Elemente de logică matematică

9 noiembrie 2004

- Calcul propozițional
- Calculul predicatelor
- Proceduri de decizie pt. realizabilitate
- Demonstrare de teoreme prin rezoluție

Logica propoziţională. Sintaxă

Simbolurile logicii propoziţionale: propoziţii atomice $p, q, r, \cdots$
conectorii $\neg$ şi $\rightarrow$, şi parantezele ( ).

Formulele logicii propoziţionale:
– orice propoziţie atomică este o formulă
– dacă α este o formulă, atunci $(\neg\alpha)$ este o formulă.
– dacă α şi β sunt formule, atunci $\alpha \rightarrow \beta$ este o formulă.

Operatorii cunoscuţi pot fi introduşi ca şi abrevieri:
– $(\alpha \wedge \beta) \stackrel{\text{def}}{=} (\neg(\alpha \rightarrow (\neg\beta)))$
– $(\alpha \vee \beta) \stackrel{\text{def}}{=} ((\neg\alpha) \rightarrow \beta)$
– $(\alpha \leftrightarrow \beta) \stackrel{\text{def}}{=} ((\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha))$

Notaţie simplificată: fără paranteze redundante;
precedenţa: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$; asociativitate la dreapta pt. $\rightarrow$.

Funcții de adevăr (evaluări)

O funcție de adevăr v: definită pentru toate formulele propoziționale, cu valori în $\{\mathsf{T}, \mathsf{F}\}$ astfel incât:

– $v(p)$ e definită pentru fiecare propoziție atomică p.

– $v(\neg\alpha) = \begin{cases} \mathsf{T} & \text{dacă } v(\alpha) = \mathsf{F} \\ \mathsf{F} & \text{dacă } v(\alpha) = \mathsf{T} \end{cases}$

– $v(\alpha \to \beta) = \begin{cases} \mathsf{F} & \text{dacă } v(\alpha) = \mathsf{T} \text{și } v(\beta) = \mathsf{F} \\ \mathsf{T} & \text{în caz contrar} \end{cases}$

interpretare = o evaluare pentru propozițiile atomice ale unei formule
O intrepretare *satisface* o formulă dacă aceasta e evaluată la T
(se spune că interpretarea e un *model* pentru formula respectivă).

formulă *validă* (*tautologie*): adevărată în toate interpretările
formulă *realizabilă* (satisfiable): adevărată în cel puțin o interpretare
formulă nerealizabilă (*contradicție*): falsă în orice interpretare

Abordare semantică și sintactică

Abordare *semantică*, bazată pe *implicația logică* (adevărul logic)

$$H \models \varphi$$

O mulțime de formule H *implică* o formulă φ dacă orice funcție de adevăr care satisface H (adică toate formulele din H) satisface pe φ.

Abordare *sintactică*: *demonstrația logică*
– bazată pe manipularea sintactică a formulelor:

Este o teoremă demonstrabilă dintr-un set de axiome, pe baza unor reguli de deducție ?

Axiome și reguli de deducție

Scheme de axiome pentru logica propozițională:

A1: $(\alpha \rightarrow (\beta \rightarrow \alpha))$

A2: $((\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma)))$

A3: $(((\neg\beta) \rightarrow (\neg\alpha)) \rightarrow (((\neg\beta) \rightarrow \alpha) \rightarrow \beta))$

(denumite *scheme* pentru că axiomele se obțin particularizând cu formule individuale din logica propozițională)

Introducem o singură regulă de deducție (*modus ponens, MP*):
Din formulele φ și $\varphi \rightarrow \psi$ putem deduce ψ.

Deducție

Fie H o mulțime de formule. Se numește *deducție* din H un șir de formule $A_1, A_2, \cdots, A_n$, astfel ca:

1. A_i este o axiomă, sau
2. A_i este o formulă din H, sau
3. A_i rezultă prin MP din doi membri anteriori A_j, A_k cu $j < i$, $k < i$.

