Třetí přednáška

NAIL062 Výroková a predikátová logika

Jakub Bulín (KTIML MFF UK)

Zimní semestr 2023

Třetí přednáška

Program

- algebra výroků
- problém splnitelnosti, SAT solvery
- 2-SAT a implikační graf
- Horn-SAT a jednotková propagace
- algoritmus DPLL

Materiály

Zápisky z přednášky, Sekce 2.5 z Kapitoly 2, Kapitola 3

2.5 Algebra výroků

Výroky až na ekvivalenci

Kolik existuje výroků nad $\mathbb{P} = \{p, q, r\}$? Nekonečně mnoho. Až na ekvivalenci? Tolik, kolik je možných množin modelů: $2^{2^3} = 256$.

Výroky až na ekvivalenci studujeme pomocí jejich množin modelů.

Ekvivalenční třídy: $\mathsf{VF}_{\mathbb{P}}/\sim$, např. $[p \to q]_\sim = \{p \to q, \neg p \vee q, \ldots\}$

Přiřazení modelů: $h : \mathsf{VF}_{\mathbb{P}}/\sim \to \mathcal{P}(\mathrm{M}_{\mathbb{P}})$ definované $h([\varphi]_\sim) = \mathrm{M}(\varphi)$ (je dobře definované, prosté, pro konečný jazyk bijekce)

Na $\mathsf{VF}_{\mathbb{P}}/\sim$ zavedeme operace $\neg, \wedge, \vee$ pomocí reprezentantů:

$$\neg[\varphi]_\sim = [\neg\varphi]_\sim$$
$$[\varphi]_\sim \wedge [\psi]_\sim = [\varphi \wedge \psi]_\sim$$
$$[\varphi]_\sim \vee [\psi]_\sim = [\varphi \vee \psi]_\sim$$

přidáme konstanty $\perp = [\perp]_\sim, \top = [\top]_\sim$, máme *Booleovu algebru*: algebru výroků jazyka $\mathbb{P}$; totéž relativně k teorii T (použijeme $\sim_T$)

Algebra výroků

Algebra výroků jazyka $\mathbb{P}$ resp. teorie T:

$$\mathbf{AV}_{\mathbb{P}} = \langle {}^{\mathrm{VF}_{\mathbb{P}}}/{\sim}; \neg, \wedge, \vee, \bot, \top \rangle$$

$$\mathbf{AV}_{\mathbb{P}}(T) = \langle {}^{\mathrm{VF}_{\mathbb{P}}}/{\sim_T}; \neg_T, \wedge_T, \vee_T, \bot_T, \top_T \rangle$$

přiřazení modelů h je prosté zobrazení algebry výroků jazyka do potenční algebry $\mathcal{P}(\mathrm{M}_{\mathbb{P}}) = \langle \mathcal{P}(\mathrm{M}_{\mathbb{P}}); {}^{-}, \cap, \cup, \emptyset, \mathrm{M}_{\mathbb{P}} \rangle$ zachovávající operace a konstanty: $h(\bot) = \emptyset$, $h(\top) = \mathrm{M}_{\mathbb{P}}$, a

$$h(\neg[\varphi]_{\sim}) = \overline{h([\varphi]_{\sim})} = \overline{\mathrm{M}(\varphi)} = \mathrm{M}_{\mathbb{P}} \setminus \mathrm{M}(\varphi)$$
$$h([\varphi]_{\sim} \wedge [\psi]_{\sim}) = h([\varphi]_{\sim}) \cap h([\psi]_{\sim}) = \mathrm{M}(\varphi) \cap \mathrm{M}(\psi)$$
$$h([\varphi]_{\sim} \vee [\psi]_{\sim}) = h([\varphi]_{\sim}) \cup h([\psi]_{\sim}) = \mathrm{M}(\varphi) \cup \mathrm{M}(\psi)$$

tj. je to homomorfismus Booleových algeber, a nad konečným jazykem bijekce, tzv. izomorfismus; stejně pro algebru výroků teorie

Důsledek: Pro bezespornou teorii T nad *konečným jazykem* $\mathbb{P}$ je algebra výroků $\mathbf{AV}_{\mathbb{P}}(T)$ izomorfní potenční algebře $\mathcal{P}(\mathrm{M}_{\mathbb{P}}(\mathbf{T}))$ prostřednictvím zobrazení $h([\varphi]_{\sim_T}) = M(T, \varphi)$.

