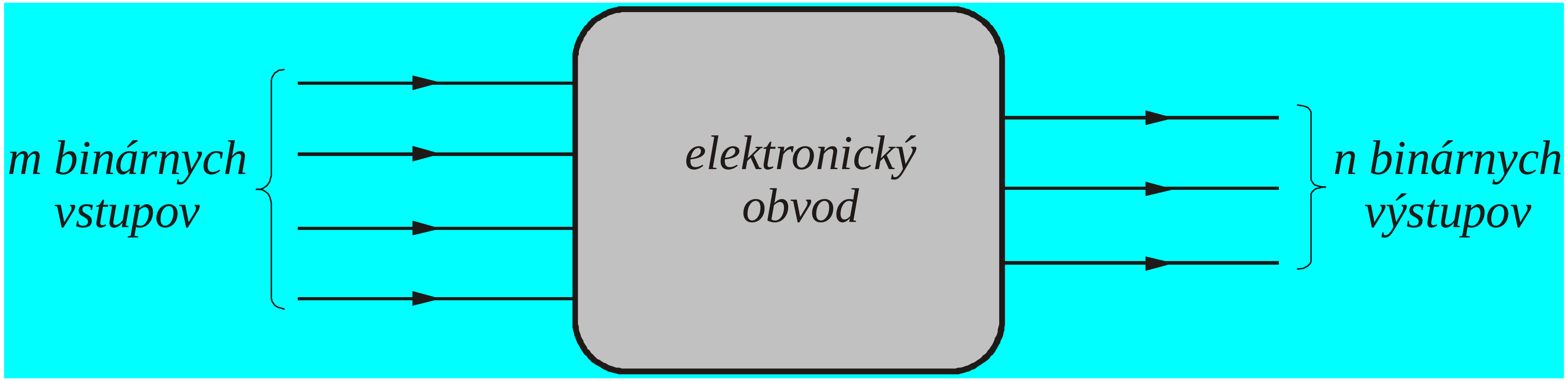
Algebraické štruktúry II

- **Boolova algebra**
- **Spínacie obvody**
- **Logické obvody**
- **Minimalizácia Boolových výrazov pomocou Quine a McCluskey metódy**

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.



Všeobecná definícia ***Boolovej funkcie*** je

$$f:\{0,1\}^m \rightarrow \{0,1\}^n$$

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.

Boolova algebra má dva známe modely, prvým je výroková logika a druhým algebra teórie množín. Medzi zákonmi výrokovej logiky a formulami teórie množín existuje „dualizmus“

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

Vo všeobecnosti, dualizmus medzi výrokovou logikou a algebrou teórie množín môžeme zosumarizovať takto

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

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

$$\cdot : B\times B\to B \text{ a } + : B\times B\to B$$

a unárna operácia komplementu

$$\bar{\ } : B\to B$$

ktoré vyhovujú týmto podmienkam

(1) komutatívnosť:

$$x\cdot y=y\cdot x \text{ a } x+y=y+x$$

(2) asociatívnosť:

$$(x\cdot y)\cdot z=x\cdot(y\cdot z) \text{ a } (x+y)+z=x+(y+z)$$

(3) distributívnosť:

$$x\cdot(y+z)=(x\cdot y)+(x\cdot z) \text{ a } x+(y\cdot z)=(x+y)\cdot(x+z)$$

(4) vlastnosť konštanty **0**:

$$x=x+\mathbf{0} \text{ a } x\cdot\bar{x}=\mathbf{0}$$

(5) vlastnosť konštanty **1** : $x=x\cdot\mathbf{1}$ a $x+\bar{x}=\mathbf{1}$

Príklad

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

$\bullet$	0	1
0	0	0
1	0	1

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

Jednoducho sa môžeme presvedčiť, že algebraická štruktúra $\left(B,+,\cdot,\bar{\ },\mathbf{0},\mathbf{1}\right)$ je Boolova algebra.

Príklad

Nech $B = \{p, q, r, ...\}$ je množina výrokových formúl, ktorá je uzavretá vzhľadom k binárnym operáciám 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

$$p \vee 0 = 0 \vee p = p \quad \text{a} \quad p \wedge 1 = 1 \wedge p = p$$

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

Princíp duality Boolovej algebry

Postulujme nejakú formulu Boolovej algebry, duálnu formu dostaneme tak, že urobíme zámenu symbolov

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

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 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 (princíp duality). Každá veta Boolovej algebry je taktiež vetou aj v duálnej forme.

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.

Definícia. Nech $\left(B,+,\cdot,^{-},\mathbf{0},\mathbf{1}\right)$ 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 $\overline{x}$

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

Definícia. Nech $\left(B,+,\cdot,^{-},\mathbf{0},\mathbf{1}\right)$ 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 $\left(X\cdot Y\right)$, $\left(X+Y\right)$, $\overline{X}$ a $\overline{Y}$ sú Boolove formuly.

Rastúca priorita operácií: (1) súčet, (2) súčin a (3) komplement. Napríklad, formulu $\left(\left(x\cdot y\right)+z\right)$ 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

Zjednodušte formulu $\left((x+y)\cdot(\overline{x}+\overline{y})\right)$. Použitím distributívneho zákona a zákona nulitnosti

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

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

Podľa predošlého príkladu $\varphi_1=(x+y)\cdot(\overline{x}+\overline{y})$ 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 $\varphi_1=\varphi_2$.

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

(1) ***Boolova funkcia*** premenných $x_1,x_2,...,x_n$, pre danú, je funkcia $f:B^n\to B$, pričom $f\left(x_1,x_2,...,x_n\right)$ je Boolova formula.

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

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

$$f:B^2\to B \qquad f\left(x_1,x_2\right)=x_1\left(\overline{x}_1+x_2\right)$$
$$g:B^2\to B \qquad g\left(x_1,x_2\right)=x_1x_2$$

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

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

Ako príklad súčinovej klauzuly premenných x_1, x_2, x_3 sú tieto formuly: $x_1 x_2 x_3$, $x_1 x_2 \overline{x}_3$, $x_1 \overline{x}_2 x_3$, $\overline{x}_1 x_2 x_3, \ldots$, $\overline{x}_1 \overline{x}_2 \overline{x}_3$.

