Logiciel certifié:
Mini-ML: compilation

Joëlle Despeyroux

INRIA Sophia-Antipolis

Module
"Méthodes formelles et fiabilité du logiciel"
Dea d'informatique, Nice, automne 2000

Cam - syntaxe

Cam: Categorical Abstract Machine.

Syntaxe:

$$\begin{array}{lcl} coms\ l & ::= & coms[] \mid coms[c.l] \\ com\ c & ::= & int\ n \mid quote\ v \mid car \mid cdr \mid cons \mid push \mid swap \mid \\ & & app \mid cur(l) \mid rec(l) \mid branch(l_1, l_2) \end{array}$$

$v \in \mathcal{V}aleur$

sortes val, com, coms
sous sortes coms $\subset$ com
constructeurs

$$\begin{array}{lcll} com_s & : & \text{com}^* & \rightarrow \text{coms} \\ int & : & & \rightarrow \text{val} \\ quote & : & \text{val} & \rightarrow \text{com} \\ car, cdr, cons & : & & \rightarrow \text{com} \\ push, swap, app & : & & \rightarrow \text{com} \\ cur, rec & : & \text{coms} & \rightarrow \text{com} \\ branch & : & \text{coms} \times \text{coms} & \rightarrow \text{com} \end{array}$$

Cam - évaluation

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

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: $env \vdash exp \rightarrow com$

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

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

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

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

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

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

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

Alternative:

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

Traduction (suite)

Les variables se traduisent par un accès à la pile:

Set *access* is

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

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

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

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

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:

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

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

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]:

$$\begin{tikzcd} L_1 \arrow[r] \arrow[d] & L_2 \arrow[d] \\ SD(L_1) \arrow[r] & SD(L_2) \end{tikzcd}$$

$$\begin{tikzcd} p \arrow[r] \arrow[d] & p' \arrow[d] \\ \alpha \arrow[r] & \alpha' \end{tikzcd}$$

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 $\mathcal{T}$:

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

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

$$\begin{tikzcd} p \arrow[r] \arrow[d] & p' \arrow[d, dashed] \\ \alpha \arrow[r] & \alpha' \end{tikzcd} \qquad \begin{tikzcd} p \arrow[r] \arrow[d, dashed] & p' \arrow[d] \\ \alpha \arrow[r] & \alpha' \end{tikzcd}$$

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)

$$\begin{tikzcd} p \arrow[r] \arrow[d] & p' \arrow[d, dashed] \\ \alpha \arrow[r] & \alpha' \end{tikzcd} \quad ou \quad \begin{tikzcd} p \arrow[r] \arrow[d] & p' \arrow[d, dashed] \\ \alpha \arrow[r, dashed] & \alpha' \end{tikzcd} \qquad \begin{tikzcd} p \arrow[r] \arrow[d, dashed] & p' \arrow[d] \\ \alpha \arrow[r] & \alpha' \end{tikzcd} \quad ou \quad \begin{tikzcd} p \arrow[r] \arrow[d, dashed] & p' \arrow[d] \\ \alpha \arrow[r, dashed] & \alpha' \end{tikzcd}$$

complétude validité

Une traduction est correcte ssi:

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

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

et

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

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

En dénotationelle:

$$\begin{tikzcd} p \arrow[r] \arrow[d] & p' \arrow[d] \\ \alpha \arrow[r, dashed] & \alpha' \end{tikzcd}$$

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

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

preuve par induction d'une règle plus générale:

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

– Règle 2

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

preuve par induction d'une règle plus générale:

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

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:

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

Preuve de la 1ère règle

Cas var.

Hypothèses:

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

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:

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

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

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

Preuve de la 1ère règle (app)

Cas app.

Hypothèses:

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

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

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

Preuve:

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

Preuve de la 1ère règle (fix)

Cas fix.

Hypothèses:

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

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

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

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:

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

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

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

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:

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

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

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

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:

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

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

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

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

[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).