Sadržaj

Poglavlje 1

Logika sudova - osnovni pojmovi

1.1 Simboli i formule

Logika sudova, ili propozicijska logika, često predstavlja objekt prvog susreta s matematičkom logikom. Bez sumnje ste se već sreli s njome na studiju. Ovdje ćemo samo malo bolje formalizirati neke definicije. To neće samo služiti preciznom uvođenju pojmova za logiku sudova, već će pokazati put kako se odgovarajući pojmovi mogu formalno uvesti u kompliciranijim logikama.

Napomena 1. Nulu smatramo prirodnim brojem!

Definicija 1. *Simboli* logike sudova su atomi i veznici, te pomoćni simboli (zagrade).

Atomi su elementi nekog fiksiranog beskonačnog (bez velikog smanjenja općenitosti možemo pretpostaviti: prebrojivog) skupa At. Atome najčešće označavamo malim slovima (obično se koriste p, q ili r), eventualno indeksiranim prirodnim brojem (poput p_0, q_5 ili r_{152}).

Veznici su funkcije čija kodomena je $\{0,1\}$, a domena je $\{0,1\}^k$, za neki prirodni broj k (*mjesnost* veznika). Veznike mjesnosti 2 zovemo *binarnima*, mjesnosti 1 *unarnima*, a mjesnosti 0 *logičkim konstantama*.

Zadatak 1. Koliko ima logičkih konstanti, koliko unarnih, a koliko binarnih veznika? Koji su to? Koliko ima veznika mjesnosti k?

Definicija 2. Formula je (konačno) uređeno stablo, definirano induktivno. Formula je jednog od dva tipa:

- list označen atomom (*atomarna* formula)
- čvor označen veznikom (*glavni* veznik) mjesnosti k, s k djece koja su korijeni formula (*glavne potformule*).

Skup svih formula označavamo s Fo.

Kako su čvorovi označenim logičkim konstantama također listovi, često se i logičke konstante smatraju atomarnim formulama, iako strogo formalno one to nisu.

Direktna posljedica definicije je dokazivanje *indukcijom po izgradnji formule*: ako sve atomarne formule imaju svojstvo P, i svaka neatomarna formula čije sve glavne potformule imaju svojstvo P također ima svojstvo P, tada sve formule imaju svojstvo P.

Također možemo rekurzivno definirati funkcije s domenom Fo: ako imamo skup S i funkciju $f_{At} : At \to S$, te za svaki veznik t mjesnosti k imamo funkciju $f_t : S^k \to S$, tada postoji jedinstvena funkcija $f : Fo \to S$ koja atomarnu formulu s jedinim atomom p preslikava u $f_{At}(p)$, a formulu s glavnim veznikom t i glavnim potformulama $F_1 \ldots, F_k$ preslikava u $f_t(f(F_1), \ldots, f(F_k))$.

Čest specijalni slučaj pojavljuje se kad je $S = Fo$ (definiramo transformaciju formula), te f_t bude upravo konstrukcija formule s glavnim veznikom t (i glavnim potformulama koje su argumenti od f_t). U tom slučaju samo trebamo definirati f_{At}, i kažemo da f *komutira s veznicima*.

Zadatak 2. Rekurzivno definirajte funkciju $Var : Fo \to \mathcal{P}(At)$, koja svakoj formuli pridružuje skup svih atoma koji se u njoj pojavljuju.

Evo jednog malo složenijeg primjera takve rekurzivne definicije. Formalno, to nije dobar primjer, jer nije definiran na svim formulama, ali skup formula na kojima je definiran je također moguće induktivno definirati.

1.2 Nizovna reprezentacija

Ako su svi veznici u formuli mjesnosti najviše 2, svakoj formuli možemo pridružiti njenu *nizovnu reprezentaciju*, koja je konačni niz simbola, na sljedeći način:

- nizovna reprezentacija lista je niz čiji jedini element je oznaka tog lista (atom ili logička konstanta)
- nizovna reprezentacija formule s unarnim glavnim veznikom je niz $x_0x_1 \dots x_k$, gdje je x_0 glavni veznik, a $x_1 \dots x_k$ nizovna reprezentacija jedine njene glavne potformule
- nizovna reprezentacija formule s binarnim glavnim veznikom je niz $x_0x_1 \dots x_k$, gdje je x_0 otvorena zagrada, x_k zatvorena zagrada, te postoji $l \in \{1, \dots, k-1\}$ takav da je x_l glavni veznik formule, $x_1 \dots x_{l-1}$ nizovna reprezentacija njene prve glavne potformule, a $x_{l+1}x_{l+2} \dots x_{k-1}$ nizovna reprezentacija njene druge glavne potformule.

Ukratko, formulu obilazimo INORDER obilaskom, i stavimo zagrade slijeva i zdesna, ako joj je glavni veznik mjesnosti 2, a PREORDER obilaskom inače.

U računalnoj interpretaciji, imena za simbole su najčešće Unicode znakovi (ako zanemarimo indeksirana imena), te nizovnu reprezentaciju možemo jednostavno smatrati stringom.

Zadatak 3. Dokažite da je nizovna reprezentacija, kao preslikavanje sa Fo u skup svih stringova, injekcija. Navedite kontraprimjer koji pokazuje da je stavljanje inorder obilaska u zagrade nužno za injektivnost.

Prethodni zadatak pokazuje da je moguće *parsiranje* nizovnih reprezentacija, odnosno *serijalizacija* formula u njihove nizovne reprezentacije ne gubi informaciju. Serijalizacija je postupak kojim se komplicirani objekti pretvaraju u nizove znakova (zapravo bitova, ali to je riješeno UTF-8 standardom), radi ispisa na ekran, zapisa u datoteku, ili slanja preko mreže. Parsiranje je postupak kojim iz stringa konstruiramo objekt.

Zadatak 4. Parsirajte string "$\neg((p \to q) \wedge p)$", odnosno nacrtajte stablo kojem je to nizovna reprezentacija.

Zahvaljujući mogućnosti parsiranja, možemo navoditi samo nizovne reprezentacije, i znat će se na koje formule mislimo. Ubuduće ćemo dakle samo govoriti stvari poput "formula $\neg((p \to q) \wedge p)$", misleći pritom na stablo koje je rezultat parsiranja tog stringa. Također, ako je glavni veznik formule binarni, često nećemo pisati vanjske zagrade kad ne postoji opasnost od zabune ("formula $p \vee q$" nam zapravo znači "formula $(p \vee q)$").

Zadatak 5. Očito, algoritam za parsiranje mora razlikovati zagrade od ostalih simbola (inače bi mogao formulu "$(p \wedge q)$" parsirati kao da su '(', 'p', '$\wedge$' i 'q' unarni veznici, a ')' atom). Ali ako nas zanima samo oblik stabla koje se dobije, je li to jedino što mora razlikovati? Odnosno, može li se broj djece svakog čvora rekonstruirati samo iz položaja njegove oznake u nizovnoj reprezentaciji u odnosu na zagrade i na ostale simbole?

Evo nekoliko često korištenih veznika, s njihovim uobičajenim oznakama.

