Vielmehr definieren wir in vielfältiger Weise neue Funktionen f und bilden dann mit Hilfe dieser Funktionen Ausdrücke wie zum Beispiel

$$X_1 \wedge f(X_2, 0, (X_1 \vee X_3)).$$

Deshalb definieren wir nun die Menge EB der *erweiterten Boole'schen Ausdrücke*. Hierfür sei $F = \{f_1, f_2, \dots\}$ eine abzählbare Menge von Funktionsnamen für Schaltfunktionen. Mit Hilfe der Funktion $s : F \rightarrow \mathbf{N}_0$ ordnen wir jeder Funktion $f \in F$ eine *Stelligkeit* zu, die einfach die Anzahl der Argumente von Funktion f angibt. Das unendliche Alphabet A wird definiert durch

$$A = V \cup F \cup \{0, 1, \wedge, \vee, \sim, (,), , \}.$$

Es enthält insbesondere das Komma. Die Menge EB der vollständig geklammerten *erweiterten Boole'schen Ausdrücke* wird nun induktiv definiert.

Definition 1.9 Es ist $EB_1 = \{0, 1\} \cup V$. Sei $i \in \mathbf{N}$, und seien $e_1, e_2, \dots \in EB_i$. Dann liegen folgende Zeichenreihen in EB_{i+1} .

1. e_1
2. $(\sim e_1)$
3. $(e_1 \wedge e_2)$
4. $(e_1 \vee e_2)$
5. $f(e_1, \dots, e_{s(f)})$ für alle $f \in F$.

Beispiel 1.4 Sei etwa $s(f) = 3$, d.h. f bezeichnet eine 3-stellige Schaltfunktion. Dann ist

- $X_1, X_2, X_3, 0 \in EB_0$
- $(X_1 \vee X_3) \in EB_1$
- $f(X_2, 0, (X_1 \vee X_3)) \in EB_2$
- $(X_1 \wedge f(X_2, 0, (X_1 \vee X_3))) \in EB_3$.

Für den späteren Gebrauch verabreden wir noch die Abkürzung $f(X)$ für $f(X_1, \dots, X_{s(f)})$.

1.2.2 Einsetzungen

Jeder Boole'sche Ausdruck $e \in B$ kann als Vorschrift zur Berechnung einer Funktion $\|e\| : \{0, 1\}^n \rightarrow \{0, 1\}$ aufgefaßt werden: für $a = (a_1, \dots, a_n) \in \{0, 1\}^n$ berechnet man den Wert der Funktion $\|e\|$ an der Stelle a , indem man für alle i die Konstante a_i für die Variable X_i einsetzt und dann auf die übliche Art auswertet. Die folgenden Definitionen formalisieren diese Vorgehensweise. Das mag manchem Leser überflüssig erscheinen, aber ohne präzise Definitionen kann man eben nichts beweisen.

Definition 1.10 *Eine Einsetzung ist eine Abbildung $\phi : V \rightarrow \{0, 1\}$.*

Einsetzung

Für alle i ist $\phi(X_i) \in \{0, 1\}$ gerade die Konstante, die für die Variable X_i eingesetzt werden soll. Durch eine Einsetzung ϕ ist bereits für *jeden* Boole'schen Ausdruck e der Wert von e an der Stelle $(\phi(X_1), \dots, \phi(X_n))$ festgelegt. Wir nennen diesen Wert $\phi(e)$ und definieren ihn formal, indem wir induktiv die Funktion ϕ von $V \subseteq EB$ auf die ganze Menge EB ausdehnen. Wir definieren $\phi(0) = 0$ und $\phi(1) = 1$. Damit ist ϕ auf EB_0 erklärt.

Definition 1.11 *Seien $e_1, e_2, \dots \in EB$. Wir definieren*

$$\begin{aligned}\phi(\sim e_1) &= \sim\phi(e_1), \\ \phi(e_1 \wedge e_2) &= \phi(e_1) \wedge \phi(e_2), \\ \phi(e_1 \vee e_2) &= \phi(e_1) \vee \phi(e_2), \\ \phi(f_i(e_1, \dots, e_{s(f_i)})) &= f_i(\phi(e_1), \dots, \phi(e_{s(f_i)})) \text{ für alle } i \in \mathbf{N}.\end{aligned}$$

Man beachte, daß wir hier die Bezeichner \wedge , \vee und \sim sowie f_i in zweifacher Weise verwendet haben. Auf der linken Seite stellen sie ein Zeichen in einem Boole'schen Ausdruck dar, auf der rechten Seite sind sie eine Aufforderung zum Auswerten von Funktionen.

Für die Funktionen ' \wedge ', ' \vee ' und ' \sim ' enthält Tabelle 1.1 die Regeln zum Auswerten. Für irgendwelche weiteren Funktionen f_i muß man zuerst Auswertungsvorschriften festlegen, bevor man die obige Definition konkret anwenden kann.

Beispiel 1.5 *Sei ϕ eine Einsetzung mit $\phi(X_1) = 1$, $\phi(X_2) = 0$ und $\phi(X_3) = 1$. Für den Ausdruck $((X_1 \wedge X_2) \vee X_3)$ gilt dann $\phi((X_1 \wedge X_2) \vee X_3) = \phi(X_1 \wedge X_2) \vee \phi(X_3)$. Für den ersten Teilausdruck ergibt sich $\phi(X_1 \wedge X_2) = \phi(X_1) \wedge \phi(X_2) = 1 \wedge 0 = 0$. Damit folgt $\phi((X_1 \wedge X_2) \vee X_3) = 0 \vee 1 = 1$.*

Beispiel 1.6 *Die 3-stellige Schaltfunktion f sei durch die Funktionstabelle 1.2 definiert. Wie im vorigen Beispiel sei ϕ eine Einsetzung mit $\phi(X_1) = 1$, $\phi(X_2) = 0$ und $\phi(X_3) = 1$. Für den Ausdruck $X_1 \wedge f(X_2, 0, (X_1 \vee X_3))$ gilt dann*

$$\phi(X_1 \wedge f(X_2, 0, (X_1 \vee X_3))) = \phi(X_1) \wedge \phi(f(X_2, 0, (X_1 \vee X_3))).$$

Für den zweiten Teilausdruck gilt

$$\phi(f(X_2, 0, (X_1 \vee X_3))) = f(\phi(X_2), \phi(0), \phi(X_1 \vee X_3)).$$

a_1	a_2	a_3	$f(a_1, a_2, a_3)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Tabelle 1.2: Wertetabelle der Funktion f aus Beispiel 1.6

Da $\phi(X_1 \vee X_3) = \phi(X_1) \vee \phi(X_3) = 1 \vee 1 = 1$, gilt für den zweiten Teilausdruck $\phi(f(X_2, 0, (X_1 \vee X_3))) = f(0, 0, 1) = 0$. Damit gilt für den gesamten Ausdruck

$$\phi(X_1 \wedge f(X_2, 0, (X_1 \vee X_3))) = 1 \wedge 0 = 0.$$

Ein subtiler Punkt ist an dieser Stelle die Tatsache, daß durch Definition 1.11 jedem Ausdruck e ein und *nur* ein Wert $\phi(e)$ zugewiesen wird. Hierfür muß man zeigen, daß es zu jedem vollständig geklammerten Ausdruck eine und *nur* eine Zerlegung in Teilausdrücke gibt, auf die man Definition 1.11 anwenden kann. Das ist im Wesentlichen der Inhalt des Zerlegungssatzes, den wir hier nicht ausführen wollen.

