2012年3月17日

TAPL-nagoya #3

@t6s

1　3章でやる事

- abstract syntax 〜 §3.2
- inductive definition 〜 §3.3
- inductive proof 〜 §3.3
- evaluation 〜 §3.5
- runtime error 〜 §3.5

§3.1 intro.

3　object lang. とmetalang.

p.24のBNFについて,

- このBNFで定義される言語がobject language
- ，ということの説明をしている日本語がmetalanguage.

4　object lang. と metalang.

メタ変数　**メタ言語に含まれる変数**

変数　**対象言語に含まれる変数**

例:

「項 t **の構造についての帰納法」**

「$(\lambda x.M)N \to_\beta M[x := N]$**」**

「任意の論理式 A **について**, $\vdash A$ iff $\vdash \forall x A$**」**

約束:

TAPL **で項を指すメタ変数は** r, s, t, u **など**.

5 object lang. と metalang.

おまけ　　　　メタ自然数と（対象言語の）自然数も違う.

例：「項0のsizeは1」

0は対象言語に含まれ, 1はメタ言語に含まれる.

TAPLではフォントを変えて区別しているはず.

6　自然数項の略記

対象言語の自然数について, $\mathrm{succ}(0)$ **を** 1, $\mathrm{succ}(\mathrm{succ}(0))$ **を** 2, **などと略記する**.

7 expression と term

expression というと,

term, type, kind, ...

などいろいろ含む.

8 abstract syntax

p.24 のBNFは, parse されたtreeが持つべき構造だけを述べている.
括弧なども含んだconcreteなsyntaxではない.

9　評価とvalue(くわしくは§3.5)

$$\texttt{term} \overset{\text{評価（実行）}}{\rightsquigarrow} \texttt{value}$$

$$iszero(1) \rightsquigarrow false$$

$$pred(succ(0)) \rightsquigarrow 0$$

$$succ(pred(0)) \rightsquigarrow 1$$

$$iszero(true) \rightsquigarrow ?$$

10　評価と value

iszero(true), succ(true), if 0 then 0 else 0

これらが型エラーとなるよう, 後の章で型を導入する.

§3.2 文法

12 Def. 3.2.1

- “smallest” **（最小性）は重要**
- T **の要素は** parse **済みの** tree **と思う**

13 Def. 3.2.2

- 3.2.1 **の言い替え。**
- “smallest” **も本当は略さず書いたほうがよい。**

14　inference ruleについて少し

- 横棒なしのものをaxiomと呼ぶことがある。
- メタ変数を含むものは本当はrule schemaと呼ぶ。

例：

$$\frac{t_1 \in T}{succ(t_1) \in T}$$ はschema. (t_1 がメタ)

これの t_1 を具体化した $$\frac{succ(0) \in T}{succ(succ(0)) \in T}$$ は、(concrete) rule

15 Def. 3.2.3

- T とは別の集合を定義している。
- 中身を具体的に書いているので、”smallest” と言わなくて良い。

“smallest” は、条件を満たす T_i に対しての $\bigcap_i T_i$

3.2.3の定義は $\bigcup_i S_i$

当日出た意見：定数true, false, 0などの選び方を自由にして同型類でTを定義すると最小性って言いにくそう

16 Exercise 3.2.4, 3.2.5

誰かやってね!

17　Prop. 3.2.6

$$T = S$$

まず$T \subset S$と$S \subset T$に分ける。

- $T \subset S$は、Tは3.2.1を満たす最小の集合なので、Sも3.2.1を満たす事を示せば良い。
- $S \subset T$は、Sが最小であることを示す。

当日出た意見：$S \subset T$の方、Sが最小である事よりも直接に$\forall i, S_i \subset T$を帰納法で示した方がかんたん

18 $T \subset S$

- $S_1 = \{true, false, 0\}$ と $S_1 \subset S$ から、3.2.1–1 は OK.
- 3.2.1–2 は、

$$t \in S \overset{S\text{の定義}}{\Leftrightarrow} \exists i, t \in S_i \overset{S_{i+1}\text{の定義}}{\Rightarrow} \exists i, succ(t) \in S_{i+1} \overset{S\text{の定義}}{\Rightarrow} succ(t) \in S$$

pred, iszero も同様。

- 3.2.1–3 は、

$$\begin{aligned} t_1, t_2, t_3 \in S \Leftrightarrow & \exists i, t_1 \in S_i \quad \wedge \\ & \exists j, t_2 \in S_j \quad \wedge \\ & \exists k, t_3 \in S_k \\ \Rightarrow & \exists ijk, t_1, t_2, t_3 \in S_{max\{ijk\}} \quad (S_i \text{ の cumulativity, 演習 3.2.5}) \\ \Rightarrow & \texttt{if } t_1 \ t_2 \ t_3 \in S_{max\{ijk\}+1} \\ \Rightarrow & \texttt{if } t_1 \ t_2 \ t_3 \in S. \end{aligned}$$

