Test Formulae Approach

Alessio Mansutti

Barbizon 2018
Memory states

A **memory state** is a pair \((s, h)\) where:

- \(s : \text{VAR} \rightarrow \text{LOC}\) is called store;
- \(h : \text{LOC} \rightarrow_{\text{fin}} \text{LOC}\) is called heap.

where \(\text{VAR} = \{ x, y, z, \ldots \}\) set of (program) variables;
\(\text{LOC}\) set of locations (typically \(\text{LOC} \cong \mathbb{N} \cong \text{VAR}\)).

**Generalisation:**
\(h\) could be any finite graph

Note: Memory states are the standard model in Separation Logic.
Splitting a Heap

$h = h_1 + h_2$ whenever

- $\text{Dom}(h_1) \cap \text{Dom}(h_2) = \emptyset$;
- $h$ is the sum of the two functions $h_1$ and $h_2$. 
What we want? To build Test Formulae

- Fix $\mathcal{X} \subseteq_{\text{fin}} \text{VAR}$ and let $n \in \mathbb{N}$;
- Test$_{\mathcal{X}}(n)$ definable finite set of sets of memory states
  - $\{(s, h) \mid \text{in } h \text{ there is a path from } s(x) \text{ to } s(y)\}, x, y \in \mathcal{X}$;
  - $\{(s, h) \mid h \text{ has a loop}\}$.

  or, equivalently Test$_{\mathcal{X}}(n)$ finite set of predicates and their semantics.

**Indistinguishability relation** $(s, h) \approx_n (s', h')$

- holds whenever $\forall T \in \text{Test}_{\mathcal{X}}(n), (s, h) \in T \iff (s', h') \in T$;
- Property: for all $n, m \in \mathbb{N}$, if $m \geq n$ then $\approx_m \subseteq \approx_n$. 
EF-style Game

Spoiler chose two structures \((s, h)\) and \((s', h')\), and \(n \in \mathbb{N}\) resources so that \((s, h) \approx_n (s', h')\). Then the games continue as follows:

- If \((s, h) \not\approx_n (s', h')\) then Spoiler wins;
- If \((s, h) \approx_n (s', h')\) and \(n = 1\) then Duplicator wins;
- Otherwise,
  - Spoiler chooses \(n_1, n_2 \in \mathbb{N}\) so that \(n = n_1 + n_2\) and two heaps \(h_1, h_2\) so that \(h = h_1 + h_2\);
  - Duplicator chooses two heaps \(h'_1, h'_2\) so that \(h' = h'_1 + h'_2\);
  - Spoiler chooses \(i \in \{1, 2\}\). The game continues on the structures \((s, h_i)\) and \((s', h'_i)\), with \(n_i\) resources.
EF-style Game

Spoiler chose two structures \((s, h)\) and \((s', h')\), and \(n \in \mathbb{N}\) resources so that \((s, h) \approx_n (s', h')\). Then the games continue as follows:

- If \((s, h) \not\approx_n (s', h')\) then Spoiler wins;
- If \((s, h) \approx_n (s', h')\) and \(n = 1\) then Duplicator wins;
- Otherwise, Spoiler chooses \(n_1, n_2 \in \mathbb{N}\) so that \(n = n_1 + n_2\) and two heaps \(h_1, h_2\) so that \(h = h_1 + h_2\);
- Duplicator chooses two heaps \(h'_1, h'_2\) so that \(h' = h'_1 + h'_2\);
- Spoiler chooses \(i \in \{1, 2\}\). The game continues on the structures \((s, h_i)\) and \((s', h'_i)\), with \(n_i\) resources.

**Problem:**
Given \(\chi(1)\), find sufficient conditions on \(\chi(n)\), for all \(n \in \mathbb{N}\), so that Duplicator has a winning strategy.
Example: A family that works

Given \( n \in \mathbb{N} \), let

- \( \#\text{loops}(\beta) \geq \beta' \) be the set

\[
\{(s, h) \mid h \text{ with at least } \beta' \text{ loops of size } \beta \leq n\}
\]

- \( \#\text{loops}^\uparrow \geq \beta' \) be the set

\[
\{(s, h) \mid h \text{ with at least } \beta' \text{ loops of size } n + 1\}
\]

- \( \text{garbage } \geq \beta \) the set

\[
\{(s, h) \mid \text{ in } \text{Dom}(h) \text{ at least } \beta \text{ locations are not part of any loop}\}
\]
Given $n \in \mathbb{N}$, let

- $\#\text{loops}(\beta) > \beta'$ be the set

Defining $\text{Test}_X(n)$ as

$$\left\{ \begin{array}{l}
\#\text{loops}(\beta) \geq \beta', \ #\text{loops}^\uparrow \geq \beta', \\
garbage \geq \beta
\end{array} \right\} \quad \beta \in [1, n]
$$

$$\beta' \in \left[ 1, \frac{1}{2}n(n + 3) - 1 \right]$$

Guarantees a strategy for Duplicator.

$$\{(s, h) \mid \text{in } \text{Dom}(h) \text{ at least } \beta \text{ locations are not part of any loop}\}$$