Available at the COMP2111 website.

COMP2111 Notes Week 10
How to Prove Forward Simulation

Kai Engelhardt

May 4, 2018

Revision: 1.5

1 Introduction

This note explains how to prove forward simulation without having to descend into the semantics of program snippets.

Consult slides 8 for the relevant definitions of the semantic notions of data type, data refinement, and forward simulation.

We develop the general case first and then specialise it to the (common) case where the correspondence between abstract and concrete level representation variables can be described by a (total) function from concrete to abstract representation variables.

2 Forward Simulation in Syntax

Next we can give syntactic versions of data types and refinement relations. We focus entirely on specification statements as operations because (a) those are expressive and (b) we should have those specifications anyway.

Definition 1 A *syntactic data type* is a tuple

$$\mathsf{B} = (\mathsf{init}^{\mathsf{B}}, (b, x : [\mathsf{pre}^{\mathsf{B}}_j, \mathsf{post}^{\mathsf{B}}_j])_{j \in J}) \tag{1}$$

where $\mathsf{init}^{\mathsf{B}}$ and the $\mathsf{pre}^{\mathsf{B}}_j$ are predicates over b, x and the $\mathsf{post}^{\mathsf{B}}_j$ are predicates over b_0, x_0, b, x, for $j \in J$.

Note how there's no finalisation. We omit it here because in this simple setup we fix all finalisations to be merely projections onto the normal variables. Let us denote that projection Π^{B}. It maps each state in $\sigma \in \Sigma^{\mathsf{B}}$ to the same state without the information about the representation variables, in this case b. Initialisations are given as predicates $\mathsf{init}^{\mathsf{B}}$. They work like simplified operations $b : [\textsc{true}, \mathsf{init}^{\mathsf{B}}]$ that have a trivial precondition and don't refer to old values.

Definition 2 The semantics $[\![\mathsf{B}]\!]$ of the syntactic data type (1) is a data type

$$\mathcal{B} = (BI, (B_j)_{j \in J}, BF) \text{ , where}$$

$$BI = \{\ (\Pi^{\mathsf{B}}(\sigma), \tau)\ \mid\ (\sigma,\tau) \in [\![b : [\textsc{true}, \mathsf{init}^{\mathsf{B}}]]\!]\ \}$$
$$B_j = [\![b, x : [\mathsf{pre}_j^{\mathsf{B}}, \mathsf{post}_j^{\mathsf{B}}]]\!] \ \text{, for } j \in J$$
$$BF = \{\ (\sigma, \Pi^{\mathsf{B}}(\sigma))\ \mid\ \sigma \in \Sigma^{\mathsf{B}}\ \}$$

The semantics of a syntactic data type contains no surprises: all we had to do is adapt state spaces and interpret specification statements.

What is somewhat inconvenient in the later proof work is that, as soon as a single operation has a non-trivial precondition (i.e., $\neq$ TRUE), its relational semantics maps every state for which the precondition fails to *all* final states. While this is great from a refinement point of view—in a refinement of such an operation we are free to do whatever we want for those initial states, for instance, introduce proper error handling or simply abort—it is rather annoying when we try to discharge the forward simulation proof obligation for this operation. It turns out that we need to ensure that the refinement relation matches some abstract state to each and every concrete state. In other words, when viewed as a relation from abstract states to concrete states, it must be *surjective* (AKA onto). In practical cases this is rather easy to achieve. Yet it is somewhat disappointing that our syntactic condition will feature a condition that is not needed in the purely semantic formulation presented in slides 8.

Definition 3 A predicate r over a, c, x is a *forward simulation predicate* between syntactic data type

$$\mathsf{A} = (\mathsf{init}^{\mathsf{A}}, (a, x : [\mathsf{pre}_j^{\mathsf{A}}, \mathsf{post}_j^{\mathsf{A}}])_{j\in J}) \ \text{, and}$$
$$\mathsf{C} = (\mathsf{init}^{\mathsf{C}}, (c, x : [\mathsf{pre}_j^{\mathsf{C}}, \mathsf{post}_j^{\mathsf{C}}])_{j\in J})$$

(with the same x and J) if the following conditions are valid[1].

