Project Weekly Report 2

Edward While

November 2020

Goals

This week I set myself the goals of working through some of the interesting issues that my scratch work programming had thrown up, this included how function type terms should be represented in the calculus and how to handle dual translation of variable terms and variable abstractions. I also aimed to do more research into the duality of CBV and CBN, and to make further progress in my programming scratch work.

What I did

Firstly I had to settle on a way to represent function types in the dual calculus, this is because Wadler [1] defines them in terms of the other connectives of the calculus, something that is not particularly intuitive in Agda. I decided to essentially treat function types, lamdbda abstractions and function application as syntactic sugar, each of which is a function defined entirely in terms of other constructs of the calculus, these definitions are given by Wadler. I have provided separate definitions of function types, lambda abstractions, and function application for Call-by-Name and Call-by-Value.

The issues I had run into in defining the dual of a variable term and of variable abstractions required defining two more duality operations. The dual translation of a context maps one context to another and recursively applies the dual translation of a type to each type in the context. The dual translation of a variable maps from the membership relation between a type and context to the same relation over the dual of that context and the dual of that type. Once these had been defined the issues above were essentially solved.

I made further progress in my programming scratch work implementing both a proof that the duality translation is an involution, and that a term is derivable iff its dual is derivable. The involution proof took up a large part of the week due to the fact that the dual calculus terms are indexed by their Contexts and Type, which will change after dual translation, thus we cannot simply show their equality as they are of different types. This was solved by adding a rewrite rule that declares the equality of those types, which I also proved. The proof that a term is derivable iff its dual is derivable turned out to be trivial due to the

intrinsically typed nature of the dual calculus terms and the previous involution proof.

My research this week involved a deeper dive into the parts of Wadler's papers [1, 2] that discuss CBV, CBN and the duality between them, as well as reading Curien and Herbelin's earlier work [3] that Wadler built on in his demonstration of this duality.

What I learned

This week I refined my knowledge of the duality between CBV and CBN as I discussed above. I have also seen an improvement of my knowledge in how to implement proofs in Agda. This includes research into possible solutions for my issues with the involution proof such as Heterogeneous Equality, and a better understanding of Propositional Equality and various rules that can be used such as subst. With the help of my supervisor I have also looked at using Pragmas for rewrite rules and a possible use of Heterogeneous Equality that may avoid the need for us to do this.

References

[1] Philip Wadler. Call-by-value is dual to call-by-name. In *Proceedings of ICFP'03*, Uppsala, Sweden, August 2003.

[2] Philip Wadler. *Call-by-Value is Dual to Call-by-Name – Reloaded.* 2005.

[3] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. *SIGPLAN Not.*, 35(9):233–243, September 2000.