

# Fault Localization & Relevance Analysis

Matthias Heizmann, Christian Schilling, Numair Mansur

June 7, 2017

**Definition 1** (Execution). *Let  $\pi$  be an error trace of length  $n$ . An execution of  $\pi$  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  $\pi[i]$ .*

*Let  $\epsilon$  represent the set of all possible executions of the error trace.*

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

**Definition 3** (Relevancy of an assignment statement). *Let  $\beta$  represent the set of all blocking executions of a trace  $\pi$ . Let there be an assignment statement of the form  $x := t$  at position  $i$ . Let  $\pi'$  represent the trace that we get after replacing  $\pi[i]$  with a havoc statement of the form  $\text{havoc}(x)$  and let  $\beta'$  represent the set of all blocking executions for  $\pi'$ .*

*We say that the assignment statement  $\pi[i]$  is relevant if the trace after the replacement has strictly more blocked executions than the trace before the replacement, i.e if  $\beta \subsetneq \beta'$ .*

**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 a statement  $st$  which is an assignment statement the following implication holds:*

$$WP(\neg Q, st) = \neg WP(Q, st)$$

**Lemma 3.** *If a set of states have some states from which if we begin the execution of a trace and that are blocking then*

$$SP(P; \text{trace}) \not\models \text{guard}(\pi[j])$$

*Think about it more !*

**Lemma 4** (nonempty post). *If  $P := WP(Q, x := t) \not\subseteq WP(Q, havoc(x))$  for some  $Q$  then  $Q \subsetneq SP(P, havoc(x))$ .*

*Proof.* We will show that  $Q \equiv SP(P, x := t) \subseteq SP(P, havoc(x)) \subsetneq Q$  from which it follows that the first inclusion is strict. The first inclusion is immediate from Lemma 3 (main document)<sup>1</sup>. By assumption  $P \not\subseteq WP(Q, havoc(x))$ , which by Lemma 1 (main document) is equivalent to the second part.  $\square$

---

<sup>1</sup>We have to add it!

**Theorem 1** (Relevancy). *Let  $\pi$  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  $P = \neg WP(\text{False}; \pi[i, n])$  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.* We will divide the proof in 3 cases.

**Case 1:**  $\text{len}(\pi) = 2$ :

Suppose we have an error trace of length  $n = 2$ , where  $\pi[0]$  is an assignment statement of the form  $x := t$  and  $\pi[1]$  is an assume statement where  $\text{guard}(\pi[1])$  is the guard of the assume statement. If we consider the assignment statement  $\pi[0]$ ,  $P$  will be  $\neg WP(\neg \text{guard}; x := t)$  and  $Q$  will be  $\text{guard}(\pi[1])$ .  
 $\Rightarrow$

If the assignment statement  $\pi[0]$  is relevant, then:

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

The relevancy of  $\pi[0]$  implies that replacing  $\pi[0]$  with  $\text{havoc}(x)$  will result in more blocking executions than before.  
i.e

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

Intuitively the above statement means that if we replace  $x := t$  with  $\text{havoc}(x)$  and it creates more blocking executions, then  $SP(P; \text{havoc}(x))$  will have more states than in  $Q$ , while  $Q$  being  $\text{guard}(\pi[1])$  here.

so,

$$SP(P; \text{havoc}(x)) \not\Rightarrow \text{guard}(\pi[1])$$

By lemma 1

$$P \not\Rightarrow WP(\text{guard}(\pi[1]); \text{havoc}(x))$$

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

$\Leftarrow$

If

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

then the assignment statement  $\pi[0]$  is relevant.

Substituting the values for  $P$  and  $Q$

$$\neg WP(\neg \text{guard}(\pi[1]), x := t) \not\Rightarrow WP(\text{guard}(\pi[1]), \text{havoc}(x))$$

From lemma 3

$$WP(\text{guard}(\pi[1]), x := t) \not\Rightarrow WP(\text{guard}(\pi[1]), \text{havoc}(x))$$

From the "duality of WP and SP" (lemma 1), if in the above expression, we consider the right hand side of the implication as  $P$ ,  $\text{guard}(\pi[1])$  as  $Q$  and  $\text{havoc}(x)$

as  $st$ , then we can write the above implication as:

$$SP(WP(\text{guard}(\pi[1]), x := t), \text{havoc}(x)) \not\Rightarrow \text{guard}(\pi[1])$$

That means that from our supposed trace which contains one assignment statement and one assume statement with error precondition  $WP(\text{guard}(\pi[1]), x := t)$ , if we get a new trace where the assignment  $x := t$  is replaced by  $\text{havoc}(x)$ , the strongest postcondition of the error precondition and  $\text{havoc}(x)$  does not imply the guard of the assume statement. Which intuitively means that replacing the assignment with havoc introduces new states which does not satisfy the guard of the assume statement. These new states also means that the number of blocking executions for the trace with  $\text{havoc}$  is more then the number of blocking executions in the trace with assignment. Which according to our definon would mean that the assignment is relevant.

**Case 2:**  $\text{len}(\pi) = n$ ;  $\pi[i]$  is an assignment statement, where  $i = 0$ :

Consider the case where the length of the trace is  $n$  and the first statement  $\pi[0]$  of the trace  $\pi$  is an assignment statement. Let  $\pi_s := \pi[1, n-1]$ . If we consider the assignment statement  $\pi[0]$ , then  $P := \neg WP(\text{false}; \pi)$ ,  $Q := \neg WP(\text{false}, \pi_s)$ . Let  $\pi'$  be the trace where  $\pi[0]$  is replaced by a havoc statement.

