Normalisation du système T avec opérateurs de contrôle Troisième version

Marquer Yoann

sous la tutelle de
Pierre Valarcher et Tristan Crolard

28 février 2012

Chapitre 1

Introduction

1.1 Commentaires

Je note ici les commentaires survenus dans le traitement de Tc2, afin de pouvoir reprendre plus facilement par la suite.

La stratégie d'évaluation.

Dans notre système par valeur à droite, $((\lambda x.t)v)$ $((\lambda y.u)w)$ serait évalué dans un ordre déterministe (d'ailleurs il faudrait le prouver pour tout terme de façon à ce que la preuve de normalisation se résume à une preuve de terminaison grâce à la traduction), reflété dans sa traduction :

let val $x_1 = (\lambda y.u^\star)w^*$ **in let val** $x_2 = (\lambda x.t^\star)v^*$ **in** $x_2 x_1$

Mais on voit bien ici que ce terme ne reflète que la stratégie par valeur, et non le choix à gauche ou à droite, puisqu'on peut réduire $(\lambda y.u^\star)w^*$ ou $(\lambda x.t^\star)v^*$ indifféremment. Pourtant on a besoin de ce choix dans le système source pour rompre l'indéterminisme de la réduction des opérateurs de contrôle.

Le rec.

Il serait plus joli de considérer le rec comme un symbole de constante jouant le rôle d'une valeur, et qui aurait la même règle de réduction

rec $0\ v\ \lambda xy.t \rightarrow_{rec} v$

rec $Sn\ v\ \lambda xy.t \rightarrow_{rec} (\lambda xy.t)\ n\ ($**rec** $n\ v\ \lambda xy.t)$

si possible dans les trois systèmes. Evidemment, il faudrait pour cela avoir un système T avec η qui normalise, ce qui fixerait la règle dans le langage cible.

L'avantage de traiter le rec comme une constante est que cela simplifie la traduction, car il est déjà prévu dans la disjonction de cas pour l'application.

Le problème avec les let est que cela sépare le rec de ses multiples arguments, donc il ne serait pas évalué. Le problème peut être contourné en η-expansant le rec à la traduction en un $\lambda nvt.$**rec** nvt qui ramasse ses arguments dans l'ordre d'évaluation avant de se résoudre. Ainsi **rec** $Sn\ v\ \lambda xy.t$ serait traduit en :

let $x_1 =$ **let** $x_2 = (\lambda nvt.$**rec** $nvt)Sn$ **in** $x_2 v^\diamond$ **in** $x_1(\lambda xy.t^\diamond)$ qui se réduit bien après des étapes de β_v-réductions et de let$_v$-réduction en **rec** $Sn\ v^\diamond\ \lambda xy.t^\diamond$.

Le soucis est que **rec** $Sn\ v^\diamond\ \lambda xy.t^\diamond$ se réduit en $(\lambda xy.t^\diamond)\ n\ (\mathbf{rec}\ n\ v^\diamond\ \lambda xy.t^\diamond)$ et pas en la traduction de $(\lambda xy.t)\ n\ (\mathbf{rec}\ n\ v\ \lambda xy.t)$, c'est à dire :

$\mathbf{let}\ x_1 = \mathbf{let}\ x_2 = \mathbf{let}\ x_3 = (\lambda nvt.\mathbf{rec}\ nvt)n\ \mathbf{in}\ x_3 v^\diamond\ \mathbf{in}\ x_2(\lambda xy.t^\diamond)\ \mathbf{in}\ \mathbf{let}\ x_4 = (\lambda xy.t^\diamond)\ n\ \mathbf{in}\ x_4 x_1$

Donc pour avoir des chances de marcher, il faudrait que le rec soit traduit en un $\lambda nvt.\mathbf{recv}\ nvt$, où **recv** est définir à partir du **rec** tel que $\mathbf{recv}\ Sn\ v^\diamond\ \lambda xy.t^\diamond$ se réduise par un nombre fini non nul d'étapes en $\mathbf{let}\ x_1 = \mathbf{let}\ x_2 = \mathbf{let}\ x_3 = (\lambda nvt.\mathbf{recv}\ nvt)n\ \mathbf{in}\ x_3 v^\diamond\ \mathbf{in}\ x_2(\lambda xy.t^\diamond)\ \mathbf{in}\ \mathbf{let}\ x_4 = (\lambda xy.t^\diamond)\ n\ \mathbf{in}\ x_4 x_1$.

Pour continuer il faut donc trouver le papier permettant la normalisation du système T avec η-réduction, et trouver comment écrire ce **recv**.

Idée presque bonne :

Si on écrit $\mathbf{recv} := \lambda nvt.\mathbf{rec}\ n\ v\ (\lambda x'y'.\Theta[y', x', v, t])$ avec $\Theta[z, n, v, t] = \mathbf{let}\ x_1 = \mathbf{let}\ x_2 = \mathbf{let}\ x_3 = zn\ \mathbf{in}\ x_3 v\ \mathbf{in}\ x_2 t\ \mathbf{in}\ \mathbf{let}\ x_4 = tn\ \mathbf{in}\ x_4 x_1$ on a que la traduction de $(\lambda xy.t)\ n\ (\mathbf{rec}\ n\ v\ \lambda xy.t)$ est $\Theta[\mathbf{recv}, n, v^\diamond, \lambda xy.t^\diamond]$, alors que la traduction de $\mathbf{rec}\ Sn\ v\ \lambda xy.t$ se réduit en $\Theta[\mathbf{rec}\ n\ v^\diamond\ (\lambda x'y'.\Theta[y', x', v^\diamond, \lambda xy.t^\diamond]), n, v^\diamond, \lambda xy.t^\diamond]$ qui ressemble beaucoup à la forme souhaité. D'ailleurs à ceci près qu'il s'agit d'un T terme et non d'un Tn terme, il se comporte de la façon souhaitée. (La règle de rec donnée ne peut s'appliquer dans Tn, donc là en fait je suis plutôt en train de réfléchir sur $\star$ et pas $\diamond$.)

Les opérateurs de contrôle.

Il reste à définir le comportement de callcc et throw, le papier A Selective CPS Transformation de Nielsen donnant une bonne base. Il utilise des constantes pour chaque contexte C[.] que je note α_C, constantes comptant naturellement comme des valeurs dans notre système de calcul. La question est alors des traduire, par exemple avec le terme $\lambda x.C[x]$ (éventuellement avec un abort/throw devant le contexte, voir le A de Felleisen). Le fait que callcc et throw soient des constantes du langage est d'ailleurs une raison de plus pour intégrer rec comme une constante et non comme un constructeur.

Les prouveurs.

Le fait que mon travail soit devenu assez technique pousse à faire des démonstrations par machine, même si j'arrive à trouver des choses simplement exprimables avec les constantes rec, callcc et throw.

1.2 Ancienne intro

Ceci est la traduction avec les entiers et la récursion du système T, mais pas les opérateurs de contrôle. Par nécessité de la traduction monadique, le système T présenté ici est muni de l'η-réduction.

Pour que le rec passe à la simulation j'ai dû changer la traduction de $\diamond$ pour une traduction avec plus de cas mais qui introduit moins de let, et aux bons endroits (notamment les redex sont traduits uniformément par des let).

La fin est un copier-coller de la première partie, la réécriture est en cours.

A cause du rec la traduction ne se fait pas uniquement en regardant les catégories syntaxiques de termes et de valeur (puisqu'on traduit directement un redex en un let), ce que je trouve insatisfaisant.

Cependant, pourquoi l'ai-je fait ?

La règle de récurrence dans le système T est :

$\mathbf{rec}(Sn, u, \lambda x.\lambda y.t) \rightarrow_{rec} t[n/x, \mathbf{rec}(n, u, \lambda x.\lambda y.t)/y]$

Nous voulons que les termes traduits soient de la forme :

$\mathbf{rec}(v_1^{\bullet}, v_2^{\circ}, \lambda x.\lambda r.\mathbf{let}\ \mathbf{val}\ y = r\ \mathbf{in}\ t^{\circ})$

Et alors dans T ils se comportent de la façon suivante :

$\mathbf{rec}(Sn, \mathbf{val}\ v^{\bullet}, \lambda x.\lambda r.\mathbf{let}\ \mathbf{val}\ y = r\ \mathbf{in}\ t^{\circ})$

$\rightarrow_{rec} \mathbf{let}\ \mathbf{val}\ y = \mathbf{rec}(n, \mathbf{val}\ v^{\bullet}, \lambda x.\lambda r.\mathbf{let}\ \mathbf{val}\ y = r\ \mathbf{in}\ t^{\circ})\ \mathbf{in}\ t^{\circ}[n/x]$

Pour avoir ces termes traduits on s'attend naturellement à avoir des termes de la forme $\mathbf{rec}(v_1, v_2, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t)$ dans T_n.

