Literatur

I. Wegener, *The Complexity of Boolean Functions*. Teubner, Stuttgart und Wiley, New York, 1987.

S. W. Jablonski und O. B. Lupanov (Herausgeber), *Diskrete Mathematik und mathematische Fragen der Kybernetik*. Akademie-Verlag, Berlin, 1980.

R. Reischuk, *Einführung in die Komplexitätstheorie*. B.G. Teubner, Stuttgart, 1990.

O. B. Lupanov, Complexity of formulae realisation of functions of logical algebra. *Probl. Kibernetiki* **3** (1962) 782–811.

O. B. Lupanov, A method of synthesis of control systems. *Probl. Kibernetiki* **23** (1970) 31–110.

McColl and M. Paterson, The depth of all Boolean functions. *SIAM J. Comp.* **6** (1977) 373–380.

Gaskov, The depth of Boolean functions. *Probl. Kibernetiki* **34** (1978) 265–268.

Boolesche Funktionen I

$$
\begin{aligned}
B_{n,m} &= \{f \mid f \text{ bildet } \{0,1\}^n \text{ in } \{0,1\}^m \text{ ab } \}\,, \\
B_n &= B_{n,1} \text{ für } n \geq 1\,, \\
B &= \bigcup_{n\geq 1} B_n
\end{aligned}
$$

Lemma: Es gibt genau 2^{2^n} Funktionen in B_n.

Boolesche Funktionen II

x	Identität x	Negation $\overline{x}$	Konstante 0 k_0	Konstante 1 k_1
0	0	1	0	1
1	1	0	0	1

x_1	x_2	Konjunktion AND-Funktion $x_1 \wedge x_2$	Disjunktion OR-Funktion $x_1 \vee x_2$	Parity-Funktion XOR-Funktion $x_1 \oplus x_2$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Boolesche Funktionen III

$$x^0 = \overline{x} \quad \text{und} \quad x^1 = x$$

i) $x_1 \wedge x_2 = \overline{\overline{x_1} \vee \overline{x_2}}$ und $x_1 \vee x_2 = \overline{\overline{x_1} \wedge \overline{x_2}}$,

ii) $\overline{\overline{x}} = x, \quad \overline{x} = x \oplus 1, \quad x \oplus x = 0, \quad x \wedge x = x \vee x = x,$

iii) $x_1 \circ x_2 = x_2 \circ x_1$ und $(x_1 \circ x_2) \circ x_3 = x_1 \circ (x_2 \circ x_3)$

für $\circ \in \{\wedge, \vee, \oplus\}$,

iv) $(x_1 \oplus x_2) \cdot x_3 = (x_1 \cdot x_3) \oplus (x_2 \cdot x_3)$.

Boolesche Funktionen IV

für $\underline{a} = (a_1, a_2, \ldots, a_n) \in \{0,1\}^n$: $m_{\underline{a}}(x_1, x_2 \ldots, x_n) = x_1^{a_1} \wedge x_2^{a_2} \wedge \ldots \wedge x_n^{a_n}$

: $s_{\underline{a}}(x_1, x_2, \ldots, x_n) = x_1^{\overline{a_1}} \vee x_2^{\overline{a_2}} \vee \ldots \vee x_n^{\overline{a_n}}$

Satz: Für jede Boolesche Funktion $f \in B_n$ gelten

$a)$ $$f(x_1, x_2, \ldots, x_n) = \bigvee_{\underline{a} \in f^{-1}(1)} m_{\underline{a}}(x_1, x_2, \ldots, x_n),$$

$b)$ $$f(x_1, x_2, \ldots x_n) = \bigwedge_{\underline{a} \in f^{-1}(0)} s_{\underline{a}}(x_1, x_2, \ldots, x_n),$$

$c)$ $$f(x_1, x_2, \ldots, x_n) = \bigoplus_{\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}} a_{i_1 i_2 \ldots i_m} x_{i_1} x_{i_2} \ldots x_{i_m}$$

für gewisse $a_{i_1 i_2 \ldots i_m} \in \{0,1\}$.

