Sémantique des Langages de Programmation (SemLP)

Projet : A Machine for CBPV

Le projet est à rendre sur Moodle et à soutenir le jeudi 23 mai. La soutenance prendra la forme de 15 minutes de présentation avec démonstration du code et explication d’une preuve de simulation.

Positive types
\[ \varphi, \psi ::= \iota \mid ! \sigma \]

General types
\[ \sigma, \tau ::= \varphi \mid \varphi \to \sigma \]

(a) Types of \( \Lambda_{HP} \)

\[ M, N ::= x \mid \bar{n} \mid M^! \mid \text{der}(M) \mid \text{succ}(M) \mid \lambda x^\sigma M \mid \langle M \rangle^N \mid \text{fix}_x^\sigma M \]
\[ \mid \text{if}(M, N, [z] P) \]

(b) Terms of \( \Lambda_{HP} \)

A typing context is an expression \( P = (x_1 : \varphi_1, \ldots , x_k : \varphi_k) \) where all types are positive and the \( x_i \)'s are pairwise distinct variables.

(c) Typing system of \( \Lambda_{HP} \).

Values are particular \( \Lambda_{HP} \) terms (they are not a new syntactic category) defined in Figure 2a. It is easy to check that they are all typed with positive types.

Figure 2 defines a deterministic weak reduction relation \( \rightarrow_w \). This reduction is weak in the sense that we never reduce within a “box” \( M^! \) or under a \( \lambda \).

The distinguishing feature of this reduction system is the role played by values in the definition of \( \rightarrow_w \). Consider for instance the case of \( \text{if} \), the term on which the test is made must be reduced to a value (necessarily of shape \( 0 \) or \( n + 1 \) if the expression is well typed) before the reduction is performed. This allows to “memoize” the value \( n \) for further usage : the value is passed to the relevant branch of the \( \text{if} \) through the variable \( z \).

We say that \( M \) is weak normal if there is no reduction \( M \rightarrow_w M' \). It is clear that any value is weak normal. When \( M \) is closed, \( M \) is weak normal iff it is a value or an abstraction.
\[ V := x \mid n \mid M'. \]

(a) Values of \( \lambda_{HP} \)

\[
\begin{align*}
\text{der}(M') & \rightarrow_w M \\
(\lambda x^\varphi M)V & \rightarrow_w M[V/x] \\
\text{fix} x^\sigma M & \rightarrow_w M \left[ \left( \text{fix} x^\sigma M \right)^1/x \right] \\
\text{succ}(n) & \rightarrow_w n + 1 \\
\text{if}(0, N, [z]P) & \rightarrow_w N \\
\text{if}(n + 1, N, [z]P) & \rightarrow_w P[n/z]
\end{align*}
\]

(b) Deterministic one-step reduction \( \rightarrow_w \)

\[ E := \text{der}(E[]) \mid \langle E[] \rangle V \mid \langle M \rangle E[] \mid \text{succ}(E[]) \mid \text{if}(E[], N, [z]P) \]

\[ E[M] \rightarrow_w E[N], \text{ whenever } M \rightarrow_w N \]

(c) Evaluation contexts and context closure of reduction \( \rightarrow_p \)

Figure 2 – Operational semantics of \( \lambda_{HP} \)

**Exercice 1:**

In this exercise, we consider \( \lambda_{HP} \) without fixpoints of terms.

1. Write an Abstract Machine without environment that simulates the evaluation of \( \lambda_{HP} \).

   **Stack Language:**
   
   \[ K := M \mid \varphi \mid \text{fun} \mid \text{arg} \mid \text{der} \mid \text{if} \mid S \text{ and } \pi := [] \mid K \cdot \pi \]

   **Reduction:**
   
   \[ (M, \pi) \rightarrow_k (M', \pi') \]

   \[
   \begin{align*}
   (\langle M \rangle N, \pi) & \rightarrow_k (N, \text{fun} \cdot M \cdot \pi) \\
   (V, \text{fun} \cdot M \cdot \pi) & \rightarrow_k (M, \text{arg} \cdot V \cdot \pi) \\
   (\lambda x^\varphi M, \text{arg} \cdot V \cdot \pi) & \rightarrow_k (M[V/x], \pi)
   \end{align*}
   \]

   Implement this Abstract Machine.

2. Prove that the reduction terminates.

3. Prove by recurrence on the length of the reduction and by case on the shape of \( M \) that if \( W \) is a value or an abstraction, then if \( M \rightarrow_w W \), then \( (M, []) \rightarrow_k (W, []) \).

   You will remark that if \( (M, []) \rightarrow_k (W, []) \), then for any \( \pi \), \( (M, \pi) \rightarrow_k^* (W, \pi) \)

4. Define a typing systems for stacks such that the translation \( * \) is compatible with types, that is:

   - If \( \vdash M : \sigma \) and \( \vdash \pi : \psi \) then \( \vdash (M, \pi) : \psi \).
   - If \( \vdash (M, \pi) : \sigma \) and \( (M, \pi) \rightarrow_k (M', \pi') \) then \( \vdash (M', \pi') : \sigma \).
   - If \( \vdash (M, \pi) : \sigma \) then \( \vdash (M, \pi)^* : \sigma \).