$$\mathsf{init}^{\mathsf{C}} \Rightarrow \exists a \left(\mathsf{init}^{\mathsf{A}} \wedge r\right) \tag{2}$$
$$\mathsf{pre}_j^{\mathsf{A}} \wedge r \Rightarrow \mathsf{pre}_j^{\mathsf{C}} \ \text{, for } j \in J \tag{3}$$
$$\mathsf{pre}_j^{\mathsf{A}}\left[{}^{a_0,x_0}/_{a,x}\right] \wedge r\left[{}^{a_0,c_0,x_0}/_{a,c,x}\right] \wedge \mathsf{post}_j^{\mathsf{C}} \Rightarrow \exists a \left(\mathsf{post}_j^{\mathsf{A}} \wedge r\right) \ \text{, for } j \in J \tag{4}$$
$$\forall c, x\ (\exists a\ (r)) \ \text{, if } \exists j, c, x \left(\neg \mathsf{pre}_j^{\mathsf{C}}\right) \tag{5}$$

These conditions are motivated by the proof obligations for forward simulation, which we recall here briefly:

$$CI \subseteq AI; R \tag{6}$$
$$R; C_j \subseteq A_j; R \ \text{, for all } j \in J \tag{7}$$
$$R; CF \subseteq AF \tag{8}$$

In particular, (6) is captured in (2), (7) lead to conditions (3)–(5), while (8) holds by definition of the semantics of finalisations. That these conditions are sufficient is stated as

Theorem 4 *If r is a forward simulation predicate between A and C then there exists a refinement relation to show data refinement between $[\![\mathsf{A}]\!]$ and $[\![\mathsf{C}]\!]$ via forward simulation.*

[1] Recall that a predicate is *valid* iff it evaluates to true regardless of the values of its free variables.

Proof: Let r be a *forward simulation predicate* between syntactic data type A and C, that is, it satisfies (2) and, for all $j \in J$, (3) and (4). Let

$$(AI, (A_j)_{j\in J}, AF) = [\![\mathsf{A}]\!]$$
$$(CI, (C_j)_{j\in J}, CF) = [\![\mathsf{C}]\!]$$
$$R = \left\{ (\sigma, \tau) \in \Sigma^{\mathsf{A}} \times \Sigma^{\mathsf{C}} \;\middle|\; \Pi^{\mathsf{A}}(\sigma) = \Pi^{\mathsf{C}}(\tau) \wedge \sigma \cup \tau \in [\![r]\!]^{\mathcal{G}} \right\}$$

We need to show the forward simulation conditions. For (6) we let $\sigma \in \Sigma$ and $\tau \in \Sigma^{\mathsf{C}}$.

$$\begin{aligned}
& (\sigma, \tau) \in CI \\
\Leftrightarrow \quad & \langle \text{def. of } CI \rangle \\
& \exists \sigma' \in \Sigma^{\mathsf{C}} \left(\Pi^{\mathsf{C}}(\sigma') = \sigma \wedge (\sigma', \tau) \in [\![c : [\textsc{true}, \mathsf{init}^{\mathsf{C}}]]\!] \right) \\
\Leftrightarrow \quad & \langle \text{trivial pre and no reference to initial vars in post} \rangle \\
& \exists \sigma' \in \Sigma^{\mathsf{C}} \left(\Pi^{\mathsf{C}}(\sigma') = \sigma = \Pi^{\mathsf{C}}(\tau) \wedge \tau \in [\![\mathsf{init}^{\mathsf{C}}]\!]^{\mathcal{G}} \right) \\
\Leftrightarrow \quad & \langle \sigma' \text{ could be } \tau \rangle \\
& \sigma = \Pi^{\mathsf{C}}(\tau) \wedge \tau \in [\![\mathsf{init}^{\mathsf{C}}]\!]^{\mathcal{G}} \\
\Rightarrow \quad & \langle \text{weakening with (2)} \rangle \\
& \sigma = \Pi^{\mathsf{C}}(\tau) \wedge \tau \in [\![\exists a \left(\mathsf{init}^{\mathsf{A}} \wedge r \right)]\!]^{\mathcal{G}} \\
\Rightarrow \quad & \langle \text{def. of } R \text{ and a bit of work} \rangle \\
& \sigma = \Pi^{\mathsf{C}}(\tau) \wedge \exists \tau' \in \Sigma^{\mathsf{A}} \left(\tau' \in [\![\mathsf{init}^{\mathsf{A}}]\!]^{\mathcal{G}} \wedge (\tau', \tau) \in R \right) \\
\Leftrightarrow \quad & \langle \text{way back via def. of } AI \rangle \\
& (\sigma, \tau) \in AI ; R
\end{aligned}$$