1.2.3 Identitäten und Ungleichungen

Ausdrücke kann man nicht nur auswerten, man kann auch mit ihnen rechnen. Dabei verfolgt man meistens eine der zwei folgenden Aktivitäten:

1. man formt Ausdrücke äquivalent um oder
2. man löst Gleichungen.

Definition 1.12 *Es seien $e_1, e_2 \in \text{EB}$ erweiterte Boole'sche Ausdrücke. Es gilt $e_1 \equiv e_2$ genau dann, wenn $\phi(e_1) = \phi(e_2)$ für alle Einsetzungen ϕ gilt.*

Für $e_1 \equiv e_2$ sagt man auch „ e_1 und e_2 sind äquivalent“, und man nennt die Zeichenreihe „ $e_1 \equiv e_2$ “ eine *Identität*.

Beim konkreten Rechnen schreibt man häufig statt ' $e_1 \equiv e_2$ ' einfach ' $e_1 = e_2$ ' und sagt ' $e_1 = e_2$ gilt identisch' oder noch einfacher ' e_1 gleich e_2 '. Dabei mißhandelt man strikt gesprochen das Gleichheitszeichen, dann man setzt Ausdrücke einander gleich, die als Zeichenreihen betrachtet in der Regel *nicht* gleich sind. Wir werden jedoch gelegentlich die Schreibweise ' $e_1 \equiv e_2$ ' verwenden.

Satz 1.3 *Sei $e \in \text{EB}$ ein erweiterter Boole'scher Ausdruck. Dann gibt es genau eine n -stellige Schaltfunktion f so daß $f(X) \equiv e$ gilt.*

Die Funktion f mit $f(X) \equiv e$ heißt die durch Ausdruck e *berechnete Funktion*.

Beweis: Wir definieren zuerst die Funktion f . Für $a = (a_1, \dots, a_n) \in \{0, 1\}^n$ sei $\phi_a : V \rightarrow \{0, 1\}$ die Einsetzung mit $\phi_a(X_i) = a_i$ für alle i . Damit $f(X) \equiv e$ gilt, muß $\phi(f(X)) = \phi(e)$ für alle Einsetzungen ϕ gelten, also insbesondere für $\phi = \phi_a$. Es folgt

$$\phi_a(e) = \phi_a(f(X)) = f(\phi_a(X_1), \dots, \phi_a(X_n)) = f(a) .$$

Also ist f eindeutig bestimmt, und um den Wert der Funktion f an der Stelle a zu berechnen muß man einfach:

- für jede Variable X_i die Konstante a_i einsetzen (ϕ_a bilden) und dann

- auswerten ($\phi_a(e)$ bilden).

Sei nun $\phi : V \rightarrow \{0, 1\}$ eine beliebige Einsetzung. Dann ist für $a = (\phi(X_1), \dots, \phi(X_n))$ auch $\phi = \phi_a$. Es folgt

$$\phi(e) = \phi_a(e) = f(a) = \phi_a(f(X)) = \phi(f(X)) ,$$

also gilt $e \equiv f$. ■

Beispiele für Identitäten liefert der folgende

$\phi(e_1)$	$\phi(\sim e_1)$	$\phi((e_1 \vee (\sim e_1)))$
0	1	1
1	0	1

Tabelle 1.3: Beweis von Identität (B6)

Satz 1.4

(B1)	$(X_1 \wedge X_2) \equiv (X_2 \wedge X_1)$ $(X_1 \vee X_2) \equiv (X_2 \vee X_1)$	<i>Kommutativität</i>
(B2)	$((X_1 \vee X_2) \vee X_3) \equiv (X_1 \vee (X_2 \vee X_3))$ $((X_1 \wedge X_2) \wedge X_3) \equiv (X_1 \wedge (X_2 \wedge X_3))$	<i>Assoziativität</i>
(B3)	$(X_1 \wedge (X_2 \vee X_3)) \equiv ((X_1 \wedge X_2) \vee (X_1 \wedge X_3))$ $(X_1 \vee (X_2 \wedge X_3)) \equiv ((X_1 \vee X_2) \wedge (X_1 \vee X_3))$	<i>Distributivität</i>
(B4)	$(X_1 \vee (X_1 \wedge X_2)) \equiv X_1$ $(X_1 \wedge (X_1 \vee X_2)) \equiv X_1$	
(B5)	$(X_1 \vee (X_2 \wedge (\sim X_2))) \equiv X_1$ $(X_1 \wedge (X_2 \vee (\sim X_2))) \equiv X_1$	
(B6)	$(X_1 \vee (\sim X_1)) \equiv 1$ $(X_1 \wedge (\sim X_1)) \equiv 0$	
(B7)	$(X_1 \vee 1) \equiv 1$ $(X_1 \vee 0) \equiv X_1$ $(X_1 \wedge 1) \equiv X_1$ $(X_1 \wedge 0) \equiv 0$	
(B8)	$(\sim(X_1 \vee X_2)) \equiv ((\sim X_1) \wedge (\sim X_2))$ $(\sim(X_1 \wedge X_2)) \equiv ((\sim X_1) \vee (\sim X_2))$	<i>Morgan-Formeln</i>
(B9)	$(\sim(\sim X_1)) \equiv X_1$	
(B10)	$(X_1 \vee X_1) \equiv X_1$ $(X_1 \wedge X_1) \equiv X_1$	

Man kann Satz 1.4 beweisen, indem man für jede der Identitäten ganz stur die höchstens acht verschiedenen Belegungen der vorkommenden Variablen aufzählt und für jede der Belegungen den Wert beider Seiten der Identitäten auswertet. Das kann man ganz schematisch in Tabellenform tun. Für die erste der Identitäten (B6) haben wir das in Tabelle 1.3 ausgeführt. Mit Hilfe der Identitäten aus Satz 1.4 kann man bis auf die vielen Klammern schon fast in gewohnter Weise rechnen. Mit Hilfe von (B3) kann man beispielsweise rechnen:

$$(X_7 \vee (X_1 \wedge (X_4 \wedge (1 \vee X_2)))) = (X_7 \vee (X_1 \wedge ((X_4 \wedge 1) \vee (X_4 \wedge X_2)))).$$

Hierbei haben wir zwei Dinge getan, nämlich:

1. Wir haben in (B3) die Variablen umbenannt und teilweise durch Konstanten ersetzt und
2. wir haben in einem Ausdruck einen Teilausdruck durch einen äquivalenten Ausdruck ersetzt.

In der Schule wurde beim Rechnen mit arithmetischen Ausdrücken die Regel 'Punktrechnung geht vor Strichrechnung' vereinbart. Der einzige Sinn dieser Regel ist das Sparen von Schreiarbeit, da man Ausdrücke nun nicht mehr vollständig klammern muß. Für Boole'sche Ausdrücke verabreden wir die Regeln

- \sim bindet stärker als \wedge und
- \wedge bindet stärker als \vee .

Insbesondere behandeln wir also ' \vee ' wie '+' (Strichrechnung) und ' \wedge ' wie '·'. (Punktrechnung). Nun können wir in gewohnter Weise Klammern weglassen.