   For instance,
5. Give a translation \( \star \) from States of the Abstract Machine to \( \Lambda_{HP} \) such that:

- If \( (M, \pi) \to_k (M', \pi') \), then \( (M, \pi)^* = (M', \pi') \).
- Thus, if \( (M, \pi) \to_k^* (V, []) \), then \( (M, \pi)^* = V \).

For instance,

\[
\begin{align*}
(N, \text{fun} \cdot M \cdot \pi)^* &= ((M)N, \pi)^* \\
(M, \text{arg} \cdot V \cdot \pi)^* &= ((M)V, \pi)^* \text{ if } M \text{ not an abstraction} \\
(\lambda x. M, \text{arg} \cdot V \cdot \pi)^* &= (M[V/x], \pi)^* \\
\ldots
\end{align*}
\]

Prove that the translation is well defined and satisfies the wanted properties.

6. Give a compilation \( \mathcal{C} \) of CBV into \( \Lambda_{HP} \) which is compatible with the reductions.

\( \mathcal{C} : \Lambda_c \to \Lambda_{HP} \) is defined on types and terms such that:

- If \( \Gamma \vdash M : A \), then \( \mathcal{C}(\Gamma) \vdash \mathcal{C}(M) : \mathcal{C}(A) \)
- If \( \Gamma \vdash M : A \Rightarrow B \), then \( \mathcal{C}(\Gamma) \vdash \mathcal{C}(M) : \mathcal{C}(A) \Rightarrow \mathcal{C}(B) \)
- \( \mathcal{C}(\Gamma) \vdash (\mathcal{C}(\mathcal{C}(M)))\mathcal{C}(N) \)

Implement this compilation and prove the simulation theorem.

7. Give a compilation \( \mathcal{D} \) of CBN into \( \Lambda_{HP} \) which is compatible with the reductions.

\( \mathcal{D} : \Lambda_n \to \Lambda_{HP} \) is defined on types and terms such that:

- If \( \Gamma \vdash M : A \), then \( !\mathcal{D}(\Gamma) \vdash \mathcal{D}(M) : \mathcal{D}(A) \)
- If \( \Gamma \vdash M : A \Rightarrow B \), then \( !\mathcal{D}(\Gamma) \vdash \mathcal{D}(M) : !\mathcal{D}(A) \Rightarrow \mathcal{D}(B) \)
- \( \mathcal{D}(\Gamma) \vdash \mathcal{D}(M) \mathcal{D}(N) \)

Implement this compilation and prove the simulation theorem.

---

**Exercice 2 :**

In this exercise, we consider the all language \( \Lambda_{HP} \) with fixpoints of terms.

1. Extend the abstract machine defined in exercise 1 question 1 to fixpoints of terms.

2. Prove that if \( M \to_w M' \), then \( (M, [ ]) \to_k^* (M', [ ]) \).

3. In order to prove that this Abstract Machine simulates the reduction of \( \Lambda_{HP} \), we introduce a new translation which can be seen as a small step description of the Abstract Machine evaluation.

We rely on the typing system introduced in exercise 1 question 4.

- If \( \varphi \vdash \pi : \psi \), then \( \vdash \pi^* : \varphi \to \psi \).
- If \( \sigma \vdash \pi : \psi \), then \( \vdash \pi^* : !\sigma \to \psi \).

The translation is partially defined as follows:

- \( (\text{fun} \cdot M \cdot \pi)^* = \lambda v^\varphi. \langle \pi^* \rangle (\langle M \rangle v) \)
- \( (\text{arg} \cdot V \cdot \pi)^* = \lambda f^\sigma. \langle \pi^* \rangle (\langle \text{der}(f) \rangle V) \)
- \( (S \cdot \pi)^* = \lambda v^\varphi. \langle \pi^* \rangle (Sv) \)

Extend it to all stacks and check it is well typed.

4. In order to prove the simulation, we need to introduce equivalences on terms (where \( E[ ] \) is an evaluation context as defined in Figure 2c):

- If \( \vdash M : \varphi \), then \( E[M] \equiv_\varphi (\lambda v^\varphi. E[v]) M \).
– If $\vdash M : \sigma$, then $E[M] \equiv_{\sigma} \langle \lambda f^{\sigma}.E[\text{der}(f)] \rangle M^i$.

Prove that the two relations are indeed equivalence on terms of $\Lambda_{HP}$.

5. Assume that $(M, \pi) \rightarrow_k (M', \pi')$ and prove that :
   - If $\varphi \vdash \pi : \psi$, then $\langle \pi^* \rangle M^i \rightarrow^*_w \equiv_{\varphi} \langle \pi^* \rangle M'$.
   - If $\sigma \vdash \pi : \psi$, then $\langle \pi^* \rangle M^i \rightarrow^*_w \equiv_{\sigma} \langle \pi^* \rangle M'$.

Références


