UPMC
Master informatique 2 – STL
NI503 – Conception de langages
Notes IV

2012

1 Données

On a
- les valeurs numériques : interprétées par des valeurs dans $\mathbb{R}$;
- les valeurs booléennes : interprétées aussi par des valeurs dans $\mathbb{R}$; 0 ou non ;
- on peut avoir les caractères : valuers numériques entières dans [0..255], par exemple.
- on a les n-uplets : interprétés par des produits cartésiens.

On a vu comment construire des suites d'éléments de A en se reposant sur le produit cartésien :

$$\begin{array}{lcl} A^0 & = & \{\emptyset\} \\ A^{n+1} & = & A \times A^n \\ A^\star & = & \bigcup_{n\in\mathbb{N}} A^n \end{array}$$

Opérateur (fermeture) de Kleene. Notons $\langle x, y\rangle$ les couples.

On peut donc interpréter les listes (polymorphes) :

Lexique : `nil cons car cdr pair?`

Syntaxe :

```
EXP  ::=  ...
        |  nil
        |  ( cons EXP EXP )
        |  ( car EXP )
        |  ( cdr EXP )
        |  ( pair? EXP )
```

Sémantique :

Un nouveau domaine de valeurs : les listes de valeurs

$$\left\{\begin{array}{lcl} \mathbb{L} & = & Val^\star \\ Val & = & \mathbb{R} \oplus \ldots \oplus \mathbb{L} \end{array}\right.$$

Définition sévèrement récursive, mais *bien fondée* : l'ensemble des valeurs n'est pas vide, il contient $\mathbb{R}$ (cas de base), $\mathbb{R}^\star$, $(\mathbb{R}^\star)^\star$, etc. (itérations). C'est sans doute trop.

La valeur d'une expression peut aussi être une liste. Domaine pour les valeurs d'expressions : $Data$.

$$\left\{\begin{array}{lcl} Data & = & \mathbb{R} \oplus \mathbb{L} \\ \mathbb{L} & = & Data^\star \\ Val & = & Data \oplus Adr \oplus FProc \oplus FFun \end{array}\right.$$

Abréviation : $inR()$ pour $inD(inR())$; $inL()$ pour $inD(inL())$.

Signature

$\mathbf{E} : Expr \to Env \to Mem \to Data$

Équations

$$
\begin{array}{lcl}
\mathbf{E}[[\texttt{nil}]]\rho\ \mu & = & inL(\emptyset) \\
\mathbf{E}[[\texttt{cons}\ e_1\ e_2]]\rho\ \mu & = & inL(\langle \mathbf{E}[[e_1]]\rho\ \mu, \mathbf{E}[[e_2]]\rho\ \mu\rangle) \\
\mathbf{E}[[\texttt{car}\ e]]\rho\ \mu & = & \mathsf{case}\ (\mathbf{E}[[e]]\rho\ \mu) : \\
 & & \quad inL(x) \to (fst\ x) \qquad \text{si } x \neq \emptyset \\
 & & \mid \_ \to \bot \\
\mathbf{E}[[\texttt{cdr}\ e]]\rho\ \mu & = & \mathsf{case}\ (\mathbf{E}[[e]]\rho\ \mu) : \\
 & & \quad inL(x) \to (snd\ x) \qquad \text{si } x \neq \emptyset \\
 & & \mid \_ \to \bot \\
\mathbf{E}[[\texttt{pair?}\ e]]\rho\ \mu & = & \mathsf{case}\ (\mathbf{E}[[e]]\rho\ \mu) : \\
 & & \quad inL(x) \to inR(1) \qquad \text{si } x \neq \emptyset \\
 & & \mid \_ \to inR(0)
\end{array}
$$

Réaliser les listes en mémoire La valeur d'une liste est soit la liste vide (0), soit l'adresse d'un *bloc* mémoire $\langle car, cdr\rangle$. Intuitivement/concrètement : un bloc est le couple des valeurs de 2 adresses *consécutives* en mémoire.

