## Presentation Abstract

# Polymorphic Staged Calculus with Cross-Stage Persistence and Side Effects

Ayane Kikuchi[1,a)] Yukiyoshi Kameyama[1]

Presented: January 17, 2019

In this presentation, we explain a new polymorphic staged calculus with cross-stage persistence (CSP) and side effects, and show a rigid proof of its type soundness. Multi-stage computing is ubiquitous in various application domains to achieve high performance and high maintainability. Despite its widespread success, foundational issues such as static type soundness remain unsolved for the polymorphic staged calculi with practical extensions including reference and CSP. CSP is a useful feature to allow values to cross the boundary of stages, namely, a value computed in the present stage may be used in a future stage. MetaOCaml, a staged extension of OCaml, allows CSP for all kinds of values, which is shown to be unsound in the presence of reference and polymorphism under value restriction. Kobayashi and Igarashi solved the problem based on Tofte's imperative type discipline, but it diverts from the current design of MetaOCaml since it needs to memoize the levels of reference types. We present a new simple solution for this problem by allowing two kinds of polymorphism; value restriction with restricted CSP and covariance restriction, as a natural extension of OCaml's relaxed value restriction. We argue that our extension fits the current (and future) design of MetaOCaml, and allows sufficiently many programs to type check. We show that our type system enjoys the type soundness property.

This is the abstract of an unrefereed presentation, and it should not preclude subsequent publication.

[1] Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan

[a)] kikuchi@logic.cs.tsukuba.ac.jp