Boolesche Funktionen V

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$
\begin{aligned}
& f(x_1, x_2, x_3) \\
& \quad = \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \vee \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee \overline{x_1} \cdot x_2 \cdot \overline{x_3} \\
& \qquad\qquad \vee x_1 \cdot x_2 \cdot \overline{x_3} \vee x_1 \cdot x_2 \cdot x_3 \\
& \quad = (x_1 \vee \overline{x_2} \vee \overline{x_3}) \cdot (\overline{x_1} \vee x_2 \vee x_3) \\
& \qquad\qquad \cdot (\overline{x_1} \vee x_2 \vee \overline{x_3}) \\
& \quad = x_1 x_2 x_3 \oplus x_1 x_2 \oplus x_2 x_3 \oplus x_1 \oplus 1
\end{aligned}
$$

Schaltkreis - Definition

Definition: Es sei $\mathcal{S}$ eine endliche Menge von Funktionen aus B. Ein (n, m)-Schaltkreis über $\mathcal{S}$ ist ein (knoten-)markierter, gerichteter und azyklischer Graph S mit folgenden Eigenschaften:

- n paarweise verschiedene Knoten von S sind mit $x_1, x_2, \ldots, x_n$ markiert,
- die mit einem x_i, $1 \leq i \leq n$, markierten Knoten haben keinen Vorgänger,
- die restlichen Knoten von S sind mit Elementen aus $\mathcal{S}$ markiert,
- die mit einem $f \in \mathcal{S} \cap B_k$ markierten Knoten haben k Vorgänger,
- m Knoten von S sind zusätzlich noch mit $y_1, y_2, \ldots, y_m$ markiert.

Schaltkreis - Komplexitätsmaße I

Definition:
Sei S ein (n,m)-Schaltkreis über $\mathcal{S}$.

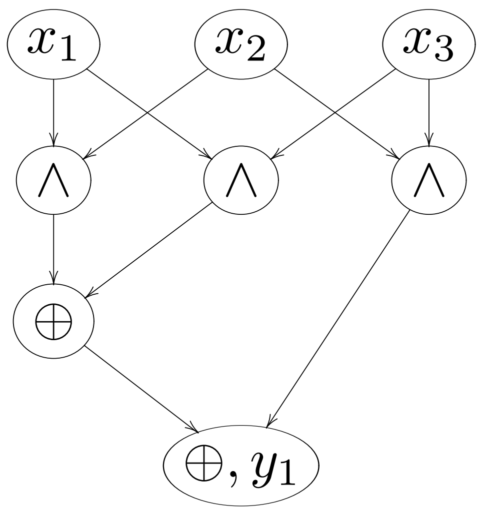
i) Wir definieren die Größe (oder Komplexität) $C(S)$ von S als die Anzahl der mit Elementen aus $\mathcal{S}$ markierten Knoten von S.

ii) Für einen Knoten g von S definieren wir die Tiefe $D(g)$ als die maximale Länge eines Weges von einem mit x_i, $1 \leq i \leq n$, markierten Knoten nach g.

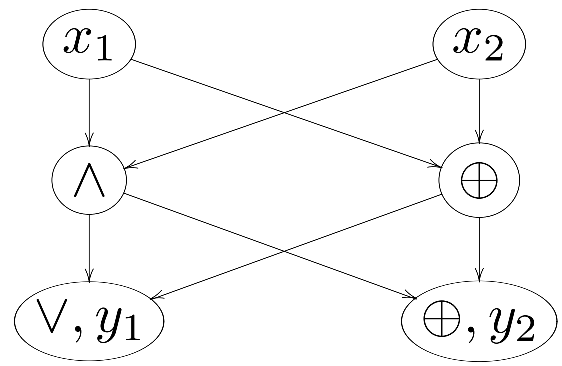
iii) Unter der Tiefe $D(S)$ des Schaltkreises S verstehen wir die maximale Tiefe der Knoten von S.

Schaltkreis - Beispiel

$$\mathcal{S} = \{\wedge, \vee, \oplus\}$$



S_1



S_2

Schaltkreis - induzierte Funktion

Definition: Sei S ein (n,m)-Schaltkreis über $\mathcal{S} \subseteq B$.

i) Dann definieren wir die Boolesche Funktion f_g, die in einem Knoten g von S induziert wird induktiv über die Tiefe des Knotens wie folgt:

– Sei $D(g) = 0$. Dann ist g mit einer Variablen x_i, $1 \leq i \leq n$, markiert, und wir setzen $f_g(x_1, x_2, \ldots, x_n) = x_i$.

