Logiciel certifié: preuves informelles (Mini-ML: évaluation et compilation)

Joëlle Despeyroux
INRIA Sophia-Antipolis

Module "Mécanisation des Preuves"
Dea d'informatique, Nice, janvier-mars 2000

Mini-ML - exemples

Mini-ML: λ-calcul avec constantes + polymorphisme (let), $case$ et fonctions d'ordre supérieur recursives.

Exemples:

$$let\ x = 0\ in\ let\ x = s(x)\ in\ x$$

$$let\ f = lam\ x.x\ in\ (f\ f)$$

$$let\ pred = lam\ x.(case\ x\ of\ 0 \Rightarrow 0 \mid s(x') \Rightarrow x')$$
$$in\ let\ twice = lam\ f.lam\ x.(f(f\ x))$$
$$in\ ((twice\ pred)\ 2)$$

$$fix\ double = lam\ x.\ case\ x\ of\ 0 \Rightarrow 0 \mid s(x') \Rightarrow s(s(double\ x'))$$

$$letrec\ plus =$$
$$(lam\ x.lam\ y.case\ x\ of\ 0 \Rightarrow y \mid s(x') \Rightarrow s(plus\ x'\ y))$$
$$in\ (plus\ 2\ 3)$$
$$\equiv let\ plus = (fix\ Plus =$$
$$lam\ x.lam\ y.case\ x\ of\ 0 \Rightarrow y \mid s(x') \Rightarrow s(Plus\ x'\ y))$$
$$in\ (plus\ 2\ 3)$$

Syntaxe

Expressions:

$$exp\ e\ ::=\ x \mid 0 \mid (s\ e) \mid lam\ x.e \mid (e_1\ e_2) \mid let\ x = e_1\ in\ e_2 \mid fix\ x = e \mid (case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3) \mid letrec\ x = e_1\ in\ e_2$$

$x \in \mathcal{V}ar$

sortes exp, var

sous sortes var $\subset$ exp

constructeurs

var	:		$\rightarrow$	var
0	:		$\rightarrow$	exp
s	:	exp	$\rightarrow$	exp
app	:	exp $\times$ exp	$\rightarrow$	exp
lam, fix	:	var $\times$ exp	$\rightarrow$	exp
$let, letrec$	:	exp $\times$ var $\times$ exp	$\rightarrow$	exp
$case$	:	exp $\times$ exp $\times$ var $\times$ exp	$\rightarrow$	exp

Note. Dans les expressions $(fix\ x = e)$ et $(letrec\ x = e\ in\ e_1)$, e est une lambda expression.

Évaluation - spécification

valeurs: $valeur\ v ::= 0 \mid (s\ v) \mid lam\ x.e$

évaluation:

$0 : 0 \hookrightarrow 0 \qquad s : \dfrac{e \hookrightarrow v}{(s\ e) \hookrightarrow (s\ v)} \qquad lam : lam\ x.e \hookrightarrow lam\ x.e$

$$app : \frac{e_1 \hookrightarrow lam\ x.e \quad e_2 \hookrightarrow v_2 \quad e[v_2/x] \hookrightarrow v}{(e_1\ e_2) \hookrightarrow v}$$ – appel par valeur

$$let : \frac{e' \hookrightarrow v \qquad e[v/x] \hookrightarrow v'}{let\ x = e'\ in\ e \hookrightarrow v'}$$

$$fix : \frac{e[fix\ x = e/x] \hookrightarrow v}{fix\ x = e \hookrightarrow v}$$

$$case_0 : \frac{e_1 \hookrightarrow 0 \qquad e_2 \hookrightarrow v}{case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \hookrightarrow v}$$

$$case_s : \frac{e_1 \hookrightarrow (s\ v') \qquad e_3[v'/x] \hookrightarrow v}{case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \hookrightarrow v}$$

Alternatives:

$$app_n : \frac{e_1 \hookrightarrow lam\ x.e \qquad e[e_2/x] \hookrightarrow v}{(e_1\ e_2) \hookrightarrow v}$$ – appel par nom

$$let_n : \frac{e[e'/x] \hookrightarrow v}{let\ x = e'\ in\ e \hookrightarrow v}$$

Remarque. $(let\ x = e'\ in\ e) \equiv (lam\ x.e\ \ e')$ pour l'évaluation.

