Le retour des portées des variables en sémantique naturelle

Équipe de Sémantique M1

V4, 12 octobre 2005

1 Portée dynamique variables et procédures

Les règles Config = (env, S, σ) et env : $p \mapsto S$.

$$\frac{(D_V, \sigma) \rightarrow_D \sigma' \quad (\texttt{upd}(env, D_P), S, \sigma') \rightarrow \sigma"}{(env, \texttt{begin } D_V D_P; S \texttt{ end }, \sigma) \rightarrow \sigma"[\texttt{DV}(D_V) \mapsto \sigma]}$$

avec :

- $\texttt{upd}(env, \varepsilon) = env$
- $\texttt{upd}(env, \texttt{proc } p \texttt{ is S}; D_P) = \texttt{upd}(env[p \mapsto S], D_P)$

$$\frac{(env, env(p), \sigma) \rightarrow \sigma'}{(env, \texttt{call } p, \sigma) \rightarrow \sigma'}$$

Les autres commandes ont des sémantiques "copiées" sur la sémantique sans bloc : (évidemment, l'environnement n'est jamais modifié dans les autres cas...) :

$$(env, x := a, \sigma) \rightarrow \sigma[x \mapsto \mathcal{A}[a]\sigma];$$

$$\frac{(env, S_1, \sigma) \rightarrow \sigma' \quad (env, S_2, \sigma') \rightarrow \sigma"}{(env, S_1; S_2, \sigma) \rightarrow \sigma"}$$

etc

Exercice Source : `http://www.daimi.au.dk/~bra8130/Wiley_book/wiley.html`

Construire un arbre de dérivation pour l'exécution du programme suivant, où l'état initial est avec $x = 3$:

```
begin
   var y:=1;                                          -- DV0
   proc fac is                                        -- début DP0
       begin                                          -- début Sp
         var z:=x;                                    -- DV1
         if x=1 then skip                             -- début S1
                else (x:=x-1;call fac;y:=z*y)         -- fin S1
       end;                                           -- fin Sp/DP0
   call fac;                                          -- S0
end;
```

2 Portée statique procédures

Les règles Config = (env, S, σ) et env : $p \mapsto (S, env)$.

$$\frac{(D_V, \sigma) \rightarrow_D \sigma' \quad (\texttt{upd}(env, D_P), S, \sigma') \rightarrow \sigma"}{(env, \texttt{begin } D_V D_P; S \texttt{ end }, \sigma) \rightarrow \sigma"[\texttt{DV}(D_V) \mapsto \sigma]}$$

- $\texttt{upd}(env, \varepsilon) = env$

– $\texttt{upd}(env, \texttt{proc } p \texttt{ is S}; D_P) = \texttt{upd}(env[p \mapsto (S, env)], D_P)$.

et l'appel, avec $env(p) = (S, env')$:

$$\frac{(env', S, \sigma) \to \sigma'}{(env, \texttt{call } p, \sigma) \to \sigma'}$$

ou [rec]

$$\frac{(env'[p \mapsto [(S, env')], S, \sigma) \to \sigma'}{(env, \texttt{call } p, \sigma) \to \sigma'}$$

3 Portée statique procédures ET variables

Règles

Config = (env_V, env_P, S, sto) ou (env_V, sto) avec :

– $env_V : x \mapsto adresse$

– $env_P : p \mapsto (env_V, env_P, S)$ pour garder en mémoire ce que valent les variables au moment de la déclaration.

– $sto : adresse \mapsto \mathbb{Z}$ (ancien $\sigma(x) = sto(env_V(x))$).

Définition des variables (var) :

– $(\varepsilon, env_V, sto) \to_D (env_V, sto)$

– $\dfrac{(D_V, env_V[x \mapsto nc], sto[nc \mapsto v]) \to_D (env'_V, sto')}{((\texttt{var } x := a; D_V), env_V, sto) \to_D (env'_V, sto')}$

avec $v = \mathcal{A}[a](sto \circ env_V)$, $nc = new(sto)$ pour que l'on ait : $x \to nc \to v$

Statement Les changements opérés sur les règles sont plus complexes, donc on va toutes les réécrire :

- Assignation :
$$(env_V, env_P, x := a, sto) \to (env_V, sto[nc \mapsto v])$$
avec $v = \mathcal{A}[a](sto \circ env_V)$ et $nc = env_V(x)$
- Skip :
$$(env_V, env_P, \texttt{skip}, sto) \to (env_V, sto)$$
- Séquence :
$$\frac{(env_V, env_P, S_1, sto) \to (env_V, sto') \quad (env_V, env_P, S_2, sto') \to (env_V, sto")}{(env_V, env_P, S_1; S_2, sto) \to (env_V, sto")}$$
- If-true et If-false et while : similaires

Statement=block

$$\frac{(D_V, env_V, sto) \to_D (env'_V, sto') \quad (env'_V, env'_P, S, sto) \to (env_V", sto")}{(env_V, env_P, \texttt{begin } D_V D_P; S \texttt{ end}, sto) \to (env_V, sto")}$$

avec $env'_P = \texttt{upd}(env'_V, env_P, D_P)$. La fonction upd est la même que la précédente, sauf que l'environnement des procédures est un triplet qui contient l'environnement des variables à la définition.

et l'appel avec $env_P(p) = (env'_V, env'_P, S)$:

$$\frac{(env'_V, env'_P, S, sto) \to (env'_V, sto')}{(env_V, env_P, \texttt{call } p, sto) \to (env_V, sto')}$$

ou [rec]

$$\frac{(env'_V, env'_P[p \mapsto (env'_V, env'_P, S)], S, sto) \to (env'_V, sto')}{(env_V, env_P, \texttt{call } p, sto) \to (env_V, sto')}$$

Exercice Le même ...