On trouvera donc en mémoire 2 catégories d'objets : les valeurs immédiates (éléments de $\mathbb{R}$) et les adresses des blocs.

$$\mathbb{W} = \mathbb{R} \oplus Adr \text{ et } Mem = Adr \to \mathbb{W}$$

On revoit les fonctions d'allocation, de lecture et d'écriture en mémoire. On alloue soit pour une valeur immédiate, soit pour un bloc :

$$
\begin{array}{lcl}
newM & : & Mem \to Adr \times Mem \\
newMB & : & Mem \to Adr \times Mem
\end{array}
$$

L'accès doit être possible, soit pour la valeur dans son entier (valeur immédiate ou adresse de bloc), soit pour les éléments d'un bloc :

$$
\begin{array}{lcl}
getM & : & Mem \to Adr \to \mathbb{W} \\
getM1 & : & Mem \to Adr \to \mathbb{W} \\
getM2 & : & Mem \to Adr \to \mathbb{W}
\end{array}
$$

Cette trichotomie se retrouve pour la modification :

$$
\begin{array}{lcl}
setM & : & Mem \to Adr \to \mathbb{W} \to Mem \\
setM1 & : & Mem \to Adr \to \mathbb{W} \to Mem \\
setM2 & : & Mem \to Adr \to \mathbb{W} \to Mem
\end{array}
$$

Ces opérateurs mémoire vérifient les axiomes

1. si $a, \mu' = newM(\mu)$ alors $(getM\ \mu\ a) = \bot$ et $(getM\ \mu'\ a) = inR(0)$.
2. si $a, \mu' = newM2(\mu)$ alors $(getM\ \mu\ a) = \bot$, $(getM1\ \mu\ a) = \bot$, $(getM2\ \mu\ a) = \bot$ et $(getM\ \mu\ a) = inA(a)$, $(getM1\ \mu'\ a) = inR(0)$ et $(getM2\ \mu'\ a) = inR(0)$ [1].
3. $(getM\ (setM\ \mu\ a\ w)\ a) = w$.
4. $(getM1\ (setM1\ \mu\ a\ w)\ a) = w$.
5. $(getM2\ (setM2\ \mu\ a\ w)\ a) = w$.

La valeur d'une liste peut-être une adresse en mémoire. L'évaluation d'une expression, en particulier la construction d'une liste a un effet sur la mémoire. La fonction d'évaluation d'une expression pouvant avoir un effet sur la mémoire, il faut modifier la signature de la fonction sémantique dévolue aux expressions :

$\mathbf{E} : Exp \to Env \to Mem \to \mathbb{W} \times Mem$

1. Valeur par défaut.

$$
\begin{array}{lcl}
\mathbf{E}[[\texttt{nil}]]\rho\ \mu & = & inR(0), \mu \\
\mathbf{E}[[\texttt{cons}\ e_1\ e_2]]\rho\ \mu & = & inA(a), \mu' \\
 & & \text{avec } a, \mu_0 = newMB(\mu) \\
 & & \text{et } w_1, \mu_1 = \mathbf{E}[[e_1]]\rho\ \mu_0 \\
 & & \text{et } w_2, \mu_2 = \mathbf{E}[[e_2]]\rho\ \mu_1 \\
 & & \text{et } \mu' = (setM2\ (setM1\ \mu_2\ a\ w_1)\ a\ w_2) \\
\mathbf{E}[[\texttt{car}\ e]]\rho\ \mu & = & \mathsf{case}\ w: \\
 & & \quad inA(a) \to (getM1\ \mu'\ a), \mu' \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu \\
\mathbf{E}[[\texttt{cdr}\ e]]\rho\ \mu & = & \mathsf{case}\ w: \\
 & & \quad inA(a) \to (getM2\ \mu'\ a), \mu' \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu \\
\mathbf{E}[[\texttt{pair?}\ e]]\rho\ \mu & = & \mathsf{case}\ w: \\
 & & \quad inA(a) \to inR(1), \mu' \\
 & & \mid \_ \to inR(0), \mu' \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu
\end{array}
$$

