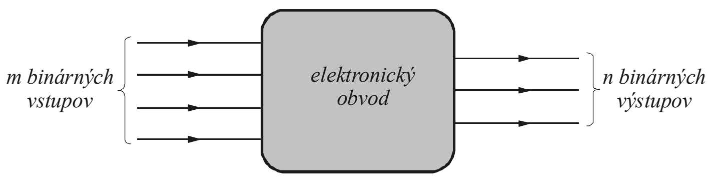
7. kapitola

Algebraické štruktúry II – Boolova algebra, Boolove funkcie, logické siete, minimalizácia Boolových výrazov.

7.1 Boolova algebra

Elektronické obvody v počítačoch a v podobných zariadeniach sú charakterizované binárnymi vstupmi a výstupmi (rovnajúcimi sa 0 alebo 1), transformácia vstupu na výstupu sa uskutočňuje prostredníctvom elektronického obvodu, ktorý tvorí jadro tohto „transformačného“ zariadenia, pozri obr. 7.1. Elektronický obvod môže byť formálne simulovaný tvz. Boolovou[1] funkciou, ktorá transformuje *m* binárnych vstupných premenných na *n* výstupných binárnych premenných.



Obrázok 7.1. Znázornenie elektronického obvodu, ktorý má *m* binárnych vstupov a *n* binárnych výstupov. Činnosť elektronického obvodu spočíva v transformácii binárnych vstupných hodnôt na binárne výstupné hodnoty.

Všeobecná definícia Boolovej funkcie je $f:\{0,1\}^m \to \{0,1\}^n$, táto funkcia transformuje binárny vektor dĺžky *m* na binárny vektor dĺžky *n*. Môžeme si položiť otázku, ako realizovať túto Boolovu funkciu, aby mala vopred špecifikované vlastnosti? Tento problém je realizovaný pomocou Boolovej algebry, ktorá pomocou premenných s 0-1 ohodnotením (t. j. binárnych) premenných a pomocou dvoch elementárnych algebraických operácií a jednej unárnej algebraickej operácie je schopná dostatočne všeobecne modelovať Boolove funkcie s vopred špecifikovanými vlastnosťami. Poznamenajme, že Boolova algebra má dva známe modely, prvým je výroková logika a druhým algebra teórie množín. Pretože obe tieto zdanlivo odťažité disciplíny majú rovnakú „metateóriu“, existuje medzi zákonmi výrokovej logiky a formulami teórie množín „dualizmus“, pomocou ktorého ku každému zákonu výrokovej logiky priradíme 1-1-značne formulu teórie množín a naopak. Dobrým, ilustratívnym príkladom tohto dualizmu sú De Morganove formuly, ktoré vo výrokovej logike a v teórii množín majú tvary

$$\neg(p\wedge q)\equiv(\neg p\vee\neg q)\Leftrightarrow \overline{A\cap B}=\overline{A}\cup\overline{B}$$

$$\neg(p\vee q)\equiv(\neg p\wedge\neg q)\Leftrightarrow \overline{A\cup B}=\overline{A}\cap\overline{B}$$

[1] George Boole (1815-1864), anglický matematik, ktorý sa svojou knihou *Laws of Thought* (1854) zaslúžil o moderný rozvoj výrokovej logiky ako špeciálnej oblasti algebry, ktorá je nazvaná jeho menom – Boolova algebra.

V týchto formulách, vo výrokovej logike unárna logická spojka negácie má ekvivalent v teórii množín v unárnej algebraickej operácii doplnku, a podobne, vo výrokovej logike binárne spojky konjunkcie a disjunkcie majú ekvivalenty v teórii množín v binárnych algebraických operáciách prieniku resp. zjednotenia. Na záver je potrebné poznamenať, že výroková spojka ekvivalentnosti má v teórii množín ekvivalent v relácii rovnosti. Tieto priradenia vo výrokovej logike a v teórii množín môžeme zosumarizovať takto

výrokové premenné $p,q,r,... \Leftrightarrow$ množiny $A,B,C,...$

spojka negácie $\neg \Leftrightarrow$ operácia doplnku $\overline{}$

spojka konjunkcie $\wedge \Leftrightarrow$ operácia prieniku $\cap$

spojka disjunkcie $\vee \Leftrightarrow$ operácia zjednotenia $\cup$

spojka ekvivalentnosti $\equiv \Leftrightarrow$ relácia rovnosti $=$

Definícia 7.1. ***Boolova algebra*** je algebraická štruktúra špecifikovaná usporiadanou 6-ticou $(B,+,\cdot,^{-},\mathbf{0},\mathbf{1})$, kde $B=\{a,b,...,x,y,...\}$ je neprázdna množina elementov (premenných Boolovej algebry), ktorá obsahuje dva špeciálne odlíšené elementy - konštanty $\mathbf{0},\mathbf{1}\in B$ a nad ktorou sú definované binárne operácie súčinu a súčtu

$$\cdot : B\times B \to B \tag{7.1a}$$

$$+ : B\times B \to B \tag{7.1b}$$

a unárna operácia komplementu

$$^{-} : B \to B \tag{7.1c}$$

ktoré vyhovujú týmto podmienkam

(1) komutatívnosť:

$$x\cdot y = y\cdot x\,, \qquad x+y=y+x \tag{7.1d}$$

(2) asociatívnosť:

$$(x\cdot y)\cdot z = x\cdot(y\cdot z), \qquad (x+y)+z=x+(y+z) \tag{7.1e}$$

(3) distributívnosť:

$$x\cdot(y+z)=(x\cdot y)+(x\cdot z), \qquad x+(y\cdot z)=(x+y)\cdot(x+z) \tag{7.1f}$$

(4) vlastnosť konštanty **0**:

$$x = x+\mathbf{0}\,, \qquad x\cdot\overline{x}=\mathbf{0} \tag{7.1g}$$

(5) vlastnosť konštanty **1** :

$$x = x\cdot\mathbf{1}\,, \qquad x+\overline{x}=\mathbf{1} \tag{7.1h}$$

V literatúre existuje mnoho alternatívnych notácií Boolovej algebry. Napríklad operácia súčinu sa alternatívne vyjadruje symbolmi $\wedge$ alebo $*$, podobne, operácia súčtu symbolmi $\vee$ a $\oplus$. Pre zjednodušenie notácie budeme vynechávať symbol súčinu, formulu $x\cdot y$ budeme zjednodušene písať ako *xy*. Z formúl (7.1g-h) vyplýva, že konštanta **1** (**0**) má úlohu neutrálneho (jednotkového) prvku pre súčin (súčet). Z bežného pohľadu na formuly (7.1g-h) by niekto mohol odvodiť záver, že výraz $\overline{x}$ je inverzná formula pre *x*. Pripomeňme si, že v kapitole 6 bol inverzný element definovaný pomocou vlastnosti $x\cdot\overline{x}=\mathbf{1}$, avšak podľa pravej formuly (7.1g) platí $x\cdot\overline{x}=\mathbf{0}$, z čoho vyplýva, že výraz $\overline{x}$ nemá vlastnosti inverzného prvku (tak vzhľadom k operácii súčtu, ako aj súčinu).

Príklad 7.1. Najjednoduchšia Boolova algebra (s veľkým významom v informatike a v logike) je založená na dvojprvkovej množine $B=\{0,1\}$. Binárne operácie súčinu, súčtu a unárna operácia komplementu sú pomocou multiplikačných tabuliek definované takto

+	0	1
0	0	1
1	1	1

$\cdot$	0	1
0	0	0
1	0	1

b	$\bar{b}$
0	1
1	0

Jednoducho sa môžeme presvedčiť, že pre takto špecifikované operácie sú splnené podmienky (7.1a-h), t. j. algebraická štruktúra $\left(B,+,\cdot,\overline{\ \ },\mathbf{0},\mathbf{1}\right)$ je Boolova algebra.

Príklad 7.2. Nech $A=\{a,b,c,...\}$ je neprázdna množina, položme $B=\mathcal{P}(A)$. Operácie $\cdot$ a $+$ sú realizované pomocou množinových operácií $\cap$ resp. $\cup$, operácia komplementu je realizovaná ako množinový komplement vzhľadom k množine A, $\overline{x}=A-x$. Potom platí:

(a) binárne operácie sú asociatívne, komutatívne,

(b) medzi binárnymi operáciami platia distributívne zákony,

(c) prázdna množina $\varnothing$ má vlastnosti jednotkového elementu pre operáciu $\cup$

$$(\forall X\in B)(X\cup\varnothing=\varnothing\cup X=X)$$

(d) množina A má vlastnosti jednotkového elementu pre operáciu $\cap$

$$(\forall X\in B)(X\cap A=A\cap X=X)$$

(e) pre každé $X\in B$ existuje komplement $\overline{X}\in B$ taký, že

$$(\forall X\in B)(X\cap\overline{X}=\varnothing)$$

$$(\forall X\in B)(X\cup\overline{X}=A)$$

To znamená, že podmienky (7.1a-h) sú splnené, t. j. algebraická štruktúra $\left(\mathcal{P}(A),\cup,\cap,\overline{\ \ },\varnothing,A\right)$ je Boolova algebra.

Príklad 7.3. Nech $B=\{p,q,r,...\}$ je množina výrokových formúl, ktorá je uzavretá vzhľadom k binárnym operácia konjunkcie ($\wedge$), disjunkcie ($\vee$) a k unárnej operácii negácie ($\neg$). Pre túto množinu je definovaná aj relácia ekvivalentnosti $'\equiv'$, dve formuly sú ekvivalentné vtedy a len vtedy, ak majú rovnakú pravdivostnú interpretáciu (logicky ekvivalentné). Z množiny B vyberieme formulu kontradikciu (napr. $p\wedge\neg p$) a označíme ju symbolom 0; podobne formula tautológia (napr. $p\vee\neg p$) je označená symbolom 1. To znamená, že symboly 0 a 1 patria do množiny B. Pre každú formulu p platia tieto vzťahy

$$p\vee 0=0\vee p=p$$

$$p\wedge 1=1\wedge p=p$$

Pretože logické spojky konjunkcie a disjunkcie sú komutatívne a asociatívne, pre tieto operácie platia taktiež distributívne zákony, podmienky z definície 7.1 sú splnené, t. j. algebraická štruktúra $(B,\vee,\wedge,\neg,0,1)$ tvorí Boolovu algebru.

7.2 Vlastnosti Boolovej algebry

V úvodnej časti kapitoly 7.1 bol zmienený princíp duality medzi algebrou teórie množín a výrokovou logikou. Ukážeme, že tento princíp je aplikovateľný aj pre rôzne Boolove algebry.

Postulujme nejaký výrok (alebo formulu), ktorý je platný v Boolovej algebre. Duálnu formu výroku dostaneme tak, že urobíme zámenu symbolov

$$\cdot \rightarrow +,\ + \rightarrow \cdot,\ \mathbf{0} \rightarrow \mathbf{1} \text{ a } \mathbf{1} \rightarrow \mathbf{0}$$

Napríklad, uvažujme formulu Boolovej algebry, $(x+y)\cdot x\cdot \overline{y}=\mathbf{0}$, duálny tvar tejto formuly je $(x\cdot y)+x+\overline{y}=\mathbf{1}$. Axiómy Boolovej algebry (7.1d-h) sú uvedené po dvojiciach duálnych formúl. To znamená, že ak v rámci Boolovej algebry odvodíme nejakú formulu, tak potom aj jej duálna forma je odvoditeľná pomocou postupu, ktorý je „duálny“ k postupu prvej formuly.

Veta 7.1 (princip duálnosti). Každá veta Boolovej algebry je taktiež vetou aj v duálnej forme.

