BOOLESK ALGEBRA OCH BOOLESKA FUNKTIONER

ANDERS BJÖRNER OCH KIMMO ERIKSSON

Boolesk algebra skapades vid 1800-talets mitt av den engelske matematikern George Boole. Den ger en gemensam ram för mängdlära, satslogik och teori för vissa digitala kretsar. Vi skall här ge en introduktion till boolesk algebra i det *ändliga* fallet, som är av särskild betydelse i datalogin.

1. Boolesk algebra

Grunden för boolesk räkning är följande två räknetabeller:

$$\begin{array}{c|cc} + & 0 & 1 \\ \hline 0 & 0 & 1 \\ 1 & 1 & 1 \end{array} \quad ; \quad \begin{array}{c|cc} \cdot & 0 & 1 \\ \hline 0 & 0 & 0 \\ 1 & 0 & 1 \end{array} \quad ; \text{ samt } \begin{cases} \bar{1} = 1 - 1 = 0 \\ \bar{0} = 1 - 0 = 1 \end{cases}$$

De fungerar som vanlig addition och multiplikation sånär som på att alla icke-noll-värden representeras av 1. Värdena 0 och 1 kan tolkas som "falskt" och "sant", varvid + motsvarar "eller" (OR) och $\cdot$ motsvarar "och" (AND). Komplement, $\bar{x}$, motsvarar "icke" (NOT).

Ytterligare en tolkning är i mängdtermer: 1 motsvarar en fylld mängd X, 0 motsvarar tomma mängden $\varnothing$, + motsvarar *union* $\cup$, $\cdot$ motsvarar *snitt* $\cap$, och komplement är komplement med avseende på X, dvs $\bar{x} = X - x$.

Objektet som vi studerat ovan är den endimensionella *booleska algebran* B_1. Den har alltså två element, $\{0, 1\}$ eller {falskt, sant}, eller $\{\varnothing, X\}$, hur vi nu vill se på den. Det här kan generaliseras till flera dimensioner:

$B_n \cong \{n$-dimensionella $\{0,1\}$-vektorer; $+, \cdot, \bar{}$ sker komponentvis$\}$
(Ex.: $00101 + 10001 = 10101$; $\overline{00101} = 11010$ $[B_5]$)
$\cong \{n$-dim. {sant, falskt}-vektorer; OR, AND, NOT sker komponentvis$\}$
(Ex.: SFS AND FSS = FFS; NOT (SFS)=FSF $[B_3]$)
$\cong$ {alla delmängder till en n-mängd X, t.ex. $\{1, 2, \ldots, n\}$; $\cup, \cap$ och $\bar{}$ m.a.p. X}
(Ex.: $\{3,5\} \cup \{1,5\} = \{1,3,5\}$; $\overline{\{3,5\}} = \{1,2,4\}$ $[B_5]$)

Tecknet $\cong$ betyder isomorfi, dvs sånär som på olika *namn* på saker och ting så är det exakt samma objekt i de tre exemplen. Isomorfin mellan de första två är trivial: $0 \leftrightarrow F, 1 \leftrightarrow S$, osv. Isomorfin mellan första och tredje exemplet är nästan lika enkel: $00101 \leftrightarrow \{3,5\}$, $10001 \leftrightarrow \{1,5\}$, dvs 1:orna anger vilka element av $\{1, \ldots 5\}$ som är med i delmängden.

För varje n finns alltså en unik boolesk algebra B_n, med flera tänkbara tolkningar. B_n har 2^n *element*, ty för var och en av de n positionerna i vektorn kan vi välja en 0:a eller en 1:a.

För att renodla egenskaperna hos den booleska algebran kan man lämna de konkreta representationerna ovan och se B_n som en abstrakt mängd med 2^n element och med tre

operationer: $\wedge, \vee$ och $^-$, som uppfyller samma egenskaper som $\cap, \cup$ och $^-$. Vissa av dessa egenskaper, från vilka de övriga kan härledas, väljs ut och kallas *axiomen* för en boolesk algebra. Till att börja med har man en partialordning $\leq$ som motsvarar $\subseteq$ för mängder eller komponentvis $\leq$ för $\{0,1\}$-vektorer. Det finns två särskilda element $\mathbf{0}$ och $\mathbf{1}$ så att $\mathbf{0} \leq x \leq \mathbf{1}$ för alla x i B_n. $\mathbf{0}$ och $\mathbf{1}$ motsvarar $\varnothing$ respektive hela $X = \{1, 2, \ldots, n\}$. Några exempel på axiom är:

$$\left.\begin{aligned} x \wedge (y \vee z) &= (x \wedge y) \vee (x \wedge z) \\ x \vee (y \wedge z) &= (x \vee y) \wedge (x \vee z) \end{aligned}\right\} \quad \text{distributivitet}$$

$$\bar{\bar{x}} = x$$

$$\left.\begin{aligned} \overline{x \wedge y} &= \bar{x} \vee \bar{y} \\ \overline{x \vee y} &= \bar{x} \wedge \bar{y} \end{aligned}\right\} \quad \text{de Morgans lagar}$$