Počítání až na ekvivalenci

Tvrzení: Mějme n-prvkový jazyk $\mathbb{P}$ a bezespornou teorii T mající právě k modelů. Potom v jazyce $\mathbb{P}$ existuje až na ekvivalenci:

- 2^{2^n} výroků (resp. teorií),
- 2^{2^n-k} výroků pravdivých (resp. lživých) v T,
- $2^{2^n} - 2 \cdot 2^{2^n-k}$ výroků nezávislých v T,
- 2^k jednoduchých extenzí teorie T (z toho 1 sporná),
- k kompletních jednoduchých extenzí T.

Dále až na T-ekvivalenci existuje:

- 2^k výroků,
- 1 výrok pravdivý v T, 1 lživý v T,
- $2^k - 2$ výroků nezávislých v T.

Důkaz: stačí spočítat možné množiny modelů $\square$

Kapitola 3: Problém splnitelnosti

Problém SAT:

- vstup: výrok φ v CNF
- otázka: je φ splnitelný?

univerzální problém: každou teorii nad konečným jazykem lze převést do CNF

Cook-Levinova věta: SAT je NP-úplný (důkaz: formalizuj výpočet nedeterministického Turingova stroje ve výrokové logice)

ale některé *fragmenty* jsou v P, efektivně řešitelné, např. 2-SAT a Horn-SAT (viz Sekce 3.2 a 3.3)

praktický problém: moderní *SAT solvery* (viz Sekce 3.1) se používají v řadě odvětví aplikované informatiky, poradí si s obrovskými instancemi

3.1 SAT solvery

SAT solvery

- existují od 60. let 20. století, v 21. století dramatický rozvoj dnes až 10^8 proměnných, viz www.satcompetition.org.
- nejčastěji založeny na jednoduchém algoritmu DPLL (viz Sekce 3.4), umí i najít řešení (model)
- řada technologií pro efektivnější řešení instancí pocházejících z různých aplikačních domén, heuristiky pro řízení prohledávání (za použití ML, NN) — desítky tisíc řádků kódu

Praktická ukázka: boardomino

Lze pokrýt šachovnici s chybějícími dvěma protilehlými rohy perfektně pokrýt kostkami domina?

těžká instance SATu (proč?), jak zakódovat?

řešič Glucose, formát vstupu: DIMACS CNF

3.2 2-SAT a implikační graf

2-SAT vs. 3-SAT

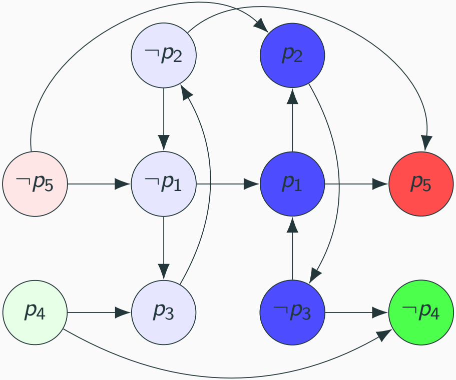
- *k*-CNF: CNF a každá klauzule nejvýše k literálů
- *k*-SAT: je daný k-CNF výrok splnitelný?
- k-SAT je NP-úplný pro $k \geq 3$ (ke každému výroku lze sestrojit ekvisplnitelný 3-CNF výrok)
- ale 2-SAT je v P, dokonce řešitelný v lineárním čase
- algoritmus využívá tzv. implikační graf:
  - 2-klauzule $p \vee q$ je ekvivalentní $\neg p \rightarrow q$ a také $\neg q \rightarrow p$
  - $p \sim p \vee p$ je ekvivalentní $\neg p \rightarrow p$
  - vrcholy jsou literály
  - hrany dané implikacemi
  - **myšlenka**: ohodnotíme-li vrchol 1, všude kam se dostaneme po hranách (komponenta silné souvislosti) musí být také 1

Implikační graf

$$V(\mathcal{G}_\varphi) = \{p, \neg p \mid p \in \mathrm{Var}(\varphi)\},$$
$$E(\mathcal{G}_\varphi) = \{(\overline{\ell_1}, \ell_2), (\overline{\ell_2}, \ell_1) \mid \ell_1 \vee \ell_2 \text{ je klauzule } \varphi\} \cup$$
$$\{(\bar{\ell}, \ell) \mid \ell \text{ je jednotková klauzule } \varphi\}$$