Évaluation - exemples

$$\frac{lam\ x.x \hookrightarrow lam\ x.x \qquad 0 \hookrightarrow 0 \qquad x[0/x] \equiv 0 \hookrightarrow 0}{(lam\ x.x\ \ 0) \hookrightarrow 0}\ app$$

Soit $pred = lam\ x.\ (case\ x\ of\ 0 \Rightarrow 0 \mid s(x')\ \Rightarrow x')$

$$\frac{pred \hookrightarrow pred \quad \dfrac{0 \hookrightarrow 0}{s(0) \hookrightarrow s(0)} \quad \dfrac{\dfrac{0 \hookrightarrow 0}{s(0) \hookrightarrow s(0)} \quad x'[0/x'] = 0 \hookrightarrow 0}{case\ s(0)\ of\ 0 \Rightarrow 0 \mid s(x') \Rightarrow x' \hookrightarrow 0}}{(pred\ s(0)) \hookrightarrow 0}$$

Soient

$P = lam\ x.lam\ y.case\ x\ of\ 0 \Rightarrow y \mid s(x') \Rightarrow s(plus\ x'\ y))$

$P' = lam\ x.lam\ y.case\ x\ of\ 0 \Rightarrow y \mid s(x') \Rightarrow$
$\quad s((fix\ plus = P)\ x'\ y))$

$P'_2 = lam\ y.case\ 2\ of\ 0 \Rightarrow y \mid s(x') \Rightarrow s((fix\ plus = P)\ x'\ y))$

$$\frac{\dfrac{P[(fix\ plus = P)/plus]] \hookrightarrow P'}{(fix\ plus = P) \hookrightarrow P'} \quad \dfrac{(P'\ 2) \hookrightarrow P'_2 \quad P'_2[3/y] \hookrightarrow 5}{((P'\ 2)\ 3) \hookrightarrow 5}}{let\ plus = (fix\ plus = P)\ in\ (plus\ 2\ 3) \hookrightarrow 5}$$

Théorème: l'évaluation donne une valeur

Rappel: $valeur\ v ::= 0 \mid (s\ v) \mid lam\ x.e$

valeurs:

$$val_0 : \quad 0 : val$$

$$val_s : \quad \frac{e : val}{s(e) : val}$$

$$val_\lambda : \quad lam\ x.e : val$$

Thm $\forall e : exp,\ v : valeur.\ e \hookrightarrow v \Rightarrow v : val.$

Preuve. par induction sur la structure de la preuve de $e \hookrightarrow v$:

- cas 0. $0 \hookrightarrow 0$. Par val_0 on a $0 : val$.
- cas s. $\dfrac{e \hookrightarrow v}{(s\ e) \hookrightarrow (s\ v)}$

  Par hyp. d'ind. sur la preuve de $e \hookrightarrow v$ on a $v : val$.

  par val_s on a $(s\ v) : val$.
- cas app. $\dfrac{e_1 \hookrightarrow lam\ x.e \quad e_2 \hookrightarrow v_2 \quad e[v_2/x] \hookrightarrow v}{(e_1\ e_2) \hookrightarrow v}$

  Par hyp. d'ind. sur la preuve de $e[v_2/x] \hookrightarrow v$ on a $v : val$.
- les autres cas sont similaires au cas app.

□

Évaluation - en Sémantique Naturelle

valeurs: $valeur\ v ::= 0 \mid (s\ v) \mid [lam\ x.e; \rho] \mid [fix\ x = e; \rho]$

évaluation:

$$var : \frac{\rho \vdash_{val} var\ x \hookrightarrow v}{\rho \vdash var\ x \hookrightarrow v}$$

$$0 : \rho \vdash 0 \hookrightarrow 0 \qquad s : \frac{\rho \vdash e \hookrightarrow v}{\rho \vdash (s\ e) \hookrightarrow (s\ v)}$$

$$lam : \rho \vdash lam\ x.e \hookrightarrow [lam\ x.e; \rho]$$

$$app : \frac{\rho \vdash e_1 \hookrightarrow [lam\ x.e; \rho_1] \quad \rho \vdash e_2 \hookrightarrow v_2 \quad (\rho_1, x \mapsto v_2) \vdash e \hookrightarrow v}{\rho \vdash (e_1\ e_2) \hookrightarrow v}$$

$$let : \frac{\rho \vdash e' \hookrightarrow v' \qquad (\rho, x \mapsto v') \vdash e \hookrightarrow v}{\rho \vdash let\ x = e'\ in\ e \hookrightarrow v}$$

$$fix : \frac{(\rho, x \mapsto [fix\ x = e; \rho]) \vdash e \hookrightarrow v}{\rho \vdash fix\ x = e \hookrightarrow v}$$

$$case_0 : \frac{\rho \vdash e_1 \hookrightarrow 0 \qquad \rho \vdash e_2 \hookrightarrow v}{\rho \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \hookrightarrow v}$$

$$case_s : \frac{\rho \vdash e_1 \hookrightarrow (s\ v') \qquad (\rho, x \mapsto v') \vdash e_3 \hookrightarrow v}{\rho \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \hookrightarrow v}$$

