§6: 項の名無し表現・§9: 型付き算術式

June 26, 2025

前回の残り

問題 5.3.7

λ**NB** の意味論を，行き詰まり項の評価先である定数項 `wrong` を入れて拡張せよ．

解答: まず，if 文の条件部および算術式のコンストラクタ内に λ 抽象と自由変数が出現してはならない．したがって，

1. $\mathsf{badbool} ::= \mathsf{nv} | \lambda \mathsf{x}.\ \mathsf{t} | \mathsf{x} | \mathsf{wrong}$
2. $\mathsf{badnat} ::= \mathsf{true} | \mathsf{false} | \lambda \mathsf{x}.\ \mathsf{t} | \mathsf{x} | \mathsf{wrong}$

のように変更される．評価規則は変わらない．

λ 項については，型無し λ 計算の評価規則のうち，2 つの合同規則について考えれば十分である．関数部に NB の値または自由変数または行き詰まり項が，引数部に行き詰まり項が出現する場合に行き詰まるから，

$$\mathsf{badfun} ::= \mathsf{x} | \mathsf{true} | \mathsf{false} | \mathsf{nv} | \mathsf{wrong}$$

なる構文要素を定め，以下の評価規則を足せばよい:

$$\mathsf{badfun}\ \mathsf{t}_2 \longrightarrow \mathsf{wrong}$$

$$\frac{\mathsf{t}_2 \longrightarrow \mathsf{wrong}}{\mathsf{v}_1\ \mathsf{t}_2 \longrightarrow \mathsf{wrong}}$$

§6. 項の名無し表現

項と文脈

- 最も内側から数えて k 番目の λ に束縛される変数を k で表わす.
  - ex. $\lambda x.\lambda y.x\ (y\ x)$ は名無し項 $\lambda.\lambda.1\ (0\ 1)$ に対応する
- 名無し項は de Bruijn 項，各束縛変数に対応する番号は de Bruijn index と呼ばれる.

項の中で高々何個の自由変数が出現するかで項を類別しよう.

n 項

$\mathcal{T}$ を，以下を満たす最小の集合族 $\{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \ldots\}$ とする:

1. 任意の $0 \leq k < n$ に対して $k \in \mathcal{T}_n$.
2. $\mathtt{t}_1 \in \mathcal{T}_n$ かつ $n > 0$ ならば，$\lambda.\mathtt{t}_1 \in \mathcal{T}_{n-1}$.
3. $\mathtt{t}_1 \in \mathcal{T}_n$ かつ $\mathtt{t}_2 \in \mathsf{T}_n$ ならば，$(\mathtt{t}_1\ \mathtt{t}_2) \in \mathcal{T}_n$.

$\mathcal{T}_n$ の要素をn 項という.

n 項の性質

$\mathcal{T}$ について，以下の性質が成り立つ:

1. 各 n に対して，$\mathcal{T}_n$ は自由変数を高々 n 個含む項全体の集合である.
   - たとえば，任意の閉項 t は任意の n に対して $\mathrm{t} \in \mathcal{T}_n$ を満たす.
2. 全ての閉項の de Bruijn 表記は一意的であり，同じ de Bruijn 表記をもつ 2 つの閉項は α 同値.

名前付き文脈

n 項の定義では自由変数の de Bruijn index を決定できない．そこで，de Bruijn index から自由変数への割当てを定め，自由変数に index を振るときはその割当てに従うようにする．

- 名前付き文脈 $\Gamma = \{\mathsf{x} \mapsto 4, \mathsf{y} \mapsto 3, \mathsf{z} \mapsto 2, \mathsf{a} \mapsto 1, \mathsf{b} \mapsto 0\}$ に対して，項 $\lambda\mathsf{w}.\ \lambda\mathsf{a}.\ \mathsf{x}$ の de Bruijn 表記は $\lambda.\lambda.6$.

