Substitutions explicites linéaires et préservation de la normalisation forte: une formalisation en Coq

Daniel El Ouraoui & Micaela Mayero & Damiano Mazza

LIPN

28 novembre 2016

Présentation

Cette présentation est le fruit d'un travail encadré par Micaela Mayero et Damiano Mazza.

Objectif

Terminer la formalisation en Coq initiée par Agathe Herrou de la preuve "papier-crayon" faite par Beniamino Accattoli de la préservation de la normalisation forte pour le calcul des substitutions linéaires.

Généralités

Les substitutions explicites

1. La substitution comme constructeur dans la syntaxe du λ-calcul
2. Ajout de règles de réécriture
3. Choix dans la stratégie d'évaluation

Généralités

Grammaire des termes du calcul de substitutions explicites

$$\Lambda : t,u : := x \mid \lambda x.t \mid t\ u \mid t[x/u]$$

Les indices de De Bruijn

1. On représente les variables dans les entiers naturels

Généralités

Les indices de De Bruijn

1. On représente les variables dans les entiers naturels
2. Chaque entier représente le nombre de lieurs qui le précède

Généralités

Les indices de De Bruijn

1. On représente les variables dans les entiers naturels
2. Chaque entier représente le nombre de lieurs qui le précède
3. Les variables libres sont représentées par des entiers de valeur supérieure ou égale au nombre de lieurs qui les précèdent (en fonction des conventions 0 ou 1)

Généralités

Quelques notations

une variable x est notée $n \in \mathbf{N}$

une λ-abstraction : $\lambda.0$ pour $\lambda x.x$

une substitution explicite : $t[u]$ pour $t[u/x]$

Code

```
Inductive term : Set :=
  | TVar : nat → term
  | TAbs : term → term
  | TApp : term → term → term
  | TSub : term → term → term.
```

Généralités

Exemple

Le terme : $\lambda x.\lambda y.(y\ (\lambda z.y\ z)\ w)$

Généralités

Exemple

Le terme : $\lambda x.\lambda y.(y\ (\lambda z.y\ z)\ w)$

est représenté par : $\lambda.\lambda.(0\ (\lambda.1\ 0)\ 2)$

Opérateur de lift

$$\uparrow_k^n(x) \begin{cases} k \leq x & \Rightarrow \quad x+n \\ k > x & \Rightarrow \quad x \end{cases}$$

$$\uparrow_k^n(\lambda t) = \lambda(\uparrow_{k+1}^n(t))$$

$$\uparrow_k^n(uv) = \uparrow_k^n(u)\uparrow_k^n(v)$$

$$\uparrow_k^n(u[v]) = \uparrow_{k+1}^n(u)[\uparrow_k^n(v)]$$

Opérateur de lift

$$\uparrow_k^n(x) \begin{cases} k \leq x & \Rightarrow \quad x+n \\ k > x & \Rightarrow \quad x \end{cases}$$

$$\uparrow_k^n(\lambda t) = \lambda(\uparrow_{k+1}^n(t))$$

$$\uparrow_k^n(uv) = \uparrow_k^n(u)\uparrow_k^n(v)$$

$$\uparrow_k^n(u[v]) = \uparrow_{k+1}^n(u)[\uparrow_k^n(v)]$$

On utilise presque toujours $\uparrow_0^n(t)$

Opérateur de lift

$$\uparrow_k^n(x) \begin{cases} k \leq x & \Rightarrow \quad x+n \\ k > x & \Rightarrow \quad x \end{cases}$$