Évaluation en SN (suite)

Recherche dans l'environnement:

Set val is

$$var_1 : (\rho, x \mapsto v) \vdash x \hookrightarrow v$$

$$var_0 : \frac{\rho \vdash x \hookrightarrow v}{(\rho, y \mapsto v) \vdash x \hookrightarrow v} \quad x \neq y$$

$$fix : \frac{(\rho, y \mapsto [e; (\rho_1, y \mapsto [fix\ y = e; \rho_1])]) \vdash x \hookrightarrow v}{\rho, y \mapsto [fix\ y = e; \rho_1] \vdash x \hookrightarrow v}$$

end val

Alternative pour la récursion - fold

valeurs: sans $[fix\ x = e; \rho]$.

$$valeur\ v ::= 0 \mid (s\ v) \mid [lam\ x.e; \rho]$$

évaluation:

$$fix : \frac{fold(x, e, \rho) \vdash e \hookrightarrow v}{\rho \vdash fix\ x = e \hookrightarrow v}$$

$$val_fix : \frac{x \mapsto [e; fold(x, e, \rho)].\rho \vdash x \hookrightarrow v}{fold(x, e, \rho) \vdash x \hookrightarrow v}$$

à comparer avec l'évaluation précédente:

$$fix : \frac{(\rho, x \mapsto [fix\ x = e; \rho]) \vdash e \hookrightarrow v}{\rho \vdash fix\ x = e \hookrightarrow v}$$

$$val_fix : \frac{(\rho, y \mapsto [e; (\rho_1, y \mapsto [fix\ y = e; \rho_1])]) \vdash x \hookrightarrow v}{\rho, y \mapsto [fix\ y = e; \rho_1] \vdash x \hookrightarrow v}$$

Alternative pour la récursion: letrec

expressions: $letrec$ remplace fix.

$$exp\ e\ ::= \cdots \mid letrec\ x = e_1\ in\ e_2$$

valeurs sans $[fix\ x = e; \rho]$:

$$valeur\ v ::= 0 \mid (s\ v) \mid [lam\ x.e; \rho]$$

évaluation:

$$letrec : \frac{(\rho, x \mapsto v_1) \vdash e_1 \hookrightarrow v_1 \qquad (\rho, x \mapsto v_1) \vdash e_2 \hookrightarrow v_2}{\rho \vdash letrec\ x = e_1\ in\ e_2 \hookrightarrow v_2}$$

Note. v_1 est une structure bouclée.

Théorème: l'évaluation donne une valeur

Rappel: $valeur\ v ::= 0 \mid (s\ v) \mid [lam\ x.e, \rho] \mid [fix\ x = e, \rho]$

valeurs:

$val_0 :\quad 0 : val$

$$val_s :\ \frac{e : val}{s(e) : val}$$

$val_\lambda :\ [lam\ x.e, \rho] : val$

$val_f :\ [fix\ x = e, \rho] : val$

Thm $\forall e : exp,\ \rho : env,\ v : valeur.\ \rho \vdash e \hookrightarrow v \Rightarrow v : val.$

Preuve. par induction sur la structure de la preuve de $e \hookrightarrow v$:

- cas 0. $\rho \vdash 0 \hookrightarrow 0$. Par val_0 on a $0 : val$.
- cas s. $$\frac{\rho \vdash e \hookrightarrow v}{\rho \vdash (s\ e) \hookrightarrow (s\ v)}$$
  Par hyp. d'ind. sur la preuve de $\rho \vdash e \hookrightarrow v$ on a $v : val$.
  par val_s on a $(s\ v) : val$.
- cas *app*.
  $$\frac{\rho \vdash e_1 \hookrightarrow [lam\ x.e; \rho_1] \quad \rho \vdash e_2 \hookrightarrow v_2 \quad (\rho_1, x \mapsto v_2) \vdash e \hookrightarrow v}{\rho \vdash (e_1\ e_2) \hookrightarrow v}$$
  Par hyp. d'ind. sur la preuve de $(\rho_1, x \mapsto v_2) \vdash e \hookrightarrow v$ on a $v : val$.
- les autres cas sont similaires au cas *app*.

□

Cam - syntaxe

Cam: Categorical Abstract Machine.