Ak použijeme formalizmus x^e, potom súčinovú klauzulu premenných $x_1, x_2, \ldots, x_n$, ktorá je špecifikovaná binárnym vektorom $\boldsymbol{e} = (e_1, e_2, \ldots, e_n)$, má tvar

$$l_{\boldsymbol{e}} = x_1^{e_1} x_2^{e_2} \ldots x_n^{e_n}$$

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

$$l_{(11011)} = x_1^1 x_2^1 x_3^0 x_4^1 x_5^1 = x_1 x_2 \overline{x}_3 x_4 x_5$$

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

Podobne ako pre súčinovú klauzulu, môžeme aj súčtovú klauzulu pre premenné $x_1, x_2, \ldots, x_n$ špecifikovať binárnym vektorom $x_1, x_2, \ldots, x_n$

$$L_{\boldsymbol{e}} = x_1^{1-e_1} + x_2^{1-e_2} + \ldots + x_n^{1-e_n}$$

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

$$L_{\boldsymbol{e}} = x_1^0 + x_2^1 + x_3^0 + x_4^1 + x_5^1 = \overline{x}_1 + x_2 + \overline{x}_3 + x_4 + x_5$$

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

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

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

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

To znamená, že pre Boolovu funkciu sú 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)

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

Boolove funkcie $f(x_1,x_2,...,x_n)$ a $F(x_1,x_2,...,x_n)$ sú ekvivalentné, t. j. majú rovnaké funkčné hodnoty pre rôzne hodnoty argumentov.

Príklad

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

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

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 ekvivalentný DNF tvar

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

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

$$f\left(x_1,x_2,...,x_n\right)=\prod_{\boldsymbol{e}}\left(f\left(e_1,e_2,...,e_n\right)+x_1^{e_1}+x_2^{e_2}+...+x_n^{e_n}\right)$$

$$=\prod_{\boldsymbol{e}}\left(f\left(e_1,e_2,...,e_n\right)+L_{\left(1-e_1,1-e_2,...,1-e_n\right)}\right)$$

$$=\prod_{\boldsymbol{e}}\left(f\left(\boldsymbol{e}\right)+L_{\overline{\boldsymbol{e}}}\right)$$

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

Vyjadrite $f(x_1, x_2) = x_1(x_1 + x_2)$ v KNF tvare. Tabuľku funkčných hodnôt tejto Boolovej funkcie má tvar

#	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 vety zostrojíme Boolovu funkciu, ktorá je ekvivalentná funkcii

$$f(x_1, x_2) = x_1(x_1 + x_2)$$

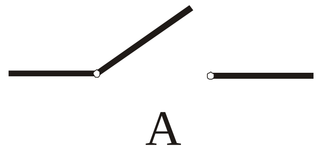
$$f(x_1, x_2) =$$

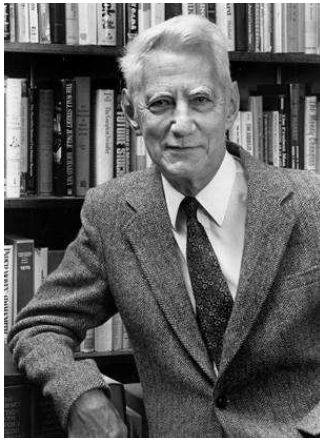
$$= \left(\cancel{\underbrace{f(\mathbf{0,0})}_{\mathbf{0}}} + x_1 + x_2\right) \cdot \left(\cancel{\underbrace{f(\mathbf{0,1})}_{\mathbf{0}}} + x_1 + \overline{x}_2\right) \cdot \underbrace{\left(\underbrace{f(\mathbf{1,0})}_{\mathbf{1}} + \overline{x}_1 + x_2\right)}_{\mathbf{1}} \cdot \underbrace{\left(\underbrace{f(\mathbf{1,1})}_{\mathbf{1}}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

$$= (x_1 + x_2) \cdot (x_1 + \overline{x}_2)$$

Spínacie obvody

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





Claude Shannon
1916-2001

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

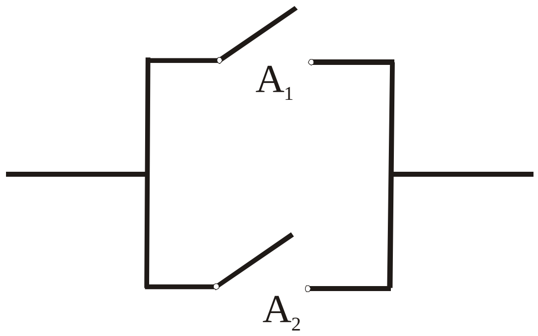
Hovoríme, že v tomto prípade sú spínače zapojené *sériovo*. Nech x_1 a x_2 sú premenné popisujúce stavy spínačov A_1 resp. A_2, tieto premenné ak sa rovnajú 1 (0) , potom daný spínač je uzavretý (otvorený). Nech $f\left(x_1,x_2\right)$ je funkcia, ktorej hodnota sa rovná 1 (0) pre tie hodnoty x_1 a x_2, ktoré umožňujú (znemožňujú) tok prúdu.

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

spínacia funkcia

$$f\left(x_1,x_2\right)=x_1x_2$$

Ďalší druh spínacieho obvodu má paralélne zapojenie spínačov

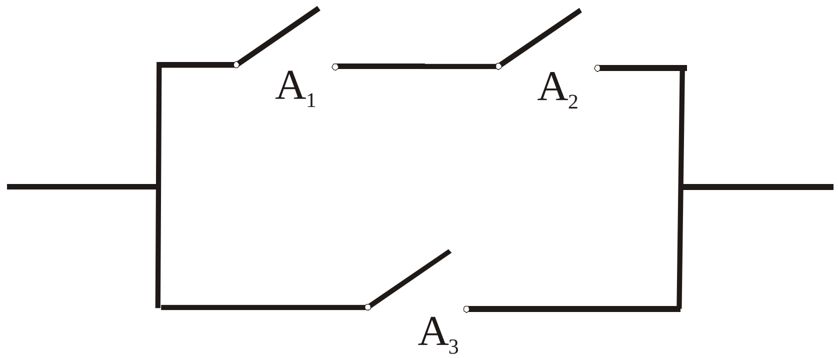


#	x_1	x_2	$g(x_1, x_2)$
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	1

Spínacia funkcia

$$g(x_1, x_2) = x_1 + x_2$$

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



Spínacia funkcia má tvar

$$f\left(x_1, x_2, x_3\right) = f_1\left(x_1, x_2\right) + f_2\left(x_3\right) = x_1 x_2 + x_3$$

Príklad

Nech na začiatku a konci chodby sú umiestnené stenové vypínače S_1 a S_2, pomocou ktorých zapneme alebo vypneme svetlo nad schodišťom. Požadujeme, aby na každom konci sme svetlo mohli vypnúť alebo zapnúť nezávisle od polohy druhého vypínača. Alternatívna formulácia, ak sú oba spínače S_1 a S_2 vypnuté alebo zapnuté, potom zariadením nepreteká prúd, ale stačí, aby bol zapnutý práve jeden vypínač, 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