$$\uparrow_k^n(\lambda t) = \lambda(\uparrow_{k+1}^n(t))$$

$$\uparrow_k^n(uv) = \uparrow_k^n(u)\uparrow_k^n(v)$$

$$\uparrow_k^n(u[v]) = \uparrow_{k+1}^n(u)[\uparrow_k^n(v)]$$

On utilise presque toujours $\uparrow_0^n(t)$

Et très souvent $\uparrow_0^1(t)$ Que l'on écrit $\uparrow(t)$

Code

```
Fixpoint lift_rec n t {struct t} : nat → term :=
  fun k ⇒
    match t with
      | TVar i ⇒
        (* si k<=i then ... else ... *)
        match le_gt_dec k i with
          | left _ ⇒ TVar (n + i)
          | right _ ⇒ TVar i
        end
      | \u ⇒ \(lift_rec n u (S k))
      | u · v ⇒ (lift_rec n u k) ·(lift_rec n v k)
      | u [v] ⇒ (lift_rec n u (S k))[(lift_rec n v k)]
    end.
```

la substitution (méta-opération)

$$\mathrm{i}\{\mathrm{v}/\mathrm{x}\} \begin{cases} i \neq x & \Rightarrow \quad i \\ i = x & \Rightarrow \quad v \end{cases}$$

$$(\lambda \mathrm{t})\{\mathrm{v}/\mathrm{x}\} = \lambda(\mathrm{t}\{\uparrow(\mathrm{v})/\mathrm{x}{+}1\})$$

$$(\mathrm{uv})\{\mathrm{w}/\mathrm{x}\} = (\mathrm{u}\{\mathrm{w}/\mathrm{x}\})(\mathrm{v}\{\mathrm{w}/\mathrm{x}\})$$

$$(\mathrm{u}[\mathrm{v}])\{\mathrm{w}/\mathrm{x}\} = (\mathrm{u}\{\uparrow(\mathrm{w})/\mathrm{x}{+}1\})[\mathrm{v}\{\mathrm{w}/\mathrm{x}\}]$$

la substitution (méta-opération)

$$\mathrm{i}\{\mathrm{v}/\mathrm{x}\} \begin{cases} i \neq x & \Rightarrow \quad i \\ i = x & \Rightarrow \quad v \end{cases}$$

$$(\lambda \mathrm{t})\{\mathrm{v}/\mathrm{x}\} = \lambda(\mathrm{t}\{\uparrow(\mathrm{v})/\mathrm{x}{+}1\})$$

$$(\mathrm{uv})\{\mathrm{w}/\mathrm{x}\} = (\mathrm{u}\{\mathrm{w}/\mathrm{x}\})(\mathrm{v}\{\mathrm{w}/\mathrm{x}\})$$

$$(\mathrm{u}[\mathrm{v}])\{\mathrm{w}/\mathrm{x}\} = (\mathrm{u}\{\uparrow(\mathrm{w})/\mathrm{x}{+}1\})[\mathrm{v}\{\mathrm{w}/\mathrm{x}\}]$$

On utilise presque toujours $\mathrm{t}\{v/0\}$

Code

```
Fixpoint subst_rec u t {struct t} : nat → term :=
  fun k ⇒
    match t with
      | TVar i ⇒
        (* si i=k then ... else ... *)
        match eq_nat_dec k i with
          | left _ ⇒ u
          | right _ ⇒ TVar i
        end
      | \v ⇒ \(subst_rec (shift u) v (S k))
      | w·v ⇒ (subst_rec u w k)·(subst_rec u v k)
      | w[v] ⇒ (subst_rec (shift u) w (S k))[(subst_rec u v k)]
    end.
```

Contextes

Grammaire

Les contextes nous permettent de définir des λ-termes, contenant des "trous".

Ils nous indiquent comment réduire les sous termes.

$$C ::= \langle\cdot\rangle \mid \lambda x.C \mid Ct \mid tC \mid C[t/x] \mid t[C/x] \quad \text{(contextes généraux)}$$

$$L ::= \langle\cdot\rangle \mid L[t/x] \quad \text{(contextes de substitution)}$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \\
\\
\\
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \\
\\
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\\
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\\
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x]
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x] \to_{db} z[y/z][(\lambda z.z)/x]
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x] \to_{db} z[y/z][(\lambda z.z)/x] \to_C \langle\cdot\rangle[y/z][(\lambda z.z)/x]
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x] \to_{db} z[y/z][(\lambda z.z)/x] \to_C \langle\cdot\rangle[y/z][(\lambda z.z)/x] \\
\to_{ls} y[y/z][(\lambda z.z)/x]
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x] \to_{db} z[y/z][(\lambda z.z)/x] \to_C \langle\cdot\rangle[y/z][(\lambda z.z)/x] \\
\to_{ls} y[y/z][(\lambda z.z)/x] \to_{Gc} y[y/z]
\end{cases}
$$

