Kapitola 1

Výroková logika

SYNTAX

Logický systém je definovaný svojou syntaxou a sémantikou. Jazyk, ktorý umožňuje vyjadrovať vety *výrokovej logiky* sa označuje ako *výrokový počet*. Jeho syntaktické pravidlá určujú, čo môžu jednotlivé výrazy obsahovať a ako môžu byť navzájom kombinované. Gramatika pre definíciu tejto syntaxe je v Tab. 1.1 v tvare BNF (Backus-Naurova Forma).

$$
\begin{array}{rcl}
<\text{veta}> & ::= & <\text{atomická veta}> \mid <\text{zložená veta}> \\
<\text{atomická veta}> & ::= & \top \mid \bot \mid <\text{symbol}> \\
<\text{zložená veta}> & ::= & <\text{unárny operátor}> <\text{veta}> \\
 & \mid & <\text{veta}> <\text{binárny operátor}> <\text{veta}> \\
 & \mid & (<\text{veta}>) \\
<\text{unárny operátor}> & ::= & \neg \mid \ldots \\
<\text{binárny operátor}> & ::= & \vee \mid \wedge \mid \rightarrow \mid \leftrightarrow \mid \oplus \mid \uparrow \mid \downarrow \mid \ldots
\end{array}
$$

Tab. 1.1: Syntax výrokového počtu

SYMBOLY A ICH OZNAČOVANIE

Pre vyjadrenie pravdivostnej hodnoty sa používajú dva špeciálne symboly, reprezentujúce vždy pravdivý symbol ($\top$) a vždy nepravdivý symbol ($\bot$). Okrem týchto dvoch znakov medzi základné stavebné prvky patria ešte symboly, syntax ktorých sa môže rôzniť. Najčastejšie sa označujú pomocou veľkých písmen (napr. P, Q, R, …) prípadne v tvare písmena doplneného jedným alebo viacerými indexmi, pozostávajúcimi z jednej alebo niekoľkých číslic (napr. P_1, V_{325}, $X_{21,32}$). Indexový tvar sa používa hlavne v prípade väčšieho počtu symbolov, keď počet dostupných samostatných písmen už nepostačuje, alebo kvôli názornejšej väzbe na riešený problém.

LOGICKÉ OPERÁTORY

Unárne operátory majú modifikačnú funkciu, pretože menia význam viet na ktoré sú aplikované. Aj keď existuje viac možných unárnych ope-

ILUSTRAČNÝ PRÍKLAD

Úloha: *Podozrivými z podvodu sú Fero, Ondro, Stano, Rudo a Jaro. Ich výpovede:*

1. *Stano: podvádzali Fero alebo Ondro, avšak nie obaja*
2. *Fero: podvádzali Rudo alebo Stano, avšak nie obaja*
3. *Ondro: podvádzali Jaro alebo Fero, avšak nie obaja*
4. *Rudo: podvádzali Ondro alebo Jaro, avšak nie obaja*
5. *Jaro: podvádzali Ondro alebo Rudo, avšak nie obaja*

V skutočnosti však iba štyri z týchto piatich výpovedí boli pravdivé, jedna bola nepravdivá. Navyše je k dispozícii ešte výpoveď dozorujúceho učiteľa, ktorú považujme za pravdivú:

6. *Učiteľ: ak Rudo podvádzal, tak potom podvádzal aj Ondro*

Je potrebné reprezentovať sadu výpovedí logickou vetou v tvare CNF.
Reprezentácia: *Prvých päť výrokov bude reprezentovať päť symbolov P_1, P_2, P_3, P_4 a P_5, kde symbol P_i reprezentuje platnosť i-tej výpovede. Jednotlivých podozrivých a ich podvádzanie budú reprezentovať symboly F (Fero podvádzal), O (Ondro podvádzal), S (Stano podvádzal), R (Rudo podvádzal) a J (Jaro podvádzal).*
Riešenie: *Uvedený problém je reprezentovaný ako zložená logická veta vyjadrujúca postupne päť výrokov študentov, pravdivosť iba štyroch výrokov a tvrdenie učiteľa:*

$$\begin{gathered}(P_1 \leftrightarrow F \oplus O) \wedge (P_2 \leftrightarrow R \oplus S) \wedge (P_3 \leftrightarrow J \oplus F)\\ \wedge(P_4 \leftrightarrow O \oplus J) \wedge (P_5 \leftrightarrow O \oplus R)\\ \wedge =_4 (P_1, P_2, P_3, P_4, P_5)\\ \wedge(R \rightarrow O)\end{gathered}$$

Túto reprezentáciu je možné transformovať na tvar CNF:

$$\begin{array}{ll} & (\neg P_1 \vee F \vee O) \wedge (\neg P_1 \vee \neg F \vee \neg O) \wedge (\neg F \vee O \vee P_1) \wedge (F \vee \neg O \vee P_1)\\ \wedge & (\neg P_2 \vee R \vee S) \wedge (\neg P_2 \vee \neg R \vee \neg S) \wedge (\neg R \vee S \vee P_2) \wedge (R \vee \neg S \vee P_2)\\ \wedge & (\neg P_3 \vee J \vee F) \wedge (\neg P_3 \vee \neg J \vee \neg F) \wedge (\neg J \vee F \vee P_3) \wedge (J \vee \neg F \vee P_3)\\ \wedge & (\neg P_4 \vee O \vee J) \wedge (\neg P_4 \vee \neg O \vee \neg J) \wedge (\neg O \vee J \vee P_4) \wedge (O \vee \neg J \vee P_4)\\ \wedge & (\neg P_5 \vee O \vee R) \wedge (\neg P_5 \vee \neg O \vee \neg R) \wedge (\neg O \vee R \vee P_5)\\ \wedge & (\neg P_1 \vee \neg P_2 \vee \neg P_3 \vee \neg P_4 \vee \neg P_5)\\ \wedge & (P_1 \vee P_2) \wedge (P_1 \vee P_3) \wedge (P_1 \vee P_4) \wedge (P_1 \vee P_5) \wedge (P_2 \vee P_3)\\ \wedge & (P_2 \vee P_4) \wedge (P_2 \vee P_5) \wedge (P_3 \vee P_4) \wedge (P_3 \vee P_5) \wedge (P_4 \vee P_5)\\ \wedge & (\neg R \vee O)\end{array}$$

pričom pre zjednodušenie vety bola z nej vypustená klauzula $(O \vee \neg R \vee P_5)$.

Ilustr. 1.1: Príklad pre reprezentáciu pomocou výrokovej logiky rátorov (celkovo štyri unárne operátory), najčastejšie sa používa operátor *negácie* ($\neg$). Budovať komplexnejšie výrazy umožňujú binárne operátory, ktoré poskytujú možnosť kombinovať prvky logických viet navzájom do zložitejších viet. Aj v prípade týchto binárnych operátorov sa typicky používa iba obmedzená množina operátorov (zo šestnástich možných). Štyri najčastejšie používané operátory sú *konjunkcia* ($\wedge$, AND), *disjunkcia* ($\vee$, OR), *implikácia* ($\rightarrow$) a *ekvivalencia* ($\leftrightarrow$). Príkladmi menej používaných operátorov sú *exkluzívna disjunkcia* ($\oplus$, XOR) alebo *odmietnutie konjunkcie* ($\uparrow$, NAND) či *odmietnutie disjunkcie* ($\downarrow$, NOR).

Zátvorky slúžia na explicitné vyjadrenie štruktúry logických viet. Ak nie sú použité, tak štruktúra je daná pomocou *precedencie* (priority) operátorov. Keďže operátory je možné podľa precedencie zoradiť v zmysle od najprioritnejšieho po najmenej prioritný do postupnosti $\neg$, $\wedge$ a $\uparrow$, $\vee$ a $\downarrow$, $\rightarrow$, $\leftrightarrow$ a $\oplus$ (teda tri dvojice operátorov s rovnakou prioritou), tak potom veta **PRECEDENCIA**

$$P \wedge \neg Q \vee R$$

bude reprezentovať vetu $(P \wedge (\neg Q)) \vee R$ a nie napríklad vetu $P \wedge (\neg(Q \vee R))$ – v prípade, keď je potrebné vyjadriť túto druhú štruktúru, je potrebné explicitne použiť zátvorky.

V prípade, že je reťazený ten istý operátor, precedencia nepomôže. V tomto prípade sa použije pravidlo. že operátory sú asociované sprava doľava a na základe tohto pravidla

$$P \vee Q \vee R$$

bude reprezentovať vetu $P \vee (Q \vee R)$ a analogicky aj $P \rightarrow Q \rightarrow R$ bude reprezentovať $P \rightarrow (Q \rightarrow R)$.

Jedným zo spôsobov ako porovnať zložitosť vyjadrenia logických viet je dĺžka viet. Je definovaná induktívnym spôsobom, kde **DĹŽKA VETY**

- ak F je symbol, tak $L(F) = 1$,
- ak $F = (\odot G)$ a $\odot$ reprezentuje jeden z definovaných unárnych operátorov, tak $L(F) = 1 + L(G)$,
- ak $F = (G \odot H)$ a $\odot$ reprezentuje jeden z definovaných binárnych operátorov, tak $L(F) = 1 + L(G) + L(H)$.

Výroková logika pozná iba dve pravdivostné hodnoty – hodnotu *TRUE* (logická pravda) a hodnotu *FALSE* (logická nepravda). Sémantika definuje vzťah medzi logickými vetami a týmito pravdivostnými hodnotami – sémantika nejakej vety výrokovej logiky definuje, aká pravdivostná hodnota má **PRAVDIVOSTNÉ HODNOTY** byť priradená tejto vete v nejakej interpretácii, teda či táto veta je pravdivá (nadobúda hodnotu TRUE) alebo nepravdivá (nadobúda hodnotu FALSE) v danej interpretácii.

INTERPRETAČNÉ PRAVIDLÁ

Pre ľubovoľnú vetu F a ľubovoľnú interpretáciu I, pravdivostná hodnota F^I, ktorá je priradená vete F interpretáciou I sa určí rekurzívnym spôsobom:

- ak F je symbol, potom F^I je dané interpretáciou I,
- ak $F = \bot$, tak F^I = FALSE,
- ak $F = \top$, tak F^I = TRUE,
- ak $F = (\odot G)$ a $\odot$ reprezentuje jeden z definovaných unárnych operátorov, tak $F^I = \odot(G^I)$,
- ak $F = (G \odot H)$ a $\odot$ reprezentuje jeden z definovaných binárnych operátorov, tak $F^I = (G^I) \odot (H^I)$.

Pravdivostná hodnota symbolu, ktorý nie je špeciálny, teda závisí od použitej interpretácie, ktorá reprezentuje pravdivosť daného symbolu v tejto interpretácii. Naproti tomu špeciálny symbol $\top$ je pravdivý v každej interpretácii a $\bot$ je zase v každej interpretácii nepravdivý.

Interpretácia je vlastne funkciou, ktorá symbolom, použitým vo vete, priraďuje jednu z dvoch možných pravdivostných hodnôt. Ak interpretácia priraďuje hodnotu iba niektorým a nie všetkým symbolom vety, hovoríme o *parciálnej* interpretácii. Niekedy pre určenie pravdivostnej hodnoty vety postačuje aj parciálna interpretácia, nie je potrebné interpretovať všetky symboly.

PRAVDIVOSTNÉ TABUĽKY

Pravdivosť zloženej vety vytvorenej niektorým z operátorov je daná pravdivosťou zložiek tejto vety a predpisom, ako sa z pravdivostných hodnôt zložiek odvádza výsledná pravdivostná hodnota. Takýto predpis je definovaný osobitne pre každý operátor vo forme *pravdivostnej tabuľky*. Tab. 1.2 uvádza niektoré používané operátory[1].