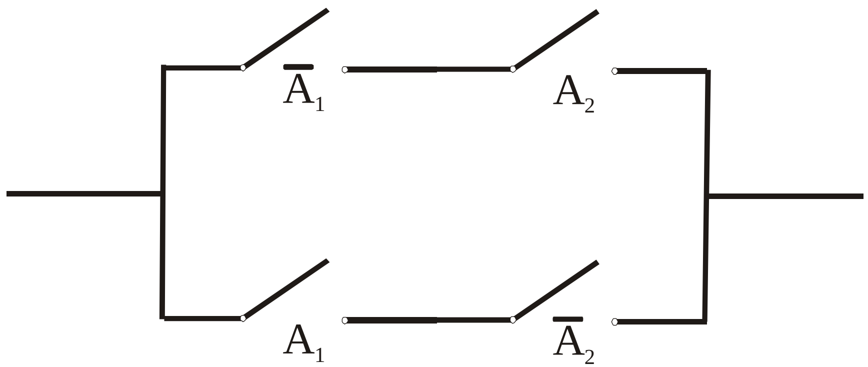
$\longrightarrow$

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

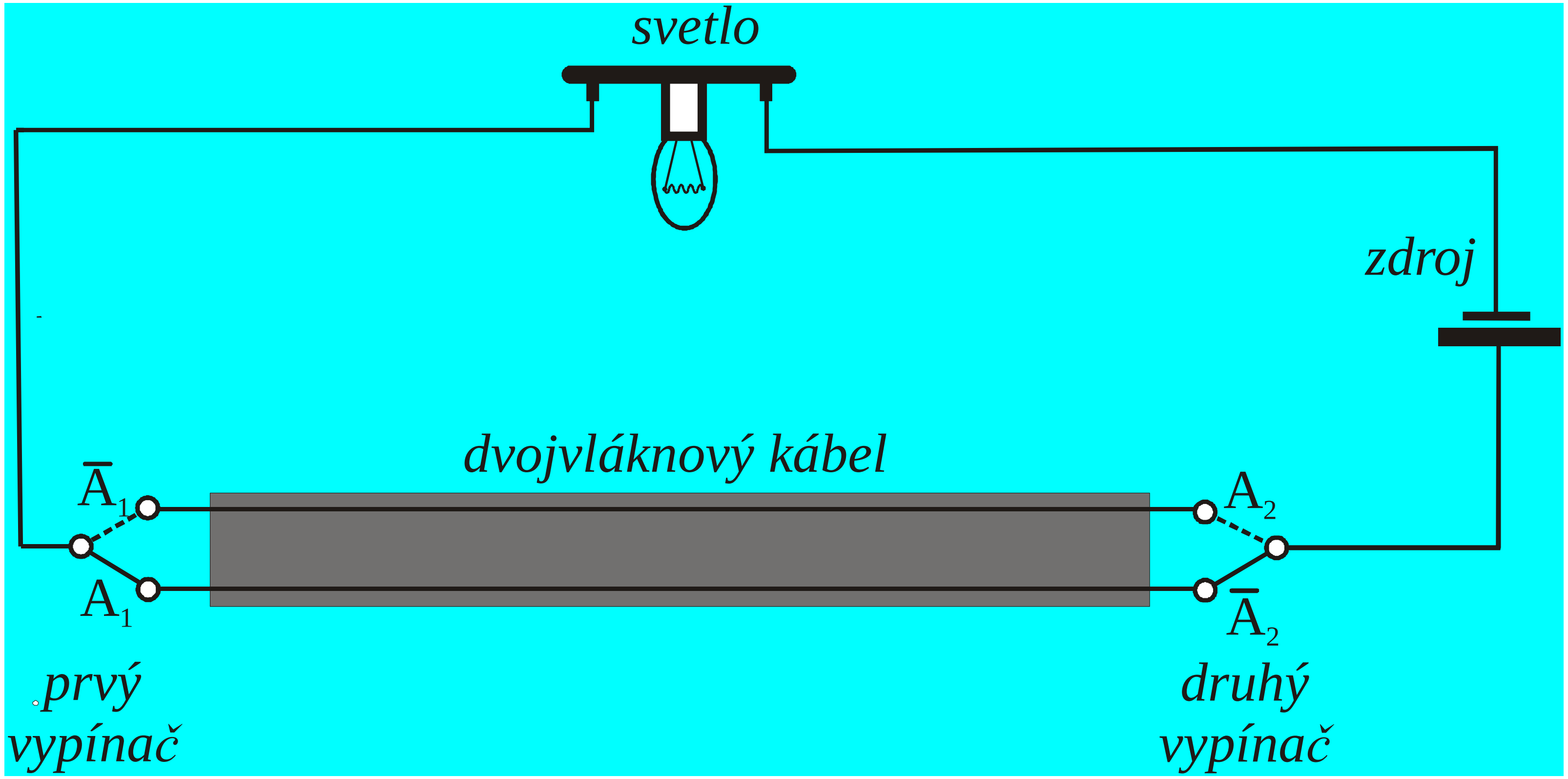
Tabuľka špecifikuje Boolovu funkciu v DNF

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

Spínací obvod so spínacou funkciou takto špecifikovanou má tvar



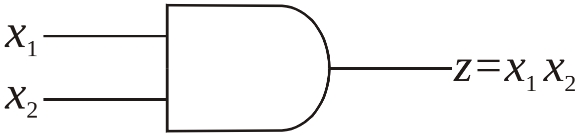
Realizácia tohto spínacieho obvodu, kde vytieňované oblasti tvoria stenové vypínače, ktoré sú spojené dvojvláknovým káblom.

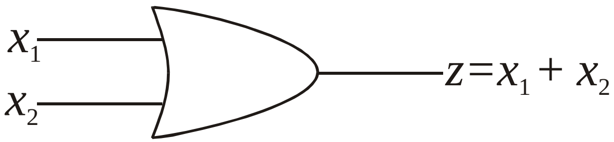


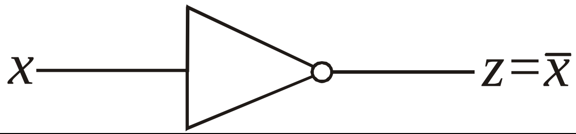
Logické obvody

Logické brány

logická brána konjunkcie	logická brána disjunkcie	logická brána negácie







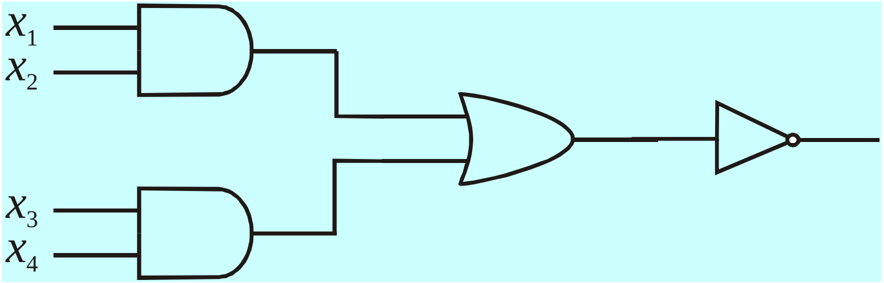
x_1	x_2	$x_1 x_2$
0	0	0
0	1	0
1	0	0
1	1	1

