Seminar/Talk

Heap-based reasoning about asynchronous programs

Johannes Kloos
MPI-SWS

Host: Tom Henzinger

Asynchronous concurrency is a model of concurrency based on the concept of tasks. It executes tasks one-at-a-time, choosing the next task to run from a set called the task buffer, adding tasks to the buffer using a "post" primitive, and scheduling a new task only when the currently running task gives up control. Task can depend on each other using explicit "wait" operations. This model and its variants are used widely in languages such as JavaScript, C# (the async and await primitives), or the monadic concurrency libraries of Haskell and OCaml. The more restricted scheduling separates it from the more commonly considered multi-threaded programming model.

In this talk, I will address talk about two projects. The first project deals with the question how we can reason about asynchronous programs. We present a program logic and a corresponding type system that allow us to reason locally about programs with asynchronous concurrency and mutable state; we instantiate this model for OCaml using the Lwt library. The key innovation is to introduce the concept of a "wait permission", which describes the resources that a given task will yield on termination. An earlier version of this work was published at ECOOP '15. The second project deals with the question how we can perform a kind of "asynchronous parallelization" optimization, where we are given a (more-or-less) sequential program and rewrite it to make use of asynchronous concurrency. We use a set of program rewriting rules, most notably replacing synchronous I/O operations with asynchronous counterparts in a safe way, and pushing wait statements as far back as possible. As it turns out, proving the soundness of these rewriting rules is surprisingly tricky; I will sketch a reasoning approach that allows us to show refinement, in the the following sense: Let e be a program, and e' the result of rewriting e using the given rules. For every terminating execution of e', there is a corresponding terminating execution of e that ends in an equivalent state.

Wednesday, September 6, 2017 05:00pm - 06:00pm
Mondi Seminar Room 3, Central Building

This invitation is valid as a ticket for the ISTA Shuttle from and to Heiligenstadt Station. Please find a schedule of the ISTA Shuttle on our webpage: https://ista.ac.at/en/campus/how-to-get-here/ The ISTA Shuttle bus is marked ISTA Shuttle (#142) and has the Institute Logo printed on the side.