- $\neg$ (negacija): $\neg(x) := 1 - x$ (unarni veznik)
- $\wedge^k$ (konjunkcija): $\wedge^k(x_1, \dots x_k) := \min\{x_1, \dots, x_k\}$.
  Umjesto $\wedge^2$ (binarni veznik) često pišemo samo $\wedge$. Primijetimo $\wedge^1 = id$.
  Ima smisla samo kad je $k \geq 1$. Ponekad se definira $\wedge^0 := \top$.
- $\vee^k$ (disjunkcija): $\vee^k(x_1, \dots, x_k) := \max\{x_1, \dots, x_k\}$.
  Umjesto $\vee^2$ (binarni veznik) često pišemo samo $\vee$. Primijetimo $\vee^1 = id$.
  Ima smisla samo kad je $k \geq 1$. Ponekad se definira $\vee^0 := \bot$.
- $\to$ (kondicional): $\to(x, y) := \max\{1 - x, y\}$ (binarni veznik)
- $\leftrightarrow$ (bikondicional): $\leftrightarrow(x, y) := \delta_{xy}$ (Kroneckerov simbol) (binarni veznik)
- $\bot$ (laž): $\bot() := 0$ (logička konstanta)
- $\top$ (istina): $\top() := 1$ (logička konstanta)

U nekim slučajevima možemo bez opasnosti od zabune proširiti našu definiciju nizovne reprezentacije i na formule koje sadrže veznike mjesnosti veće od 2. Konkretno, disjunkciju odnosno konjunkciju od k formula pišemo jednostavno kao $(F_1 \vee \cdots \vee F_k)$, odnosno $(F_1 \wedge \cdots \wedge F_k)$. Primijetimo da nema unutarnjih zagrada — ovo nije iterirana binarna konjunkcija odnosno disjunkcija (iako joj je dakako ekvivalentna). Svih k formula su na istoj dubini 1 u stablu, a glavni veznik, iako se pojavljuje na $k-1$ mjesta u nizovnoj reprezentaciji, u samoj formuli pojavljuje se samo jednom: u korijenu.

1.3 Interpretacije

Imati mnogo veznika je dobro za izražajnost logike, ali često kod dokazivanja indukcijom po izgradnji formule moramo promotriti svaki veznik posebno: dakle, za dokazivanje općenitih teorema nam je dobro imati što manje veznika. Neki veznici se mogu izraziti preko nekih drugih, ali da bismo to formalno definirali, treba nam pojam interpretacije.

Definicija 3. Neka je F formula. *Interpretacija za* F je bilo koje preslikavanje $I : T \to \{0, 1\}$, gdje je $Var(F) \subseteq T \subseteq At$. Ako je I neka interpretacija za F, *vrijednost* od F *pod* interpretacijom I je broj $I(F)$, definiran na sljedeći način:

- Vrijednost atomarne formule pod I je vrijednost preslikavanja I na njenom jedinom atomu (na taj način nema opasnosti od zabune pri korištenju oznake $I(p)$).
- Vrijednost formule s glavnim veznikom t mjesnosti k pod interpretacijom I jednaka je $t(I(F_1), \ldots, I(F_k))$, gdje su $F_1, \ldots, F_k$ njene glavne potformule redom.

Primijetimo da se to može shvatiti kao rekurzivna definicija: $I_{At} = I$ (početno zadana), a $I_t = t$ za svaki veznik t.

Kažemo da je formula F *lažna* pod interpretacijom I ako je $I(F) = 0$, a da je *istinita* pod I ako je $I(F) = 1$.

Parcijalna interpretacija je bilo koja parcijalna funkcija $I' : At \rightharpoonup \{0, 1\}$ (dakle, funkcija čija domena je neki podskup od At). Ako je S neki skup formula, *skup vrijednosti* od S pod parcijalnom interpretacijom I' je skup

$$I'[S] := \left\{ I(F) \ : \ F \in S \ \& \ I \text{ je interpretacija za } F \ \& \ I \text{ proširuje } I' \right\}.$$

Kažemo da je skup formula S *lažan* pod parcijalnom interpretacijom I' ako je $I'[S] = \{0\}$, a da je *istinit* ako je $I'[S]$ ili $\{1\}$ ili $\emptyset$ (primijetimo, prazan skup je istinit pod svakom interpretacijom). Kažemo da je formula F lažna/istinita pod I' ako je jednočlan skup $\{F\}$ lažan/istinit pod I'.

Primijetimo da je formula uvijek istinita ili lažna pod interpretacijom za nju, ali može biti ni istinita ni lažna pod parcijalnom interpretacijom.

Funkcije sa čitavog skupa At u $\{0,1\}$ (i samo one) su interpretacije za svaku formulu: njih jednostavno zovemo *interpretacijama*. Ako želimo istaći da su definirane na čitavom At, zovemo ih *totalnim* interpretacijama.

Prazan skup $\emptyset$ je parcijalna funkcija iz At u $\{0,1\}$ (s praznom domenom): zovemo je *prazna interpretacija*.

Definicija 4. Neka je F formula. Kažemo da je F *valjana* ako je istinita pod praznom interpretacijom, a da je *proturječna* ako je lažna pod praznom interpretacijom. Valjane formule još zovemo *tautologijama*, a proturječne *antitautologijama*. Kažemo da je F *ispunjiva* ako nije proturječna, a da je *oboriva* ako nije valjana.

Neka je S skup formula. Kažemo da je S *ispunjiv* ako postoji interpretacija pod kojom je S istinit, a da je *oboriv* ako postoji interpretacija pod kojom je lažan.

Zadatak 6. Dokažite alternativnu definiciju: Za formulu F:

- F je valjana ako i samo ako je istinita pod svakom interpretacijom.
- F je proturječna ako i samo ako je lažna pod svakom interpretacijom.
- F je ispunjiva ako i samo ako je istinita pod nekom interpretacijom.
- F je oboriva ako i samo ako je lažna pod nekom interpretacijom.

1.4 Ekvivalencija i supstitucija

Definicija 5. Za formule F i G kažemo da su *ekvivalentne*, i pišemo $F \Leftrightarrow G$, ako imaju istu vrijednost pod svakom interpretacijom za obje formule.

Za skup S i formulu F kažemo da F *logički slijedi* iz S, i pišemo $S \models F$, ako je F istinita pod svakom interpretacijom pod kojom je S istinit. Ako je S konačan skup, često ispuštamo vitičaste zagrade: pišemo $F_1, \ldots, F_m \models F$ umjesto $\{F_1, \ldots, F_m\} \models F$.

Primijetimo da $\emptyset \models F$ vrijedi točno za valjane F: dakle, s $\models F$ možemo označavati da je F valjana.

Zadatak 7. Dokažite da su sljedeće tvrdnje ekvivalentne sa $F \Leftrightarrow G$:

- Za svaku interpretaciju I koja je istovremeno interpretacija za F i za G, F je istinita pod I ako i samo ako je G istinita pod I
- $F \leftrightarrow G$ je tautologija
- $F \models G$ i $G \models F$

Također, dokažite da su sljedeće tvrdnje ekvivalentne sa $F_1 \ldots, F_m \models G$:

- $(F_1 \wedge \cdots \wedge F_m) \rightarrow G$ je tautologija
- $F_1 \rightarrow (\cdots (F_m \rightarrow G) \cdots)$ je tautologija
- skup $\{F_1, \ldots, F_m, \neg G\}$ nije ispunjiv

Osnovno svojstvo ekvivalentnih formula je supstitutabilnost (zamjenjivost): ako je $F \Leftrightarrow G$, tad u bilo kojoj formuli H u kojoj se pojavljuje potformula F, zamjenom te pojave F za G dobivamo formulu ekvivalentnu s H. Da bismo to formalno definirali, treba nam pojam supstitucije (zamjene).

Definicija 6. Neka je $s : At \rightharpoonup Fo$ parcijalna funkcija. Definiramo *zamjenu po* s kao preslikavanje $[s]$ na formulama, induktivno.

- $p\,[s] := s(p)$, ako je $p \in \mathrm{Dom}(s)$
- $q\,[s] := q$, ako je $q \in At \setminus \mathrm{Dom}(s)$
- $[s]$ komutira s veznicima.

