Yosef Goren

November 29, 2022

Question 1

Question 2

Let $P \in FPG'$ (extended Flowchart Programming Languge - with interrupts). And let $CFG : FPL \longrightarrow [n] \times [n]^2$ be a function returning the control flow graph any program $P \in FPG$ (unmodified).

Define $CFG' : FPL' \longrightarrow [n] \times [n]^2$ on input (P, P_{int}) to be the graph $CFG(P)$ with the following modifications: Any node $v \in P$ now has an additional output goes into the initial state of a private copy of $CFG(P_{int})$ (each $v \in P$ has it's own copy); Denote this private copy with S_v. Additionally, all final states of S_v go into a new node F_v, which goes back into v.

Define a reachability condition R and state transformation T for any path τ of FPL' program graphs as follows:

- Denote $\tau = l_{i_0}, l_{i_1}, ..., l_{i_k}$.
- Denote the state transformation of P_{int} (from it's initial to final state - or the one appended to it to be exact) with T_{int}.
- Let:
$$Option_{int}(S) = S_\tau(\bar{x}) \cup \{T_{int}(s) \mid s \in S_\tau(\bar{x}) \wedge q_{int}(s)\}$$
Inituitively, this transformation expands the set of states possible by possibly applying the state transformation of P_{int} to all states.
- Base case:
$$T_\tau^k(\bar{x}) = \{(\bar{x})\}, R_\tau^k(\bar{x}) = true$$
Note how now the transformation function outputs a set rather than a single state. This is meant to capture the indetermenism of the interrupt; it represents the set of all possible states that can result from an initial (input) state.
- Inductive case: Given the functions have been defined for $l_{i_{m+1}}$, define them over what is at label l_{i_m}:

– *start* or *end*:

$$T_\tau^m(\bar{x}) = Option_{int}(T_\tau^{m+1}(\bar{x}))$$
$$R_\tau^m(\bar{x}) = R_\tau^{m+1}(\bar{x})$$

– $\bar{x} := \bar{y}$:
WLOG we can assume all assignments are to all variables - when this is not the case, we can append identity assignments.

$$T_\tau^m(\bar{x}) = Option_{int}(T_\tau^{m+1}(\bar{y}))$$
$$R_\tau^m(\bar{x}) = R_\tau^{m+1}(\bar{y})$$

– $B(\bar{x})$ (boolean branch expression):

$$T_\tau^m(\bar{x}) = Option_{int}(T_\tau^{m+1}(\bar{x}))$$

$$R_\tau^m(\bar{x}) = \begin{cases} R_\tau^{m+1}(\bar{x}) \land [\exists s \in B(T_\tau^{m+1}(\bar{x}))] & \text{if } l_{i_m} \to^T l_{i_{m+1}} \\ R_\tau^{m+1}(\bar{x}) \land [\exists s \in \neg B(T_\tau^{m+1}(\bar{x}))] & \text{if } l_{i_m} \to^F l_{i_{m+1}} \end{cases}$$

Inituitively; we require to get to the conditional label, and then have the boolean condition satisfiable by one of the possible states.

Now we define a modified floyed proof system which follows the steps on input (P, P_{int}):

1. Choose a set of cut points s.t.:
   - The set contains all initial and final states.
   - Every cycle in the graph $CFG'(P, P_{int})$ contains at least one cut point.
   - For every cut point l find an inductive assertion $I_l(\bar{x})$. Additionally it is required that: $I_{l_0}(\bar{x}) = q_1(\bar{x}), I_{l_*}(\bar{x}) = q_2(\bar{x})$ where l_0 is the initial state and l_* is a terminal state.
   - For every simple path between two cut points l_{i_m}, l_{i_j},

$$[I_{l_{i_m}}(\bar{x}) \land R_\tau^m(\bar{x}) \to I_{l_{i_j}}(\bar{x})]$$

The proof system is sound similarly to the original one; after the conditions have been shown - we know that for any path from l_0 to l_*, $I_{l_0}(\bar{x}) \to I_{l_*}(\bar{x})$ from closure on transitivity of the cut points conditions.
Since the initial and final conditions are the same as $q_1(\bar{x})$ and $q_1(\bar{x})$ - we have '$\{q_1(\bar{x})\}(P, P_{int})q_1(\bar{x})$'.

The new proof system is also reasonably complete, as haveing an interrupt which an [*false*] predicate would mean our proof system is equivalent to the original one.

Question 3