19 $S \subset T$

Sが最小　iff

3.2.1を満たす任意のS'について、

また任意のiについて、 $S_i \subset S'$

赤字部分をiについてのcomplete inductionで示す。

- 帰納法の仮定(IH)は、$\forall j, j < i \Rightarrow S_j \leq S'$
- さらに、$i = 0$と$i > 0$（即ち$\exists k, i = k + 1$）とに場合分けする。
  - $i = 0$のときは、$S_i = \emptyset$で明らか。
  - $i = k + 1$のときは、$t \in S_i$をS_iの定義からさらに場合分けして、
    1. $t = true, false, 0$のいずれか
    2. または、ある$t' \in S_k$について$t = succ(t'), pred(t'), iszero(t')$
    3. または、ある$t_1, t_2, t_3 \in S_k$について$t = \texttt{if}\ t_1\ t_2\ t_3$

    (1)のときは3.2.1–1より、(2)のときは3.2.1–2とIHより、(3)のときは3.2.1–3とIHより、それぞれ$t \in S'$となって、おわり。

§3.3 項についての帰納法

21 Def. 3.3.1, 3.3.2

- $\top$が最小なので、これで定義になっている。
- 即ち、この場合分け(pattern match)はexhaustive.

22 Thm. 3.3.4

complete induction の変種二つと structural induction

- （復習）complete induction は、
  $(\forall n \in \mathbb{N}, (\forall i \in \mathbb{N}, i < n \Rightarrow P(i)) \Rightarrow P(n)) \Rightarrow (\forall n \in \mathbb{N}, P(n))$
- Induction on depth は、
  $(\forall s \in T, (\forall r \in T, depth(r) < depth(s) \Rightarrow P(r)) \Rightarrow P(s))$
  $\Rightarrow (\forall s \in T, P(s))$
- Induction on size は、"depth" を "size" に。
- structural induction は、
  $(\forall s \in T, (\forall r \in T,$ (r は s の**直接の** subterm) $\Rightarrow P(r)) \Rightarrow P(s))$
  $\Rightarrow (\forall s \in T, P(s))$
- （おまけ）structural で complete な induction も考えることができて、
  $(\forall s \in T, (\forall r \in T,$ (r は s の subterm) $\Rightarrow P(r)) \Rightarrow P(s))$
  $\Rightarrow (\forall s \in T, P(s))$

赤字は帰納法の仮定 (IH) 部分。

23 Thm. 3.3.4

当日出た意見：この定理は、3章の言語に対して証明するのは簡単だけど、もっと一般の場合を考えて証明するのは大変そう。

24 Cor. (Lemma) 3.3.3

- t の depth についての帰納法で示す。
- IH は、$\forall r \in T, depth(r) < depth(t) \Rightarrow (|consts(r)| \leq size(r))$
- IH を使っているところでは depth がちゃんと減っている事を言わなければならない（本文に補足が必要）
- たとえば三つ目、if $t_1\ t_2\ t_3$ の case で、t_1, t_2, t_3 それぞれについて IH が使われている。
- depth の定義などから、

$$\begin{aligned} depth(t_1) &\leq \max\{depth(t_1), depth(t_2), depth(t_3)\} \\ &< \max\{depth(t_1), depth(t_2), depth(t_3)\} + 1 \\ &= depth(\texttt{if}\ t_1\ t_2\ t_3) \\ &= depth(t) \end{aligned}$$

  (t_2, t_3 も同様)
- となって、depth は確かに減っている。

25 「証明：tについてのinduction。以下略。」

structural inductionなどによる証明は定型になりやすく、p.32上の例のように略記されることが多い。もっと言うと、書いてあるのを読むよりも自分で証明を構成したほうが速く楽なことが多いので、その場合証明を”by induction on t” と丸ごと略すこともある。