Verification witnesses [1, 2] accompany the results of a verifier and contain information about the verification process.

```c
int main(void) {
    int x = nondet();
    int y = x;
    int y = x + 1;
    while (x < 1024) {
        x++;
        y++;
    }
    assert(x == y);
}
```

Correctness witnesses encode invariants which aid in proving the program correct.

True Correctness Witness

Violation witnesses represent a set of paths which lead to a violation of the specification.

Verifiers participating in SV-COMP 2024; all export witnesses

**Validation**

Witness increase the confidence in the verification result by making its validation possible.

For example, LIV [5] creates a Hoare style proof from a witness. Which allows for independent validation of the verification result.

- \( \{P\} B \{I\} \) (Reachability)
  - \( \{P\} B \{I\} \) (Inductiveness)
  - \( \{I \wedge C\} B \{I\} \) (Safety)
  - \( \{C \wedge I\} Q \) (Safety)

**Cooperation**

Witnesses provide a common interface to exchange information between formal methods tools and theories.

<table>
<thead>
<tr>
<th>Source</th>
<th>Communication Methods</th>
<th>Receiver</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic Verification</td>
<td>Interactive Theorem Proving</td>
<td>Understanding Witnesses</td>
<td></td>
</tr>
<tr>
<td>Automatic Verification</td>
<td>Deductive Verification</td>
<td>Programs with only Loops</td>
<td></td>
</tr>
<tr>
<td>Automatic Verification</td>
<td>Manual Proofs</td>
<td>Hoare style proofs; Programs with only Loops</td>
<td></td>
</tr>
<tr>
<td>Automatic Verification</td>
<td>Unknown</td>
<td>Interactive Theorem Proving</td>
<td></td>
</tr>
<tr>
<td>Deductive Verification</td>
<td>Unknown</td>
<td>Interactive Theorem Proving</td>
<td></td>
</tr>
</tbody>
</table>

**References**
