Chair of Programming Methodology

Master's Thesis

Multiple Heaps for Ownership-Based Verification

Christoph Studer
chstuder@student.ethz.ch

14-11-2008

Supervisors: Prof. Peter Müller, Arsenii Rudich
{peter.mueller,arsenii.rudich}@inf.ethz.ch

Introduction

Ownership simplifies verification of complex object-oriented programs by restricting how references can be passed around and used. It effectively imposes structure on the object store and ensures locality of modifications, which facilitates verification of invariants.

Spec# currently encodes the ownership restrictions into axioms and theorems. The object store, however, is modeled as one global heap, disregarding all structural information from the ownership type system.

Goal

The goal of this thesis is to define and evaluate a technique which preserves structural information from the ownership type system by conceptually splitting the global heap into smaller heaps for verification.

We believe that encoding the structural information on a heap level is simpler and more efficient than current approaches. With heap splitting, theorems and axioms encoding the ownership type system can be removed, and some proofs can be simplified because of more confined heaps.

Core Parts

The following parts need to be completed for the thesis to be considered sufficient:

(1) The first part of the thesis will be the **definition** of a set of heap splitting rules, such that as much of the structural information as possible is transfered from the ownership type system to the theorem prover. These rules must preserve semantic equivalence. (2) Thereafter, the new technique will be analyzed and refined to improve reasoning about **pure methods**, which we believe will profit a lot from more confined heaps. (3) The third part will be the application of the new technique to **example programs** and performance will be compared to the traditional Spec# encoding. (4) Finally, the new technique will be **implemented** in Spec# such that programs can be automatically verified using heap splitting.

Possible Extensions

1. Refine heap split technique to support ownership transfer.
2. Add support for read and write effects.