Topics in Program Analysis

Separation Logic Assignment

February 16, 2016

The aim of this assignment is to make the motivation for separation logic more explicit, and to help gain some hands-on experience in writing separation logic proofs.

1 Problem 1

This problem explores the separation logic operations in some detail.

1a.

For each of the formulas below, state whether each is satisfiable or unsatisfiable, and provide a brief explanation of your answer.

1. $(x \mapsto c) * (x \mapsto c)$
2. $(x \mapsto c) \wedge (x \mapsto c)$
3. $(x \mapsto c) * !(x \mapsto c)$
4. $(x \mapsto c) \wedge !(x \mapsto c)$

1b.

Recall that we had a definition $E \to F = \texttt{true} * (E \mapsto F)$. Define $\mapsto$ in terms of the operators $\to$, $\wedge$, $*$, $!$ and `emp`.

2 Problem 2

One of the key motivations for the use of separation logic is the ease of defining data structures recursively. For example, a heaplet has the structure of a binary tree iff it satisfies the formula below

$$\begin{aligned} \mathit{tree}(E) \iff & \mathit{if}\ \texttt{isAtom}(E)\ ?\ \mathit{then}\ \texttt{emp} \\ & \mathit{else}\ \exists x,\ y:\ E[l:x, r:y] * \mathit{tree}(x) * \mathit{tree}(y) \end{aligned}$$

Here, `isAtom` is a predicate which returns `False` if the argument is an valid, or allocated, memory location, and returns `True` if the input is the special character `null`.

```
 1: procedure DispTree(p)
 2:{
 3:     Tree x, y;
 4:     if (p != null)
 5:     {
 6:             x = p->right;
 7:             y = p->left;
 8:             DispTree(x);
 9:             DispTree(y);
10:             free(p);
11:     }
```

Figure 1: Deleting a Binary Tree

```
1: procedure CopyTree1(Tree p, Tree q)
2: {
3:     q = p;
4: }
```

Figure 2: Copying a Binary Tree

1. Assume you are in a universe where separation logic hasn't been invented yet. In such a universe, how would you write the formula *tree*? In other words, you need to rewrite the formula *tree* without the aid of the separating conjunct *. You can assume the existence of other necessary predicates.

2. Consider the program in Fig. 2. Prove that the program satisfies

$$\{tree(p)\}\,\texttt{DispTree(p)}\,\{emp\}$$

   Note that the there is a recursive procedure call, and you need to use the specification of `DispTree` as part of your proof.
   Errata. Note that in Fig. 2, there is should be an additional line number `12` containing `"}"`.

3. Consider the program in Fig. 2. Does it satisfy

$$\{tree(p)\}\,\texttt{CopyTree1(p)}\,\{tree(p) * tree(q)\}?$$

   If yes, then write the proof. If no, then make appropriate changes to the `CopyTree1` procedure such that the post condition indeed holds, and then write the proof.

4. In the `CopyTree1` problem, do you observe any drawbacks in the proof (more precisely, in the pre- and post-conditions)?