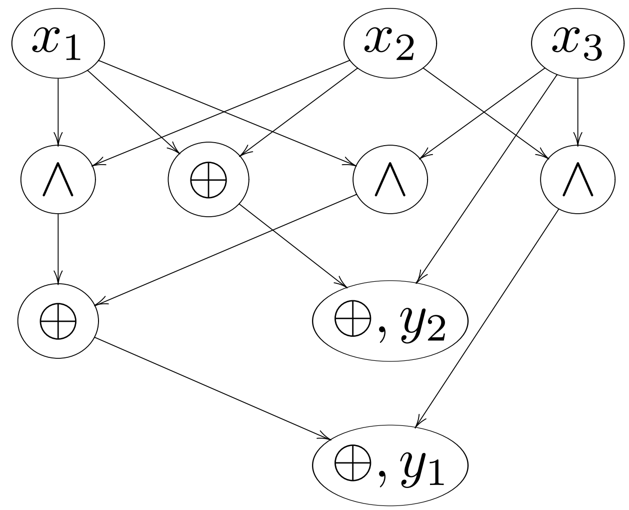
– Sei $D(g) > 0$. Dann ist g mit einer Funktion $f \in \mathcal{S}$ markiert. Sind $f \in B_k$, $g_1, g_2, \ldots, g_k$ die Vorgängerknoten von g und $f_{g_1}, f_{g_2}, \ldots, f_{g_k}$ die in den Vorgängerknoten induzierten Funktionen, so setzen wir $f_g(x_1, \ldots, x_n) = f(f_{g_1}(x_1, \ldots, x_n), f_{g_2}(x_1, \ldots, x_n), \ldots, f_{g_k}(x_1, \ldots, x_n))$.

ii) Sind $h_1, h_2, \ldots h_m$ die Knoten von S, die zusätzlich mit $y_1, y_2, \ldots, y_m$ markiert sind, so berechnet S die Funktion $f : \{0,1\}^n \to \{0,1\}^m$, die durch

$$f(x_1, \ldots, x_n) = ((f_{h_1}(x_1, \ldots, x_n), f_{h_2}(x_1, \ldots, x_n), \ldots, f_{h_m}(x_1, \ldots, x_n))$$

definiert ist.

Schaltkreis zur Addition von drei Dualziffern



Komplexitäten Boolescher Funktionen

Definition:

Für eine Boolesche Funktion $f : \{0,1\}^n \rightarrow \{0,1\}^m$ und eine endliche Teilmenge $\mathcal{S}$ von B definieren wir die Größe $C_{\mathcal{S}}(f)$ und die Tiefe $D_{\mathcal{S}}(f)$ von f bezüglich $\mathcal{S}$ durch

$$C_{\mathcal{S}}(f) = \min\{C(S) \mid \ S \text{ berechnet } f \text{ und ist Schaltkreis über } \mathcal{S}\}$$

und

$$D_{\mathcal{S}}(f) = \min\{D(S) \mid \ S \text{ berechnet } f \text{ und ist Schaltkreis über } \mathcal{S}\}.$$

Schaltkreis - Komplexitätsmaße II

Definition:

i) Wir sagen, dass ein Schaltkreis S fan-out-k-beschränkt ist, wenn der Ausgangsgrad eines jeden Knotens von S höchstens k ist.

ii) Für eine Boolesche Funktion $f : \{0,1\}^n \to \{0,1\}^m$ und eine endliche Teilmenge $\mathcal{S}$ von B definieren wir $C_{k,\mathcal{S}}(f)$ als das Minimum der Größen $C(S)$, wobei das Minimum über alle Schaltkreise über $\mathcal{S}$ genommen wird, die f berechnen und fan-out-k-beschränkt sind.

Formeln und Länge

Wir definieren Formeln bzw. Ausdrücke über $\mathcal{S}$ durch die drei folgenden Bedingungen:

1. Ist f eine n-stellige Funktion aus $\mathcal{S}$, $n \geq 1$, so ist $f(x_1, x_2, \ldots x_n)$ eine Formel über $\mathcal{S}$.

2. Ist f eine n-stellige Funktion aus $\mathcal{S}$ und sind $H_1, H_2, \ldots H_n$ Formeln über $\mathcal{S}$, so ist auch $f(H_1, H_2, \ldots H_n)$ eine Formel über $\mathcal{S}$.

3. Formeln entstehen nur mittels 1. und 2.

Als *Länge* $L_{\mathcal{S}}(F)$ einer Formel F definieren wir nun die Anzahl der in F vorkommenden Funktionssymbole.

Fakt: $L_{\mathcal{S}}(F) = C_{1,\mathcal{S}}(S)$, wobei S der Schaltkreis zu F ist.

Vollständigkeit I

Definition:
Eine Menge $\mathcal{S} \subseteq B$ heißt vollständig, falls jede Boolesche Funktion durch einen Schaltkreis über $\mathcal{S}$ berechnet werden kann.

Lemma: Eine Menge $\mathcal{S} \subseteq B$ ist genau dann vollständig, wenn jede Boolesche Funktion aus B durch einen Schaltkreis über $\mathcal{S}$ berechnet werden kann.