V predchádzajúcej kaptole bola dokázaná veľmi všeobecná veta 6.1 o jednoznačnosti jednotkového (neutrálneho) elementu. Tento dôkaz bol založený na predpoklade existencie jednotkového elementu, rovnaký dôvod môže byť použitý aj pre dôkaz jednoznačnosti jednotkových elementov **1** a **0**.

Veta 7.2. Jednotkové elementy **1** a **0** existujú jednoznačne.

Podobne sa dá dokázať aj jednoznačnosť existencie inverzných elementov v Boolovej algebre.

Veta 7.3. Pre každý element $x\in B$ existuje jednoznačne element $\overline{x}\in B$ taký, že $x\cdot\overline{x}=\mathbf{0}$ a $x+\overline{x}=\mathbf{1}$ (t. j .sú splnené podmienky 7.1g-h).

Veta 7.4. Nech $(B,+,\cdot,^{-},\mathbf{0},\mathbf{1})$ je Boolova algebra, potom platia tieto formulu

(1) Involutívnosť komplementu

$$(\forall x\in B)(\overline{\overline{x}}=x) \tag{7.2a}$$

(2) Idempotentnosť

$$(\forall x\in B)(x\cdot x=x) \tag{7.2b}$$

$$(\forall x\in B)(x+x=x) \tag{7.2c}$$

(3) De Morganove zákony

$$(\forall x,y\in B)(\overline{x+y}=\overline{x}\cdot\overline{y}) \tag{7.2d}$$

$$(\forall x,y\in B)(\overline{x\cdot y}=\overline{x}+\overline{y}) \tag{7.2e}$$

(4) Nulitnosť

$$(\forall x\in B)(x+\mathbf{1}=\mathbf{1}) \tag{7.2f}$$

$$(\forall x\in B)(x\cdot\mathbf{0}=\mathbf{0}) \tag{7.2g}$$

(5) Absorpcia

$$(\forall x,y\in B)(x+(x\cdot y)=x) \tag{7.2h}$$

$$(\forall x\in B)(x\cdot(x+y)=x) \tag{7.2i}$$

(6) Komplementy konštánt

$$\overline{\mathbf{0}}=\mathbf{1} \tag{7.2j}$$

$$\overline{\mathbf{1}}=\mathbf{0} \tag{7.2k}$$

(7) Vlastnosti konštánt vzhľadom k binárnym operáciam

$$\mathbf{0}+\mathbf{0}=\mathbf{0},\mathbf{0}+\mathbf{1}=\mathbf{1},\mathbf{1}+\mathbf{0}=\mathbf{1},\mathbf{1}+\mathbf{1}=\mathbf{1} \tag{7.2l}$$

$$\mathbf{0}\cdot\mathbf{0}=\mathbf{0},\mathbf{0}\cdot\mathbf{1}=\mathbf{0},\mathbf{1}\cdot\mathbf{0}=\mathbf{0},\mathbf{1}\cdot\mathbf{1}=\mathbf{1} \tag{7.2m}$$

Dôkaz týchto vlastností prenecháme na cvičenie.

7.3 Boolove funkcie

V úvode k tejto kapitole bola Boolova funkcia definovaná ako funkcia nad binárnymi premennými $\{0,1\}$. Tento pomerne zjednodušený pohľad na Boolovu funkciu bude teraz rozšírený tak, aby koncepcia Boolovej funkcie bola časťou Boolovej algebry. Základný pojem pre definíciu Boolovej funkcie je pojem Boolovej premennej. Použijeme analogický prístup, aký sa používa pre definíciu reálnej premennej, je to veličina, ktorá môže nadobúdať hodnoty z množiny reálnych čísel.

Definícia 7.2. Nech $(B,+,\cdot,^{-},\mathbf{0},\mathbf{1})$ je Boolova algebra. Potom,
(1) ***Boolova premenná*** je taká premenná, ktorá nadobúda hodnoty z množiny *B,*
(2) ***komplement premennej*** *x*, označený $\overline{x}$, je taká premenná, ktorej hodnota sa rovná komplementu hodnoty premennej *x* (t. j. ak $x=b\in B$, potom $\overline{x}=\overline{b}\in B$,
(3) ***literál*** je Boolova premenná *x* alebo jej komplement $\overline{x}$.

V ďalšom texte budeme používať notáciu, ktorá umožní rozlíšiť literál

$$x^e=\begin{cases} x & (pre\ e=1) \\ \overline{x} & (pre\ e=0) \end{cases} \tag{7.3}$$

Podobne ako pre reálnu premennú, aj Boolova premenná môže byť kombinovaná do tvaru Boolových formúl použitím binárnych operácií súčinu, súčtu a komplementu.

Definícia 7.3. Nech $(B,+,\cdot,^{-},\mathbf{0},\mathbf{1})$ je Boolova algebra. Potom ***Boolova formula***, obsahujúca Boolove premenné $x_1,x_2,...,x_n$, je definovaná takto:
(1) konštanty **0** a **1** sú Boolove formuly,
(2) Boolove premenné $x_1,x_2,...,x_n$ sú Boolove formuly,
(3) ak *X* a *Y* sú Boolove formuly, potom aj výrazy $(X\cdot Y)$, $(X+Y)$, $\overline{X}$ a $\overline{Y}$ sú Boolove formuly.

V ďalšom texte budeme používať konvenciu, že ak bude jasné o akú formulu sa jedná, tak termín ´Boolova formula´ budeme skracovať na ´formula´. Podobne, ako vo výrokovej logike môžeme si definovať rastúcu prioritu operácií takto: (1) súčet, (2) súčin a (3) komplement. Napríklad, formulu $((x\cdot y)+z)$ môžeme pomocou tejto konvencie vyjadriť v zjednodušenom tvare bez zátvoriek $x\cdot y+z$. Konečne, podobne ako v štandardnej algebre, budeme vynechávať znak súčinu, napríklad predchádzajúci ilustračný príklad má tvar $xy+z$.

Príklad 7.4. Zjednodušte formulu $((x+y)\cdot(\overline{x}+\overline{y}))$.
Použitím distributívneho zákona a (7.1g-h)

$$((x+y)\cdot(\overline{x}+\overline{y}))=(x\cdot\overline{x})+(x\cdot\overline{y})+(y\cdot\overline{x})+(y\cdot\overline{y})=\underbrace{x\overline{x}}_{\mathbf{0}}+x\overline{y}+y\overline{x}+\underbrace{y\overline{y}}_{\mathbf{0}}=x\overline{y}+\overline{x}y$$

Definícia 7.4. Dve Boolove formule sú ***ekvivalentné*** (alebo ***rovné***) vtedy a len vtedy, ak jedna pomocou konečného počtu aplikácií axióm Booloej algebry je pretransformovaná na druhú formulu.

Podľa príkladu 7.4 formule $\varphi_1 = (x+y)\cdot(\overline{x}+\overline{y})$ a $\varphi_2 = x\overline{y}+\overline{x}y$ sú ekvivalentné, pretože druhú formulu získame z prvej použitím konečného počtu aplikácií axióm Boolovej algebry, potom $\varphi_1 = \varphi_2$.

Konečne sa dostávame k definícii Boolovej funkcie $f(x_1,x_2,...,x_n)$ ako Boolovej formuly, ktorá obsahuje premenné $x_1,x_2,...,x_n$. Napríklad

$$f(x_1,x_2,x_3) = x_1(x_2+\overline{x}_3)$$

Definícia 7.5. Nech $(B,+,\cdot,^{-},\mathbf{0},\mathbf{1})$ je Boolova algebra.

(1) ***Boolova funkcia*** $f(x_1,x_2,...,x_n)$ premenných $x_1,x_2,...,x_n$, je zobrazenie $f: B^n \to B$, pričom $f(x_1,x_2,...,x_n)$ je špecifikovaná ako Boolova formula.

(2) Všetky Boolove formule, ktoré sú navzájom ekvivalentné, definujú rovnakú funkciu.

Z tejto definície vyplýva, že ekvivaletné Boolove formuly špecifikujú rovnakú Boolovu formulu. Napríklad, máme dve funkcie

$$f: B^2 \to B \qquad f(x_1,x_2) = x_1(\overline{x}_1+x_2)$$

$$g: B^2 \to B \qquad g(x_1,x_2) = x_1x_2$$

Použitím distribučného zákona ľahko dokážeme, že formuly sú ekvivaletné, $x_1(\overline{x}_1+x_2) = x_1x_2$, potom funkcie *f* a *g* sú rovnaké.

Pretože Boolova funkcia môže byť vyjadrená mnohými rôznymi formulami, ktoré sú navzájom ekvivaletné, vzniká otázka, ako efektívne rozhodnúť, či dve Boolove formuly sú ekvivalentné, alebo či dve Boolové funkcie sú rovnaké. Ukážeme postup, ktorý nie je založený na transformácii jednej formuly na druhú, aby sme rozhodli, či funkcie sú rovnaké, ale navrhne sa „kanonická“ reprezentácia Boolovej funkcie, podľa ktorej môžeme jednoducho rozhodnúť, či dve Boolove funkcie sú rovnaké alebo nie.

Definícia 7.6. ***Súčinová klauzula*** premenných $x_1,x_2,...,x_n$ je Boolova formula, ktorá obsahuje súčin *n* literálov (t. j. premennú alebo jej komplement) pre každú premennú.

Ako príklad súčinovej klauzuly premenných x_1,x_2,x_3 sú tieto formuly: $x_1x_2x_3$, $x_1x_2\overline{x}_3$, $x_1\overline{x}_2x_3$, $\overline{x}_1x_2x_3,\ldots$, $\overline{x}_1\overline{x}_2\overline{x}_3$. Ak použijeme formalizmus x^e, potom súčinovú klauzulu premenných $x_1,x_2,...,x_n$, ktorá je špecifikovaná binárnym vektorom $\boldsymbol{e} = (e_1,e_2,...,e_n)$, má tvar

$$l_{\boldsymbol{e}} = x_1^{e_1}x_2^{e_2}...x_n^{e_n} \tag{7.4}$$

Napríklad, pre $\boldsymbol{e} = (11011)$ súčinová klauzula má tvar

$$l_{(11011)} = x_1^1x_2^1x_3^0x_4^1x_5^1 = x_1x_2\overline{x}_3x_4x_5$$

Pretože binárnych vektorov $\boldsymbol{e} = (e_1,e_2,...,e_n)$ je 2^n, potom aj ***rôznych*** súčinových klauzúl je 2^n.

Definícia 7.7. ***Súčtová klauzula*** premenných $x_1, x_2, ..., x_n$ je Boolova formula, ktorá obsahuje súčet n literálov (t. j. premennú alebo jej komplement) pre každú premennú.

Podobne ako pre súčinovú klauzulu, môžeme aj súčtovú klauzulu pre premenné $x_1, x_2, ..., x_n$ špecifikovať binárnym vektorom $\boldsymbol{e} = (e_1, e_2, ..., e_n)$

$$L_{\boldsymbol{e}} = x_1^{e_1} + x_2^{e_2} + ... + x_n^{e_n} \tag{7.5}$$

Pre $\boldsymbol{e} = (10100)$ súčtová klauzula má tvar

$$L_{\boldsymbol{e}} = x_1^1 + x_2^0 + x_3^1 + x_4^0 + x_5^0 = x_1 + \overline{x}_2 + x_3 + \overline{x}_4 + \overline{x}_5$$

Pretože každá súčtová klauzula premenných $x_1, x_2, ..., x_n$ je špecifikovaná binárnym vektorom $\boldsymbol{e} = (e_1, e_2, ..., e_n)$, potom počet súčtových klauzúl je taktiež 2^n.