Príklad 1.1 Ak by sme teda napríklad chceli zistiť pravdivostnú hodnotu vety tvaru $P \wedge (\neg(Q \vee R))$ zloženej z troch symbolov a troch operátorov, tak by sme postupne získali

$$\begin{aligned}(P \wedge (\neg(Q \vee R)))^I &= P^I \wedge (\neg(Q \vee R))^I \\ &= P^I \wedge (\neg(Q \vee R)^I) \\ &= P^I \wedge (\neg(Q^I \vee R^I))\end{aligned}$$

[1]Keďže existujú dve pravdivostné hodnoty, je možné definovať $2^2 = 4$ rôzne unárne operátory a $4^2 = 16$ rôznych binárnych operátorov.

X	Y	$\neg$ X	X $\wedge$ Y	X $\vee$ Y	X $\rightarrow$ Y	X $\leftrightarrow$ Y
FALSE	FALSE	TRUE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE

X	Y	X $\oplus$ Y	X $\uparrow$ Y	X $\downarrow$ Y
FALSE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	TRUE	TRUE	FALSE
TRUE	TRUE	FALSE	FALSE	FALSE

Tab. 1.2: Pravdivostné tabuľky pre logické operátory

a napríklad pri interpretácii $I = \{P^I$=TRUE, Q^I=FALSE, R^I=TRUE$\}$ by po dosadení bolo:

$$\begin{aligned} TRUE \wedge (\neg(FALSE \vee TRUE)) &= TRUE \wedge (\neg TRUE) \\ &= TRUE \wedge FALSE \\ &= FALSE \end{aligned}$$

zatiaľ čo interpretácia $I = \{P^I$=TRUE, Q^I=FALSE, R^I=FALSE$\}$ by viedla na opačnú pravdivostnú hodnotu:

$$\begin{aligned} TRUE \wedge (\neg(FALSE \vee FALSE)) &= TRUE \wedge (\neg FALSE) \\ &= TRUE \wedge TRUE \\ &= TRUE \end{aligned}$$

Pre určenie hodnoty vety $P \wedge (\neg(Q \vee R))$ by stačila aj parciálna interpretácia $I = \{P^I = TRUE, Q^I = TRUE\}$ avšak parciálna interpretácia $I = \{P^I = TRUE, Q^I = FALSE\}$ bez interpretovania symbolu R je nepostačujúca.

•

Pravdivostná tabuľka pre nejakú logickú vetu reprezentuje sémantiku tejto vety zobrazením pravdivostnej hodnoty pre každú možnú interpretáciu. Ak veta obsahuje n rôznych symbolov, tak tabuľka má $n + 1$ stĺpcov (n stĺpcov pre reprezentáciu interpretácie a jeden pre reprezentáciu vety ako celku) a 2^n riadkov (každý riadok pre inú interpretáciu). V prípade zložitejších viet je možné pridať aj stĺpce pre podvýrazy vety.

Príklad 1.2 Vete $P \wedge (\neg(Q \vee R))$ by zodpovedala nasledujúca tabuľka s $2^3 = 8$ riadkami, pričom bol do nej pridaný aj jeden stĺpec pre hodnoty podvýrazu pre jednoduchšie určenie hodnôt pôvodnej vety, ktoré sú uvedené v poslednom stĺpci.

P	Q	R	$\neg(Q \vee R)$	$P \wedge (\neg(Q \vee R))$
FALSE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	TRUE	FALSE	FALSE
FALSE	TRUE	FALSE	FALSE	FALSE
FALSE	TRUE	TRUE	FALSE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
TRUE	FALSE	TRUE	FALSE	FALSE
TRUE	TRUE	FALSE	FALSE	FALSE
TRUE	TRUE	TRUE	FALSE	FALSE

Tabuľka pre každú z ôsmich kombinácií hodnôt vstupných symbolov obsahuje pravdivostnú hodnotu vety ako celku. •

Vlastnosti binárnych operátorov

Na základe pravdivostných tabuliek je možné určiť základné vlastnosti jednotlivých operátorov. Pre binárne operátory to sú tieto vlastnosti:

- *komutatívnosť* (nezáleží na poradí vstupov): $P \odot Q = Q \odot P$
- *asociatívnosť* (združovanie): $P \odot (Q \odot R) = (P \odot Q) \odot R$
- *idempotencia* (zloženie argumentu s ním samým): $P \odot P = P$
- *zachovanie pravdivosti* (ak oba vstupy sú $\top$ tak aj výstup je $\top$): $\top \odot \top = \top$
- *zachovanie nepravdivosti* (ak oba vstupy sú $\perp$ tak aj výstup je $\perp$): $\perp \odot \perp = \perp$

Vlastnosti jednotlivých binárnych operátorov sú uvedené v Tab. 1.3, kde je vyznačené či jednotlivé operátory dané vlastnosti majú alebo nie.

Možné svety a modely

Jednotlivé interpretácie sa líšia pravdivostnými hodnotami symbolov, pričom každá z nich definuje pravdivostné hodnoty pre všetky použité symboly – interpretácia tak definuje *možný svet*, ktorý je reprezentovaný jednou kombináciou pravdivostných hodnôt použitých symbolov. Keďže každý symbol môže mať iba jednu z dvoch možných hodnôt, tak ak je použitých n rôznych symbolov, potom existuje 2^n rôznych interpretácií. Tá interpretácia, v ktorej je nejaká veta pravdivá, sa označuje ako *model* danej vety.

	$\wedge$	$\vee$	$\rightarrow$	$\leftrightarrow$	$\oplus$	$\uparrow$	$\downarrow$
komutatívnosť	áno	áno	nie	áno	áno	áno	áno
asociatívnosť	áno	áno	nie	áno	áno	nie	nie
idempotencia	áno	áno	nie	nie	nie	nie	nie
zachovanie pravdivosti	áno	áno	áno	áno	nie	nie	nie
zachovanie nepravdivosti	áno	áno	nie	nie	áno	nie	nie

Tab. 1.3: Základné vlastnosti binárnych operátorov

Množina modelov zloženej vety sa dá odvodiť z množín modelov tých viet, z ktorých je daná zložená veta vytvorená. Platí napríklad

$$\begin{aligned} \mathcal{M}(A) &= \mathcal{S}(A) \setminus \mathcal{M}(\neg A) \\ \mathcal{M}(A \wedge B) &= \mathcal{M}(A) \cap \mathcal{M}(B) \\ \mathcal{M}(A \vee B) &= \mathcal{M}(A) \cup \mathcal{M}(B) \end{aligned}$$

kde $\mathcal{S}(F)$ je množina všetkých svetov, ktoré môžu byť interpretáciami symbolov logickej vety F (a teda pre každú interpretáciu symbolov vety F platí $I \in \mathcal{S}(F)$) a $\mathcal{M}(F)$ je množina modelov tejto logickej vety F.

Splniteľnosť a validnosť

Ak pre nejakú vetu existuje model, potom táto veta sa označuje ako *splniteľná*. Ak pre ňu žiadny model neexistuje (teda nie je pravdivá pri žiadnej interpretácii), tak sa označuje ako *nesplniteľná*. Veta, ktorá je pravdivá pri ľubovoľnej interpretácii a teda každá možná interpretácia je jej modelom, sa označuje ako *validná* a nazýva sa *tautológia*. Príkladom tautológie je veta $A \rightarrow (B \rightarrow (A \rightarrow B))$, čo možno overiť pomocou pravdivostnej tabuľky pre túto vetu. Ak pre nejakú vetu existuje svet, ktorý nie je jej modelom, tak sa táto veta označuje ako *falzifikovateľná*.

Ak je veta F validná, tak $\neg F$ je vetou nesplniteľnou. Ak je F splniteľnou vetou, $\neg F$ musí byť falzifikovateľnou. Teda validná veta je zároveň aj splniteľná, nesplniteľná veta je zároveň aj falzifikovateľná a splniteľná veta, ktorá nie je validná, je zároveň aj falzifikovateľná.

Ekvivalentnosť viet

V prípade, že dve vety, používajúce rovnaké symboly, nadobúdajú pre každú možnú interpretáciu rovnaké pravdivostné hodnoty (teda im prislúchajú rovnaké pravdivostné tabuľky resp. obe vety majú rovnaké množiny modelov), tak potom tieto vety sú navzájom *logicky ekvivalentné*. Ekvivalentné formuly môžu teda mať rôzne literály. Ako znak ekvivalentnosti dvoch viet sa používa $\equiv$.

Ak dve vety používajú rôzne množiny symbolov, tak tieto vety sú logicky ekvivalentné vtedy, ak rovnaké interpretácie spoločných symbolov vedú na rovnaké pravdivostné hodnoty oboch viet. Napríklad platí ekvivalentnosť medzi vetami

$$(C \land \neg C) \lor A \equiv A \land (B \lor \neg B)$$

kde spoločným symbolom je A a pre $A^I = TRUE$ sú obe vety pravdivé a pre interpretáciu $A^I = FALSE$ sú obe vety nepravdivé (nezávisle od interpretácie symbolov, ktoré nie sú spoločnými).

Pomocou pravdivostných tabuliek je možné overiť napríklad komutatívnosť konjunkcie či disjunkcie alebo na druhej strane ukázať, že implikácia nie je komutatívna. Niektoré ekvivalencie sú uvedené v Tab. 1.4.

Typ ekvivalencie	Príklady ekvivalencie
zjednodušenie	$P \lor P \equiv P$ $\quad P \land P \equiv P$ $P \to P \equiv \top$ $\quad P \leftrightarrow P \equiv \top$ $P \downarrow P \equiv \neg P$ $\quad P \uparrow P \equiv \neg P$ $P \oplus P \equiv \bot$
absorpcia symbolu	$P \lor \bot \equiv P$ $\quad P \land \top \equiv P$ $\quad P \oplus \top \equiv \neg P$ $P \lor \top \equiv \top$ $\quad P \land \bot \equiv \bot$ $\quad P \oplus \bot \equiv P$ $P \to \top \equiv \top$ $\quad \bot \to P \equiv \top$ $\quad P \leftrightarrow \bot \equiv \neg P$ $P \to \bot \equiv \neg P$ $\quad \top \to P \equiv P$ $\quad P \leftrightarrow \top \equiv P$
transpozícia	$P \to Q \equiv \neg Q \to \neg P$
AND eliminácia	$P \land Q \equiv P, Q$
absorpcia	$P \land (P \lor Q) \equiv P$ $\quad P \land (\neg P \lor Q) \equiv P \land Q$ $P \lor (P \land Q) \equiv P$ $\quad P \lor (\neg P \land Q) \equiv P \lor Q$
distributívnosť	$P \land (Q \lor R) \equiv (P \land Q) \lor (P \land R)$ $P \lor (Q \land R) \equiv (P \lor Q) \land (P \lor R)$
kompozícia	$(P \to Q) \land (P \to R) \equiv P \to (Q \land R)$
De Morganove pravidlá	$\neg(P \land Q) \equiv \neg P \lor \neg Q$ $\neg(P \lor Q) \equiv \neg P \land \neg Q$

Tab. 1.4: Vybrané príklady ekvivalentnosti viet

Vďaka ekvivalentnosti viet je možné jeden operátor nahradiť pomocou iných operátorov. Príkladmi takejto ekvivalentnosti sú:

- $P \leftrightarrow Q : (P \to Q) \land (Q \to P)$
- $P \to Q : \neg P \lor Q$ alebo $P \uparrow \neg Q$
- $P \uparrow Q : P \to \neg Q$ alebo $\neg P \lor \neg Q$

- $P \downarrow Q : \neg P \wedge \neg Q$ alebo $\neg(\neg Q \rightarrow P)$
- $P \oplus Q : (P \vee Q) \wedge \neg(P \wedge Q)$ alebo $\neg(P \rightarrow Q) \vee \neg(Q \rightarrow P)$

Následkom toho nie je nutné používať všetky možné operátory, pretože namiesto nejakého operátora môže byť použité jeho vyjadrenie pomocou iných operátorov. Taká podmnožina operátorov, ktorá dokáže ekvivalentne vyjadriť všetky vety reprezentovateľné pomocou výrokového počtu a nemôže byť redukovaná odobraním nejakého operátora z tejto množiny, sa označuje ako *minimálna* funkčne úplná množina. Existujú takéto dve jednoprvkové množiny $\{\uparrow\}$ a $\{\downarrow\}$, z dvojprvkových najznámejšími sú dvojice $\{\neg, \vee\}$ a $\{\neg, \wedge\}$, menej známou dvojicou je $\{\neg, \rightarrow\}$.

