Type and Effect Systems

Pepijn Kokke, Wout Elsinghorst

Dept. of Homeland Security `kingofthehill@cs.uu.nl`

Revision: no revision

1 Introduction

Join the army! They said.
There can be only one.
It is a good day to die
Make it so.
iddqd
Pot of gold!
idkfa
See the world! They said.
Finding nemo.

2 Motivation

Our original motivation for this blasfamous work was to obtain a grade. Further motivation can be found under your chair.

3 Preliminaries

As you should know by now, type systems rule. Effect systems rule even more so.

4 Syntax

u	$\in$	**Unicode**	Unicode characters
z	$\in$	$\mathbb{Z}$	integers
r	$\in$	$\mathbb{R}$	reals
D	$\in$	**Con**	data constructors

Equivalence $\boxed{\varrho \equiv \varrho'}$

$$\frac{}{\varrho \equiv \varrho}\ [q\text{-}refl] \qquad \frac{\varrho' \equiv \varrho}{\varrho \equiv \varrho'}\ [q\text{-}symm] \qquad \frac{\varrho \equiv \varrho'' \quad \varrho'' \equiv \varrho'}{\varrho \equiv \varrho'}\ [q\text{-}trans]$$

Figure 1: Definitional equivalence of ...

Refined typing $\boxed{\widehat{\Gamma} \vdash e :: \widehat{\tau}@\varrho}$

$$\frac{}{\widehat{\Gamma} \vdash u :: \mathit{Char}@\{u\}}\ [r\text{-}chr] \qquad \frac{}{\widehat{\Gamma} \vdash z :: \mathit{Integer}@\{z\}}\ [r\text{-}int] \qquad \frac{}{\widehat{\Gamma} \vdash r :: \mathit{Double}@\{r\}}\ [r\text{-}flt]$$

$$\frac{\widehat{\Gamma}(D) = \widehat{\tau}}{\widehat{\Gamma} \vdash D :: \widehat{\tau}@\{D\}}\ [r\text{-}con] \qquad \frac{\begin{array}{c}\widehat{\Gamma} \vdash e_1 :: \mathit{Bool}@(\{\mathit{True}\} \cup \{\mathit{False}\}) \\ \widehat{\Gamma} \vdash e_2 :: \widehat{\tau}@\varrho \quad \widehat{\Gamma} \vdash e_3 :: \widehat{\tau}@\varrho\end{array}}{\widehat{\Gamma} \vdash \textbf{if } e_1 \textbf{ then } e_2 \textbf{ else } e_3 :: \widehat{\tau}@\varrho}\ [r\text{-}if]$$

$$\frac{\widehat{\Gamma} \vdash e :: \widehat{\tau}@\varrho' \quad \varrho' \subseteq \varrho}{\widehat{\Gamma} \vdash e :: \widehat{\tau}@\varrho}\ [r\text{-}sub]$$

Figure 2: Static semantics.

l	$\in$	**Literal**	literals
e	$\in$	**Exp**	expressions

$$\begin{array}{lcl} l & ::= & u \mid z \mid r \\ e & ::= & l \mid D \mid \textbf{if } e_1 \textbf{ then } e_2 \textbf{ else } e_3 \end{array}$$

5 Type system

Well typed programs can't go wrong. Untyped lambda calculus is never wrong.

6 Algorithm

Using Quantum Mechanics and other obfuscation techniques, our algorithm has attained a form of True Elegance. No further explanation necessary.

7 Related work

It might be the case that our fellow students duplicated our work. Their results might prove fruitful.

$$
\begin{array}{l}
\mathbf{E}(\widehat{\Gamma}, l) \qquad\qquad\qquad\qquad\qquad\quad = \mathbf{L}(l) \\
\mathbf{E}(\widehat{\Gamma}, D) \qquad\qquad\qquad\qquad\qquad\quad = \text{let } \zeta \text{ be fresh in } (\widehat{\Gamma}(D)\ ,\ \zeta\ ,\ \{\{D\} \subseteq \zeta\}) \\
\mathbf{E}(\widehat{\Gamma}, \textbf{if } e_1 \textbf{ then } e_2 \textbf{ else } e_3) = \\
\quad \text{let } \zeta \text{ be fresh} \\
\qquad (\widehat{\tau}_1, \zeta_1, C_1) = \mathbf{E}(\widehat{\Gamma}, e_1) \\
\qquad (\widehat{\tau}_2, \zeta_2, C_2) = \mathbf{E}(\widehat{\Gamma}, e_2) \\
\qquad (\widehat{\tau}_3, \zeta_3, C_3) = \mathbf{E}(\widehat{\Gamma}, e_3) \\
\quad \text{in if} \quad \widehat{\tau}_1 = \mathit{Bool} \wedge \widehat{\tau}_2 = \widehat{\tau}_3 \\
\qquad \text{then } (\widehat{\tau}_2\ ,\ \zeta\ ,\ C_1 \cup C_2 \cup C_3 \cup \{(\{\mathit{True}\} \cup \{\mathit{False}\}) \supseteq \zeta_1\} \cup \{\zeta_2 \subseteq \zeta\} \cup \{\zeta_3 \subseteq \zeta\}) \\
\qquad \text{else fail}
\end{array}
$$

Figure 3: Reconstruction algorithm: expressions.

```
solve(C) = do
  ...
  for all c in C do worklist := worklist ∪ {c}
  ...
  while worklist ≠ { } do
    let C1 ⊎ {c} = worklist
    in do ···
          case c of
            ...
  return analysis
```

Figure 4: Worklist algorithm for constraint solving.

8 Conclusion and future work

There is not much to conclude. We rule.