名前付き文脈

$\mathsf{x}_0, \ldots, \mathsf{x}_n \in \mathcal{V}$．名前付き文脈 $\Gamma = \mathsf{x}_n, \mathsf{x}_{n-1}, \ldots, \mathsf{x}_1, \mathsf{x}_0$ は各変数名 x_i を de Bruijn index i に割当てる．Γ に出現する変数名全体の集合を $\mathrm{dom}(\Gamma) = \{\mathsf{x}_n, \ldots, \mathsf{x}_0\}$ を書く．

演習 6.1.4

問題
n 項の集合を Def.3.2.3 の様式で構成し直し，それが上で定義したものと等しいことを示せ．

構成.
Naïve な形だと非可述になるので，各 n について n 項を項の大きさ k で類別する．$\mathcal{S}_n = \bigcup_{k=1}^{\infty} \mathcal{S}_{n,k}$，ただし

$$\mathcal{S}_{0,1} = \emptyset$$

$$\mathcal{S}_{n,1} = \mathcal{S}_{n-1,1} \cup \{n-1\}$$

$$\mathcal{S}_{n,k} = \{\lambda.\mathtt{t}_1 | \mathtt{t}_1 \in \mathcal{S}_{n+1,k-1}\} \cup \bigcup_{i=1}^{k-1} \{\mathtt{t}_1\ \mathtt{t}_2 | \mathtt{t}_1 \in \mathcal{S}_{n,i}, \mathtt{t}_2 \in \mathcal{S}_{n,n-i}\}$$

□

Proof.
n を固定する．示すべきは，$\forall \mathtt{t}, (\exists k, \mathtt{t} \in \mathcal{S}_{n,k}) \Leftrightarrow \mathtt{t} \in \mathcal{T}_n$ なること．

1. $(\Rightarrow)$ $\mathtt{t} \in \mathcal{S}_{n,k}$ なる k をとる．$k = 1$ のときは高々 n の index なので正しい．$k > 1$ のとき．
   - $\mathtt{t}$ が抽象のとき，$\mathtt{t} = \lambda \mathtt{t}_1, \mathtt{t}_1 \in \mathcal{S}_{n+1,k-1}$ なる $\mathtt{t}_1$ を取る．このとき $\mathtt{t}_1 \in \mathcal{S}_{\mathtt{n}+1}$ より $\mathtt{t} \in \mathcal{T}_{n+1}$ なので $\mathtt{t} \in \mathcal{T}_{n+1}$ が従う．
   - 適用の場合も同様．
2. $\mathtt{t}$ の場合分けを行なえばよい．

□

演習 6.1.5.

removenames の定義.
次のように定義する:

$$
\begin{aligned}
\mathit{removenames}_\Gamma(\mathsf{x}) &= \Gamma(\mathsf{x}) \\
\mathit{removenames}_\Gamma(\lambda \mathsf{x}.\mathsf{t}_1) &= \lambda.\mathit{removenames}_{\Gamma,x}(\mathsf{t}_1) \\
\mathit{removenames}_\Gamma(\mathsf{t}_1\ \mathsf{t}_2) &= \mathit{removenames}_\Gamma(\mathsf{t}_1)\ \mathit{removenames}_\Gamma(\mathsf{t}_2)
\end{aligned}
$$

□

restorenames の定義.

$$
\begin{aligned}
\mathit{restorenames}_\Gamma(\mathsf{k}) &= \Gamma \text{ における k 番目の名前} \\
\mathit{restorenames}_\Gamma(\lambda.\mathsf{t}_1) &= \lambda \mathsf{x}.\mathit{restorenames}_{\Gamma,x}(\mathsf{t}_1) \\
&\qquad \mathsf{x}{:}x \notin \mathit{dom}(\Gamma) \text{ なる最初の名前} \\
\mathit{restorenames}_\Gamma(\mathsf{t}_1\ \mathsf{t}_2) &= \mathit{restorenames}_\Gamma(\mathsf{t}_1)\ \mathit{restorenames}_\Gamma(\mathsf{t}_2)
\end{aligned}
$$

□

性質
[1]

1. $\forall \mathtt{t}, \mathtt{t}$は名無し項 $\Rightarrow \mathit{removenames}_\Gamma(\mathit{restorenames}_\Gamma(\mathtt{t})) = \mathtt{t}$
2. $\forall \mathtt{t}, \mathit{restorenames}_\Gamma(\mathit{removenames}_\Gamma(\mathtt{t})) \equiv_\alpha \mathtt{t}$

Proof.