Ako je s konačna parcijalna funkcija, često ispuštamo vitičaste zagrade: pišemo $G\,[p_1 \mapsto F_1, \ldots, p_m \mapsto F_m]$ umjesto $G\,[\{p_1 \mapsto F_1, \ldots, p_m \mapsto F_m\}]$.

Teorem 1. Neka su F, G i H formule, te p atom. Ako je $F \Leftrightarrow G$, tada je i $H\,[p \mapsto F] \Leftrightarrow H\,[p \mapsto G]$.

Dokaz. Indukcijom po izgradnji formule H. Ako je H atomarna, imamo dva slučaja: ako je njen atom p, tvrdnja postaje $F \Leftrightarrow G$, što imamo po pretpostavci. Ako je pak njen atom neki drugi q, tvrdnja postaje $q \Leftrightarrow q$, što je očito.

Ako H ima glavni veznik t mjesnosti k i glavne potformule $H_1, \ldots H_k$, za koje vrijedi pretpostavka indukcije, tada za svaku interpretaciju I za $H\,[p \mapsto$

$F]$ i $H\,[p \mapsto G]$, vrijedi

$$I(H\,[p \mapsto F]) = t\Big(I(H_1\,[p \mapsto F]), \ldots, I(H_k\,[p \mapsto F])\Big) =$$
$$= t\Big(I(H_1\,[p \mapsto G]), \ldots, I(H_k\,[p \mapsto G])\Big) = I(H\,[p \mapsto G])\,. \quad \square$$

Poglavlje 2

Normalne forme

2.1 Tablice istinitosti

Vidjeli smo da interpretaciju možemo zadati na atomima, i ona se onda prirodno proširuje na sve formule koje sadrže te atome. Zapravo, ako fiksiramo formulu F i variramo interpretaciju na atomima, vidimo da smo dobili preslikavanje sa skupa $\{0,1\}^{Var(F)}$ u $\{0,1\}$. Ako standardiziramo redoslijed atoma u F, zapravo vidimo da smo dobili veznik.

Teorem 2. Za svaki prirodan broj k i formulu F takvu da je $Var(F) \subseteq \{p_1, \ldots, p_k\}$ postoji veznik v_F^k mjesnosti k, takav da za sve interpretacije I za F vrijedi

$$I(F) = v_F^k\Big(I(p_1), \ldots, I(p_k)\Big) .$$

Dokaz. Zapravo samo treba definirati v_F^k kao funkciju na svakoj k-torki nula i jedinica. Za proizvoljne $x_1, \ldots, x_k \in \{0,1\}$, definiramo interpretaciju I_x tako da je $I_x(p_i) = x_i$ za sve $i \in \{1, \ldots, k\}$ (na ostalim atomima I_x ostavimo nedefiniranom), ustanovimo da je I_x interpretacija za F jer je $\operatorname{Dom} I_x = \{p_1, \ldots, p_k\} \supseteq Var(F)$, i definiramo

$$v_F^k(x_1, \ldots, x_k) := I_x(F) . \quad \square$$

Definicija 7. Veznik v_F^k iz prethodnog teorema zovemo *veznik (mjesnosti k) koji odgovara formuli F*. Ako je $Var(F) = \{p_1, \ldots, p_k\}$, onda često

umjesto v_F^k pišemo samo v_F. Odnosno, podrazumijevana mjesnost veznika koji odgovara formuli F je upravo jednaka broju različitih atoma u F.

Veznik koji odgovara formuli možemo definirati i za formule koje sadrže druge varijable osim p_i. Zapravo, sve što nam treba je neki kanonski totalni uređaj na atomima. Možemo propisati da su slova uređena po abecedi, neindeksirana dolaze prije indeksiranih, a indeksirana su uređena po indeksima.

$$\cdots \prec p \prec p_0 \prec p_1 \prec p_2 \prec \cdots \prec q \prec q_0 \prec \cdots \prec r \prec \cdots$$

Graf veznika v_F zovemo *tablicom istinitosti* formule F.

Primijetimo, tablica istinitosti je skup od 2^k $(k+1)$-torki nad skupom $\{0,1\}$, u kojima prvih k komponenti predstavlja proizvoljnu kombinaciju nula i jedinica, a zadnja komponenta odgovara vrijednosti v_F na toj kombinaciji.

2.2 Klauzule

Na početku točke 1.3 primijetili smo da su formule vrlo generalno definirane, i operatori mogu biti bilo kakvi. U prethodnoj točki smo to potvrdili, dokazavši da svaku formulu, ako je gledamo semantički, možemo shvatiti kao jedan operator, čija je mjesnost jednaka broju različitih atoma u formuli.

Treba li nam tolika generalnost? Pokazat ćemo da (za semantiku) ne treba, štoviše postoje vrlo mali skupovi veznika takvi da svaka formula ima ekvivalentnu *normalnu formu* koja se sastoji od atoma i veznika iz tog skupa. Da bismo to lakše dokazali, a i jer će nam trebati kasnije, uvodimo nekoliko pomoćnih pojmova.

Napomena 2. Za skup X, s $\mathcal{P}_f(X)$ označimo skup svih konačnih podskupova od X. Primijetimo $\mathcal{P}_f(X) = \mathcal{P}(X)$ ako i samo ako je X konačan.

Definicija 8. Bilo koju relaciju na $\mathcal{P}_f(At)$ zovemo *klauzalnom formom*. Dakle, elementi klauzalne forme su uređeni parovi konačnih skupova atoma. Zovemo ih *klauzulama*, i klauzulu $(\{r_1, \ldots, r_n\}, \{q_1, \ldots, q_m\})$ standardno zapisujemo u obliku

$$r_1, \ldots, r_n \leftarrow q_1, \ldots, q_m$$

(možemo shvatiti kao da nam je $\leftarrow$ kanonsko ime za relaciju klauzalne forme, a kao i obično ispuštamo vitičaste zagrade kod konačnih skupova). Klauzalnu formu obično pišemo tako da sve njene klauzule pišemo jednu ispod druge, kao sustav, ali ako je želimo napisati kao skup, moramo klauzule staviti u zagrade (inače ne znamo koji zarez razdvaja atome u klauzuli, a koji razdvaja klauzule). Ako klauzulu pišemo s velikim slovima $A \leftarrow B$, podrazumijevamo da su A i B skupovi atoma, a ne atomi.

Slično kao za formule, za interpretaciju I kažemo da je *interpretacija za* klauzulu $A \leftarrow B$ ako je definirana na čitavom skupu $Var(A \leftarrow B) := A \cup B$. Kažemo da je klauzula $A \leftarrow B$ *istinita pod* interpretacijom I ako je $I(r) = 1$ za neki $r \in A$, ili je $I(q) = 0$ za neki $q \in B$; kažemo da je *lažna pod* interpretacijom I ako je $I(r) = 0$ za sve $r \in A$, te je $I(q) = 1$ za sve $q \in B$. Primijetimo da klauzula može biti lažna samo pod interpretacijom za nju, ali istinita može biti i pod interpretacijom koja nije definirana na svim njenim atomima.

Zadatak 8. Dokažite: klauzula $A \leftarrow B$ je *valjana* (istinita pod praznom interpretacijom) ako i samo ako A i B nisu disjunktni. Ta klauzula je *proturječna* ako i samo ako su A i B prazni. Dakle, postoji samo jedna proturječna klauzula ($\leftarrow$): zovemo je *praznom klauzulom*.

Definicija 9. Kao i za skupove formula, za klauzalnu formu kažemo da je istinita pod interpretacijom ako su sve klauzule u njoj istinite pod tom interpretacijom. *Interpretacija za* klauzalnu formu je interpretacija za svaku njenu klauzulu. Primijetimo, prazna klauzalna forma (prazna relacija) je istinita pod svakom interpretacijom.