Substitution linéaire

Les règles de réécriture

$$
\begin{aligned}
L\langle \lambda x.t\rangle u &\to_{\mathsf{db}} L\langle t[u/x]\rangle \\
C\langle x\rangle[u/x] &\to_{\mathsf{ls}} C\langle u\rangle[u/x] \\
t[u/x] &\to_{\mathsf{gc}} t \qquad\qquad \text{si } x \notin \mathrm{fv}(t)
\end{aligned}
$$

exemple

On note $\to_C$ la règle de propagation des contextes

$$
\begin{cases}
(\lambda x.xy)(\lambda z.z) \to_{db} xy[(\lambda z.z)/x] \to_C \langle\cdot\rangle y[(\lambda z.z)/x] \\
\to_{ls} (\lambda z.z)y[(\lambda z.z)/x] \to_{db} z[y/z][(\lambda z.z)/x] \to_C \langle\cdot\rangle[y/z][(\lambda z.z)/x] \\
\to_{ls} y[y/z][(\lambda z.z)/x] \to_{Gc} y[y/z] \to_{Gc} y
\end{cases}
$$

Formalisation

Lemma `lift_commut`: $\forall$ `a k i j p`,

$$\texttt{k} <= \texttt{p} \rightarrow \uparrow_k^i (\uparrow_p^j a) = \uparrow_{(i+p)}^j (\uparrow_k^i a).$$

Formalisation

Lemma `lift_commut`: $\forall$ `a k i j p`,

$$\texttt{k} <= \texttt{p} \rightarrow \uparrow^{i}_{k}(\uparrow^{j}_{p} a) = \uparrow^{j}_{(i+p)}(\uparrow^{i}_{k} a).$$

Lemma `simpl_lift` : $\forall$ `t n`, $\uparrow^{(Sn)} t = \uparrow(\uparrow^{n} t)$.

Formalisation

Lemma `lift_commut`: $\forall$ `a k i j p`,
$$\texttt{k} <= \texttt{p} \rightarrow \uparrow^{i}_{k}(\uparrow^{j}_{p} a) = \uparrow^{j}_{(i+p)}(\uparrow^{i}_{k} a).$$

Lemma `simpl_lift` : $\forall$ `t n`, $\uparrow^{(Sn)} t = \uparrow(\uparrow^{n} t)$.

Lemma `shift_commut`: $\forall$ `u j`, $\uparrow^{j}(\uparrow u) = \uparrow(\uparrow^{j} u)$.

Formalisation

Lemma lift_commut: $\forall$ a k i j p,
k <= p $\rightarrow \uparrow^{i}_{k}(\uparrow^{j}_{p} a) = \uparrow^{j}_{(i+p)}(\uparrow^{i}_{k} a)$.

Lemma simpl_lift : $\forall$ t n, $\uparrow^{(Sn)} t = \uparrow(\uparrow^{n} t)$.

Lemma shift_commut: $\forall$ u j, $\uparrow^{j}(\uparrow u) = \uparrow(\uparrow^{j} u)$.

Lemma commut_lift_subst_rec : $\forall$ t u n k h,
k <= h $\rightarrow \uparrow^{k}_{n}(t\{h := u\}) = (\uparrow^{k}_{n} t)\ \{(n+h) := (\uparrow^{k}_{n} u)\}$.

Formalisation

Lemma lift_commut: $\forall$ a k i j p,
$$\texttt{k} <= \texttt{p} \rightarrow \uparrow^{i}_{k}(\uparrow^{j}_{p} a) = \uparrow^{j}_{(i+p)}(\uparrow^{i}_{k} a).$$

Lemma simpl_lift : $\forall$ t n, $\uparrow^{(Sn)} t = \uparrow(\uparrow^{n} t)$.

