Methods for Deriving Auxiliary Invariants

The methods for deriving auxiliary invariants (which can be used to strengthen a non-inductive assertion) can be partitioned into

- Bottom-Up methods. Analyze the program independently of the goal assertion to be proven.
- Top-Down methods. Take into account both the program and the assertion whose invariance we wish to prove.

The successive strengthening method we have previously described, using the TLV tool, is a typical top-down method.

We will proceed to describe additional methods of each of the classes, starting with bottom-up methods.

Transition Affirmed Invariants

In some cases, we can identify that all transitions entering location ℓ, cause an assertion φ to hold in the post-state of the transition. If, in addition, no action of a parallel process can invalidate φ then the assertion

$$at\_\ell \quad \rightarrow \quad \varphi$$

is an invariant.

Following are some configurations of statements and the candidate assertions corresponding to them

Configuration	Candidate	Provided
$\left[\ell_i : \; y := f(\vec{x})\right]$	$at\_\ell_i \rightarrow y = f(\vec{x})$	$y \notin \vec{x}$
$\left[\ell_i : \; \textbf{await } c\right]$	$at\_\ell_i \rightarrow c$	
$\left[\ell_2 : \; \textbf{while } c \textbf{ do } \ell_1 : S\right]$	$\left(\begin{array}{ll} & at\_\ell_1 \rightarrow c \\ \wedge & at\_\ell_2 \rightarrow \neg c\end{array}\right)$	
$\left[\begin{array}{l}\textbf{if } c \;\; \textbf{then } \ell_1 : S_1 \\ \qquad \textbf{else } \ell_2 : S_2\end{array}\right]$	$\left(\begin{array}{ll} & at\_\ell_1 \rightarrow c \\ \wedge & at\_\ell_2 \rightarrow \neg c\end{array}\right)$	

Forward Propagation

Consider a program segment of the form $\ell_1 : y := e;\ \ell_2$, and assume that

- We previously derived an invariant $at\_\ell_1 \rightarrow \varphi$.
- The assignment $y := e$ preserves the assertion φ. For example, φ does not depend on y.
- No statement parallel to this process can invalidate φ.

Then, we can conclude that $at\_\ell_2 \rightarrow \varphi$ is also an invariant.

Example: Peterson's Mutual Exclusion for 2 Processes

$$
\begin{array}{l}
\textbf{local} \quad y_1, y_2 \ : \textbf{boolean where } y_1 = y_2 = 0 \\
\qquad\qquad s \quad\ \ : \{1,2\} \textbf{ where } s = 1 \\
P_1 :: \left[\begin{array}{l}\ell_0 : \textbf{loop forever do} \\ \left[\begin{array}{l}\ell_1 : \textbf{Non-Critical} \\ \ell_2 : (y_1, s) := (1,1) \\ \ell_3 : \textbf{await } y_2 = 0 \ \vee \ s \neq 1 \\ \ell_4 : \textbf{Critical} \\ \ell_5 : y_1 := 0\end{array}\right]\end{array}\right]
\ \| \
P_2 :: \left[\begin{array}{l}m_0 : \textbf{loop forever do} \\ \left[\begin{array}{l}m_1 : \textbf{Non-Critical} \\ m_2 : (y_2, s) := (1,2) \\ m_3 : \textbf{await } y_1 = 0 \ \vee \ s \neq 2 \\ m_4 : \textbf{Critical} \\ m_5 : y_2 := 0\end{array}\right]\end{array}\right]
\end{array}
$$

- Using the method of transition affirmed invariants, we can derive the invariant

$$at\_\ell_0 \rightarrow y_1 = 0 \qquad \wedge \qquad at\_\ell_3 \rightarrow y_1 > 0$$

Using forward propagation, we can extend this to

$$at\_\ell_{3..5} \iff y_1 > 0$$

- Applying the second clause of the transition affirmed invariants method to statement ℓ_3, we can derive the invariant

$$at\_\ell_4 \quad \rightarrow \quad y_2 = 0 \ \vee \ s \neq 1$$

This requires showing that no statement parallel to ℓ_4 can invalidate the assertion $y_2 = 0 \ \vee \ s \neq 1$. Special attention must be given to m_2 which modifies both y_2 and s. However, since it sets s to $2 \neq 1$, it only revalidates $y_2 = 0 \ \vee \ s \neq 1$.

Loop Derived Invariants

Consider the following loop:

$$\begin{array}{ll} \ell_j: & i := 1 \\ \ell_{j+1}: & \textbf{while } i \le n \textbf{ do} \\ & \left[\begin{array}{ll} \ell_{j+2}: & \cdots \\ & \cdots \\ \ell_k: & \cdots \\ \ell_{k+1}: & i := i+1 \end{array}\right] \\ \ell_{k+2}: & \cdots \end{array}$$

where none of the statements $\ell_{j+2}, \ldots, \ell_k$ and no statement parallel to this process modifies i.

