Demonstrații

Metode de demonstrare în limbaje de programare

Traian Florin Șerbănuță

Departamentul de Informatică, FMI, UNIBUC
traian.serbanuta@fmi.unibuc.ro

4 noiembrie 2014

Ce este o expresie/program IMP?

Am zis că mulțimea expresiilor IMP este definită de următoarea sintaxă:

$$
\begin{aligned}
e ::= {} & n \mid b \mid e\ op\ e \mid \texttt{if}\ e\ \texttt{then}\ e\ \texttt{else}\ e \\
& \mid\ !\,l \mid l := e \\
& \mid \texttt{skip} \mid e\ ;\ e \mid \texttt{while}\ e\ \texttt{do}\ e\ \texttt{done}
\end{aligned}
$$

Cum interpretăm/cum reprezentăm ca obiect matematic expresia:

$$\texttt{if}\ 0 \leq\ !l\ \texttt{then}\ \texttt{skip}\ \texttt{else}\ (\texttt{skip}\ ;\ l := 0)$$

Cum reprezentăm ca obiect matematic programul

$\mathtt{if}\ 0 <= !l\ \mathtt{then}\ \mathtt{skip}\ \mathtt{else}\ (\mathtt{skip}\ ;\ l := 0)$

Cum reprezentăm ca obiect matematic programul

$\texttt{if}\ 0 \mathrel{<=} !l\ \texttt{then skip else}\ (\texttt{skip}\ ;\ l := 0)$

Ca o listă de caractere

['i', 'f', ' ', '0', '<', '=', '!', 'l', ' ', 't', 'h', 'e', 'n', ' ', 's', 'k', 'i', 'p', ' ', 'e', 'l', . . .]

Cum reprezentăm ca obiect matematic programul

`if` 0 <= !*l* `then skip else` (`skip` ; *l* := 0)

Ca o listă de caractere

['i', 'f', ' ', '0', '<', '=', '!', 'l', ' ', 't', 'h', 'e', 'n', ' ', 's', 'k', 'i', 'p', ' ', 'e', 'l', . . .]

Ca o listă de simboluri (tokens)

[IF, INT(0), LTE, DEREF, LOC("l"), THEN, SKIP, ELSE, LPAREN, . . .]

Cum reprezentăm ca obiect matematic programul

`if 0 <= !l then skip else (skip ; l := 0)`

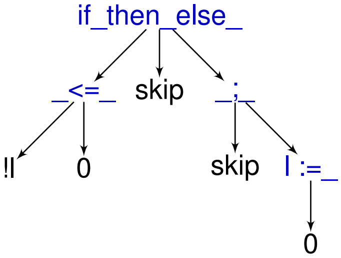
Ca o listă de caractere

['i', 'f', ' ', '0', '<', '=', '!', 'l', ' ', 't', 'h', 'e', 'n', ' ', 's', 'k', 'i', 'p', ' ', 'e', 'l', . . .]

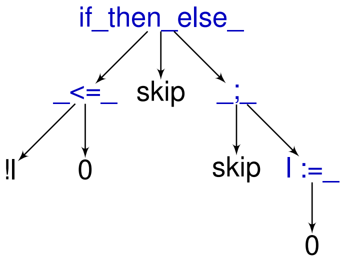
Ca o listă de simboluri (tokens)

[IF, INT(0), LTE, DEREF, LOC("l"), THEN, SKIP, ELSE, LPAREN, . . .]

Ca un arbore de sintaxă (abstract)



Arbore abstract de sintaxă

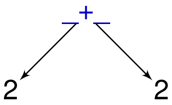


- Pune în evidența structura: fiecare subarbore corespunde unei expresii
- Ignoră detaliile „gramaticale"
- Fiecare nod corespunde unei producții a lui e din gramatica dată

