YuchiKaml

Yuchiki

2018 Dec.

Contents

1 Introduction

YuchiKaml is a toy language. and YuchiKaml interpreter is an implementation of interpreter of YuchiKaml. Both are created in order to get accustomed to Sprache, a C#Parser Combinator Library. In this article, I introduce both the language and the interpreter.

2 YuchiKaml Language

YuchiKaml is a dynamic typed language with-ML like surface grammar.

2.1 Syntax

Expressions of YuchiKaml are defined by the following BNF equations:

$$
\begin{aligned}
e ::= &() \mid x \mid n \mid \text{true} \mid \text{false} \mid s \mid (e) \\
&\mid e\ e \mid\ !\,e \\
&\mid e * e \mid e/e \\
&\mid e + e \mid e - e \\
&\mid e \leq e \mid e < e \mid e \geq e \mid e > e \\
&\mid e = e \mid e \neq e \\
&\mid e\ \&\&\ e \\
&\mid e \,\|\, e \\
&\mid e \rhd e \mid e \gg e \\
&\mid \text{if } e \text{ then } e \text{ else } e \mid \text{let } x\ \tilde{a} = e \text{ in } e \mid \text{let rec } x\ a_1\ \tilde{a} = e \text{ in } e \mid \lambda x.\ e \\
&\mid e\,;e
\end{aligned}
$$

The operators defined in earlier rows have stronger precedences than the operators defined in later rows. For example, $1+2*3$ is not parsed as $(1+2)*3$, but $1+(2*3)$.

In real source codes, the symbols above are notated as follows:

$$
\begin{array}{cc}
\leq & <= \\
\geq & >= \\
\neq & != \\
\& & \&\& \\
\| & \| \\
\triangleright & |> \\
\gg & >> \\
\lambda x.\, e & \backslash x->e
\end{array}
$$

Example 2.1 (GCD). This is an example of a YuchiKaml source code of a program which calculates the greatest common divisor of 120 and 45.

Listing 1: Samples/gcd

```
let rec gcd m n =
    if m > n then gcd (m − n) n
    else if m < n then gcd m (n − m)
    else m
in gcd 120 45
```

2.1.1 Syntax Sugar

TODO: explain it.

2.2 Semantics

Then we define the semantics of the expressions.

2.2.1 Value

Values of YuchiKaml is listed as below:

$$
\begin{aligned}
v(\text{value}) &::= n \mid b \mid s \mid \text{cl} \mid f_b \\
\rho(\text{valuation}) &\in \text{Var} \nrightarrow \text{Val} \\
f_b(\text{built-in function}) &\in \text{Val} \nrightarrow \text{Val} \\
\text{cl(closure)} &::= (x, e, \rho)
\end{aligned}
$$

Here Var is the set of the variables and Val is the set of the values.

2.2.2 Evaluation

We define the *evaluation* process of expression by a big-step semantics shown below.

An *evaluation relation* is a four-term relation of the form $\rho \vdash e_1 \longrightarrow e_2$.

The evaluation rules of YuchiKaml are shown below:

$$\frac{\rho(x) = v}{\rho \vdash x \longrightarrow v} \tag{E-VAR}$$

$$\frac{\rho \vdash e_1 \longrightarrow e_1'}{\rho \vdash e_1\ e_2 \longrightarrow e_1'\ e_2} \tag{E-APPLEFT}$$

$$\frac{\rho \vdash e_2 \longrightarrow e_2'}{\rho \vdash v_1\ e_2 \longrightarrow v_1'\ e_2'} \tag{E-APPRIGHT}$$

$$\frac{\rho' \cup \{x \mapsto v\} \vdash e_1 \longrightarrow e_1'}{\rho \vdash (x, e_1, \rho')\ v_2 \longrightarrow (x, e_1', \rho')v_2} \tag{E-APPCLS}$$

$$\frac{}{\rho \vdash (x, v_1, \rho')\ v_2 \longrightarrow v_1} \tag{E-APPCLS2}$$

$$\frac{f_1 @ v_2 = v_3}{\rho \vdash f_1\ v_2 \longrightarrow v_3} \tag{E-APPBUILTIN}$$