Týmto sa dostávame k formulácii hlavného výsledku tejto kapitoly, že každá Boolova funkcia môže byť jednoznačne vyjadrená ako sumácia súčinových klauzúl (tento tvar sa nazýva vo výrokovej logike ***disjunktívna normálna forma***, skratka DNF).

Príklad 7.5. Vyjadrite Boolovu funkciu $x_1x_2(x_1 + x_3)$ pomocou súčtu súčinových klauzúl (DNF)

$$\begin{aligned}
x_1x_2(x_1 + x_3) &= x_1x_2x_1 + x_1x_2x_3 \\
&= \underbrace{x_1x_1}_{x_1}x_2 + x_1x_2x_3 = x_1x_2 + x_1x_2x_3 \\
&= x_1x_2\mathbf{1} + x_1x_2x_3 = x_1x_2(x_3 + \overline{x}_3) + x_1x_2x_3 \\
&= \underbrace{x_1x_2x_3 + x_1x_2x_3}_{x_1x_2x_3} + x_1x_2\overline{x}_3 \\
&= x_1x_2x_3 + x_1x_2\overline{x}_3
\end{aligned}$$

Veta 7.5. Každá Boolova funkcia $f(x_1, x_2, ..., x_n)$, ktorá sa identicky nerovná nule, môže byť špecifikovaná ako suma súčinových klauzúl

$$\begin{aligned}
f(x_1, x_2, ..., x_n) &= \sum_{\boldsymbol{e}} f(e_1, e_2, ..., e_n)\, x_1^{e_1} x_2^{e_2} ... x_n^{e_n} \\
&= \sum_{\boldsymbol{e}} f(e_1, e_2, ..., e_n)\, l_{(e_1, e_2, ..., e_n)} \\
&= \sum_{\boldsymbol{e}} f(\boldsymbol{e})\, l_{\boldsymbol{e}}
\end{aligned} \tag{7.6}$$

Presný dôkaz tejto vety vykonaný pomocou matematickej indukcie je pomerne zdĺhavý, preto ho nebudeme uvádzať. Naznačíme jednoduchý konštruktívny dôkaz. Boolova funkcia $f(x_1, x_2, ..., x_n)$ je vlastne špecifikovaná jej funkčnými hodnotami $f(e_1, e_2, ..., e_n)$ pre všetky hodnoty binárneho vektora $\boldsymbol{e} = (e_1, e_2, ..., e_n)$. Hovoríme, že funkcia f je špecifikovaná tabuľkou funkčných hodnôt, ktorá obsahuje 2^n riadkov

#	$\boldsymbol{e}=(e_1,e_2,...,e_n)$	$l_{(e_1,e_2,...,e_n)}$
1	(00........00)	**0**
2	(00........01)	**1**
....	....	....
i	$(e_1^{(i)},e_2^{(i)},...,e_n^{(i)})$	**1/0**
....	....	....
2^n	(11........11)	**0**

Súčinová klauzula $l_{(e_1,e_2,...,e_n)}\left(x_1,x_2,...,x_n\right)=x_1^{e_1}x_2^{e_2}...x_n^{e_n}$ má zaujímavú vlastnosť, jej funkčná hodnota sa rovná **1** len pre $\left(x_1,x_2,...,x_n\right)=\left(\boldsymbol{e}_1,\boldsymbol{e}_2,...,\boldsymbol{e}_n\right)$, kde $\boldsymbol{e}_i\in\{\boldsymbol{0},\boldsymbol{1}\}$, pre všetky iné prípady funkčná hodnota je **0**

$$l_{(e_1,e_2,...,e_n)}\left(x_1,x_2,...,x_n\right)=\begin{cases}\boldsymbol{1} & \left(pre\left(x_1,x_2,...,x_n\right)=\left(\boldsymbol{e}_1,\boldsymbol{e}_2,...,\boldsymbol{e}_n\right)\right)\\ \boldsymbol{0} & \left(pre\left(x_1,x_2,...,x_n\right)\neq\left(\boldsymbol{e}_1,\boldsymbol{e}_2,...,\boldsymbol{e}_n\right)\right)\end{cases} \tag{7.7}$$

To znamená, že pre konštrukciu (7.6) sú pre nás dôležité len funkčné hodnoty **1**, funkčné hodnoty **0** nie sú podstatné pre náš konštruktívny dôkaz. Zostrojíme Boolovu formulu ako sumáciu týchto klauzúl (t. j. v DNF tvare)

$$F\left(x_1,x_2,...,x_n\right)=\sum_{\boldsymbol{e}} f\left(e_1,e_2,...,e_n\right)l_{(e_1,e_2,...,e_n)} \tag{7.8}$$

Z konštrukcie tejto Boolovej funkcie, vyplýva, že jej funkčné hodnoty sú špecifikované tabuľkou funkčných hodnôt Boolovej funkcie $f\left(x_1,x_2,...,x_n\right)$. To znamená, že Boolove funkcie $f\left(x_1,x_2,...,x_n\right)$ a $F\left(x_1,x_2,...,x_n\right)$ sú ekvivalentné, t. j. majú rovnaké funkčné hodnoty pre rôzne hodnoty argumentov. Týmto sme zavŕšili jednoduchý intuitívny konštruktívny dôkaz vety 7.5.

Poznamenajme, že DNF tvar Boolovej funkcie je určený jednoznačne až na permutácie argumentov v súčtových klauzulách, alebo až na permutácie súčtových klauzúl. Táto nejednoznačnosť DNF tvaru vyplýva zo skutočnosti, že binárne operácie súčtu a súčinu sú komutatívne. Môžeme teda konštatovať, že DNF sú základné charakteristiky (niečo ako odtlačky prstov alebo zloženie DNA) Boolových funkcií. Aby sme odstránili prípadné nejednoznačnosti zapisujeme DNF v tzv. kanonickom tvare, t. j. jednotlivé argumenty sa zapisujú postupne podľa rastúceho indexu (tým sme odstránili nejednoznačnosti v dôsledku kumutatívnosti súčinu) a potom jednotlivé súčinové klauzule sú písané v poradí rastúcej číselnej hodnoty „indexu“ $\boldsymbol{e}=\left(e_1,e_2,...,e_n\right)$.

Príklad 7.6. Zostrojte Boolovu funkciu $f\left(x_1,x_2\right)=x_1+x_2$ v tvare DNF. Podľa vety 7.5 DNF tvar tejto funkcie je

$$f\left(x_1,x_2\right)=f\left(\boldsymbol{0},\boldsymbol{0}\right)\overline{x}_1\overline{x}_2+f\left(\boldsymbol{0},\boldsymbol{1}\right)\overline{x}_1x_2+f\left(\boldsymbol{1},\boldsymbol{0}\right)x_1\overline{x}_2+f\left(\boldsymbol{1},\boldsymbol{1}\right)x_1x_2$$

kde jednotlivé funkčné hodnoty sú uvedené v tabuľke

#	e_1	e_2	$f(e_1,e_2)$
1	**0**	**0**	**0**
2	**0**	**1**	**1**
3	**1**	**0**	**1**
4	**1**	**1**	**1**

Potom funkcia *f* má tvar

$$\begin{aligned} f(x_1,x_2) &= \mathbf{0}\bar{x}_1\bar{x}_2 + \mathbf{1}\bar{x}_1x_2 + \mathbf{1}x_1\bar{x}_2 + \mathbf{1}x_1x_2 \\ &= \bar{x}_1x_2 + x_1\bar{x}_2 + x_1x_2 \end{aligned}$$

Ľahko dokážeme, že takto definovaná funkcia $f(x_1,x_2) = \bar{x}_1x_2 + x_1\bar{x}_2 + x_1x_2$ je ekvivalentná s pôvodnou Boolovou funkciou $f(x_1,x_2) = x_1 + x_2$

$$f(x_1,x_2) = \bar{x}_1x_2 + x_1\bar{x}_2 + x_1x_2 = \bar{x}_1x_2 + x_1\bar{x}_2 + x_1x_2 + x_1x_2 = \Big(\underbrace{\bar{x}_1 + x_1}_{1}\Big)x_2 + x_1\Big(\underbrace{\bar{x}_2 + x_2}_{1}\Big) = x_1 + x_2$$

Príklad 7.7. Zostrojte Boolovu funkciu $f(x_1,x_2,x_3) = x_2x_3 + x_1x_3$ v tvare DNF. Táto Boolova funkcia je určená tabuľkou funkčných hodnôt

#	e_1	e_2	e_3	e_2e_3	e_1e_3	$e_2e_3+e_1e_3$
1	**0**	**0**	**0**	**0**	**0**	**0**
2	**0**	**0**	**1**	**0**	**0**	**0**
3	**0**	**1**	**0**	**0**	**0**	**0**
4	**0**	**1**	**1**	**1**	**0**	**1**
5	**1**	**0**	**0**	**0**	**0**	**0**
6	**1**	**0**	**1**	**0**	**1**	**1**
7	**1**	**1**	**0**	**0**	**0**	**0**
8	**1**	**1**	**1**	**1**	**1**	**1**

Potom funkcia $f(x_1,x_2,x_3)$ (uvažujeme len jednotkové funkčné hodnoty) má DNF tvar

$$f(x_1,x_2,x_3) = \bar{x}_1x_2x_3 + x_1\bar{x}_2x_3 + x_1x_2x_3$$

Použijeme duálny princíp z vety 7.1, veta 7.5 má potom tento duálny tvar

Veta 7.6. Každá Bolova funkcia $f(x_1,x_2,...,x_n)$, ktorá sa identicky nerovná jednotke, môže byť špecifikovaná ako súčin sumačných klauzúl

$$\begin{aligned} f(x_1,x_2,...,x_n) &= \prod_{\boldsymbol{e}} \left(f(e_1,e_2,...,e_n) + x_1^{\bar{e}_1} + x_2^{\bar{e}_2} + ... + x_n^{\bar{e}_n} \right) \\ &= \prod_{\boldsymbol{e}} \left(f(e_1,e_2,...,e_n) + L_{(\bar{e}_1,\bar{e}_2,...,\bar{e}_n)} \right) \\ &= \prod_{\boldsymbol{e}} \left(f(\boldsymbol{e}) + L_{\bar{\boldsymbol{e}}} \right) \end{aligned} \tag{7.9}$$

kde $\bar{\boldsymbol{e}} = (\bar{e}_1,\bar{e}_2,...,\bar{e}_n) = (1-e_1,1-e_2,...,1-e_n)$.

Táto veta reprezentuje hlavný duálny výsledok tejto kapitoly, že každá Boolova funkcia môže byť jednoznačne vyjadrená ako súčin súčtových klauzúl (tento tvar sa nazýva vo výrokovej logike ***konjunktívna normálna forma***, skratka KNF).

Príklad 7.8. Vyjadrite $f(x_1,x_2) = x_1(x_1 + x_2)$ v KNF tvare.