- $\mathtt{t}$ の構造帰納法.
  - $\mathtt{t}$ が index k のとき，Γ における k 番目の名前を $\mathtt{x}_k$ とする．このとき与式は $\mathit{removenames}_\Gamma(\mathtt{x}_k) = k$ となり従う.
  - $\mathtt{t}$ の直接の部分項について主張を仮定する．$\mathtt{t}$ が $\lambda.\mathtt{t}_1$ なら，x を $x \notin \mathit{dom}(\Gamma)$ となる最初の名前として $\mathit{removenames}_{\Gamma,x}(\lambda x.\mathit{restorenames}_{\Gamma,x}(\mathtt{t}_1)) = \lambda.\mathit{removenames}_{\Gamma,x}(\mathit{restorenames}_{\Gamma,x}(\mathtt{t}_1))$．帰納法の仮定より従う．適用の場合も同様.
- $\mathtt{t}$ の構造帰納法．$\mathtt{t}$ が通常の λ 項になること以外変わらない.

□

[1] ただし $\mathit{FV}(\mathtt{t}) \in \mathit{dom}(\Gamma)$

シフトと代入

de Bruijn 記法での代入操作を考える．代入項の自由変数が捕獲されないためには，代入項の束縛変数の index を変えず，自由変数の index を最外束縛子を指す index ぶんだけ増加させる．これを形式化しよう．

- ex. $[1 \mapsto 2(\lambda.0)]\,\lambda.2 = \lambda.4(\lambda.0)$(名前付き項での $[\mathtt{x} \mapsto \mathtt{z}(\lambda\mathtt{w}.\mathtt{w})]\,(\lambda\mathtt{y}.\mathtt{x}) = \lambda\mathtt{y}.\mathtt{z}(\lambda\mathtt{w}.\mathtt{w})$ に対応)

シフト

項 $\mathtt{t}$ の，打ち切り値 c 以上の d シフトを $\uparrow_c^d(\mathtt{t})$ と書き，次のように定義する:

$$\uparrow_c^d(\mathtt{k}) = \begin{cases} \mathtt{k} & \text{if } k < c \\ \mathtt{k} + \mathtt{d} & \text{otherwise} \end{cases}$$

$$\uparrow_c^d(\lambda.\mathtt{t}_1) = \lambda.\uparrow_{c+1}^d(\mathtt{t}_1)$$

$$\uparrow_c^d(\mathtt{t}_1\ \mathtt{t}_2) = \uparrow_c^d(\mathtt{t}_1)\ \uparrow_c^d(\mathtt{t}_2)$$

ただし $\uparrow^d(\mathtt{t}) = \uparrow_0^d(\mathtt{t})$

演習 6.2.3

問
t が n 項とする．加えて $d < 0$ である場合には，t の自由変数はどれも $|d|$ 以上とする．このとき，$\uparrow_c^d(\mathsf{t})$ が $\max(n+d, 0)$ であることを示せ．

Proof.
d の正負で分ける．n 項 t に関する構造的帰納法で示す．

1. $d \geq 0$ のとき．
   1. t が index $k(< n)$ の場合は明らか ($k < d$ のときはシフト後 k，そうでない場合シフト後は $k + d < n + d$ となり，$n + d$ 項となる．)
   2. 各項 t に対して，t の任意の直接の部分項 s と任意の $k \geq 0$ に対して $\uparrow_{c+k}^d(\mathsf{s}) \in \mathcal{T}_{\max(n+d+k,0)}$ を仮定する．$\mathsf{t} = \lambda.\mathsf{t}_1$ の場合は $\uparrow_c^d(\mathsf{t}) = \lambda.\uparrow_{c+1}^d(\mathsf{t}_1)$ となり，$\uparrow_{c+1}^d(\mathsf{t}_1) \in \mathcal{T}_{\max(n+d+1,1)}$ だから $\uparrow_c^d(\mathsf{t}) \in \mathcal{T}_{\max(n+d+1,0)}$．$\mathsf{t}_1\ \mathsf{t}_2$ の場合は $\uparrow_c^d(\mathsf{t}_1), \uparrow_c^d(\mathsf{t}_2) \in \mathcal{T}_{\max(n+d+1,1)}$ より従う．
2. $d < 0$ のとき．t の自由変数は全て $|d| = -d$ 以上とする．
   1. t が index k の場合は仮定より $-d \leq k$．よって $0 \leq k + d < n + d$ より $\max(n + d, 0)$ 項．
   2. 題意の各項 t に対して，t の任意の直接の部分項と任意の $k \geq 0$ に対して $\uparrow_c^d(\mathsf{s}) \in \mathcal{T}_{\max(n+d+k,0)}$ を仮定する．抽象および関数適用の場合は全く (1) と同様に証明できる．