Beispiel 1.7 $X_1 \vee \sim X_2 \wedge X_3 \vee X_4$ ist Abkürzung für $((X_1 \vee ((\sim X_2) \wedge X_3)) \vee X_4)$.

Die *unvollständig geklammerten Ausdrücke*, die durch das Weglassen von Klammern entstehen, sind nichts weiter als Abkürzungen für die ursprünglichen — hoffentlich eindeutig rekonstruierbaren — vollständig geklammerten Ausdrücke. Eine strenge Beschreibung und Rechtfertigung dieses Vorgehens ist mit erheblichem Aufwand verbunden. Der interessierte Leser findet die entsprechenden Konstruktionen und Sätze zum Beispiel in [1, Kap. 1]

unvollständig
geklammerter
Ausdruck

Wir vereinfachen die Schreibweise noch weiter. Ist e ein erweiterter Boole'scher Ausdruck, so schreibt man statt $\sim e$ oft \bar{e} .

Beispiel 1.8 Statt $\sim(X_1 \wedge X_2)$ schreibt man oft $\overline{X_1 \wedge X_2}$.

In arithmetischen Ausdrücken läßt man oft das Multiplikationszeichen '·' weg. Ebenso läßt man in Boole'schen Ausdrücken oft das ' \wedge ' weg.

Beispiel 1.9 Statt $X_1 \wedge X_2 \wedge \overline{X_3}$ schreibt man oft $X_1 X_2 \overline{X_3}$.

Aus den Morgan-Formeln von Satz 1.4 kann man durch Induktion direkt die *allgemeinen Morgan-Formeln*

allgemeine
Morgan-Formeln

$$\begin{aligned} \overline{X_1 \vee \cdots \vee X_n} &\equiv \overline{X_1} \wedge \cdots \wedge \overline{X_n} \\ \overline{X_1 \wedge \cdots \wedge X_n} &\equiv \overline{X_1} \vee \cdots \vee \overline{X_n} \end{aligned} \quad (1.3)$$

herleiten. Außerdem folgen aus Regeln (B3) und (B6) von Satz 1.4 direkt die sogenannten *Resolutionsregeln*

$$\begin{aligned} X_1 X_3 \vee X_2 \overline{X_3} &\equiv X_1 X_3 \vee X_2 \overline{X_3} \vee X_1 X_2 \\ (X_1 \vee X_3)(X_2 \vee \overline{X_3}) &\equiv (X_1 \vee X_3)(X_2 \vee \overline{X_3})(X_1 \vee X_2) \end{aligned} \quad (1.4)$$

In Analogie zur Summennotation von arithmetischen Ausdrücken verabreden wir für erweiterte Boole'sche Ausdrücke e_1, \dots, e_m die Schreibweisen

$$\begin{aligned} \bigwedge_{i=1}^m e_i &= e_1 \wedge \cdots \wedge e_m, \\ \bigvee_{i=1}^m e_i &= e_1 \vee \cdots \vee e_m. \end{aligned}$$

Für den Sonderfall, daß man das UND bzw. ODER von einer leeren Menge von Ausdrücken bildet, verabreden wir

$$\bigwedge_{i \in \emptyset} e_i = 1 \quad \text{und} \quad \bigvee_{i \in \emptyset} e_i = 0 .$$

Definition 1.13 *Es seien e_1 und e_2 erweiterte Boole'sche Ausdrücke. Es gilt $e_1 \leq e_2$ genau dann, wenn $\phi(e_1) \leq \phi(e_2)$ für alle Einsetzungen ϕ gilt.*

Aus den Definitionen schließt unmittelbar für Ausdrücke $a, a', b, b' \in \text{EB}$:

1. aus $a \leq b$ und $a' \leq b$ folgt $a \vee a' \leq b$ und
2. aus $a \leq a'$ und $b \leq b'$ folgt $a \vee a' \leq b \vee b'$.

1.2.4 Lösen von Gleichungen

Das Gleichheitszeichen zwischen verschiedenen Ausdrücken e_1 und e_2 kommt außer beim äquivalenten Umformen noch in einem ganz anderen Zusammenhang vor, nämlich beim Lösen von Gleichungen.

Gleichung

Definition 1.14 *Eine Gleichung ist eine Zeichenreihe der Form ' $e_1 = e_2$ ', wobei e_1 und e_2 beliebige Ausdrücke sein dürfen. Man löst eine Gleichung, indem man alle Einsetzungen $\phi : V \rightarrow \{0, 1\}$ bestimmt, so daß $\phi(e_1) = \phi(e_2)$ gilt.*

Beispiel 1.10 *Die Gleichung $X_1 \overline{X_2} \vee \overline{X_1} X_2 = 1$ hat zwei Lösungen, nämlich*

1. $\phi(X_1) = 1, \phi(X_2) = 0$ und
2. $\phi(X_1) = 0, \phi(X_2) = 1$.

Wir leiten einige Regeln zum Lösen von Gleichungen her. Es seien e_1, \dots, e_n vollständig geklammerte Boole'sche Ausdrücke und es sei ϕ eine Einsetzung. Aus Definition 1.11 und Tabelle 1.1 folgt direkt:

$$\begin{aligned} \phi((e_1 \wedge e_2)) = 1 &\Leftrightarrow \phi(e_1) \wedge \phi(e_2) = 1 \\ &\Leftrightarrow \phi(e_1) = 1 \text{ und } \phi(e_2) = 1 . \end{aligned}$$

Durch Induktion über n folgt:

$$\phi((e_1 \wedge \dots \wedge e_n)) = 1 \Leftrightarrow \phi(e_i) = 1 \text{ für alle } i \in \{1, \dots, n\}$$

Dem Leser wird auffallen, daß man eine Menge Schreibarbeit sparen kann, wenn man beim Gleichungslösen die ϕ 's einfach wegfassen läßt. Aus dem Zusammenhang des Gleichungslösens geht dann hervor, daß man statt den Ausdrücken e in Wirklichkeit die Werte $\phi(e)$ meint. Das ist in der Tat gängige Praxis, der wir auch folgen werden. Nur bei ganz seltenen Anlässen muß man sich daran erinnern, daß man diese Vereinfachung vorgenommen hat. Insbesondere hätte man oben ohne Bezugnahme auf ϕ nicht folgern können:

$$(e_1 \wedge e_2) = 1 \Leftrightarrow e_1 = 1 \text{ und } e_2 = 1 .$$

Nach dem gleichen Muster beweist man das folgende Lemma. Es ist in der vereinfachten Form formuliert, aber für den Induktionsanfang der Beweise muß die Vereinfachung rückgängig gemacht werden.