Lemma shift_commut: $\forall$ u j, $\uparrow^{j}(\uparrow u) = \uparrow(\uparrow^{j} u)$.

Lemma commut_lift_subst_rec : $\forall$ t u n k h,
$$\texttt{k} <= \texttt{h} \rightarrow \uparrow^{k}_{n}(t\{h := u\}) = (\uparrow^{k}_{n} t)\ \{(\texttt{n+h}) := (\uparrow^{k}_{n} u)\}.$$

Lemma commut_lift_subst : $\forall$ t u k,
$$(\uparrow t)\ \{(\texttt{S k}) := \uparrow u\} = \uparrow \texttt{t}\ \{\texttt{k} := \texttt{u}\}.$$

Formalisation

Traduction des règles

1. formalisation des règles $\rightarrow_{\mathsf{db}}$ $\rightarrow_{\mathsf{ls}}$ $\rightarrow_{\mathsf{gc}}$

Formalisation

Traduction des règles

1. formalisation des règles $\rightarrow_{\mathsf{db}}$ $\rightarrow_{\mathsf{ls}}$ $\rightarrow_{\mathsf{gc}}$
2. ces trois règles peuvent être exprimées modulo des contextes arbitraires

Formalisation

Traduction des règles

1. formalisation des règles $\rightarrow_{\mathsf{db}}$ $\rightarrow_{\mathsf{ls}}$ $\rightarrow_{\mathsf{gc}}$
2. ces trois règles peuvent être exprimées modulo des contextes arbitraires
3. explicitation des contextes

Formalisation

Traduction des règles

1. formalisation des règles $\rightarrow_{\mathsf{db}}$ $\rightarrow_{\mathsf{ls}}$ $\rightarrow_{\mathsf{gc}}$
2. ces trois règles peuvent être exprimées modulo des contextes arbitraires
3. explicitation des contextes
4. Nous avons choisi cette solution pour exprimer ces trois règles dans Coq

Formalisation

DB

$$\frac{}{(\lambda x.t)\,u \rightarrow_{db} t[u]} \quad (db\_Lam)$$

$$\frac{t\ u \rightarrow_{db} r}{\uparrow t[s]\ u \rightarrow_{db} \uparrow r[s]} \quad (db\_App)$$

Formalisation

Ls

$$\frac{}{x[u] \rightarrow^{x}_{lsa} \uparrow u[u]} \quad (ls\_Var)$$

$$\frac{t[\uparrow u] \rightarrow^{x+1}_{lsa} t'[\uparrow u]}{(\lambda t)[u] \rightarrow^{x}_{lsa} (\lambda t')[u]} \quad (ls\_Lam)$$

Formalisation

Traduction des règles

On exprime la cloture réflexive et transitive de ces trois règles par la réduction $\rightarrow^*$

Formalisation

lemme 1.1

Soient t,u $\in \Lambda$ alors : Si u $\rightarrow$ u' alors t{u/x} $\rightarrow^*$ t{u'/x}

Formalisation

lemme 1.1

Soient t,u $\in \Lambda$ alors : Si u $\rightarrow$ u' alors t{u/x} $\rightarrow^*$ t{u'/x}

preuve par induction sur la structure de t

Formalisation

lemme 1.1

Soient t,u $\in \Lambda$ alors : Si u $\longrightarrow$ u' alors t{u/x} $\longrightarrow^*$ t{u'/x}

preuve par induction sur la structure de t

Problème

il est nécessaire de démontrer que la réduction est transparente par rapport à l'opérateur de lift :

$$u \longrightarrow u' \Rightarrow \uparrow u \longrightarrow \uparrow u'$$

Formalisation

lemme 1.1

Soient t,u $\in \Lambda$ alors : Si u $\rightarrow$ u' alors t{u/x} $\rightarrow^*$ t{u'/x}