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

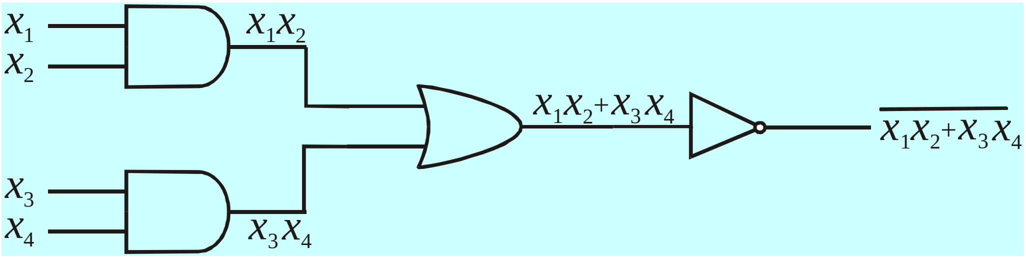
x_1	$\overline{x}_1$
0	1
1	0

Príklad

Zostrojte Boolovu funkciu pre logický obvod



Jednotlivé spoje tohto logického obvodu ohodnotíme takto



To znamená, že Boolova funkcia priradená tomuto obvodu má tvar

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

Sumátor binárnych čísel

Prvá etapa spočíva v návrhu logického obvodu (nazývaného *polosumátor*) ktorý sčíta dve jednobitové čísla x a y

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

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

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

$$c = g(x,y) = xy$$

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

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

$$\begin{array}{rr} & x \\ & y \\ \hline c & s \\ & z \\ \hline u & v \end{array}$$

$$u = \bar{x}yz + x\bar{y}z + xy\bar{z} + xyz = \left(\bar{x}y + x\bar{y}\right)z + xy\left(z + \bar{z}\right)$$

$$= \left(x \oplus y\right)z + xy$$

$$v = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz = \left(\bar{x}y + x\bar{y}\right)\bar{z} + \left(\bar{x}\,\bar{y} + xy\right)z$$

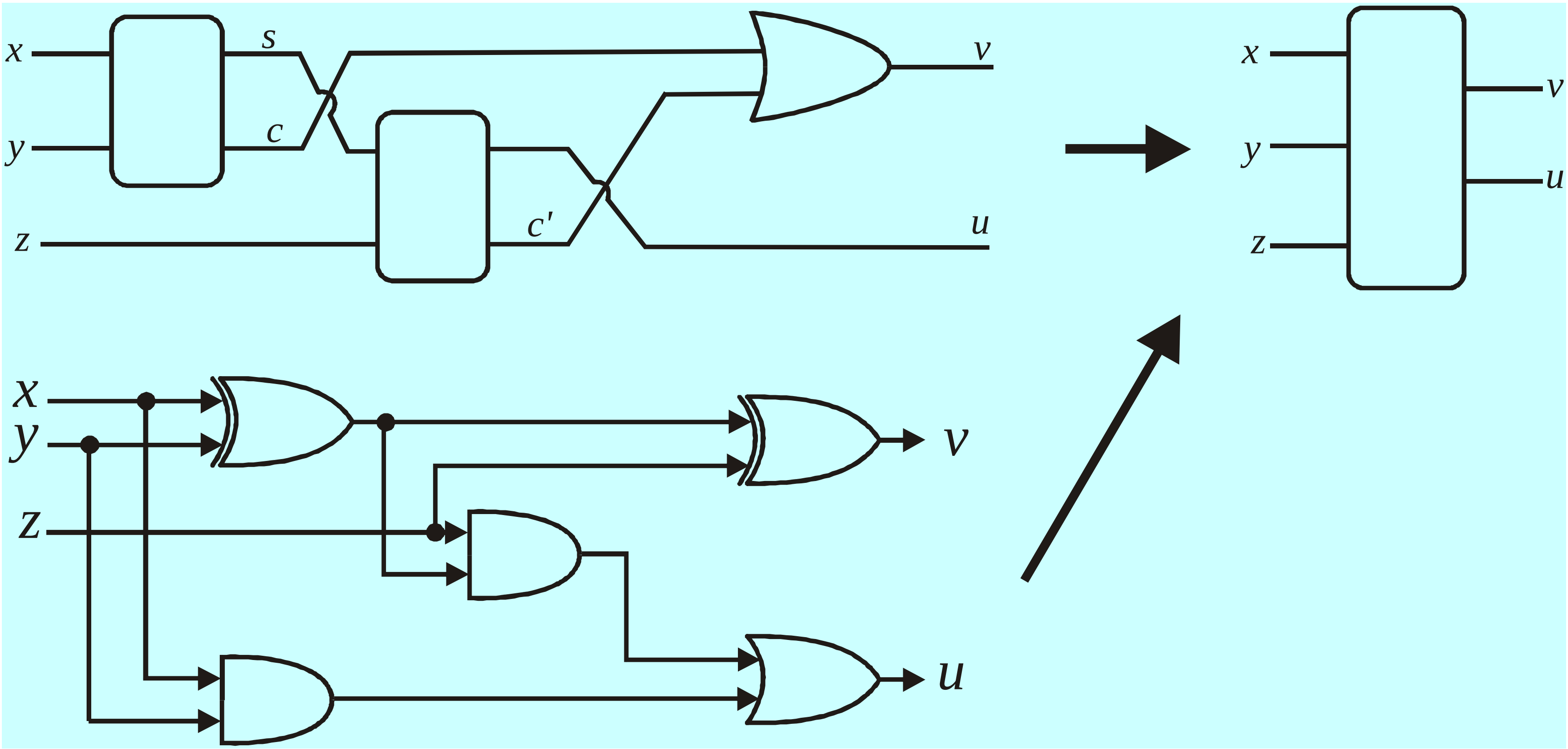
$$= \left(x \oplus y\right) \oplus z$$

Tretia etapa využije dvojitý sumátor (ako blok) k realizácii sčítania dvoch bitový čísel ľubovolnej dĺžky *n*. Ako ilustračný príklad študujme sčítanie dvoch trojbitových čísel