MINIMÁLNA MNOŽINA OPERÁTOROV

Použitie logiky pre reprezentáciu znalostí nezaručuje automaticky, že získané výsledky budú správne. Výroková logika predstavuje formálny systém, garantujúci platnosť záverov – ale iba v prípade platnosti predpokladov, z ktorých sa vychádza. Pre praktické použitie je preto dôležité, akým spôsobom budú symboly ako aj zložené vety *ukotvené* v realite (reprezentovanej doméne). Reprezentácia faktov a vzťahov, platiacich v nejakej doméne, do tvaru logických viet nie je nijako špeciálne náročná, treba však mať na mysli rozdiely medzi významom týchto faktov a vzťahov (najmä ak nie sú vyjadrené prísne formálnym spôsobom) a sémantikou reprezentačných nástrojov, ponúkaných výrokovou logikou.

UKOTVENIE V REALITE

Voľba symbolov môže byť rôzna, avšak mala by sa riadiť účelom reprezentácie a voľbou vhodnej granularity. Je síce možné jedným symbolom reprezentovať ľubovoľnú časť domény, avšak nie vždy to je najvhodnejšia voľba.

GRANULARITA

Príklad 1.3 Ukážme si ukotvenie na ilustračnom prípade zo strany 2. Je možných niekoľko príkladov ukotvenia symbolov:

> "Fero podvádza" – táto voľba umožňuje uvažovať o Ferovom podvádzaní samostatne a prípadne ho pomocou zložitejších viet spájať s inými skutočnosťami.
>
> "Fero alebo Ondro podvádzajú" – neumožňuje pracovať s Ferovým podvádzaním bez uvažovania Ondrovho podvádzania, nedá sa vyjadriť súvislosť so symbolom reprezentujúcim "Jaro alebo Fero podvádzajú", hoci tieto dva symboly sú navzájom závislé cez Ferovo podvádzanie.
>
> "Stanovo tvrdenie" – vhodné v prípade, že je potrebné uvažovať o Stanovom tvrdení ako celku, teda o tom aká je jeho pravdivosť (re-

prezentovať "Učiteľovo tvrdenie" je možné avšak zbytočné, keďže jeho pravdivosť je známa).

Vhodnou voľbou v danom prípade by teda bolo reprezentovať podvádzania osobitne (napr. symbol F pre "Fero podvádza", O pre "Ondro podvádza", atď.) a aj výroky ako celky okrem šiesteho výroku (napr. symbol P_1 pre prvý výrok – Stanove tvrdenie, P_2 pre druhý výrok, atď.). Hoci symboly môžu byť označené ľubovoľne, je vhodné použiť označenia evokujúce to, čo daný symbol označuje. •

KÓDOVANIE "A", "AJ"

Tvrdenia, využívajúce spojky typu "a" či "aj" s významom existencie viacerých konceptov, ktoré sú súčasne pravdivé, je možné reprezentovať pomocou $\wedge$ operátora. Napríklad súčasná pravdivosť Stanovho a Ferovho výroku by mohla byť reprezentovaná ako $P_1 \wedge P_2$, kde symbol P_1 reprezentuje Stanov výrok a symbol P_2 zase Ferov výrok.

KÓDOVANIE "ALEBO"

Zložitejšie je to v prípade spojky "alebo" – tejto spojke môžu zodpovedať dva operátory. Jedným je $\vee$, ktorý sa niekedy označuje ako *inkluzívna* disjunkcia so zdôraznením, že zahŕňa nielen platnosť každého z argumentov ale aj ich súčasnú platnosť. Na rozdiel od neho operátor $\oplus$ označovaný ako *exkluzívna* disjunkcia zahŕňa platnosť každého z argumentov samostatne ale vylučuje platnosť oboch argumentov súčasne. To, ktorý z nich má byť použitý, závisí od toho ako má byť interpretovaný prípad súčasnej platnosti argumentov. Keďže Stanov výrok P_1 explicitne vylučuje, aby Fero a Ondro podvádzali súčasne, tak toto tvrdenie je možné reprezentovať ako

$$F \oplus O \equiv (F \vee O) \wedge \neg(F \wedge O)$$

s významom, že platí že Fero alebo Ondro podvádza a zároveň neplatí, že podvádzajú obaja súčasne.

KÓDOVANIE "AK-POTOM"

Podobne je potrebné byť obozretným aj v prípade viet tvaru "ak … potom …" vyjadrujúcich, že nejaký predpoklad má za následok nejaký záver. Aj v tomto prípade sú možné dve rôzne chápania zmyslu tvrdenia. Prvé za pravdivé pokladá nielen explicitne vyjadrenú závislosť na platnosti predpokladu ale aj implicitne vyjadrenú závislosť na neplatnosti predpokladu (v prípade ktorej záver neplatí) – to je chápanie v zmysle "vtedy a iba vtedy záver ak predpoklad". V tomto prípade je vhodnou voľbou $\leftrightarrow$ operátor. Druhé chápanie je založené iba na uvažovaní explicitne vyjadrenej závislosti bez nejakého implicitného dodatku pod povrchom. V takomto prípade je voľbou $\rightarrow$ operátor. Ak budeme učiteľov výrok chápať iba v explicitne podanom zmysle, potom šieste tvrdenie bude reprezentované ako

$$R \rightarrow O$$

kde symbol R reprezentuje Rudovo podvádzanie a symbol O analogicky Ondrovo podvádzanie.

Kódovanie platnosti výrokov

Zložitejšia situácia nastáva v prípade výrokov o iných výrokoch. V našom prípade veta $F \oplus O$ zo Stanovho výroku P_1 platí iba vtedy, ak Stanov výrok je pravdivý. Ak je však jeho výrok nepravdivý, tak táto veta nie je pravdivá. Na druhej strane, ak je pravdivý obsah Stanovho výroku P_1, tak je pravdivý aj samotný výrok. Preto Stanov výrok je potrebné reprezentovať ako

$$P_1 \leftrightarrow F \oplus O \equiv (P_1 \to (F \vee O) \wedge \neg(F \wedge O)) \wedge ((F \vee O) \wedge \neg(F \wedge O) \to P_1)$$

Kódovanie "Aspoň" / "Najviac jeden"

Požiadavka, aby jedno z piatich tvrdení bolo nepravdivé, sa dá vyjadriť ako dvojica *kardinalitných ohraničení* (viac o kódovaní týchto ohraničení v kap. 3) – aspoň jedno tvrdenie je nepravdivé a súčasne najviac jedno z tvrdení je nepravdivé. Vyjadriť prvé je možno jednoducho pomocou

$$\neg P_1 \vee \neg P_2 \vee \neg P_3 \vee \neg P_4 \vee \neg P_5$$

zatiaľ čo vyjadrenie druhého ohraničenia je zložitejšie. Je možné vyjsť z toho, že žiadne dve z tvrdení nesmú byť súčasne nepravdivé (a teda v rámci každej možnej dvojice tvrdení musí byť aspoň jedno tvrdenie pravdivé).

Reprezentácia viet

Logické vety je možné reprezentovať viacerými spôsobmi – nielen ako výraz ľubovoľného tvaru, ktorý je prípustný definíciou syntaxe výrokového počtu. Niektorými ďalšími možnosťami sú *pravdivostná tabuľka*, *suma mintermov*, *súčin maxtermov*, *Karnaughova mapa*, *Vennove diagramy*, atď. Vďaka ekvivalentnosti logických výrazov je možné logické vety transformovať z jedného reprezentačného tvaru na iný tvar. Ako príklad použime Stanov výrok, ktorý má tvar $P_1 \leftrightarrow F \oplus O$.

P_1	F	O	veta	mintermy	maxtermy
FALSE	FALSE	FALSE	TRUE	$\neg P_1 \wedge \neg F \wedge \neg O$	
FALSE	FALSE	TRUE	FALSE		$P_1 \vee F \vee \neg O$
FALSE	TRUE	FALSE	FALSE		$P_1 \vee \neg F \vee O$
FALSE	TRUE	TRUE	TRUE	$\neg P_1 \wedge F \wedge O$	
TRUE	FALSE	FALSE	FALSE		$\neg P_1 \vee F \vee O$
TRUE	FALSE	TRUE	TRUE	$P_1 \wedge \neg F \wedge O$	
TRUE	TRUE	FALSE	TRUE	$P_1 \wedge F \wedge \neg O$	
TRUE	TRUE	TRUE	FALSE		$\neg P_1 \vee \neg F \vee \neg O$

Tab. 1.5: Mintermy a maxtermy

Reprezentácia pomocou pravdivostnej tabuľky je ilustrovaná v ľavej časti Tab. 1.5. Pravdivostná tabuľka obsahuje $3+1$ stĺpcov kvôli trom symbolom v reprezentovanej vete. Každému zo symbolov prislúcha jeden stĺpec s rôznymi pravdivostnými hodnotami priradenými danému symbolu. V tabuľke je 8 riadkov, toľko je navzájom rôznych kombinácií pravdivostných hodmôt symbolov – každý riadok teda zodpovedá jednému možnému svetu. Posledný stĺpec vyjadruje pravdivostnú hodnotu reprezentovanej vety v danom svete. Uvedenie hodnoty TRUE znamená, že príslušný svet je modelom danej vety.

Minterm pre nejaký svet je daný ako konjunkcia všetkých symbolov, pričom ak v tomto svete nejaký symbol má priradenú hodnotu TRUE, tak sa uvažuje symbol priamo, zatiaľ čo pri hodnote FALSE sa uvažuje jeho negácia. Musí teda obsahovať všetky symboly či už v priamej alebo komplementárnej forme.

Naproti tomu *maxterm* nejakého sveta je negovaný minterm toho sveta. Je to teda disjunkcia všetkých symbolov vyjadrených priamo alebo v komplementárnej forme (opačne ako pri minterme), získaná pri použití De Morganovho pravidla $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$ a asociatívnosti konjunkcie a disjunkcie. Príkladom je druhý riadok Tab. 1.5

$$\begin{aligned}
\neg(\neg P_1 \wedge \neg F \wedge O) &\equiv \neg(\neg P_1 \wedge (\neg F \wedge O)) \\
&\equiv P_1 \vee \neg(\neg F \wedge O) \\
&\equiv P_1 \vee (F \vee \neg O) \\
&\equiv P_1 \vee F \vee \neg O
\end{aligned}$$

Minterm daného sveta teda reprezentuje daný svet, zatiaľ čo maxterm daného sveta reprezentuje všetky svety okrem daného.

DISJUNKTÍVNA NORMÁLNA FORMA

Spojením všetkých tých mintermov, ktoré prislúchajú tým svetom, ktoré sú modelmi danej vety, vznikne DNF (*disjunktívna normálna forma*). Táto forma má tieto charakteristické vlastnosti:

- používajú sa iba dva binárne operátory a to $\wedge$ a $\vee$, iné binárne operátory nie sú povolené,
- negované môžu byť iba symboly, zložené vety nie sú negované,
- konjunkcia je prípustná v rámci disjunkcie avšak disjunkcia v rámci konjunkcie prípustná nie je.

Jedná sa teda o disjunkciu konjunktov. Nie je však nutné aby každý konjunkt bol konjunkciou všetkých symbolov – v prípade, že DNF obsahuje dva konjunkty, ktoré sa líšia iba tým, že v jednom sa vyskytuje nejaký symbol priamo a v druhom zase v komplementárnej forme, tak je možné použiť zjednodušenie:

$$(P \wedge Q \wedge R) \vee (P \wedge \neg Q \wedge R) \equiv P \wedge R$$

Každý konjunkt teda reprezentuje jeden alebo niekoľko modelov – presne 2^n modelov, kde n je počet symbolov, ktoré boli z konjunktu vypustené.

