1.3.17 Souvislost s logickými obvody. Každá výroková formule se dá realizovat jako logický obvod. Potřebujeme proto hradla pro všechny logické spojky (nejčastější jsou hradlo NOT – též zvaný invertor, hradlo AND a hradlo OR). Logický obvod má na každém vstupu 0 či 1 (přesněji logickou proměnnou) a každé kombinaci vstupních hodnot přiřazuje buď hodnotu 0 nebo hodnotu 1.

Tautologii odpovídá logický obvod, který na každou kombinaci vstupních hodnot dá výstup 1, kontradikci pak obvod, jehož výstup je vždy 0. Splnitelné formuli odpovídá logický obvod, který alespoň na jednu kombinaci vstupních hodnot dá výstup 1.

1.4 Úplné systémy logických spojek

1.4.1 Definice. Řekneme, že množina logických spojek Δ tvoří *úplný systém logických spojek* (též *funkčně úplný systém logických spojek*), jestliže pro každou formuli α existuje formule β s ní tautologicky ekvivalentní, která používá pouze spojky z množiny Δ. □

1.4.2 Tvrzení. Nechť Δ tvoří úplný systém logických spojek a nechť Π je množina spojek. Jestliže platí

1. pro každou binární spojku $\square \in \Delta$ existuje formule α obsahující pouze spojky z množiny Π a taková, že $\alpha \models\!\!\models x \square y$,
2. pro každou unární spojku $\diamond \in \Delta$ existuje formule β obsahující pouze spojky z množiny Π a taková, že $\beta \models\!\!\models \diamond x$,
3. pro každou nulární spojku $K \in \Delta$ existuje formule γ obsahující pouze spojky z množiny Π a taková, že $\gamma \models\!\!\models K$,

pak Π je také úplný systém logických spojek. □

1.4.3 Příklady úplných systémů logických spojek.

1. Množina $\{\neg, \wedge, \vee, \Rightarrow\}$ tvoří úplný systém logických spojek.

   To je proto, že další logické spojky $\Leftrightarrow$, $|$, $\downarrow$, $\oplus$, **F** a **T** můžeme „utvořit“ ze spojek z množiny $\{\neg, \wedge, \vee, \Rightarrow\}$. Přesněji

   $$\alpha \Leftrightarrow \beta \models\!\!\models (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha),$$

   zbytek plyne z 1.3.10, 1.3.12, 1.3.14 a 1.3.15.

2. Množina $\Delta = \{\neg, \vee\}$ tvoří úplný systém logických spojek.

   Víme, že $\{\neg, \wedge, \vee, \Rightarrow\}$ tvoří úplný systém logických spojek. Nyní si stačí uvědomit, že platí:

   $$(a \Rightarrow b) \models\!\!\models (\neg a \vee b) \quad \text{a} \quad (a \wedge b) \models\!\!\models \neg(\neg a \vee \neg b).$$

3. Množina $\Delta = \{\neg, \wedge\}$ tvoří úplný systém logických spojek.

   Protože množina $\Delta = \{\neg, \vee\}$ tvoří úplný systém logických spojek, stačí ověřit, že $\vee$ můžeme „vytvořit“ pomocí logických spojek $\neg, \wedge$. Přitom

   $$(a \vee b) \models\!\!\models \neg(\neg a \wedge \neg b).$$

4. Množina $\Delta = \{\neg, \Rightarrow\}$ tvoří úplný systém logických spojek.

   Obdobně jako výše si stačí uvědomit, že pomocí spojek $\neg$ a $\Rightarrow$ „vytvoříme“ spojku $\vee$ a/nebo $\wedge$.

   $$(a \vee b) \models\!\!\models (\neg a \Rightarrow b) \quad \text{nebo} \quad (a \wedge b) \models\!\!\models \neg(a \Rightarrow \neg b).$$

5. Množina $\{\wedge, \vee, \Rightarrow, \Leftrightarrow\}$ (a tudíž ani žádná její podmnožina) netvoří úplný systém logických spojek.

   Stačí si uvědomit, že každá formule obsahující pouze spojky $\wedge$, $\vee$, $\Rightarrow$ a $\Leftrightarrow$ je pravdivá v pravdivostním ohodnocení u_0, v němž jsou pravdivé všechny logické proměnné. Ovšem formule $\neg a$ (a je logická proměnná) je v tomto ohodnocení u_0 nepravdivá.