$$\frac{\rho \vdash e_1 \longrightarrow e_1'}{\rho \vdash e_1\ \text{op}\ e_2 \longrightarrow e_1'\ \text{op}\ e_2} \tag{E-BINOP-LEFT}$$

$$\frac{\rho \vdash e_2 \longrightarrow e_2'}{\rho \vdash v_1\ \text{op}\ e_2 \longrightarrow v_1\ \text{op}\ e_2'} \tag{E-BINOP-RIGHT}$$

$$\frac{n_1 \,[\![*]\!]\, n_2 = n_3}{\rho \vdash n_1 * n_2 \longrightarrow n_3} \tag{E-MUL}$$

$$\frac{n_1 \,[\![/]\!]\, n_2 = n_3}{\rho \vdash n_1 / n_2 \longrightarrow n_3} \tag{E-DIV}$$

$$\frac{n_1 \,[\![+]\!]\, n_2 = n_3}{\rho \vdash n_1 + n_2 \longrightarrow n_3} \tag{E-PLUS}$$

$$\frac{n_1 \,[\![-]\!]\, n_2 = n_3}{\rho \vdash n_1 - n_2 \longrightarrow n_3} \tag{E-MINUS}$$

$$\frac{n_1 \, [\![\leq]\!] \, n_2 = b_3}{\rho \vdash n_1 \leq n_2 \longrightarrow b_3} \qquad \text{(E-Leq)}$$

$$\frac{n_1 \, [\![<]\!] \, n_2 = b_3}{\rho \vdash n_1 < n_2 \longrightarrow b_3} \qquad \text{(E-Lt)}$$

$$\frac{n_1 \, [\![\geq]\!] \, n_2 = b_3}{\rho \vdash n_1 \geq n_2 \longrightarrow b_3} \qquad \text{(E-Geq)}$$

$$\frac{n_1 \, [\![>]\!] \, n_2 = b_3}{\rho \vdash n_1 > n_2 \longrightarrow b_3} \qquad \text{(E-Gt)}$$

$$\frac{b_1 \, [\![\&\&]\!] \, b_2 = b_3}{\rho \vdash b_1 \, \&\& \, b_2 \longrightarrow b_3} \qquad \text{(E-And)}$$

$$\frac{b_1 \, [\![||]\!] \, b_2 = b_3}{\rho \vdash b_1 \, || \, b_2 \longrightarrow b_3} \qquad \text{(E-Or)}$$

$$\frac{(v_1 = v_2) = b_3}{\rho \vdash v_1 = v_2 \longrightarrow b_3} \qquad \text{(E-Eq)}$$

$$\frac{(v_1 \neq v_2) = b_3}{\rho \vdash v_1 \neq v_2 \longrightarrow b_3} \qquad \text{(E-Neq)}$$

$$\frac{\rho \vdash e_1 \longrightarrow e_1'}{\rho \vdash \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \longrightarrow \text{if } e_1' \text{ then } e_2 \text{ else } e_3} \qquad \text{(E-IfCond)}$$

$$\frac{}{\rho \vdash \text{if true then } e_2 \text{ else } e_3 \longrightarrow e_2} \qquad \text{(E-IfTrue)}$$

$$\frac{}{\rho \vdash \text{if true then } e_2 \text{ else } e_3 \longrightarrow e_3} \qquad \text{(E-IfFalse)}$$

$$\frac{\rho \vdash e_1 \longrightarrow e_1'}{\rho \vdash \text{let } x = e_1 \in e_2 \longrightarrow \text{let } x = e_1' \text{ in } e_2} \qquad \text{(E-LetBody)}$$

$$\frac{\rho \cup \{x \mapsto v_1\} \vdash e_2 \longrightarrow e_2'}{\rho \vdash \text{let } x = v_1 \text{ in } e_2 \longrightarrow \text{let } x = v_1 \text{ in } e_2'} \qquad \text{(E-LetRem)}$$

$$\frac{}{\rho \vdash \text{let } x = v_1 \text{ in } v_2 \longrightarrow v_2} \qquad \text{(E-LetValue)}$$

$$\text{(E-LetRec)}$$

TODO: define it.

$$\frac{}{\rho \vdash \lambda x. \, e \longrightarrow (x, e, \rho)} \qquad \text{(E-Abs)}$$

3 YuchiKaml Interpreter

3.1 Usage

3.2 Preprocess