Veta, ktorá je reprezentovaná tabuľkou Tab. 1.5 bude v DNF tvare pozostávať zo štyroch konjunktov, pričom každý z nich reprezentuje jeden model tejto vety

$$(\neg P_1 \wedge \neg F \wedge \neg O) \vee (\neg P_1 \wedge F \wedge O) \vee (P_1 \wedge \neg F \wedge O) \vee (P_1 \wedge F \wedge \neg O)$$

Spojením všetkých tých maxtermov, ktoré prislúchajú tým svetom, ktoré nie sú modelom danej vety, vznikne CNF (*konjunktívna normálna forma*). Táto forma má charakteristické vlastnosti: Konjunktívna normálna forma

- používajú sa iba dva binárne operátory a to $\wedge$ a $\vee$, iné binárne operátory nie sú povolené,
- negované môžu byť iba symboly, zložené vety nie sú negované,
- disjunkcia je prípustná v rámci konjunkcie avšak konjunkcia v rámci disjunkcie prípustná nie je.

Prvé dve vlastnosti sú rovnaké ako pri DNF, zatiaľ čo posledná vlastnosť definuje, že CNF má tvar konjunkcie disjunktov. Analogicky ani teraz nie je nutné, aby každý disjunkt bol disjunkciou všetkých symbolov – v prípade, že CNF obsahuje dva disjunkty, ktoré sa líšia iba tým, že v jednom sa vyskytuje nejaký symbol priamo a v druhom negovane, tak je možné použiť zjednodušenie:

$$(P \vee Q \vee R) \wedge (P \vee \neg Q \vee R) \equiv P \vee R$$

Každý disjunkt teda reprezentuje niekoľko svetov (všetky modely vety a aj iné svety, ktoré nie sú modelmi) – presne $2^m - 2^n$ svetov, kde m je celkový počet symbolov a n je počet symbolov, ktoré boli z disjunktu vypustené.

Veta, ktorá je reprezentovaná tabuľkou Tab. 1.5 môže v CNF tvare pozostávať zo štyroch disjunktov, pričom každý z nich reprezentuje sedem svetov (vrátane všetkých štyroch modelov tejto vety)

$$(P_1 \vee F \vee \neg O) \wedge (P_1 \vee \neg F \vee O) \wedge (\neg P_1 \vee F \vee O) \wedge (\neg P_1 \vee \neg F \vee \neg O)$$

CNF je jedným z tých možných tvarov, ktorý sa široko používa. Definícia tejto formy v tvare BNF je v Tab. 1.6 (definícia DNF je podobná iba sa

$<$ veta $>$	$::=$	$<$ klauzula $>$
	$\mid$	$<$ klauzula $>$ $\wedge$ $<$ veta $>$
$<$ klauzula $>$	$::=$	$<$ literál $>$
	$\mid$	$<$ literál $>$ $\vee$ $<$ klauzula $>$
$<$ literál $>$	$::=$	$\neg$ $<$ symbol $>$ $\mid$ $<$ symbol $>$

Tab. 1.6: Konjunktívna normálna forma

navzájom vymenia $\vee$ a $\wedge$ – v štruktúre klauzuly bude $\wedge$ a v štruktúre vety bude $\vee$).

Prípustnými sú všetky vety pozostávajúce z konjunkcie klauzúl, ktoré sú reprezentované ako disjunkcie literálov. Literálom môže byť priamo symbol (*pozitívny* literál) alebo jeho negácia (*negatívny* literál). Ak F je symbolom, potom $\{F, \neg F\}$ je komplementárny pár literálov.

Príkladom syntakticky správnej vety v tvare CNF je $(\neg P \vee Q) \wedge \neg R$. Samostatná konjunkcia $P \wedge Q$ je tiež v tvare CNF (obe klauzuly majú jednotkovú dĺžku) rovnako ako samostatná disjunkcia $P \vee Q$ (iba jedna klauzula) alebo samostatný symbol či už negovaný $\neg P$ alebo bez negácie P (iba jedna klauzula jednotkovej dĺžky).

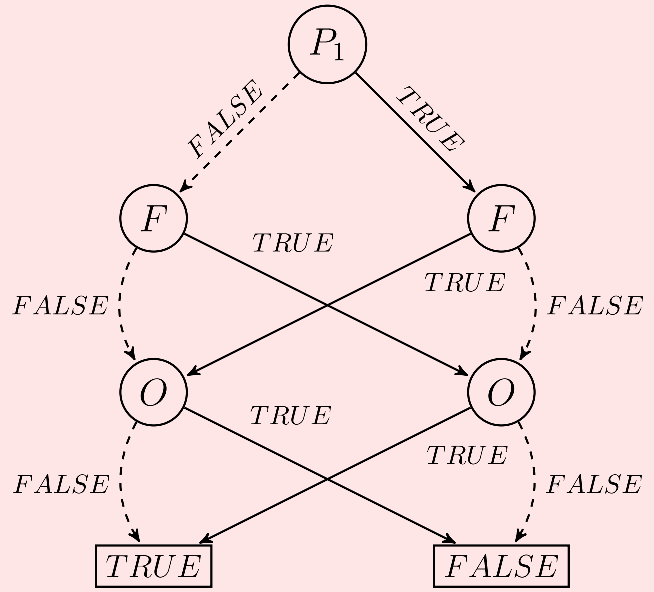
Obe uvedené formy, DNF aj CNF, patria do skupiny NNF (*negatívna normálna forma*). Je to tvar, ktorý je obmedzený na použitie iba troch operátorov ($\neg$, $\vee$, $\wedge$), pričom unárny operátor môže byť aplikovaný iba na symboly avšak nie na zložené vety. Teda DNF aj CNF sú súčasne vždy aj NNF, avšak naopak to neplatí – môže byť veta v tvare NNF, ktorá nie je ani DNF ani CNF.

Binárny rozhodovací diagram

Reprezentantom grafickej metódy pre reprezentáciu logických viet je *binárny rozhodovací diagram*. Logická veta je zobrazená ako orientovaný acyklický graf, ktorý pozostáva z *rozhodovacích* a *terminálnych* uzlov. Každý rozhodovací uzol reprezentuje niektorý symbol a má dvoch potomkov podľa toho, či tento symbol nadobúda hodnotu TRUE alebo FALSE. Tým pádom hrana, vychádzajúca z rozhodovacieho uzla, reprezentuje priradenie niektorej z pravdivostných hodnôt symbolu, reprezentovanému daným uzlom. Každá cesta od koreňa k terminálnemu uzlu reprezentuje priradením hodnôt symbolov jeden alebo viac svetov a hodnota príslušného terminálneho uzla pravdivosť reprezentovanej vety v danom svete.

Ak binárny rozhodovací diagram používa rovnaké poradie symbolov vo všetkých svojich vetvách, tak sa označuje ako *zotriedený*. A ak spĺňa požiadavky

- všetky izomorfné subgrafy boli zlúčené do jedného,



Obr. 1.1: Binárny rozhodovací diagram

- všetky rozhodovacie uzly, ktorých potomkovia sú izomorfní, boli eliminované,

tak sa označuje ako *redukovaný*. Výhodou redukovaného zotriedeného tvaru je jeho jedinečnosť pre nejakú logickú vetu a poradie symbolov (teda porovnaním je možno testovať ekvivalentnosť logických viet – ak dve vety sú reprezentované rovnakými redukovanými zotriedenými binárnymi rozhodovacími diagramami, tak sú ekvivalentné). Obr. 1.1 ilustruje redukovaný zotriedený binárny rozhodovací diagram pre logickú vetu podľa rozhodovacej tabuľky Tab. 1.5. Bol vytvorený nasledujúcim postupom:

1. pre dané poradie symbolov (P_1, F, O) bol vytvorený graf vo forme úplného binárneho rozhodovacieho stromu,

2. redundantné uzly na poslednej úrovni boli zlúčené do neredundantnej množiny uzlov,

3. to isté sa dialo na ostatných úrovniach, pričom sa postupovalo smerom zdola nahor,

4. všetky uzly, ktoré mali iba jedného potomka, boli eliminované.

Uvedený postup je síce jednoduchý avšak s exponenciálnou časovou zložitosťou (kvôli exponenciálne veľkému počtu uzlov úplného rozhodovacieho stromu). Preto sa častejšie používajú zložitejšie postupy vychádzajúce z tvaru logickej vety zapísanej vo výrokovom počte.

PREVOD DO CNF

Ľubovoľná veta vo výrokovom počte môže byť transformovaná na ekvivalentnú vetu v tvare NNF, ktorá môže byť ďalej transformovaná na CNF. Neznamená to však vždy zjednodušenie v zmysle počtu prvkov zápisu, výsledkom môže byť aj nárast zložitosti. Príkladom tohto je veta $(X_1 \wedge Y_1) \vee (X_2 \wedge Y_2)$, ktorej zodpovedá ekvivalentná CNF tvaru

$$(X_1 \vee X_2) \wedge (X_1 \vee Y_2) \wedge (Y_1 \vee X_2) \wedge (Y_1 \vee Y_2)$$

ALGORITMUS PRE PREVOD DO CNF

Naivný algoritmus pre transformáciu logickej vety do tvaru CNF pozostáva zo štyroch krokov[2] (Alg. 1.1), pričom každý z nich je založený na využití jedného alebo viacerých *odvodzovacích pravidiel*.

vstup: veta F v ľubovoľnom tvare
výstup: veta F v CNF tvare

1. F := *eliminácia_ekvivalencií(F)*
2. F := *eliminácia_implikácií(F)*
3. F := *vnorenie_negácií(F)*
4. F := *úprava_na_CNF(F)*

Alg. 1.1: Transformácia do CNF

Ekvivalencie sú eliminované na základe

$$P \leftrightarrow Q \;\equiv\; (P \rightarrow Q) \wedge (Q \rightarrow P)$$

zatiaľ čo implikácie je možné odstrániť využitím ekvivalentnosti

$$P \rightarrow Q \;\equiv\; \neg P \vee Q$$

[2]Ak veta obsahuje aj ďalšie operátory ako $\oplus$, $\uparrow$ a $\downarrow$, tak je potrebné pridať ďalšie eliminačné kroky, aby vo vete ostali iba operátory $\neg$, $\vee$ a $\wedge$.

Pre realizáciu tretieho kroku je potrebné využiť De Morganove pravidlá

$$\neg(P \wedge Q) \equiv \neg P \vee \neg Q \qquad \neg(P \vee Q) \equiv \neg P \wedge \neg Q$$

a pravidlo dvojitej negácie

$$\neg(\neg P) \equiv P$$

Výsledná úprava na CNF tvar je založená na distributívnosti konjunkcie nad disjunkciou respektívne disjunkcie nad konjunkciou

$$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R) \qquad P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$$

ako aj na komutatívnosti a asociatívnosti konjunkcie a disjunkcie.

Príklad 1.4 Ako ukážku postupnej úpravy je možné upraviť prepis Stanovho výroku z ilustračného príkladu na strane 2, ktorý má tvar

$$P_1 \leftrightarrow (F \oplus O) \equiv P_1 \leftrightarrow (F \vee O) \wedge \neg(F \wedge O)$$

ktorý sa náhradou elvivalencie prevedie na tvar vytvorený konjunkciou dvoch implikácií

$$(P_1 \rightarrow (F \vee O) \wedge \neg(F \wedge O)) \wedge ((F \vee O) \wedge \neg(F \wedge O) \rightarrow P_1)$$

Pre prvú časť výrazu (ľavá implikácia) sa po nahradení implikácie získa $\neg P_1 \vee ((F \vee O) \wedge \neg(F \wedge O))$, prechod na tvar $\neg P_1 \vee ((F \vee O) \wedge (\neg F \vee \neg O))$ bol dosiahnutý aplikáciou jedného z De Morganových pravidiel a následné $(\neg P_1 \vee (F \vee O)) \wedge (\neg P_1 \vee (\neg F \vee \neg O))$ je výsledkom využitia distributívnosti disjunkcie nad konjunkciou. Následným využitím asociatívnosti sa konečne získa $(\neg P_1 \vee F \vee O) \wedge (\neg P_1 \vee \neg F \vee \neg O)$.