Arbori abstracți de sintaxă

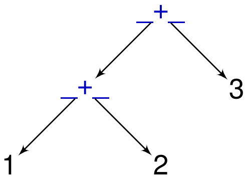
Observații

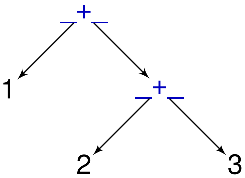
$2 + 2 \neq 4$



$\neq$ 4

$1 + 2 + 3$ e ambiguă și $1 + (2 + 3) \neq (1 + 2) + 3$





Parantezele nu fac parte din gramatică — au doar rol de dezambiguizare

$$1 + 2 = (1 + 2) = (1) + (2) = (((1) + ((((2))))))$$

Tip abstract de date pentru sintaxă

Reprezentare în OCaml

```
type l = string
type op = Plus | Mic                    (* op ::= + | <=              *)
type e =                                (* e ::=                      *)
(*** frunze / noduri terminale ***)
  | Int of int                          (*      n                     *)
  | Bool of bool                        (*    | b                     *)
  | Skip                                (*    | skip                  *)
  | Loc of l                            (*    | ! l                   *)
(*** Noduri unare ***)
  | Atrib of l * e                      (*    | l := e                *)
(*** Noduri binare ***)
  | Op of e * op * e                    (*    | e op e                *)
  | Secv of e * e                       (*    | e ; e                 *)
  | While of e * e                      (*    | while e do e          *)
(*** Noduri ternare ***)
  | If of e * e * e                     (*    | if e then e else e    *)
```

Reprezentarea AST ca termen

Program

```
( x := 10) ;
((sum := 0) ;
 while (1 <= !x)
 do
    (sum := (!sum + !x)) ;
    (x := (!x + −1))
 done
 )
```

Arbore de sintaxă (Reprezentare în OCaml)

```
Seq(Atrib("x", Int (10)),
Seq(Atrib("sum",Int (0)),
     While(Op(Int(0), Mic, Loc("x")),
       Seq(Atrib("sum", Op(Loc("sum"), Plus, Loc("x"))),
           Atrib ("x", Op(Loc(x), Plus, Int (−1)))
       ))
))
```

- Pentru a ușura notația lucrăm cu reprezentarea sintactică (stânga)
- Folosim paranteze pentru dezambiguizare
- Asociem (mental) fiecărei fraze program termenul AST corespunzător
- Identificăm termenul AST cu arborele asociat

Definiții recursive

Exemple

Numerele naturale (Peano)

- $0 \in \mathbb{N}$
- Dacă $n \in \mathbb{N}$ atunci și $s(n) \in \mathbb{N}$

Limbajul descris de o gramatică $\quad e ::= n \mid\ !\ l \mid e\ op\ e$

- Orice număr întreg n este expresie
- Pentru orice locație l, $!\ l$ este expresie
- Pentru orice simbol de operație întreagă o
  Dacă e_1 și e_2 sunt expresii, atunci și $e_1\ o\ e_2$ este expresie

- Arbori de parsare/arbori AST
- Termeni AST descriși de un tip abstract de date
- Termeni descriși de o signatură algebrică

Reguli de deducție

Definiție

O regulă de deducție r peste o mulțime T este o pereche (H, c) unde

- $H = \{h_1, \ldots, h_n\} \subseteq T$ este mulțimea (finită) de ipoteze ale regulii
- $c \in T$ este concluzia regulii
- Prezentare arborescentă: $\dfrac{h_1 \quad h_2 \quad \cdots \quad h_n}{c}$

Dacă mulțimea H a ipotezelor e vidă, r se numește axiomă.

Exemplu (Mulțimea numerelor Peano)

- Fie $T = \{0, s, (,)\}^\star$ limbajul tuturor cuvintelor peste alfabetul $\{0, s, (,)\}$
- Definim recursiv o submulțime a lui T prin regulile de deducție $\mathcal{R}$:

$$\mathcal{R} = \{(\emptyset, 0)\} \cup \{(\{t\}, s(t)) \mid t \in T\}$$

Scheme de reguli de deducție

Scop: Prezentarea compactă a unui sistem de reguli de deducție

Definiție: O schemă de reguli de deducție este de forma:

$$(\text{ETICHETĂ}) \quad \frac{h_1 \cdots h_n}{c} \quad \textit{dacă } \textbf{condiții}$$

unde:

- $c, h_1, h_2, \ldots, h_n$ sunt termeni cu (meta)variabile
- **condiții** sunt constrângeri asupra variabilelor

Regulile de deducție asociate schemei sunt instanțe ale acesteia

- Obținute prin substituirea metavariabilelor cu valori concrete
- Astfel încât constrângerile sunt satisfăcute

Exemplu (expresii aritmetice în IMP):

Axiome: $(\text{INT}) \quad n \quad \textit{dacă } n$ intreg $\qquad (\text{LOC}) \quad !\, l \quad \textit{dacă } l$ locație

Regulă: $$(\text{OP}) \quad \frac{e_1 \quad e_2}{e_1 \ o \ e_2} \quad \textit{dacă } o \in \{+, \leq\}$$

Exemplu

Semantica tranzițională

Semantica tranzițională este dată de un sistem de scheme de reguli

- Care definesc recursiv relația de tranziție într-un pas $\longrightarrow$
- Ca submulțime a lui *Config* $\times$ *Config*
- Exemplu de schemă de reguli:

$$(\textsc{OpS}) \quad \frac{\langle e_1, s\rangle \to \langle e_1', s'\rangle}{\langle e_1 \ o \ e_2, s\rangle \to \langle e_1' \ o \ e_2, s'\rangle}$$

Demonstrații deductive

Prezentare ca arbori

Definiție

Un arbore de demonstrație folosind regullile $\mathcal{R}$ peste mulțimea T satisface:

- Orice nod este etichetat cu un element din T
- Daca multimea etichetelor copiilor unui nod este H și eticheta nodului este c, atunci $(H, c) \in \mathcal{R}$
- Înălțimea arborelui este finită

Un arbore de demonstrație e o demonstrație pentru eticheta rădăcinii lui.

Exemplu (expresii aritmetice)

$$(\text{OP})\ \dfrac{(\text{OP})\ \dfrac{(\text{INT})\ \dfrac{\checkmark}{3} \qquad (\text{LOC})\ \dfrac{\checkmark}{!x}}{3+!x} \qquad (\text{INT})\ \dfrac{\checkmark}{5}}{(3+!x) \le 5}$$

Demonstrații deductive

Prezentare liniară

Definiție

O demonstrație a lui c folosind mulțimea de reguli $\mathcal{R}$ este o secvență de reguli $(H_1, c_1), \ldots (H_n, c_n)$ satisfacând următoarele

- $(H_i, c_i) \in \mathcal{R}$ pentru orice $1 \leq i \leq n$
- $c_n = c$
- Orice ipoteză $h \in H_i$ a fost deja demonstrată, i.e., există $c_j = h$ cu $j < i$

Exemplu

$(\emptyset, 3)$, $(\emptyset, 5)$, $(\emptyset, !x)$, $(\{3, !x\}, 3{+}!x)$, $(\{3{+}!x, 5\}, (3{+}!x) \leq 5)$.

Definiții recursive

Definiție formală

Definiție

Mulțimea definită recursiv de regulile $\mathcal{R}$ este submulțimea elementelor c ale lui T demonstrabile folosind $\mathcal{R}$.
Denumire alternativă: $Th(\mathcal{R})$ — mulțimea teoremelor lui $\mathcal{R}$

Teoremă

Mulțimea teoremelor lui $\mathcal{R}$ este cea mai mică mulțime A „închisă“ la regulile din $\mathcal{R}$, i.e., satisfăcând proprietatea următoare:

$$\textit{pentru orice regulă } (H, c) \in \mathcal{R}, \textit{ dacă } H \subseteq A \textit{ atunci } c \in A.$$

Mulțimea teoremelor

Demonstrație

Teoremă

Mulțimea teoremelor lui $\mathcal{R}$ este cea mai mică mulțime A „închisă“ la regulile din $\mathcal{R}$, i.e., satisfăcând proprietatea următoare:

$$\textit{pentru orice regulă } (H, c) \in \mathcal{R}, \textit{ dacă } H \subseteq A \textit{ atunci } c \in A.$$

Schiță de demonstrație — prezentarea liniară.