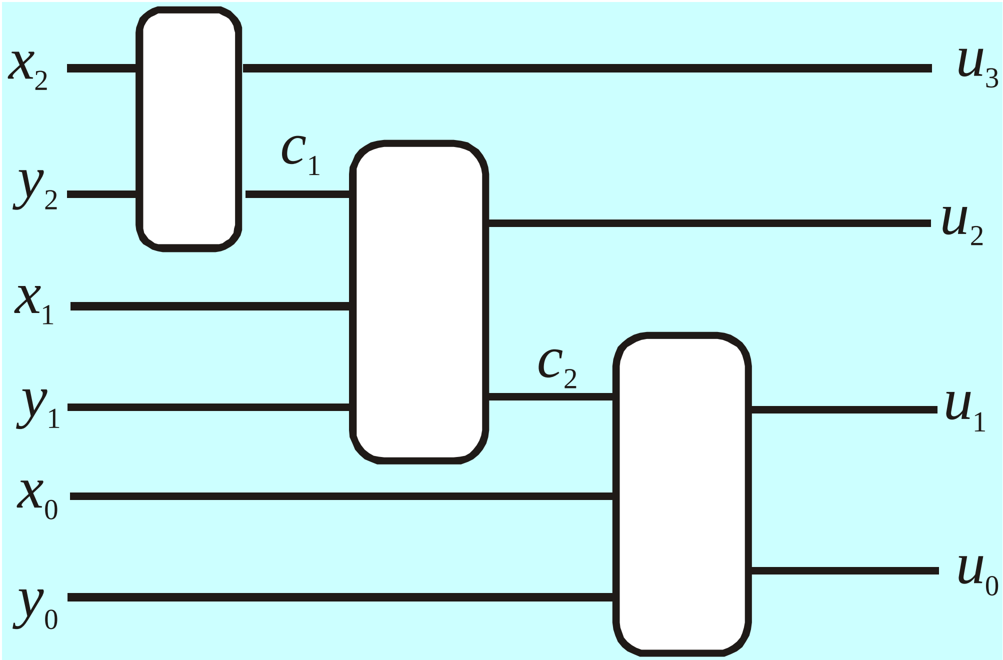
$$\begin{array}{r} x_0\ x_1\ x_2 \\ y_0\ y_1\ y_2 \\ \hline u_0\ u_1\ u_2\ u_3 \end{array}$$

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

$$(u_3, c_1) = PS(x_2, y_2)$$
$$(u_2, c_2) = DS(x_1, y_1, c_1)$$
$$(u_0, u_1) = DS(x_0, y_0, c_2)$$



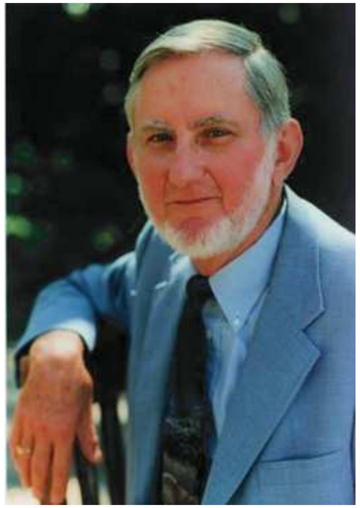
Túto postupnosť príkazov môžeme diagramaticky reprezentovať ako logický obvod s polosumátorom a dvoma dvojitými sumátormi



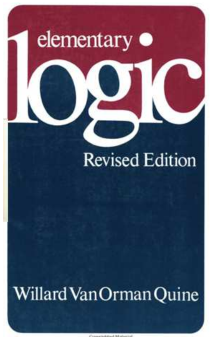
Pomocou blokov pre polosumátor a dvojitý sumátor môžeme zostrojiť logický obvod pre sumáciu dvoch digitálnych čísel ľubovolnej dĺžky.

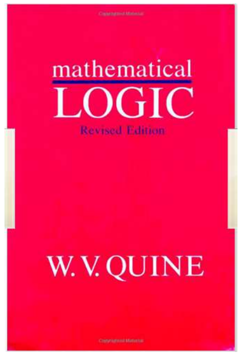
Minimalizácia Boolových výrazov pomocou Quine a McCluskey metódy

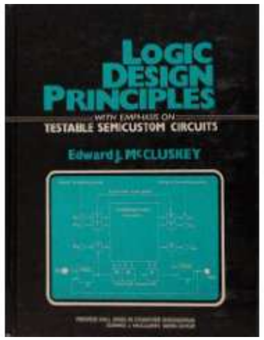
Willard Van Orman Quine (* 1908 – † 2000)



Edward J. McCluskey (*1929)







Ilustračný príklad

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

#	x	y	z	f
1	0	0	0	1
2	0	0	1	1
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	0
8	1	1	1	1

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

Každá klauzula môže byť reprezentovaná bitovým reťazcom $\boldsymbol{e} = (e_1, e_2, e_3) \in \{0,1\}^3$

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

kde $\xi^e = \xi$, ak $e = 1$, $\xi^e = \overline{\xi}$, ak $e = 0$, pre $\xi = x, y, z$.

$$f(x,y,z) = (000) + (001) + (011) + (101) + (111)$$

Pre takto definovanú binárnu reprezentáciu môžeme použiť metriku ***Hammingovej vzdialenosti*** ku kvantifikácii podobnosti medzi binárnymi vektormi. Nech $\boldsymbol{e}_i = \left(e_1^{(i)}, e_2^{(i)}, \ldots, e_n^{(i)}\right)$ a $\boldsymbol{e}_j = \left(e_1^{(j)}, e_2^{(j)}, \ldots, e_n^{(j)}\right)$ sú dve binárne reprezentácie klauzúl, potom

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

Táto vzdialenosť pre binárne vektory nám špecifikuje počet polôh v ktorých sa binárne vektory vzájomne odlišujú.

Pre ilustračný príklad Boolova funkcia je reprezentovaná pomocou množiny 5 binárnych reťazcov dĺžky 3

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

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.

Proces sčítania klauzúl

Z množiny U_f vyberieme prvú a druhú klauzulu, ich binárne reprezentácie (111) a (101) sa líšia len hodnotou binárnej premennej v druhej polohe, $d_H\left(111{,}101\right)=1$. Tieto dve klauzuly sú sčítané takto

$$xyx + x\bar{y}z = x\underbrace{\left(y+\bar{y}\right)}_{1}z = xz$$

V binárnej reprezentácii tento proces zjednodušenia formálne vyjadríme takto

$$\left(111\right)+\left(101\right)=sum\left(\left(111\right),\left(101\right)\right)=\left(1\#1\right)$$

„*prázdny*“ symbol ´#´ reprezentuje prázdne miesto v binárnej reprezentácii novej klauzuly *xz*.

Nové klauzuly obsahujúce jeden symbol ´#´ tvoria množinu

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

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 $U_f^{(1)}$

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

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.