Lemma 1.5 *Seien e_1, \dots, e_n vollständig geklammerte Boole'sche Ausdrücke. Dann gilt:*

1. $e_1 \wedge \dots \wedge e_n = 1 \Leftrightarrow e_i = 1$ für alle $i \in \{1, \dots, n\}$
2. $e_1 \wedge \dots \wedge e_n = 0 \Leftrightarrow e_i = 0$ für (mindestens) ein $i \in \{1, \dots, n\}$
3. $e_1 \vee \dots \vee e_n = 1 \Leftrightarrow e_i = 1$ für ein $i \in \{1, \dots, n\}$
4. $e_1 \vee \dots \vee e_n = 0 \Leftrightarrow e_i = 0$ für alle $i \in \{1, \dots, n\}$
5. $\bar{e}_1 = 1 \Leftrightarrow e_1 = 0$

Damit sind wir die ϕ 's glücklich wieder los, und der Leser sollte hier natürlich die Frage stellen: warum haben wir nicht gleich einfach so weitergerechnet „wie in der Schule“ (was immer das heißt)? Die vielleicht verblüffende Antwort auf diese Frage ist: so geht es schneller! In der Schule lernt man das Rechnen und Gleichungslösen durch Nachahmen, das allein dauert Monate. Nach einigen Jahren, in denen der Kalkül nicht versagt hat, glaubt man dann vielleicht, daß der Kalkül nicht zu unsinnigen Ergebnissen führt. Verglichen mit diesem Zeitaufwand sind ein paar Seiten Mathematik, auf denen man die genauen Rechenregeln *herleitet*, wenig. Überdies hat der Leser hierbei vielleicht erkannt, wie ungeheuer raffiniert und subtil der „gewöhnliche“ Gleichungskalkül ist.

1.2.5 Der Darstellungssatz

Der zentrale Satz dieses Abschnitts läßt sich nun sehr leicht herleiten. Für Variablen $X_i \in V$ und $\epsilon \in \{0, 1\}$ verabreden wir die Schreibweise

$$X_i^\epsilon = \begin{cases} \bar{X}_i & \text{falls } \epsilon = 0 \\ X_i & \text{falls } \epsilon = 1. \end{cases}$$

Offensichtlich gilt

$$X_i^\epsilon = 1 \Leftrightarrow X_i = \epsilon.$$

Boole'sche Ausdrücke der Form X_i^ϵ nennt man *Literale*.

Literal

Definition 1.15 *Für $a = (a_1, \dots, a_n) \in \{0, 1\}^n$ definieren wir die Boole'schen Ausdrücke $m(a)$ und $c(a)$ durch*

$$m(a) = \bigwedge_{i=1}^n X_i^{a_i},$$

$$c(a) = \bigvee_{i=1}^n X_i^{\bar{a}_i}.$$

Der Ausdruck $m(a)$ heißt der zu a gehörige Minterm und $c(a)$ der zu a gehörige Maxterm.

Minterm
Maxterm

Beispiel 1.11 Es ist $m(0, 1, 0) = \overline{X_1}X_2\overline{X_3}$ und $c(0, 1, 0) = X_1 \vee \overline{X_2} \vee X_3$.

Aus Lemma 1.5 folgt

$$m(a) = \bigwedge_{i=1}^n X_i^{a_i} = 1 \Leftrightarrow X_i^{a_i} = 1 \text{ für alle } i \in \{1, \dots, n\}$$

$$\Leftrightarrow X = (X_1, \dots, X_n) = a \quad (1.5)$$

$$c(a) = 0 \Leftrightarrow X = a \quad (1.6)$$

Für n -stellige Schaltfunktionen f heißt die Menge

$$\text{Tr}(f) = \{a \in \{0, 1\}^n \mid f(a) = 1\}$$

Träger der Träger von f . Offenbar ist

$$\text{Tr}(f) = f^{-1}(1) \text{ und } \{0, 1\}^n \setminus \text{Tr}(f) = f^{-1}(0) .$$

Es gilt

Satz 1.6 (Darstellungssatz) Sei $f : \{0, 1\}^n \rightarrow \{0, 1\}$ eine Schaltfunktion. Dann gilt

$$f(X) \equiv \bigvee_{a \in \text{Tr}(f)} m(a)$$

$$f(X) \equiv \bigwedge_{a \notin \text{Tr}(f)} c(a)$$

kanonische
disjunktive
Normalform
kanonische
konjunktive
Normalform

Die erste Darstellung heißt die *kanonische disjunktive Normalform* von f , die zweite Darstellung die *kanonische konjunktive Normalform*. Der Ausdruck „kanonisch“ rührt daher, dass diese Formen jeweils bis auf die Reihenfolge Min- bzw. Maxterme eindeutig sind.

Beispiel 1.12 Sei f die in Tabelle 1.2 definierte Funktion. Dann gilt

$$f(X) \equiv \overline{X_1}X_2\overline{X_3} \vee X_1X_2\overline{X_3}$$

$$\equiv (X_1 \vee X_2 \vee X_3) \wedge (X_1 \vee X_2 \vee \overline{X_3}) \wedge (X_1 \vee \overline{X_2} \vee \overline{X_3})$$

$$\wedge (\overline{X_1} \vee X_2 \vee X_3) \wedge (\overline{X_1} \vee X_2 \vee \overline{X_3}) \wedge (\overline{X_1} \vee \overline{X_2} \vee \overline{X_3}) .$$

Beweis des Darstellungssatzes: Es gilt

$$\bigvee_{a \in \text{Tr}(f)} m(a) = 1 \Leftrightarrow m(b) = 1 \text{ für ein } b \in \text{Tr}(f)$$

$$\Leftrightarrow X = b \text{ für ein } b \in \text{Tr}(f) .$$

Behauptung 1 folgt nun direkt aus Lemma 1.5. Behauptung 2 beweist man ebenso. ■

1.2.6 Kosten von Ausdrücken

Wir suchen im folgenden sehr oft zu einer vorgegebenen Schaltfunktion f möglichst *einfache* Ausdrücke e , die f berechnen. Hierbei messen wir die Kompliziertheit eines Ausdrucks einfach durch die folgende Kostenfunktion.

Definition 1.16 Sei $e \in B$ ein Boole'scher Ausdruck. Die Kosten $L(e)$ von e sind definiert als die Anzahl von Vorkommen der Zeichen \wedge , \vee und \sim in e .

Beispiel 1.13 $L(X_1 \wedge \sim X_2 \wedge X_3) = 3$.

Die obige Definition scheint wörtlich genommen nur sinnvoll zu sein für Ausdrücke e , bei denen wir gewisse vereinfachte Schreibweisen nicht verwenden. Wir erinnern jedoch daran, daß für uns vereinfacht aufgeschriebene Ausdrücke ebenso wie unvollständig geklammerte Ausdrücke bloß Abkürzungen für vollständig geklammerte Ausdrücke aus B sind. Es ist deshalb

$$L(X_1 \overline{X_2} X_3) = L(X_1 \wedge \sim X_2 \wedge X_3) = L((X_1 \wedge ((\sim X_2) \wedge X_3))) = 3.$$

Offenbar ist $L(X_i^\epsilon) \in \{0, 1\}$, d.h. Literale haben stets Kosten 0 oder 1. Sei nun f eine n -stellige Schaltfunktion. Jeder Minterm m der vollständigen disjunktiven Normalform von f besteht aus genau n Literalen und $n - 1$ \wedge -Zeichen. Es folgt $L(m) \leq 2n - 1$. Sei nun p die vollständige disjunktive Normalform von f . Dann besteht p aus genau $\#\text{Tr}(f)$ Mintermen. Für die Anzahl v der \vee -Zeichen in p gilt

$$v = \begin{cases} \#\text{Tr}(f) - 1 & \text{falls } \#\text{Tr}(f) \geq 2 \\ 0 & \text{falls } \#\text{Tr}(f) \leq 1 \end{cases}$$

Wegen $\#\text{Tr}(f) \leq \#\{0, 1\}^n = 2^n$ folgt

$$L(p) \leq n2^{n+1}.$$

Für jede n -stellige Schaltfunktion f gibt es also einen Boole'schen Ausdruck e mit Kosten höchstens $n2^{n+1}$, der f berechnet. Wir wären natürlich gern in der Lage, zu jeder vorgegebenen Schaltfunktion einen *billigsten* Ausdruck mit dieser Eigenschaft sowie seine Kosten zu bestimmen.

