Project Weekly Report 1

Edward While

November 2020

Goals

I began this week with the goal of starting in earnest my research into how I was going to implement the formalisation of the dual calculus in Agda, as I felt that I already had a relatively strong understanding of the dual calculus itself. As such I intended to complete all the relevant parts of Programming Language Foundations in Agda [1], and have studied in detail the chapter on an intrinsically-typed representation of the Simply Typed λ-Calculus. I also thought that more research into the Duality of Call-by-Name and Call-by-Value, as well as Continuations and CPS transformations would prove useful for when I look to formalise those concepts in Agda. And finally I aimed to research into intrinsically-typed representation of terms to make sure I had a solid understanding of how to use them and why they were useful before I tried to implement them.

What I did and learned

I spent early parts of the week reading through various papers to gain a better understanding of some concepts that should prove relevant in my project. This included Guillaume Allais' Thesis [2] on indexed type families and the intrinsically typed and scoped syntax that I would be using for my Dual Calculus formalisation. While I had previously implemented the Simply Typed λ-Calculus using this technique I felt that I needed to gain a better understanding of how it worked and why it was useful. This thesis also contains a section on CPS transformations that I studied quite intently as my project will cover this specific area. Continuing in this subject area I also read Conor McBride's paper [3] in which he presents an algorithm for substitution for the STλC that is statically guaranteed to respect type and scope using a similar technique.

I also spent some time researching more into continuations, something I have struggled to get my head around before but that will prove important for understanding later parts of the project when I look at CPS transformations. This research included looking at the history of our understanding of continuations and how it has changed in John Reynold's paper 'The Discoveries of

Continuations" [4], and Filinski's paper [5] that studies CPS transformations in depth.

I continued my work through the Programming Language Foundations in Agda textbook [1] and completed all the sections that should prove relevant for the project, I know feel like I have a strong enough grasp on Agda to make a start.

After discussing with my supervisor I decided it would be a good idea to start doing some programming scratch work to get a stronger idea of what my code might look like and to work out areas that will require more thought in implementation. As such I have written up code allowing me to encode terms of the dual calculus in Agda. The implementation looked about as expected, the main changes from the STλC encoding being that terms are indexed by both their type and two separate contexts of variables and covariables. I then used this encoding to prove the law of the excluded middle, a result that only occurs in classical logic. I continued this work to start on producing a dual translation with some success but have discovered some interesting issues that will require resolution, such as how to represent function types in terms of other logical connectives, and how the dual translation of variable/covariable abstractions will be handled. Next week will include more work on these issues, and research into duality of CBV and CBN.

References

[1] Philip Wadler, Wen Kokke, and Jeremy G. Siek. *Programming Language Foundations in Agda.* July 2020.

[2] Guillaume Allais. *Syntaxes with Binding, Their Programs, and Proofs.* PhD thesis, University of Strathclyde.

[3] Conor McBride. Type-preserving renaming and substitution. *Functional Programming.*

[4] John Reynolds. The discoveries of continuations. *LISP and Symbolic Computation*, 6:233–248, 1993.

[5] Oliver Danvy and Andrzex Filinski. Representing control: a study of the cps transformation. *Mathematical Structures in Computer Science*, 2(4):361–391, 1992.