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

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

Rekurentná formula je inicializovaná množinou $U_f^{(0)} = U_f$. Musí existovať také kladné celé číslo n, že tento proces tvorby nových množín je ukončený, t. j. platí $U_f^{(n+1)} = \mathcal{A}\left(U_f^{(n)}\right) = \emptyset$.

- 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 rekurentný proces je ukončený vtedy, ak operátor $\mathcal{A}$ aplikovaný na množinu $U_f^{(n)}$ produkuje prázdnu množinu, t. j. $\mathcal{A}\left(U_f^{(n)}\right) = \emptyset$.

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

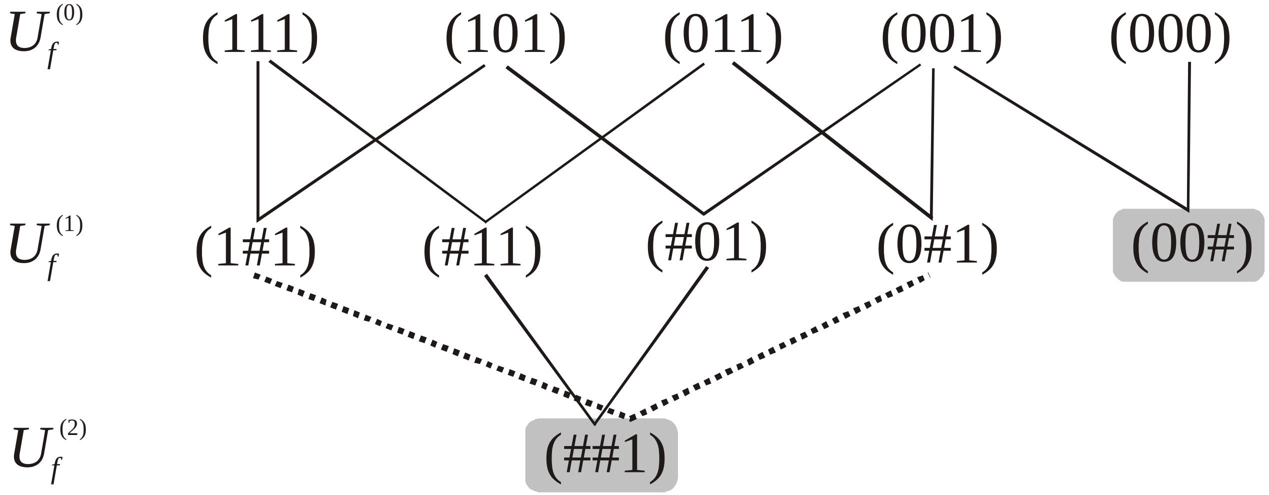
V stĺpcoch pre prvú a druhú etapu sú uvedené aj dvojice indexov klauzúl z predchádzajúceho stĺpca, ktoré boli použité v sumačnom procese.

Úloha: ako vybrať taký minimálny počet klauzúl zostrojených v prvej alebo v ďalších etapách, ktoré sú odvoditeľné zo všetkých pôvodných klauzúl z množiny $U_f^{(0)} = U_f$.

Množina klauzúl, ktorá vznikne zjednotením množín $U_f^{(0)}, U_f^{(1)}, U_f^{(3)}, \ldots$

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

je čiastočne usporiadaná a je 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.

Úloha: Vybrať také minimálne klauzuly, ktoré pokrývajú pôvodné klauzuly z množiny $U_f^{(0)}$. Pre každú klauzulu $\boldsymbol{e} \in U_f^{(1)} \cup U_f^{(2)} \cup \ldots$, ktorá obsahuje aspoň jeden prázdny symbol, zostrojíme množinu

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

ktorá obsahuje všetky pôvodné klauzuly (neobsahujúce prázdne symboly #), ktoré sú pokryté klauzulou ***e***

Nech množina minimálnych klauzúl je označená V, potom hľadáme takú jej podmnožinu $\tilde{V} \subseteq V$, ktorej klauzuly plne pokrývajú množinu $U_f^{(0)}$

$$\bigcup_{e \in \tilde{V}} U(e) = U_f$$

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

$$\tilde{V} = arg \min_{V' \subseteq V} [V']$$

- 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

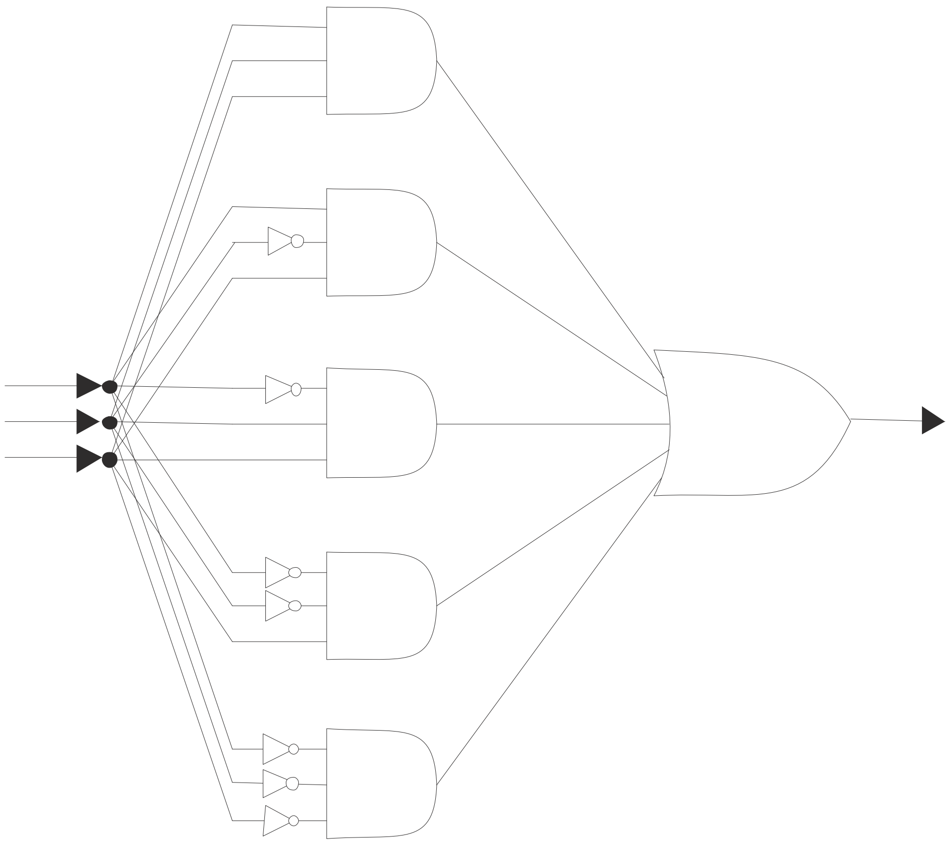
$$f\left(x,y,z\right)=xyz+x\overline{y}z+\overline{x}yz+\overline{x}\,\overline{y}z+\overline{x}\,\overline{y}\,\overline{z}$$

v ekvivalentnom tvare

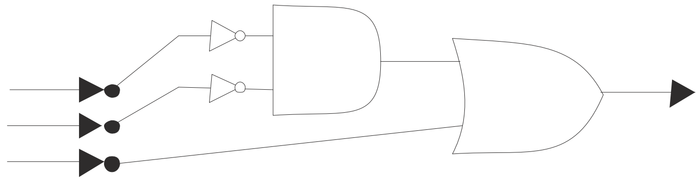
$$f\left(x,y,z\right)=z+\overline{x}\,\overline{y}$$

