How to Write Postconditions with Multiple Cases

Wolfgang Schreiner
Research Institute for Symbolic Computation (RISC)
Johannes Kepler University, Linz, Austria

November 19, 2009

Abstract

We investigate and compare the two major styles of writing program/function postconditions with multiple cases: as conjunctions of implications or as disjunctions of conjunctions. We show that both styles not only have different syntax but also different semantics and pragmatics and give recommendations for their use.

The specification of a program/function $F$ typically consists of two parts: a precondition $I$ on the input (state) and a postcondition $O$ that relates the input (state) of the program/function to its output (state). The requirement on the correctness of the program is then

$$\forall x, y : I(x) \land y = F(x) \Rightarrow O(x, y)$$

i.e. for any input $x$ satisfying $I$ (we call this a legal input) the function $F$ must return a result $y$ which is related to $x$ by $O(x, y)$.

However, there may be different kinds of legal inputs, for which $F$ yields different kinds of output. Without loss of generality, let us assume, there are two kinds of inputs denoted by conditions $P_1$ and $P_2$ and correspondingly two kinds of outputs related to the inputs by conditions $Q_1$ and $Q_2$. Now there are two obvious choices: to define $O$, either as

$$O_1(x, y) : \iff (P_1(x) \Rightarrow Q_1(x, y)) \land (P_2(x) \Rightarrow Q_2(x, y))$$

or as

$$O_2(x, y) : \iff (P_1(x) \land Q_1(x, y)) \lor (P_2(x) \land Q_2(x, y))$$

Naturally, the question arises which of the two choices shall be preferred?

This question is apparently a problem of propositional logic (rather than predicate logic), thus we rewrite the choices as

$$O_1 : \iff (P_1 \Rightarrow Q_1) \land (P_2 \Rightarrow Q_2)$$

$$O_2 : \iff (P_1 \land Q_1) \lor (P_2 \land Q_2)$$
and depict in Figure 1 the truth values of $O_1$ and $O_2$ for all possible truth values of $P_1, P_2, Q_1, Q_2$ (there $F$ denotes the negation of condition $F$ and $FG$ denotes the conjunction of conditions $F$ and $G$). We see that both and $O_1$ and $O_2$ have different truth ranges and that no interpretation is stronger than the other one: only $O_1$ is true if both $P_1$ and $P_2$ are false and only $O_2$ is true, if both $P_1$ and $P_2$ and one of $Q_1$ and $Q_2$ are true.

One possibility to reconcile both interpretations is to restrict the truth range of both $O_1$ and $O_2$ to the second and third line of Figure 1, i.e., to those cases where exactly one of $P_1$ and $P_2$ is true:

$$\text{If } (P_1 \lor P_2) \land \neg(P_1 \land P_2) \Rightarrow O_1 \equiv O_2$$

In other words, if we demand that the conditions $P_1$ and $P_2$ decompose the space of legal inputs (those satisfying precondition $I$) disjointly, then both postconditions $O_1$ and $O_2$ are equivalent.

However, we may also explicitly add constraints to $O_1$ and $O_2$ such that the resulting interpretations coincide yielding the formulas

$$O_1 \land (P_1 \lor P_2)$$
$$O_2 \land \neg(P_1 \land P_2)$$

i.e. we either add to $O_1$ the demand that $P_1$ and $P_2$ must cover the whole space of legal inputs or add to $O_2$ the demand that $P_1$ and $P_2$ must not overlap. We then have

$$O_1 \land (P_1 \lor P_2) \iff O_2 \land \neg(P_1 \land P_2)$$
$$O_1 \land (P_1 \lor P_2) \Rightarrow O_2$$
$$O_2 \land \neg(P_1 \land P_2) \Rightarrow O_1$$

i.e. adding the constraints to both $O_1$ and $O_2$ yields equivalent results, adding the constraint to only one of $O_1$ or $O_2$ yields a result that is stronger than $O_2$ respectively $O_1$.

What is a consequence of above investigations?

1. If $P_1$ and $P_2$ do not decompose the space of legal inputs disjointly, then $O_1$ respectively $O_2$ should be extended by an additional constraint:

<table>
<thead>
<tr>
<th>$P_1P_2$</th>
<th>$\overline{Q}_1\overline{Q}_2$</th>
<th>$Q_1\overline{Q}_2$</th>
<th>$\overline{Q}_1Q_2$</th>
<th>$Q_1Q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1P_2$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
<td>$\times$</td>
</tr>
<tr>
<td>$P_1P_2$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td></td>
</tr>
<tr>
<td>$P_1P_2$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\times$</td>
</tr>
<tr>
<td>$P_1P_2$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

Figure 1: Truth Table ($\times$ for $O_1$, $\circ$ for $O_2$, $\otimes$ for both)
• $O_1$ should be extended by the constraint $P_1 \lor P_2$
• $O_2$ should be extended by the constraint $\neg(P_1 \land P_2)$

2. If $P_1$ and $P_2$ decompose the space of legal inputs disjointly, i.e. if we have

$$(P_1 \lor P_2) \land \neg(P_1 \land P_2)$$

then it is not necessary to add a constraint and both $O_1$ and $O_2$ are equivalent.

From this, it seems that none of $O_1$ or $O_2$ should be a priori preferred over each other. However, there are two reasons why the situation is actually not completely symmetric: First, in the case of $n$ condition pairs $P_i, Q_i$ ($i = 1 \ldots n$), the constraint for $O_1$ becomes

$$P_1 \lor P_2 \lor \ldots \lor P_n$$

(i.e. a disjunction of $n$ formulas) while the constraint for $O_2$ becomes

$$\neg(P_1 \land P_2) \land \neg(P_1 \land P_3) \land \ldots \land \neg(P_{n-1} \land P_n)$$

(i.e. a conjunction of $n \cdot (n - 1)$ formulas) which is cumbersome to write.

Second, assume a situation where a specifier erroneously believes that some conditions $P_i$ ($i = 1 \ldots n$) decompose the legal input space disjointly and thus does not add an explicit constraint:

1. In the case of $O_1$, for a legal input for which none of the $P_1$ holds, any output becomes legal (for $O_2$, no output is legal).

2. In the case of $O_2$, for a legal input for which multiple $P_i$ hold, the output must only satisfy any of the corresponding $Q_i$ (for $O_1$, all $Q_i$ must be satisfied).

The first kind of “underspecification” error is certainly more “dangerous” than the second one.

Taking these two considerations into account, we recommend:

1. Either to use the form $O_1$ and add explicit constraints as shown above:

$$(P_1 \Rightarrow Q_1) \land \ldots \land (P_n \Rightarrow Q_n) \land 
(P_1 \lor P_2 \lor \ldots P_n)$$

2. (Only) if the constraints seem redundant and explicitly adding them seems too cumbersome, use $O_2$:

$$(P_1 \land Q_1) \lor \ldots \lor (P_n \land Q_n)$$

However, then one should be aware, that from this specification, $Q_i$ is not a consequence of $P_i$ (for $i = 1 \ldots n$).

If nothing else, above discussion should at least have clarified the features/differences of both specification formats.