6. Každá z množin $\{|\}$ (tj. NAND) a $\{\downarrow\}$ (tj. NOR) je úplný systém logických spojek.

   Uvažujme nejprve spojku NAND. Využijeme fakt, že $\{\neg, \wedge\}$ tvoří úplný systém logických spojek. Stačí proto „vytvořit“ spojky $\neg$ a $\wedge$ pomocí spojky $|$. A to je možné, protože

   $$\neg a \barwedge (a \mid a) \quad \text{a} \quad (a \wedge b) \barwedge \neg(a \mid b) \barwedge (a \mid b) \mid (a \mid b).$$

   Pro spojku $\downarrow$ postupujeme analogicky, pouze používáme $\{\neg, \vee\}$ jako úplný systém logických spojek místo množiny $\{\neg, \wedge\}$. Máme

   $$\neg a \barwedge (a \downarrow a) \quad \text{a} \quad (a \vee b) \barwedge \neg(a \downarrow b) \barwedge (a \downarrow b) \downarrow (a \downarrow b).$$

Všimněte si, že ani pro spojku NAND ani pro spojku NOR neplatí asociativní zákon, tj. neplatí $(\alpha \mid \beta) \mid \gamma \barwedge \alpha \mid (\beta \mid \gamma)$; obdobně pro $\downarrow$.

1.5 CNF a DNF

Každé formuli o n logických proměnných odpovídá pravdivostní tabulka. Na tuto tabulku se můžeme dívat jako na zobrazení, které každé n-tici 0 a 1 přiřazuje 0 nebo 1. Ano, řádek pravdivostní tabulky je popsán n-ticí 0 a 1, hodnota je pak pravdivostní hodnota formule pro toto dosazení do logických proměnných. Zobrazení z množiny všech n-tic 0 a 1 do množiny $\{0, 1\}$ se nazývá *Booleova funkce*. Naopak platí, že pro každou Booleovu funkci existuje formule, která této funkci odpovídá. V dalším ukážeme, že dokonce můžeme volit formuli ve speciálním tvaru, v tzv. *konjunktivním normálním tvaru* a/nebo *disjunktivním normálním tvaru*.

1.5.1 Booleova funkce.

Definice. *Booleovou funkcí n proměnných*, kde n je přirozené číslo, rozumíme každé zobrazení $f\colon \{0,1\}^n \to \{0,1\}$, tj. zobrazení, které každé n-tici $(x_1, x_2, \ldots, x_n)$ nul a jedniček přiřazuje nulu nebo jedničku (označenou $f(x_1, x_2, \ldots, x_n)$). □

1.5.2 Formule v disjunktivním normálním tvaru.

Definice. *Literál* je logická proměnná nebo negace logické proměnné. *Minterm* je literál nebo konjunkce konečně mnoha literálů.

Řekneme, že formule je v *disjunktivním normálním tvaru*, zkráceně v *DNF*, jestliže je mintermem nebo disjunkcí konečně mnoha mintermů. □

1.5.3 Poznámka. Jestliže ve formuli, která je v DNF, obsahuje každý minterm všechny proměnné, říkáme, že se jedná o *úplnou DNF*.

Poznamenejme, že někteří autoři požadují, aby minterm neobsahoval současně proměnnou i její negaci. My to ale nevyžadujeme.

1.5.4 Konvence. Z tvrzení 1.3.4 víme, že pro spojky $\vee$ a $\wedge$ platí asociativní zákon; tj. jde-li nám jen o tautologickou ekvivalenci formulí, je jedno, jak „závorkujeme“ ve formuli, která je disjunkcí nebo konjunkcí n, $n > 1$, formulí. Budeme proto v dalším vynechávat vnitřní závorky tj. formuli, která je disjunkcí formulí α_i, pro $i = 1, 2, \ldots, n$, zapisujeme

$$\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n.$$

Obdobně pro konjunkci.

1.5.5 Věta. Ke každé Booleově funkci f existuje formule φ v DNF, která odpovídá Booleově funkci f. □

Zdůvodnění. Jestliže funkce f má všechny hodnoty rovny nule, je jí odpovídající formule kontradikce. Kontradikci můžeme reprezentovat např. formulí $x \wedge \neg x$ (v našem přístupu se jedná o minterm, tedy formuli v DNF).