代入

§6.3 で見るように，β 簡約を定めるには文脈中最後にある変数, i.e. $j = 0$ への代入があればよいが，たとえば $(\lambda.\lambda.1)(\lambda.0)$ を β 簡約するときのように，本体で index 1 の変数にも代入出来なければならない．より多くの束縛子がネストしている場合も同様なので，一般の変数に対して代入操作ができなければならないことがわかる．そこで，次のように定義する．

Def.6.2.4. 代入

項 t における，変数番号 j への項 s の代入を，$[\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{t}$ と書き，次のように定義する．

$$[\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{k} = \begin{cases} \mathsf{s} & \text{if } \mathsf{k} = \mathsf{j} \\ \mathsf{k} & \text{otherwise} \end{cases}$$

$$[\mathsf{j} \mapsto \mathsf{s}](\lambda.\mathsf{t}_1) = \lambda.\left[\mathsf{j}+1 \mapsto \uparrow^1 (\mathsf{s})\right] \mathsf{t}_1$$

$$[\mathsf{j} \mapsto \mathsf{s}](\mathsf{t}_1\ \mathsf{t}_2) = ([\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{t}_1)([\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{t}_2)$$

演習 6.2.6

問題
s, t が n 項で $\mathsf{j} \leq \mathsf{n}$ のとき，$[\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{t}$ は n 項である．

Proof.
t の帰納法による．

1. t が変数のとき．$\mathsf{t} = \mathsf{j}$ ならば代入の結果は s となり，仮定より従う．
   それ以外の場合は代入の結果は t のままで，t も n 項より示される．
2. $\forall n \in \mathbb{N}, \forall j \leq n, \forall \mathsf{s}, \mathsf{t} \in \mathcal{T}_n, [\mathsf{j} \mapsto \mathsf{s}]\,\mathsf{t} \in \mathcal{T}_n$ を仮定する．
   1. $ß\mathsf{t} = \lambda.\mathsf{t}_1$ のとき．演習 6.2.3 より $\uparrow^1 (\mathsf{s})$ は $n+1$ 項．また，特に t も $n+1$ 項なので $\left[\mathsf{j}+1 \mapsto \uparrow^1 (\mathsf{s})\right] \mathsf{t}_1$ は $n+1$ 項．よって
   2. $\mathsf{t} = \mathsf{t}_1\ \mathsf{t}_2$ のとき．帰納法の仮定より従う．

□

演習 6.2.8

問題
名無し項についての代入の定義は，通常の項についての代入の定義[2]と合致すべきであるが，(1) その対応を厳密に正当化するためにはどのような定理が証明される必要があるかを考え，(2) それを証明せよ．

removenames が代入操作について準同型写像となっていることを示せばよい．つまり，以下の定理を示す．

Thm.de Bruijn 記法での代入操作と Def.5.3.5 の一致性.

任意の通常の項 s, t，変数 x について $\mathrm{removenames}_\Gamma([\mathsf{x} \mapsto \mathsf{s}]\,\mathsf{t}) = [\Gamma(x) \mapsto \mathrm{removenames}_\Gamma(\mathsf{s})]\,\mathrm{removenames}_\Gamma(\mathsf{t})$．ただし Γ は名前付き文脈とする．

[2] Def.5.3.5

演習 6.2.8.(証明)