Men, eftersom den booleska algebran är isomorf med tolkningarna ovan så finns det ingen anledning att lära sig dessa axiom utantill emedan de följer av räknereglerna för t.ex. mängder eller $\{0,1\}$-vektorer.

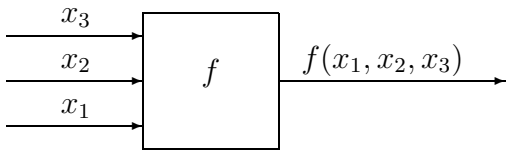
I fortsättningen betraktar vi B_n just som $\{n$-dim. $\{0,1\}$-vektorer$\}$ och använder alltså operatorerna $+$, $\cdot$ och $^-$. Men i livlig åtanke har vi alltid omtolkningen till räkning med sanningsvärden.

2. Booleska funktioner

En *boolesk funktion* är en funktion $f : B_n \to B_1$, dvs en funktion $f(x_1, x_2, \ldots, x_n) = y$ där alla x_i och y är 0 eller 1. Man representerar enklast f med en "sanningstabell", t.ex.

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Men man kan också se f som en "logisk krets"



som matar ut ett värde beroende på ingångsvärdena x_1, x_2, x_3.

Den funktion f som ges av tabellen ovan kan också uttryckas som ett "booleskt polynom", dvs ett uttryck med $+, \cdot$ och $^-$ samt variabler:

$$f(x_1, x_2, x_3) = \bar{x}_1\bar{x}_2x_3 + \bar{x}_1x_2\bar{x}_3 + x_1\bar{x}_2\bar{x}_3 + x_1x_2x_3.$$

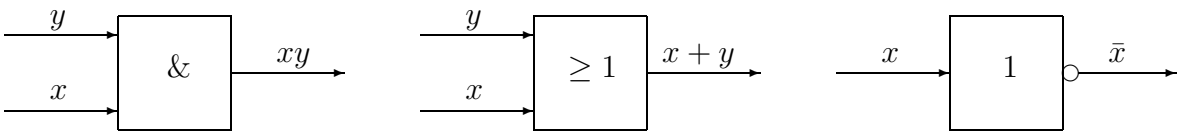
Hur vet man det? Jo, varje term motsvarar en rad i tabellen där f ska vara 1. Att t.ex. $f(0,0,1) = 1$ motsvaras av $\bar{x}_1\bar{x}_2x_3$, ty denna produkt blir 1 omm alla faktorer är 1, dvs omm $x_1 = 0$, $x_2 = 0$, $x_3 = 1$. På samma sätt är $\bar{x}_1x_2\bar{x}_3 = 1$ omm $(x_1, x_2, x_3) = (0, 1, 0)$. Alltså kommer summan av dessa termer att vara 1 precis när f ska vara 1. Detta sätt att uttrycka f kallas "disjunktiv normalform".

Vi kan ur detta dra slutsatsen att *varje boolesk funktion kan skrivas som ett booleskt polynom.*

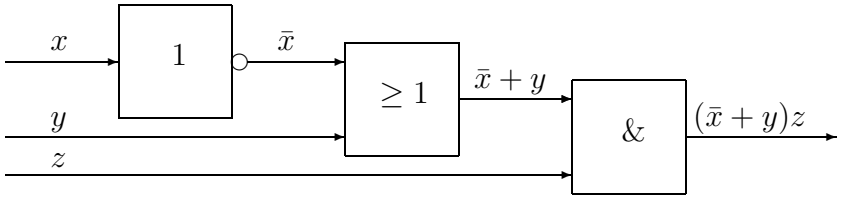
Men det finns i allmänhet många olika sätt att uttrycka samma booleska funktion. Vi har t.ex.

$$\begin{cases} xy + x\bar{y} = x \\ xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\bar{z} = x \end{cases}$$