Spunem că A_n rezultă din H (e deductibil, e consecință): $H \vdash A_n$

Exemplu: demonstrăm că $(\varphi \to \varphi)$

(1) $\varphi \to ((\varphi \to \varphi) \to \varphi))$ **A1**

(2) $\varphi \to ((\varphi \to \varphi) \to \varphi)) \to ((\varphi \to (\varphi \to \varphi) \to (\varphi \to \varphi))$ **A2**

(3) $(\varphi \to (\varphi \to \varphi) \to (\varphi \to \varphi)$ MP(1,2)

(4) $\varphi \to (\varphi \to \varphi)$ **A1**

(5) $\varphi \to \varphi$ MP(3,4)

Teorema deducției

Fie H o mulțime de formule și α, β două formule.
Atunci $H \vdash \alpha \to \beta$ dacă și numai dacă $H \cup \{\alpha\} \vdash \beta$.
- utilizată ca regulă de inferență suplimentară, simplifică demonstrațiile

Alte corolarii:
- dacă $H \vdash \alpha$ și $H \vdash \alpha \to \beta$, atunci $H \vdash \beta$
- dacă $G \subseteq H$ și $G \vdash \alpha$, atunci $H \vdash \alpha$
- dacă $H \vdash G$ și $G \vdash \alpha$, atunci $H \vdash \alpha$

Consistență și completitudine

Noțiuni care stabilesc corespondența între abordarea sintactică, bazată pe deducție, și cea semantică, bazată pe valoarea de adevăr.

Consistență: Dacă H e o mulțime de formule, și α este o formulă astfel ca $H \vdash \alpha$, atunci $H \models \alpha$.
(Orice teoremă în logica propozițională este o tautologie).

Completitudine: Dacă H e o mulțime de formule, și α este o formulă astfel ca $H \models \alpha$, atunci $H \vdash \alpha$. (Orice tautologie este o teoremă).

Demonstrația: bazată pe următoarele noțiuni și rezultate auxiliare:

O mulțime de formule H este *inconsistentă* dacă există o formulă α astfel încât $H \vdash \alpha$ și $H \vdash \neg\alpha$.

Orice mulțime consistentă de formule poate fi extinsă la o mulțime *maximal consistentă* (adăugarea oricărei formule o face inconsistentă).

O mulțime de formule este *consistentă* dacă și numai dacă e *realizabilă*.

Limbaje de ordinul I

Simbolurile unui limbaj de ordinul I sunt:
- parantezele ( )
- conectorii $\neg$ și $\rightarrow$
- cuantificatorul $\forall$ (universal)
- o mulțime de identificatori $v_0, v_1, \cdots$ pentru *variabile*
- o mulțime (posibil vidă) de simboluri pentru *constante*
- pt. orice $n \geq 1$ o mulțime de simboluri de *funcții* n-are
- pt. orice $n \geq 1$ o mulțime de simboluri de *predicate* (relații) n-are

Limbajele de ordinul I cu egalitate: conțin și $=$ ca simbol special pe lângă cele de mai sus.

Termeni și formule de ordinul I

Termenii unui limbaj de ordinul I (definiți structural recursiv)
- orice simbol de variabilă v_n
- orice simbol de constantă c
- $f(t_1, \cdots, t_n)$, dacă f e un simbol de funcție n-ară și $t_1, \cdots, t_n$ sunt termeni

Formulele (bine formate) (well-formed formulas) unui limbaj de ord. I:
- $P(t_1, \cdots, t_n)$, unde P e un predicat n-ar și $t_1, \cdots, t_n$ sunt termeni
- $t_1 = t_2$, unde t_1 și t_2 sunt termeni (pt. limbaje cu egalitate)
- $\neg\alpha$, unde α este o formulă
- $\alpha \rightarrow \beta$, unde α, β sunt formule
- $\forall v_n \varphi$ unde v_n e o variabilă și φ e o formulă

Interpretări și evaluări

O *interpretare* (*structură*) I pt. limbajul de predicate $\mathcal{L}$ constă din:
- o mulțime nevidă U numită *universul* sau *domeniul* lui I (mulțimea valorilor pe care le pot lua variabilele)
- pentru orice simbol de constantă c, o valoare $c_I \in U$
- pentru orice simbol de funcție n-ară f, o funcție $f : U^n \to U$
- pentru orice simbol de predicat n-ar P, o submulțime $P_I \subseteq U^n$.

