ETH zürich

Alexander J. Summers

Program Verification

Exercise Sheet 9: Heap Reasoning and Permissions

Assignment 1 (Pure Assertions)

An IDF assertion is called *pure* if it doesn't contain any accessibility predicates. For example, $x.f = 4$ is a pure assertion. In practice, pure assertions are typically expressions, but can technically also include the separating conjunction $*$.

1. Prove that, for all states H, P, σ and pure assertions A, if $H, P, \sigma \vDash A$ then $H, \varnothing, \sigma \vDash A$ (where $\varnothing$ represents the empty permissions mask, which maps all field locations to 0).
2. Prove that, for any assertion A' and any pure assertion A the assertions $A * A'$ and $A \wedge A'$ are logically equivalent (i.e. they are true in the same states)[1].

Assignment 2 (Permissions Required by an Assertion)

Recall that permission masks P (as defined in the lecture) are maps from an (object, field-name) pair to a rational number in the range $[0, 1]$. Let *Perms*$(A)_{(H,\sigma)}$ be a function that maps an assertion A to the (pointwise) *minimal* permission mask P_A such that: when applying the definition of $H, P_A, \sigma \vDash A$ (from slide 213), the *second case* of the definition (that for accessibility predicates) never evaluates to false. In other words, *Perms*$(A)_{(H,\sigma)}$ must contain the minimal possible permissions *necessarily required* due to accessibility predicates occurring in A (in the given H and σ).

1. Why does the result of this operation need to depend on H and σ? Hint: think how would you handle an implication $e \Rightarrow A$.
2. How will the case for *Perms*$(A_1 \wedge A_2)_{(H,\sigma)}$ differ from the case for *Perms*$(A_1 * A_2)_{(H,\sigma)}$?
3. Write down a definition for *Perms*$(A)_{(H,\sigma)}$ for each case in a similar style to the $\vDash$ definition on slide 213.

[1] Note that, as a simple consequence of this latter result, any pure assertion can be equivalently represented as a (boolean-typed) expression.