Lemma: Ist $\mathcal{S}_1$ eine vollständige Menge und ist jede Funktion $f \in \mathcal{S}_1$ durch einen Schaltkreis über $\mathcal{S}_2$ berechenbar, so ist auch $\mathcal{S}_2$ vollständig.

Vollständigkeit II

$$
\begin{aligned}
T_0 &= \{f \mid f \in B_n, n \geq 1, f(0,0,\ldots,0) = 0\},\\
T_1 &= \{f \mid f \in B_n, n \geq 1, f(1,1,\ldots,1) = 1\},\\
Lin &= \{f \mid f \in B_n, n \geq 1, f(x_1,\ldots x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_nx_n\\
&\qquad\qquad \text{für gewisse } a_i \in \{0,1\}, 1 \leq i \leq n\},\\
Mon &= \{f \mid f \in B_n, n \geq 1, f(a_1,a_2,\ldots,a_n) \leq f(b_1,b_2,\ldots b_n)\\
&\qquad\qquad \text{für } a_i \leq b_i, 1 \leq i \leq n\},\\
Sd &= \{f \mid f \in B_n, n \geq 1, f(a_1,a_2,\ldots,a_n) = \overline{f(\overline{a_1},\overline{a_2},\ldots,\overline{a_n})}\}.
\end{aligned}
$$

Satz: (Vollständigkeitskriterium von Post)
Eine Menge $\mathcal{S} \subseteq B$ ist genau vollständig, wenn sie in keiner der Mengen T_0, T_1, Lin, Mon und Sd enthalten ist.

Beziehungen zwischen Komplexitätsmaßen I

Satz: Sind $\mathcal{S}_1$ und $\mathcal{S}_2$ zwei vollständige Mengen und $k \geq 1$ eine natürliche Zahl, so gibt es (von $\mathcal{S}_1$ und $\mathcal{S}_2$ abhängige) Konstanten c_1, c_2, d_1, d_2 und $c_{k,1}, c_{k,2}$, so dass für jede Boolesche Funktion f

$$c_1 \cdot C_{\mathcal{S}_2}(f) \leq C_{\mathcal{S}_1}(f) \leq c_2 \cdot C_{\mathcal{S}_2}(f),$$
$$d_1 \cdot D_{\mathcal{S}_2}(f) \leq D_{\mathcal{S}_1}(f) \leq d_2 \cdot D_{\mathcal{S}_2}(f),$$
$$c_{k,1} \cdot C_{k,\mathcal{S}_2}(f) \leq C_{k,\mathcal{S}_1}(f) \leq c_{k,2} \cdot C_{k,\mathcal{S}_2}(f)$$

gelten.

Beziehungen zwischen Komplexitätsmaßen II

Satz: Für jede natürliche Zahl $k \geq 2$ und jede vollständige Menge $\mathcal{S}$ gibt es eine Konstante c derart, dass

$$C_{k,\mathcal{S}}(f) \leq c \cdot C_{\mathcal{S}}(f)$$

jede Funktion f aus B gilt.

Satz: Seien $\mathcal{S}$ eine vollständige Menge mit $\mathcal{S} \subseteq B_2$ und f eine beliebige Funktion aus B. Dann gelten

$$\log(L_{\mathcal{S}}(f) + 1) \leq D_{\mathcal{S}}(f).$$

und

$$D_{\mathcal{S}}(f) \leq k(\mathcal{S}) \cdot \log(L_{\mathcal{S}}(f) + 1) \text{ mit } k(\mathcal{S}) = \frac{1 + D_{\mathcal{S}}(\overline{x}y \vee xz)}{\log(3) - 1}.$$

Beziehungen zwischen Komplexitätsmaßen III

Satz:
Für jede vollständige Menge $\mathcal{S} \subseteq B_2$ und jede Funktion $f \in B$ gelten

$$C_{\mathcal{S}}(f) \leq L_{\mathcal{S}}(f) \quad \text{und} \quad \log(C_{\mathcal{S}}(f) + 1) \leq D_{\mathcal{S}}(f).$$

Satz:
Für jede vollständige Menge $\mathcal{S} \subseteq B$ gibt es eine Konstante $k'(\mathcal{S})$ derart, dass für jede Funktion $f \in B$

$$k'(\mathcal{S}) \cdot D_{\mathcal{S}}(f) \cdot \log(D_{\mathcal{S}}(f)) \leq C_{\mathcal{S}}(f)$$

gilt.