Or le terme $\mathbf{let}\ \mathbf{val}\ y = \mathbf{rec}(n, \mathbf{val}\ v^{\bullet}, \lambda x.\lambda r.\mathbf{let}\ \mathbf{val}\ y = r\ \mathbf{in}\ t^{\circ})\ \mathbf{in}\ t^{\circ}[n/x]$ est la traduction du terme $\mathbf{let}\ y = \mathbf{rec}(n, v, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t)\ \mathbf{in}\ t[n/x]$.

Il faudrait donc que $\mathbf{rec}(Sn, v, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t)$ se réduise en $\mathbf{let}\ y = \mathbf{rec}(n, v, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t)\ \mathbf{in}\ t[n/x]$, ce qui serait une règle de rec a priori naturelle.

Pour forcer l'évaluer du rec avant de la passer en argument de t, la règle de rec dans le système T_v est : $\mathbf{rec}(Sn, v, \lambda x.\lambda y.t) \rightarrow_{rec} (\lambda y.t[n/x])\ \mathbf{rec}(n, v, \lambda x.\lambda y.t)$.

Avec la traduction précédente, $\mathbf{rec}(Sn, v, \lambda x.\lambda y.t)$ serait traduit en $\mathbf{rec}(Sn, v^{\diamond}, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^{\diamond})$, alors que $(\lambda y.t[n/x])\ \mathbf{rec}(n, v, \lambda x.\lambda y.t)$ aurait été traduit en $\mathbf{let}\ z = \mathbf{rec}(n, v^{\diamond}, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^{\diamond})\ \mathbf{in}\ (\lambda y.t^{\diamond}[n/x])z$.

Or en appliquant la règle du rec dans T_n, $\mathbf{rec}(Sn, v^{\diamond}, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^{\diamond})$ se réduirait en $\mathbf{let}\ y = \mathbf{rec}(n, v^{\diamond}, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^{\diamond})\ \mathbf{in}\ t^{\diamond}[n/x]$ qui ne se réduit pas en $\mathbf{let}\ z = \mathbf{rec}(n, v^{\diamond}, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^{\diamond})\ \mathbf{in}\ (\lambda y.t^{\diamond}[n/x])z$.

C'est donc pour éviter cette situation que j'ai traduit les redex directement, ce qui permet de faire de la simulation presque étape par étape sans avoir besoin d'une β-réduction dans T_n puisque la règle du let suffit.

Table des matières

1 Introduction **1**
- 1.1 Commentaires 1
- 1.2 Ancienne intro 2

2 Les systèmes utilisés **5**
- 2.1 Le système par valeur T_v 5
- 2.2 Le système nommé T_n 5
- 2.3 Le système T (avec η-réduction) 6
- 2.4 Résultats généraux sur les contextes 7

3 Traduction $\diamond$ de $\mathbf{T}_v$ dans $\mathbf{T}_n$ **8**
- 3.1 Traduction des termes 8
- 3.2 Traduction des contextes 11
- 3.3 Simulation de la réduction 13

4 Traduction $\circ$ de $\mathbf{T}_n$ dans T **14**
- 4.1 Abréviations 14
- 4.2 Traduction des termes 14
- 4.3 Traduction des contextes 16
- 4.4 Simulation de la réduction 16

5 Traduction $\star$ de $\mathbf{T}_v$ dans T **18**
- 5.1 Normalisation 18
- 5.2 Complexité 18

Chapitre 2

Les systèmes utilisés

2.1 Le système par valeur $\mathbf{T}_v$

Soit un ensemble dénombrable {x, y, z ... } de variables, et trois symboles : 0, S et **rec**.

Définition 2.1.1.
$$n := 0 \mid (S)n$$
$$v := x \mid n \mid \lambda x.t$$
$$t := v \mid (t_1)t_2 \mid \boldsymbol{rec}(t_1, t_2, \lambda x.\lambda y.t_3)$$

Conventions usuelles pour le renommage de variable, le parenthésage et la substitution. (définition d'un succ ?)

Définition 2.1.2.
$$(\lambda x.t)v \rightarrow_{\beta_v} t[v/x]$$
$$\boldsymbol{rec}(0, v, \lambda x.\lambda y.t) \rightarrow_{rec} v$$
$$\boldsymbol{rec}(Sn, v, \lambda x.\lambda y.t) \rightarrow_{rec} (\lambda y.t[n/x])\ \boldsymbol{rec}(n, v, \lambda x.\lambda y.t)$$

Notons $\rightarrow$ la réunion de $\rightarrow_{\beta_v}$ et $\rightarrow_{rec}$.

Définition 2.1.3.
$$C[.] := .\ \mid tC[.] \mid C[.]v \mid S(C[.])\ ?$$
$$\mid \boldsymbol{rec}(C[.], t_1, \lambda x.\lambda y.t_2) \mid \boldsymbol{rec}(v, C[.], \lambda x.\lambda y.t)$$

Définition 2.1.4. *Règle de passage au contexte :*
$$\frac{t \rightarrow u}{C[t] \mapsto C[u]}$$

Un seul chemin de réduction ? Si oui pour avoir la normalisation il suffit d'avoir des réductions finies (garanties par la simulation).

2.2 Le système nommé $\mathbf{T}_n$

Soit un ensemble dénombrable {x, y, z ... } de variables.

Définition 2.2.1. $n := 0 \mid (S)n$
$v := x \mid n \mid \lambda x.t$
$t := v \mid (v_1)v_2 \mid \textbf{\textit{let}}\ x = t_1\ \textbf{\textit{in}}\ t_2$
$\mid \textbf{\textit{rec}}(v_1, v_2, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t)$

Conventions usuelles pour le renommage de variable, le parenthésage et la substitution. (définition d'un succ ?)

Définition 2.2.2. $\textbf{\textit{let}}\ x = v\ \textbf{\textit{in}}\ t \to_{let} t[v/x]$
$\textbf{\textit{rec}}(0, v, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t) \to_{rec} v$
$\textbf{\textit{rec}}(Sn, v, \lambda x.\lambda r.\textbf{\textit{let}}\, y = r\ \textbf{\textit{in}}\, t) \to_{rec} \textbf{\textit{let}}\, y = \textbf{\textit{rec}}(n, v, \lambda x.\lambda r.\textbf{\textit{let}}\, y = r\ \textbf{\textit{in}}\, t)\ \textbf{\textit{in}}\, t[n/x]$
(Pas de β-réduction)

Notons $\to$ la réunion de $\to_{let}$ et $\to_{rec}$, $\to^+$ sa clôture transitive et $\to^*$ sa clôture transitive et réflexive.

Définition 2.2.3. $C[.] := .\ \mid \textbf{\textit{let}}\ x = C[.]\ \textbf{\textit{in}}\ t$

Définition 2.2.4. *Règle de passage au contexte :*
$$\frac{t \to u}{C[t] \mapsto C[u]}$$

2.3 Le système T (avec η-réduction)

Soit un ensemble dénombrable {x, y, z ... } de variables.

Définition 2.3.1. $n := 0 \mid (S)n$
$t := x \mid n \mid (t_1)t_2 \mid \textbf{\textit{rec}}(t_1, t_2, \lambda x.\lambda y.t_3)$

Conventions usuelles pour le renommage de variable, le parenthésage et la substitution. (définition d'un succ ?)