Syntaxe:

$coms\ l\ ::=\ coms[]\ |\ coms[c.l]$

$com\ c\ ::=\ int\ n\ |\ quote\ v\ |\ car\ |\ cdr\ |\ cons\ |\ push\ |\ swap\ |$
$app\ |\ cur(l)\ |\ rec(l)\ |\ branch(l_1, l_2)$

$v \in \mathcal{V}aleur$

sortes val, com, coms

sous sortes coms $\subset$ com

constructeurs

com_s	:	com*	$\rightarrow$	coms
int	:		$\rightarrow$	val
$quote$	:	val	$\rightarrow$	com
$car, cdr, cons$	:		$\rightarrow$	com
$push, swap, app$	:		$\rightarrow$	com
cur, rec	:	coms	$\rightarrow$	com
$branch$	:	coms $\times$ coms	$\rightarrow$	com

Cam - évaluation

$valeurs : \quad v ::= int\ n \mid (v, v') \mid [c; v]$

évaluation:

$nop : s \vdash coms[] \hookrightarrow s \qquad coms : \dfrac{s \vdash c_1 \hookrightarrow s_1 \quad s_1 \vdash c_2 \hookrightarrow s_2}{s \vdash c_1; c_2 \hookrightarrow s_2}$

$int : \alpha.s \vdash quote(n) \hookrightarrow int\ n.s$

$car : (\alpha, \beta).s \vdash car \hookrightarrow \alpha.s \qquad cdr : (\alpha, \beta).s \vdash cdr \hookrightarrow \beta.s$

$cons : \alpha.\beta.s \vdash cons \hookrightarrow (\beta, \alpha).s$

$push : \alpha.s \vdash push \hookrightarrow \alpha.\alpha.s \qquad swap : \alpha.\beta.s \vdash swap \hookrightarrow \beta.\alpha.s$

$cur : \rho.s \vdash cur(c) \hookrightarrow [c, \rho].s \qquad app : \dfrac{(\rho, \alpha).s \vdash c \hookrightarrow s_1}{([c, \rho], \alpha).s \vdash app \hookrightarrow s_1}$

$rec : \dfrac{(\rho, \alpha).s \vdash c \hookrightarrow \alpha.s}{\rho.s \vdash rec(c) \hookrightarrow \alpha.s}$

$branch_0 : \dfrac{\rho.s \vdash c_1 \hookrightarrow s_1}{(\rho, int\ 0).s \vdash branch(c_1, c_2) \hookrightarrow s_1}$

$branch_n : \dfrac{(\rho, int\ n).s \vdash c_2 \hookrightarrow s_1}{(\rho, int\ (n+1)).s \vdash branch(c_1, c_2) \hookrightarrow s_1}$

Traduction de Mini-ML → Cam

Judgement: $env \vdash exp \rightarrow com$

$$var : \frac{\rho \vdash_{access} var\ x \rightarrow c}{\rho \vdash var\ x \rightarrow c}$$

$$0 : \rho \vdash 0 \rightarrow quote(0) \qquad s : \frac{\rho \vdash e \rightarrow quote(n)}{\rho \vdash (s\ e) \rightarrow quote(n+1)}$$

$$lam : \frac{(\rho, x) \vdash e \rightarrow c}{\rho \vdash lam\ x.e \rightarrow cur(c)}$$

$$app : \frac{\rho \vdash e_1 \rightarrow c_1 \quad \rho \vdash e_2 \rightarrow c_2}{\rho \vdash (e_1\ e_2) \rightarrow push; c_1; swap; c_2; cons; app}$$

$$let : \frac{\rho \vdash e_1 \rightarrow c_1 \qquad (\rho, x) \vdash e_2 \rightarrow c_2}{\rho \vdash let\ x = e_1\ in\ e_2 \rightarrow push; c_1; cons; c_2}$$

$$fix : \frac{(\rho, x) \vdash e \rightarrow c}{\rho \vdash fix\ x = e \rightarrow rec(c)}$$

$$case : \frac{\rho \vdash e_1 \rightarrow c_1 \quad \rho \vdash e_2 \rightarrow c_2 \quad (\rho, x) \vdash e_3 \rightarrow c_3}{\begin{array}{c}\rho \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \rightarrow \\ push; c_1; cons; branch(c_2, c_3)\end{array}}$$

Alternative:

$$letrec : \frac{(\rho, x) \vdash e_1 \rightarrow c_1 \qquad (\rho, x) \vdash e_2 \rightarrow c_2}{\rho \vdash letrec\ x = e_1\ in\ e_2 \rightarrow push; rec(c_1); cons; c_2}$$

Traduction (suite)

Les variables se traduisent par un accès à la pile:

Set *access* is

$$init : \frac{\rho \mapsto coms[] \vdash x \to c}{\rho \vdash x \to c}$$

$$var_1 : x \mapsto c.\phi \vdash x \to c$$

$$var_0 : \frac{\phi \vdash x \to c}{y \mapsto c'.\phi \vdash x \to c} \quad x \neq y$$

$$pair : \frac{\rho_2 \mapsto c; cdr, \rho_1 \mapsto c; car.\phi \vdash x \to c'}{(\rho_1, \rho_2) \mapsto c.\phi \vdash x \to c'}$$

end *access*

Traduction des valeurs sémantiques

Soit ρ un env pour Mini-Ml, de la forme $((x \mapsto \alpha, y \mapsto \beta), 0 \mapsto \gamma)$.