Naravno, sad možemo definirati ekvivalentnost dviju klauzalnih formi, ili klauzalne forme i formule: ekvivalentne su ako imaju istu istinitost pod svakom interpretacijom za obje.

Za klauzalnu formu $K := \{(A_1 \leftarrow B_1), \ldots, (A_n \leftarrow B_n)\}$ kažemo da je *savršena* ako nijedna njena klauzula nije valjana ($A_i \cap B_i = \emptyset$ za sve i), te sve klauzule imaju isti skup atoma (Var). Kažemo da je klauzalna forma K *savršena za* formulu F ako je savršena, ekvivalentna s F, te za svaku klauzulu $(A \leftarrow B) \in K$ vrijedi $Var(A \leftarrow B) = Var(F)$.

Teorem 3. Za svaku oborivu formulu postoji savršena klauzalna forma.

Dokaz. Neka je F bilo koja oboriva formula. Po teoremu 2 postoji njoj

odgovarajući veznik v_F. Prema onom što smo rekli nakon tog teorema, bez smanjenja općenitosti možemo pretpostaviti da je $Var(F) = \{p_1, \ldots, p_k\}$.

Promotrimo sada "jezgru" od v_F: skup svih k-torki koje v_F preslikava u 0. Za svaku takvu k-torku $\vec{x} := (x_1, \ldots, x_k)$ konstruiramo klauzulu $A_{\vec{x}} \leftarrow B_{\vec{x}}$, gdje su

$$A_{\vec{x}} := \{p_i \in Var(F) \ : \ x_i = 0\}$$
$$B_{\vec{x}} := \{p_i \in Var(F) \ : \ x_i = 1\}$$

Iz konstrukcije skupova je očito da je uvijek $A \cap B = \emptyset$, te je $A \cup B = Var(F)$. Kako je F oboriva, "jezgra" je neprazna, pa za dokazati da je

$$K := \{A_{\vec{x}} \leftarrow B_{\vec{x}} \ : \ v_F(\vec{x}) = 0\}$$

savršena klauzalna forma za F, samo još treba dokazati $K \Leftrightarrow F$.

Neka je I interpretacija za K i F. Ako je $I(K) = 0$, tada postoji pod I lažna klauzula $(A_{\vec{x}} \leftarrow B_{\vec{x}}) \in K$. Dakle za sve $p_i \in A_{\vec{x}}$ je $I(p_i) = 0$, dok je za sve $p_i \in B_{\vec{x}}$ pak $I(p_i) = 1$. Ukratko, iz definicije skupova $A_{\vec{x}}$ i $B_{\vec{x}}$ vidimo da je $x_i = I(p_i)$ za sve i, te je prema teoremu 2

$$I(F) = v_F(I(p_1), \ldots, I(p_k)) = v_F(x_1, \ldots, x_k) = v_F(\vec{x}) = 0$$

(ovo zadnje jer je $(A_{\vec{x}} \leftarrow B_{\vec{x}}) \in K$, po definiciji K).

Ako je pak $I(K) = 1$, trebamo dokazati da je $I(F) = 1$. Pretpostavimo suprotno da je $I(F) = 0$ (kako je I interpretacija za F, doista je $I(F)$ ili 1 ili 0). To bi značilo da za $\vec{x} := (I(p_1), \ldots, I(p_k))$ vrijedi $0 = I(F) = v_F(\vec{x})$, pa je $(A_{\vec{x}} \leftarrow B_{\vec{x}}) \in K$.

Za svaki $p_i \in A_{\vec{x}}$ po definiciji $A_{\vec{x}}$ je $I(p_i) = x_i = 0$, dok je za svaki $p_i \in B_{\vec{x}}$ jednako tako $I(p_i) = x_i = 1$. Dakle klauzula $(A_{\vec{x}} \leftarrow B_{\vec{x}}) \in K$ je lažna pod interpretacijom I, što je kontradikcija s pretpostavkom $I(K) = 1$. □

Što je s valjanim formulama? Vidjeli smo u zadatku 8 da je prazna klauzalna forma valjana, i trivijalno je savršena. Ipak, to nije savršena klauzalna forma *za* bilo koju valjanu formulu F, jer je $Var(\emptyset) = \emptyset \neq Var(F)$. Štoviše, imamo sljedeći teorem.

Teorem 4. Neka je F valjana formula. Tada ne postoji savršena klauzalna forma za F.

Dokaz. Pretpostavimo da je K savršena klauzalna forma za F. Kako mora biti $K \Leftrightarrow F$, vidimo da i K mora biti valjana. No budući da je $Var(F) \neq \emptyset = Var(\emptyset)$, K ne smije biti prazna. Neka je $A \leftarrow B$ neka klauzula u K. Kako je K savršena, $A \cap B = \emptyset$. Sada po zadatku 8 slijedi da postoji interpretacija pod kojom je $A \leftarrow B$ lažna, pa je pod tom interpretacijom i K lažna, što je kontradikcija s njenom valjanošću. □

2.3 Konjunktivna normalna forma

Vidjeli smo da za svaku formulu F postoji savršena klauzalna forma ekvivalentna njoj, te da ona ima iste varijable kao i F ako je F oboriva. Sad nam je cilj vratiti se natrag u formule, te vidjeti da klauzule i klauzalne forme također možemo prikazati pomoću veznika.

Prisjetimo se, konjunkcija je definirana za svaku mjesnost k kao minimum istinitosnih vrijednosti, dok je disjunkcija definirana kao maksimum. Specijalni slučajevi su:

$$\wedge^0 = \top \qquad \vee^0 = \bot \qquad \wedge^1 = \vee^1 = id \qquad \wedge^2 = \wedge \qquad \vee^2 = \vee$$

Definicija 10. *Formulska reprezentacija* klauzule $r_1, \ldots, r_n \leftarrow q_1, \ldots, q_m$ je formula $(q_1 \wedge \cdots \wedge q_m) \rightarrow (r_1 \vee \cdots \vee r_n)$.

Formulska reprezentacija konačne klauzalne forme je konjunkcija formulskih reprezentacija svih njenih klauzula.

Zadatak 9. Dokažite: svaka klauzula, i svaka konačna klauzalna forma, ekvivalentna je svojoj formulskoj reprezentaciji.

Vidimo da svaki veznik možemo zapisati koristeći samo konjunkciju, disjunkciju i kondicional: njegovu klauzalnu formu pretvorimo u formulsku reprezentaciju. Možemo se riješiti i kondicionala, ako uvedemo negaciju.

Zadatak 10. Dokažite: za sve atome $q_1, \ldots, q_m, r_1, \ldots, r_n$, vrijedi

$$(q_1 \wedge \cdots \wedge q_m) \rightarrow (r_1 \vee \cdots \vee r_n) \Longleftrightarrow (\neg q_1 \vee \cdots \vee \neg q_m \vee r_1 \vee \cdots \vee r_n) \,.$$

Definicija 11. Atomarne formule i negacije atomarnih formula zovemo *literalima*. Dakle, svaka formulska reprezentacija klauzule jednaka je disjunkciji literala, koju još zovemo *elementarnom disjunkcijom*. Sada je jasno da

je svaka formulska reprezentacija konačne klauzalne forme ekvivalentna konjunkciji elementarnih disjunkcija, koju zovemo *konjunktivnom normalnom formom* početne klauzalne forme.

Napomena 3. Često se u literaturi *klauzulama* zovu ono što smo mi nazvali *elementarne disjunkcije*, dok se *klauzalnom formom* zove ono što smo mi nazvali *konjunktivna normalna forma*.