Proof.
通常の項 s と変数 x を任意に取る．項 t に関する構造的帰納法で示す．

1. t が変数のとき．
   - $t = x$ のとき．(左辺) = $\mathrm{removenames}_\Gamma([x \mapsto s]\, x) = \mathrm{removenames}_\Gamma(s)$，(右辺) = $[\Gamma(x) \mapsto \mathit{removenames}_\Gamma(s)]\, \Gamma(x) = \mathit{removenames}_\Gamma(s)$ より従う．
   - $t = y \neq x$ のとき．$\mathit{removenames}_\Gamma(y) = \Gamma(y)$ とすれば，Def.6.2.4. より (左辺) = $\Gamma(y)$．一方で Def.5.3.5 の定義により，(右辺) = $\mathit{removenames}_\Gamma([x \mapsto s]\, y) = \mathit{removenames}_\Gamma(y) = \Gamma(y)$ より示される．
2. 各 t の任意の直接の部分項 u に対して主張を仮定．適用の場合は明らか．

   - $t = \lambda y.t_1$ の場合．α 同値な項の同一視により，$y = x$ と $y \neq x \wedge y \notin \mathit{FV}(s)$ を考えればよい．
     1. $y = x$ のとき．(左辺) = $\mathit{removenames}_\Gamma(\lambda x.t_1) = \lambda.\mathit{removenames}_{\Gamma,x}(t_1)$．一方で右辺は $[\Gamma(x) \mapsto \mathit{removenames}_\Gamma(s)]\, (\lambda.\mathit{removenames}_{\Gamma,x}(t_1)) = \lambda.\left[\Gamma(x)+1 \mapsto\uparrow^1 (s)\right] \mathit{removenames}_{\Gamma,x}(t_1)$．ここで帰納法の仮定から $\lambda.\mathit{removenames}_{\Gamma,x}(t_1)$[3]となり示される．
     2. $y \neq x \wedge y \notin \mathit{FV}(s)$．(左辺) = $\lambda.\mathit{removenames}_{\Gamma,y}([x \mapsto s]\, t_1)$．帰納法の仮定により $\mathit{removenames}_{\Gamma,y}([x \mapsto s]\, t_1) = [\mathit{removenames}_{\Gamma,y}(x) \mapsto \mathit{removenames}_{\Gamma,y}(s)]\, \mathit{removenames}_{\Gamma,y}(t_1)$．一方右辺は $\left[\Gamma(x)+1 \mapsto\uparrow^1 (\mathit{removenames}_\Gamma(s))\right] \mathit{removenames}_{\Gamma,y}(t_1)$ より示される．

□

[3] $x \in \Gamma$ でなければならない．

演習 6.2.8.(続き)

ただし 2-(1), (2) の証明では以下の補題を用いた:

Lem. 名前付き文脈に関する補題

1. 任意の文脈 Γ，変数 x，項 t について，
   $\uparrow^1 (\mathit{removenames}_\Gamma(\mathtt{t})) = \mathit{removenames}_{\Gamma,\mathtt{t}}(\mathtt{t})$

簡約基を簡約することで束縛変数が消費されるので，代入結果の中の変数に番号を振り直す必要がある．また，被代入項の束縛子の数は代入項より多いから，de Bruijn index が負となることを防ぐために予め 1 を加えるシフトを行なう必要がある．以上より，

β 簡約規則

$$(\lambda.\mathtt{t}_{12})\mathtt{v}_2 \to \uparrow^{-1}\left(\left[\mathtt{0} \mapsto \uparrow^{1}(\mathtt{v}_2)\right]\mathtt{t}_{12}\right) \text{(E-APPABS)}$$

他の評価規則は同一．

§8. 型付き算術式

算術式の構文は

$$\mathtt{t} ::= \mathtt{true} | \mathtt{false} | \mathtt{if\ t\ then\ t\ else\ t} | \mathtt{0} | \mathtt{succ\ t} | \mathtt{pred\ t} | \mathtt{iszero\ t}$$

により定義され，各項を評価すると結果は行き詰まり項もしくは値，つまりブール値 $\mathtt{v} ::= \mathtt{true} | \mathtt{false}$ または数値 $\mathtt{nv} ::= \mathtt{0} | \mathtt{succ\ nv}$ になる．

ある項が行き詰まり項の状態に至ることを，その項を実際に評価せずにいいたい．そのためには，数値に評価される項とブール値に評価される項を区別し，各コンストラクタが引数として取れる項を限定する必要がある．そのために，型 $\mathrm{Nat}, \mathrm{Bool}$ を導入する．