Den disjunktiva normalformen är ofta den *dummaste* om man är ute efter ett enkelt uttryck. Varför är man ute efter ett enkelt uttryck för f? Jo, om man ska realisera f som logisk krets vill man bygga den så billigt som möjligt, dvs med så få grundkomponenter, s.k. *grindar*, som möjligt. Typiska grindar är:



Till exempel kan $(\bar{x} + y) \cdot z$ byggas:



På disjunktiv normalform är $(\bar{x} + y) \cdot z = xyz + \bar{x}yz + \bar{x}\bar{y}z$, och att bygga upp kretsen så är naturligtvis mycket dyrare. Hur gör man då för att finna enklast möjliga uttryck för f? För stora n är detta problem *mycket svårt*, man får i praktiken nöja sig med en halvbra lösning, stora datorer till trots. För $n \leq 6$ kan man dock lösa problemet för hand, med hjälp av så kallade *Karnaugh-diagram.* Metoden illustreras för $n = 2$ i följande exempel.

x	y	f
0	0	1
0	1	0
1	0	1
1	1	1

$\Longrightarrow$

x \ y	$(\bar{y})$ 0	(y) 1
$(\bar{x})$ 0	1	0
(x) 1	1	1

$\Longrightarrow \quad f(x,y) = x + \bar{y}$

Idén är alltså att översätta sanningstabellen (eller den disjunktiva normalformen) till en 2-dimensionell tabell, och där plocka ihop block av 1:or till enkla termer. Kolonnen under $y = 0$ motsvarar ju $\bar{y} = 1$, och att den kolonnen bara innehåller 1:or motsvarar termen $\bar{y}$ i uttrycket för f. Observera att samma 1:a gärna får ingå i flera block (eftersom 1+1=1).

För 3 och 4 variabler gäller det att utforma tabellen på ett listigt sätt, så att grannrutor får klumpas ihop. Enklast ser man på ett exempel:

xy \ zw	$(\bar{z})$ 00	01	(z) 11	10
$(\bar{x})$ 00	1	0	1	0
01	0	0	1	1
(x) 11	0	0	1	1
10	1	0	1	0
	$(\bar{w})$		(w)	

(y) $(\bar{y})$

$\Longrightarrow \quad f = \bar{y}\bar{z}\bar{w} + zw + yz$

Man räknar alltså upp värdeparen i ordningen 00, 01, 11, 10. På det sättet kommer alltid två par i följd att ha en gemensam siffra; det sista paret är också granne med det första. I tabellrutan plockar man ihop block av ettor, där man alltså får utnyttja denna "wrap-around"-effekt. T.ex., de två ettorna i första kolonnen finns i kolonn $\bar{z}\bar{w}$ och i raderna $\bar{y}$, så de bidrar med termen $\bar{y}\bar{z}\bar{w}$.

Metoden med Karnaugh-diagram blir mer otymplig för $n = 5, 6$. För $n \geq 7$ bör datorer användas för att hitta minimala polynom, eller approximativt minimala polynom, till booleska funktioner. Vi går inte vidare in på detta ämne.

I komplexitetsteori studeras hur "svåra" olika beräkningsproblem är. Svårighetsgrad mäts på olika sätt, t.ex. hur många elementära steg som en algoritm för problemet måste ta som funktion av indatas storlek. I det sammanhanget spelar booleska funktioner en stor roll. Man försöker uppskatta storleken på minimala "booleska kretsar", ett allmännare begrepp än booleska polynom, för givna funktioner.

En viktig roll i teoretisk datalogi spelar också det så kallade "satisfierbarhetsproblemet" (SAT): *Givet ett booleskt polynom* $f(x_1, x_2, \ldots, x_n)$ *finns det värden* $a_i \in \{0, 1\}$ *så att* $f(a_1, a_2, \ldots, a_n) = 1$? Man vet inte om detta problem har en polynomiell lösning, dvs

om det finns en algoritm som använder högst n^d steg, för något fixt heltal d. Satisfierbarhetsproblemet för booleska polynom är det kanske viktigaste (och historiskt första, Cook 1971) exemplet på ett *NP-fullständigt* problem. Frågan om det har en polynomiell lösning är ekvivalent med det berömda "P=NP?" problemet. Vi hänvisar till referenserna, eller kurser i algoritm- och komplexitetsteori, för mer om booleska funktioners roll i teoretisk datalogi.