Definition 1.17 Für Schaltfunktionen f heißt die Zahl

$$L(f) = \min\{L(e) \mid e \in B, e \equiv f(X)\}$$

die Formelgrößerand (engl. *formula size*) von f .

Aus dem oben Gesagten folgt sofort

Satz 1.7 Für jede n -stellige Schaltfunktion f gilt $L(f) \leq n2^{n+1}$.

Oft möchte man, wenn man das Wachstum von Funktionen beschreibt, von konstanten Faktoren abstrahieren, da man an der „Größenordnung“ des Wachstums interessiert ist. Um den Begriff „größenordnungsmäßig“ formal zu fassen, führen wir Notationen für asymptotisches Wachstum ein.

Definition 1.18 Seien $f, g : \mathbf{N} \rightarrow \mathbf{N}$ Funktionen. Wir sagen f ist asymptotisch durch g beschränkt, in Zeichen $f \leq_a g$, falls es ein $n_0 \in \mathbf{N}$ gibt mit $f(n) \leq g(n)$ für alle $n \geq n_0$. Wir definieren

$$\begin{aligned} O(g) &= \{f \mid \exists k \in \mathbf{N} : f \leq_a k \cdot g\}, \\ \Omega(g) &= \{f \mid \exists k \in \mathbf{N} : g \leq_a k \cdot f\}, \\ \Theta(g) &= O(g) \cap \Omega(g) \quad . \end{aligned}$$

Normalerweise schreibt man $f = O(g)$ statt $f \in O(g)$.

Beispiel 1.14 Es ist $3n^2 - 4n + 5 \in O(n^3)$, $e^n = \Omega(n^{10})$ und $4n^5 = \Theta(n^5)$.

Damit läßt sich Satz 1.7 umformulieren zu
Für jede n -stellige Schaltfunktion f gilt $L(f) = O(n2^n)$.

1.3 Minimalpolynome

1.3.1 Polynome und Primimplikanten

Wir untersuchen im folgenden besonders einfache Mengen von Boole'schen Ausdrücken, nämlich die sogenannten *Boole'schen Polynome* und die *konjunktiven Normalformen*.

Definition 1.19

- | | |
|---|--|
| Literal | • Ein Literal ist ein Ausdruck der Form X_i^ϵ mit $X_i \in V$ und $\epsilon \in \{0, 1\}$. |
| Monom
Konjunktionsterm | • Ein Monom oder Konjunktionsterm ist ein Ausdruck der Form $\bigwedge_{i \in I} L_i$, wobei die L_i Literale sind für alle i in einer endlichen Indexmenge I . |
| Boole'sches
Polynom
disjunktive
Normalform | • Ein (Boole'sches) Polynom oder disjunktive Normalform (DNF) ist ein Ausdruck der Form $\bigvee_{i \in I} M_i$, wobei die M_i Monome sind für alle i in einer endlichen Indexmenge I . |
| Klausel
Disjunktionsterm | • Eine Klausel oder Disjunktionsterm ist ein Ausdruck der Form $\bigvee_{i \in I} L_i$, wobei die L_i Literale sind für alle i in einer endlichen Indexmenge I . |
| konjunktive
Normalform | • Eine konjunktive Normalform (KNF) ist ein Ausdruck der Form $\bigwedge_{i \in I} C_i$, wobei die C_i Klauseln sind für alle i in einer endlichen Indexmenge I . |

Alle Minterme sind Monome, und alle Maxterme sind Klauseln. Jede kanonische disjunktive Normalform ist ein Polynom, und jede kanonische konjunktive Normalform ist eine konjunktive Normalform.

In den obigen Definitionen sind auch leere Indexmengen I erlaubt. Es folgt, daß 0 sowohl ein Polynom und als auch eine Klausel ist, und daß 1 sowohl ein Monom als auch eine disjunktive Normalform ist.

Naturgemäß interessiert man sich zu einer vorgegebenen Schaltfunktion f für *billigste* Polynome p , die f berechnen.

Definition 1.20 Sei f eine Schaltfunktion und p ein Boole'sches Polynom. Dann heißt p ein Minimalpolynom oder kürzeste disjunktive Normalform von f , falls die folgenden beiden Bedingungen gelten:

1. $p \equiv f(X)$, d.h. p berechnet f .
2. $L(p) = \min\{L(q) \mid q \text{ ist Boole'sches Polynom und } q \equiv f(X)\}$, d.h. p ist ein billigstes Polynom mit dieser Eigenschaft.

Wir werden im folgenden zu vorgegebener Funktion f die Monome, die in Minimalpolynomen von f auftreten können, charakterisieren, und wir werden angeben, wie man diese Monome finden kann. Im unmittelbaren Anschluß daran stoßen wir schon auf das mit Abstand berühmteste offene Problem der Informatik.

Definition 1.21 Seien m und m' Monome. Dann heißt m' Teilmonom von m , falls die folgenden beiden Bedingungen gelten:

Teilmonom

1. jedes Literal in m' kommt auch in m vor, oder $m' = 1$;
2. in m kommt mindestens ein Literal vor, das nicht in m' vorkommt.

Beispiel 1.15 Die Monome X_1X_4 , 1 und $X_2\bar{X}_3$ sind Teilmonome von $X_1X_2\bar{X}_3X_4$, die Monome $X_1X_2X_3X_4$ und $X_1X_2\bar{X}_3X_4$ hingegen nicht.

Lemma 1.8 Es sei m' Teilmonom von m . Dann gilt $m \leq m'$.

Beweis: Es sei $m = \bigwedge_{i \in I} L_i$, wobei die L_i Literale sind. Das Monom m' ist Teilmonom von m und hat deshalb die Form $m' = \bigwedge_{i \in J} L_i$ mit $J \subset I$. Aus Lemma 1.5 folgt

$$\begin{aligned} m = 1 &\Leftrightarrow L_i = 1 \text{ für alle } i \in I \\ &\Rightarrow L_i = 1 \text{ für alle } i \in J \\ &\Leftrightarrow m' = 1. \end{aligned}$$

■

Definition 1.22 Sei f eine Schaltfunktion und m ein Monom. Dann heißt m ein Implikant von f falls $m \leq f(X)$ gilt. Ein Implikant von f heißt ein Primimplikant von f falls kein Teilmonom von m Implikant von f ist.

Implikant
Primimplikant

Die konstante Funktion f mit $f(a) = 0$ für alle a hat nur ein Minimalpolynom, nämlich 0 . Für alle anderen Schaltfunktionen f werden die Implikanten, die in Minimalpolynomen von f vorkommen können, charakterisiert durch

Satz 1.9 Es sei f eine Schaltfunktion, und f sei nicht identisch gleich 0 . Es sei p ein Minimalpolynom von f . Dann besteht p nur aus Primimplikanten von f .