Définition 2.3.2. $(\lambda x.t)u \to_\beta t[u/x]$
$\lambda x.tx \to_\eta t$ *(si x n'est pas libre dans t)*
$\textbf{\textit{rec}}(0, u, \lambda x.\lambda y.t) \to_{rec} u$
$\textbf{\textit{rec}}(Sn, u, \lambda x.\lambda y.t) \to_{rec} t[n/x, \textbf{\textit{rec}}(n, u, \lambda x.\lambda y.t)/y]$

Notons $\to$ la réunion de $\to_\beta$, $\to_\eta$ et $\to_{rec}$, $\to^+$ sa clôture transitive et $\to^*$ sa clôture transitive et réflexive.

Définition 2.3.3. $C[.] := .\ \mid tC[.] \mid C[.]t \mid \lambda x.C[.] \mid S(C[.])?$
$\mid \textbf{\textit{rec}}(C[.], u, \lambda x.\lambda y.t) \mid \textbf{\textit{rec}}(u, C[.], \lambda x.\lambda y.t) \mid \textbf{\textit{rec}}(t, u, \lambda x.\lambda y.C[.])$

Définition 2.3.4. *Règle de passage au contexte :*
$$\frac{t \to u}{C[t] \mapsto C[u]}$$

Admis : normalisation (papier pour l'η-réduction ?).

2.4 Résultats généraux sur les contextes

Notons $\mapsto^+$ la clôture transitive de $\mapsto$.

Lemme 2.4.1. *Si $t \to^+ u$ alors $C[t] \mapsto^+ C[u]$.*

Démonstration. Comme $t \to^+ u$ il existe $k > 0$ tel que $t = t_0 \to t_1 \to ... \to t_k = u$. Pour tout $0 \leq i < k$ comme $t_i \to t_{i+1}$ nous avons par passage au contexte C que $C[t_i] \mapsto C[t_{i+1}]$, donc $C[t] = C[t_0] \mapsto C[t_1] \mapsto ... \mapsto C[t_k] = C[u]$, c'est à dire $C[t] \mapsto^+ C[u]$. □

Lemme 2.4.2. *Si $t \mapsto u$ alors il existe un unique contexte C[.] et deux termes t' et u' tels que $t = C[t']$, $u = C[u']$ et $t' \to u'$.*

Démonstration. L'existence vient par définition de la règle de passage au contexte.

Pour l'unicité supposons que d'une part il existe un contexte C[.] et deux termes t' et u' tels que $t = C[t']$, $u = C[u']$ et $t' \to u'$, et d'autre part qu'il existe un contexte C'[.] et deux termes t" et u" tels que $t = C'[t'']$, $u = C'[u'']$ et $t'' \to u''$.

A finir. □

Chapitre 3

Traduction $\diamond$ de T$_v$ dans T$_n$

3.1 Traduction des termes

Définition 3.1.1.

$$\begin{aligned}
0^\diamond &:= 0\\
(Sn)^\diamond &:= Sn^\diamond\\
x^\diamond &:= x\\
(\lambda x.t)^\diamond &:= \lambda x.t^\diamond
\end{aligned}$$

$$\begin{aligned}
((\lambda x.t_1)t_2)^\diamond &:= \textbf{\textit{let}}\ x = t_2^\diamond\ \textbf{\textit{in}}\ t_1^\diamond \quad \textit{(attention : voir intro)}\\
(v_1 v_2)^\diamond &:= v_1^\diamond v_2^\diamond\\
(tv)^\diamond &:= \textbf{\textit{let}}\ x = t^\diamond\ \textbf{\textit{in}}\ x v^\diamond\\
(vt)^\diamond &:= \textbf{\textit{let}}\ x = t^\diamond\ \textbf{\textit{in}}\ v^\diamond x\\
(t_1 t_2)^\diamond &:= \textbf{\textit{let}}\ y = t_2^\diamond\ \textbf{\textit{in}}\ \textbf{\textit{let}}\ x = t_1^\diamond\ \textbf{\textit{in}}\ xy
\end{aligned}$$

$$\begin{aligned}
\textbf{\textit{rec}}(v_1, v_2, \lambda x.\lambda y.t)^\diamond &:= \textbf{\textit{rec}}(v_1^\diamond, v_2^\diamond, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t^\diamond)\\
\textbf{\textit{rec}}(v, t_1, \lambda x.\lambda y.t_2)^\diamond &:= \textbf{\textit{let}}\ z = t_1^\diamond\ \textbf{\textit{in}}\ \textbf{\textit{rec}}(v^\diamond, z, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t_2^\diamond)\\
\textbf{\textit{rec}}(t_1, v, \lambda x.\lambda y.t_2)^\diamond &:= \textbf{\textit{let}}\ z = t_1^\diamond\ \textbf{\textit{in}}\ \textbf{\textit{rec}}(z, v^\diamond, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t_2^\diamond)\\
\textbf{\textit{rec}}(t_1, t_2, \lambda x.\lambda y.t_3)^\diamond &:= \textbf{\textit{let}}\ z_1 = t_1^\diamond\ \textbf{\textit{in}}\ \textbf{\textit{let}}\ z_2 = t_2^\diamond\ \textbf{\textit{in}}\\
&\qquad\qquad \textbf{\textit{rec}}(z_1, z_2, \lambda x.\lambda r.\textbf{\textit{let}}\ y = r\ \textbf{\textit{in}}\ t_3^\diamond)
\end{aligned}$$

Traduction limitant les redex administratifs (on a presque une simulation étape par étape).

Lemme 3.1.2. *Pour tout T_v-terme t, $t^\diamond$ est un T_n-terme tel que si t est un T_v-entier si alors $t^\diamond$ est un T_n-entier, et si t est une T_v-valeur alors $t^\diamond$ est une T_n-valeur.*

Quand je suis dans un système je considère en fait que je pourrais utiliser l'ensemble des mots formés par la syntaxe, donc les termes sont pour moi une catégorie syntaxique à part entière, et pas l'ensemble de tous les individus.

Démonstration. La preuve se fait par induction sur t.

Si t est 0 alors $t^\diamond$ est 0, donc un T$_n$-entier.

Si t est un successeur Sn alors $t^\diamond = Sn^\diamond$, et par induction $n^\diamond$ est un T_n-entier, donc $t^\diamond$ est un T_n-entier.

Si t est une variable alors $t^\diamond$ est une variable, donc une T_n-valeur.

Si t est une abstraction $\lambda x.t'$ alors $t^\diamond = \lambda x.t'^\diamond$, or par induction $t'^\diamond$ est un T_n-terme donc $t^\diamond$ est une abstraction, et donc une T_n-valeur.

Si t est un redex $(\lambda x.t_1)t_2$ alors $t^\diamond = \textbf{let } x = t_2^\diamond \textbf{ in } t_1^\diamond$, or par induction $t_1^\diamond$ et $t_2^\diamond$ sont des T_n-termes, donc $t^\diamond$ est un T_n-terme.

Si t est une application de deux valeurs v_1v_2 alors $t^\diamond = v_1^\diamond v_2^\diamond$, or par induction $v_1^\diamond$ et $v_2^\diamond$ sont des T_n-valeurs, donc $t^\diamond$ est un T_n-terme.

Si t est une application t'v alors $t^\diamond = \textbf{let } x = t'^\diamond \textbf{ in } xv^\diamond$, or par induction $t'^\diamond$ est un T_n-terme et $v^\diamond$ est une T_n-valeur, donc $t^\diamond$ est un T_n-terme.

Si t est une application vt' alors $t^\diamond = \textbf{let } x = t'^\diamond \textbf{ in } v^\diamond x$, or par induction $t'^\diamond$ est un T_n-terme et $v^\diamond$ est une T_n-valeur, donc $t^\diamond$ est un T_n-terme.

Si t est une application t_1t_2 alors $t^\diamond = \textbf{let } y = t_2^\diamond \textbf{ in let } x = t_1^\diamond \textbf{ in } xy$, or par induction $t_1^\diamond$ et $t_2^\diamond$ sont des T_n-termes, donc $t^\diamond$ est un T_n-terme.

Si t est un $\textbf{rec}(v_1, v_2, \lambda x.\lambda y.t)$ alors $t^\diamond = \textbf{rec}(v_1^\diamond, v_2^\diamond, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t^\diamond)$, or par induction $v_1^\diamond$ et $v_2^\diamond$ sont des T_n-valeurs et $t'^\diamond$ est un T_n-terme, donc $t^\diamond$ est un T_n-terme.