Iako su, kao što smo vidjeli, ti pojmovi vrlo bliski i pretvorbe u oba smjera su trivijalne i „mehaničke", dobro je razlikovati te pojmove: prvo, jer su pojmovi elementarne disjunkcije i konjunktivne normalne forme uvedeni već u prijašnjem kolegiju (Matematička logika), a drugo, jer su klauzule kao zasebni entiteti izuzetno korisne za shvaćanje logičkog programiranja.

Intuitivno, formule su statički objekti, koji ništa ne „rade". Klauzule, s druge strane, omogućuju *izračunavanje* i *pretraživanje* prostora mogućnosti. Više o tome reći ćemo kad budemo pričali o logičkom programiranju.

2.4 Izrazivost veznika

Sada ćemo se malo detaljnije pozabaviti veznicima. U definiciji formule smo samo rekli da je veznik bilo koja funkcija sa $\{0,1\}^k$ u $\{0,1\}$ za neki $k \in \mathbb{N}$. Preciznije, za svaki prirodni k imamo skup $Conn^k := \{0,1\}^{\{0,1\}^k}$ veznika mjesnosti k, dok je skup svih veznika jednak $Conn := \bigcup_{k\in\mathbb{N}} Conn^k$.

Definicija 12. Neka je $A \subseteq Conn$. Za formulu kažemo da je *A-formula* ako je ili atomarna, ili joj je glavni veznik element od A, a sve glavne potformule su joj A-formule. Neformalno, formula je A-formula ako su svi veznici koji se u njoj pojavljuju, elementi od A.

Za veznik $v \in Conn$ kažemo da je *izraziv pomoću A* ako odgovara nekoj A-formuli. Za *skup* $B \subseteq Conn$ kažemo da je *izraziv pomoću A* ako je svaki veznik iz B izraziv pomoću A. Za A kažemo da je *dovoljan skup veznika* ako je čitav $Conn$ izraziv pomoću A.

Uzevši u obzir teorem 2, operativno dokazujemo da je v^k izraziv pomoću A tako da dokažemo da je formula s glavnim veznikom v i glavnim potformulama $p_1, \ldots, p_k$ ekvivalentna nekoj A-formuli (čiji skup atoma ne mora biti

jednak $\{p_1, \ldots, p_k\}$).

Rezultat iz prethodne točke sada možemo iskazati formalno: označimo

$$Conj := \{\wedge^0, \wedge^1, \wedge^2, \wedge^3, \wedge^4, \cdots\} \qquad Disj := \{\vee^0, \vee^1, \vee^2, \vee^3, \vee^4, \cdots\}$$

Teorem 5. $Cl := Conj \cup Disj \cup \{\rightarrow\}$ je dovoljan skup veznika.

Dokaz. Neka je $v \in Conn$ bilo koji veznik, i označimo njegovu mjesnost s k. Ako v nikad ne poprima vrijednost 0, očito odgovara bilo kojoj tautologiji, recimo $p \rightarrow p$, koja je Cl-formula jer je $\rightarrow \in Cl$.

Ako pak v može poprimiti vrijednost 0, to znači da je formula F čiji je glavni veznik v a glavne potformule su joj $p_1, \ldots, p_k$ redom, oboriva. Kako je po definiciji $v_F^k = v$, dovoljno je dokazati da je F ekvivalentna nekoj Cl formuli.

Znamo da za F postoji savršena klauzalna forma C. Po definiciji je dakle $F \Leftrightarrow C$. Za C (koja je očito konačna, $|C| \leq 2^k$) postoji formulska reprezentacija G, koja joj je ekvivalentna po zadatku 9. Lako je sada vidjeti da G jest Cl-formula, te je $G \Leftrightarrow F$. □

Sada bismo htjeli iskoristiti zadatak 10 da dokažemo da je skup veznika $\{\neg\} \cup Cl \setminus \{\rightarrow\}$ također dovoljan. Formalno, za to nam treba sljedeći rezultat.

Teorem 6. Neka su $A, B \subseteq Conn$. Tada je B izraziv pomoću A ako i samo ako je svaka B-formula ekvivalentna nekoj A-formuli.

Dokaz. Jedan smjer je jednostavan: ako je svaka B-formula ekvivalentna nekoj A-formuli, tada je specijalno za svaki $v^k \in B$, formula s glavnim veznikom v^k i glavnim potformulama $p_1, \ldots, p_k$ ekvivalentna nekoj A-formuli F, a to upravo znači $v_F^k = v^k$.

Drugi smjer dokazujemo indukcijom po B-formulama. Atomarne formule su očito ekvivalentne same sebi, i A-formule su. Neka je sada F formula s glavnim veznikom $v^k \in B$, kojoj su sve glavne potformule (nazovimo ih $F_1, \ldots, F_k$) B-formule. Za njih vrijedi pretpostavka indukcije, pa je svaka od njih ekvivalentna nekoj A-formuli, označimo jednu takvu s $G_i :\Leftrightarrow F_i$.

S druge strane, sam v^k, kao element od B i po pretpostavci izraziv pomoću A, odgovara nekoj A-formuli H. Označimo

$$G := H[p_1 \mapsto G_1, \ldots, p_k \mapsto G_k] \,.$$

Očito je G također A-formula (dobivena je supstitucijom nekoliko A-formula u A-formulu). Želimo dokazati $F \Leftrightarrow G$. U tu svrhu, neka je I_1 interpretacija za obje. Definiramo pomoćnu interpretaciju I_2 sa

$$I_2(p_i) := I_1(G_i) \quad \text{za sve} \quad i \in \{1, \ldots, k\} \, .$$

Sada, uzimajući u obzir mnoge dosadašnje rezultate, dobivamo

$$I_1(F) = v^k(I_1(F_1), \ldots, I_1(F_k)) = v^k(I_1(G_1), \ldots, I_1(G_k)) =$$

$$= v^k(I_2(p_1), \ldots, I_2(p_k)) = I_2(H) = I_1(H[p_1 \mapsto G_1, \ldots, p_k \mapsto G_k]) = I_1(G) \, .$$

□

Korolar 1. Relacija "je izraziv pomoću" na skupu $\mathcal{P}(Conn)$ je tranzitivna.

Dokaz. Neka su $A, B, C \subseteq Conn$ takvi da je A izraziv pomoću B, a B izraziv pomoću C. Neka je F proizvoljna A-formula. Po prethodnom teoremu, ona je ekvivalentna nekoj B-formuli G. Opet po prethodnom teoremu, G je ekvivalentna nekoj C-formuli H. Sada vidimo da je svaka A-formula ekvivalentna nekoj C-formuli, pa po prethodnom teoremu (obrnuti smjer) vidimo da je A izraziv pomoću C. □

Uvrštavanjem $A := Conn$ u dokaz prethodnog korolara odmah dobivamo

Korolar 2. Ako je dovoljan skup veznika C izraziv pomoću skupa veznika B, tada je i B dovoljan.

Budući da imamo jedan relativno strukturiran dovoljan skup (Cl), pomoću njega možemo dobivati ostale. Primijetimo da za dokaz da je A izraziv pomoću B, moramo naći odgovarajuće B-formule samo za veznike iz $A \setminus B$.

Konkretno, skup $Nf := Conj \cup Disj \cup \{\neg\}$ je dovoljan, jer je Cl izraziv pomoću njega. Jedini veznik iz $Cl \setminus Nf$ je $\rightarrow$, a on je izraziv pomoću Nf jer je $p_1 \rightarrow p_2 \Leftrightarrow \neg p_1 \vee p_2$ (po definiciji od $\rightarrow$).