記法
$\mathsf{S}, \mathsf{T}, \mathsf{U}$ は型の上を動くメタ変数とする．また，項 $\mathtt{t}$ が値 T を持つというとき，$\mathtt{t}$ を評価した結果が明らかに，i.e. 静的に型 T の値になることを意味する．

算術式のための型付け関係を $t : T$ と書き，以下の推論規則によって定める．

B のための型付け規則

新しい構文形式

$$T ::= \mathtt{Bool}$$

新しい型付け規則

$$\frac{}{\mathtt{true} : \mathtt{Bool}}\ \text{T-TRUE}$$

$$\frac{}{\mathtt{false} : \mathtt{Bool}}\ \text{T-FALSE}$$

$$\frac{t_1 : \mathtt{Bool} \qquad t_2 : T \qquad t_3 : T}{\mathtt{if}\ t_1\ \mathtt{then}\ t_2\ \mathtt{else}\ t_3 : T}\ \text{T-IF}$$

NB のための型付け規則

新しい構文形式

$$T ::= \ldots \mathtt{Nat}$$

新しい型付け規則

$$\frac{}{\mathtt{0} : \mathtt{Nat}}\ \text{T-ZERO}$$

$$\frac{t_1 : \mathtt{Nat}}{\mathtt{succ}\ t_1 : \mathtt{Nat}}\ \text{T-SUCC}$$

$$\frac{t_1 : \mathtt{Nat}}{\mathtt{pred}\ t_1 : \mathtt{Nat}}\ \text{T-PRED}$$

$$\frac{t_1 : \mathtt{Nat}}{\mathtt{iszero}\ t_1 : \mathtt{Bool}}\ \text{T-ISZERO}$$

Def.8.2.1.

算術式のための型付け関係は p.19 における規則の全てのインスタンスを満たす，項と型の最小の二項関係である．項 t が型付け可能 (正しく型付けされている) とは，$\exists T, t : T$ なること．

たとえは $\mathtt{succ}\ t_1$ が型付け可能のとき，その型は Nat かつ $t_1 : \mathtt{Nat}$ である．このような性質は逆転補題 (生成補題) と呼ばれる．

Lem8.2.2. 型付け関係の逆転.

1. $\mathtt{true} : R$ ならば $R = \mathtt{Bool}$.
2. $\mathtt{false} : R$ ならば $R = \mathtt{Bool}$.
3. $\mathtt{if}\ t_1\ \mathtt{then}\ t_2\ \mathtt{else}\ t_3 : R$ ならば $t_1 : \mathtt{Bool} \land t_2 : R \land t_3 : R$.
4. $0 : R$ ならば $R : \mathtt{Nat}$.
5. $\mathtt{succ}\ t_1 : R$ ならば $R : \mathtt{Nat} \land t_1 : \mathtt{Nat}$.
6. $\mathtt{pred}\ t_1 : R$ ならば $R : \mathtt{Nat} \land t_1 : \mathtt{Nat}$.
7. $\mathtt{iszero}\ t_1 : R$ ならば $R : \mathtt{Nat} \land t_1 : \mathtt{Nat}$.

演習 8.2.3

問題
正しく型付けされた項の全ての部分項は正しく型付けされている.

Proof.
項 $\mathtt{t}$ の帰納法.

1. $\mathtt{t}$ が定数のとき. $\mathtt{t}$ の任意の部分項は $\mathtt{t}$ 自身に限られ, p.19 の規則から全て well-typed なことより従う.
2. 各項 $\mathtt{t}$ に対し, well-typed な $\mathtt{t}$ の直接の部分項たち $\mathtt{t}'$ の任意の部分項 $\mathtt{s}$ は well-typed なことを仮定する.
   1. $\mathtt{t}$ が $\mathtt{t}_1$ に $\mathtt{succ}, \mathtt{pred}, \mathtt{iszero}$ のいずれかを適用した項のとき. $\mathtt{t}$ が well-typed なことを仮定する. 逆転補題により, $\mathtt{t}_1 : \mathtt{Bool}$ より $\mathtt{t}_1$ は well-typed である. 帰納法の仮定より, $\mathtt{t}_1$ の任意の部分項は well-typed, かつ $\mathtt{t}$ はそれ自身の部分項なので示される.
   2. $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ のとき. $\mathtt{t}$ が well-typed なことを仮定すると, 逆転補題より $\mathtt{t}_1, \mathtt{t}_2, \mathtt{t}_3$ は well-typed. 帰納法の仮定より, (1) と全く同様にして主張が従う.