Si t est un $\textbf{rec}(v, t_1, \lambda x.\lambda y.t_2)$ alors $t^\diamond = \textbf{let } z = t_1^\diamond \textbf{ in rec}(v^\diamond, z, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t_2^\diamond)$, or par induction $t_1^\diamond$ et $t_2^\diamond$ sont des T_n-termes et $v^\diamond$ est une T_n-valeur, donc $t^\diamond$ est un T_n-terme.

Si t est un $\textbf{rec}(t_1, v, \lambda x.\lambda y.t_2)$ alors $t^\diamond = \textbf{let } z = t_1^\diamond \textbf{ in rec}(z, v^\diamond, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t_2^\diamond)$, or par induction $t_1^\diamond$ et $t_2^\diamond$ sont des T_n-termes et $v^\diamond$ est une T_n-valeur, donc $t^\diamond$ est un T_n-terme.

Si t est un $\textbf{rec}(t_1, t_2, \lambda x.\lambda y.t_3)$ alors $t^\diamond = \textbf{let } z_1 = t_1^\diamond \textbf{ in let } z_2 = t_2^\diamond \textbf{ in}$ $\textbf{rec}(z_1, z_2, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t_3^\diamond)$, or par induction $t_1^\diamond$, $t_2^\diamond$ et $t_3^\diamond$ sont des T_n-termes donc $t^\diamond$ est un T_n-terme. □

Lemme 3.1.3. *Dans T_v, $t[v/x]$ est une valeur si et seulement si t est une valeur.*

Démonstration. Si t est une valeur alors t est un entier, une variable ou une abstraction.

Si t est un entier, t est clos donc $t[v/x] = t$ est une valeur.

Si t est une variable, ou bien $t = y$ et alors $t[v/x] = y$ qui est une variable et donc une valeur, ou bien $t = x$ et alors $t[v/x] = v$ qui est une valeur.

Si t est une abstraction $\lambda y.t'$, alors $t[v/x] = \lambda y.t'[v/x]$, qui est une abstraction et donc une valeur.

Réciproquement si t n'est pas une valeur alors t est une application ou un rec.

Si t est une applications t_1t_2, alors $t[v/x] = t_1[v/x]t_2[v/x]$ est une application donc n'est pas une valeur.

Si t est un $\textbf{rec}(t_1, t_2, \lambda x.\lambda y.t_3)$ alors $t[v/x] = \textbf{rec}(t_1[v/x], t_2[v/x], \lambda x'.\lambda y'.t_3[v/x])$ qui est un rec et donc pas une valeur. □

Lemme 3.1.4. $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$

Démonstration. La preuve se fait par induction sur t.

Si t est 0 alors $(t[v/x])^\diamond = 0^\diamond = 0$ et $t^\diamond[v^\diamond/x] = 0[v^\diamond/x] = 0$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un successeur Sn alors $(t[v/x])^\diamond = (Sn)^\diamond = Sn$ et $t^\diamond[v^\diamond/x] = (Sn)[v^\diamond/x] = Sn$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une variable y alors $(t[v/x])^\diamond = y^\diamond = y$ et $t^\diamond[v^\diamond/x] = y[v^\diamond/x] = y$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est x alors $(t[v/x])^\diamond = v^\diamond$ et $t^\diamond[v^\diamond/x] = x[v^\diamond/x] = v^\diamond$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une abstraction $\lambda x.t'$ alors $(t[v/x])^\diamond = (\lambda x.t'[v/x])^\diamond = \lambda x.(t'[v/x])^\diamond$ et $t^\diamond[v^\diamond/x] = (\lambda x.t'^\diamond)[v^\diamond/x] = \lambda x.t'^\diamond[v^\diamond/x]$, or par induction $(t'[v/x])^\diamond = t'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un redex $(\lambda y.t_1)t_2$ alors $(t[v/x])^\diamond = ((\lambda y.t_1[v/x])t_2[v/x])^\diamond = \mathbf{let}\ y = (t_2[v/x])^\diamond\ \mathbf{in}\ (t_1[v/x])^\diamond$ et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ y = t_2^\diamond\ \mathbf{in}\ t_1^\diamond)[v^\diamond/x] = \mathbf{let}\ y = t_2^\diamond[v^\diamond/x]\ \mathbf{in}\ t_1^\diamond[v^\diamond/x]$, or par induction $(t_1[v/x])^\diamond = t_1^\diamond[v^\diamond/x]$ et $(t_2[v/x])^\diamond = t_2^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une application de deux valeurs v_1v_2 alors $(t[v/x])^\diamond = (v_1[v/x]v_2[v/x])^\diamond = (v_1[v/x])^\diamond(v_2[v/x])^\diamond$ et $t^\diamond[v^\diamond/x] = (v_1^\diamond v_2^\diamond)[v^\diamond/x] = v_1^\diamond[v^\diamond/x]v_2^\diamond[v^\diamond/x]$, or par induction $(v_1[v/x])^\diamond = v_1^\diamond[v^\diamond/x]$ et $(v_2[v/x])^\diamond = v_2^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une application t'v' alors $(t[v/x])^\diamond = (t'[v/x]v'[v/x])^\diamond = \mathbf{let}\ y = (t'[v/x])^\diamond\ \mathbf{in}\ y(v'[v/x])^\diamond$ et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ y = t'^\diamond\ \mathbf{in}\ yv'^\diamond)[v^\diamond/x] = \mathbf{let}\ y = t'^\diamond[v^\diamond/x]\ \mathbf{in}\ yv'^\diamond[v^\diamond/x]$, or par induction $(t'[v/x])^\diamond = t'^\diamond[v^\diamond/x]$ et $(v'[v/x])^\diamond = v'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une application v't' alors $(t[v/x])^\diamond = (v'[v/x]t'[v/x])^\diamond = \mathbf{let}\ y = (t'[v/x])^\diamond\ \mathbf{in}\ (v'[v/x])^\diamond y$ et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ y = t'^\diamond\ \mathbf{in}\ v'^\diamond y)[v^\diamond/x] = \mathbf{let}\ y = t'^\diamond[v^\diamond/x]\ \mathbf{in}\ v'^\diamond[v^\diamond/x]y$, or par induction $(t'[v/x])^\diamond = t'^\diamond[v^\diamond/x]$ et $(v'[v/x])^\diamond = v'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est une application t_1t_2 alors $(t[v/x])^\diamond = (t_1[v/x]t_2[v/x])^\diamond = (\mathbf{let}\ z = (t_2[v/x])^\diamond\ \mathbf{in}\ \mathbf{let}\ y = (t_1[v/x])^\diamond\ \mathbf{in}\ yz)^\diamond$ et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ z = t_2^\diamond\ \mathbf{in}\ \mathbf{let}\ y = t_1^\diamond\ \mathbf{in}\ yz)[v^\diamond/x] = \mathbf{let}\ z = t_2^\diamond[v^\diamond/x]\ \mathbf{in}\ \mathbf{let}\ y = t_1^\diamond[v^\diamond/x]\ \mathbf{in}\ yz$, or par induction $(t_1[v/x])^\diamond = t_1^\diamond[v^\diamond/x]$ et $(t_2[v/x])^\diamond = t_2^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un $\mathbf{rec}(v_1, v_2, \lambda x'.\lambda y.t')$ alors $(t[v/x])^\diamond = \mathbf{rec}(v_1[v/x], v_2[v/x], \lambda x'.\lambda y.\ t'[v/x])^\diamond = \mathbf{rec}((v_1[v/x])^\diamond, (v_2[v/x])^\diamond, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ (t'[v/x])^\diamond)$ et $t^\diamond[v^\diamond/x] = \mathbf{rec}(v_1^\diamond, v_2^\diamond, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^\diamond)[v^\diamond/x] = \mathbf{rec}(v_1^\diamond[v^\diamond/x], v_2^\diamond[v^\diamond/x], \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t^\diamond[v^\diamond/x])$, or par induction $(v_1[v/x])^\diamond = v_1^\diamond[v^\diamond/x]$, $(v_2[v/x])^\diamond = v_2^\diamond[v^\diamond/x]$ et $(t'[v/x])^\diamond = t'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un $\mathbf{rec}(v', t_1, \lambda x'.\lambda y.t_2)$ alors $(t[v/x])^\diamond = (\mathbf{rec}(v'[v/x], t_1[v/x], \lambda x'.\lambda y.\ t_2[v/x]))^\diamond = \mathbf{let}\ z = (t_1[v/x])^\diamond\ \mathbf{in}\ \mathbf{rec}((v'[v/x])^\diamond, z, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ (t_2[v/x])^\diamond)$, et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ z = t_1^\diamond\ \mathbf{in}\ \mathbf{rec}(v'^\diamond, z, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_2^\diamond))[v^\diamond/x] = \mathbf{let}\ z = t_1^\diamond[v^\diamond/x]\ \mathbf{in}\ \mathbf{rec}(v'^\diamond[v^\diamond/x], z, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_2^\diamond[v^\diamond/x])$, or par induction $(t_1[v/x])^\diamond = t_1^\diamond[v^\diamond/x]$, $(t_2[v/x])^\diamond = t_2^\diamond[v^\diamond/x]$ et $(v'[v/x])^\diamond = v'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un $\mathbf{rec}(t_1, v', \lambda x'.\lambda y.t_2)$ alors $(t[v/x])^\diamond = (\mathbf{rec}(t_1[v/x], v'[v/x], \lambda x'.\lambda y.\ t_2[v/x]))^\diamond = \mathbf{let}\ z = (t_1[v/x])^\diamond\ \mathbf{in}\ \mathbf{rec}(z, (v'[v/x])^\diamond, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ (t_2[v/x])^\diamond)$, et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ z = t_1^\diamond\ \mathbf{in}\ \mathbf{rec}(z, v'^\diamond, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_2^\diamond))[v^\diamond/x] = \mathbf{let}\ z = t_1^\diamond[v^\diamond/x]\ \mathbf{in}\ \mathbf{rec}(z, v'^\diamond[v^\diamond/x], \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_2^\diamond[v^\diamond/x])$, or par induction $(t_1[v/x])^\diamond = t_1^\diamond[v^\diamond/x]$, $(t_2[v/x])^\diamond = t_2^\diamond[v^\diamond/x]$ et $(v'[v/x])^\diamond = v'^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