Asymptotische Komplexität

$f \in B_n$ hängt wesentlich von der Variablen x_i ab, , $1 \leq i \leq n$, wenn es Werte a_j für $1 \leq j \leq n$, $j \neq i$, derart gibt, dass

$$f(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_n) \neq f(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_n)$$

Satz: Für jede Boolesche Funktion $f \in B_n$, die von allen Variablen wesentlich abhängt, gilt $C_{B_2}(f) \geq n-1$.

Definition: Für ein Komplexitätsmaß $K \in \{C, L, D\}$ und eine Menge $\mathcal{S} \subseteq B$ von Basisfunktionen definieren wir die Funktion $K_{\mathcal{S}} : \mathbf{N} \to \mathbf{N}$ vermöge

$$K_{\mathcal{S}}(n) = \max\{K_{\mathcal{S}}(f) \mid f \in B_n\}\,.$$

Obere Schranken I

Satz: Es gibt eine natürliche Zahl n_0 derart, dass für jede natürliche Zahl $n \geq n_0$

$$C(n) \leq \frac{2^n}{n} + o\Big(\frac{2^n}{n}\Big)$$

gilt.

Satz: Es gibt eine natürliche Zahl $n_0 \geq 1$ derart, dass für jede natürliche Zahl $n \geq n_0$

$$L(n) \leq \frac{2^{n+1}}{\log(n)} + o\Big(\frac{2^n}{\log(n)}\Big)$$

gilt.

Obere Schranken II

Satz: Für jede natürliche Zahl $n \geq 2$ gilt

$$D(n) \leq n + \lceil \log(n) \rceil \,.$$

Satz: Es gibt eine natürliche Zahl $n_0 \geq 1$ derart, dass für jede natürliche Zahl $n \geq n_0$

$$D(n) \leq n - \log(\log(n)) + 2 + o(1)$$

gilt.

Einige Abschätzungen

Für natürliche Zahlen $n \geq 1$ und $b \geq 1$ und ein Komplexitätsmaß $K \in \{C, D, L\}$ setzen wir

$$K(n,b) = \#(\{f \mid f \in B_n, K_{B_2}(f) \leq b\}) .$$

Lemma: Für natürliche Zahlen $n \geq 1$ und $b \geq 1$ gelten

$$L(n,b) \leq n^{b+1} 64^b \quad \text{und} \quad C(n,b) \leq \frac{b \cdot 16^b \cdot (n+b)^{2b}}{b!} .$$

Corollary: Es gibt eine natürliche Zahl n_0 derart, dass für alle $n \geq n_0$

$$C(n,b) \leq b e^n (16e(n+b))^b$$

gilt.

Untere Schranken

Satz: Für jede natürliche Zahl $n \geq 1$ gilt

$$\frac{2^n}{\log(n)}(1 - \delta(n)) < L(n) \quad \text{mit} \quad \delta(n) = \frac{\log(n)}{2^n} + 6\frac{1}{\log(n)}\,.$$

Satz: Es gibt eine natürliche Zahl n_0 derart, dass für alle $n \geq n_0$

$$n - \log(\log(n)) - 2 < D(n)$$

gilt.

Satz: Für jede (beliebig kleine) Zahl δ gibt es eine natürliche Zahl n_0 derart, dass

$$\frac{2^n}{n}(1 - \delta) < C(n)$$

für alle $n \geq n_0$ gilt.

Asymptotisches Verhalten

Satz:

Es gelten die folgenden Beziehungen:

i) $C(n) = \Theta(\frac{2^n}{n})$,

ii) $L(n) = \Theta(\frac{2^n}{\log(n)})$,

iii) $D(n) = \Theta(n - \log(\log(n)))$.

Eine komplizierte Funktion

$f \in B_{2^k+3k+3}$

Einteilung der $2^k + 3k + 3$ Variablen:

$\underline{x} = (x_1, x_2, \ldots, x_{2^k})$, $\underline{a} = (x_{2^k+1}, x_{2^k+2}, \ldots x_{2^k+k})$,
$\underline{b} = (x_{2^k+k+1}, x_{2^k+k+2}, \ldots x_{2^k+2k})$, $\underline{c} = (x_{2^k+2k+1}, x_{2^k+2k+2}, \ldots x_{2^k+3k})$,
$p = x_{2^k+3k+1}$, $q = x_{2^k+3k+2}$, $r = x_{2^k+3k+3}$

a, b and c — die natürlichen Zahlen mit den Binärdarstellungen $\underline{a}$, $\underline{b}$ und $\underline{c}$

$$\begin{aligned} f(x_1, \ldots x_{2^k+3k+3}) &= f(\underline{x}, \underline{a}, \underline{b}, \underline{c}, p, q, r) \\ &= \big[q \wedge [(x_a \wedge x_b) \vee (p \wedge x_b \wedge x_c^r)]\big] \vee [\overline{q} \wedge (x_a \oplus x_b)]. \end{aligned}$$

