

Evaluation with environments

Jonathan Lam

2022/01/03

1 Motivation

Evaluation with substitution is not efficient because it forces the re-evaluation of the substituted expression every time it is encountered. A more efficient involves an environment model, where variable values are evaluated and stored in an environment when bound and looked-up when encountered.

(Is evaluation with substitution considered normal order evaluation? This seems similar to normal/applicative order evaluation described in SICP 1.1.5.)

2 Overview

The irreducible judgment (for internal expressions) in Hazel is not $d \text{ val}$, but rather $E \vdash d \text{ final}$. Thus, final expressions evaluate to themselves. Variables evaluate to the final value that they are bound to (assuming they are bound; otherwise they are free and thus final). Lambdas evaluate to a closure type. The evaluation of a let-expression or function application extends the current environment with the newly-bound variable. For function applications, the current environment is first extended with the closure environment before binding the new variable. When extending an environment ($E :: E'$ or $E, x \leftarrow d$), bindings on the right overwrite bindings on the left. The following metatheorem states that environments only include final terms.

Theorem 1 *If the variable binding $x \leftarrow d$ exists in E , then $d \text{ final}$.*

This can be proved by induction on an empty environment by observing that all terms added to an environment must be final.

Big-step semantics

The judgment rules for evaluating variables, lambdas (which evaluate to closures), function application, **let**-expressions (very similar to function application), and a sample binary operator are shown.

$\boxed{E \vdash d \Downarrow d'}$ Internal expression d evaluates to d' given environment E

$$\frac{E \vdash d \text{ final}}{E \vdash d \Downarrow d} \text{ EvalB-Final} \qquad \frac{}{E, x \leftarrow d \vdash x \Downarrow d} \text{ EvalB-Var}$$

$$\frac{}{E \vdash (\lambda x : \tau. d) \Downarrow [E](\lambda x : \tau. d)} \text{ EvalB-Lam}$$

$$\frac{E \vdash d_1 \Downarrow d'_1 \quad d'_1 \neq ([E']\lambda x : \tau. d) \quad E \vdash d_2 \Downarrow d'_2}{E \vdash d_1(d_2) \Downarrow d'_1(d'_2)} \text{ EvalB-App}_1$$

$$\frac{E \vdash d_1 \Downarrow ([E']\lambda x : \tau. d'_1) \quad E \vdash d_2 \Downarrow d'_2 \quad E :: E', x \leftarrow d'_2 \vdash d'_1 \Downarrow d}{E \vdash d_1(d_2) \Downarrow d} \text{ EvalB-App}_2$$

$$\frac{E \vdash d_2 \Downarrow d'_2 \quad E, x \leftarrow d'_2 \vdash d_1 \Downarrow d}{E \vdash \text{let } x = d_2 \text{ in } d_1 \Downarrow d} \text{ EvalB-Let}$$

$$\frac{E \vdash d_1 \Downarrow d'_1 \quad E \vdash d_2 \Downarrow d'_2 \quad (d_1 \neq \underline{n_1} \vee d_2 \neq \underline{n_2})}{E \vdash d_1 + d_2 \Downarrow d'_1 + d'_2} \text{ EvalB-Op}_1$$

$$\frac{E \vdash d_1 \Downarrow \underline{n_1} \quad E \vdash d_2 \Downarrow \underline{n_2}}{E \vdash d_1 + d_2 \Downarrow \underline{n_1 + n_2}} \text{ EvalB-Op}_2$$

Small-step semantics

The small-step evaluation judgments equivalent to the above big-step judgments are shown below.

$\boxed{E \vdash d \rightarrow d'}$ Internal expression d takes an instruction transition to d'
given environment E

$$\frac{}{E, x \leftarrow d \vdash x \rightarrow d} \text{EvalS-Var} \qquad \frac{}{E \vdash (\lambda x : \tau. d) \rightarrow ([E'] \lambda x : \tau. d)} \text{EvalS-Lam}$$

$$\frac{E \vdash d_1 \rightarrow d'_1}{E \vdash d_1(d_2) \rightarrow d'_1(d_2)} \text{EvalS-App}_1 \qquad \frac{E \vdash d_2 \rightarrow d'_2}{E \vdash d_1(d_2) \rightarrow d_1(d'_2)} \text{EvalS-App}_2$$

$$\frac{E \vdash d_2 \text{ final} \quad E :: E', x \leftarrow d_2 \vdash d_1 \rightarrow d'_1}{E \vdash ([E'] \lambda x : \tau. d_1)(d_2) \rightarrow ([E'] \lambda x : \tau. d'_1)(d_2)} \text{EvalS-App}_3$$

$$\frac{E \vdash d_2 \text{ final} \quad E :: E', x \leftarrow d_2 \vdash d_1 \text{ final}}{E \vdash ([E'] \lambda x : \tau. d_1)(d_2) \rightarrow d_1} \text{EvalS-App}_4$$

$$\frac{E \vdash d_2 \rightarrow d'_2}{E \vdash \text{let } x = d_2 \text{ in } d_1 \rightarrow \text{let } x = d'_2 \text{ in } d_1} \text{EvalS-Let}_1$$

$$\frac{E \vdash d_2 \text{ final} \quad E, x \leftarrow d_2 \vdash d_1 \rightarrow d'_1}{E \vdash \text{let } x = d_2 \text{ in } d_1 \rightarrow \text{let } x = d_2 \text{ in } d'_1} \text{EvalS-Let}_2$$

$$\frac{E \vdash d_2 \text{ final} \quad E, x \leftarrow d_2 \vdash d_1 \text{ final}}{E \vdash \text{let } x = d_2 \text{ in } d_1 \rightarrow d_1} \text{EvalS-Let}_3$$

$$\frac{E \vdash d_1 \rightarrow d'_1}{E \vdash d_1 + d_2 \rightarrow d'_1 + d_2} \text{EvalS-Op}_1 \qquad \frac{E \vdash d_2 \rightarrow d'_2}{E \vdash d_1 + d_2 \rightarrow d_1 + d'_2} \text{EvalS-Op}_2$$

$$\frac{}{E \vdash \underline{n_1} + \underline{n_2} \rightarrow \underline{n_1 + n_2}} \text{EvalS-Op}_3$$