Beweis: Ist f identisch gleich 1, so hat f nur ein Minimalpolynom, nämlich 1, und der Satz gilt offensichtlich. Andernfalls gilt

$$f(X) \equiv p = m_1 \vee \dots \vee m_s$$

für ein $s \in \mathbf{N}$ und Monome m_i , $i \in \{1, \dots, s\}$. Jedes der Monome m_i ist ein Implikant von f , da es sonst eine Einsetzung ϕ gäbe mit $\phi(f) = 0$, aber $1 = \phi(m_i) = \phi(p)$.

Wir nehmen nun an, daß mindestens eins der Monome m_i kein Primimplikant von f ist. Ohne Beschränkung der Allgemeinheit können wir $i = 1$ annehmen (sonst numerieren wir die Monome um.) Sei nun m'_1 Teilmonom von m_1 und Implikant von f . Wir bilden das Polynom p' , indem wir in p das Monom m_1 durch das billigere Monom m'_1 ersetzen:

$$p' = m'_1 \vee m_2 \vee \dots \vee m_s .$$

Offensichtlich ist dann $L(p') < L(p)$. Wegen Lemma 1.8 ist $m_1 \leq m'_1$ und deshalb $p \leq p'$. Andererseits gilt $m'_1 \leq f$ und $m_i \leq f$ für alle i , denn sowohl m'_1 als auch alle m_i sind Implikanten von f . Es folgt $p' \leq f \equiv p$. Es folgt $p' \equiv p \equiv f$. Also war p kein Minimalpolynom von f . ■

1.3.2 Bestimmung von Minimalpolynomen

Um ein Minimalpolynom zu bestimmen, gibt es eine Reihe von Verfahren. Bei den meisten bildet man ausgehend von der Wertetabelle oder der kanonischen DNF zunächst alle Primimplikanten. Hierunter ist das bekannteste das Verfahren von Quine und McCluskey. Wir werden hier lediglich exemplarisch zeigen, wie man die Primimplikanten mittels des Karnaugh-Diagramms bestimmt.

Hierzu stellen wir zunächst fest, dass jedes Rechteck in einem Karnaugh-Diagramm, dessen Seitenlängen Zweierpotenzen sind, einem Monom entspricht. Folglich ist jedes dieser Rechtecke, das in einem Karnaugh-Diagramm nur Einsen überdeckt, ein Implikant. Lässt sich kein größeres Monom-Rechteck finden, das das gegenwärtige Rechteck enthält, und nur Einsen überdeckt, dann ist das gegenwärtige Rechteck schon ein Primimplikant.

Beispiel 1.16 *In dem Karnaugh-Diagramm aus Abbildung 1.3 lassen sich sechs Primimplikanten finden: zwei Quadrate mit Seitenlänge 1 in der rechten Spalte und untersten Zeile, und vier Quadrate mit Seitenlänge 2 in dem linken oberen 3×3 -Block aus Einsen.*

Es bleibt das auf den ersten Blick einfache Restproblem, aus den Primimplikanten einer Schaltfunktion ein Minimalpolynom zusammenzubauen. Ein solches Minimalpolynom wird i.A. nicht aus allen Primimplikanten bestehen. Man muß deshalb eventuell unter den Primimplikanten eine Auswahl treffen. Hierfür gibt es zwei ganz naheliegende Kriterien.

Definition 1.23 *Sei e ein Boole'scher Ausdruck und $a \in \{0, 1\}^n$. Wir sagen e überdeckt a genau dann, wenn $\phi_a(e) = 1$ gilt. Ist M eine Menge von Monomen*

a_1	a_2	a_3	a_4	$f_1(a_1, a_2, a_3, a_4)$	$f_2(a_1, a_2, a_3)$
0	0	0	0	1	0
0	0	0	1	1	
0	0	1	0	1	1
0	0	1	1	1	
0	1	0	0	0	1
0	1	0	1	0	
0	1	1	0	0	1
0	1	1	1	1	
1	0	0	0	0	1
1	0	0	1	0	
1	0	1	0	0	1
1	0	1	1	0	
1	1	0	0	0	1
1	1	0	1	0	
1	1	1	0	0	0
1	1	1	1	1	

Tabelle 1.4: Funktionstabellen der Schaltfunktionen f_1 und f_2

und $F \subseteq \{0, 1\}^n$, so heißt die Abbildung $I : M \times F \rightarrow \{0, 1\}$,

$$I(m, a) = \begin{cases} 1 & \text{falls } m \text{ überdeckt } a \\ 0 & \text{falls sonst} \end{cases}$$

die Implikantentafel von M und F .

Implikantentafel

Der Name ‘Implikantentafel’ kommt daher, daß man I als Matrix aufschreiben kann, deren Zeilen mit den Elementen $m \in M$ und deren Spalten mit den Elementen $a \in F$ indiziert sind. Ist M die Menge der Primimplikanten einer Schaltfunktion f und ist $F = \{a \mid f(a) = 1\}$, so heißt I die *Primimplikantentafel* von f . Die Primimplikantentafeln der beiden Schaltfunktionen $f_1 : \{0, 1\}^4 \rightarrow \{0, 1\}$ und $f_2 : \{0, 1\}^3 \rightarrow \{0, 1\}$, deren Funktionstabellen in Tabelle 1.4 zu sehen sind, findet man in Tabelle 1.5.

Jeder Menge S von Monomen ordnen wir das Polynom

$$p(S) = \bigvee_{m \in S} m$$

zu. Eine Implikantentafel I definiert ein zugehöriges *Überdeckungsproblem*: finde eine Teilmenge $S \subseteq M$ von Monomen, so daß gilt: $p(S)$ überdeckt F . Eine solche Teilmenge heißt eine *Lösung* des Überdeckungsproblems. Ist I die Primimplikantentafel von f , so gilt offensichtlich

$$p(S) \equiv f(X)$$

für alle Lösungen S von I ; und die Lösungen S , für die $p(S)$ minimale Kosten hat, sind offensichtlich gerade die Minimalpolynome von f .

Für das Lösen von Überdeckungsproblemen gibt es zwei ganz offensichtliche Kriterien.

		f_1					
		0000	0001	0010	0011	0111	1111
(a)	$\overline{X_1}X_3X_4$	0	0	0	1	1	0
	$X_2X_3X_4$	0	0	0	0	1	1
	$\overline{X_1}\overline{X_2}$	1	1	1	1	0	0

		f_2					
		001	010	011	100	101	110
(b)	$\overline{X_1}X_3$	1	0	1	0	0	0
	$\overline{X_2}X_3$	1	0	0	0	1	0
	$\overline{X_1}X_2$	0	1	1	0	0	0
	$X_2\overline{X_3}$	0	1	0	0	0	1
	$X_1\overline{X_3}$	0	0	0	1	0	1
	$X_1\overline{X_2}$	0	0	0	1	1	0

Tabelle 1.5: Primimplikantentafeln für f_1 und f_2

f_1	0111	1111
$\overline{X_1}X_3X_4$	1	0
$X_2X_3X_4$	1	1

Tabelle 1.6: Vereinfachte Primimplikantentafel für f_1

wesentliches
Monom

Definition 1.24 Sei $I : M \times F \rightarrow \{0, 1\}$ ein Überdeckungsproblem und $m \in M$. Dann heißt m wesentlich, falls es ein $a \in F$ gibt, so daß a nur von m und keinem anderen Monom in M überdeckt wird.