Définition. $\bar{\rho}$: env pour la traduction:

- $\overline{\emptyset} = _$
- $\overline{x \mapsto \alpha} = x$
- $\overline{(\rho_1, \rho)} = (\bar{\rho}_1, \bar{\rho})$

Définition. $\vec{\rho}$: valeur pour la Cam:

- $\vec{\emptyset} = 0$
- $\overrightarrow{x \mapsto \alpha} = t(\alpha)$
- $\overrightarrow{(\rho_1, \rho)} = (\vec{\rho}_1, \vec{\rho})$

Example. Pour $\rho = ((x \mapsto 1, y \mapsto 2), 0 \mapsto 3)$, on a:
$\bar{\rho} = (((_, x), y), 0)$ et $\vec{\rho} = (((0, 1), 2), 3)$.

Set t **is**

$$0 : t(0) = int\ 0 \qquad s : \frac{t(v) = n}{t(s\ v) = int\ (n+1)}$$

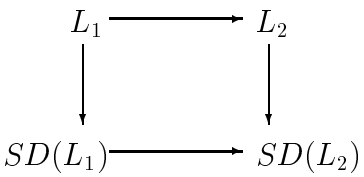
$$clos : \frac{\overline{\rho} \vdash_{ml_cam} lam\ x.e \rightarrow cur(c)}{t([lam\ x.e, \rho]) = [c, \vec{\rho}]}$$

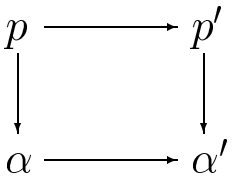
$$clos_fix : \frac{\overline{\rho} \vdash_{ml_cam} fix\ x = e \rightarrow rec(cur(c))}{t([fix\ x = e, \rho]) = \alpha = [c, (\vec{\rho}, \alpha)]}$$

end t

Preuves de correction de traduction

Commutativité d'un diagramme [Morris73]:





Programmes infinis

Def. Les *formes normales approximatives* d'un langage sont les valeurs auxquelles on ajoute $\perp$.

Def. Ordre partiel sur les formes normales approximatives:
$\forall t, t \leq \perp, \quad c_i \leq c_j \Rightarrow i = j, \quad (t,u) \leq (t',u') \Rightarrow t \leq t'$ et $u \leq u'$.

Extension de l'évaluation: On ajoute l'axiome: $\rho \vdash e \hookrightarrow \perp$.

Def. α est une *forme normale approximative* d'un terme t ssi

– α est une forme normale approximative et

– $\vdash t \hookrightarrow \alpha$ est valide.

Thm – Non-déterministe apparent:

$$\vdash p \hookrightarrow \alpha \ \& \ \vdash p \hookrightarrow \beta \Rightarrow \exists \gamma, \ \gamma \succeq \alpha, \ \gamma \succeq \beta \ s.t. \ \vdash p \hookrightarrow \gamma$$

avec: toute suite α_n tq $\vdash p \hookrightarrow \alpha_n$ admet une limite.

D'où l'existence et l'unicité du résultat.
Thm facile à prouver pour Mini-ML.

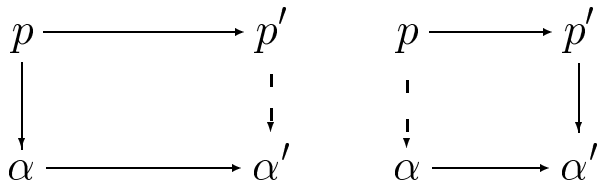
Critères de correction d'une traduction

$$\mathcal{T} = T \cup L_1_DS \cup L_2_DS \cup t$$

Éssai.
Une traduction est correcte ssi (1) et (2) sont valides dans $\mathcal{T}$:

$$\frac{\vdash_1 p \hookrightarrow \alpha \qquad \vdash_T p \to p' \qquad \vdash_t \alpha \to \alpha'}{\vdash_2 p' \hookrightarrow \alpha'} \quad (1)$$