V prvom kroku zostrojíme tabuľku funkčných hodnôt tejto Boolovej funkcie

#	e_1	e_2	e_1+e_2	$e_1(e_1+e_2)$
1	**0**	**0**	**0**	**0**
2	**0**	**1**	**1**	**0**
3	**1**	**0**	**1**	**1**
4	**1**	**1**	**1**	**1**

Použitím (7.9) dostaneme vyjadrenú Boolovu funkciu $f(x_1,x_2)=x_1(x_1+x_2)$ v KNF

$$f(x_1,x_2)=\left(\underbrace{f(\mathbf{0},\mathbf{0})}_{0}+x_1+x_2\right)\cdot\left(\underbrace{f(\mathbf{0},\mathbf{1})}_{0}+x_1+\overline{x}_2\right)\cdot\underbrace{\left(\underbrace{f(\mathbf{1},\mathbf{0})}_{1}+\overline{x}_1+x_2\right)}_{1}\cdot\underbrace{\left(\underbrace{f(\mathbf{1},\mathbf{1})}_{1}+\overline{x}_1+\overline{x}_2\right)}_{1}$$
$$=(x_1+x_2)\cdot(x_1+\overline{x}_2)$$

Z tohto príkladu vyplýva, že pre konštrukciu KNF sú dôležité nulové funkčné hodnoty danej Boolovej funkcie. Táto vlastnosť je duálnu k vlastnosti DNF, kde sú relevantné jednotkové funkčné hodnoty Boolovej funkcie. Z tohto faktu vyplýva skutočnosť, že si zvolíme DNF tvar Boolovej funkcie vtedy, keď tabuľka obsahuje v prevažnej miere nulové funkčné hodnoty, KNF si zvolíme vtedy, keď tabuľka obsahuje v prevažnej miere jednotkové funkčné hodnoty. V prípade, že tabuľka obsahuje rovnaký počet nulových a jednotkových funkčných hodnôt z pohľadu „zložitosti“ konštrukcie je jedno, aký tvar Boolovej funkcie sme zvolili.

Príklad 7.9. Zostrojte KNF Boolovej funkcie $f(x_1,x_2,x_3)=(\overline{x}_1+x_2)\cdot(\overline{x}_1+\overline{x}_3)$. Tabuľka funkčných hodnôt má tvar

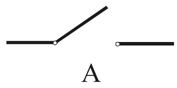
#	e_1	e_2	e_3	$\overline{e}_1$	$\overline{e}_3$	$\overline{e}_1+e_2$	$\overline{e}_1+\overline{e}_3$	$(\overline{e}_1+e_2)\cdot(\overline{e}_1+\overline{e}_3)$
1	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
2	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**1**
3	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**
4	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**
5	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**0**
6	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
7	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**1**
8	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**

KNF má potom tvar (využívame len tri riadky s nulovou výslednou funkčnou hodnotou)

$$f(x_1,x_2,x_3)=(\overline{x}_1+x_2+x_3)\cdot(\overline{x}_1+x_2+\overline{x}_3)\cdot(\overline{x}_1+\overline{x}_2+\overline{x}_3)$$

7.3 Spínacie obvody

Mnohé elektronické zariadenia, akými sú napr. počítače, telefónne ústredne, zariadenia na riadenie dopravy, obsahujú ako časť spínancie obvody. Spínač môže byť chápaný ako ako taký spoj v obvode, ktorý ak je uzavretý, potom ním prechádza elektrický, v opačnom prípade, ak je otvorený, elektrický prúd ním neprechádza. Spínač môžeme znázorniť takto:



(7.10)

Predpokladajme , že v spínacom obvode máme spínač A. Stav tohto spínača označíme premennou x, ak $x = 1$ ($x = 0$), potom spínač A je uzavretý (otvorený).

O trochu zložitejší prípad spínacieho obvodu obsahuje dva spínače A_1 a A_2

(7.11)

Hovoríme, že v tomto prípade sú spínače zapojené *sériovo*. Nech x_1 a x_2 sú premenné popisujúce stavy spínačov A_1 resp. A_2, tieto premenné ak sa rovnajú 1 (0) , potom daný spínač je uzavretý (otvorený). Nech $f(x_1, x_2)$ je funkcia, ktorej hodnota sa rovná 1 (0) pre tie hodnoty x_1 a x_2, ktoré umožňujú (znemožňujú) tok prúdu. Táto funkcia môže byť chápaná ako binárna funkcia $f:\{0,1\}^2 \to \{0,1\}$, ktorej funkčné hodnoty sú určené tabuľkou

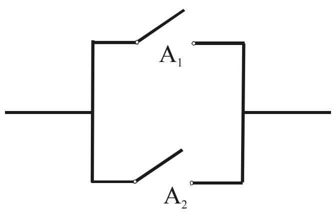
#	x_1	x_2	$f(x_1,x_2)$
1	0	0	0
2	0	1	0
3	1	0	0
4	1	1	1

Z tejto tabuľky vyplýva, že funkcia f je vyjadrená ako súčin premenných x_1 a x_2

$$f(x_1, x_2) = x_1 x_2 \tag{7.12}$$

Táto funkcia je definovaná nad Boolovou algebrou $(\{0,1\}, +, \cdot, ^{-}, 0, 1)$, t. j. premenné patria do množiny $\{0,1\}$.

Nový druh spínacieho obvodu, ktorý je podobný sériovému obvodu (7.11), ale v paralelnom zapojení, má tvar



(7.13)

Podobne ako v predchádzajúcom príklade (7.11), nech spínače A_1 a A_2 sú popísané premennými x_1 a x_2 , tento obvod je popísaný funkciou $g(x_1,x_2)$ nad Boolovou algebrou $(\{0,1\}, +, \cdot, ^{-}, 0, 1)$, ktorá je špecifikovaná tabuľkou

#	x_1	x_2	$g(x_1,x_2)$
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	1

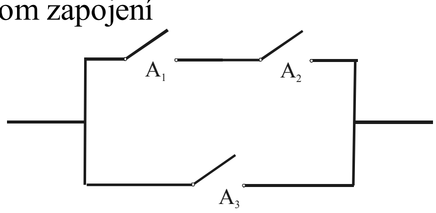
Potom funkcia $g(x_1, x_2)$ je špecifikovaná ako súčet premenných

$$g(x_1, x_2) = x_1 + x_2 \tag{7.14}$$

ktorá je definovaná nad Boolovou algebrou $(\{0,1\}, +, \cdot, ^-, 0, 1)$

Funkcie, ktoré sú podobné (7.12) a (7.14) popisujú vlastnosti spínacieho obvodu pomocou stavov spínačov, ktoré sú súčasťou daného obvodu. Takéto funkcie sa nazývajú ***spínacie funkcie***. Majme n spínačov, ktorých stavy sú špecifikované premennými x_1, x_2, ..., x_n. Spínacia funkcia $f : \{0,1\}^n \to \{0,1\}$ popisuje správanie sa spínacieho obvodu pre všetky možné 2^n stavy spínačov. Ako už bolo ukázané na predchádzajúcich ilustračných príkladoch, funkcia f môže byť reprezentovaná Boolovou formulou a teda aj Boolovou funkciou.

Nasledujúci príklad spínacieho obvodu bude zložitejší spínací obvod, ktorý obsahuje tri spoínače v sériovo-paralelnom zapojení

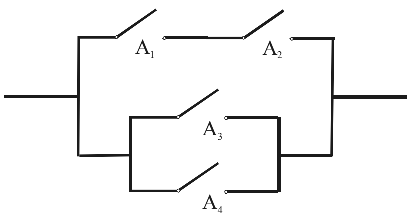


(7.15)

Nech jednotlivé spínače A$_1$, A$_2$ a A$_3$ sú špecifikované premennými x_1, x_2 resp. x_3. Nech funkcia $f(x_1, x_2)$ popisuje vlastnosti hornej časti obvodu, ktorý obsahuje dva sériovo zapojené spínače A$_1$ a A$_2$. Pomocou predchádzajúceho príkladu (7.11) funkcia, ktorá špecifikuje takýto obvod má tvar $f_1(x_1, x_2) = x_1 x_2$. Celkový obvod potom môžeme zložiť z dvoch paralelných podštruktúr, horná je reprezentovaná funkciou $f_1(x_1, x_2) = x_1 x_2$ a dolná je reprezentovaná funkciou obsahuje paralelné $f_2(x_3) = x_3$. Spojením týchto dvoch funkcií pomocou (7.14) dostaneme Boolovu funkciu celého spínacieho obvodu (7.15)

$$f(x_1, x_2, x_3) = f_1(x_1, x_2) + f_2(x_3) = x_1 x_2 + x_3 \tag{7.16}$$

Príklad 7.10. Zostrojte spínaciu funkciu f spínacieho zariadenia

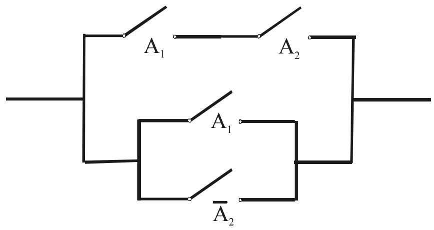


(7.17)

Nech x_1, x_2, x_3 a x_4 sú premenné označujúce stavy spínačov A_1, A_2, A_3 resp. A_4. Potom celková spínacia funkcia zariadenia má tvar $f(x_1, x_2, x_3, x_4)$. V prvom kroku určíme pomocné spínacie funkcie $f_1(x_1, x_2)$ a $f_2(x_3, x_4)$, ktoré sú priradené hornej časti obsahujúcej spínače A_1, A_2 resp. dolnej časti obsahujúcej spínače A_3, A_4. Tieto funkcie sú určené spínacími funkciami (7.12) a (7.14), $f_1(x_1, x_2) = x_1 x_2$ resp. $f_2(x_3, x_4) = x_3 + x_4$. Hľadaná spínacia funkcia má potom tvar

$$f(x_1, x_2, x_3, x_4) = f_1(x_1, x_2) + f_2(x_3, x_4) = x_1 x_2 + x_3 + x_4 \tag{7.18}$$

Príklad 7.11. Zostrojte spínaciu funkciu f zariadenia, ktoré vznikne malou modifikáciou spínacieho zariadenia (7.17)



(7.19)

kde pôvodné spínače A_1 a A_3 sú teraz spolu spriahnuté, t. j. obe sú súčasne zapnuté alebo vypnuté (preto sú obe označené A_1). Ďalšie dva pôvodné spínače A_2 a A_4 sú teraz spolu taktiež spriahnuté, ale opačným spôsobom, t. j. ak je jeden spínač zapnutý, druhý je vypnutý a naopak (preto sú oba označené A_2 a $\overline{A}_2$). Spínaciu funkciu takto špecifikovaného zariadenia ľahko zostrojíme pomocou spínacej funkcie (7.18) pôvodného spínacieho zariadenia, keď položíme $x_3 = x_1$ a $x_4 = \overline{x}_2$

$$g(x_1, x_2) = f(x_1, x_2, x_1, \overline{x}_2) = x_1 x_2 + x_1 + \overline{x}_2 \tag{7.20}$$

Tabuľka funkčných hodnôt tejto spínacej funkcie má tvar

x_1	x_2	$\overline{x}_2$	x_1x_2	$x_1x_2 + x_1 + \overline{x}_2$
0	0	1	0	1
0	1	0	0	0
1	0	1	0	1
1	1	0	1	1

Ak použime vetu 7.6, zostrojíme Boolovu funkciu v tvare KNF, ktorá simuluje túto tabuľku, z ktorej si vyberieme riadok s nulovou funkčnou hodnotou, potom $f(x_1, x_2) = x_1 + \overline{x}_2$. Spínacie zariadenie (7.19) má túto zjednodušenú spínaciu funkciu

$$g(x_1, x_2) = x_1 + \overline{x}_2 \tag{7.21}$$

Ľahko dokážeme, že funkcie (7.20) a (7.21) sú ekvivalentné

$$g(x_1, x_2) = x_1 x_2 + x_1 + \overline{x}_2 = \Bigg(\underbrace{x_2 + 1}_{1}\Bigg) x_1 + \overline{x}_2 = 1x_1 + \overline{x}_2 = x_1 + \overline{x}_2$$