Fie I o interpretare cu univers U pentru $\mathcal{L}$, și fie V mulțimea tuturor simbolurilor de variabile din $\mathcal{L}$. O *evaluare* este o funcție $s : V \to U$.
Extinzând evaluarea s la termeni și formule obținem o funcție (valoare) de adevăr pentru formulele din $\mathcal{L}$. Notăm $I \models s(\varphi)$ sau $I \models \varphi[s]$.
Definim: $I \models s(\forall x\varphi)$ dacă $I \models s_{x\leftarrow d}(\varphi)$ pentru orice $d \in U$, unde $s_{x\leftarrow d}$ este atribuirea $s_{x\leftarrow d}(v) = \begin{cases} d & \text{dacă variabila } v \text{ este } x \\ \text{s(v)} & \text{pentru orice altă variabilă } v \end{cases}$

Notăm $I \models \varphi$ (I e un *model* pt. φ) dacă $I \models s(\varphi)$ pt. orice evaluare s.

Axiomele calculului predicatelor

Definim: variabila x se poate *substitui* cu termenul t în $\forall y \varphi$ dacă:
- x nu apare liber în φ sau
- y nu apare în t și x se poate substitui cu t în φ

A1: $(\alpha \rightarrow (\beta \rightarrow \alpha))$

A2: $((\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma)))$

A3: $(((\neg\beta) \rightarrow (\neg\alpha)) \rightarrow (((\neg\beta) \rightarrow \alpha) \rightarrow \beta))$

A4: $(\forall x(\alpha \rightarrow \beta) \rightarrow (\forall x\alpha \rightarrow \forall x\beta))$

A5: $(\forall x\alpha \rightarrow \alpha[x \leftarrow t])$, dacă x poate fi substituit cu t în α

A6: $(\alpha \rightarrow \forall x\alpha)$ dacă x nu apare liber în α

Pentru egalitate, adăugăm și

A7: $x = x$

A8: $x = y \rightarrow \alpha = \beta$

unde β se obține din α înlocuind oricâte din aparițiile lui x cu y.

Consistență și completitudine

Fie H o mulțime de formule și φ o formulă. Spunem că H implică φ ($H \models \varphi$) dacă pentru orice interpretare I, $I \models H$ implică $I \models \varphi$.

Calculul predicatelor de ordinul I este consistent și complet (la fel ca și logica propozițională):

Pentru orice mulțime de ipoteze H, și orice formulă φ, $H \vdash \varphi$ dacă și numai dacă $H \models \varphi$.

Obs: Noțiunea de completitudine de mai sus este diferită de cea de a stabili dacă din axiome se poate deduce orice formulă (sau negația ei).

Întrebarea dacă $H \vdash \varphi$ este în general nedecidabilă.

Realizabilitate. Aplicaţii

Problema: Să se determine dacă o formulă propoziţională e realizabilă.

Contextul: în general, formule complexe, de sute sau mii de variabile.

Problema apare:
- în determinarea echivalenţei a două circuite sau modele
- ca pas elementar în demonstrarea de teoreme
- folosire în loc de BDD-uri pentru model checking simbolic

Reprezentarea: formă canonică (normală) conjunctivă

Proceduri de decizie performante: bazate pe algoritmul Davis-Putnam.

Noţiuni: clauză unitate: formată dintr-un singur literal
literal pur: apare numai pozitiv (similar: numai negat)

Algoritmul Davis-Putnam

function Satisfiable (listă de clauze S)
repeat
    for fiecare clauză unitate sau literal pur L din S **do**
        elimină toate clauzele ce conțin L
        elimină $\neg$ L din toate clauzele
    if S e vidă **return** TRUE
    elsif S conține clauza nulă **return** FALSE
until nu mai apar modificări
alege un literal L din S pt. descompunere (adevărat/fals)
if Satisfiable (S $\cup$ $\{L\}$) **return** TRUE
elsif Satisfiable (S $\cup$ $\{\neg L\}$) **return** TRUE
else return FALSE

Demonstratoare de teoreme

O mare varietate:
- pentru demonstrarea de rezultate din matematică
- pentru verificarea de sisteme (în special programe)

În general, realizate pentru logici de ordin superior
- admit tipuri descrise prin intermediul predicatelor
- au capabilități de inducție

- prin înlănțuire înainte (derivă teoreme apropiindu-se de scop)
- sau înapoi (generează concluzii intermediare pentru scopul dat)
- aplicarea regulilor de inferența: controlată prin *tactici*

Metoda rezoluţiei. Formă clauzală

Orice formulă fără variabile libere din calculul predicatelor poate fi scrisă în formă clauzală trecând printr-o serie de 8 paşi simpli.