Također, skup $Basic := \{\wedge, \vee, \neg\}$ je dovoljan (i prvi primjer *konačnog* dovoljnog skupa koji imamo) jer je Nf izraziv pomoću njega. Naime, indukcijom po k lako dobijemo da je za svaki $k \geq 2$, $\wedge^k$ izraziv pomoću $Basic$ (u koraku, izrazimo ga pomoću $\wedge = \wedge^2$ i $\wedge^{k-1}$, i primijenimo tranzitivnost), i jednako tako za $\vee^k$.

Još su nam ostali $\vee^1 = \wedge^1 = id$, što možemo riješiti primjerice pomoću $id\, p_1 \Leftrightarrow (p_1 \wedge p_1)$, te $\vee^0 = \bot \Leftrightarrow (p \wedge \neg p)$ i $\wedge^0 = \top \Leftrightarrow (p \vee \neg p)$.

Sada je lako vidjeti da je $\{\neg, \vee\}$ dovoljan (de Morganovo pravilo nam daje $\wedge$), pa je i $\{\neg, \to\}$ dovoljan ($(p_1 \vee p_2) \Leftrightarrow (\neg p_1 \to p_2)$), a onda je i $\{\bot, \to\}$ dovoljan ($\neg p_1 \Leftrightarrow (p_1 \to \bot)$).

Zadatak 11. (a) Dokažite da skup $\{\neg, \leftrightarrow\}$ nije dovoljan.

(b) Dokažite da skup $Conn^0 \cup Conn^1$ nije dovoljan.

(c) Nađite neki jednočlan dovoljan skup veznika, i dokažite da je dovoljan.

Poglavlje 3

Rezolucija

3.1 SAT problem

U prošlom poglavlju definirali smo klauzule i klauzalne forme, i vidjeli kako se pomoću njih mogu zapisati proizvoljne formule. Svaka tautologija ekvivalentna je praznoj klauzalnoj formi, dok svaka oboriva formula ima svoju (jedinstvenu) savršenu klauzalnu formu kojoj je ekvivalentna. Dakle, klauzalne forme predstavljaju neku vrstu kanonskog oblika zapisa formule.

Vidjeli smo da je klauzalna forma valjana ako i samo ako su sve njene klauzule valjane (ovo naravno uključuje i slučaj kad je prazna), a valjane je klauzule lako prepoznati: samo mora postojati atom koji se nalazi i na lijevoj i na desnoj strani.

Drugim riječima, za klauzalnu formu postoji *linearni* algoritam koji ustanovljuje je li valjana. Ovisno o implementaciji atoma i skupova taj algoritam može biti $O(n \log n)$ ili čak $O(n^2)$, ali u svakom slučaju je polinoman.

S druge strane, *ispunjivost* klauzalne forme je puno teže ispitati. To je slavni SAT (*satisfiability*) problem, prvi problem za kojeg je dokazano da je *NP-potpun*. Više o tome čut ćete na kolegiju Složenost algoritama, ali ukratko, NP označava klasu problema za koje je u polinomnom vremenu ($p(n)$ koraka, gdje je p neki polinom a n broj bitova ulaza) moguće provjeriti je li zadani niz bitova (opet, veličine ograničene polinomom od n) validni *certifikat* za rješenje.

Recimo, problem provjere je li zadani broj složen je u NP: faktorizacija može biti teška, ali ako nam netko dade faktor, lako je provjeriti da je ulaz njime djeljiv u polinomnom vremenu.

Svakako, sada vidimo da je SAT u NP: naći interpretaciju može biti teško, ali ako imamo interpretaciju, lako je u polinomnom (precizno, kvadratnom) vremenu *provjeriti* da je pod njom forma istinita. Zapravo vrijedi puno više od toga: ne samo da je SAT u NP, već se svaki problem X u NP može *svesti* na SAT: postoji polinomni algoritam koji preslikava instance problema X u instance problema SAT (to su klauzalne forme), čuvajući rješivost: zadana instanca problema X ima rješenje ako i samo ako je odgovarajuća klauzalna forma ispunjiva.

Iz toga slijedi da, kad bismo imali polinomni algoritam za rješavanje problema SAT, zapravo bismo imali polinomni algoritam za rješavanje *svih* problema u NP, odnosno vrijedilo bi $P = NP$ (P označava klasu problema za čije rješavanje postoji polinomni algoritam). Mnogi stručnjaci za složenost u to sumnjaju, a za matematički dokaz točnog odnosa između P i NP Clay Institute nudi milijun dolara još od 2000. godine.

Iako mnogi vjeruju da svaki algoritam za SAT problem ima puno slučajeva u kojima nema bitno boljeg rješenja od jednostavnog eksponencijalnog ispitivanja svih 2^n mogućnosti (gdje je n broj različitih atoma koji se pojavljuju u klauzalnoj formi), ipak postoje mnoge heuristike, koje dodatno dobivaju na značenju činjenicom da su mnogi vrlo stvarni problemi (primjerice, problem rasporeda sati) u NP, pa se mogu izraziti kao instance SAT problema.

Ne sve, ali mnoge od tih heuristika zasnivaju se na *rezoluciji*, jednostavnom logičkom principu koji nam omogućuje da od dvije klauzule dobijemo jednu novu, koju onda možemo koristiti dalje. No prvo pogledajmo neke slučajeve koje možemo riješiti jednustavnijim heuristikama.

3.2 Jednostavni slučajevi i DPLL algoritam

Iako moramo samo odrediti je li forma ispunjiva (da/ne), za dalja razmišljanja pomoći će ako se ponašamo kao da moramo baš naći interpretaciju I_1 koja je čini istinitom.

Prvo, očito atome koji se uopće ne pojavljuju možemo ignorirati, odnosno interpretaciju I_1 ne moramo definirati na njima. Što je s atomima koji se pojavljuju samo jednom? Lako je vidjeti da ih možemo iskoristiti kao "džokere": ako se pojavljuju na lijevoj strani klauzule, definiramo I_1 na njima kao 1, a ako se pojavljuju na desnoj, definiramo I_1 na njima kao 0.

Ta klauzula time postaje zadovoljena i možemo je brisati iz klauzalne forme. Štoviše, sad se lako vidi da potpuno isti pristup funkcionira i ako se određeni atom pojavljuje više puta, ali uvijek na istoj strani klauzule (uvijek na lijevoj, ili uvijek na desnoj). Takvi atomi zovu se *čisti* atomi.

Idemo dalje. Što ako se neki atom pojavljuje dvaput, na različitim stranama klauzule? Ako se pojavljuje na različitim stranama *iste* klauzule, tad znamo da je ta klauzula tautologija, pa je uvijek zadovoljena i možemo je brisati. Netrivijalan slučaj nastupa ako se isti atom pojavljuje na različitim stranama dviju različitih klauzula, i razmatranje tog slučaja vodi na rezoluciju. Zasad samo pogledajmo još nekoliko jednostavnih slučajeva.

Ako se neka („jedinična") klauzula sastoji samo od jednog atoma p, vrijednost $I_1(p)$ je time fiksirana: ako imamo klauzulu $p \leftarrow$, mora biti $I_1(p) = 1$, a ako imamo klauzulu $\leftarrow p$, mora biti $I_1(p) = 0$. Time ne samo da je ta klauzula zadovoljena, već možemo brisati p iz svih ostalih klauzula u kojima se pojavljuje. Ako se pojavljuje na istoj strani kao u jediničnoj klauzuli, možemo obrisati i čitavu tu klauzulu, jer znamo da je ispunjena. Ako je na suprotnoj strani nego u jediničnoj klauzuli, možemo samo brisati taj atom, ali time i ta klauzula može postati jedinična, te se postupak može nastaviti dalje.