Il faut adapter quelques autres clauses de $\mathbf{E}$:

$$
\begin{array}{lcl}
\mathbf{E}[[x]]\rho\ \mu & = & \mathsf{case}\ (\rho\ x): \\
 & & \quad inA(a) \to (getM\ \mu\ a), \mu \\
 & & \mid v \to v, \mu \\
\mathbf{E}[[e_1\texttt{+}e_2]]\rho\ \mu & = & \mathsf{case}\ w_1, w_2: \\
 & & \quad inR(v1), inR(v2) \to inR(v_1 + v_2), \mu_2 \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w_1, \mu_1 = \mathbf{E}[[e_1]]\rho\ \mu \\
 & & \text{et } w_2, \mu_2 = \mathbf{E}[[e_2]]\rho\ \mu_1 \\
\vdots & & \\
\mathbf{E}[[(f\ e)]]\rho\ \mu & = & \mathsf{case}\ (\rho\ f): \\
 & & \quad inF(t) \to (t\ w\ \mu') \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu
\end{array}
$$

Et aussi, $\mathbf{C}$

$$
\begin{array}{lcl}
\mathbf{C}[[\texttt{MOVE}\ e]]\rho\ \mu & = & (setM\ (setM\ \mu'\ a_2\ inR(x + k\cos(\alpha)))\ a_3\ inR(y + k\sin(\alpha))) \\
 & & \text{avec } \alpha = outR(getM\ \mu\ a_1),\ x = outR(getM\ \mu\ a_2),\ y = outR(getM\ \mu\ a_3) \\
 & & \text{et } w, \mu' = \mathbf{E}[[e]]\rho\ \mu \\
 & & \text{et } k = outR(w) \\
\vdots & & \\
\mathbf{C}[[\texttt{CALL}\ f\ e]]\rho\ \mu & = & \mathsf{case}\ (\rho\ f): \\
 & & \quad inP(p) \to (p\ w\ \mu') \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu \\
\mathbf{C}[[x\ \texttt{:=}\ e]]\rho\ \mu & = & \text{case}\ (\rho\ x): \\
 & & \quad inA(a) \to (setM\ \mu'\ a\ w) \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu
\end{array}
$$

Des blocs de taille n opérateurs mémoires

$$
\begin{array}{lcl}
newM & : & \mathbb{N} \to Mem \to Adr \times Mem \\
getM & : & \mathbb{N} \to Mem \to Adr \to \mathbb{W} \\
setM & : & \mathbb{N} \to Mem \to Adr \to \mathbb{W} \to Mem
\end{array}
$$

On interprète la cas où le premier argument est égal à zéro comme celui dévolu aux valeurs immédiates.

Axiomes

1. si $a, \mu' = (newM\ n\ \mu)$ alors, pour tout $i \in [0 \dots n]$, $(getM\ i\ \mu) = \bot$ et $(getM\ i\ \mu') = inR(0)$.
2. $(getM\ i\ (setM\ i\ \mu\ a\ w)\ a) = w$.

Les listes

$$
\begin{array}{lcl}
\mathbf{E}[[\texttt{nil}]]\rho\ \mu & = & inR(0), \mu \\
\mathbf{E}[[\texttt{cons}\ e_1\ e_2]]\rho\ \mu & = & inA(a), \mu' \\
 & & \text{avec } a, \mu_0 = (newM\ 2\ \mu) \\
 & & \text{et } w_1, \mu_1 = \mathbf{E}[[e_1]]\rho\ \mu_0 \\
 & & \text{et } w_2, \mu_2 = \mathbf{E}[[e_2]]\rho\ \mu_1 \\
 & & \text{et } \mu' = (setM\ 2\ (setM\ 1\ \mu_2\ a\ w_1)\ a\ w_2) \\
\mathbf{E}[[\texttt{car}\ e]]\rho\ \mu & = & \texttt{case}\ w : \\
 & & \quad inA(a) \to (getM\ 1\ \mu'\ a), \mu' \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu \\
\mathbf{E}[[\texttt{cdr}\ e]]\rho\ \mu & = & \texttt{case}\ w : \\
 & & \quad inA(a) \to (getM\ 2\ \mu'\ a), \mu' \\
 & & \mid \_ \to \bot \\
 & & \text{avec } w, \mu' = \mathbf{E}[[e]]\rho\ \mu
\end{array}
$$

Les tableaux Syntaxe : déclaration de tableau, lecture/écriture d'une cellule d'un tableau.

On rajoute les crochets : `[` et `]`

$$
\begin{array}{lcl}
\text{DEC} & ::= & \dots \\
 & \mid & \texttt{VAR ident = [}\text{EXP}\texttt{]}
\end{array}
$$