Exemplu: Pornim de la
$\forall x[\neg P(x) \rightarrow \exists y(D(x,y) \wedge \neg(E(f(x),y) \vee E(x,y))] \wedge \neg\forall x P(x)$

(1) Eliminarea tuturor conectorilor în afară de $\wedge$, $\vee$, $\neg$:
$\forall x[\neg\neg P(x) \vee \exists y(D(x,y) \wedge \neg(E(f(x),y) \vee E(x,y)))] \wedge \neg\forall x P(x)$

(2) Translatarea negaţiilor înăuntru, până la predicate:
$\forall x[P(x) \vee \exists y(D(x,y) \wedge \neg E(f(x),y) \wedge \neg E(x,y))] \wedge \exists x \neg P(x)$

(3) Redenumirea variabilelor, cu nume unic pt. orice cuantificator
$\forall x[P(x) \vee \exists y(D(x,y) \wedge \neg E(f(x),y) \wedge \neg E(x,y))] \wedge \exists z \neg P(z)$

Forma clauzală (cont.)

(4) Eliminarea cuantificatorilor existențiali (skolemizare)
Pentru $\exists y$ situat în interiorul unui cuantificator $\forall x$, se creează o *funcție* Skolem $y = g(x)$ (valoarea lui y depinde în general de cea luată de x).
Altfel, se alege o nouă *constantă* Skolem.
$\forall x[P(x) \vee (D(x, g(x)) \wedge \neg E(f(x), g(x)) \wedge \neg E(x, g(x)))] \wedge \neg P(a)$

(5) Se aduce la forma normală prenex (toți cuantificatorii $\forall$ în față):
$\forall x([P(x) \vee (D(x, g(x)) \wedge \neg E(f(x), g(x)) \wedge \neg E(x, g(x)))] \wedge \neg P(a))$

(6) Se elimină prefixul cu cuantificatorii universali
$[P(x) \vee (D(x, g(x)) \wedge \neg E(f(x), g(x)) \wedge \neg E(x, g(x)))] \wedge \neg P(a)$

(7) Se convertește la forma normală conjunctivă
$(P(x) \vee D(x, g(x))) \wedge (P(x) \vee \neg E(f(x), g(x))) \wedge (P(x) \vee \neg E(x, g(x))) \wedge \neg P(a)$
(8) Se elimină $\wedge$ și se scriu disjuncții ca și clauze separate

Principiul rezoluției

Fie două clauze, scrise ca și mulțimi de termeni disjunctivi.
Considerăm întâi cazul formulelor propoziționale.
Numim *rezolvent* a două clauze C_1, C_2 în raport cu literalul l (pentru care $l \in C_1, (\neg l) \in C_2$): $\text{rez}_l(C_1, C_2) = (C_1 \setminus \{l\}) \cup (C_2 \setminus \{\neg l\})$.
Exemplu: $\text{rez}_p(\{p, q, r\}, \{\neg p, s\}) = \{q, r, s\}$.
$(p \vee q \vee r) \wedge (\neg p \vee s) \rightarrow (q \vee r \vee s)$
Propoziție: $C_1, C_2 \models \text{rez}_l(C_1, C_2)$.
Corolar: $C_1 \wedge C_2$ e realizabilă dacă ș. n. dacă $\text{rez}_l(C_1, C_2)$ e realizabilă.

Determinăm realizabilitatea unei formule în formă normală conjunctivă repetând adăugarea de rezolvenți, si verificând dacă se poate deduce clauza vidă.

Unificarea termenilor

Pentru calculul predicatelor, procedăm la fel; în loc de un literal l și negația lui, $\neg l$ considerăm însă negația $\neg l'$ a unul literal l' are să poată fi *unificat* cu el.

Doi literali se pot unifica dacă există o substituție de termeni pentru variabilele care apar, astfel încât literalii să devină identici.

Exemplu: $P(a, x, y)$ și $P(z, f(z), b)$ se pot unifica în $P(a, f(a), b)$.

Pentru a unifica doi literali: se unifică succesiv termenii de pe aceeași poziție a argumentelor (pt. funcții și predicate) până când se ajunge la același literal, sau unificarea devine imposibilă (se ajunge la simboluri de funcții diferite, sau la unificarea lui x cu un termen ce conține pe x)