Předpokládejme, že funkce f není konstantní 0, tj. pro aspoň jednu n-tici $(x_1, x_2, \ldots, x_n)$ má hodnotu 1. Pro každou takovou n-tici utvoříme minterm, který tuto n-tici popisuje takto: je-li $x_i = 1$ dáme do mintermu literál x_i, je-li $x_i = 0$ dáme do mintermu literál $\neg x_i$. (Např. pro n-tici, kde $x_1 = 0$ a ostatní $x_i = 1$ má odpovídající minterm α tvar $\alpha = \neg x_1 \wedge x_2 \wedge \ldots \wedge x_n$. Uvědomte si, že α je formule, která je pravdivá pouze v jednom ohodnocení; a to $u(x_1) = 0$, $u(x_2) = u(x_3) = \ldots = u(x_n) = 1$.) Výsledná formule je pak disjunkcí všech mintermů odpovídajících n-ticím, ve kterých je hodnota funkce f rovna 1.

Poznamenejme, že takto vzniklá formule je úplná DNF.

1.5.6 Důsledek. Ke každé formuli α existuje formule β, která je v DNF a navíc $\alpha \models\!\!\models \beta$. □

Ano, nejprve zkonstruujeme Booleovu funkci f odpovídající formuli α a k ní pak následně tautologicky ekvivalentní formuli β v DNF.

1.5.7 Formule v konjunktivním normálním tvaru. Definice. *Maxterm* je literál nebo disjunkce konečně mnoha literálů.

Řekneme, že formule je v *konjunktivním normálním tvaru*, zkráceně v *CNF*, jestliže je maxterm nebo konjunkcí konečně mnoha maxtermů. □

1.5.8 Poznámka. Poznamenejme, že maxtermu se také říká *klauzule* a to hlavně v rezoluční metodě. I my budeme dávat přednost termínu klauzule.

V některé literatuře se požaduje, aby maxterm neobsahoval současně logickou proměnnou i její negaci. My ale nic takového nevyžadujeme.

Jestliže každý maxterm/klauzule formule v CNF obsahuje všechny proměnné, říkáme, že se jedná o *úplnou CNF*.

1.5.9 Věta. Ke každé Booleově funkci f existuje formule ψ v CNF, která odpovídá Booleově funkci f. □

Zdůvodnění. Jestliže funkce f má všechny hodnoty rovny 1, je jí odpovídající formule tautologie. Tautologii můžeme reprezentovat např. formulí $x \vee \neg x$ (což je klauzule).

Předpokládejme, že funkce f není konstantní 1, tj. pro aspoň jednu n-tici $(x_1, x_2, \ldots, x_n)$ má hodnotu 0. Vytvoříme funkci f' takovou, že má přesně opačné hodnoty než f. To znamená

$$f'(x_1, x_2, \ldots, x_n) = 1 \text{ právě tehdy, když } ; f(x_1, x_2, \ldots, x_n) = 0.$$

K funkci f' najdeme formuli β v DNF podle 1.5.5. Hledanou formuli α v CNF odpovídající funkci f dostaneme z formule $\neg\beta$ dvojím použitím de Morganova pravidla. Uvědomte si, že indukcí podle n se lehce dokáže, že pro každé $n \geq 1$ platí

$$\neg(\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n) \models\!\!\models (\neg\alpha_1 \vee \neg\alpha_2 \vee \ldots \vee \neg\alpha_n).$$

Poznamenejme, že takto vzniklá formule je úplná CNF.

1.5.10 Pozorování. V důkazu předchozí věty jsme také mohli postupovat přímo, podobně jako ve větě 1.5.5, jen s tím, že bychom popisovali řádky obsahující 0, v případě, že by x_i měla hodnotu 0, do formule bychom dali literál x_i, jinak literál $\neg x_i$ a zaměnili disjunkci s konjunkcí a naopak.

1.5.11 Důsledek. Ke každé formuli α existuje formule β, která je v CNF a navíc $\alpha \models\!\models \beta$. □

1.5.12 Karnaughovy mapy. Pro zjednodušení formulí v disjunktivní a konjunktivní normální formě (viz 1.5.5 a 1.5.9) je možné použít tzv. Karnaughovy mapy. Vhodné je to hlavně pro tři nebo čtyři logické proměnné.