Beispiel 1.17 Bei Funktion f_1 aus Tabelle 1.4 sind die Monome $\overline{X_1}\overline{X_2}$ und $X_2X_3X_4$ wesentlich. Monom $\overline{X_1}X_3X_4$ ist nicht wesentlich. Bei Funktion f_2 ist kein Monom wesentlich.

Ist Monom m wesentlich, so muß es offensichtlich in jeder Lösung vorkommen. Das Überdeckungsproblem $I(m) : M' \times F' \rightarrow \{0, 1\}$ entstehe aus I durch Entfernen von m aus M und durch Entfernen aller von m überdeckten Tupel a aus F . Dann ist eine Teilmenge S' von M' genau dann Lösung $S(m)$ von $I(m)$ wenn $S' \cup \{m\}$ Lösung von I ist. Eine billigste Lösung von Problem I kann also stets aus einer billigsten Lösung des kleineren Problems I' gewonnen werden.

Beispiel 1.18 Sei I das Überdeckungsproblem aus Tabelle 1.5(a) und $m = \overline{X_1}\overline{X_2}$. Dann ist I' das Problem aus Tabelle 1.6.

Wir können natürlich das gleiche Kriterium nochmals anwenden, um in diesem Fall die einzige Lösung minimaler Kosten zu bestimmen.

Tabelle 1.6 illustriert auch noch das zweite Kriterium:

Sei $I : M \times F \rightarrow \{0, 1\}$ ein Überdeckungsproblem, und es seien $m, m' \in M$. Wir sagen, daß m das Monom m' dominiert, falls $L(m) \leq L(m')$ und falls jedes Tupel a , das von m' überdeckt wird auch von m überdeckt wird.

Beispiel 1.19 In Tabelle 1.6 dominiert $X_2X_3X_4$ das Monom $\overline{X_1}X_3X_4$.

Wird m' von m dominiert, so kann man m' in jeder Lösung von I durch m ersetzen. Man erhält wieder eine Lösung, und diese ist nicht teurer als die alte Lösung. Deshalb kann man in diesem Fall m' einfach aus M entfernen.

Man kommt leider häufig in Situationen, in denen keines der beiden Kriterien anwendbar ist, beispielsweise in Tabelle 1.5(b). Im allgemeinen kommt man an dieser Stelle nur noch mit roher Gewalt weiter: Man sucht für *alle* $m \in M$ eine billigste Lösung $S(m)$ des Problems $I(m)$ und sucht dann unter den Lösungen $m \vee \bigvee_{r \in S(m)} r$ eine billigste aus. Da man bei den entstehenden kleineren Problemen immer wieder in die gleiche Situation geraten kann, wird das sehr schnell extrem aufwendig:

Für $k \in \mathbf{N}$ sei $i(k)$ die größte Zahl von Überdeckungsproblemen, die insgesamt generiert werden, wenn man mit einem Überdeckungsproblem mit k Monomen startet. Dann ist

$$i(1) = 1,$$

und für $k > 1$ können wir nach dem oben Gesagten $i(k)$ nur abschätzen durch

$$i(k) \leq k \cdot i(k-1).$$

Durch Induktion über k folgt

$$i(k) \leq k! = 1 \cdot 2 \cdot \dots \cdot k.$$

Ob man solche Probleme sehr viel schneller lösen kann, ist eine offene Frage, deren Bedeutung weit über die Grenzen der Optimierung von Boole'schen Ausdrücken hinausreicht [2, 4].

1.4 Schaltkreise

1.4.1 Gatter

Wir haben bereits in Abschnitt 1.2 über Gatter gesprochen. Das sind Schaltungen mit wenigen Eingängen und einem Ausgang, die gewisse einfache Schaltfunktionen berechnen. Wir gehen ab jetzt davon aus, daß uns Gatter zur Berechnung der folgenden Schaltfunktionen zur Verfügung stehen:

1. die bereits bekannten Schaltfunktionen \wedge , \vee und \sim .
2. NAND : $\{0, 1\}^2 \rightarrow \{0, 1\}$ mit

$$\text{NAND}(x, y) = \overline{x \wedge y} \text{ für alle } x, y.$$

3. \oplus : $\{0, 1\}^2 \rightarrow \{0, 1\}$ mit

$$\oplus(x, y) = 1 \Leftrightarrow x + y = 1 \text{ für alle } x, y.$$

Die letzte Schaltfunktion heißt auch *exklusives ODER (EXOR)* oder *Plus*

exklusives Oder

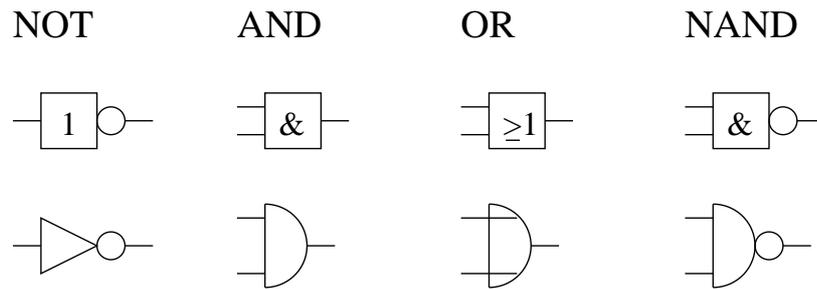


Abbildung 1.4: Schaltsymbole

modulo zwei, da

$$\oplus(x, y) = x + y \text{ mod } 2$$

für alle x, y gilt. Statt $\oplus(x, y)$ schreibt man gewöhnlich $x \oplus y$. Da die Funktion \oplus assoziativ ist, kann man in Ausdrücken wie $(x \oplus (y \oplus z))$ die Klammern weglassen. Wir bemerken noch, daß die Schaltfunktionen \wedge , \vee , NAND und \oplus alle kommutativ sind.

Ist f eine Schaltfunktion, so nennt man Gatter, die f berechnen f -Gatter. Uns stehen also jetzt \wedge -Gatter (AND-Gatter), \vee -Gatter (OR-Gatter), \sim -Gatter (Inverter), NAND-Gatter und \oplus -Gatter (EXOR-Gatter) zur Verfügung. Die Menge der direkt durch Gatter realisierbaren Funktionen fassen wir in der Menge

$$K = \{\wedge, \vee, \sim, \text{NAND}, \oplus\}$$

Schaltsymbol

zusammen. Man verwendet für Gatter üblicherweise die *Schaltsymbole* aus Abbildung 1.4. Die obere Reihe zeigt Schaltsymbole nach DIN, die untere Reihe Schaltsymbole wie sie im wissenschaftlichen Bereich verwendet werden. Die untere Reihe entspricht bis auf das OR-Gatter dem amerikanischen IEEE Standard. Wir werden sowohl die Schaltsymbole der oberen wie der unteren Reihe benutzen, allerdings in einer Schaltkreiszeichnung nur Symbole einer Reihe.

1.4.2 Schaltkreise

Schaltkreise erhält man nun, indem man Gatter auf spezielle Weise zusammenschaltet. Man geht dabei in vier Schritten vor.