Let  $P' := \neg WP(\text{false}; \pi') := WP(Q; \text{havoc}(x))$  and  $Q' := SP(P; \text{havoc}(x))$ . Observe here that  $P$  can also be stated as  $WP(Q; x := t)$ .

" $\Rightarrow$ "

If the assignment statement  $\pi[0]$  is relevant, then:

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

i.e the trace after the replacement of  $\pi[0]$  to  $\text{havoc}$  has strictly more blocking executions than the trace before the replacement. That means that there exists an assume statement in the trace  $\pi$  at position  $j$ , which is blocking more executions then before. Or we can say that there are more states  $s_j$  now for which that assume statement  $\pi[j]$  is blocking.

OR

$$SP(P; \pi[0, j-1]) \subsetneq SP(P; \pi'[0, j-1])$$

$SP(P; \pi'[0, j-1])$  contains more states for which  $\text{guard}(\pi'[j])$  does not hold.

Hence,

$$SP(P; \pi'[0, j-1]) \not\Rightarrow \text{guard}(\pi'[j])$$

From lemma 1:

$$P \not\Rightarrow WP(\text{guard}(\pi'[j]; \pi'[0, j-1]))$$

OR

$$P \not\Rightarrow WP(\text{guard}(\pi'[j]); \pi'[0], \pi'[1, j-1])$$

By the recursive definition of  $WP()$

$$P \not\Rightarrow WP(WP(\text{guard}(\pi'[j]); \pi'[1, j-1]); \pi'[0]) \quad (1)$$

We also know that: (NO WE DON'T KNOW THAT ANYMORE)  
According to our new definition, the trace might already contain blocking execution. In that case the following implication cannot hold anymore.

$$SP(Q; \pi'[1, j-1]) \Rightarrow guard(\pi'[j])$$

and consequently from lemma (1):

$$Q \Rightarrow WP(guard(\pi'[j]); \pi'[1, j-1]) \quad (2)$$

Let  $WP(guard(\pi'[j]); \pi'[1, j-1]) := R$   
Then (5) and (6) can be written as:

$$P \not\Rightarrow WP(R; \pi'[0])$$

$$Q \Rightarrow R$$

From Lemma (4)

$$P \not\Rightarrow WP(Q; havoc(x)) \quad (3)$$

THIS IS NOT CORRECT. CHANGE THIS !!

" $\Leftarrow$ "

If

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

then the assignment statement  $\pi[0]$  is relevant.

By the definition of  $P$ :

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

By lemma 4 :

$$Q \subsetneq SP(Q; havoc(x))$$

or

$$Q \subsetneq Q'$$

Let  $R = Q' \setminus Q$  or  $Q' = R \uplus Q$  (disjoint union of  $R$  and  $Q$ ).

Now we want to show a contradiction for the states in  $Q'$  but from

$$SP(X \cup Y; \pi) = SP(X; \pi) \cup SP(Y; \pi) \quad (4)$$

we can conclude that it suffices to show a contradiction for the states in  $R$ .

——I THINK HERE WE HAVE TO ASSUME THAT THE NEW TRACE  
HAVE THE SAME AMOUNT OF BLOCKING EXECUTIONS——

---

Let us assume that for all  $1 \leq j \leq n$ , statement  $\pi'[j]$  is not restrictive. i.e

$$SP(Q'; \pi'[1, j-1]) \Rightarrow guard(\pi'[j]) \quad \forall 1 \leq j \leq n$$

---

By equation (8) we can write:

$$SP(R; \pi'[1, j-1]) \Rightarrow guard(\pi'[j]) \quad \forall 1 \leq j \leq n \quad (5)$$

We know that  $R \Rightarrow \neg Q$  and hence

$$R \Rightarrow WP(false; \pi_s) \quad (6)$$

Considering (9) and (10) and lemma (6) we get a contradiction. That means that one of the assume statement is getting restrictive and hence the statement  $x := t$  is relevant.  $\square$

**Definition 4** (Relevancy of a havoc statement). *Let  $\pi$  be a feasible error trace and  $\pi[i]$  be a Havoc statement at position  $i$  having the form  $havoc(x)$ , where  $x$  is a variable. Let  $x := t$  be an assignment statement where  $x$  is the same variable as in  $havoc(x)$  and  $t$  is an expression. The havoc statement  $\pi[i]$  is relevant if there exists an assignemnt  $x := t$  such that if we replace the havoc in  $\pi$  with  $x := t$  to get a new trace  $\pi'$ , then all the executions of  $\pi'$  are blocking (the trace is infeasible).*

$$SP(true; \pi') \Rightarrow false$$

**Theorem 2** (Security Error Relevancy). *Let  $\pi$  be an error trace of length  $n$  and  $\pi[i]$  be a non-deterministic assignment statement at position  $i$  having the form  $havoc(x)$ , where  $x$  is a variable. Let  $P$  and  $Q$  be two predicates where  $P = \neg WP(False, \pi[i, n])$  and  $Q = \neg WP(False, \pi[i+1, n])$ . The statement  $\pi[i]$  is relevant iff:*

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

*Proof.* The same as before.  $\square$