- $Th(\mathcal{R})$ închisă la $\mathcal{R}$
- $Th(\mathcal{R})$ cea mai mică mulțime închisă la $\mathcal{R}$

□

Mulțimea teoremelor

Demonstrație

Teoremă

Mulțimea teoremelor lui $\mathcal{R}$ este cea mai mică mulțime A „închisă“ la regulile din $\mathcal{R}$, i.e., satisfăcând proprietatea următoare:

$$\textit{pentru orice regulă } (H, c) \in \mathcal{R}, \textit{ dacă } H \subseteq A \textit{ atunci } c \in A.$$

Schiță de demonstrație — prezentarea liniară.

- $Th(\mathcal{R})$ închisă la $\mathcal{R}$
  - Fie $(H, c) \in \mathcal{R}$ astfel încât $H = \{h_1, h_2, \ldots, h_n\} \subseteq Th(\mathcal{R})$
  - Pentru fiecare $h_i \in H$, fie $D(h_i)$ o demonstrație pentru h_i
  - Atunci $D(h_1), D(h_2), \ldots, D(h_n), (H, c)$ e o demonstrație pentru c
- $Th(\mathcal{R})$ cea mai mică mulțime închisă la $\mathcal{R}$

□

Mulțimea teoremelor

Demonstrație

Teoremă

Mulțimea teoremelor lui $\mathcal{R}$ este cea mai mică mulțime A „închisă“ la regulile din $\mathcal{R}$, i.e., satisfăcând proprietatea următoare:

pentru orice regulă $(H, c) \in \mathcal{R}$, dacă $H \subseteq A$ atunci $c \in A$.

Schiță de demonstrație — prezentarea liniară.

- $Th(\mathcal{R})$ închisă la $\mathcal{R}$
  - Fie $(H, c) \in \mathcal{R}$ astfel încât $H = \{h_1, h_2, \ldots, h_n\} \subseteq Th(\mathcal{R})$
  - Pentru fiecare $h_i \in H$, fie $D(h_i)$ o demonstrație pentru h_i
  - Atunci $D(h_1), D(h_2), \ldots, D(h_n), (H, c)$ e o demonstrație pentru c
- $Th(\mathcal{R})$ cea mai mică mulțime închisă la $\mathcal{R}$
  - Fie A închisă la $\mathcal{R}$. Arătăm că $c \in Th(\mathcal{R}) \implies c \in A$
  - Inducție (totală) după lungimea demonstrației lui c

□

Cele două definiții ale demonstrabilității sunt echivalente

Teoremă

c e demonstrabilă folosind arbori de demonstrație dacă și numai dacă c e demonstrabilă folosind demonstrații liniare.

Idee de demonstrație (Temă).

Demonstrăm teorema precedentă folosind prezentarea arborescentă. $\square$

Inducție deductivă

Formulare matematică

Dacă $\mathcal{R}$ sistem de reguli de deducție peste T și P proprietate peste T,

$$(\forall(H,c)\in\mathcal{R}.((\forall h\in H.P(h))\implies P(c)))\implies\forall t\in\mathrm{Th}(\mathcal{R}).P(t)$$

Adică

Dacă mulțimea elementelor care satisfac proprietatea P este închisă la $\mathcal{R}$, atunci ea conține teoremele lui $\mathcal{R}$.

Inducție deductivă pentru mulțimi definite recursiv

Principiul inducției deductive

Pentru a demonstra că P este adevărată pentru orice element al unei mulțimi A definită recursiv de regulile din $\mathcal{R}$:

- Considerăm mulțimea elementelor din A pentru care P e adevărată
- Arătăm că este închisă la regulile din $\mathcal{R}$

Observații

- Instanțierea rezultatului precedent pentru $P'(c) = c \in A \wedge P(c)$
- Concret, pentru fiecare regulă (H, c), cu $H \subseteq A$, demonstrăm că
  - Dacă $P(h)$ e adevărată pentru orice ipoteză $h \in H$,
  - Atunci și $P(c)$ e adevărată.