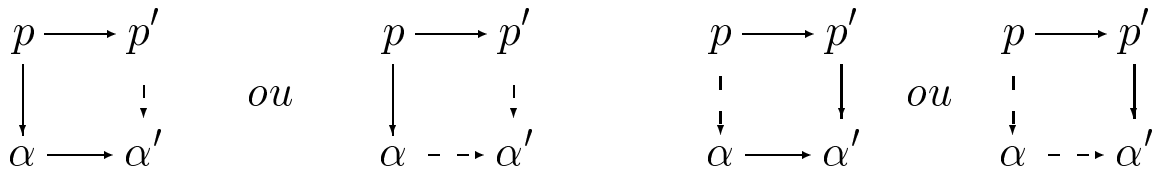
$$\frac{\vdash_T p \to p' \qquad \vdash_1 p' \hookrightarrow \alpha' \qquad \vdash_t \alpha \to \alpha'}{\vdash_1 p \hookrightarrow \alpha} \quad (2)$$



complétude validité

Problème: ces critères peuvent être trop forts:
(1) est trop fort si $\vdash_2$ est non-déterministe et t est une fonction.
(2) est trop fort si $\vdash_1$ est non-déterministe et t est injective.

Critères de correction (suite)



complétude validité

Une traduction est correcte ssi:

$$\frac{\vdash_1 p \hookrightarrow \alpha \qquad \vdash_T p \to p' \qquad \vdash_t \alpha \to \alpha'}{\vdash_2 p' \hookrightarrow \alpha'} \ (1) \quad ou$$

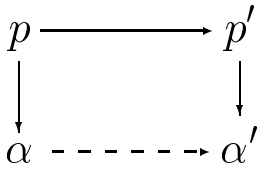
$$\exists \alpha' \frac{\vdash_1 p \hookrightarrow \alpha \qquad \vdash_T p \to p'}{\vdash_2 p' \hookrightarrow \alpha' \qquad \vdash_t \alpha \to \alpha'} \ (1')$$

et

$$\frac{\vdash_T p \to p' \qquad \vdash_1 p' \hookrightarrow \alpha' \qquad \vdash_t \alpha \to \alpha'}{\vdash_1 p \hookrightarrow \alpha} \ (2) \quad ou$$

$$\exists \alpha \frac{\vdash_T p \to p' \qquad \vdash_1 p' \hookrightarrow \alpha'}{\vdash_1 p \hookrightarrow \alpha \qquad \vdash_t \alpha \to \alpha'} \ (2')$$

En dénotationelle:



correction

Preuve de correction de la traduction

Rappel: On prouve la validité de la règle d'inférence par induction sur la structure des preuves des hypothèses.
Pour chaque preuve des hypothèses H_i, il faut exiber une preuve de la conclusion C.

– Règle 1

$$\frac{\vdash_{ml} e \hookrightarrow \alpha \qquad \vdash_{ml_cam} e \to c}{\vdash_{cam} c \hookrightarrow t(\alpha)}$$

preuve par induction d'une règle plus générale:

$$\frac{\rho \vdash_{ml} e \hookrightarrow \alpha \qquad \overline{\rho} \vdash_{ml_cam} e \to c}{\vec{\rho}.s \vdash_{cam} c \hookrightarrow t(\alpha), s}$$

– Règle 2

$$\frac{\vdash_{ml_cam} e \to c \qquad \vdash_{cam} c \hookrightarrow \alpha'}{\exists \alpha. \quad \vdash_{ml} e \hookrightarrow \alpha \quad \vdash_t \alpha \to \alpha'}$$

preuve par induction d'une règle plus générale:

$$\frac{\overline{\rho} \vdash_{ml_cam} e \to c \qquad \vec{\rho}.s \vdash_{cam} c \hookrightarrow \alpha'.s}{\exists \alpha. \quad \rho \vdash_{ml} e \hookrightarrow \alpha \quad \vdash_t \alpha \to \alpha'}$$

Lemmes

– **Lemme env**

Correction de la traduction des identificateurs en un access dans la pile:

$$\frac{\rho \vdash_{ml} x \hookrightarrow \alpha \qquad \overline{\rho} \vdash_{access} x \mapsto c}{\vec{\rho}.s \vdash_{cam} c \hookrightarrow t(\alpha).s}$$

– **Lemme cam**

Un code provenant de la traduction d'une exp ML ne modifie que le sommet de la pile:

$$\frac{\rho \vdash_{ml_cam} e \rightarrow c \qquad \alpha.s \vdash_{cam} c \hookrightarrow s'}{\exists \alpha'.\ \ \alpha.s \vdash_{cam} c \hookrightarrow \alpha'.s}$$

Preuve de la 1ère règle

Cas var.

Hypothèses:

$$\frac{\rho \vdash_{val} var\ x \hookrightarrow v}{\rho \vdash var\ x \hookrightarrow v} \quad et \quad \frac{\rho \vdash_{access} var\ x \to c}{\rho \vdash var\ x \to c}$$

Preuve: $Lemma\ env : \dfrac{\rho \vdash_{val} x \hookrightarrow \alpha \qquad \overline{\rho} \vdash_{access} x \mapsto c}{\vec{\rho}.s \vdash_{cam} c \hookrightarrow t(\alpha).s}$

Cas lam.