□

型付け導出とその一意性.

Def. 型付け導出

型付け関係 $t : T$ の型付け導出は，$t : T$ を結論とする型付け規則のインスタンスの木である.

以下の性質は一般には成りたたない．たとえば subtyping などを持つ型体系においては，ある項が存在して，型付け可能かつ 2 通り以上の型をもちうる.

Thm.8.2.4

各項 t は高々一つの型を持つ．i.e.t が型付け可能ならばその型は一意的である.

p.19 で定めた型体系の安全性[4]，つまり任意の項 t に対し，t が型付け可能ならば t の評価は行き詰まらないことを示す．以下の 2 つの補題を示すことによって証明しよう．

Thm.8.3.2. 進行定理

$$\forall \mathsf{t}, \exists \mathsf{T}, \mathsf{t} : \mathsf{T} \Rightarrow (\mathsf{t}\text{は値} \vee \exists \mathsf{t}', \mathsf{t} \to \mathsf{t}')$$

Thm.8.3.3. 保存定理

$$\forall \mathsf{t}, \mathsf{t}', \forall T, \mathsf{t} : \mathsf{T} \wedge \mathsf{t} \to \mathsf{t}' \Rightarrow \mathsf{t}' : \mathsf{T}$$

[4]つまりは健全性

証明の準備

Bool 型と Nat 型の標準形，i.e. それらの型を持つ正しく型付けされた値について，以下の性質が成り立つ．

Lem.8.3.1. 標準形

1. v が Bool 型の値ならば，v は true または false．
2. v が Nat 型の値ならば，v は nv ::= 0|succ nv の定める文法による数値．

Proof.
(1) ブール値の文法 v ::= true|false と併せると，この言語における値は 4 つの形 true, false, 0, succ nv(ただし nv は数値) を持つ．はじめの 2 つの場合は **B** の型付け規則より即座に成り立つ．後者 2 つについては，逆転補題によりともに Bool = Nat が導かれ，これは不可能．

(2) はじめ 2 つの場合は逆転補題より Nat = Bool が導かれ，これは不可能．後者 2 つに対しては即座に成り立つ． □

Thm.8.3.2. の証明

$\mathtt{t} : \mathtt{T}$ の導出に関する帰納法で以下の定理を示す．

Thm.8.3.2. 進行定理

$$\forall \mathtt{t}, \exists \mathtt{T}, \mathtt{t} : \mathtt{T} \Rightarrow (\mathtt{t}\text{は値} \vee \exists \mathtt{t}', \mathtt{t} \to \mathtt{t}')$$

Proof.

1. T-TRUE，T-FALSE，T-ZERO の場合は $\mathtt{t}$ は値より従う．
2. それ以外の場合．$\mathtt{t}$ の直接の部分導出に対して主張を仮定．
   1. T-IF の場合．$\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3 : \mathtt{T}$ とする．逆転補題から $\mathtt{t}_1 : \mathtt{Bool}, \mathtt{t}_2 : \mathtt{T}, \mathtt{t}_3 : \mathtt{T}$．帰納法の仮定より $\mathtt{t}_1$ は値または $\exists \mathtt{t}_1', \mathtt{t}_1 \to \mathtt{t}_1'$．$\mathtt{t}_1$ が値のとき，Lem.8.3.1 より $\mathtt{t}_1 = \mathtt{true}$ または $\mathtt{t}_1 = \mathtt{false}$．このとき E-TRUE(FALSE) が適用できる．そうでない場合は E-IF が適用される．
   2. T-SUCC(PRED|ISZERO) の場合．$\mathtt{t} = \mathtt{succ}\ \mathtt{t}_1$ の場合を考えればよい．逆転補題より $\mathtt{t}_1 : \mathtt{Nat}$．帰納法の仮定より $\mathtt{t}_1$ は値または $\exists \mathtt{t}_1', \mathtt{t}_1 \to \mathtt{t}_1'$．値のときは Lem.8.3.1 より $\mathtt{t}_1$ は数値で，$\mathtt{t}$ も数値．1 ステップの評価が可能な場合は E-SUCC が適用され，$\mathtt{t} \to \mathtt{succ}\ \mathtt{t}_1'$．

□

Thm.8.3.3. の証明.