Referenser

[1] M.R. Garey and D.S. Johnson, Computers and intractability: A guide to the theory of NP-completeness, W.H. Freeman, San Fransisco, 1979

[2] I. Wegener, The complexity of Boolean functions, Wiley-Teubner, Stuttgart, 1987

3. ÖVNINGAR:

1. Utöver de axiom för boolesk algebra som nämndes i texten (distributivitet, dubbel negation och de Morgans lagar), så finns följande axiom i det vanligaste axiomsystemet:

(a)	$x \vee x = x \quad , \quad x \wedge x = x$	(idempotens)
(b)	$x \vee y = y \vee x \quad , \quad x \wedge y = y \wedge x$	(kommutativitet)
(c)	$x \vee (y \vee z) = (x \vee y) \vee z,$	
	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$	(associativitet)
(d)	$x \vee (x \wedge y) = x \quad , \quad x \wedge (x \vee y) = x$	(absorption)
(e)	$x \vee [y \wedge (x \vee z)] = (x \vee y) \wedge (x \vee z)$	
	$x \wedge [y \vee (x \wedge z)] = (x \wedge y) \vee (x \wedge z)$	(modularitet)
(f)	$x \vee \mathbf{0} = x \quad , \quad x \wedge \mathbf{0} = \mathbf{0}$	
	$x \vee \mathbf{1} = \mathbf{1} \quad , \quad x \wedge \mathbf{1} = x$	
(g)	$x \vee \bar{x} = \mathbf{1} \quad , \quad x \wedge \bar{x} = \mathbf{0}$	(komplementlagar)

   Verifiera att dessa axiom gäller för systemet av delmängder till en n-mängd där $\vee, \wedge$ och $^-$ betyder union, skärning och komplement.

2. Visa följande dualitetsprincip för boolesk algebra: Om P är ett sant påstående om boolesk algebra och om P' erhålls genom att byta alla $\wedge$-tecken mot $\vee$-tecken och vice versa, så är också P' ett sant påstående.

3. Bevisa följande booleska identitet:

$$(x \wedge y) \vee (y \wedge z) \vee (z \wedge x) = (x \vee y) \wedge (y \vee z) \wedge (z \vee x).$$

4. Låt $x \oplus y = \bar{x}y + x\bar{y}$. Detta motsvarar den logiska funktionen "exklusivt eller" (XOR).
   (a) Skriv upp en sanningsvärdestabell för $x \oplus y$.
   (b) Visa att $(x \oplus y) \oplus z = x \oplus (y \oplus z)$.
   (c) Skriv $x \oplus y \oplus z$ på disjunktiv normalform.

5. Vi har sett att varje boolesk funktion kan uttryckas med hjälp av de tre elementära operationerna $\{+, \cdot, \ ^- \}$; vi kallade sådana uttryck booleska polynom. Visa att följande par av elementära operationer räcker för att uttrycka alla booleska funktioner:
   (a) $\{+, \ ^-\}$
   (b) $\{\cdot, \ ^-\}$
   (c) $\{\oplus, \ \cdot\}$
   (d) Kan $x \oplus y$ uttryckas enbart med $\{+, \ \cdot\}$? n
   (e) Har dessa resultat någon betydelse för konstruktionen e
   av logiska kretsar?

6. En boolesk funktion $f(x)$ kallas *monoton* om $x \leq y \Longrightarrow f(x) \leq f(y)$ för alla $x, y \in B_n$. Visa följande:

(a) Om $f(x)$ kan uttryckas med enbart operationerna $\{+,\ \cdot\}$ så är funktionen monoton,
(b) Om $f(x)$ är monoton så kan den uttryckas enbart med operationerna $\{+,\ \cdot\}$.

7. Hitta enklast möjliga uttryck för följande booleska funktioner, och rita motsvarande logiska krets:
   (a) $\bar{x}y + x\bar{y} + xy$
   (b) $\bar{x}\bar{y}z + \bar{x}yz + \bar{x}y\bar{z} + x\bar{y}z + xyz$
   (c) $\bar{x}\bar{y}\bar{z}w + \bar{x}\bar{y}z\bar{w} + \bar{x}y\bar{z}\bar{w} + \bar{x}y\bar{z}w + xy\bar{z}\bar{w} + xyz\bar{w} + xyzw$.
8. Ett cirkulärt arrangemang av elementen i B_n så att två element i följd skiljer sig i exakt en position kallas en *Gray-kod*. Vid konstruktionen av Karnaugh-diagrammet för $n = 4$ använde vi Gray-koden 00, 01, 11, 10.
   (a) Konstruera en Gray-kod för B_3.
   (b) Visa att en Gray-kod existerar för alla B_n, $n \geq 1$.