Satz: $C(f) \geq 3 \cdot 2^k - 3.$

K- and KA-Ausdrücke

Definition:
Es sei X eine (unendliche) Menge von Variablen.

i) K-Ausdrücke sind wie folgt induktiv definiert.
— Für jede Variable $x \in X$ sind x und $\overline{x}$ K-Ausdrücke.
— Ist U ein K-Ausdruck und kommt weder die Variable x selbst noch ihre Negation $\overline{x}$ in U vor, so sind $U \wedge x$ und $U \wedge \overline{x}$ auch K-Ausdrücke.

ii) KA-Ausdrücke sind wie folgt induktiv definiert.
— Jeder K-Ausdruck ist ein KA-Ausdruck.
— Sind V_1 und V_2 zwei KA-Ausdrücke, so ist $V_1 \vee V_2$ ein KA-Ausdruck.

iii) V ist ein KA-Ausdruck für $f \in B_n$, falls $f = V$ gilt.

Kosten eines KA-Ausdruckes

Definition:

Die Kosten eines KA-Ausdrucks sind induktiv wie folgt definiert:

— Für $x \in X$ gilt $k(x) = k(\overline{x}) = 1$.

— Für $x \in X$ und einen K-Ausdruck U, der weder x noch $\overline{x}$ enthält, gilt $k(U \wedge x) = k(U \wedge \overline{x}) = k(U) + 1$.

— Sind V_1 und V_2 KA-Ausdrücke, so gilt $k(V_1 \vee V_2) = k(V_1) + k(V_2)$.

Implikanten

Definition:
i) Ein K-Ausdruck U heißt Implikant der Funktion f, falls er für jede Belegung der Variablen U nur dann den Wert 1 annimmt, wenn auch f den Wert 1 annimmt.

ii) Ein K-Ausdruck U heißt Primimplikant von f, wenn U ein Implikant von f ist und bei Streichung einer beliebigen (einfachen oder negierten) Variable in U ein K-Ausdruck entsteht, der kein Implikant von f ist.

$I(f)$ – Menge der Implikanten von f

$PI(f)$ – Menge der Primimplikanten von f

Einige Aussagen

Lemma:
Ein hinsichtlich der Kosten minimaler KA-Ausdruck für eine Boolesche Funktion f ist die Alternative von Primimplikanten von f.

Lemma:
Ein K-Ausdruck U, der weder x noch $\overline{x}$ enthält, ist genau dann ein Implikant für f, wenn $U \wedge x$ und $U \wedge \overline{x}$ Implikanten von f sind.

Theorem:
Das Problem de Bestimmung eines minimalen KA-Ausdrucks für die Funktion f aus der reduzierten PI-Tafel von f ist NP-vollständig.

Algorithmus von Quine und McCluskey

Eingabe: Boolesche Funktion $f \in B_n$ als Wahrheitstabelle,
Ausgabe: Menge $PI(f)$ der Primimplikanten von f

$Q_n := \{m_{\underline{a}} : \underline{a} \in f^{-1}(1)\ \}$;
$i := n$;
WHILE $Q_i \neq \emptyset$
  BEGIN $i := i - 1$;
    $Q_i := \{U$: es gibt eine Variable x derart, dass x und $\overline{x}$ in U nicht vorkommen und $U \wedge x, U \wedge \overline{x} \in Q_{i+1}$ gilt $\}$;
    $P_{i+1} := \{U : U \in Q_{i+1}$ und jede Menge $V \in Q_i$ ist kein Teilausdruck von $U\ \}$
  END ;
$PI(f) := \bigcup_{k=i+1}^{n} P_k$

Algorithmus zur Reduktion einer PI-Table

Eingabe: PI-Tafel der Funktion f

Ausgabe: Teilmenge R von Primimplikanten von f und reduzierte PI-Tafel

$R := \emptyset$; $T_0 :=$ Matrix mit einer Zeile und Spalte und Eintrag 0 ;
$T_1 :=$ PI-Tafel von f; $i := 1$;
WHILE $T_i \neq T_{i-1}$
BEGIN $M_i := \{\underline{a}$: zu $\underline{a}$ gehörige Spalte enthält genau eine $1\}$;
$N_i := \{U : U$ ist Primimplikant in $T_i, U(\underline{a}) = 1$ für ein $a \in M_i\}$;
$R := R \cup N_i$;
T_i' entstehe aus T_i durch Streichen aller Zeilen $U \in N_i$ und
Streichen aller Spalten zu $\underline{b}$ mit $U(\underline{b}) = 1$ für ein $U \in N_i$;
T_{i+1} entstehe aus T_i' durch Streichen aller Spalten c,
für die eine Spalte c' mit $c' \leq c$ existiert;
$i := i + 1$
END