Príklad 7.12. Budeme riešiť veľmi praktickú úlohu, ktorá pre mnohých z nás je záhadou ako to vlastne funguje. Predstavme si schodište, na začiatku a konci ktorého sú umiestnené stenové vypínače S_1 a S_2, pomocou ktorých zapneme alebo vypneme svetlo nad schodišťom. Hlavná požiadavka je taká, aby sa na jednom konci mohlo svetlo buď vypnúť, ak na druhom konci je zapnuté, alebo zapnúť , ak je na druhom konci vypnuté. Túto podmienku môžeme formulovať alternatívne tak, že ak sú oba spínače S_1 a S_2 vypnuté alebo zapnuté, potom zariadením nepreteká prúd, ale stačí, aby bolo zapnuté práve jedno, potom zariadením preteká prúd, čo môžeme vyjadriť touto tabuľkou

S_1	S_2	prúd
zapnuté	zapnuté	nie
zapnuté	vypnuté	áno
vypnuté	zapnuté	áno
vypnuté	vypnuté	nie

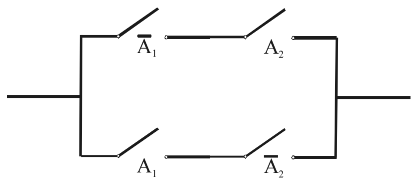
Predpokladajme, že vypínače S_1 a S_2 sú realizovaný pomocou dvoch spínačov A_1 a A_2. Ak použijeme premenné x_1 a x_2 , ktoré označujú stavy spínačov A_1 a A_2 , potom vyššie uvedená tabuľka môže byť prepísaná do tvaru

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

Podľa vety 7.5, táto tabuľka špecifikuje Boolovu funkciu v DNF

$$f\left(x_1,x_2\right)=\overline{x}_1x_2+x_1\overline{x}_2 \qquad (7.22)$$

Spínací obvod so spínacou funkciou takto špecifikovanou má tvar

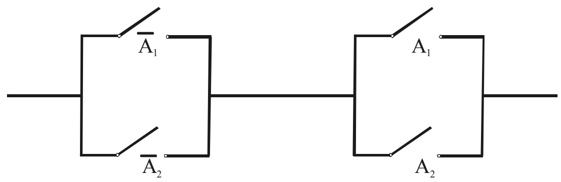


(7.23)

Ak použijeme duálnu vetu 7.6, potom KNF Boolova funkcia určená tabuľkou má tvar

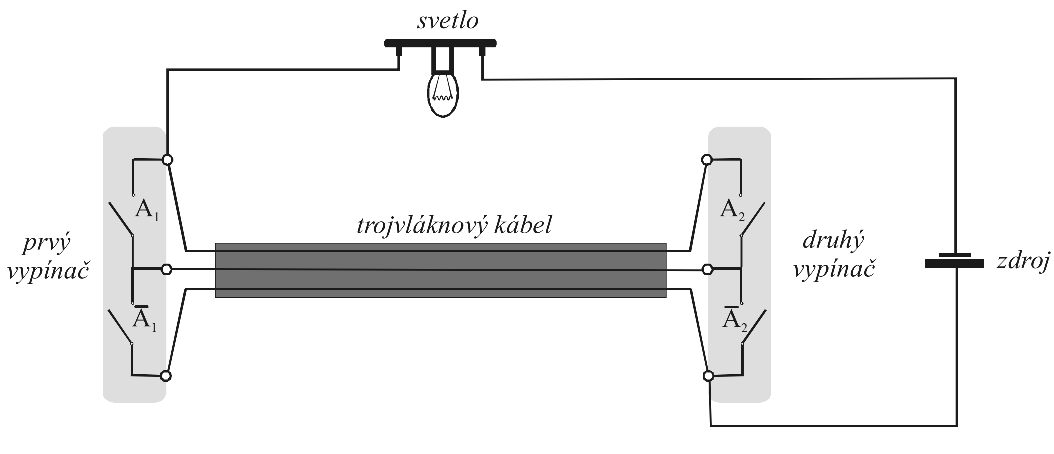
$$f\left(x_1,x_2\right)=\left(\overline{x}_1+\overline{x}_2\right)\cdot\left(x_1+x_2\right) \qquad (7.24)$$

K takto určenej spínacej funkcii môžeme zostrojiť diagramatickú reprezentáciu spínacieho obvodu v alternatívnom tvare



(7.25)

Diagram (7.26) znázorňuje realizáciu tohto spínacieho obvodu, kde vytieňované oblasti tvoria stenové vypínače, ktoré sú spojené trojvláknovým káblom.



(7.26)

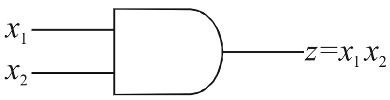
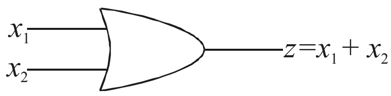
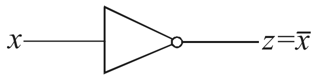
Tento jednoduchý príklad demonštruje užitočnosť teórie spínacích obvodov vybudovanej pomocou Boolových funkcií. Vyriešili sme praktický príklad, ako realizovať zapínanie a vypínanie svetla na schodišti (alebo na dlhej chodbe) tak, že každým vypínačom môžeme svetlo zapnúť alebo vypnúť, nezávisle od polohy druhého vypínača.

7.3 Logické obvody

Na tomto mieste je vhodné pripomenúť[2] si priekopnícku prácu McCullocha a Pittsa z r. 1943, v ktorej boli formulované teoretické základy neurónových sietí s logickými neurónmi a ktorá je v súčasnosti pokladaná za jednu prvých prác, na základe ktorých vznikol nový informatický vedný odbor ***umelá inteligencia***. Autori dokázali, že ľubovoľná Boolova funkcia môže byť simulovaná pomocou obvodu (neurónovej siete) obsahujúcej elementárne obvody – neuróny, ktoré simulujú disjunktívne a konjunktívne klauzuly. V prípade logických obvodov sa jedná o abstrakciu, ktorá ignoruje vnútornú architektúru neurónov a postuluje existenciu elementárnych logických brán pre operácie disjunkcie, konjunkcie a negácie.

V tejto kapitole sa budeme zaoberať logickými obvodmi, ktoré tvoria základné funkčné jednotky v počítačoch. Logické obvody obsahujú logické brány typu disjunkcie, konjunkcie a negácie, pozri tab. 7.1.

Tabuľka 7.1. Logické brány

logická brána konjunkcie	Logická brána disjunkcie	logická brána negácie
		

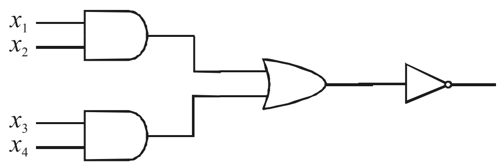
x_1	x_2	x_1x_2
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	x_1+x_2
0	0	0
0	1	1
1	0	1
1	1	1

x_1	$\overline{x}_1$
0	1
1	0

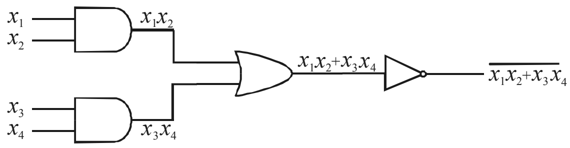
Konjunktívna a disjunktívna brána má dva binárne vstupy a jeden binárny výstup, jednoduchšia brána negácie má jeden binárny vstup a jeden binárny výstup.

Príklad 7.13. Zostrojte Boolovu funkciu pre logický obvod



Použitím tabuľky 7.1 jednotlivé spoje tohto logického obvodu ohodnotíme takto

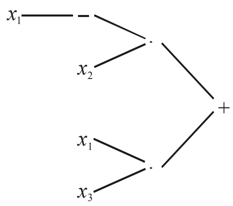
[2] Pozri náš učebný text *Matematická logika*, kapitola 4.



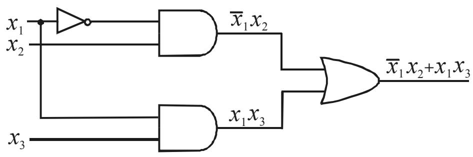
To znamená, že Boolova funkcia priradená tomuto obvodu má tvar

$$f\left(x_1,x_2,x_3,x_4\right)=\overline{x_1x_2+x_3x_4}$$

Príklad 7.14. Zostrojte logický obvod, ktorý simuluje Boolovu funkciu $f\left(x_1,x_2,x_3,x_4\right)=\overline{x}_1x_2+x_1x_3$. Pre ilustráciu zostrojíme syntaktický strom tejto funkcie



Logický obvod zostrojíme jednoducho z tohto syntaktického stromu, ktorý interpretuje Boolovu funkciu $f\left(x_1,x_2,x_3,x_4\right)=\overline{x}_1x_2+x_1x_3$, tak, že jednotlivé vrcholy znázorňujúce algebraické operácie nahradíme príslušnými logickými bránami



Sumátor binárnych čísel

Technika logických obvodov je aplikovateľná ku konštrukcii súčtu dvoch kladných celých čísel, ktoré sú prezentované v binárnej reprezentácií. Táto konštrukciu obsahuje tri etapy. *Prvá etapa* spočíva v návrhu logického obvodu (nazývaného *polosumátor*) ktorý sčíta dve jednobitové čísla x a y

$$\begin{array}{r} x \\ y \\ \hline c\ \ s \end{array} \tag{7.27}$$

Uvedieme tabuľku všetkých prípadov tejto schémy, ktoré môžu nastať

vstup		výstup	
x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

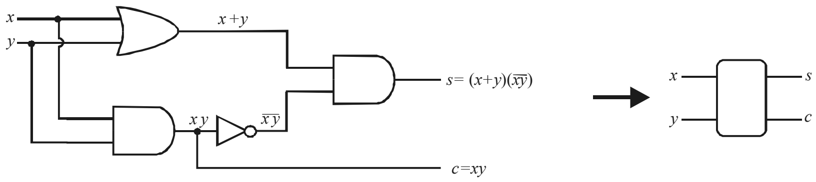
(7.28)

Ak použijeme vetu 7.5 výstupné premenné z tejto tabuľky sú určené pomocou sumácie klauzúl (v DNF tvare)

$$s=f\left(x,y\right)=\overline{x}y+x\overline{y}=\left(x+y\right)\left(\overline{x}+\overline{y}\right)=\left(x+y\right)\left(\overline{xy}\right) \tag{7.29a}$$

$$c=g\left(x,y\right)=xy \tag{7.29b}$$

kde pri konštrukcii alternatívnej pravej strany Boolovej funkcie f bol použitý distributívny zákon.