Si t est un $\mathbf{rec}(t_1, t_2, \lambda x.\lambda y.t_3)$ alors $(t[v/x])^\diamond = \mathbf{rec}(t_1[v/x], t_2[v/x], \lambda x'.\lambda y.\ t_3[v/x])^\diamond = \mathbf{let}\ z_1 = (t_1[v/x])^\diamond\ \mathbf{in}\ \mathbf{let}\ z_2 = (t_2[v/x])^\diamond\ \mathbf{in}\ \mathbf{rec}(z_1, z_2, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ (t_3[v/x])^\diamond)$, et $t^\diamond[v^\diamond/x] = (\mathbf{let}\ z_1 = t_1^\diamond\ \mathbf{in}\ \mathbf{let}\ z_2 = t_2^\diamond\ \mathbf{in}\ \mathbf{rec}(z_1, z_2, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_3^\diamond))[v^\diamond/x] = \mathbf{let}\ z_1 = t_1^\diamond[v^\diamond/x]\ \mathbf{in}\ \mathbf{let}\ z_2 = t_2^\diamond[v^\diamond/x]\ \mathbf{in}\ \mathbf{rec}(z_1, z_2, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t_3^\diamond[v^\diamond/x])$, or par induction $(t_1[v/x])^\diamond = t_1^\diamond[v^\diamond/x]$, $(t_2[v/x])^\diamond = t_2^\diamond[v^\diamond/x]$ et $(t_3[v/x])^\diamond = t_3^\diamond[v^\diamond/x]$, donc $(t[v/x])^\diamond = t^\diamond[v^\diamond/x]$.

□

3.2 Traduction des contextes

Définition 3.2.1. $.^\diamond := .$

$$
\begin{aligned}
((\lambda x.t)C[.])^\diamond &:= \textbf{\textit{let}}\ x = (C[.])^\diamond\ \textbf{\textit{in}}\ t^\diamond \\
(vC[.])^\diamond &:= \textbf{\textit{let}}\ x = (C[.])^\diamond\ \textbf{\textit{in}}\ v^\diamond x \\
(C[.]v)^\diamond &:= \textbf{\textit{let}}\ x = (C[.])^\diamond\ \textbf{\textit{in}}\ x v^\diamond \\
(tC[.])^\diamond &:= \textbf{\textit{let}}\ y = (C[.])^\diamond\ \textbf{\textit{in}}\ \textbf{\textit{let}}\ x = t^\diamond\ \textbf{\textit{in}}\ xy
\end{aligned}
$$

Notons $C^\diamond[.]$ la traduction $(C[.])^\diamond$ du contexte $C[.]$, et $C[t]^\diamond$ la traduction $(C[t])^\diamond$ du terme $C[t]$.

Lemme 3.2.2. *Pour tout T_v-contexte $C[.]$, $C^\diamond[.]$ est un T_n-contexte.*

Démonstration. La preuve se fait par induction sur C[.].

Si $C[.] = .$ alors $C^\diamond[.] = .$ donc est un T_n-contexte.

Sinon $C^\diamond[.] = \mathbf{let}\ x = C'^\diamond[.]\ \mathbf{in}\ t$ (où le terme t dépend de la forme de C[.]), or par induction $C'^\diamond[.]$ est un T_n-contexte donc $C^\diamond[.]$ est un T_n-contexte. □

Lemme 3.2.3. *$C^\diamond[t^\diamond] \mapsto^* C[t]^\diamond$ (en au plus une étape).*

De plus si t n'est pas une valeur alors $C^\diamond[t^\diamond] = C[t]^\diamond$.

Démonstration. La preuve se fait par induction sur C[.], en distinguant comme cas de base les cas où C[.] est de profondeur 0 ou 1.

La notion de profondeur devrait-elle être développée davantage ?

Si $C[.] = .$ alors $C^\diamond[.] = .$ donc $C^\diamond[t^\diamond] = t^\diamond$.

De plus $C[t] = t$ d'où $C[t]^\diamond = t^\diamond$.

D'où $C^\diamond[t^\diamond] = C[t]^\diamond$.

Si $C[.] = v.$ alors $C^\diamond[.] = \mathbf{let}\ x = .\ \mathbf{in}\ v^\diamond x$, d'où $C^\diamond[t^\diamond] = \mathbf{let}\ x = t^\diamond\ \mathbf{in}\ v^\diamond x$.

De plus $C[t]^\diamond = (vt)^\diamond = \mathbf{let}\ x = t^\diamond\ \mathbf{in}\ v^\diamond x$ si t n'est pas une valeur, donc $C^\diamond[t^\diamond] = C[t]^\diamond$. Sinon $C[t]^\diamond = v^\diamond t^\diamond$, et alors $C^\diamond[t^\diamond] = \mathbf{let}\ x = t^\diamond\ \mathbf{in}\ v^\diamond x \to v^\diamond t^\diamond = C[t]^\diamond$.

Si $C[.] = .v$ alors $C^\diamond[.] = \textbf{let } x = . \textbf{ in } xv^\diamond$, d'où $C^\diamond[t^\diamond] = \textbf{let } x = t^\diamond \textbf{ in } xv^\diamond$.

De plus $C[t]^\diamond = (tv)^\diamond = \textbf{let } x = t^\diamond \textbf{ in } xv^\diamond$ si t n'est pas une valeur, donc $C^\diamond[t^\diamond] = C[t]^\diamond$. Sinon $C[t]^\diamond = t^\diamond v^\diamond$, et alors $C^\diamond[t^\diamond] = \textbf{let } x = t^\diamond \textbf{ in } xv^\diamond \to t^\diamond v^\diamond = C[t]^\diamond$.

Si $C[.] = t'.$ alors $C^\diamond[.] = \textbf{let } y = . \textbf{ in let } x = t'^\diamond \textbf{ in } xy$, d'où $C^\diamond[t^\diamond] = \textbf{let } y = t^\diamond \textbf{ in let } x = t'^\diamond \textbf{ in } xy$.