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

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



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



Príklad

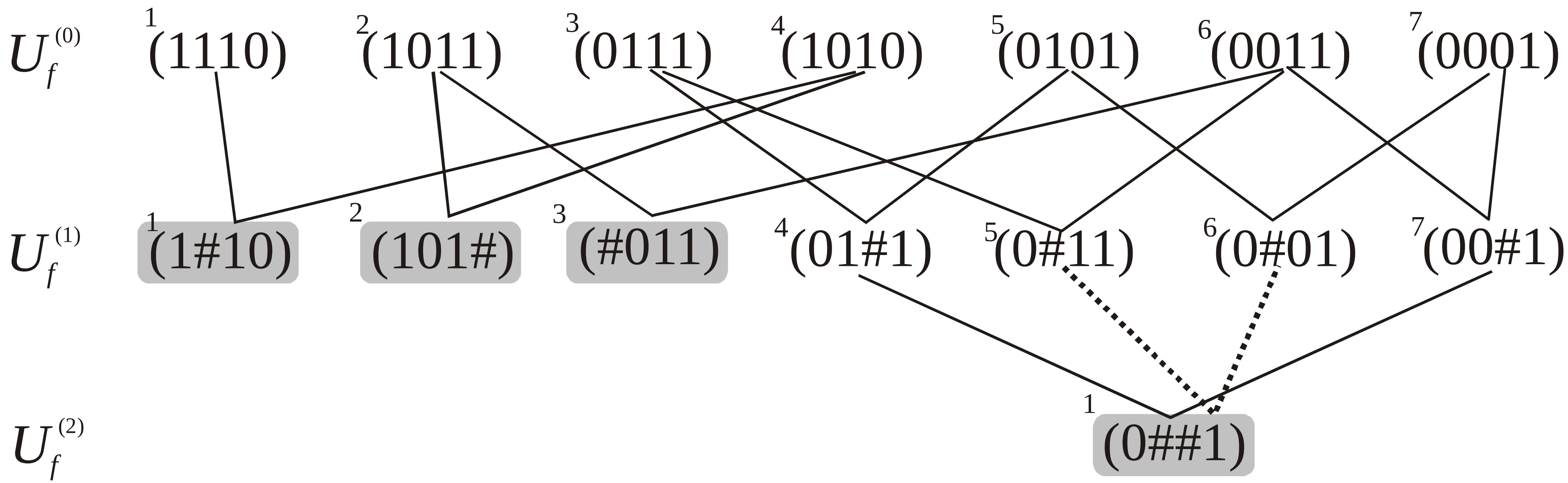
Uvažujme Boolovu funkciu

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

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)				

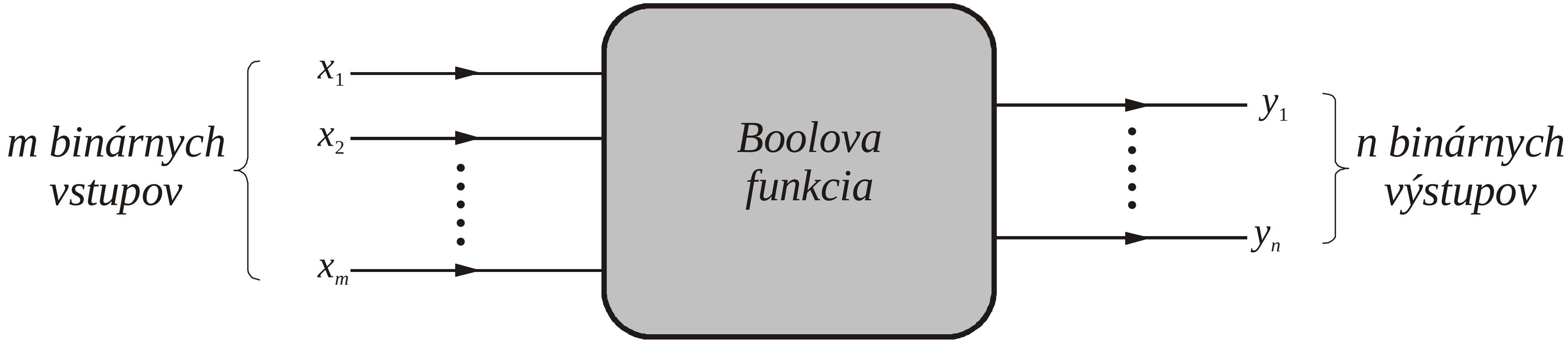
Hasseho diagram



Teraz stojíme pred problémom ako vybrať taký minimálny počet klauzúl, ktoré nám budú pokrývať celú pôvodnú množinu klauzúl.

Optimalizácia Boolovej funkcie s *viacerými* funkčnými hodnotami

$$f:\{0,1\}^m \rightarrow \{0,1\}^n\text{, kde } n \geq 2$$



Poznámka: Optimalizačná metóda Q-M sa ľahko zovšeobecní aj pre tento prípad. Jednotlivé členy Boolovej funkcie budú označené horným indexom výstupu, ktorý špecifikuje.