Verzweigungsprogramm – Definition

Definition: Ein Verzweigungsprogramm ist ein gerichteter Graph mit Kanten- und Knotenmarkierungen und drei Sorten von Knoten:

- genau einer Quelle, die mit einer Booleschen Variablen markiert ist, deren Eingangsgrad 0 und Ausgangsgrad 2 betragen und von den beiden vom Knoten ausgehenden Kanten ist die eine mit 0 und die andere mit 1 markiert,
- inneren Knoten, die mit einer Booleschen Variablen markiert sind, deren Eingangsgrad mindestens 1 ist, deren Ausgangsgrad 2 ist und von den beiden vom Knoten ausgehenden Kanten ist die eine mit 0 und die andere mit 1 markiert, und
- zwei Senken, die mit 0 bzw. 1 markiert sind und deren Ausgangsgrade 0 sind.

Verzweigungsprogramm – berechnete Funktion

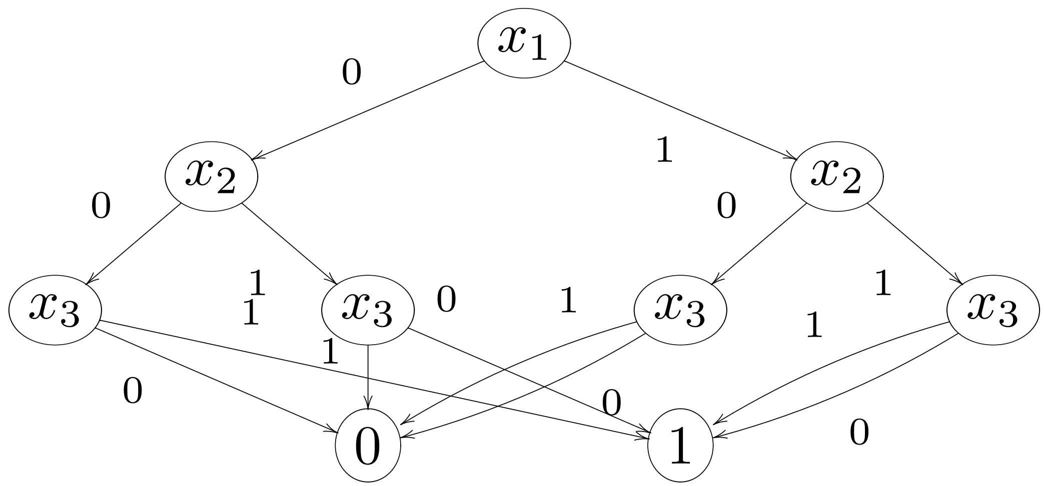
Defintion:

Sei G ein Verzweigungsprogramm, dessen innere Knoten mit x_i, $1 \leq i \leq n$, markiert sind.

Wir ordnen G wie folgt eine Funktion f_G zu:

Sei $\underline{a} = (a_1, a_2, \ldots, a_n) \in \{0,1\}^n$. Wir beginnen mit der Quelle und folgen stets bei einem Knoten x_i, $1 \leq i \leq n$, der mit a_i markierten Kante. Der Funktionswert $f_G(\underline{a})$ ist durch Markierung der erreichten Senke gegeben.

Verzweigungsprogramm – Beispiel I



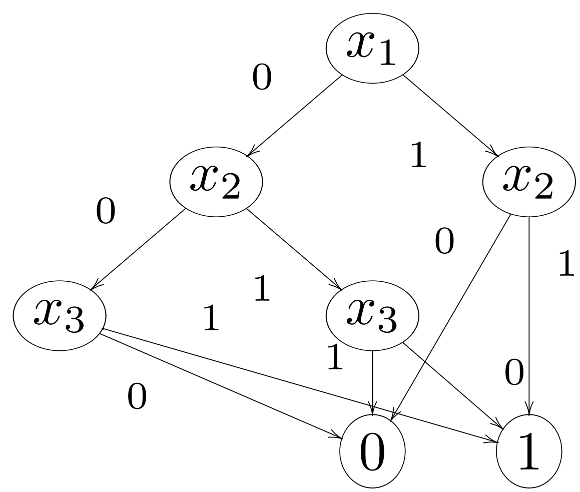
Größe und Tiefe eines Verzweigungsprogramms

Definition:
Für ein Verzweigungsprogramm G definieren wir die Größe $VC(G)$ bzw. die Tiefe $VD(G)$ als die um 1 erhöhte Anzahl der inneren Knoten von G bzw. die Tiefe von G.