(7.30)

Druhá etapa konštrukcie sumátora spočíva v konštrukcii logického obvodu (nazývaného *dvojitý sumátor*) pre sčítanie troch jednobitových čísel

$$\begin{array}{r} x \\ y \\ \underline{z} \\ u\ v \end{array} \tag{7.31}$$

Všetky možné prípady tejto schémy sú uvedené v tabuľke

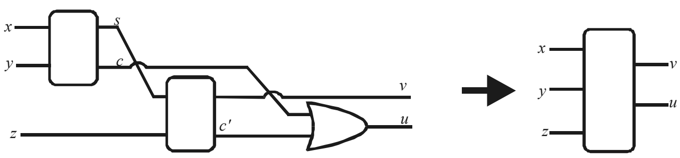
vstup			výstup	
x	y	z	u	v
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

(7.32)

Analýzou tejto tabuľky by sme zostrojili DNF Boolove funkcie pre výstupy u a v tieto by sme sa snažili vyjadriť pomocou pomocných funkcií (7.29) z polovičného sumátora, takto by sme dospeli k obvodu, ktorý simuluje sumáciu (7.31) pomocou polosumátora. Tento program, napriek jeho priamočiarosti, nie je algebraicky jednoduchý, preto nahradíme intuitívnym prístupom, ktorý je založený na prepise (7.31) pomocou dvoch medzikrokov

$$\begin{array}{r} x \\ \underline{y} \\ c\ s \\ \underline{z} \\ u\ v \end{array} \tag{7.33}$$

kde bitové komponenty medzivýsledku s c sú spočítané pomocou polosumátora, druhá sumácia je taktiež realizovaná pomocou polosumátora, avšak pri konštrukcii bitovej komponenty u musím zahrnúť aj bitový medzivýsledok c' (vzniká pri výpočte v ako druhá výstupná komponenta polosumátora), ktorý použijeme pre výpočet výstupnej bitovej komponenty u ako súčtu s a c' (poznamenajme, že z podstaty študovaného problému vyplýva, že s a c' súčasne obsahujú maximálne len jednu jednotkovú komponentu, takže pri ich súčte nevzniká prenos bitu do ďalšej pozície). Tento heuristický pohľad na uskutočnenie sumácie troch jednobitových čísel (1.31) pomocou jeho prepisu do tvaru (7.33) môže byť jednoducho vyjadrený týmto logickým obvodom, ktorý už využíva blok pre polosumátor špecifikovaný v (7.30)



(7.34)

Záverečná tretia etapa využije dvojitý sumátor (ako blok) k realizácii sčítanie dvoch bitový čísel ľubovolnej dľžky *n*. Ako ilustračný príklad študujme sčítane dvoch trojbitových čísel

$$\begin{array}{r} x_0\ x_1\ x_2 \\ \underline{y_0\ y_1\ y_2} \\ u_0\ u_1\ u_2\ u_3 \end{array} \tag{7.35}$$

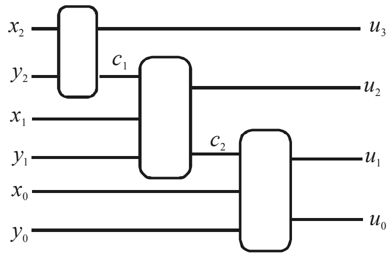
kde idúc postupne, z pravej do ľavej strany, uskutočňujeme sumáciu pomocou blokov polosumátora (*PS*) a dvojitéto sumátora (*DS*)

$$(u_3, c_1) = PS(x_2, y_2) \tag{7.36a}$$

$$(u_2, c_2) = DS(x_1, y_1, c_1) \tag{7.36b}$$

$$(u_0, u_1) = DS(x_0, y_0, c_2) \tag{7.36c}$$

Túto postupnosť príkazov môžeme diagramaticky reprezentovať ako logický obvod s polosumátorom a s dvoma dvojitými sumátormi



(7.37)

Pomocou blokov pre polosumátor a dvojitý sumátor môžeme zostrojiť logický obvod pre sumáciu dvoch digitálnych čísel ľubovolnej dĺžky.

7.4 Optimalizácia logických obvodov

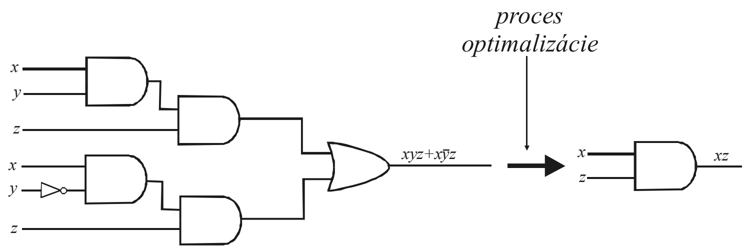
Efektívnosť logických obvodov závisí na počte a prepojení jej logických brán. Proces návrhu logického obvodu začína návrhom tabuľky špecifikujúcej Boolovu funkciu, ktorá transfomuje vstupné binárne veličiny na výstupné binárne veličiny. Boolova funkcia je zostrojená podľa vety 7.5 pomocou súčtu konjunktívnych klauzúl. Aj keď je tento prístup ku konštrukcii Boolovej funkcie všeobecný, veta 7.5 nezaručuje optimálnosť zostrojenej funkcie. Pod optimálnosťou rozumieme to, že Boolova funkcia, ktorá simuluje danú tabuľku funkčných hodnôt pre všetky kombinácie vstupných hodnôt, má minimálny počet literálov.

Uvažujme logický obvod, ktorý má výstup 1 vtedy a len vtedy, ak $x = y = z = 1$ alebo ak $x = z = 1$ a $y = 0$. Boolova funkcia zostrojená podľa vety 7.5, ktorá simuluje tento logický obvod má tvar $f(x,y,z) = xyz + z\overline{y}z$, môže byť podstatne zjednodušená takto

$$f(x,y,z) = xyz + x\overline{y}z = x(y+\overline{y})z = x \cdot 1 \cdot z = xz$$

To znamená, že táto funkcia xz, ktorá obsahuje dva literály, je ekvivalentná s pôvodnou funkciou obsahujúcej 6 literálov; môžeme teda povedať, že optimálny tvar Boolovej funkcie $f(x,y,z) = xyz + z\overline{y}z$ je nová funkcia $g(x,z) = xz$, ktorá aj keď je s pôvodnou ekvivalentná, má podstatne menej literálov.

Tento jednoduchý príklad dostatočne jasne ukazuje dôležitosť hľadať optimálny tvar Boolovej funkcie, pôvodne zostrojene podľa vety 7.5, ak má táto slúžiť ako podklad pre návrh logického obvodu. Optimálny tvar Boolovej funkcie môže v špeciálnych prípadoch podstatné zjednodušiť navrhovaný logický obvodu. Ako ilustračný príklad budeme študovať logické obvody priradené Boolovej funkcii $f(x,y,z)=xyz+z\bar{y}z$ a jej optimálnej formy $g(x,z)=xz$



– (7.38)

Logický obvod zostrojený pomocou optimálneho tvaru (vpravo), obsahujúceho minimálny počet literálov je podstatne jednoduchší ako logický obvod (vľavo), ktorý obsahuje šesť logických hradiel.

Quinova[3] a McCluskeyho[4] metóda

Táto metóda patrí medzi moderné prístupy k optimalizácii Boolových funkcií, jej hlavnou prednosťou pred ostatnými klasickými prístupmi k tomuto optimalizačnému problému (napr. známej metóde Karnaughových máp) je jej konceptuálna jednoduchosť a priamočiara algoritmizovateľnosť.

Quinovu a McCluskeyho metóda bude ilustrovaná konkrétnym prípadom optimalizácie jednoduchej Boolovej funkcie

$$f(x,y,z)=xyz+x\bar{y}z+\bar{x}yz+\bar{x}\bar{y}z+\bar{x}\bar{y}\,\bar{z} \qquad (7.39)$$

Každá súčinová klauzula (pozri definícia 7.6 a formulu (7.3)) môže byť reprezentovaná bitovým reťazcom $\boldsymbol{e}=(e_1,e_2,e_3)\in\{0,1\}^3$

$$l_{\boldsymbol{e}}(x,y,z)=x^{e_1}y^{e_2}z^{e_3}\rightarrow(e_1,e_2,e_3) \qquad (7.40)$$

kde $\xi^e=\xi$, ak $e=1$, $\xi^e=\bar{\xi}$, ak $e=0$, pre $\xi=x,y,z$. Pre takto definovanú binárnu reprezentáciu môžeme použiť metriku Hammingovej vzdialenosti ku kvantifikácii podobnosti medzi binárnymi vektormi. Nech $\boldsymbol{e}_i=\left(e_1^{(i)},e_2^{(i)},...,e_n^{(i)}\right)$ a $\boldsymbol{e}_j=\left(e_1^{(j)},e_2^{(j)},...,e_n^{(j)}\right)$ sú dve binárne reprezentácie klauzúl, potom

$$d_H(\boldsymbol{e}_i,\boldsymbol{e}_j)=\sum_{k=1}^{n}\left|e_k^{(i)}-e_k^{(j)}\right| \qquad (7.41)$$

Táto vzdialenosť pre binárne vektory nám špecifikuje počet polôh v ktorých sa binárne vektory vzájomne odlišujú. Napríklad, ak Hammingova vzdialenosť medzi dvoma binárnymi vektormi je 2, potom tieto vektory sa navzájom odlišujú na dvoch miestach binárneho reťazca.

---

[3] Willard Van Orman Quine (1908-2000), americký logik a filozof. Patril medzi najvplyvnejších filozofov 20. storočia. Jeho fundamentálne príspevky k teórii vedomostí, matematickej logike, teórii množín a k filozofii logiky a jazyka sú stále aktuálne a zdrojom inšpirácie pre filozofov, logikov a matematikov. Jeho kniha *New Foundations of Mathematical Logic* (1937) zásadným spôsobom ovplyvnila rozvoj matematickej logiky.

[4] Edward J. McCluskey (1929), jedna z vedúcich osobností súčasnej americkej počítačovej vedy. Zaslúžil sa o oddelenie počítačovej vedy od elektrotechniky ako samostatnej disciplíny, ktorá má svoj špecifický predmet výskumu – počítače. Jeho vedecké aktivity sú veľmi rozsiahle, od počítačových architektúr, testovanie a návrh logických obvodov.

Každá Boolova formula v DNF forme je 1-1-značne reprezentovaná pomocou množiny bitových reťazcov, ktoré sú priradené jednotlivým klauzulám študovanej formuly. Pre ilustračný príklad (7.39) táto množina obsahuje 5 binárnych reťazcov dĺžky 3

$$U_f = \{(111),(101),(011),(001),(000)\} \tag{7.42}$$

Táto možnosť reprezentovať Boolovu funkciu v DNF forme pomocou množiny binárnych vektorov vyplýva zo skutočnosti, že operácia sumácie je komutatívna a asociatívna, čiže nezáleží v akom poradí sčítame jednotlivé klauzuly v Boolovej funkcii.

Dve klauzuly z množiny U_f môžu byť vzájomne sčítané do jednej klauzuly vtedy a len vtedy ak sa líšia ich binárne reťazce práve v jednej polohe, čiže ak ich vzájomná Hammingova vzdialenosť sa rovná 1. Napríklad, z množiny (7.42) vyberieme prvú a druhú klauzulu, ich binárne reprezentácie (111) a (101) sa líšia len hodnotou binárnej premennej v druhej polohe, $d_H(111,101)=1$. Tieto dve klauzuly sú sčítané takto

$$xyz + x\overline{y}z = x\left(\underbrace{y+\overline{y}}_{1}\right)z = xz \tag{7.43a}$$

V binárnej reprezentácii tento proces zjednodušenia formálne vyjadríme takto

$$(111)+(101) = sum\left((111),(101)\right) = (1\#1) \tag{7.43b}$$

kde bol použitý nový „*prázdny*“ symbol ´#´, ktorý reprezentuje prázdne miesto v binárnej reprezentácii novej klauzuly *xz*, pozri (7.43a). Takto zostrojené nové klauzuly obsahujúce jeden symbol ´#´ tvoria množinu

$$U_f^{(1)} = \{(1\#1),(\#11),(\#01),(0\#1),(00\#)\} \tag{7.44}$$

V ďalšej etape vytvárame z množiny $U_f^{(1)}$ novú množinu $U_f^{(2)}$, ktorá obsahuje klauzuly s dvoma prázdnymi symbolmi ´#´ a ktoré boli vytvorené operáciou súčtu klauzúl z množiny $U_f^{(1)}$

$$U_f^{(2)} = \{(\#\#1)\} \tag{7.45}$$

Proces sčítania klauzúl obsahujúcich symboly ´#´ musí byť podrobnejšie špecifikovaný:

(a) Sčítať môžeme len také dve klauzuly, ktoré obsahujú rovnaký počet symbolov ´#´, pričom tieto symboly v oboch použitých klauzulách musia byť umiestnené v rovnakých polohách v oboch binárnych reprezentáciách.