Eingang

1. Man spezifiziert eine endliche Folge $X = (X_1, \dots, X_n)$ von *Eingängen*. Diese Eingänge werden eine ähnliche Rolle wie die Variablen in Boole'schen Ausdrücken spielen.
2. Man spezifiziert einen zyklfreien Graphen $G = (V, E)$ mit den folgenden Eigenschaften:
 - $\{0, 1\} \cup \{X_1, \dots, X_n\} \subseteq V$, d. h. jeder Eingang ist Knoten des Graphen G . Zusätzlich gibt es zwei spezielle Knoten 0 und 1. Diese Knoten werden später (kostenlos) die konstanten Signale 0 und 1 liefern.

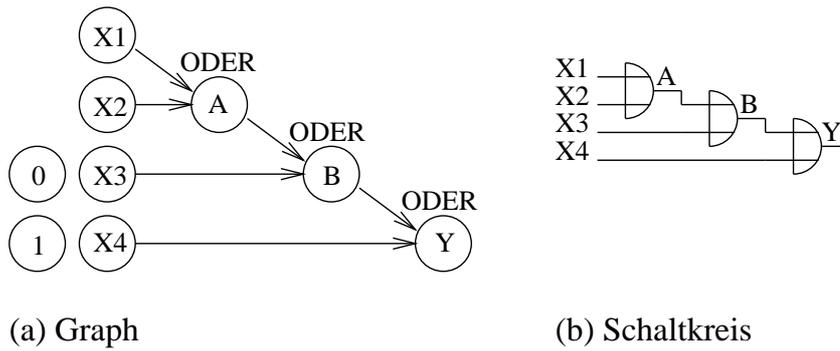


Abbildung 1.5: Zeichnen von Schaltkreisen

- Die Menge $\{0, 1\} \cup \{X_1, \dots, X_n\}$ der Eingänge bildet die Quellen von G .
- Jeder Knoten aus $I = V \setminus (\{X_1, \dots, X_n\} \cup \{0, 1\})$ hat Ingrad 1 oder 2.

Die Menge I heißt die Menge der *Gatter*. Die Kanten des Graphen geben die Verdrahtung der Gatter untereinander an. Da Gatter einen oder zwei Eingänge haben, muß auch der Ingrad jedes Gatters 1 oder 2 sein.

- Man spezifiziert eine Abbildung $g : I \rightarrow K$, die für jedes Gatter angibt, welche Funktion es berechnet. Diese Funktion muß sich mit dem Ingrad des Gatters vertragen, d. h. es muß gelten:

$$g(v) \in \begin{cases} \{\wedge, \vee, \text{NAND}, \oplus\} & \text{falls } \text{indeg}(v) = 2 \\ \{\sim\} & \text{falls } \text{indeg}(v) = 1 \end{cases}$$

- Man zeichnet eine Folge $Y = (Y_1, \dots, Y_n)$ von Knoten $Y_i \in V$ als *Ausgänge* aus.

Ausgang

Jedes 4-Tupel $S = (X, G, g, Y)$ mit den oben genannten Eigenschaften spezifiziert einen *Schaltkreis*. Zusätzlich werden wir auch den Begriff *Schaltnetz* als Synonym benutzen.

Schaltkreis

Man kann einen Schaltkreis S zeichnen, indem man den Graphen G zeichnet, und jedes Gatter v zusätzlich mit $g(v)$ beschriftet. Ein Beispiel findet man in Abbildung 1.5(a). Statt ein Gatter v mit $g(v)$ zu beschriften zeichnet man in der Regel jedoch direkt das zugehörige Schaltsymbol. Den Namen v des Gatters schreibt man an den Ausgang des Schaltsymbols. Weil aus den Schaltsymbolen die Richtung der Kanten hervorgeht, spart man sich beim Zeichnen die Spitzen der Pfeile. Die Kreise um die Quellen $0, 1, X_1, \dots, X_n$ läßt man weg. Werden die speziellen Knoten 0 und 1 nicht als Eingänge von Gattern oder als Ausgänge des Schaltkreises benutzt, läßt man sie in Zeichnungen ebenfalls einfach weg. Aus Abbildung 1.5(a) entsteht so Abbildung 1.5(b).

Beispiel 1.20 *Abbildung 1.6 zeigt vier weitere Schaltungen. Davon ist nur (a) ein Schaltkreis, die Schaltungen (b), (c) und (d) nicht. Bei (b), (c) und (d) gibt*

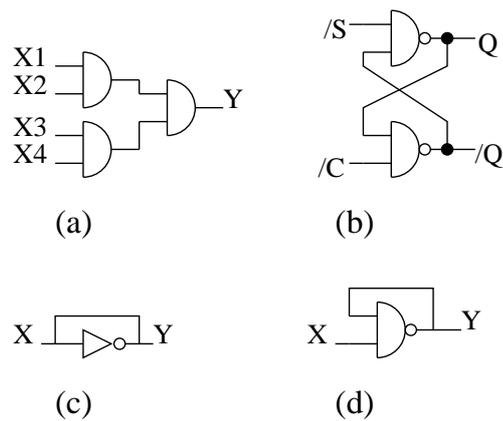


Abbildung 1.6: Beispiele und Gegenbeispiele von Schaltkreisen

es einen Zyklus. Überdies hat bei (c) ein Knoten mit einem Inverter Ingrad 2.

Man kann natürlich jede der Schaltungen aus Abbildung 1.6 physikalisch aufbauen und den Strom anschalten. Wir werden in Kurseinheit 2 sehen, daß eine dieser Schaltungen sogar sehr nützliche Arbeit leistet. Andere fangen eher an zu qualmen und gehen kaputt.

1.5 Rechnen mit Schaltkreisen

1.5.1 Einsetzungen

Wir definieren nun die Arbeitsweise von Schaltkreisen. Sei $S = (X, G, g, Y)$ ein Schaltkreis, $X = (X_1, \dots, X_n)$ und $G = (V, E)$. Weiter sei $\phi : \{X_1, \dots, X_n\} \rightarrow \{0, 1\}$ eine Einsetzung, die jedem Eingang X_i ein Signal $\phi(X_i) \in \{0, 1\}$ zuordnet. Wir definieren nun auf ziemlich offensichtliche Weise für jeden Knoten $v \in V$ den im Schaltkreis S durch v bei Einsetzung ϕ berechneten Wert $\phi(v)$.

Weil G zykelfrei ist, hat jeder Knoten $v \in V$ eine Tiefe. Wir setzen für die speziellen Knoten 0 und 1

$$\phi(0) = 0 \text{ und } \phi(1) = 1 .$$

Damit ist $\phi(v)$ definiert für alle Knoten v mit Tiefe 0. Wir definieren nun $\phi(v)$ durch Induktion über t für alle Gatter v .

Sei $t \in \mathbf{N}$, und $\phi(u)$ sei definiert für alle Gatter v mit Tiefe $t - 1$. Es sei v ein Gatter mit Tiefe t . Dann sind zwei Fälle möglich.

1. Ist $\text{indeg}(v) = 1$, so hat v einen direkten Vorgänger u mit Tiefe $t - 1$, es ist $g(v) = \sim$ und wir definieren

$$\phi(v) = \sim (\phi(u)) .$$