Definition:
Für eine Boolesche Funktion f und $K \in \{C, D\}$ setzen wir

$$VK(f) = \min\{VK(G) \mid G \text{ berechnet } f\}\,.$$

Verzweigungsprogramm – Beispiel II



Resultate zur Komplexität von Verzweigungsprogrammen I

Theorem:
Für jede Boolesche Funktion $f \in B_n$ gelten

$$VC(f) \leq 2^n - 1 \quad \text{und} \quad VD(f) \leq n\,.$$

Theorem:
Für jede vollständige Menge $\mathcal{S}$ gibt es zwei Konstanten c_1 und c_2 derart, dass für jede Boolesche Funktion $f \in B_n$

$$C_{\mathcal{S}}(f) \leq c_1 \cdot (VC(f) + 2) \quad \text{und} \quad D_{\mathcal{S}}(f) \leq c_2 \cdot (VD(f) + 1)$$

gelten.

Resultate zur Komplexität von Verzweigungsprogrammen II

Bezeichnung: $VK(n) = \max\{VK(f) \mid f \in B_n\}$ für $K \in \{C, D\}$

Theorem: Für hinreichend großes n gelten

$$\frac{2^n}{3n} - 2 \leq VC(n) \leq 2^n - 1$$

und

$$\frac{n - \log(n) - c}{2} - 1 \leq VD(n) \leq n\,,$$

wobei c eine Konstante ist.

Turing-Maschine

Eine Turing-Maschine ist ein Tupel $M = (X, Z, z_0, Q, \delta)$, wobei
– X das Eingabealphabet ist,
– Z die Menge der Zustände ist,
– $z_0 \in Z$ der Anfangszustand ist,
– $Q \subseteq Z$ die Menge der Endstände ist, und
– die Überführungsfunktion δ eine Funktion von $(Z \setminus Q) \times (X \cup \{*\})$ in $Z \times (X \cup \{*\}) \times \{R, N, L\}$ ist.

$*$, R, L and N bezeichnen das Symbol für die Leere einer Bandzelle und die Richtungen *nach rechts* oder *nach links* für die Bewegung des Kopfes bzw. keine Kopfbewegung.

Akzeptanz durch Turing- Maschinen

Die Turing-Maschine M akzeptiert das Wort $w = a_1a_2 \dots a_n$, $a_i \in X$ für $1 \leq i \leq n$, wenn folgende Bedingungen erfüllt sind:

1) zu Beginn der Arbeit steht w auf dem Band,
2) zu Beginn der Arbeit ist M im Zustand z_0,
3) zu Beginn der Arbeit steht der Kopf über der Zelle, in der a_1 steht,
4) die Maschine stoppt in einem Endzustand,
5) bei Beendigung der Arbeit steht auf dem Band nur ein ausgezeichnetes Symbol $1 \notin X$,
6) 1 steht in der Zelle, in der zu Beginn a_1 stand, und der Kopf befindet sich erneut über dieser Zelle.

w wird von M abgelehnt, wenn in Punkt 5 der vorstehenden Definition der Akzeptanz 1 durch $0 \notin X$ ersetzt wird.

Turing-Maschinen versus Schaltkreise

Definition: Eine Turing-Maschine M heißt bewegungsuniform, wenn die Position des Kopfes nach i Schritten nur von der Länge n des Eingabewortes (und nicht vom Wort selbst) abhängt.

Theorem: Wird eine Sprache L durch eine bewegungsuniforme Turing-Maschine in der Zeit t entschieden, so gibt es eine Konstante c und eine Folge von Schaltkreisen, die L mit der Komplexität t' mit $t'(n) = ct(n)$ entscheidet.

Theorem: Zu jeder Turing-Maschine, die eine Sprache L in der Zeit t entscheidet, gibt es eine bewegungsuniforme Turing-Maschine, die L in $O(t(n)\log(t(n)))$ entscheidet.

Theorem: Wird eine Sprache L durch eine Turing-Maschine in der Zeit t entschieden, so gibt es eine Folge von Schaltkreisen, die L mit der Komplexität $O(t(n)\log(t(n)))$ entscheidet.

Beispiel für einen Schaltkreis zur Simulation einer Turing-Maschine