A Denotational Semantics for Polymorphic Effect Systems

A Part III project proposal

A. J. Taylor (*at736*), St John's College

Project Supervisor: Prof Alan Mycroft

Abstract

A category-theoretic approach to build a graded monad-based denotational semantics for a language with polymorphism over effects.

1 Introduction, approach and outcomes (500 words)

A denotational semantics of a programming language is an extremely useful tool for static program analysis. Since Scott and Strachey's original work using lattices in the 1970s, denotational models have been developed for increasingly complex languages, such as using monads to model effects[1] and categories to model polymorphism over types[2]. No work, however, appears to have been published to date which describes a denotational semantics for a language with polymorphism over effects. Allowing such polymorphism may be useful to generalise over effects and open the door to better reasoning about functions in an effectful language. (Cf. Theorems for Free[3] for a similar phenomenon with polymorphic types.) In this project, I intend to formalise such a denotational semantics for a language with effect-polymorphism.

The approach I intend to approach consists of two broad phases: constructing a denotational semantics for a polymorphism-free language by combining existing techniques, then the novel step of incorporating effect-polymorphism to the base language and extending the denotational semantics to account for polymorphism. The first phase shall occur over the Christmas vacation and involve building a simple, lambda calculus-based language free of effect-polymorphism. This language shall contain an abstract graded monad to handle effects and effect-subtyping. After proving simple properties of the language, such as the weakening and substitution lemmas, and theorems related to type safety, I shall construct a category-theoretic description of the denotational semantics of the this language in a manner similar to that of Moggi[4]. Moving into the second phase in Lent term, I shall then attempt to incorporate effect-polymorphism into the abstract denotational semantics of the language. In doing so, I hope to characterise the constraints on a category that allow it to be a model of a language with polymorphic effects.

Although the denotational semantics for a language, such as PLC, with polymorphic types was originally difficult to construct due to a manifestation of Russell's paradox in fully polymorphic types

[1] Moggi, Notions of computation and monads: https://core.ac.uk/download/pdf/21173011.pdf

[2] Jacobs, Categorical Logic and Type Theory, Chapter 8

[3] https://people.mpi-sws.org/ dreyer/tor/papers/wadler.pdf

[4] https://core.ac.uk/download/pdf/21173011.pdf

being able to [5], there is good reason to expect that the semantics of polymorphic effects would be easier to formalise due to effects not being self referential in the same way as polymorphic types .

Deliverable The main deliverable of this project will the presentation of a sound denotational semantics for the aforementioned language, along with proofs of the standard denotational semantics theorems. These theorems are soundness, compositionality, and potentially adequacy for individual concrete cases).

[5]"Polymorphism is not set-theoretic" by Reynolds. https://hal.inria.fr/inria-00076261/document

2 Workplan (500 words)

2^{nd}December - 15^{th}December	Construct a simple, polymorphism-free, graded-monadic lambda calculus-based language with a type system and operational semantics. This language shall be designed such that effect polymorphism can be appended onto the core in an easy and intuitive way. I expect that I shall take the route of having an explicit graded monad in the language, and effect-polymorphism shall later be added in a similar fashion to how type-polymorphism is expressed in the polymorphic lambda calculus. That is, explicit generalisation and specialisation terms shall be added.
16^{th}December - 29^{th}December	Prove simple properties of operational semantics without effect polymorphism. These shall include the Weakening and Substitution Lemmas, Type Preservation, Progress, Type Safety.
30^{th}December - 12^{th}January	Characterise an abstract model for the language in category theory using cartesian-closed categories. This shall be performed in a similar fashion to Andrew Pitts' example for STLC and the original paper by E. Moggi. This step should be relatively simple as this work has been done before. This chunk of time also coincides with the hand-in dates for several taught course that I am taking.
13^{th}January - 26^{th}January	Add effect polymorphism to the language and extend the proofs of simple operational properties to the new polymorphic language. This is the first section of the project to steer into novel territory.
27^{th}January - 9^{th}February	Extend denotational semantics to the effect-polymorphic language.
10^{th}February - 23^{rd}February	
24^{th}February - 9^{th}March	Continue extension of denotations, aiming to formalise and prove the standard properties of a denotational semantics (Soundness, Adequacy, equal denotations $\Rightarrow$ contextual equivalence). This is likely to be the hardest part of the project, and may require iteration with the previous step to refine the denotational model.
10^{th}March - 23^{rd}March	Extensions. This time chunk is deliberately left vague as it not yet known what I might find in the previous step. Potential contents here might include instantiating the abstract model to model particular concrete languages with particular effects, proving adequacy for a particular concrete model and language, finding deficiencies in categorical approaches which prevent the construction of such a denotational system, or further generalisation to include type polymorphism.
24^{th}March - 6^{th}April	Collation of results and optimisation of proofs for presentation
7^{th}April - 20^{th}April	Write dissertation, including iterations incorporate feedback.
21^{st}April - 4^{th}May	
5^{th}May - 18^{th}May	Contingency and hand in.
19^{th}May - 31^{st}May	