Exemplu

Inducție pe numere naturale

- Fie $\mathbb{N}$ mulțimea definită de regulile:

$$\frac{}{0} \qquad\qquad \frac{n}{s(n)}$$

- Instanțiem principiul inducției deductive pentru acest sistem

$$(((\forall h \in \emptyset.P(h)) \implies P(0)) \wedge \forall n \in \mathbb{N}.((\forall h \in \{n\}.P(h)) \implies P(s(n)))) \implies \forall n \in \mathbb{N}.P(n)$$

- Prin simplificare se obține principiul inducției matematice

$$(P(0) \wedge (\forall n \in \mathbb{N}.P(n) \implies P(s(n)))) \implies \forall n \in \mathbb{N}.P(n)$$

Principiul inducției structurale

Inducție pe termeni definiți recursiv

Scop: Pentru a demonstra că P este adevărată pentru orice AST (termen) dat de o gramatică (tip abstract de date)

Metoda: Instanțiem principiul inducției deductive pentru sistemul de reguli care definește recursiv termenii

Exemplu — expresii aritmetice în IMP:

Axiome: $(\textsc{int})\quad n \quad \textit{dacă } n \text{ întreg} \qquad (\textsc{loc})\quad !\ l \quad \textit{dacă } l \text{ locație}$

Regulă: $(\textsc{op})\quad \dfrac{e_1 \quad e_2}{e_1\ o\ e_2} \quad \textit{dacă } o \in \{+, \leq\}$

Ca să arătăm că P ține pentru orice expresie aritmetică IMP

- pentru orice întreg n, demonstrăm $P(n)$
- pentru orice locație l, demonstrăm $P(!\ l)$
- Arătăm că dacă $P(e_1)$ și $P(e_2)$, atunci $P(e_1\ o\ e_2)$ pentru $o \in \{+, \leq\}$

Observație: Demonstrațiile se reduc la analize de caz

Cazuri de inducție structurală pentru IMP

Cazuri de bază. Demonstrăm

- $P(n)$ pentru orice $n \in \mathbb{Z}$,
- $P(\mathit{true})$ și $P(\mathit{false})$
- $P(\texttt{skip})$
- $P(\texttt{!}\ l)$ pentru orice $l \in \mathbb{L}$

Recursie simplă. Demonstrăm că dacă $P(e)$ atunci și

- $P(l\ \texttt{:=}\ e)$ pentru orice $l \in \mathbb{L}$

Recursie dublă. Demonstrăm că dacă $P(e_1)$ și $P(e_2)$, atunci și

- $P(e_1\ \texttt{o}\ e_2)$ pentru orice $o \in \{\leq, +\}$
- $P(e_1\ \texttt{;}\ e_2)$
- $P(\texttt{while}\ e_1\ \texttt{do}\ e_2\ \texttt{done})$

Recursie triplă. Demonstrăm că dacă $P(e_1)$, $P(e_2)$ și $P(e_3)$, atunci și

- $P(\texttt{if}\ e_1\ \texttt{then}\ e_2\ \texttt{else}\ e_3)$

Determinism puternic

Teoremă (Limbajul IMP este puternic determinist)

Dacă $\langle e, s\rangle \rightarrow \langle e_1, s_1\rangle$ *și* $\langle e, s\rangle \rightarrow \langle e_2, s_2\rangle$*, atunci* $e_1 = e_2$ *și* $s_1 = s_2$*.*

Determinism puternic

Teoremă (Limbajul IMP este puternic determinist)

Dacă $\langle e, s\rangle \to \langle e_1, s_1\rangle$ *și* $\langle e, s\rangle \to \langle e_2, s_2\rangle$, *atunci* $e_1 = e_2$ *și* $s_1 = s_2$.

Idee de demonstrație.

Fie P proprietatea definită de

$P(e) \stackrel{def}{=} \forall s, e_1, s_1, e_2, s_2.$

$$\langle e, s\rangle \to \langle e_1, s_1\rangle \wedge \langle e, s\rangle \to \langle e_2, s_2\rangle \implies \langle e_1, s_1\rangle = \langle e_2, s_2\rangle$$

Demonstrăm că P e adevărată pentru toate expresiile IMP prind inducție asupra structurii lui e. $\square$