Zapažanje iz prethodnog odlomka možemo generalizirati na još jedan način: za klauzulu $A \leftarrow B$ kažemo da *obuhvaća* klauzulu $C \leftarrow D$ ako je $A \subseteq C \wedge B \subseteq D$. Lako je vidjeti da u tom slučaju bilo koja interpretacija pod kojom je istinita $A \leftarrow B$, čini istinitom i klauzulu $C \leftarrow D$ — odnosno, ako je $A \leftarrow B$ element naše klauzalne forme čiju ispunjivost ispitujemo, tada možemo iz nje brisati sve klauzule $C \leftarrow D$ koje su njome obuhvaćene.

Sistematska razrada upravo opisanih ideja vodi na DPLL, vjerojatno najpoznatiji „generički" algoritam za rješavanje SAT problema. Na profinjenjima toga algoritma radi se i danas, prvenstveno u tri smjera:

1. Ako ne nastupa nijedan od gore navedenih specijalnih slučajeva, moramo pametno odabrati neki atom, te vrijednost koju ćemo mu pokušati

pridružiti. Naravno, ako ta grana ne uspije, moramo se vratiti (*backtrack*) i pokušati mu pridružiti suprotnu vrijednost. Strategijâ za odabir ima puno, vrlo je netrivijalno naći pravu, a bitno utječu na kasnije odvijanje algoritma i veličinu preostalog prostora mogućnosti koje treba ispitati.

2. Eliminacija atoma koji se pojavljuju u jediničnim klauzulama, rekli smo, može dovesti do stvaranja novih jediničnih klauzula, što može stvoriti duge „domino-lance" klauzulâ. Netrivijalno je smisliti dobru strukturu podataka koja će takvu operaciju izvršavati vrlo brzo, pogotovo uzevši u obzir da i vraćanje prethodnog stanja mora biti brzo zbog čestog *backtracking*a.

3. U zadnjem desetljeću naučili smo mnoge nove trikove vezane uz strojno učenje. Vrlo je plodno područje pokušaja implementacije tih trikova u DPLL algoritmu. Recimo, jedna od mogućnosti je natjerati algoritam da uoči ako mu se neki *cluster* u I_1 uvijek iznova pokaže proturječnim, te ga počne izbjegavati (odnosno prije njegovog ispitivanja pokušati nešto drugo) u kasnijim grananjima.

3.3 Rezolucija

Jednostavna analogija koja nam može pomoći razumjeti ideju rezolucije je sljedeća: zamislimo da ispred sebe imamo puno klauzula, i moramo naći vrijednosti koje ćemo pridružiti atomima tako da sve one budu istinite. Kad ga gledamo na taj način, problem donekle podsjeća na rješavanje sustava jednadžbi: svaka može biti zadovoljena na razne načine jer ima puno nepoznanica, ali uvjet da *sve* moraju biti zadovoljene bitno smanjuje broj mogućnosti.

Tehnika koju obično koristimo kod rješavanja sustava algebarskih jednadžbi je *supstitucija*. Njome iz para jednadžbi eliminiramo jednu nepoznanicu, tako da je izrazimo iz jedne i uvrstimo izraz za nju u onu drugu. Zapravo, često ako su jednadžbe dovoljno pravilne strukture, i ne moramo izražavati pa uvrštavati: jednostavnim agregiranjem (zbrajanjem) jednadžbi kojima se nepoznanica nalazi na suprotnim stranama, ona se reducira (poni-

štava) i time nestaje iz rezultatne jednadžbe.

$$\left.\begin{aligned} x + \not{y} &= a + b \\ c &= \not{y} + z \end{aligned}\right\} \quad \Longrightarrow \quad x + c = a + b + z$$

Klauzule *jesu* vrlo pravilne strukture (iako nesimetričnost znači da bolje odgovaraju *nejednadžbama* nego jednadžbama), i pokazuje se da je vrlo sličan manevar moguće napraviti s njima.

Dakle, rezoluciju primjenjujemo kad imamo isti atom na različitim stranama dvije klauzule. Takve klauzule zovemo *ulančanima*. Precizno, klauzule $A \leftarrow B$ i $C \leftarrow D$ su ulančane ako je $(A \cap D) \cup (B \cap C) \neq \emptyset$. Odaberimo jedan element tog presjeka, neka je to q. (Poslije ćemo vidjeti da je zapravo jedini zanimljiv slučaj kad je q jedinstven.) Bez smanjenja općenitosti (inače zamijenimo klauzule), $q \in A \cap D$. Vrijednost $I_1(q)$ je ili 0 ili 1. Ako je 0, klauzula $C \leftarrow D$ je istinita (jer je $q \in D$), i trebamo samo gledati $A \setminus \{q\} \leftarrow B$. Ako je pak 1, klauzula $A \leftarrow B$ je istinita (jer je $q \in A$), i trebamo samo gledati $C \leftarrow D \setminus \{q\}$.

Dakle, ako je I_1 rješenje, tada je ili neki atom iz $A \setminus \{q\}$ istinit, ili je neki atom iz B lažan, ili je neki atom iz C istinit, ili je neki atom iz $D \setminus \{q\}$ lažan. Pišući to drugim redom (grupirajući posebno istinite a posebno lažne mogućnosti), dobivamo da je pod I_1 istinita klauzula

$$(A \setminus \{q\}) \cup C \leftarrow B \cup (D \setminus \{q\}) \,,$$

koju zovemo *rezolventom* klauzula $A \leftarrow B$ i $C \leftarrow D$, i označavamo je oznakom $(C \leftarrow D)\ \not{q}\ (A \leftarrow B)$. Proces kojim je ta klauzula dobivena od početnih zovemo *rezolucijom*.

Vidjeli smo da ako su dvije klauzule ulančane i istinite pod nekom interpretacijom, tada je pod istom interpretacijom istinita i njihova rezolventa s obzirom na neki atom po kojem su ulančane. Što ako ima više takvih atoma? Lako je vidjeti da ako su različiti atomi q i r svaki na različitim stranama dvije klauzule, tada ako napravimo rezoluciju po q, rezolventa će imati r na obje strane, i jednako tako ako napravimo rezoluciju po r, rezolventa će imati q na obje strane. U svakom slučaju rezolventa će biti tautologija, pa je možemo brisati iz klauzalne forme. Iz tog razloga rezolventa se obično gleda samo kad imamo jedinstveni atom na suprotnim stranama dvije različite klauzule.

Opet, jasno je da se postupak može provesti i ako se q javlja na više od dva mjesta, u ostalim klauzulama. Tada nismo eliminirali nepoznanicu u potpunosti, ali smo svejedno napravili korak prema pojednostavljenju.

Nažalost, kao i u slučaju rješavanja linearnih sustava, ne možemo rezolvirane klauzule jednostavno zamijeniti rezolventom, već moramo rezolventu dodati u skup klauzula koje razmatramo. Ipak, taj postupak će za konačne klauzalne forme stati iz jednostavnog razloga što u početku imamo samo konačno mnogo atoma, rezolucija ne stvara nove atome, a *svih* klauzula nad konačno mnogo atoma ima samo konačno mnogo.

Zadatak 12. - Dokažite da svih relevantnih (ispunjivih i oborivih) klauzula nad n atoma ima $3^n - 1$.