$(\neg p_1 \vee p_2) \wedge (\neg p_2 \vee \neg p_3) \wedge (p_1 \vee p_3) \wedge (p_3 \vee \neg p_4) \wedge (\neg p_1 \vee p_5) \wedge (p_2 \vee p_5) \wedge p_1 \wedge \neg p_4$

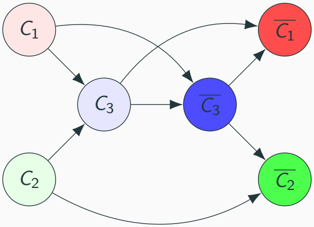
- najdeme komponenty silné souvislosti
- literály v komponentě musí být ohodnoceny stejně (jinak "$1 \to 0$")
- pokud má nějaká komponenta opačné literály, je φ nesplnitelný
- jinak sestrojíme model



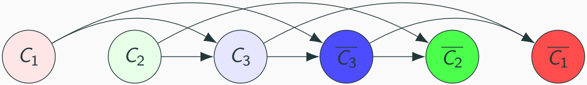
Konstrukce modelu

Všimněte si: stačí, aby z žádné komponenty ohodnocené 1 nevedla hrana do komponenty ohodnocené 0

provedeme kontrakci komponent, výsledný graf $\mathcal{G}_\varphi^*$ je acyklický



najdeme nějaké topologické uspořádání; v něm najdeme nejlevější dosud neohodnocenou komponentu, ohodnotíme ji 0, opačnou komponentu ohodnotíme 1, a opakujeme



0 0 0 1 1 1

Shrnutí

Tvrzení: φ je splnitelný, právě když žádná silně souvislá komponenta v $\mathcal{G}_\varphi$ neobsahuje dvojici opačných literálů.

Důkaz: $\Rightarrow$ literály v komponentě musí být ohodnoceny stejně

$\Leftarrow$ ohodnocení zkonstruované výše je model φ:

- jednotková klauzule ℓ platí kvůli hraně $\overline{\ell} \to \ell$, komponenta s $\overline{\ell}$ byla ohodnocena dříve, a to 0, takže $v(\ell) = 1$
- podobně pro 2-klauzuli $\ell_1 \vee \ell_2$, máme hrany $\overline{\ell_1} \to \ell_2$, $\overline{\ell_2} \to \ell_1$ pokud jsme ℓ_1 ohodnotili dříve než ℓ_2, museli jsme jako první narazit na komponentu s $\overline{\ell_1}$ a ohodnotit ji 0, tedy ℓ_1 platí; v opačném případě symetricky platí ℓ_2 $\square$

Důsledek: 2-SAT je řešitelný v lineárním čase, včetně konstrukce modelu (pokud existuje).

Důkaz: Komponenty silné souvislosti i topologické uspořádání najdeme v čase $\mathcal{O}(|V| + |E|)$, stačí je projít jednou $\square$

3.3 Horn-SAT a jednotková propagace

Horn-SAT

- hornovská klauzule: *nejvýše jeden *pozitivní* literál*

$$\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_n \vee q \ \sim\ (p_1 \wedge p_2 \wedge \cdots \wedge p_n) \to q$$

  základ logického programování (Prolog `q:-p1,p2,...,pn.`)
- Horn-SAT, tj. splnitelnost hornovského výroku (konjunkce hornovských klauzulí) je opět v P, v lineárním čase
- algoritmus využívá tzv. jednotkovou propagaci:
  - jednotková klauzule vynucuje hodnotu výrokové proměnné
  - tím můžeme výrok zjednodušit, např. pro $\neg p$ ($p = 0$): odstraníme klauzule s literálem $\neg p$, už jsou splněné odstraníme literál p (nemůže být splněný)
  - žádná jednotková klauzule $\Rightarrow$ každá klauzule má aspoň jeden negativní literál $\Rightarrow$ vše nastavíme na 0

Jednotková propagace

$\varphi = (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_2 \vee \neg p_3) \wedge (\neg p_5 \vee \neg p_4) \wedge p_4$

- nastav $v(p_4) = 1$, odstraň klauzule obsahující literál p_4, z ostatních klauzulí odstraň $\neg p_4$

$$\varphi^{p_4} = (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_2 \vee \neg p_3) \wedge \neg p_5$$