Pre druhú časť výrazu (pravá implikácia) sa po nahradení implikácie získa $\neg((F \vee O) \wedge \neg(F \wedge O)) \vee P_1$. De Morganove pravidlá a pravidlo dvojitej negácie poskytnú $((\neg F \wedge \neg O) \vee (F \wedge O)) \vee P_1$ čo použitím distributívnych zákonov prejde na $((\neg F \vee (F \wedge O)) \wedge (\neg O \vee (F \wedge O))) \vee P_1$ a opakovaným použitím distributívnych zákonov na $((\neg F \vee O) \wedge (\neg O \vee F)) \vee P_1$. Záverečné použitie distributívnosti a následne asociatívnosti vedie na výsledný tvar $(\neg F \vee O \vee P_1) \wedge (\neg O \vee F \vee P_1)$.

Stanov výrok $P_1 \leftrightarrow (F \oplus O)$ je teda možné transformovať do ekvivalentnej formy vyjadrenej v tvare CNF, ktorá bude celkove pozostávať z konjunkcie štyroch disjunktov (klauzúl)

$$(\neg P_1 \vee F \vee O) \wedge (\neg P_1 \vee \neg F \vee \neg O) \wedge (\neg F \vee O \vee P_1) \wedge (\neg O \vee F \vee P_1)$$

TSEITINOVA TRANSFORMÁCIA

Inou alternatívou pre prevod logického výrazu do CNF je *Tseitinova transformácia*, ktorá na rozdiel od predchádzajúceho algoritmu je založená na rozklade logickej vety na podvýrazy a pridávaní nových pomocných symbolov, reprezentujúcich tieto podvýrazy. Samotná transformácia pozostáva zo štyroch krokov, ktoré sú uvedené v Alg. 1.2.

vstup: veta F v ľubovoľnom tvare
výstup: veta F v CNF tvare

1. F := *parsovanie_vety(F)*
2. F := *doplnenie_pomocných_symbolov(F)*
3. F := *vytvorenie_ekvivalencií(F)*
4. F := *úprava_klauzúl_na_CNF(F)*

Alg. 1.2: Tseitinova transformácia do CNF

Pre ilustráciu tejto transformácie je možné opäť použiť prepis Stanovho výroku z ilustračného príkladu na strane 2, ktorý má tvar

$$P_1 \leftrightarrow (F \oplus O) \equiv P_1 \leftrightarrow (F \vee O) \wedge \neg(F \wedge O)$$

V rámci prvého kroku algoritmu je tento tvar možné reprezentovať *parsovacím* stromom, ktorý je znázornený na Obr. 1.2. Listové uzly tohto stromu reprezentujú jednotlivé symboly, ktoré boli použité vo vete, zatiaľ čo koreňový uzol a medziľahlé uzly reprezentujú podvýrazy. Strom ako celok je reprezentáciou daného výrazu.

Keďže strom obsahuje päť uzlov, ktoré reprezentujú jednotlivé identifikované podvýrazy, tak v druhom kroku algoritmu je potrebné doplniť k pôvodnej sade troch symbolov (P_1, F a O) ešte päť nových pomocných symbolov: S_1, S_2, S_3, S_4 a S_5.

V rámci tretieho kroku každému podvýrazu bude priradený jeden z nových symbolov – takto každý nový symbol bude reprezentovať príslušný podvýraz, čiže symbol bude ekvivalentný s daným podvýrazom. Pre ukážový príklad sa získa päť ekvivalencií

$$\begin{aligned} S_1 &\leftrightarrow (F \wedge O) \\ S_2 &\leftrightarrow \neg S_1 \\ S_3 &\leftrightarrow (F \vee O) \\ S_4 &\leftrightarrow (S_3 \wedge S_2) \\ S_5 &\leftrightarrow (P_1 \leftrightarrow S_4) \end{aligned}$$

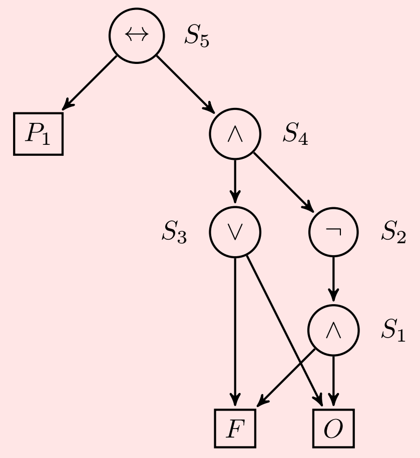
Pretože musia platiť všetky tieto ekvivalencie a zároveň je požadovaná aj platnosť pôvodnej vety (reprezentovanej vlastne symbolom S_5), tak pôvodnú vetu je možné vyjadriť ako konjunkciu

$$\begin{aligned} S_5 \quad \wedge \quad & (S_5 \leftrightarrow (P_1 \leftrightarrow S_4)) \wedge (S_4 \leftrightarrow (S_3 \wedge S_2)) \wedge \\ & (S_3 \leftrightarrow (F \vee O)) \wedge (S_2 \leftrightarrow \neg S_1) \wedge \\ & (S_1 \leftrightarrow (F \wedge O)) \end{aligned}$$

Záverečným krokom algoritmu je upraviť jednotlivé podvýrazy do tvaru CNF, čo je vlastne už iba rutinná úloha. Na ich úpravu je možné použiť predchádzajúci algoritmus Alg. 1.1 alebo použiť predpripravené úpravy v Tab. 1.7.

Po úprave teda Stanova veta bude v CNF forme vyjadrená ako

$$S_5 \quad \wedge \quad (\neg S_5 \vee \neg P_1 \vee S_4) \wedge (\neg S_5 \vee \neg S_4 \vee P_1) \wedge (P_1 \vee S_4 \vee S_5) \wedge (\neg P_1 \vee \neg S_4 \vee S_5)$$



Obr. 1.2: Rozparsovaný Stanov výrok

Typ ekvivalencie	CNF (z $\rightarrow$)	CNF (z $\leftarrow$)
$S \leftrightarrow \neg A$	$(\neg S \vee \neg A)$	$(A \vee S)$
$S \leftrightarrow (A \vee B)$	$(\neg S \vee A \vee B)$	$(\neg A \vee S) \wedge (\neg B \vee S)$
$S \leftrightarrow (A \wedge B)$	$(\neg S \vee A) \wedge (\neg S \vee B)$	$(\neg A \vee \neg B \vee S)$
$S \leftrightarrow (A \rightarrow B)$	$(\neg S \vee \neg A \vee B)$	$(A \vee S) \wedge (\neg B \vee S)$
$S \leftrightarrow (A \uparrow B)$	$(\neg S \vee \neg A \vee \neg B)$	$(A \vee S) \wedge (B \vee S)$
$S \leftrightarrow (A \downarrow B)$	$(\neg S \vee \neg A) \wedge (\neg S \vee \neg B)$	$(A \vee B \vee S)$
$S \leftrightarrow (A \leftrightarrow B)$	$(\neg S \vee \neg A \vee B) \wedge$ $(\neg S \vee A \vee \neg B)$	$(A \vee B \vee S) \wedge$ $(\neg A \vee \neg B \vee S)$
$S \leftrightarrow (A \oplus B)$	$(\neg S \vee A \vee B) \wedge$ $(\neg S \vee \neg B \vee \neg A)$	$(\neg A \vee B \vee S) \wedge$ $(A \vee \neg B \vee S)$

Tab. 1.7: Prevod vybraných ekvivalencií do tvaru CNF

$$\begin{aligned}
\wedge &\quad (\neg S_4 \vee S_3) \wedge (\neg S_4 \vee S_2) \wedge (\neg S_3 \vee \neg S_2 \vee S_4) \\
\wedge &\quad (\neg S_3 \vee F \vee O) \wedge (\neg F \vee S_3) \wedge (\neg O \vee S_3) \\
\wedge &\quad (\neg S_2 \vee \neg S_1) \wedge (S_1 \vee S_2) \\
\wedge &\quad (\neg S_1 \vee F) \wedge (\neg S_1 \vee O) \wedge (\neg F \vee \neg O \vee S_1)
\end{aligned}$$

Je očividné, že tvar získaný pomocou Alg. 1.2 je zložitejší než tvar podľa Alg. 1.1 – a to nielen v počte použitých symbolov ale aj v počte výsledných klauzúl. A tak to aj veľmi často býva. Napriek tomu má Tseitinova transformácia použitie kvôli

- reprezentácii logických obvodov zložených z prepojených logických hradiel,
- efektivite generovania v najhoršom prípade.

V prvom prípade nie je potrebné generovať parsovací strom logického výrazu ale namiesto toho je možné priamo využiť zapojenie logických obvodov – každé logické hradlo bude predstavovať jeden podvýraz a bude reprezentované jedným novým symbolom. Splnenie každého podvýrazu (reprezentujúceho že nejaký nový symbol je ekvivalentný s činnosťou nejakého logického hradla) zabezpečí vhodné modelovanie činnosti jednotlivých hradiel.

Čo sa týka druhého prípadu, existujú aj také tvary logických viet, pre ktoré je výhodnejšia Tseitinova transformácia. Napríklad existuje taký tvar, pre ktorý algoritmus Alg. 1.1 generuje CNF s počtom klauzúl, ktorý je exponenciálne závislý na počte klauzúl originálnej vety, zatiaľ čo algoritmus

Alg. 1.2 generuje CNF s počtom klauzúl, ktorý na počte klauzúl originálnej vety závisí iba lineárne. V takýchto prípadoch je Tseitinova transformácia oveľa efektívnejšia.

Z hľadiska najhoršieho prípadu (a práve najhorší možný prípad sa uvažuje pri určovaní zložitosti algoritmov) je možné garantovať transformáciu do tvaru CNF iba v exponenciálnom čase a priestore pri využití iba tých symbolov, ktoré boli použité v originálnej vete, zatiaľ co v prípade možnosti pridania nových symbolov prevod do CNF vyžaduje iba lineárny čas a priestor.

ZLOŽITOSŤ KÓDOVANIA DO CNF

CNF vety podľa Alg. 1.1 a Alg. 1.2 sú *ekvisplniteľné* ale nie sú ekvivalentné – prvá je splniteľná iba vtedy, ak druhá je splniteľná a naopak (inými slovami alebo sú obe splniteľné alebo sú obe nesplniteľné). Tseitinova transformácia zachováva splniteľnosť originálnej vety ale nie ekvivalentnosť s ňou. Napríklad veta $A \vee \neg A$ je tautológiou ale pri Tseitinovej transformácii bude vyjadrená ako $S \wedge (S \leftrightarrow (A \vee \neg A)$ čo vedie na $(\neg S \vee A \vee \neg A) \wedge (\neg A \vee S) \wedge (\neg\neg A \vee S)$. Toto už nie je tautológiou ale je splnené iba vtedy, ak symbol S je interpretovaný ako pravdivý. Pri Tseitinovej transformácii nikdy nie je vytvorená validná veta (pravdivá vo všetkých svetoch).

Modely pre originálnu podobu vety (a jej prvú formu podľa Alg. 1.1, ktorá je voči originálnej vete ekvivalentná) je možné získať tak, že z modelov pre formu podľa Alg. 1.2 sa vypustia doplnené nové premenné.

Na ďalšie zníženie počtu klauzúl, generovaných Tseitinovou transformáciou, cieli *Plaisted-Greenbaumova* transformácia, ktorá je založená na *polarite* logických hradiel (a je určená pre tie logické operátory, ktoré sa vyskytujú ako hradlá).