$C[t]^\diamond = (t't)^\diamond = \textbf{let } y = t^\diamond \textbf{ in let } x = t'^\diamond \textbf{ in } xy$ si t n'est pas une valeur, donc $C^\diamond[t^\diamond] = C[t]^\diamond$. Sinon $C[t]^\diamond = \textbf{let } x = t'^\diamond \textbf{ in } xt^\diamond$, et alors $C^\diamond[t^\diamond] = \textbf{let } y = t^\diamond \textbf{ in let } x = t'^\diamond \textbf{ in } xy \to \textbf{let } x = t'^\diamond \textbf{ in } xt^\diamond = C[t]^\diamond$.

Si $C[.] = (\lambda x.t')C'[.]$ alors $C^\diamond[.] = \textbf{let } x = C'^\diamond[.] \textbf{ in } t'^\diamond$, d'où $C^\diamond[t^\diamond] = \textbf{let } x = C'^\diamond[t^\diamond] \textbf{ in } t'^\diamond$.

De plus $C[t]^\diamond = ((\lambda x.t')C'[t])^\diamond = \textbf{let } x = C'[t]^\diamond \textbf{in } t'^\diamond$.

Or par induction $C'^\diamond[t^\diamond] \mapsto^* C'[t]^\diamond$, d'où par passage au contexte nous avons que $C^\diamond[t^\diamond] \mapsto^* C[t]^\diamond$ (en autant d'étapes).

De plus, si t n'est pas une valeur alors par induction $C'^\diamond[t^\diamond] = C'[t]^\diamond$, d'où $C^\diamond[t^\diamond] = C[t]^\diamond$.

Si $C[.] = vC'[.]$ (avec $C'[.] \neq .$) alors $C^\diamond[.] = \textbf{let } x = C'^\diamond[.] \textbf{ in } v^\diamond x$, d'où $C^\diamond[t^\diamond] = \textbf{let } x = C'^\diamond[t^\diamond] \textbf{ in } v^\diamond x$.

De plus $C[t]^\diamond = (vC'[t])^\diamond = \textbf{let } x = C'[t]^\diamond \textbf{ in } v^\diamond x$ car C'[t] n'est pas une valeur. En effet, comme $C'[.] \neq .$ nous avons que C'[t] est une application.

Or par induction $C'^\diamond[t^\diamond] \mapsto^* C'[t]^\diamond$, d'où par passage au contexte nous avons que $C^\diamond[t^\diamond] \mapsto^* C[t]^\diamond$ (en autant d'étapes).

De plus, si t n'est pas une valeur alors par induction $C'^\diamond[t^\diamond] = C'[t]^\diamond$, d'où $C^\diamond[t^\diamond] = C[t]^\diamond$.

Si $C[.] = C'[.]v$ (avec $C'[.] \neq .$) alors $C^\diamond[.] = \textbf{let } x = C'^\diamond[.] \textbf{ in } xv^\diamond$, d'où $C^\diamond[t^\diamond] = \textbf{let } x = C'^\diamond[t^\diamond] \textbf{ in } xv^\diamond$.

De plus $C[t]^\diamond = (C'[t]v)^\diamond = \textbf{let } x = C'[t]^\diamond \textbf{ in } xv^\diamond$ car C'[t] n'est pas une valeur. En effet, comme $C'[.] \neq .$ nous avons que C'[t] est une application.

Or par induction $C'^\diamond[t^\diamond] \mapsto^* C'[t]^\diamond$, d'où par passage au contexte nous avons que $C^\diamond[t^\diamond] \mapsto^* C[t]^\diamond$ (en autant d'étapes).

De plus, si t n'est pas une valeur alors par induction $C'^\diamond[t^\diamond] = C'[t]^\diamond$, d'où $C^\diamond[t^\diamond] = C[t]^\diamond$.

Si $C[.] = t'C'[.]$ (avec $C'[.] \neq .$) alors $C^\diamond[.] = \textbf{let } y = C'^\diamond[.] \textbf{ in let } x = t'^\diamond \textbf{ in } xy$, d'où $C^\diamond[t^\diamond] = \textbf{let } y = C'^\diamond[t^\diamond] \textbf{ in let } x = t'^\diamond \textbf{ in } xy$.

De plus $C[t]^\diamond = (t'C'[t])^\diamond = \textbf{let } y = C'[t]^\diamond \textbf{ in let } x = t'^\diamond \textbf{ in } xy$ car C'[t] n'est pas une valeur. En effet, comme $C'[.] \neq .$ nous avons que C'[t] est une application.

Or par induction $C'^\diamond[t^\diamond] \mapsto^* C'[t]^\diamond$, d'où par passage au contexte nous avons que $C^\diamond[t^\diamond] \mapsto^* C[t]^\diamond$ (en autant d'étapes).

De plus, si t n'est pas une valeur alors par induction $C'^\diamond[t^\diamond] = C'[t]^\diamond$, d'où $C^\diamond[t^\diamond] = C[t]^\diamond$.

Ainsi, on ne fait une étape de réduction qu'à la profondeur 1 de C[.], et les trois cas où cela arrive sont mutuellement exclusifs, ce qui garantit que la réduction totale se fait en au plus une étape.

Remarquons que la profondeur 1 de C[.] peut correspondre à un redex, où il n'y a pas de réduction, donc il peut ne pas y avoir d'étape de réduction même pour des contextes non dégénérés. □

3.3 Simulation de la réduction

Proposition 3.3.1. *Si $t \to u$ alors $t^\diamond \to u^\diamond$. étape par étape !*

Démonstration. Comme $t \to u$ nous avons :

Ou bien $t = (\lambda x.t')v$ et $u = t'[v/x]$, et alors $t^\diamond = \mathbf{let}\ x = v^\diamond\ \mathbf{in}\ t'^\diamond \to t'^\diamond[v^\diamond/x] = t'[v/x]^\diamond = u^\diamond$.

Ou bien $t = \mathbf{rec}(0, v, \lambda x.\lambda y.t')$ et $u = v$, et alors $t^\diamond = \mathbf{rec}(0, v^\diamond, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t'^\diamond) \to v^\diamond = u^\diamond$

Ou bien $t = \mathbf{rec}(Sn, v, \lambda x.\lambda y.t')$ et $u = (\lambda y.t'[n/x])\ \mathbf{rec}(n, v, \lambda x.\lambda y.t')$, et alors $t^\diamond = \mathbf{rec}(Sn, v^\diamond, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t'^\diamond) \to \mathbf{let}\ y = \mathbf{rec}(n, v^\diamond, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t'^\diamond)\ \mathbf{in}\ t'^\diamond[n/x] = u^\diamond$ □

Corollaire 3.3.2. *Si $t \mapsto u$ alors $t^\diamond \mapsto^+ u^\diamond$ (en au plus deux étapes).*

L'étape supplémentaire vient des contextes de profondeur 1 lorsqu'on remplace le trou par une valeur.

Démonstration. Comme $t \mapsto u$ alors il existe un unique contexte C et deux termes t' et u' tels que $t = C[t']$, $u = C[u']$ et $t' \to u'$, d'où par la proposition précédente $t'^\diamond \to u'^\diamond$.

Par passage au contexte $C^\diamond[.]$, nous avons $C^\diamond[t'^\diamond] \mapsto C^\diamond[u'^\diamond]$.

Comme $t' \to u'$, nous avons que t' est un redex ou un rec, donc n'est pas une valeur, d'où $C[t']^\diamond = C^\diamond[t'^\diamond]$.

De plus $C^\diamond[u'^\diamond] \to^* C[u']^\diamond$ (en au plus une étape).