$\mathtt{t} : \mathtt{T}$ の導出に関する帰納法で以下の定理を示す．$\mathtt{t} \to \mathtt{t}'$ の評価の導出に関する induction による証明は Ex.8.3.4 で行なう．

Thm.8.3.3. 保存定理

$$\forall \mathtt{t}, \mathtt{t}', \forall T, \mathtt{t} : \mathtt{T} \land \mathtt{t} \to \mathtt{t}' \Rightarrow \mathtt{t}' : \mathtt{T}$$

Thm.8.3.3. の証明.

Proof.
帰納法の各ステップにおいて，任意の部分導出に対して題意を仮定する．
i.e.$\mathsf{s} : \mathsf{S}$ が部分導出で証明されるとき，$\forall \mathsf{s}'.\mathsf{s} : \mathsf{S} \land \mathsf{s} \to \mathsf{s}' \Rightarrow \mathsf{s}' : \mathsf{S}$ を仮定する．最後の導出で使われた型付け規則で分ける．

1. T-(TRUE|FALSE|ZERO) のとき．規則の形より $\mathsf{t} = \mathtt{true}$，かつ $\mathsf{T} : \mathtt{Bool}$ なることが分かる．しかし t は正規形より $\forall \mathsf{t}', \neg(\mathsf{t} \to \mathsf{t}')$ なので，主張は成立．他の規則も同様.
2. T-IF のとき．規則の形より $\mathsf{t} = \mathtt{if}\ \mathsf{t}_1\ \mathtt{then}\ \mathsf{t}_2\ \mathtt{else}\ \mathsf{t}_3, \mathsf{t}_1 : \mathsf{T}, \mathsf{t}_2 : \mathsf{T}, \mathsf{t}_3 : \mathsf{T}$ なる項 $\mathsf{t}_1, \mathsf{t}_2, \mathsf{t}_3$ および型 T を取る．帰納法の仮定より，$\mathsf{t}_1 : \mathtt{Bool}, \mathsf{t}_2 : \mathsf{T}, \mathsf{t}_3 : \mathsf{T}$ を結論とする部分導出が存在する．$\mathsf{t} \to \mathsf{t}'$ を仮定すると，規則は E-IF(TRUE|FALSE)，E-IF のいずれかが適用できる.
   - E-IF(TRUE|FALSE) のとき．規則の形より $\mathsf{t}_1 = \mathtt{true}$ で，結果の項は $\mathsf{t}' = \mathsf{t}_2 : \mathsf{T}$ より従う．$\mathsf{t}_1 = \mathtt{false}$ の場合も同様.
   - E-IF のとき．規則の形より $\mathsf{t}_1 \to \mathsf{t}_1'$ が存在する．帰納法の仮定より $\mathsf{t}_1 : \mathtt{Bool}$ を併せると $\mathsf{t}_1' : \mathtt{Bool}$．よって規則 T-IF を使うと $\mathsf{t}' = \mathtt{if}\ \mathsf{t}_1'\ \mathtt{then}\ \mathsf{t}_2\ \mathtt{else}\ \mathsf{t}_3 : \mathsf{T}$ より正しい.
3. T-(SUCC|PRED|ISZERO) のときは (2) と同様.

□

Thm.8.3.3 の別証

Proof.
評価導出による証明．帰納法の各ステップにおいて，任意の部分導出に対して題意を仮定する:$s \to s'$ が部分導出で証明できるとき，
$\forall S, s : S \land s \to s' \Rightarrow s' : S$ を仮定し，最後に使われた規則で分ける．

1. E-IFTRUE のとき．$t = \texttt{if true then } t_2 \texttt{ else } t_3 \to t_2$.

□