Tabuľka špecifikujúca Boolovu funkciu

#	x_1	x_2	x_3	x_4	f_A	f_B
1	0	0	0	0	0	0
2	0	0	0	1	1	1
3	0	0	1	0	1	1
4	0	0	1	1	0	0
5	0	1	0	0	0	0
6	0	1	0	1	1	1
7	0	1	1	0	0	0
8	0	1	1	1	0	0

#	x_1	x_2	x_3	x_4	f_A	f_B
9	1	0	0	0	0	0
10	1	0	0	1	0	0
11	1	0	1	0	1	1
12	1	0	1	1	0	0
13	1	1	0	0	0	0
14	1	1	0	1	0	0
15	1	1	1	0	1	1
16	1	1	1	1	1	0

Vybrané riadky s jednotkovou funkčnou hodnotou

#	x_1	x_2	x_3	x_4	f_X
2	0	0	0	1	A,B
3	0	0	1	0	A,B
6	0	1	0	1	A,B
11	1	0	1	0	A,B
15	1	1	1	0	A,B
16	1	1	1	1	A

#	r_i - r_j	x_1	x_2	x_3	x_4	f_X
1	2 - 6	0	#	0	1	A,B
2	3 - 11	#	0	1	0	A,B
3	11 - 15	1	#	1	0	A,B
4	15 - 16	1	1	1	#	A

Poznámka: Budeme vyberať do procesu „súčtu“ len také dva riadky, ktoré

(1) majú jednotkovú Hammingovu vzdialenosť,

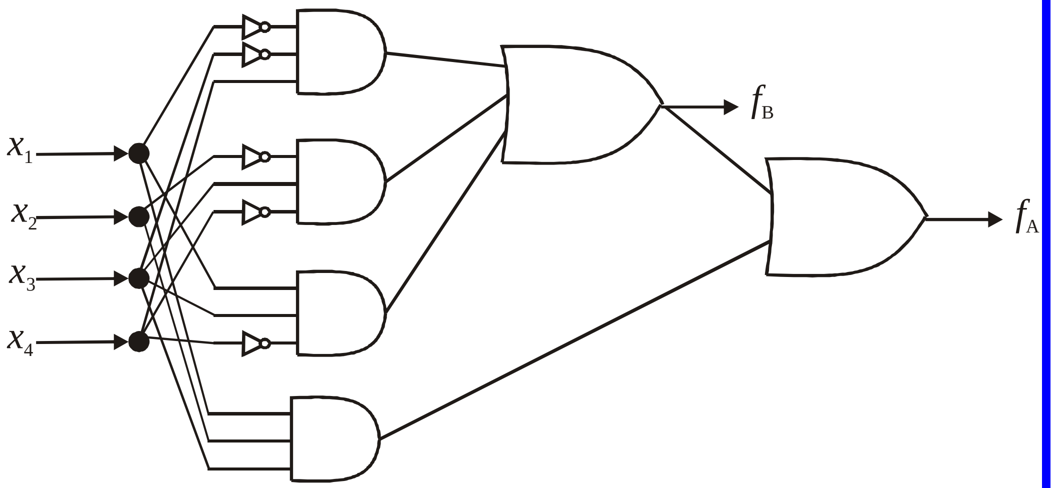
(2) prienik indexov funkčných hodnôt je neprázdna množina, pričom výsledný súčet je označený týmto prienikom.

$0001^{A,B}$ $0010^{A,B}$ 0101^{AB} $1010^{A,B}$ $1110^{A,B}$ 1111^{A}

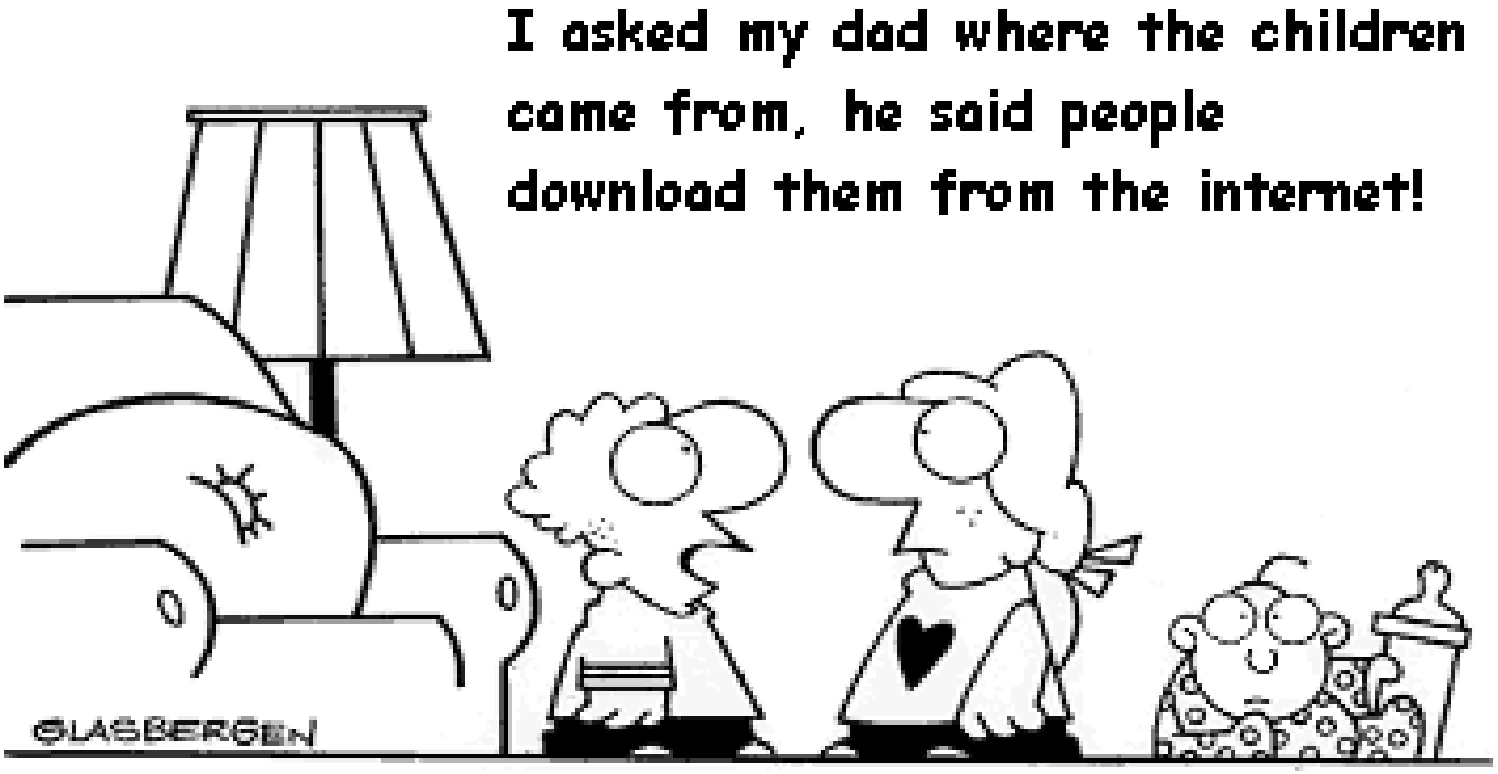
$0\#01^{A,B}$ $\#010^{A,B}$ $1\#10^{A,B}$ $111\#^{A}$

$$f_A = 0\#01 + \#010 + 1\#10 + 111\#$$

$$f_B = 0\#01 + \#010 + 1\#10$$



The End


I asked my dad where the children came from, he said people download them from the internet!
GLASBERGEN