Donc $t^\diamond = C[t']^\diamond = C^\diamond[t'^\diamond] \mapsto C^\diamond[u'^\diamond] \mapsto^* C[u']^\diamond = u^\diamond$, c'est à dire $t^\diamond \mapsto^+ u^\diamond$ (en au plus deux étapes). □

Chapitre 4

Traduction $\circ$ de $\mathbf{T}_n$ dans T

4.1 Abréviations

Définition 4.1.1. $\boldsymbol{val}\, t := \lambda z.zt$
$\boldsymbol{let}\ \boldsymbol{val}\ x = t_1\ \boldsymbol{in}\ t_2 := \lambda z.t_1(\lambda x.t_2 z)$

4.2 Traduction des termes

Définition 4.2.1.

$$\begin{aligned}
0^\bullet &:= 0 \\
(Sn)^\bullet &:= Sn^\bullet \\
x^\bullet &:= x \\
(\lambda x.t)^\bullet &:= \lambda x.t^\circ \\[1ex]
v^\circ &:= \boldsymbol{val}\, v^\bullet \\
(v_1 v_2)^\circ &:= v_1^\bullet v_2^\bullet \\
(\boldsymbol{let}\ x = t_1\ \boldsymbol{in}\ t_2)^\circ &:= \boldsymbol{let}\ \boldsymbol{val}\ x = t_1^\circ\ \boldsymbol{in}\ t_2^\circ \\
\boldsymbol{rec}(v_1, v_2, \lambda x.\lambda r.\boldsymbol{let}\ y = r\ \boldsymbol{in}\ t)^\circ &:= \boldsymbol{rec}(v_1^\bullet, v_2^\circ, \lambda x.\lambda r.\boldsymbol{let}\ \boldsymbol{val}\ y = r\ \boldsymbol{in}\ t^\circ)
\end{aligned}$$

Lemme 4.2.2. *Pour tout T_n-terme t, t° est un T-terme.*
De plus si t est une valeur alors $t^\bullet$ est un T-terme.

Démonstration. La preuve se fait par induction sur t.

Si t est une variable ou un entier, $t^\bullet = t$ est un T-terme, de même que $t^\circ = \mathbf{val}\ t$.

Si t est une abstraction $\lambda x.t'$ alors $t^\bullet = \lambda x.t'^\circ$. Or par induction t'° est un T-terme, donc $t^\bullet$ aussi. De plus $t^\circ = \mathbf{val}\ t^\bullet$ en est un également.

Si t est une application de deux valeurs $v_1 v_2$ alors $t^\circ = v_1^\bullet v_2^\bullet$. Or par induction $v_1^\bullet$ et $v_2^\bullet$ sont des T-termes, donc t° aussi.

Si t est un **let** $x = t_1$ **in** t_2 alors $t^\circ =$ **let val** $x = t_1^\circ$ **in** t_2°. Or par induction t_1° et t_2° sont des T-termes, donc t° aussi.

Si t est un $\mathbf{rec}(v_1, v_2, \lambda x.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t')$ alors $t^\circ = \mathbf{rec}(v_1^\bullet, v_2^\circ, \lambda x.\lambda r.\mathbf{let}\ \mathbf{val}\ y = r\ \mathbf{in}\ t'^\circ)$. Or par induction $v_1^\bullet$, v_2° et t'° sont des T-termes, donc t° aussi. □

Lemme 4.2.3. $(t[v/x])^\circ = t^\circ[v^\bullet/x]$
De plus si t est une valeur alors $(t[v/x])^\bullet = t^\bullet[v^\bullet/x]$.

Démonstration. La preuve se fait par induction sur t.

Si t est un entier alors t est clos, d'où $(t[v/x])^\circ = t^\circ = t^\circ[v^\bullet/x]$ car $t^\circ = \mathbf{val}\ t$ est clos également, et de plus $(t[v/x])^\bullet = t^\bullet = t^\bullet[v^\bullet/x]$ car $t^\bullet = t$ est clos.

Si t est une variable.
Si t = y, alors $(t[v/x])^\circ = y^\circ = \mathbf{val}\ y$, or $t^\circ = \mathbf{val}\ y$ d'où $t^\circ[v^\bullet/x] = \mathbf{val}\ y$.
Si t = x, alors $(t[v/x])^\circ = v^\circ = \mathbf{val}\ v^\bullet$, or $t^\circ = \mathbf{val}\ x$ d'où $t^\circ[v^\bullet/x] = \mathbf{val}\ v^\bullet$.
Dans tous les cas $(t[v/x])^\circ = t^\circ[v^\bullet/x]$.
Si t = y, alors $(t[v/x])^\bullet = y^\bullet = y$, or $t^\bullet = y$ d'où $t^\bullet[v^\bullet/x] = y$.
Si t = x, alors $(t[v/x])^\bullet = v^\bullet$, or $t^\bullet = x$ d'où $t^\bullet[v^\bullet/x] = v^\bullet$.
Dans tous les cas $(t[v/x])^\bullet = t^\bullet[v^\bullet/x]$.

Si t est une abstraction $\lambda y.t'$.
$(t[v/x])^\circ = (\lambda y.t'[v/x])^\circ = \mathbf{val}\ \lambda y.(t'[v/x])^\circ$.
$t^\circ[v^\bullet/x] = (\mathbf{val}\ \lambda y.t'^\circ)[v^\bullet/x] = \mathbf{val}\ \lambda y.t'^\circ[v^\bullet/x]$.
Or par induction $(t'[v/x])^\circ = t'^\circ[v^\bullet/x]$ donc $(t[v/x])^\circ = t^\circ[v^\bullet/x]$.
$(t[v/x])^\bullet = (\lambda y.t'[v/x])^\bullet = \lambda y.(t'[v/x])^\circ$.
$t^\bullet[v^\bullet/x] = (\lambda y.t'^\circ)[v^\bullet/x] = \lambda y.t'^\circ[v^\bullet/x]$.
Or par induction $(t'[v/x])^\circ = t'^\circ[v^\bullet/x]$ donc $(t[v/x])^\bullet = t^\circ[v^\bullet/x]$.

Si t est une application $v_1 v_2$.
$(t[v/x])^\circ = (v_1[v/x]v_2[v/x])^\circ = (v_1[v/x])^\bullet(v_2[v/x])^\bullet$.
$t^\circ[v^\bullet/x] = (v_1^\bullet v_2^\bullet)[v^\bullet/x] = v_1^\bullet[v^\bullet/x]v_2^\bullet[v^\bullet/x]$.
Or par induction $(v_1[v/x])^\bullet = v_1^\bullet[v^\bullet/x]$ et $(v_2[v/x])^\bullet = v_2^\bullet[v^\bullet/x]$.
Donc $(t[v/x])^\circ = t^\circ[v^\bullet/x]$.

Si t est un **let** $y = t_1$ **in** t_2.
$(t[v/x])^\circ = (\mathbf{let}\ y = t_1[v/x]\ \mathbf{in}\ t_2[v/x])^\circ = \mathbf{let\ val}\ y = (t_1[v/x])^\circ\ \mathbf{in}\ (t_2[v/x])^\circ$.
$t^\circ[v^\bullet/x] = (\mathbf{let\ val}\ y = t_1^\circ\ \mathbf{in}\ t_2^\circ)[v^\bullet/x] = \mathbf{let\ val}\ y = t_1^\circ[v^\bullet/x]\ \mathbf{in}\ t_2^\circ[v^\bullet/x]$.
Or par induction $(t_1[v/x])^\circ = t_1^\circ[v^\bullet/x]$ et $(t_2[v/x])^\circ = t_2^\circ[v^\bullet/x]$.
Donc $(t[v/x])^\circ = t^\circ[v^\bullet/x]$.

Si t est un $\mathbf{rec}(v_1, v_2, \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t')$.
$(t[v/x])^\circ = \mathbf{rec}(v_1[v/x], v_2[v/x], \lambda x'.\lambda r.\mathbf{let}\ y = r\ \mathbf{in}\ t'[v/x])^\circ = \mathbf{rec}((v_1[v/x])^\bullet,$ $(v_2[v/x])^\circ, \lambda x.\lambda r.\mathbf{let\ val}\ y = r\ \mathbf{in}\ (t'[v/x])^\circ)$.
$t^\circ[v^\bullet/x] = \mathbf{rec}(v_1^\bullet, v_2^\circ, \lambda x'.\lambda r.\mathbf{let\ val}\ y = r\ \mathbf{in}\ t'^\circ)[v^\bullet/x] = \mathbf{rec}(v_1^\bullet[v^\bullet/x], v_2^\circ[v^\bullet/x],$ $\lambda x.\lambda r.\mathbf{let\ val}\ y = r\ \mathbf{in}\ t'^\circ[v^\bullet/x])$.
Or par induction $(v_1[v/x])^\bullet = v_1^\bullet[v^\bullet/x]$, $(v_2[v/x])^\circ = v_2^\circ[v^\bullet/x]$ et $(t'[v/x])^\circ = t'^\circ[v^\bullet/x]$.
Donc $(t[v/x])^\circ = t^\circ[v^\bullet/x]$. □

Lemme 4.2.4. ***let val*** $x =$ ***val*** u ***in*** $t \to^+ t[u/x]$ *(en trois étapes)*

Démonstration. $\textbf{let val } x = \textbf{val } u \textbf{ in } t$
$= \lambda z.(\lambda z'.z'u)(\lambda x.tz)$
$\rightarrow_\beta \lambda z.(\lambda x.tz)u$
$\rightarrow_\beta \lambda z.t[u/x]z$
$\rightarrow_\eta t[u/x]$ □

4.3 Traduction des contextes

Définition 4.3.1. $.^\circ := .$
$(\textbf{\textit{let}}\ x = C[.]\ \textbf{\textit{in}}\ t)^\circ := \textbf{\textit{let val}}\ x = (C[.])^\circ\ \textbf{\textit{in}}\ t^\circ$

Notons $C^\circ[.]$ la traduction $(C[.])^\circ$ du contexte $C[.]$, et $C[t]^\circ$ la traduction $(C[t])^\circ$ du terme $C[t]$.

