Imperial College London

Department of Computing

Exception Handling in Haskell

by

William S. Fisher

supervised by

Steffen van Bakel

Submitted in partial fulfillment of the requirements for the MSc
degree in Computing Science of Imperial College London

September 2016

Acknowledgements

Thanks me

Contents

CHAPTER 1

Introduction

Hel

CHAPTER 2

Background

Formal Systems

λ-Calculus

Logic, Types, and their Computation Interpretation

Continuations.

Delimited-Continuations.

Haskell

Monads.

$\lambda\mu$-Calculus

λ^{try}-Calculus

Delimited-Continuation Calculus

Simon Peyton-Jones *et al.* extended the λ-calculus with additional operators in order create a framework for implementing delimited continuations [**1**]. This calculus will be referred to as the delimited-continuation calculus or DCC. Many calculi have been devised with control mechanisms. Like the $\lambda\mu$-calculus, these control mechanisms are all specific instances of delimited and undelimited continuations. DCC provides a set of operations that are capable of expressing many of these other common control mechanisms.

The grammar of DCC is an extension of the standard λ-calculus:

Definition 0.1 (Grammar rules for DCC)

$$
\begin{array}{llcl}
\text{(Variables)} & x, y, \ldots & & \\
\text{(Expressions)} & e & ::= & x \mid \lambda x.e \mid e\ e' \\
 & & \mid & newPrompt \mid pushPrompt\ e\ e \\
 & & \mid & withSubCont\ e\ e \mid pushSubCont\ e\ e
\end{array}
$$

The additional terms behave as follows:

- *newPrompt* returns a new and distinct prompt.
- *pushPrompt*'s first argument is a prompt which is pushed onto the continuation stack before evaluating its second argument.
- *withSubCont* captures the subcontinuation from the most recent occurrence of the first argument (a prompt) on the excution stack to the current point of execution. Aborts this continuation and applies the second argument (a λ-abstraction) to the captured continuation.

- *pushSubCont* pushes the current continuation and then its first argument (a subcontinuation) onto the continuation stack before evaluating its second argument.

CHAPTER 3

DCC Interpreter

Interpreter

Whereas the original grammar for the DCC abstract machine presents sequences as values, it has an implicit semantics that is unpacked in the implementation details. Here, we present sequences as expressions and define their semantics explicitly as:

Definition 0.2 (Semantics of a sequence of continuations)
Let D_i denote some term with a hole and $D_i[v]$ denote the term D_i with the hole filled by $v4$:

$$\langle (D_1 : D_2 : \cdots : D_n), D', E, q \rangle \quad \Rightarrow \quad \langle \lambda x. D_n[D_{n-1}[\ \ldots D_1[x] \ldots]], D', E, q \rangle$$

CHAPTER 4

Translations

λ^{try}-to-$\lambda\mu$

$\lambda\mu$-to-DCC

λ^{try}-to-DCC

CHAPTER 5

Conclusion

Evaluation

Conclusion

Future Work

Bibliography

[1] R. Kent Dybvig, Simon L. Peyton Jones, and Amr Sabry. A monadic framework for delimited continuations. *J. Funct. Program.*, 17(6):687–730, 2007.