preuve par induction sur la structure de t

Problème

il est nécessaire de démontrer que la réduction est transparente par rapport à l'opérateur de lift :

$$u \rightarrow u' \Rightarrow \uparrow u \rightarrow \uparrow u'$$

Solution :

$$\uparrow u = u\{i_1 + 1/i_1\} \cdots \{i_n + 1/i_n\}$$

où $i_1, \ldots, i_n$ sont les indices correspondants aux variables libres de u

Formalisation

lemme 1.2

Soient t,u $\in \Lambda$ alors : Si t $\longrightarrow$ t' alors t$\{u/x\}$ $\longrightarrow$ t'$\{u/x\}$.

preuve par induction sur la structure de $\longrightarrow$

Formalisation : représentation des termes fortement normalisants

Plusieurs méthodes peuvent être employées pour représenter cet ensemble.

Formalisation : représentation des termes fortement normalisants

Plusieurs méthodes peuvent être employées pour représenter cet ensemble.

Par le typage

Formalisation : représentation des termes fortement normalisants

Plusieurs méthodes peuvent être employées pour représenter cet ensemble.

Par le typage

Par un ensemble de termes bien fondés (Bibliothèque Coq wf)

Formalisation : représentation des termes fortement normalisants

Plusieurs méthodes peuvent être employées pour représenter cet ensemble.

- Par le typage
- Par un ensemble de termes bien fondés (Bibliothèque Coq wf)
- Par une caractérisation inductive des termes SN (méthode décrite Par Délia Kesner "Perpetuality Theorem")

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de $\mathbf{IE}$

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de $\mathbf{IE}$

On définit une fonction η : Terme $\longrightarrow$ nat.

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de $\mathbf{IE}$

On définit une fonction η : Terme $\longrightarrow$ nat.
η exprime la longueur de la plus longue réduction d'un terme t.

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de $\mathbf{IE}$

> On définit une fonction η : Terme $\rightarrow$ nat.
> η exprime la longueur de la plus longue réduction d'un terme t.

Cette axiomatisation repose sur 6 axiomes.

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de $\mathbf{IE}$

> On définit une fonction η : Terme $\rightarrow$ nat.
> η exprime la longueur de la plus longue réduction d'un terme t.

Cette axiomatisation repose sur 6 axiomes.

Les deux axiomes fondamentaux :

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de **IE**

> On définit une fonction η : Terme $\longrightarrow$ nat.
> η exprime la longueur de la plus longue réduction d'un terme t.

Cette axiomatisation repose sur 6 axiomes.

Les deux axiomes fondamentaux :

```
Axiom Ax1 : ∀ t : term,
        (∀ 't: term, t ⟶ t' ⇒ sn t' ) ⇒ sn t.
```

Formalisation : représentation des termes fortement normalisants

Nous avons choisi d'axiomatiser SN afin de pouvoir se concentrer essentiellement sur la preuve de **IE**

> On définit une fonction η : Terme $\rightarrow$ nat.
> η exprime la longueur de la plus longue réduction d'un terme t.

Cette axiomatisation repose sur 6 axiomes.

Les deux axiomes fondamentaux :

```
Axiom Ax1 : ∀ t : term,
        (∀ 't: term, t → t' ⇒ sn t' ) ⇒ sn t.

Axiom Ax2 : ∀ t 't: term,
        sn t ⇒ t → t'⇒  (sn t' ∧ η(t') < η(t)).
```

Formalisation : Les outils

Quelques lemmes utiles pour exprimer les occurrences de x dans un terme après substitution.