Lemme 4.3.2. *Pour tout T_n-contexte C[.], $C^\circ[.]$ est un T-contexte.*

Démonstration. La preuve se fait par induction sur C[.].
Si $C[.] = .$ alors $C^\circ[.] = .$ donc est un T-contexte.
Si $C[.] = \textbf{let } x = C'[.] \textbf{ in } t$ alors $C^\circ[.] = \textbf{let val } x = C'^\circ[.] \textbf{ in } t^\circ = \lambda z.C'^\circ[.](\lambda x.t^\circ z)$. Par induction $C'^\circ[.]$ est T-contexte, donc $C^\circ[.]$ aussi. □

Lemme 4.3.3. $C^\circ[t^\circ] = C[t]^\circ$.

Démonstration. La preuve se fait par induction sur C[.].

Si $C[.] = .$ alors $C^\circ[.] = .$ donc $C^\circ[t^\circ] = t^\circ$.
$C[t] = t$ d'où $C[t]^\circ = t^\circ$.
Donc $C^\circ[t^\circ] = C[t]^\circ$.

Si $C[.] = \textbf{let } x = C'[.] \textbf{ in } u$ alors $C^\circ[.] = \textbf{let val } x = C'^\circ[.] \textbf{ in } u^\circ$, donc $C^\circ[t^\circ] = \textbf{let val } x = C'^\circ[t^\circ] \textbf{ in } u^\circ$.
$C[t] = \textbf{let } x = C'[t] \textbf{ in } u$ d'où $C[t]^\circ = \textbf{let val } x = C'[t]^\circ \textbf{ in } u^\circ$.
Or par induction $C'^\circ[t^\circ] = C'[t]^\circ$.
Donc $C^\circ[t^\circ] = C[t]^\circ$. □

4.4 Simulation de la réduction

Proposition 4.4.1. *Si $t \rightarrow u$ alors $t^\circ \rightarrow^+ u^\circ$ (en au plus trois étapes).*
Les trois étapes viennent de la réduction du let val val, le reste se faisant en une étape.

Démonstration. Comme $t \rightarrow u$ nous avons :
Ou bien $t = \textbf{let } x = v \textbf{ in } t'$ et $u = t'[v/x]$, et alors $t^\circ = \textbf{let val } x = \textbf{val } v^\bullet \textbf{ in } t'^\circ \rightarrow^+ t'^\circ[v^\bullet/x] = (t'[v/x])^\circ = u^\circ$ (en trois étapes).
Ou bien $t = \textbf{rec}(0, v, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t')$ et $u = v$, et alors $t^\circ = \textbf{rec}(0, v^\circ, \lambda x.\lambda r.\textbf{let val } y = r \textbf{ in } t'^\circ) \rightarrow v^\circ = u^\circ$ (en une étape).
Ou bien $t = \textbf{rec}(Sn, v, \lambda x.\lambda r.\textbf{let } y = r \textbf{ in } t')$ et $u = \textbf{let } y = \textbf{rec}(n, v, \lambda x.\lambda r.$ $\textbf{let } y = r \textbf{ in } t') \textbf{ in } t[n/x]$, et alors $t^\circ = \textbf{rec}(Sn, v^\circ, \lambda x.\lambda r.\textbf{let val } y = r \textbf{ in } t'^\circ) \rightarrow$

let val y = **rec**$(n, v^\circ, \lambda x.\lambda r.$**let val** $y = r$ **in** $t'^\circ)$ **in** $t'^\circ[n/x] = u^\circ$ (en une étape). $\square$

Corollaire 4.4.2. *Si $t \mapsto u$ alors $t^\circ \mapsto^+ u^\circ$ (en au plus trois étapes).*

Ici les contextes ne rajoutent pas de nouvelle étape.

Démonstration. Comme $t \mapsto u$ alors il existe un unique contexte C et deux termes t' et u' tels que $t = C[t']$, $u = C[u']$ et $t' \to u'$, d'où par la proposition précédente $t'^\circ \to^+ u'^\circ$ (en au plus trois étapes).

Par passage au contexte $C^\circ[.]$, nous avons $C^\circ[t'^\circ] \mapsto^+ C^\circ[u'^\circ]$ (en au plus trois étapes).

Comme $C[t']^\circ = C^\circ[t'^\circ]$ et $C[u']^\circ = C^\circ[u'^\circ]$ nous avons donc que :

$t^\circ = C[t']^\circ = C^\circ[t'^\circ] \mapsto^+ C^\circ[u'^\circ] = C[u']^\circ = u^\circ$, c'est à dire $t^\circ \mapsto^+ u^\circ$ (en au plus trois étapes). $\square$

Chapitre 5

Traduction $\star$ de T_v dans T

5.1 Normalisation

Définition 5.1.1. *Notons $\star$ la composition de $\diamond$ suivie de $\circ$.*

Lemme 5.1.2. *Pour tout T_v-terme t, $t^\star$ est un T-terme.*

Démonstration. t est un T_v-terme donc $\mathrm{t}^\diamond$ est un T_n-terme et donc $t^\star = (t^\diamond)^\circ$ est un T-terme. □

Proposition 5.1.3. *Si $t \mapsto u$ alors $t^\star \mapsto^+ u^\star$.*

Démonstration. Comme $t \mapsto u$ dans T_v nous avons que $t^\diamond \mapsto^+ u^\diamond$ dans T_n, c'est à dire qu'il existe k non nul tel que $t^\diamond = t_0 \mapsto t_1 \mapsto ... \mapsto t_{k-1} \mapsto t_k = u^\diamond$.

Pour tout entier i entre 1 et k, comme $t_{i-1} \mapsto t_i$ dans T_n nous avons que $t^\circ_{i-1} \mapsto^+ t^\circ_i$ dans T.

Donc $t^\star = t^\circ_0 \mapsto^+ t^\circ_1 \mapsto^+ ... \mapsto^+ t^\circ_{k-1} \mapsto^+ t^\circ_k = u^\star$.

D'où $t^\star \mapsto^+ u^\star$. □

Théorème 5.1.4. *Normalisation dans T_v.*

Démonstration. Il suffit de montrer qu'il n'y a pas de réduction infinie.

Supposons par l'absurde qu'il existe une suite infinie de réductions, c'est à dire $(t_n)_{n\in\mathbb{N}}$ dans T_v telle que pour tout entier i $t_i \mapsto t_{i+1}$, alors $t^\star_i \mapsto^+ t^\star_{i+1}$ dans T, c'est à dire qu'il existe un entier non nul k_i tel que $t^\star_i = t^0_i \mapsto t^1_i \mapsto ... \mapsto t^{k_i}_i = t^\star_{i+1}$.

En notant $t'_{(k+\sum\limits_{j<i} k_j)} := t^k_i$ nous avons une suite $(t'_n)_{n\in\mathbb{N}}$ dans T telle que pour tout entier i $t'_i \mapsto t'_{i+1}$ (à prouver bien sûr), c'est à dire une suite infinie de réductions, ce qui contredit la normalisation dans T. □

5.2 Complexité

Définition explicite de $\star$ et précision de la simulation. Est-ce utile/intéressant ?