Hypothèses:

$$\rho \vdash lam\ x.e \hookrightarrow [lam\ x.e; \rho] \qquad \frac{(\overline{\rho}, x) \vdash e \to c}{\overline{\rho} \vdash lam\ x.e \to cur(c)} \quad et$$

$$\frac{\overline{\rho} \vdash lam\ x.e \to cur(c)}{t([lam\ x.e, \rho]) = [c, \vec{\rho}]}$$

Preuve: $\vec{\rho}.s \vdash cur(c) \hookrightarrow [c, \vec{\rho}].s$

Preuve de la 1ère règle (app)

Cas app.

Hypothèses:

$$\frac{\rho \vdash e_1 \hookrightarrow [lam\ x.e; \rho_1] \quad \rho \vdash e_2 \hookrightarrow v_2 \quad (\rho_1, x \mapsto v_2) \vdash e \hookrightarrow v}{\rho \vdash (e_1\ e_2) \hookrightarrow v}$$

$$\frac{\overline{\rho} \vdash e_1 \to c_1 \quad \overline{\rho} \vdash e_2 \to c_2}{\overline{\rho} \vdash (e_1\ e_2) \to push; c_1; swap; c_2; cons; app} \quad et$$

$$\frac{\dfrac{(\overline{\rho_1}, x) \vdash e \to c}{\overline{\rho_1} \vdash lam\ x.e \to cur(c)}}{t([lam\ x.e, \rho_1]) = [c, \vec{\rho_1}]}$$

Preuve:

$$\frac{\begin{array}{r}(\vec{\rho_1}, t(v_2)).s \vdash c \hookrightarrow t(v).s \quad [ind] \\ ([c, \vec{\rho_1}], t(v_2)).s \vdash app \hookrightarrow t(v).s \\ \vec{\rho}.[c, \vec{\rho_1}].s \vdash c_2 \hookrightarrow t(v_2).[c, \vec{\rho_1}].s \quad [ind] \\ \vec{\rho}.\vec{\rho}.s \vdash c_1 \hookrightarrow [c, \vec{\rho_1}].\vec{\rho}.s \quad [ind]\end{array}}{\vec{\rho}.s \vdash push; c_1; swap; c_2; cons; app \hookrightarrow t(v).s}$$

Preuve de la 1ère règle (fix)

Cas fix.

Hypothèses:

$$\frac{(\rho, x \mapsto [fix\ x = e; \rho]) \vdash e \hookrightarrow v}{\rho \vdash fix\ x = e \hookrightarrow v} \qquad \frac{(\overline{\rho}, x) \vdash e \to cur(c)}{\overline{\rho} \vdash fix\ x = e \to rec(cur(c))}$$

$$\frac{(\vec{\rho}, \alpha).s \vdash cur(c) \hookrightarrow \alpha.s}{\vec{\rho}.s \vdash rec(cur(c)) \hookrightarrow \alpha.s} \quad et$$

$$\frac{\overline{\rho} \vdash fix\ x = e \to rec(cur(c))}{t([fix\ x = e, \rho]) = \alpha = [c, (\vec{\rho}, \alpha)]}$$

Preuve:

$$\frac{(\vec{\rho}, \alpha).s \vdash cur(c) \hookrightarrow [c, (\vec{\rho}, \alpha)].s}{\vec{\rho}.s \vdash rec(cur(c)) \hookrightarrow [c, (\vec{\rho}, \alpha)].s}$$

où $\alpha = [c, (\vec{\rho}, \alpha)] = t([fix\ x = e; \rho])$

D'où $t(v) = [c, (\vec{\rho}, \alpha)] = \alpha \quad [ind]$.

Preuve de la 1ère règle (letrec)

Cas letrec.

Hypothèses:

$$\frac{(\rho, x \mapsto v_1) \vdash e_1 \hookrightarrow v_1 \qquad (\rho, x \mapsto v_1) \vdash e_2 \hookrightarrow v_2}{\rho \vdash letrec\ x = e_1\ in\ e_2 \hookrightarrow v_2}$$

$$\frac{(\overline{\rho}, x) \vdash e_1 \rightarrow c_1 \qquad (\overline{\rho}, x) \vdash e_2 \rightarrow c_2}{\overline{\rho} \vdash letrec\ x = e_1\ in\ e_2 \rightarrow push; rec(c_1); cons; c_2} \quad et$$

$$\frac{(\vec{\rho}, \alpha).s \vdash c \hookrightarrow \alpha.s}{\vec{\rho}.s \vdash rec(c) \hookrightarrow \alpha.s}$$

