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

Definition 2 (Blocking Execution). *An execution of a trace π of size n is called a blocking execution if there exists a sequence of states $s_0, s_1 ... s_j$ 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 exits an assume statement in the trace π at position j such that $s_j \not\Rightarrow guard(\pi[j])$.*

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

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

Lemma 3 (nonempty post). *If $P := WP(Q, x := t) \nsubseteq WP(Q, havoc(x))$ for some Q then $Q \subsetneq SP(P, havoc(x))$.*

Proof. We will show that $Q \equiv SP(P, x := t) \subseteq SP(P, havoc(x)) \nsubseteq Q$ from which it follows that the first inclusion is strict. The first inclusion is immediate from Lemma 2. By assumption $P \nsubseteq WP(Q, havoc(x))$, which by Lemma 1 is equivalent to the second part. □

Lemma 4 (restrictiveness). *If the last statement of a trace π of length n is an assume statement and $P \Rightarrow WP(\bot, \pi)$, then $SP(P, \pi[0, j-1]) \not\Rightarrow guard(\pi[j])$ for some $1 \leq j \leq n$.*

Proof. Induction over n (length of π): For $n = 1$ we have $WP(\bot, \pi[0]) \equiv guard(\pi[1]) \Rightarrow \bot \equiv \neg guard(\pi[1])$. Now let $n > 1$ and let $Q := WP(\bot, \pi[1, n])$. By induction hypothesis $SP(Q, \pi[1, j]) \not\Rightarrow guard(\pi[j])$ for some j. If $\pi[0]$ is an assignment or havoc statement we just apply the hypothesis. If $\pi[0]$ is an assume statement then $P \equiv guard(\pi[1]) \Rightarrow Q$, so if $P \not\Rightarrow Q$ then $P \not\Rightarrow guard(\pi[1])$. □

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

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

Proof. Let π' be the trace where the assignment statement $\pi[i]$ is replaced by a havoc statement.
Note that here we can also write P as $WP(Q; x := t) \cap SP(True; \pi[1, i-1])$.
Let $Q' := SP(P; havoc(x))$ and $P' := WP(Q; havoc(x)) \cap SP(True; \pi[1, i-1])$.
Since from lemma 2 , we know that

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

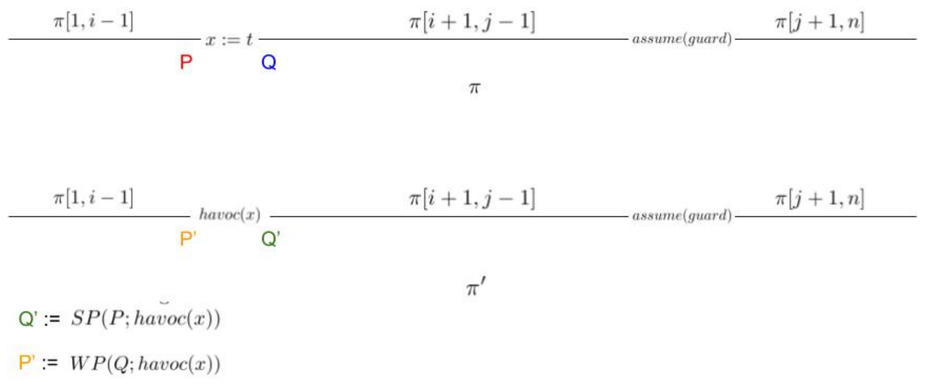
and also,

$$WP(Q; havoc(x)) \cap SP(True; \pi[1, i-1]) \subseteq WP(Q; x := t) \cap SP(True; \pi[1, i-1])$$

therefore:

$$P' \subseteq P$$

For simplicity in the proof, lets ignore the term $SP(True; \pi[1, i-1])$ from P and P'. We simplify P and P' to be $WP(Q; x := t)$ and $WP(Q; havoc(x))$ respectively.



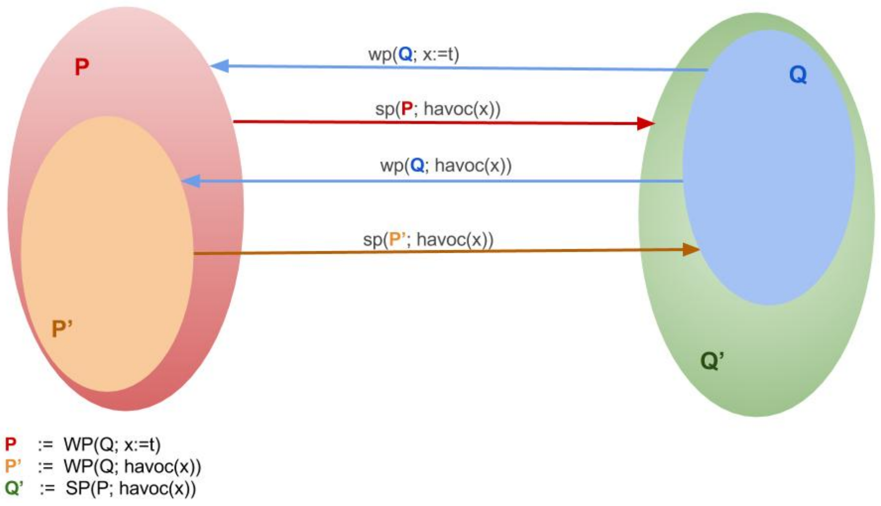
"$\Rightarrow$"
If the assignment statement $\pi[i]$ is relevant then:

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

Relevancy of $x := t$ implies that replacing it with $havoc(x)$ gives us strictly more blocking executions then before. Therefore

$$Q \subsetneq Q'$$

Lets look at the following diagram to help us see the states a little better and come to the following conculsions:



$$SP(P; havoc(x)) = Q'$$

$$SP(P'; havoc(x)) = Q$$

and we know that $Q \subsetneq Q'$. This means that $\exists S \in P \setminus P'$, such that there is a transition from S to Q' if we execute $havoc(x)$. The existence of the state S means:

$$P \not\Rightarrow P'$$

or

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

"$\Leftarrow$"
If

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

then the assignment statement $x := t$ at position i, is relevant.
By the definition of P:

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

By lemma 3:

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

or

$$Q \subsetneq Q'$$

Let $R = Q' \setminus Q$ or $Q' = R \biguplus Q$ (disjoint union of R and Q).
Now if we can show that there are states in R such that there is a blocking execution from those states, then that would mean that replacing the assignment with havoc have introduced new blocking execitions. We can show this fact by

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

$$SP(Q'; \pi[i; j-1]) \Rightarrow guard(\pi[j]) \qquad \forall i \leq j \leq n \tag{1}$$

We know that

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

So we can write 1 as:

$$SP(R; \pi[i; j-1]) \cup SP(Q; \pi[i; j-1]) \Rightarrow guard(\pi[j]) \qquad \forall i \leq j \leq n$$

We only have to show the contradiction on R

$$SP(R; \pi[i; j-1]) \Rightarrow guard(\pi[j]) \qquad \forall i \leq j \leq n \tag{2}$$

We know that $R \Rightarrow \neg Q$, or

$$R \Rightarrow WP(false; \pi[i+1, n]) \tag{3}$$

Considering equation(1) and (2) and lemma(4), we get a contradiction. That means that we must have atleast one execution which is blocking and hence the statement $x := t$ is relevant. $\square$