Formalisation : Les outils

Quelques lemmes utiles pour exprimer les occurrences de x dans un terme après substitution.

```
Lemma One_step_ls :
      forall t t' u, t [u] →ls t' [u] → |t'|0 < |t'|0.
```

```
Lemma lsa_red : ∀ t 't u :term,
      t →^0_u 't → 't{0:= (↑u)} = t {0:= (↑u)}.
```

Formalisation : Les outils

Quelques lemmes utiles pour exprimer les occurrences de x dans un terme après substitution.

```
Lemma One_step_ls :
      forall t t' u, t [u] →ls t' [u] → |t'|0 < |t'|0.
```

```
Lemma lsa_red : ∀ t 't u :term,
      t →^0_u 't → 't{0:= (↑u)} = t {0:= (↑u)}.
```

Un ordre lexicographique :

```
Inductive lex : (nat * nat) → (nat * nat) → Prop :=
 | lt_L : ∀ s1 s2, (fst s1) < (fst s2) → lex s1 s2
 | lt_G : ∀ s1 s2, (fst s1) = (fst s2) → (snd s1) < (snd s2) → lex s1 s2.
```

Formalisation

Propriété IE

$$\forall\ t\ u \in \Lambda, \forall\ \overline{v}\ , \forall\ x \in \mathbb{N},$$
$$u \in \mathcal{SN} \rightarrow t\ \{\ u\ /\ x\ \}\ \overline{v} \in \mathcal{SN} \rightarrow t[u/x]\ \overline{v} \in \mathcal{SN}\ .$$

Formalisation : $\mathbf{IE}$

Intuition

Pour démontrer sn(t[u/x]) , on utilise l'induction forte sur la paire lexicographique :

Formalisation : IE

Intuition

Pour démontrer sn(t[u/x]) , on utilise l'induction forte sur la paire lexicographique :

$$(\eta(t\{0 := u\}) + \eta(u), |t|_0).$$

Formalisation : IE

Intuition

Pour démontrer sn(t[u/x]) , on utilise l'induction forte sur la paire lexicographique :

$$(\eta(t\{0 := u\}) + \eta(u), |t|_0).$$

La présence de η dans cette paire est justifiée par l'axiome Ax2 qui énonce que η décroit strictement à chaque pas de réduction.

Formalisation : IE (Le schéma d'induction)

```
Axiom sn_ind :
∀ P : term → term → Prop,
(∀ t u:term,(∀ t' u':term,
(η(t' {0 := u' }) + η(u') , |t|_0) <lex (η(t {0 := u }) + η(u) → P t' u')
→ (P t u) ) → ∀ t u:term, P t u .
```

Conclusion

Les difficultés principales de cette formalisation

Conclusion

Les difficultés principales de cette formalisation

1. Les indices de De Bruijin.

Conclusion

Les difficultés principales de cette formalisation

1. Les indices de De Bruijin.
2. La traduction des règles originales dans Coq (principalement dû à l'usage de contextes)

Conclusion

Les difficultés principales de cette formalisation

1. Les indices de De Bruijin.
2. La traduction des règles originales dans Coq (principalement dû à l'usage de contextes)
3. La représentation de Sn

Conclusion

Moralité :

Conclusion

Moralité :

1. Les indices de de Bruijn, une difficultée certainement contournable :

Conclusion

Moralité :

1. Les indices de de Bruijn, une difficultée certainement contournable :
   1. Par l'usage d'une bibliothèque dédiée (DBlib,..)

Conclusion

Moralité :

1. Les indices de de Bruijn, une difficultée certainement contournable :
   1. Par l'usage d'une bibliothèque dédiée (DBlib,..)
   2. Par le choix d'un assistant de preuve permettant d'y faire face (Abella,...)

Conclusion

Moralité :

1. Les indices de de Bruijn, une difficultée certainement contournable :
   1. Par l'usage d'une bibliothèque dédiée (DBlib,..)
   2. Par le choix d'un assistant de preuve permettant d'y faire face (Abella,...)
2. L'axiomatisation de Sn : pas de difficulté majeure au premier abord

Conclusion

Idéalement :

1. Etendre $\mathbf{IE}$ aux listes de substitutions

Conclusion

Idéalement :

1. Etendre **IE** aux listes de substitutions
2. Coupler cette formalisation à celle de PSN déjà réalisée.

Fin

Merci Pour votre attention

Fin

Merci Pour votre attention

Des questions ?