Type inference with constraints for the Static Pattern Calculus

Jose Alberto Vergara Medina

October 1, 2010

Abstract

In pattern calculus the ability to arbitrarily traverse data structures is called path polymorphism (Jay 2009). Typing of path polymorphic functions is challenging because the type of a compound data-structure does not determine the types of its components (Jay 2009). Typed Static Pattern Calculus (SPC) already types path polymorphic functions and proivides type inference, but there are few resources apart from the **bondi** programming language that can be used to implement type inference for a SPC compiler.

This talk presents a type inference algorithm (J-SPC) for the typed static pattern calculus, using a constraint-based approach. J-SPC is inspired on othe work of Pottier and Remy (Pierce 2005) on type inference with constraints for the OCaml programming language and the work on the HM(X) framework by Odersky, Sulzmann and Wehr(Odersky, Sulzmann and Wehr 1999) . It is claimed in (Pierce 2005) that the separation of constraint-generation and constraint-solving is desirable to achieve better modularity. It is to be shown if better modularity contributes to simplify type inference for the SPC in a way that makes it more sutiable for a compiler implementation of the Static pattern Calculus.

References

Jay, B. 2009, *Pattern calculus: computing with functions and structures*, Springer Verlag.

Odersky, M., Sulzmann, M. and Wehr, M. 1999, Type inference with constrained types, *Theory and practice of object systems* **5**(1), 35–55.

Pierce, B. 2005, *Advanced topics in types and programming languages*, The MIT Press.