Preuve:

$$\frac{\dfrac{(\vec{\rho}, t(v_1)).s \vdash c_1 \hookrightarrow t(v_1).\vec{\rho}.s\ [ind]}{\vec{\rho}.\vec{\rho}.s \vdash rec(c_1) \hookrightarrow t(v_1).\vec{\rho}.s} \quad (\vec{\rho}, t(v_1)).s \vdash c_2 \hookrightarrow t(v_2).s\ [ind]}{\vec{\rho}.s \vdash push; rec(c_1); cons; c_2 \hookrightarrow t(v_2).s}$$

Preuve de la 1ère règle (case)

Cas case_0.

Hypothèses:

$$\frac{\rho \vdash e_1 \hookrightarrow 0 \qquad \rho \vdash e_2 \hookrightarrow v}{\rho \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \hookrightarrow v}$$

$$\frac{\overline{\rho} \vdash e_1 \to c_1 \quad \rho \vdash e_2 \to c_2 \quad (\overline{\rho}, x) \vdash e_3 \to c_3}{\begin{array}{c}\overline{\rho} \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \to \\ push; c_1; cons; branch(c_2, c_3)\end{array}}$$

$$\frac{\vec{\rho}.s \vdash c_1 \hookrightarrow s_1}{(\vec{\rho}, int\ 0).s \vdash branch(c_1, c_2) \hookrightarrow s_1}$$

Preuve:

$$\frac{\vec{\rho}.\vec{\rho}.s \vdash c_1 \hookrightarrow t(0).\vec{\rho}.s\ [ind] \qquad \dfrac{\vec{\rho}.s \vdash c_2 \hookrightarrow t(v_2).s\ [ind]}{(\vec{\rho}, t(0)).s \vdash branch(c_2, c_3) \hookrightarrow t(v_2).s}}{\vec{\rho}.s \vdash push; c_1; cons; branch(c_2, c_3) \hookrightarrow t(v_2).s}$$

Preuve de la 1ère règle (case)

Cas case_s.

Hypothèses:

$$\frac{\rho \vdash e_1 \hookrightarrow (s\ v_1) \qquad (\rho, x \mapsto v_1) \vdash e_3 \hookrightarrow v_2}{\rho \vdash case\ e_1\ of\ 0 \Rightarrow e_2 | (s\ x) \Rightarrow e_3 \hookrightarrow v_2}$$

$$\frac{\overline{\rho} \vdash e_1 \to c_1 \quad \rho \vdash e_2 \to c_2 \quad (\overline{\rho}, x) \vdash e_3 \to c_3}{\begin{array}{c}\overline{\rho} \vdash case\ e_1\ of\ 0 \Rightarrow e_2 \mid (s\ x) \Rightarrow e_3 \to \\ push; c_1; cons; branch(c_2, c_3)\end{array}}$$

$$\frac{(\vec{\rho}, int\ n).s \vdash c_2 \hookrightarrow s_1}{(\vec{\rho}, int\ (n+1)).s \vdash branch(c_1, c_2) \hookrightarrow s_1}$$

Preuve:

$$\frac{\vec{\rho}.\vec{\rho}.s \vdash c_1 \hookrightarrow t(s\ v_1).\vec{\rho}.s\ [ind] \qquad \dfrac{(\vec{\rho}, t(v_1)).s \vdash c_3 \hookrightarrow t(v_2).s\ [ind]}{(\vec{\rho}, t(s\ v_1)).s \vdash branch(c_2, c_3) \hookrightarrow t(v_2).s}}{\vec{\rho}.s \vdash push; c_1; cons; branch(c_2, c_3) \hookrightarrow t(v_2).s}$$

Bibliographie

References

[CDDK86] Dominique Clement, Joëlle Despeyroux, Thierry Despeyroux, and Gilles Kahn. A simple applicative language: Mini-ML. In *proceedings of the conference Lisp and Functional Programming, LFP'86*. ACM Press, August 1986. Also appears as INRIA Research Report RR-529, May 1986.

[Des86] Joëlle Despeyroux. Proof of translation in Natural Semantics. In *proceedings of the first Symp. on Logic In Computer Science, LICS'86*. IEEE Computer Society, June 1986. Also appears as INRIA Research Report RR-514, April 1986.

[Des98] Joëlle Despeyroux. Sémantique Naturelle: Spécifications et Preuves. Research Report RR-3359, INRIA, February 1998. Lecture notes for a graduate course in the "DEA" "Mathématiques Discrètes et Fondements de l'Informatique" (MDFI), Marseille, 1995-1999 (80 pages, in french).