(b) Klauzuly, ktoré vyhovujú podmienke (1) môžeme sčítať len vtedy, ak ich binárne komponenty sa líšia len v jednej polohe.

Uvažujme dve klauzuly $\boldsymbol{e}_i = \left(e_1^{(i)}, e_2^{(i)}, ..., e_n^{(i)}\right)$ a $\boldsymbol{e}_j = \left(e_1^{(j)}, e_2^{(j)}, ..., e_n^{(j)}\right)$, ktoré chceme sčítať. Podľa podmienky (a) musí existovať taká množina indexov $I \subseteq \{1,2,...,n\}$, že

$$(\forall k \in I)\left(e_k^{(i)} = e_k^{(j)} = \#\right) \text{ a } (\forall k \notin I)\left(e_k^{(i)}, e_k^{(j)} \in \{0,1\}\right) \tag{7.46}$$

Ako príklad uvedieme dvojicu klauzúl $\boldsymbol{e}_i = (10\#0\#11)$ a $\boldsymbol{e}_j = (11\#0\#10)$, pre množinu indexov $I = \{3,5\}$ platia obe podmienky. V prípade, že takáto množina neexistuje, potom klauzuly $\boldsymbol{e}_i = \left(e_1^{(i)}, e_2^{(i)}, ..., e_n^{(i)}\right)$ a $\boldsymbol{e}_j = \left(e_1^{(j)}, e_2^{(j)}, ..., e_n^{(j)}\right)$ nemôžu byť použité v procese sčítania klauzúl. Zavedieme zovšeobecnenú Hammingovu vzdialenosť pre klauzuly, ktoré vyhovujú podmienke (a)

$$d_H\left(\boldsymbol{e}_i,\boldsymbol{e}_j\right)=\sum_{k\notin I}\left|e_k^{(i)}-e_k^{(j)}\right| \tag{7.47}$$

t. j. v sumácii sú aktívne len binárne členy. Potom podmienka (b) požaduje, že Hammingova vzdialenosť medzi klauzulami musí byť rovná 1. Bez zníženia všeobecnosti môžeme postulovať, že Hammingova vzdialenosť medzi vektormi $\boldsymbol{e}_i=\left(e_1^{(i)},e_2^{(i)},\ldots,e_n^{(i)}\right)$ a $\boldsymbol{e}_j=\left(e_1^{(j)},e_2^{(j)},\ldots,e_n^{(j)}\right)$, ktoré nevyhovujú podmienke (a) (majú buď rôzny počet symbolov symbol ′#′, alebo ak majú rovnaký počet týchto symbolov, potom ich polohy nie sú rovnaké) je nekonečne veľká. Týmto jednoduchým predpokladom máme zabezpečené, že podmienka (a) je splnená.

Zavedieme operátor $\mathcal{A}$, ktorý špecifikuje prechod množiny $U_f^{(k)}$ na množinu $U_f^{(k+1)}$

$$U_f^{(k+1)}=\mathcal{A}\left(U_f^{(k)}\right) \tag{7.48}$$

Musí existovať také kladné celé číslo n, že tento proces tvorby nových množín je ukončený, t. j. platí $U_f^{(n+1)}=\mathcal{A}\left(U_f^{(n)}\right)=\emptyset$. Rekurentný proces (7.48) je inicializovaný množinou $U_f^{(0)}=U_f$. Môžeme hovoriť o etapách procesu tvorby nových klauzúl z pôvodnej (počiatočnej) množiny klauzúl. V 1. etape vytvoríme procesom sčítania dvoch klauzúl z množiny $U_f^{(0)}=U_f$ klauzuly s jedným prázdnym symbolom ′#′, v 2. etape vytvoríme z množiny $U_f^{(1)}$ klauzuly s dvoma symbolmi #. Tento rekuretný proces je ukončený vtedy, ak operátor $\mathcal{A}$ aplikovaný na množinu $U_f^{(n)}$ produkuje prázdnu množinu, t. j. $\mathcal{A}\left(U_f^{(n)}\right)=\emptyset$.

Ako ilustračný príklad rekurentnej tvorby množín $U_f^{(k)}$ použijeme množinu (7.42), výsledky je možné uviesť vo forme tabuľky

0. etapa			1. etapa			2. etapa		
1	(111)		1	(1,2)	(1#1)	1	(1,4), (2,3)	(##1)
2	(101)		2	(1,3)	(#11)			
3	(011)		3	(2,4)	(#01)			
4	(001)		4	(3,4)	(0#1)			
5	(000)		5	(4,5)	(00#)			

V stĺpcoch pre prvú a druhú etapu sú uvedené aj dvojice indexov klauzúl z predchádzajúceho stĺpca, ktoré boli použité v sumačnom procese.

Stojíme pred úlohou, ako vybrať taký minimálny počet klauzúl zostrojených v prvej alebo v ďalších etapách, ktoré sú odvoditeľné zo všetkých pôvodných klauzúl z množiny $U_f^{(0)}=U_f$. K presnej špecifikácii tohto posledného kroku Quinovej a McCluskyeho metódy musíme zaviesť nový pojem „pokrytie“. Klauzula ***e′ pokrýva*** klauzulu $\boldsymbol{e}$, formálne $\boldsymbol{e}'\subseteq\boldsymbol{e}$, vtedy a len vtedy, ak platí pre každé i=1,2,...,n práve jedna z týchto podmienok

$$\begin{aligned}&1.\ \left(e_i'=1\right)\Rightarrow\left(e_i=1\right)\\&2.\ \left(e_i'=0\right)\Rightarrow\left(e_i=0\right)\\&3.\ \left(e_i'=\#\right)\Rightarrow\left(e_i\in\{0,1,\#\}\right)\end{aligned} \tag{7.49}$$

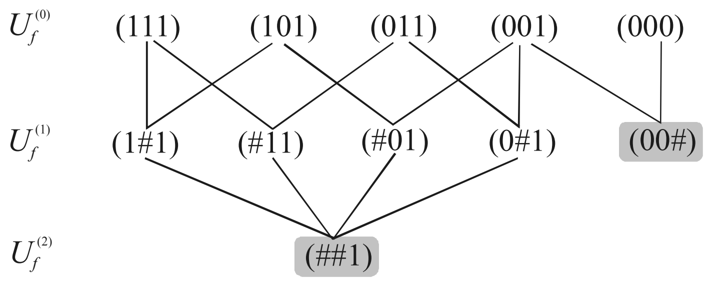
Pre ilustráciu tejto relácie uvedieme jednoduchý ilustračný príklad ktorý vyhovuje týmto podmienkam

0	1	#	1	#	0	1	0	1	***e***
0	1	#	1	#	#	1	0	#	***e′***

Ľahko sa presvedčíme, že sa jedná o ***reláciu čiastočného usporiadania*** nad množinou vektorov $\boldsymbol{e} \in \{0,1,\#\}^n$ (pozri kapitolu 3.2). Potom množina klauzúl, ktorá vznikne zjednotením množín $U_f^{(0)}$, $U_f^{(1)}$, $U_f^{(3)}$,...

$$\tilde{U}_f = U_f^{(0)} \cup U_f^{(1)} \cup U_f^{(2)} \cup$$

je čiastočne usporiadaná a diagramatický znázornená Hasseho diagramom



Z tohto diagramu vyplýva, že má 5 maximálnych klauzúl (klauzuly patriace do množiny $U_f^{(0)}$) a dve minimálne klauzuly (##1) a (00#), ktoré sú na Hasseho diagrame vysvietené.

Pre každú klauzulu $\boldsymbol{e}$ obsahujúcu aspoň jeden prázdny symbol, $e \in U_f^{(1)} \cup U_f^{(2)} \cup \ldots$, zostrojíme množinu $U(\boldsymbol{e}) \subseteq U_f$, ktorá obsahuje všetky pôvodné klauzuly (neobsahujúce prázdne symboly #), ktoré sú pokryté klauzulou $\boldsymbol{e}$

$$U(\boldsymbol{e}) = \left\{\boldsymbol{e}' ; (\boldsymbol{e}' \in U_f) \wedge (\boldsymbol{e} \subseteq \boldsymbol{e}')\right\} \tag{7.50}$$

Pre lepšie pochopenie tejto množinovej koncepcie uvedieme príklady tejto množiny vychádzajúce z vyššie uvedeného Hassovho diagramu

$$U(1\#1) = \{(111),(101)\}$$
$$U(\#11) = \{(111),(011)\}$$
$$U(\#01) = \{(101),(001)\}$$
$$U(0\#1) = \{(011),(001)\}$$
$$\boxed{U(00\#)} = \{(001),(000)\}$$
$$\boxed{U(\#\#1)} = \{(1,1,1),(101),(011),(001)\}$$

kde posedné dve klauzuly sú označené ako minimálne

Našim cieľom je vybrať také minimálne klauzuly, ktoré pokrývajú pôvodné klauzuly z množiny $U_f^{(0)}$. Množinu týchto minimálnych klauzúl označíme V, potom v rámci tejto množiny hľadáme takú podmnožinu $\widetilde{V} \subseteq V$, ktorej klauzuly plne pokrývajú množinu $U_f^{(0)}$

$$\bigcup_{e \in \widetilde{V}} U(e) = U_f \tag{7.51}$$

Podmnožina $\widetilde{V}$ je určená podmienkou minimálnosti počtu literálov, $[\widetilde{V}]$, ktoré obsahuje

$$\widetilde{V} = arg \min_{V' \subseteq V} [V'] \tag{7.52}$$

Riešenie tohto optimalizačného problému je pre malý počet premenných (cca do päť) obvykle zvládnuteľný ručne tak, že preberieme všetky možnosti, ktoré pokrývajú množinu U_f. Pre väčšie problémy môže byť použitá metóda spätného prehľadávania, ktorá systematicky preskúma všetky možnosti. Žiaľ, tento prístup je nepoužiteľný pre niekoľko desiatok premenných, v dôsledku exponenciálneho rastu zložitosti. Potom nastupujú evolučné metódy, ktoré poskytujú v reálnom čase kvalitné suboptimálne riešenie, ktoré je často rovné optimálnemu riešeniu.

V tomto konkrétnom prípade sa jedná o jednoduchý problém, musíme vybrať obe minimálne klauzuly (##1) a (00#), ktoré pokrývajú pôvodné klauzuly. Potom môžeme písať Boolovu funkciu (7.39) v ekvivalentnom tvare

$$f\left(x,y,z\right)=z+\overline{x}\,\overline{y}$$

čo reprezentuje podstatné zjednodušenie (optimalizáciu) pôvodnej Boolovej funkcie (7.39), ktorej počet literálov z 15 klesol na 3.