PLAISTED GREENBAUMOVE KÓDOVANIE

Polarita hradla môže byť pozitívna alebo negatívna – podľa toho, aká pravdivostná hodnota by bola vhodná na výstupe hradla. Ak hradlo reprezentuje nejaký výraz F, tak po zavedení nového symbolu S, ktorý bude toto hradlo reprezentovať, Tseitinova transformácia definuje ekvivalenciu $(S \leftrightarrow F)$, čo po náhrade ekvivalencie pomocou dvoch implikácií vedie na

$$(S \rightarrow F) \wedge (F \rightarrow S)$$

Na rozdiel od Tseitinovej transformácie, Plaisted-Greenbaumova transformácia nemusí vždy uvažovať obe implikácie – závisí to práve od polarity hradla. Ak je polarita pozitívna (a teda sa očakáva, že na výstupe hradla bude hodnota TRUE), tak sa uvažuje iba implikácia $(S \rightarrow F)$, ktorej splnenie na základe pravdivosti symbolu S zabezpečí aj splnenie výrazu F. Ak je polarita naopak negatívna (a teda sa očakáva, že na výstupe hradla bude hodnota FALSE), tak sa uvažuje iba implikácia $(F \rightarrow S)$, ktorej splnenie na základe nepravdivosti symbolu S zabezpečí aj nesplnenie výrazu F.

Pre konkrétne hradlá platia polaritné pravidlá

- ak je hradlo typu $\wedge$ alebo $\vee$, tak ak na výstupe hradla je požadovaná nejaká polarita, potom rovnaká polarita je požadovaná na každom jeho vstupe,
- ak je hradlo typu $\neg$, tak ak na výstupe hradla je požadovaná nejaká polarita, potom na jeho vstupe je požadovaná opačná polarita.

Na základe týchto pravidiel je možné odvodiť aj pravidlá pre ďalšie hradlá ako $\oplus$, $\uparrow$ alebo $\downarrow$.

Príklad 1.5 Pre ilustráciu uvažujme vetu $\neg(A \wedge B) \wedge (A \vee B)$, pre ktorú by Tseitinova transformácia uvažovala

$$S_4 \wedge (S_4 \leftrightarrow (S_2 \wedge S_3)) \wedge (S_3 \leftrightarrow (A \vee B)) \wedge (S_2 \leftrightarrow \neg S_1) \wedge (S_1 \leftrightarrow (A \wedge B))$$

Pretože hľadáme takú interpretáciu, pri ktorej je veta pravdivá, tak požadujeme pozitívnu polaritu pre $\neg(A \wedge B) \wedge (A \vee B)$. Z toho vyplýva pozitívna polarita pre $(A \vee B)$ a aj pre $\neg(A \wedge B)$ a teda negatívna polarita pre $(A \wedge B)$. Po zohľadnení týchto polarít Plaisted-Greenbaumova transformácia bude uvažovať náhradu originálnej vety pomocou

$$S_4 \wedge (S_4 \rightarrow (S_2 \wedge S_3)) \wedge (S_3 \rightarrow (A \vee B)) \wedge (S_2 \rightarrow \neg S_1) \wedge ((A \wedge B) \rightarrow S_1)$$

Pri použití tejto transformácie by teda originálna veta bola v CNF forme vyjadrená ako (opäť možno využiť Tab. 1.7)

$$\begin{aligned} S_4 \quad \wedge & \quad (\neg S_4 \vee S_2) \wedge (\neg S_4 \vee S_3) \wedge (\neg S_3 \vee A \vee B) \\ \wedge & \quad (\neg S_2 \vee \neg S_1) \wedge (\neg A \vee \neg B \vee S_1) \end{aligned}$$

Výsledkom je teda rovnaká výsledná množina symbolov avšak iba podmnožina klauzúl, generovaných pri Tseitinovej transformácii. •

LOGICKÉ VYPLÝVANIE

V oblasti odvodzovania je dôležitým pojmom *logické vyplývanie*. Formálne logické vyplývanie $P \models Q$ (Q vyplýva z P) je definované tak, že vo všetkých tých možných svetoch, v ktorých je P pravdivé, je pravdivé aj Q – teda každý model vety P je súčasne aj modelom vety Q. Opačne to však nemusí platiť, model vety Q nemusí byť modelom vety P. Príkladom logického vyplývania je odvodenie tvaru

$$(A \wedge B) \models (A \vee B)$$

Aby platilo jednoduché $\models Q$ (teda možné svety, ktoré majú byť modelmi Q, nie sú obmedzené žiadnym P), tak Q musí byť tautológiou. Naopak, v

Rozšírenie: Transformácia CNF do kCNF

CNF → κCNF

Občas je potrebné transformovať vetu, vyjadrenú v tvare CNF do tvaru kCNF – tvaru, v ktorom každá klauzula je disjunkciou presne k literálov. Je to kvôli uniformite vyjadrenia, keď všetky klauzuly majú rovnakú dĺžku. Taktiež sa používa pri určovaní zložitosti – ak nejaký problém je možné v polynomiálnom čase redukovať na kCNF, tak je rovnako zložitý ako kCNF (3CNF je NP-úplný).

Pri transformácii sa každá klauzula transformuje osobitne bez závislosti na ostatných klauzulách nasledovne:

- klauzula s práve k literálmi ostáva v pôvodnom tvare bez zmeny,
- klauzula s viac ako k literálmi je nahradená konjunkciou viacerých klauzúl, pričom klauzuly sú navzájom previazané prostredníctvom pridaných nových pomocných symbolov,
- klauzula s menej ako k literálmi je doplnená na potrebnú dĺžku pridaním potrebného počtu literálov, pričom je možné použiť existujúce symboly klauzuly alebo pridať symboly nové.

Hodnota k je väčšia ako 2 (neexistuje spôsob ako nahradiť klauzulu ľubovoľnej dĺžky pomocou klauzúl s dvomi literálmi). Prevažne sa používa $k = 3$, čo je považované za "najjednoduchší" prípad.

Pre tento prípad príliš dlhá klauzula $A \vee B \vee C \vee D \vee E$ môže byť nahradená tromi klauzulami s pridaním $n - 3$ (tu 2) nových symbolov

$$(A \vee B \vee S_1) \wedge (\neg S_1 \vee C \vee S_2) \wedge (\neg S_2 \vee D \vee E)$$

pričom tieto symboly budú jedinečné pre danú klauzulu – iná dlhá klauzula pridá iné pomocné symboly. Naproti tomu krátka klauzula $A \vee B$ môže byť doplnená pomocou literálu, ktorý je v nej obsiahnutý, na $A \vee B \vee A$ alebo môže byť pridaním nového pomocného symbolu, ktorý je jedinečný pre danú klauzulu (iné krátke klauzuly pridajú iné pomocné symboly), nahradená konjunkciou dvoch klauzúl

$$(A \vee B \vee S) \wedge (A \vee B \vee \neg S)$$

(ak má krátka klauzula iba jeden literál, tak pridaním dvoch nových pomocných symbolov je nahradená konjunkciou štyroch klauzúl) alebo pridaním pomocného symbolu, zdieľaného všetkými krátkymi klauzulami, pričom tento symbol musí byť interpretovaný ako nepravdivý

$$(A \vee B \vee S) \wedge (\neg S \vee \neg S \vee \neg S)$$

prípade $(P \land \neg P) \models Q$ môže byť Q hocijakou vetou – keďže pre $P \land \neg P$ neexistuje žiadny model, tak potom podľa definície vyplývania Q nemusí byť v žiadnom svete pravdivé. Povedané inak, v prípade výskytu sporu je možné odvodiť čokoľvek. To však nie je veľmi užitočné a preto je snaha sa sporom vyhýbať.

Jedným špeciálnym prípadom logického vyplývania je prípad, keď P v $P \models Q$ je konjunkciou dvoch disjunktívnych klauzúl s komplementárnymi literálmi. Vtedy vyplývanie má tvar

$$\frac{\neg P \lor Q \qquad \neg Q \lor R}{\neg P \lor R}$$

REZOLVENCIA

reprezentujúci *rezolvenčné odvodzovacie pravidlo* $C = C_1 \otimes C_2$, podľa ktorého z platnosti dvoch disjunkcií (klauzúl C_1 a C_2), obsahujúcich komplementárne vyjadrenie literálu nejakého symbolu (prítomného ako v priamej tak aj v negovanej podobe) je možné odvodiť platnosť disjunkcie (klauzuly C) vzniknutej spojením oboch vstupných disjunkcií s vynechaním oboch komplementárnych literálov predmetného symbolu. Výsledná disjunkcia sa označuje ako *rezolventa* – obsahuje všetky literály oboch vstupných klauzúl s výnimkou komplementárnej dvojice literálov (ak vstupné klauzuly obsahujú viac dvojíc komplementárnych literálov, tak vo výslednej rezolvente je vynechaná iba jedna z nich – a teda je možné vytvoriť viac rezolvent).

Rezolvenčné odvodzovacie pravidlo zachováva splniteľnosť – ak množina klauzúl (CNF veta) je splniteľná, tak aj množina rezolvent je splniteľná.

SÉMANTIKA REZOLVENČNÉHO PRAVIDLA

Zmysel rezolvenčného odvodzovacieho pravidla nie je na prvý pohľad najzreteľnejší – situácia sa zmení, ak pravidlo bude vyjadrené v ekvivalentnom tvare

$$\frac{P \to Q \qquad Q \to R}{P \to R}$$

čo je vlastne definíciou *tranzitívnosti* operácie implikácie.

Špecifickým prípadom rezolvencie je situácia, keď jedna zo vstupných disjunkcií má jednotkovú dĺžku (je samostatný literál), ktorá sa označuje ako *jednotková rezolvencia*. Vtedy dĺžka rezolventy je o jeden literál menšia ako dĺžka dlhšej zo vstupných klauzúl – dochádza ku skracovaniu dĺžok. Príkladom je

$$\frac{P \qquad P \to R}{R} \qquad\qquad \frac{\neg R \qquad P \to R}{\neg P}$$

pričom tvar vľavo je známy ako odvodzovacie pravidlo *modus ponens* a tvar vpravo zase ako *modus tolens*.

Každá z klauzúl C_1 a C_2 v rezolvenčnom pravidle $C = C_1 \otimes C_2$ môže obsahovať ľubovoľný počet literálov (jeden, dva alebo viac). Počet literálov vo výslednej rezolvente je rovný súčtu počtov literálov vstupných klauzúl znížený o dva.

Vygenerovaná podoba CNF nejakej logickej vety F nemusí byť minimálna - niekedy ju je možné redukovať na jednoduchší tvar F^r (ktorý ale je tiež CNF). V zásade to je možné dvomi spôsobmi: ZJEDNODUŠOVANIE CNF VIET

- vypustenie symbolu,
- vypustenie klauzuly.

V prípade, že F^r používa rovnakú množinu symbolov ako pôvodná veta, tak sa požaduje ekvivalencia oboch viet – F^r musí mať rovnakú množinu modelov ako F. Ak sa množina symbolov redukuje, teda F^r používa menej symbolov ako pôvodná veta, potom obe vety musia byť ekvisplniteľné - jedna je splniteľná iba vtedy, ak je splniteľná aj druhá. Navyše, ak sa z modelov pôvodnej vety F vypustia interpretácie tých symbolov, ktoré boli z F^r vypustené, tak množina modelov pôvodnej vety sa redukuje na množinu modelov zjednodušenej vety.