- nastav $v(p_5) = 0$, proveď jednotkovou propagaci $\neg p_5$

$$(\varphi^{p_4})^{\neg p_5} = (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_2 \vee \neg p_3)$$

- už žádná jednotková klauzule, v každé klauzuli alespoň dva literály ale nejvýše jeden pozitivní, tj. alespoň jeden negativní: $v(p_1) = v(p_2) = v(p_3) = 0$, model $v = (0, 0, 0, 1, 0)$

$$\varphi^{\ell} = \{C \setminus \{\overline{\ell}\} \mid C \in \varphi, \ell \notin C\} \qquad \text{(množinový zápis)}$$

Pozorování: φ^{ℓ} neobsahuje ℓ ani $\overline{\ell}$, modely = modely φ splňující ℓ

$\psi = p \wedge (\neg p \vee q) \wedge (\neg q \vee r) \wedge \neg r$ je nesplnitelný, co se stane?

Algoritmus pro Horn-SAT

vstup: výrok φ v Hornově tvaru,

výstup: model φ nebo informace, že φ není splnitelný

1. Pokud φ obsahuje dvojici opačných jednotkových klauzulí $\ell, \overline{\ell}$, není splnitelný.
2. Pokud φ neobsahuje žádnou jednotkovou klauzuli, je splnitelný, ohodnoť všechny zbývající proměnné 0.
3. Pokud φ obsahuje jednotkovou klauzuli ℓ, ohodnoť literál ℓ hodnotou 1, proveď jednotkovou propagaci, nahraď φ výrokem φ^ℓ, a vrať se na začátek.

Tvrzení: Algoritmus je korektní.

Důsledek: Horn-SAT lze řešit v lineárním čase.

Důkaz: Korektnost plyne z pozorování a z diskuze. V každém kroku stačí projít, výrok zkrátíme (kvadratický horní odhad, ale při vhodné implementaci lineární) $\square$

3.4 DPLL algoritmus pro řešení problému SAT

Algoritmus DPLL (Davis-Putnam-Logemann-Loveland, 1961)

myšlenka: čistý výskyt p buď jen v pozitivních nebo jen v negativních literálech $\Rightarrow$ lze mu nastavit příslušnou hodnotu!

DPLL = jednotková propagace + čistý výskyt + větvení (rekurze)

vstup: výrok φ v CNF,

výstup: model φ nebo informace, že φ není splnitelný

1. Dokud φ obsahuje jednotkovou klauzuli ℓ, ohodnoť literál ℓ hodnotou 1, proveď jednotkovou propagaci, nahraď φ výrokem φ^{ℓ}.
2. Dokud existuje literál ℓ, který má ve φ čistý výskyt, ohodnoť ℓ hodnotou 1, a odstraň klauzule obsahující ℓ.
3. Pokud φ neobsahuje žádnou klauzuli, je splnitelný.
4. Pokud φ obsahuje prázdnou klauzuli, není splnitelný.
5. Jinak zvol dosud neohodnocenou výrokovou proměnnou p, a zavolej algoritmus rekurzivně na $\varphi \wedge p$ a na $\varphi \wedge \neg p$.

Ukázkový běh

$$(\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee \neg s) \wedge (p \vee \neg r \vee \neg s) \wedge$$
$$(q \vee \neg r \vee s) \wedge (p \vee s) \wedge (p \vee \neg s) \wedge (q \vee s)$$

žádná jednotková klauzule, $\neg r$ má čistý výskyt: nastav $v(r) = 0$ a odstraň klauzule obsahující $\neg r$:

$$(\neg p \vee \neg q \vee \neg s) \wedge (p \vee s) \wedge (p \vee \neg s) \wedge (q \vee s)$$

už žádný čistý výskyt, rekurzivně zavolej na:

1. $(\neg p \vee \neg q \vee \neg s) \wedge (p \vee s) \wedge (p \vee \neg s) \wedge (q \vee s) \wedge p$
2. $(\neg p \vee \neg q \vee \neg s) \wedge (p \vee s) \wedge (p \vee \neg s) \wedge (q \vee s) \wedge \neg p$

a pokračuj dále v obou větvích výpočtu

$$\vdots$$

$$\mathrm{M}_\varphi = \{(1, a, 0, b, c) \mid a, b, c \in \{0, 1\}\}$$