

Definition 1 (Execution). Let π be an error trace of length n . An execution of π is a sequence of states $s_0, s_1 \dots s_n$ such that $s_i, s_{i+1} \models T$, where T is the transition formula of $\pi[i]$.

Definition 2 (Blocked Execution). An execution of a trace π of size n is called a blocked execution, if there exists a sequence of states $s_0, s_1 \dots s_j$ where $i < j \leq n$ such that $s_i, s_{i+1} \models T$ where T is the transition formula of $\pi[i]$ and there exists an assume statement in the trace π at position j such that $s_j \not\models \text{guard}(\pi[j])$

Definition 3 (Relevant Statement). Let $\pi = st_1, \dots, st_n$ be an error trace of length n where st_i is an assignment statement of the form $x := t$. The assignment statement at position i is relevant if there exists an execution $s_1, \dots s_{n+1}$ of π and some value v such that every execution of the trace $x := v; \pi[i+1, n]$ starting in s_i has a blocked execution.

Lemma 1. For a program statement st and predicates P and Q , where P is condition that is true before the execution of the statement and Q is a post condition, the following two implications are equivalent (also known as the duality of WP and SP):

$$SP(P, st) \Rightarrow Q$$

$$P \Rightarrow WP(Q, st)$$

Lemma 2. For a predicate Q and an assignment statement of the form $x := t$ where x is a variable and t is an expression, we have:

$$WP(Q; \text{havoc}(x)) \subseteq WP(Q; x := t)$$

and

$$SP(P; x := t) \subseteq SP(P; \text{havoc}(x))$$

Lemma 3 (nonempty post). If $P := WP(Q, x := t) \not\subseteq WP(Q, \text{havoc}(x))$ for some Q then $Q \subsetneq SP(P, \text{havoc}(x))$.

Proof. We will show that $SP(P; x := t) \subseteq SP(P; \text{havoc}(x)) \not\subseteq SP(P; x := t)$ from which it follows that the first inclusion is strict. The first inclusion is from Lemma 2. It is obvious that a state reachable after $x := t$ is also reachable after $\text{havoc}(x)$. Hence $SP(P; x := t) \subseteq SP(P; \text{havoc}(x))$.

By assumption $WP(Q; x := t) \not\subseteq WP(Q, \text{havoc}(x))$, which by Lemma 1 is equivalent to.

$$SP(WP(Q; x := t); \text{havoc}(x)) \not\subseteq Q$$

or

$$SP(P; \text{havoc}(x)) \not\subseteq SP(P; x := t)$$

□

Theorem 1 (Relevancy of an assignment statement). Let π be an error trace of length n and $\pi[i]$ be an assignment statement at position i having the form $x := t$, where x is a variable and t is an expression. Let P and Q be two

predicates where $P = \neg WP(\text{False}; \pi[i, n]) \cap SP(\text{True}; \pi[1, i-1])$ and $Q = \neg WP(\text{False}; \pi[i+1, n])$. The statement $\pi[i]$ is relevant iff:

$$P \not\Rightarrow WP(Q, \text{havoc}(x))$$

Proof. Let $P' = WP(Q; \text{havoc}(x)) \cap SP(\text{True}; \pi[1, i-1])$ and $Q' = SP(P; \text{havoc}(x))$.

It is obvious that P can also be written as $WP(Q; x := t) \cap SP(\text{True}; \pi[1, i-1])$ and Q as $SP(P; x := t)$.

" \Rightarrow "

If $\pi[i]$ is relevant, then

$$P \not\Rightarrow WP(Q; \text{havoc}(x))$$

Obviously all the transition from P' end up in Q . Relevancy of $x := t$ implies that there is a state in $s \in P$ such that there is a transition from s to $\neg Q$. That would mean:

$$P \not\Rightarrow P'$$

$$P \not\Rightarrow WP(Q; \text{havoc}(x))$$

" \Leftarrow "

If $P \not\Rightarrow WP(Q; \text{havoc}(x))$, then the assignment statement $x := t$ is relevant.

By the definition of P :

$$WP(Q; x := t) \not\Rightarrow WP(Q; \text{havoc}(x))$$

By lemma 3:

$$Q \subsetneq SP(P; \text{havoc}(x))$$

or

$$Q \subsetneq Q'$$

This shows the existence of a state s in Q' such that $s \in \neg Q$ and hence a value v for x such that if we replace $x := t$ with $x := v$, then every execution is becoming blocking.

□