Nejaký symbol možno eliminovať z dvoch klauzúl pomocou rezolvenčného odvodzovacieho pravidla, ak v jednej klauzule sa nachádza jeho pozitívny literál a v druhej zase jeho negatívny literál. Výsledok rezolvencie obsahuje literály oboch klauzúl okrem dvojice komplementarnych literálov daného symbolu. ELIMINÁCIA SYMBOLOV Z VETY

Na tomto je možné založiť úplnú elimináciu nejakého symbolu z logickej vety reprezentovanej formou CNF. Nech S_X je množina klauzúl, v ktorých symbol X vystupuje v tvare pozitívneho literálu a $S_{\neg X}$ je množina tých klauzúl, v ktorých X má podobu negatívneho literálu. Potom pre všetky možné dvojice klauzúl, kde jedna je z S_X a druhá z $S_{\neg X}$ je možné urobiť rezolvenciu

$$S_x \otimes S_{\neg x} = \{C_1 \otimes C_2 \mid C_1 \in S_x, C_2 \in S_{\neg x}, C_1 \otimes C_2 \neq \top\}$$

a výsledné rezolventy sa pridajú ako nové klauzuly do danej vety (ak nie sú tautológiami), pričom všetky klauzuly z S_X aj $S_{\neg X}$ sa vypustia

$$(F \setminus (S_x \cup S_{\neg x})) \cup (S_x \otimes S_{\neg x})$$

kde F je logická veta v tvare CNF (konjunkcia klauzúl). Keďže počet takýchto rezolvencií môže prevyšovať počet klauzúl v ktorých figuruje symbol X či už priamo alebo negovane, tak táto náhrada sa zvykne robiť iba vtedy, ak sa celkový počet klauzúl vety nezvýši. Postup sa označuje ako *eliminácia premenných distribúciou klauzúl.*

Iný spôsob je založený na identifikácii klauzúl, ktoré kódujú ekvivalenciu medzi symbolmi. Ak sa zistí, že dva symboly majú byť ekvivalentné, tak jeden je nahradený druhým. Ak napríklad veta obsahuje okrem iných aj klauzuly

$$(A \vee \neg B) \wedge (\neg A \vee B)$$

tak sa vlastne jedná o prekódovanie $(A \leftrightarrow B)$ čo vyjadruje ekvivalentnosť A a B.

ELIMINÁCIA SYMBOLOV Z KLAUZÚL

Na rozdiel od úplnej eliminácie symbolu z logickej vety, niekedy je možné symbol eliminovať iba z niektorých klauzúl, pričom však symbol ostáva prítomným v danej vete. Môže to byť založené na rezolvencii tvaru

$$\frac{P \qquad \neg P \vee R}{R}$$

Keďže P reprezentuje celú klauzulu, tak aby výsledná CNF bola interpretovaná ako pravdivá, tak musí byť ako pravdivá interpretovaná každá jej klauzula – a teda aj P. Z toho vyplýva, že $\neg P$ musí byť interpretovaná ako nepravdivá a preto je zbytočné, aby vystupovala ako súčasť inej klauzuly a môže byť z nej vypustená. Ak P je jediný literál a nie disjunkcia literálov, tak sa hovorí o *jednotkovej propagácii.*

ELIMINÁCIA KLAUZÚL

Niekedy je klauzula identifikovaná ako nadbytočná a preto ju možno odstrániť s tým, že nová veta bude ekvivalentná s vetou pôvodnou. Jedným takým prípadom je výskyt dvoch klauzúl P a $P \vee Q$, kde P aj Q reprezentujú nejaké výrazy – hovoríme že prvá (kratšia) klauzula *zahŕňa* druhú (dlhšiu) klauzulu. V tomto prípade je nadbytočnou druhá kluzula a preto vo vete ostáva iba prvá. V prípade, že P reprezentuje disjunkciu dvoch literálov, tak identifikácia možnosti vypustenia klauzuly môže prebiehať prostredníctvom vkladania literálov do kontrolovanej klauzuly a kontroly výskytu tautológie. Pre každý literál kontrolovanej klauzuly sa hľadá klauzula obsahujúca dva literály - literál z kontrolovanej klauzuly a nejaký iný literál. Ak sa nájde, tak do kontrolovanej klauzuly sa vloží negácia toho iného literálu a urobí sa kontrola na výskyt tautológie.

Nech napríklad jedna klauzula je $A \vee B$ a druhá kontrolovaná klauzula je disjunkcia viacerých literálov vrátane A a B. Pri kontrole literálu A kontrolovanej klauzuly $(... \vee A \vee ... \vee B \vee ...)$ sa nájde klauzula $(A \vee B)$ a tak sa do kontrolovanej klauzuly pridá literál $\neg B$ a teda kontrolovaná klauzula sa zmení na $(... \vee A \vee ... \vee B \vee ... \vee \neg B)$. To vloženie je možné urobiť, pretože ak krátka klauzula je splnená vďaka B tak pridané $\neg B$ v kontrolovanej klauzule nehrá žiadnu úlohu a ak krátka klauzula je splnená vďaka A tak aj kontrolovaná klauzula je splnená vďaka A a opäť pridané $\neg B$ v kontrolovanej klauzule nehrá žiadnu úlohu. A pretože kontrolovaná klauzula teraz obsahuje B aj $\neg B$ ktoré spolu v disjunkcii vytvárajú tautológiu, tak kontrolovanú klauzulu je možné vypustiť. Tento postup je známy pod označením *skrytá eliminácia tautológií*. Inými variantmi sú priama *eliminácia tautológií* alebo *asymetrická eliminácia tautológií*.

Iným prípadom je existencia dvoch klauzúl tvaru $P \vee Q$ a $P \vee \neg Q$. V tomto prípade je možné odstrániť obe takéto klauzuly a namiesto nich vložiť iba jednu klauzulu P.

Prekvapujúcim prístupom k eliminácii klauzúl je pridanie nového symbolu k existujúcim symbolom. Jedná sa vlastne o opačný postup ako pri eliminácii premenných distribúciou klauzúl. Pre vetu F, ktorá neobsahuje symbol X, sa nájde nejaká jej časť S a vytvoria sa množiny klauzúl S_x (v ktorých bude symbol X v tvare priameho literálu) a $S_{\neg x}$ (v ktorých bude symbol X v tvare negovaného literálu) tak, aby platilo

$$S = S_x \otimes S_{\neg x}$$

Z pôvodnej vety sa vypustia klauzuly, ktoré tvoria S a pridajú sa klauzuly z S_x a $S_{\neg x}$

$$(F \setminus (S_x \otimes S_{\neg x})) \cup (S_x \cup S_{\neg x})$$

Tento postup sa označuje ako *pridávanie premenných*. Aby sa počet klauzúl znížil, tak sa aplikuje iba vtedy, ak počet klauzúl v S je väčší ako súčet počtov klauzúl v S_x a $S_{\neg x}$.

Príklad 1.6 Pre ilustráciu tohto prístupu uvažujme vetu v tvare CNF, pozostávajúcu zo siedmich klauzúl

$$(A \vee C) \wedge (A \vee D) \wedge (A \vee E) \wedge (B \vee C) \wedge (B \vee D) \wedge (B \vee E) \wedge (B \vee F)$$

Množina S bude obsahovať prvých šesť klauzúl, pretože pre tieto klauzuly sa dá identifikovať množina vymeniteľných literálov $\{A, B\}$ a množina vzorov $\{(?\vee C), (?\vee D), (?\vee E)\}$ tak, že nahradenie otázniku v každom vzore každým literálom z množiny vymeniteľných literálov poskytne všetky klauzuly z množiny S.

Nech novým symbolom bude X, potom S_x sa vytvorí z nového symbolu a každého vymeniteľného literálu ako $S_x = (A \vee X) \wedge (B \vee X)$ a $S_{\neg x}$ sa vytvorí z X a množiny vzorov ako $S_{\neg x} = (\neg X \vee C) \wedge (\neg X \vee D) \wedge (\neg X \vee E)$. Výsledná podoba vety bude

$$(A \vee X) \wedge (B \vee X) \wedge (\neg X \vee C) \wedge (\neg X \vee D) \wedge (\neg X \vee E) \wedge (B \vee F)$$

pričom bude mať o jeden symbol viac a o jednu klauzulu menej ako pôvodná podoba vety.

V uvedenom príklade pre prvých šesť klauzúl tvoriacich S bolo možné určiť aj inú množinu vymeniteľných literálov $\{C, D, E\}$. K tejto by prislúchala množina vzorov $\{(A \vee ?), (B \vee ?)\}$. Takáto voľba by viedla na výslednú podobu vety

$$(C \vee X) \wedge (D \vee X) \wedge (E \vee X) \wedge (A \vee \neg X) \wedge (B \vee \neg X) \wedge (B \vee F)$$

Postup je možné zovšeobecniť – je možné vymeniteľné literály nahradiť väčšími celkami. Napríklad pre vetu

$$(A \vee D) \wedge (A \vee E) \wedge (B \vee C \vee D) \wedge (B \vee C \vee E)$$

je možné použiť množinu vymeniteľných disjunktov $\{A, B \vee C\}$, čo s množinou vzorov $\{(? \vee D), (? \vee E)\}$ vedie na

$$(A \vee X) \wedge (B \vee C \vee X) \wedge (\neg X \vee D) \wedge (\neg X \vee E)$$

•

Existuje aj iná trieda metód zjednodušovania logických viet, ktorá si nerobí nárok na zachovanie vzťahu ekvivalentnosti medzi pôvodnou a redukovanou vetou ani v prípade, že F^r používa rovnakú množinu symbolov ako pôvodná veta F. Už sa nepožaduje ekvivalencia oboch viet – F^r nemusí mať rovnakú množinu modelov ako F.

Eliminácia blokovaných klauzúl

Tieto metódy majú použitie iba v určitom špecifickom kontexte. Takýmto kontextom môže byť napríklad rozhodnutie, či veta je splniteľná alebo nesplniteľná – či existuje nejaký jej model alebo nie. Príkladom vypúšťania klauzúl pre tento kontext je *eliminácia blokovaných klauzúl*. Pre pochopenie metódy je potrebná definícia určujúca, ktorá klauzula je blokovaná. Pod blokovanou klauzulou sa jednoducho rozumie taká klauzula, ktorá obsahuje literál, ktorý túto klauzulu blokuje. A *blokujúci literál*: literál L, ktorý je súčasťou nejakej klauzuly C ktorá je súčasťou vety F vyjadrenej v CNF tvare, blokuje túto klauzulu C s ohľadom na F vtedy, ak pre každú inú klauzulu $C' \in F$ obsahujúcu komplementárnu podobu literálu L, výsledok rezolvencie C a C' (na báze vypustenia L a jeho komplementu) je tautológia. Teda pre C a každú C' musí byť možnosť rezolvencie v tvare

$$\frac{C = ... \vee X \vee ... \vee L \vee ... \qquad ... \vee \neg L \vee ... \vee \neg X \vee ... = C'}{(C \setminus \{L\}) \vee (C' \setminus \{\neg L\}) = ... \vee X \vee ... \vee \neg X \vee ...}$$

kde vstupné klauzuly musia obsahovať aspoň dve dvojice komplementárnych literálov – jedna (viazaná na blokujúci literál) sa odstráni ale druhá dvojica sa prenesie do výslednej rezolventy a spôsobí, že tá sa stane tautológiou.

Identifikované blokované klauzuly je možné z vety odstrániť, pričom splniteľnosť vety ostane zachovaná – ak pôvodná veta bola nesplniteľná, tak ostáva nesplniteľnou aj po eliminácii blokovaných klauzúl, ak pôvodná veta bola splniteľná, tak aj redukovaná veta je splniteľná (avšak môže mať viac modelov ako pôvodná veta – môže mať ako modely aj také interpretácie, ktoré nie sú modelmi pôvodnej vety). Výsledok odstránenia všetkých blokovaných klauzúl je nezávislý od poradia, v akom sa klauzuly odstraňujú.

Ak by veta F mala tvar $(P \vee Q) \wedge (P \vee \neg Q \vee \neg R) \wedge (\neg P \vee R)$, teda bola by zložená z troch klauzúl, tak prvá klauzula nie je blokovaná, druhá je blokovaná literálom P a tiež je blokovaná literálom $\neg R$ a tretia klauzula je blokovaná literálom R. Literál $\neg P$ nie je blokujúcim pre tretiu klauzulu, pretože síce výsledok rezolvencie s druhou klauzulou je tautológia $(\neg Q \vee \neg R \vee R)$ avšak výsledok rezolvencie tretej klauzuly s prvou klauzulou tautológiou nie je.