Then, we can conclude the following invariant:

$$at\_\ell_{j+1..k+1} \to 1 \le i \le n + at\_\ell_{j+1} \quad \land \quad at\_\ell_{k+2} \to i = n+1$$

We can draw similar conclusions about the loop

$$\ell_{j+1}: \textbf{for } i = 1 \textbf{ to } n \textbf{ do } S;\ \ell_{k+2}:$$

Top-Down Derivation Methods: Generalization

Consider the following program:

$$\begin{array}{ll} \ell_0: & sum := 0 \\ \ell_1: & \textbf{for } i := 1 \textbf{ to } n \textbf{ do} \\ & \quad \ell_2: \quad sum := sum + A[i] \\ \ell_3: & \ldots \end{array}$$

for which we wish to prove the invariance of the assertion

$$\varphi: \quad at\_\ell_3 \ \to\ sum = \sum_{r=1}^{n} A[r]$$

Since we know that, at location ℓ_3, $i = n+1$, this can be rewritten as:

$$at\_\ell_3 \ \to\ i = n+1 \ \land\ sum = \sum_{r<i} A[r]$$

It is possible to generalize and conjecture the more general invariant

$$at\_\ell_{1..3} \ \to\ sum = \sum_{r<i} A[r]$$

This corresponds to the following insight:

> If the purpose of the complete loop is to compute the sum $A[1] + \cdots + A[n]$ and i measures the incremental progress, then it seems reasonable that, at an intermediate stage, sum should contain the partial sum $A[1] + \cdots + A[i-1]$.

Top-Down Methods: Systematic Strengthening

Premise I2 of rule INV requires establishing the validity of $\varphi \wedge \rho \rightarrow \varphi'$. As ρ consists of a disjunction $\bigvee_\ell \rho_\ell$, where each statement ℓ contributes its own transition relation ρ_ℓ, this is often established by showing separately

$$\varphi \ \wedge \ \rho_\ell \quad \rightarrow \quad \varphi'$$

for each statement ℓ. Equivalently, this can be written as $\varphi \rightarrow pre(\ell, \varphi)$, where $pre(\ell, \varphi) = \forall V' : (\rho_\ell \rightarrow \varphi')$.

In our case, all individual transition relations have the form $\rho_\ell : c_\ell \wedge V' = E_\ell$, where c_ℓ is a boolean expression over V, and E_ℓ is a set of expressions defining the new values of the variables V. For these cases, the pre-condition $pre(\ell, \varphi)$ can be simplified to

$$pre(\ell, \varphi) : \quad c_\ell \rightarrow \varphi(E_\ell),$$

where $\varphi(E_\ell)$ is obtained from φ by substituting the expressions E_ℓ for the state variables V.

Claim 4. *If the assertion φ is an invariant of system $\mathcal{D}$, the so is $pre(\ell, \varphi)$, for every statement ℓ.*

This claim leads to the following strengthening strategy:

Strategy 1. *If the verification condition $\varphi \wedge \rho_\ell \rightarrow \varphi'$ fails to be $\mathcal{D}$-valid, strengthen φ by conjuncting it with $pre(\ell, \varphi)$.*

Example of Applying the Strategy

Reconsider program PETERSON2. We may start the search for an invariant with the assertion of mutual exclusion

$$\varphi_0 : \quad \pi_1 \neq 4 \ \vee \ \pi_2 \neq 4$$

Checking the verification conditions, we find out that this assertion fails to be inductive after execution of the statements ℓ_3 and m_3. Observing that the enabling condition for ℓ_3 is $c_{\ell_3} : \pi_1 = 3 \ \wedge \ (y_2 = 0 \ \vee \ s \neq 1)$ and the variable assignment is $\pi_1 := 4$, we compute $pre(\ell_3, \varphi_0)$ and obtain:

$$\begin{aligned} \varphi_1 : \quad & \pi_1 = 3 \ \wedge \ (y_2 = 0 \ \vee \ s \neq 1) \quad \rightarrow \quad (4 \neq 4 \ \vee \ \pi_2 \neq 4) \quad \sim \\ & at\_\ell_3 \ \wedge \ at\_m_4 \quad \rightarrow \quad y_2 \neq 0 \ \wedge \ s = 1 \end{aligned}$$

In a similar way, $pre(m_4, \varphi_0)$ yields

$$\varphi_2 : \quad at\_\ell_4 \ \wedge \ at\_m_3 \quad \rightarrow \quad y_1 \neq 0 \ \wedge \ s = 2$$

Together with the bottom-up derived invariants

$$\varphi_3 : \quad at\_\ell_{3..5} \rightarrow y_1 = 1 \qquad\qquad \varphi_4 : \quad at\_m_{3..5} \rightarrow y_2 = 1,$$

This set of assertions is inductive and implies φ_0 which specifies mutual exclusion.