Chapter 1

Background on separation logic

In this chapter we give a background on separation logic by specifying and proving the correctness of a program on marked linked lists (MLLs), as seen in **??**.

We will specify and proof deletion of an index in a marked linked list step by step. Thus, we will proof that delete defined in section 1.4, conforms to the specification below.

$$\{\mathsf{isMLL}\ hd\ vs\}\ \text{delete}\ hd\ i\ \{\mathsf{isMLL}\ hd\ (\mathsf{remove}\ i\ vs)\}$$

We will start by looking at separation as we will use it in the rest of this thesis in section 1.1. Next we show how to use separation logic to create hoare triples in the form of weakest preconditions to give a specification of a program in section 1.2. Then, we will focus on how to create predicates that represent recursive data structures using fixpoints in section 1.3. Lastly we will finish the specification of a program manipulating marked linked lists in section 1.4.

1.1 Separation logic

$$\begin{aligned} \tau &::= T \mid 0 \mid 1 \mid \tau \to \tau \\ t, P &::= x \mid \mathsf{False} \mid \mathsf{True} \mid t =_\tau t \mid P \wedge P \mid P \vee P \mid P * P \mid P \mathrel{-\!\!*} P \mid \\ &\quad \exists x : \tau.\, P \mid \forall x : \tau.\, P \mid \Box\, P \mid \triangleright P \mid \{P\}\, t\, \{v.\, P\} \end{aligned}$$

- We want to be able to reason about memory, heaps
- Use a logic that has extra connectives for talking about memory

- Most important, points to, $\mapsto$
- Picture of memory with $l \mapsto x$ next to it
- $l \mapsto x$ means there is only one location in memory, l and it has value x
- $\wedge$ now no longer works
- introduce *
- Another picture with logic next to it
- Describe rules of *

$$\begin{aligned} \mathsf{True} * P &\dashv\vdash P \\ P * Q &\vdash Q * P \\ (P * Q) * R &\vdash P * (Q * R) \end{aligned}$$

$$\frac{P_1 \vdash Q_1 \qquad P_2 \vdash Q_2}{P_1 * P_2 \vdash Q_1 * Q_2}\ {*\text{-MONO}}$$

- This does not include $P \vdash P * P$

1.2 Writing specifications of programs

- We will write our programs in heaplang, …
- Start by verifying simple program

```
Definition copy_singleton : val :=
  λ: "l", let: "x" := !"l" in
          SOME (Alloc "x").
```

- We now want to specify what happens to the memory when the program executes
- Use hoare triples

```
Lemma copy_singleton_spec (x : val) (l : loc) :
  {{{ l ↦ (x, NONE) }}}
    copy_singleton #l
  {{{ v, RET v; v ↦ (x, NONE) * l ↦ (x, NONE) }}}.
```

- Start with a description of the memory before copy singleton, precondition
- program to execute,
- `#l` transforms a location into a value in heaplang
- describe post condition

1.3 Representation predicates

- Describe more parts of the logic, like persistent, later and magic wand

1.4 Proof of delete in MLL

$$
\begin{array}{l}
\textbf{rec}_{\mathbf{v}}\ \text{delete}(l, i) = \textbf{match}\ l\ \textbf{with} \\
\quad \textbf{none} \Rightarrow \textbf{none}\ | \\
\quad \textbf{some}\ hd \Rightarrow \\
\quad\quad \textbf{let}\ (x, mark, tl) = !\, hd\ \textbf{in} \\
\quad\quad \textbf{if}\ mark = \textbf{false} \wedge i = 0\ \textbf{then} \\
\quad\quad\quad hd \leftarrow (x, \textbf{true}, tl) \\
\quad\quad \textbf{else if}\ mark = \textbf{false}\ \textbf{then} \\
\quad\quad\quad hd <- (x, \textbf{false}, \text{delete}\ tl\ (i-1)) \\
\quad\quad \textbf{else} \\
\quad\quad\quad hd <- (x, \textbf{true}, \text{delete}\ tl\ (i)) \\
\textbf{end}
\end{array}
$$