Inými metódami eliminácie klauzúl, ktoré zachovávajú splniteľnosť, sú ďalšie varianty založené na blokovaných klauzulách (napr. *skryté BCE*, *asymetrické BCE*) ale aj varianty založené na eliminácii pokrytých klauzúl (napr. základné *CCE* a tiež skrytý a asymetrický variant).

Príklad 1.7 Ak by sme z príkladu na strane 2 vybrali iba všetky tie klauzuly, ktoré sa týkajú Ruda, tak by sme mali kaluzuly

$$\begin{array}{ll} & (\neg P_2 \vee R \vee S) \wedge (\neg P_2 \vee \neg R \vee \neg S) \wedge (\neg R \vee S \vee P_2) \wedge (R \vee \neg S \vee P_2) \\ \wedge & (\neg P_5 \vee O \vee R) \wedge (\neg P_5 \vee \neg O \vee \neg R) \wedge (\neg O \vee R \vee P_5) \wedge (O \vee \neg R \vee P_5) \\ \wedge & (\neg R \vee O) \end{array}$$

Toto sú jediné klauzuly v pôvodnej vete, ktoré obsahujú literály symbolov R a S – ostatné literály sa vyskytujú aj v iných klauzulách vety, ktoré tu nie sú uvedené. V tejto vzorke klauzúl je možné robiť postupne tieto zjednodušenia, týkajúce sa symbolov R a S:

- klauzula $(O \vee \neg R \vee P_5)$ je vypustená, pretože kvôli poslednej klauzule bola metódou skrytej eliminácie tautológií identifikovaná tautológia,

- zhodou okolností prvé štyri klauzuly vo vybranej vzorke klauzúl sú všetky klauzuly, v ktorých vystupuje Stano. Všetky štyri sú blokované, pričom blokujúcim literálom je S pri prvej a tretej a $\neg S$ pri druhej a štvrtej klauzule, takže všetky štyri môžu byť vypustené (a zároveň vo vete už nebude symbol reprezentujúci Stana). Rovnaký výsledok by bol dosiahnutý snahou o elimináciu symbolu S distribúciou klauzúl.

Vybraná vzorka klauzúl bola zjednodušená na štyri ostávajúce klauzuly

$$(\neg P_5 \vee O \vee R) \wedge (\neg P_5 \vee \neg O \vee \neg R) \wedge (\neg O \vee R \vee P_5) \wedge (\neg R \vee O)$$

ktoré je možné ďalej zjednodušovať:

- pre klauzulu $(\neg P_5 \vee \neg O \vee \neg R)$ je $\neg R$ bokujúcim literálom a pre klauzulu $(\neg O \vee R \vee P_5)$ je zase blokujúcim literálom R, takže obe môžu byť vypustené,
- ostávajúce dve klauzuly $(\neg P_5 \vee O \vee R)$ a $(\neg R \vee O)$ sú posledné s literálmi symbolu R, ktorý môžme pomocou eliminácie distribúciou klauzúl odstrániť. Tieto dve klauzuly je teda možné nahradiť jednou novou klauzulou $(\neg P_5 \vee O)$.

Upravená pôvodná veta, v ktorej vybraná vzorka klauzúl bola nahradená novo vygenerovanou klauzulou, je splniteľná. To znamená, že aj pôvodná podoba vety je splniteľná a má aspoň jeden model. •

Cvičenia

1. Vytvorte pravdivostnú tabuľku pre nasledovné výrazy vo výrokovom počte:

   a. $\neg A \wedge (A \vee B) \wedge (B \vee C)$
   b. $((\neg A \rightarrow \neg B) \rightarrow A) \rightarrow ((\neg B \rightarrow \neg A) \rightarrow \neg B)$
   c. $((\neg A \rightarrow \neg B) \rightarrow A) \wedge ((\neg B \rightarrow \neg A) \rightarrow B)$
   d. $(P \rightarrow Q) \wedge (P \rightarrow R) \wedge P \wedge (\neg Q \vee \neg R)$
   e. $(A \wedge B) \vee (B \wedge C) \vee (C \wedge A)$
   f. $\neg(P \rightarrow Q \vee R \leftrightarrow (P \rightarrow Q) \vee (P \rightarrow R))$

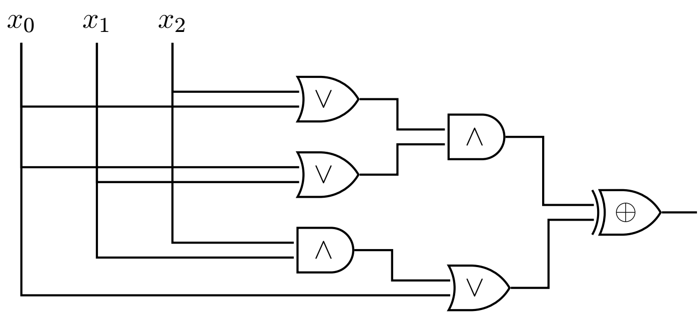
   Čo sa dá povedať o pravdivosti týchto viet ?

2. Vety vo výrokovom počte z predchádzajúceho príkladu reprezentujte v tvare

   - disjunktívnej normálnej formy,
   - binárneho rozhodovacieho diagramu,

   pričom vychádzajte z pravdivostných tabuliek uvedených viet.

3. Transformujte vety vo výrokovom počte z prvého príkladu do tvaru CNF pri zachovaní použitej množiny symbolov pomocou
    - použitia pravdivostných tabuliek,
    - algoritmu prevodu do CNF.
4. Dokážte:
    - pre jednotlivé binárne operátory, či sú alebo nie sú komutatívne a asociatívne,
    - platnosť prepisu ekvivalencií do tvaru CNF, ktoré sú uvedené v Tab. 1.7.
5. Reprezentujte funkciu logického obvodu pomocou logického výrazu v tvare CNF pomocou Tseitinovej transformácie.



Porovnajte s výsledkom Plaisted-Greenbaumovej transformácie (uvažujte hodnotu TRUE na výstupe obvodu).

6. Transformujte nasledujúce prípady do CNF pre rôzne hodnoty n a určte závislosť výsledného počtu premenných a klauzúl na počte v originálnej forme.
    a) $(A_1 \wedge B_1) \vee ... \vee (A_n \wedge B_n)$
    b) $(A_1 \wedge ... \wedge A_n) \vee (B_1 \wedge ... \wedge B_n)$

    Použite ako transformáciu so zachovaním symbolov tak aj transformáciu s pridaním nových symbolov.
7. Prepíšte do výrokového počtu nasledujúce tvrdenia o piatich predmetoch (dve guličky a tri kocky). Tvrdenia sú tvaru "Na stole je …"
    a. … aspoň jedna gulička a aspoň jedna kocka

b. … viac guličiek ako kociek
c. … maximálne jedna gulička a maximálne jedna kocka
d. … vtedy a len vtedy nejaká gulička ak tam je aj nejaká kocka
e. … ak nepárny počet guličiek tak aj nepárny počet kociek
f. … párny počet predmetov

Pri všetkých výberoch uvažujte všetkých päť predmetov.

8. Pre ilustračný príklad na strane 2 prepíšte znalosti o príklade pomocou výrokového počtu a transformujte ich do tvaru CNF.

9. Prepíšte do výrokového počtu nasledujúce znalosti o piatich priateľoch (Anna, Hedviga, Klára, Rudolf a Vladimír), ktorí radi navzájom online komunikujú. Vieme, že platí nasledovné:

   - Alebo Klára alebo Hedviga alebo obidve sú on-line.
   - Rudolf alebo Vladimír sú online, avšak nie obaja.
   - Ak Anna je on-line, potom Rudolf je tiež on-line.
   - Vladimír je on-line vtedy a iba vtedy, keď je on-line Klára.
   - Ak je Hedviga on-line, potom sú online aj Anna a Klára.

   Následne reprezentáciu transformujte do tvaru CNF.

10. Zjednodušte nasledujúce vety v CNF tak, aby počet klauzúl vo vetách poklesol:

   - $(D \vee F) \wedge (A \vee B \vee E) \wedge (\neg A \vee \neg B \vee F) \wedge$ $(C \vee \neg E) \wedge (\neg B \vee \neg D \vee \neg F) \wedge (A \vee \neg F) \wedge$ $(A \vee \neg C \vee E) \wedge (B \vee \neg F) \wedge (\neg B \vee \neg E) \wedge (C \vee \neg F)$
   - $(\neg A \vee \neg B) \wedge (\neg H \vee A) \wedge (A \vee B \vee \neg C) \wedge (E \vee D \vee \neg C \vee \neg G) \wedge$ $(E \vee \neg H) \wedge (\neg A \vee C) \wedge (B \vee F \vee G) \wedge (\neg A \vee H) \wedge (B \vee D \vee \neg G)$
   - $(A \vee B \vee C \vee D) \wedge (A \vee B \vee C \vee E) \wedge (A \vee B \vee C \vee F) \wedge$ $(A \vee B \vee D \vee E) \wedge (A \vee B \vee D \vee F) \wedge (A \vee B \vee E \vee F) \wedge$ $(A \vee C \vee D \vee E) \wedge (A \vee C \vee D \vee F) \wedge (A \vee C \vee E \vee F) \wedge$ $(A \vee D \vee E \vee F) \wedge (B \vee C \vee D \vee E) \wedge (B \vee C \vee D \vee F) \wedge$ $(B \vee C \vee E \vee F) \wedge (B \vee D \vee E \vee F) \wedge (C \vee D \vee E \vee F)$

Literatúra

1. Coppin, B.: Artificial Intelligence Illuminated. Jones and Bartlett Publishers, Sudbury, MA, 2004.

2. Eén, N. – Biere, A.: Effective Preprocessing in SAT through Variable and clause Elimination. In: Theory and Application of Satisfiability Testing SAT 2005, Springer, Berlin Heidelberg, LNCS 3569, 2005, 61–75.

3. Heule, M. – Järvisalo, M. – Biere, A.: Covered Clause Elimination. In: LPAR-17-short. short papers for 17th International Conference on Logic for Programming, Artificial intelligence, and Reasoning, Voronkov, A. et al. (eds,), vol 13, 2010, 41–46.

4. Heule, M. – Järvisalo, M. – Biere, A.: Clause Elimination Procedures for CNF Formulas. In: Logic for Programming, Artificial Intelligence, and Reasoning. LPAR 2010, Fermüller C.G. – Voronkov A. (eds.), LNCS, vol 6397. Springer, Berlin, Heidelberg, 2010, 357–371.

5. Järvisalo, M. – Biere, A. – Heule, M.: Blocked Clause Elimination. In: Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2010, Esparza J. – Majumdar R. (eds.), LNCScience, vol 6015. Springer, Berlin, Heidelberg, 2010, 129–144.

6. Manthey, N. – Heule, M.J.H. – Biere, A.: Automated Reencoding of Boolean Formulas. In: Hardware and Software: Verification and Testing, HVC 2012. Springer, Berlin, Heidelberg, LNCS 7857, 2013, 102-117.

7. Mordechai B.-A.: Mathematic Logic for Computer Science (3. vyd.). Springer, London, 2012.

8. Plaisted, D.A. – Greenbaum, S.: A Structure Preserving Clause Form Transform. J. Symbolic Computation, Vol 2(3), 1986.

9. Prestwich, S.: CNF Encodings. In: Handbook of satisfiability, Biere et al. (eds.). IOS Press, Amsterdam, 2009, 75–97.

10. Tseitin, G.: On the complexity of derivation in propositional calculus. Automation of Reasoning: Classical Papers in Computational Logic, 2:466–483, 1983. Springer-Verlag.