Dakle, algoritam rezolucije je sljedeći:

Require: S je konačna klauzalna forma (skup klauzula).
  for all klauzule $(A \leftarrow B), (C \leftarrow D) \in S$ **do**
    if $|A \cap D| = 1$ & $B \cap C = \emptyset$ **then**
      $q :=$ jedini element od $A \cap D$
      rezolventa $:= (A \setminus \{q\}) \cup C \leftarrow B \cup (D \setminus \{q\})$
      if rezolventa $= (\leftarrow)$ **then return** „S je proturječna!”
      $S := S \cup \{\text{rezolventa}\}$

Sasvim je jasno iz prethodnih razmatranja (ako su dvije ulančane klauzule istinite pod I_1, tada je i njihova rezolventa istinita pod istom interpretacijom) da algoritam rezolucije pokrenut na ispunjivoj klauzalnoj formi S može generirati samo ispunjive rezolvente, pa nikada ne može javiti da je S proturječna. To znači da je algoritam rezolucije *adekvatan* za logiku sudova.

3.4 Potpunost algoritma rezolucije

Zapravo vrijedi i obrat: algoritam rezolucije je i *potpun* za logiku sudova, što znači da ako je početna klauzalna forma proturječna, algoritam ne može ispitati *sve* parove klauzula (uključujući novododane klauzule) a da ne generira praznu klauzulu. Štoviše, jer svih klauzula koje algoritam rezolucije može generirati ima konačno mnogo, algoritam će sigurno stati, i adekvatnost i

potpunost tada znače da će dati točan rezultat: „S je proturječna" ako i samo ako S doista jest proturječna.

Precizan dokaz potpunosti može se naći u [1] i predstavlja dobru temu za seminar. Ovdje samo dajemo ideju dokaza.

Poredajmo atome u S u kanonskom redoslijedu (slova po abecedi, neindeksirani prije indeksiranih, indeksirani atomi istog slova po indeksu) i nazovimo ih a_1, a_2, ..., a_n. Za $1 \leq k \leq n$, k-klauzulom zovemo oborivu klauzulu $A \leftarrow B$ čiji je Var skup prvih k atoma $\{a_1, ...a_k\}$. Sada promotrimo potpuno binarno stablo visine n na čijoj k-toj razini se nalaze upravo sve k-klauzule (njih 2^k), te su djeca (za $k < n$) svake takve klauzule $A \leftarrow B$ dvije klauzule koje nastaju dodavanjem atoma a_{k+1} na svaku stranu ($A \cup \{a_{k+1}\} \leftarrow B$ lijevo, $A \leftarrow B \cup \{a_{k+1}\}$ desno).

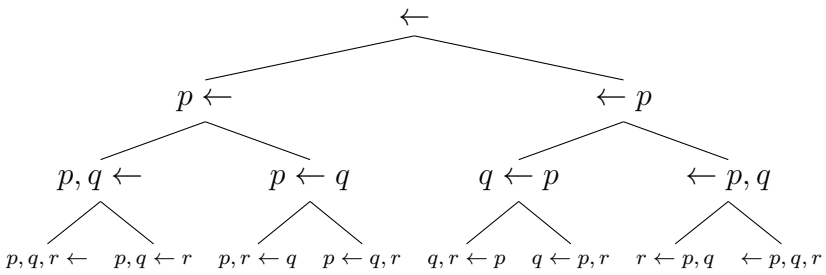
Dakle, u korijenu je prazna klauzula: jedina 0-klauzula. Također primijetimo da u svakom putu u stablu, gornja klauzula obuhvaća donju. Za kraj još primijetimo da je svaki unutarnji čvor upravo rezolventa svoje djece:

$$(A \cup \{a_{k+1}\} \leftarrow B) \underset{a_{k+1}}{\diagup\!\!\diagdown} (A \leftarrow B \cup \{a_{k+1}\}) = A \leftarrow B \ .$$

Neke od tih $2^{n+1} - 1$ klauzula su u S, ali moguće je da u S imamo i klauzule koje nisu u stablu (npr. $a_3 \leftarrow a_1$). Ipak, za svaku klauzulu $K \in S$ ima bar jedna (među listovima njih točno $2^{n-|Vars(K)|}$) klauzula koju K obuhvaća.

Ako sada u stablu „blokiramo" sve klauzule obuhvaćene elementima iz S, lako se vidi da proturječnost od S znači da je svaki put od korijena do lista negdje blokiran. Naime, dualno, putovi od korijena do lista odgovaraju interpretacijama za S, pri čemu izbor smjera kojim treba ići na k. razini odgovara vrijednosti interpretacije na a_k (lijevo znači $I(a_k) = 0$, desno znači $I(a_k) = 1$). Lako se vidi da takva interpretacija čini lažnima samo klauzule do kojih njen put dolazi. Kad neki put ne bi bio blokiran, interpretacija njime određena bi činila istinitom čitav S.

Blokirane klauzule ne moraju dakle nužno biti doista elementi od S, ali možemo se pretvarati kao da jesu: obuhvaćene su njima, pa će rezolucija na blokiranim klauzulama dati klauzulu koja je obuhvaćena rezolventom neke dvije klauzule koje doista jesu u S. (Ovo je najmutniji dio, zbog kojeg je ovo samo ideja dokaza. Trebalo bi pokazati da rezolucija čuva relaciju uređaja po obuhvaćenosti.)



Sada je ključno primijetiti da svako takvo „blokiranje" mora ili imati blokiran korijen (u kom slučaju smo gotovi, jer je prazna klauzula element od S), ili imati neblokiran unutarnji čvor čija oba djeteta su blokirana. Primjerice, bilo koji neblokirani čvor maksimalne udaljenosti do korijena mora imati to svojstvo.

No vidjeli smo da je svaki čvor, pa tako i taj, rezolventa svoje djece. Kako su mu djeca blokirana, rezolucijom možemo dobiti njega, obuhvaćenog nekom rezolventom elemenata iz S. To znači da i taj čvor možemo blokirati. Sada smo dobili novo stablo, iste strukture kao prije, samo s jednim blokiranim čvorom više.

Kako je i s tim stablom moguće ponoviti isti postupak, i tako u nedogled, a svih čvorova ima konačno mnogo, postupak će prije ili kasnije završiti, a jedino može završiti blokiranjem korijena. To bi značilo da je prazna klauzula obuhvaćena nekim elementom koji je došao u S — no prazna klauzula može biti obuhvaćena jedino samom sobom, dakle prazna klauzula je u nekom trenutku došla u S.

Kako algoritam rezolucije ispituje parove klauzula nekim redom, prije ili kasnije će naići upravo na par koji smo razmatrali ovdje, te će nakon toga u nekom trenutku „nabasati" na pravi par za dalje smanjenje, i tako dalje. Dakle, nakon dovoljno mnogo vremena, algoritam rezolucije pokrenut na proturječnoj klauzalnoj formi otkrit će da je proturječna (možda vrlo brzo, ako mu se „posreći", a možda vrlo sporo, ali svakako u konačno mnogo koraka).

Jednako tako, ispunjivost klauzalne forme znači postojanje bar jednog puta od korijena do lista koji nije blokiran. Tada ako označimo sve takve putove, lako se vidi da primjena rezolucije nikada ne može blokirati nijedan čvor na takvim putovima. Kad označi sve ostale čvorove, algoritam rezolucije će stati, te ćemo znati da forma nije proturječna.

Takvo stablo, tzv. semantičko stablo, pruža dobar uvid i u ideju dokaza teorema kompaktnosti za logiku sudova: čak i ako je stablo beskonačno u visinu, svejedno će svaka staza od korijena prije ili kasnije biti blokirana, ili će postojati beskonačni put od korijena prema dolje, što će generirati interpretaciju.

Bibliografija

[1] Mordechai Ben-Ari. *Mathematical Logic for Computer Science*. Springer Publishing Company, Incorporated, 3rd edition, 2012.