Príklad 7.15. Quinova a McCluskeyho metóda pôvodne nevyužívala algoritmus spätného prehľadávanie, bola založená na heuristickom prístupe naformulovanom v podobe tabuliek. Uvažujme Boolovu funkciu

$$f\left(w,x,y,z\right)=wxy\overline{z}+w\overline{x}yz+\overline{w}xyz+w\overline{x}y\overline{z}+\overline{w}x\overline{y}z+\overline{w}\overline{x}yz+\overline{w}\,\overline{x}\,\overline{y}z$$

V nasledujúcej tabuľke je znázornený postup vytvárania všetkých možných sumácií medzi klauzulami (v binárnej reprezentácii) k tejto Boolovej funkcie

0. etapa			1. etapa				2. etapa		
1	(1110)		1	(1,4)	(1#10)		1	(4,7),(5,6)	(0##1)
2	(1011)		2	(2,4)	(101#)				
3	(0111)		3	(2,6)	(#011)				
4	(1010)		4	(3,5)	(01#1)				
5	(0101)		5	(3,6)	(0#11)				
6	(0011)		6	(5,7)	(0#01)				
7	(0001)		7	(6,7)	(00#1)				

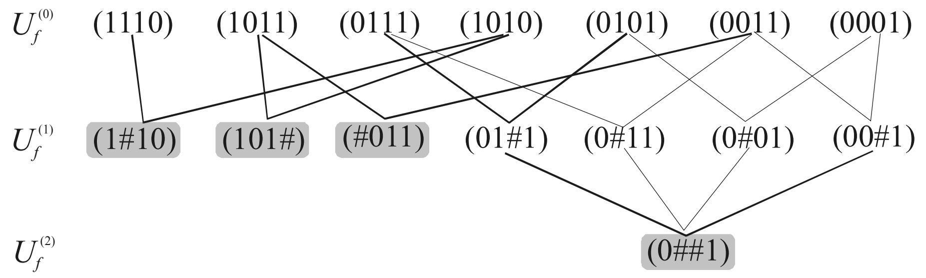
V 0. etape máme zoznam všetkých klauzúl z Boolovej funkcie, množina $U_f^{(0)}$ má tvar

$$U_f^{(0)}=\left\{(1110),(1011),(0111),(1010),(0101),(0011),(0001)\right\}$$

V 1. etape vytvoríme sumáciami nové klauzuly, ktoré obsahujú jeden prázdny znak # (tieto klauzuly sú uvedené v šiestom stĺpci vyššie uvedenej tabuľky

$$U_f^{(1)}=\mathcal{A}\left(U_f^{(0)}\right)=\left\{(1\#10),(101\#),(\#011),(01\#1),\left(0\#11\right),(0\#01),(00\#1)\right\}$$

V 2. etape vytvoríme množinu kroku množina $U_f^{(2)}=\mathcal{A}\left(U_f^{(1)}\right)=\left\{\left(0\#\#1\right)\right\}$ obsahuje už len jednu klauzulu (0##1). Hasseho diagram má tvar



Teraz stojíme pred problémom ako vybrať taký minimálny počet klauzúl, ktoré nám budú pokrývať celú pôvodnú množinu klauzúl. Existujú dve alternatívne možnosti výberu minimálnych klauzúl, ktoré plne pokrývajú pôvodné klauzuly

$$U_f^{(2)} = \{(0\#\#1),(1\#10),(101\#)\}$$
$$U_f^{(2')} = \{(0\#\#1),(1\#10),(\#011)\}$$

Každá z týchto množín je ohodnotená 8 (obsahuje 8 binárnych symbolov) a taktiež 3 komplementy (tri symboly 0). Potom, získané alternatívne riešenia sú rovnocenné. Ekvivalentné Boolove funkcie, ktoré sú určené týmito klauzulami majú alternatívne formy

$$f_1(w,x,y,z) = \overline{w}z + wy\overline{z} + w\overline{x}y$$
$$f_2(w,x,y,z) = \overline{w}z + wy\overline{z} + \overline{x}yz$$

Cvičenia

Cvičenie 7.1. Aká je hodnota Boolovej premennej, ktorá je určená podmienkou
(a) $x \cdot \mathbf{1} = \mathbf{0}$,
(b) $x + x = \mathbf{0}$,
(c) $x \cdot \mathbf{1} = x$,
(d) $x + \overline{x} = \mathbf{1}$.
(e) $x \cdot \overline{x} = \mathbf{0}$

Cvičenie 7.2. Zostrojte tabuľku funkčných hodnôt Boolovej funkcie
(a) $f(x,y,z) = \overline{x}y$,
(b) $f(x,y,z) = x + yz$,
(c) $f(x,y,z) = x\overline{y} + \overline{xyz}$,

Cvičenie 7.3. Znázornite Boolove funkcie $f(x,y,z)$ z cvičenia 7.2 na 3-rozmernej kocke tak, že hodnoty 1 (0) budú reprezentované na kocke čiernym (bielym) bodom.

Cvičenie 7.4. Pre ktoré hodnoty x a y platí $xy = x + y$.

Cvičenie 7.5. Zostrojte tabuľku všetkých možných binárnych Boolových funkcií a identifikujte v nej známe Boolove binárne operácie súčinu a súčtu. Vyjadrite ostatné binárne operácie pomocou súčtu, súčinu a komplementu.

Cvičenie 7.6. Niekedy je výhodné v Boolovej algebre definovať novú binárnu operáciu označenú symbolom ⊕, jej tabuľka funkčných hodnôt má tvar

⊕	**0**	**1**
0	**0**	**1**
1	**1**	**0**

Poznamenajme, že vo výrokovej logike, podobná logická spojka je označovaná „exkluzívna disjunkcia" (XOR). Zjednodušte tieto výrazy
(a) $x \oplus \mathbf{0}$,
(b) $x \oplus \mathbf{1}$,
(c) $x \oplus x$,
(d) $x \oplus \overline{x}$.

Cvičenie 7.7. Dokážte, že platia rovnosti
(a) $x \oplus y = (x + y)(\overline{xy})$,
(b) $x \oplus y = \overline{x}y + x\overline{y}$.

Cvičenie 7.8. Zostrojte duálne výrazy k týmto Boolovym funkciam
(a) $x+y$,
(b) $\overline{x}\,\overline{y}$,
(c) $xyz+\overline{x}\,\overline{y}\,\overline{z}$.

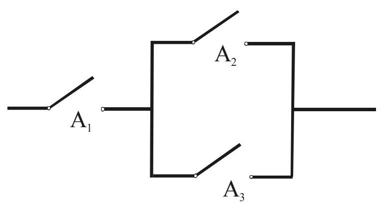
Cvičenie 7.9. Dokážte, že duálny tvar $f_d\left(x_1,x_2,...,x_n\right)$ k Boolovej funkcii $f\left(x_1,x_2,...,x_n\right)$ vyhovuje podmienke $f_d\left(x_1,x_2,...,x_n\right)=\overline{f\left(\overline{x}_1,\overline{x}_2,...,\overline{x}_n\right)}$.

Cvičenie 7.10. Zostrojte Boolovu funkciu $f\left(x,y,z\right)$ vo forme sumy produktov klauzúl k premenným x, y a z, ktorá má hodnotu **1**vtedy a len vtedy, ak
(a) $x=y=\mathbf{0}, z=\mathbf{1}$,
(b) $x=\mathbf{0}, y=\mathbf{1}, z=\mathbf{0}$,
(c) $y=z=\mathbf{1}$.

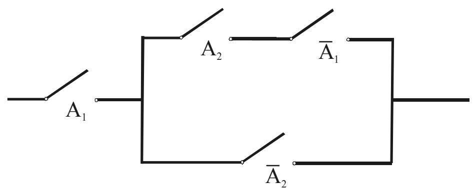
Cvičenie 7.11. Zostrojte Boolovu funkciu $f\left(x,y,z\right)$ vo forme sumy produktov klauzúl k premenným x, y a z, ktorá je ekvivalentná s funkciou
(a) $F\left(x,y,z\right)=x+y+\overline{z}$,
(b) $F\left(x,y,z\right)=x\overline{z}$.

Cvičenie 7.12. Zostrojte spínacie funkcie pre spínacie obvody

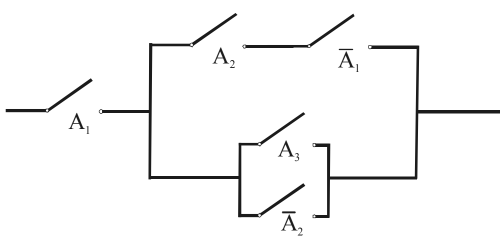
(a)



(b)



(c)

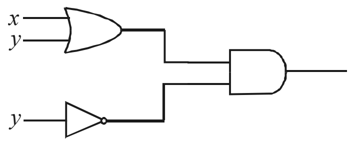


Cvičenie 7.13. Ústredné kúrenie v rodinnom dobe je riadené troma termostatmi, ktoré sú umiestnené v každej izbe domu. termostaty sú nastavené na 18°C, pričom z dôvodu šetrenia energiou sa požaduje, aby systém ústredného kúrenia bol zapnutý keď teplota aspoň v dvoch izbách je menšia ako 18°C, v opačnom prípade systém je vypnutý. Navrhnite spínačový

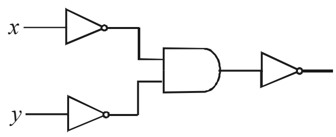
systém, ktorý prijíma signály z termostatov a ktorý riadi ústredné kúrenie. Pokúste sa minimalizovať navrhnutý systém, aby bol čo najjednoduchší. .

Cvičenie 7.14. Zostrojte tabuľku výstupov logických obvodov

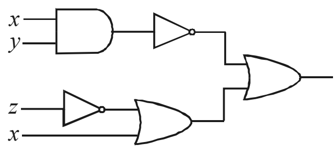
(a)



(b)



(c)



Cvičenie 7.15. Zostrojte logické obvody, ktoré simulujú Boolove funkcie

(a) $\overline{x}+y$,

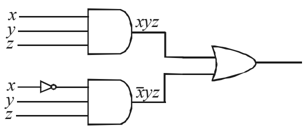
(b) $\left(\overline{x+y}\right)x$,

(c) $xyz+\overline{x}\,\overline{y}\,\overline{x}$,

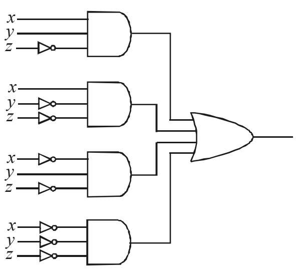
(d) $\overline{\left(\overline{x}+z\right)\left(y+\overline{z}\right)}$.

Cvičenie 7.16. Zjednodušte logické obvody

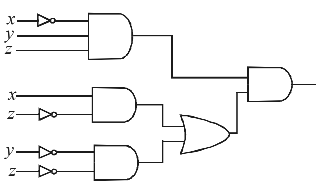
(a)



(b)



(c)



Cvičenie 7.17. Pomocou Quinovej a McCluskeyho metódy nájdite optimálne výrazy k Boolovým funkciám

(a) $wxyz + wx\overline{y}z + wx\overline{y}\,\overline{z} + w\overline{x}y\overline{z} + w\overline{x}\,\overline{y}z$,

(b) $wxy\overline{z} + wx\overline{y}z + w\overline{x}yz + \overline{w}x\overline{y}z + \overline{w}\,\overline{x}y\overline{z} + \overline{w}\,\overline{x}\,\overline{y}z$,

(c) $wxyz + wxy\overline{z} + wx\overline{y}z + w\overline{x}\,\overline{y}z + w\overline{x}\,\overline{y}\,\overline{z} + \overline{w}x\overline{y}z + \overline{w}\,\overline{x}y\overline{z} + \overline{w}\,\overline{x}\,\overline{y}z$.

Cvičenie 7.18. Napíšte program pre Quinovu a McCluskeyho optimalizáciu Boolových funkcií, kde optimalizačný problém minimálneho pokrytia (7.52) sa rieši pomocou evolučného algoritmu. Program by mal v reálnom čase vyriešiť úlohu pre 10 premenných a niekoľko sto klauzúl. Táto úloha je len pre mimoriadne zdatných programátorov, ktorí sú schopní taktiež aj teoretickej analýzy problému. Najlepšie riešenie bude ohodnotené 100 bodmi!!.