For (7) we first reformulate (4) with the help of (3), or, more precisely with the help of an instance of (3) obtained by substituting a_0, c_0, x_0 for a, c, x:

$$\mathsf{pre}_j^{\mathsf{A}}[^{a_0,x_0}/_{a,x}] \wedge r[^{a_0,c_0,x_0}/_{a,c,x}] \Rightarrow \mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}]$$

which allows us to add the conjunct $\mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}]$ to the LHS of (4):

$$\mathsf{pre}_j^{\mathsf{A}}[^{a_0,x_0}/_{a,x}] \wedge r[^{a_0,c_0,x_0}/_{a,c,x}] \wedge \mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}] \wedge \mathsf{post}_j^{\mathsf{C}} \Rightarrow \exists a \left(\mathsf{post}_j^{\mathsf{A}} \wedge r \right)$$

We reorganise this (logic: $(p \wedge q \Rightarrow s) \Leftrightarrow (p \Rightarrow (q \Rightarrow s))$) so that the pieces resembling $R ; C$ end up to the left of the leading implication and those resembling $A ; R$ to the right:

$$r[^{a_0,c_0,x_0}/_{a,c,x}] \wedge \mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}] \wedge \mathsf{post}_j^{\mathsf{C}} \Rightarrow (\mathsf{pre}_j^{\mathsf{A}}[^{a_0,x_0}/_{a,x}] \Rightarrow \exists a \left(\mathsf{post}_j^{\mathsf{A}} \wedge r \right))$$

Since a is not free in $\mathsf{pre}_j^{\mathsf{A}}[^{a_0,x_0}/_{a,x}]$ we can push the existential quantifier left (logic: $p \wedge \exists y \, (q) \Leftrightarrow \exists y \, (p \wedge q)$, if y not free in p). Similarly, since c_0 is implicitly quantified in the whole formula and it doesn't occur on the RHS, we can push its quantification into the LHS (logic: $\forall y \, (p \Rightarrow q) \Leftrightarrow \exists y \, (p) \Rightarrow q$, if y not free in q).

$$\exists c_0 \left(r[^{a_0,c_0,x_0}/_{a,c,x}] \wedge \mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}] \wedge \mathsf{post}_j^{\mathsf{C}} \right) \Rightarrow \exists a \left(\mathsf{pre}_j^{\mathsf{A}}[^{a_0,x_0}/_{a,x}] \Rightarrow \mathsf{post}_j^{\mathsf{A}} \wedge r \right) \qquad (9)$$

The remainder of this proof proceeds as follows. Beginning with $(\sigma^{\mathsf{A}}, \tau^{\mathsf{C}}) \in R ; C_j$ we arrive at a fork in the road: if $\mathsf{pre}_j^{\mathsf{C}}[^{c_0,x_0}/_{c,x}]$ holds at the junction between R and C_j then

we can use (9). Otherwise we have that $\mathsf{pre}_j^{\mathsf{A}}[{}^{a_0,x_0}/_{a,x}]$ must be false, too, by (3). We then depend on (5) to provide the missing link from τ^{C} through r to some state τ^{A}. Either way, we show that $(\sigma^{\mathsf{A}}, \tau^{\mathsf{C}}) \in A_j; R$.

This is merely the road map for the complete proof of this point.

Finally, for (8) we argue

$$
\begin{array}{ll}
 & (\sigma^{\mathsf{A}}, \tau) \in R; CF \\
\Leftrightarrow & \quad \langle \text{defs of ";", } R \text{ and } CF \rangle \\
 & \exists \sigma^{\mathsf{C}} \in \Sigma^{\mathsf{C}} \left(\Pi^{\mathsf{A}}(\sigma^{\mathsf{A}}) = \Pi^{\mathsf{C}}(\sigma^{\mathsf{C}}) \wedge \sigma^{\mathsf{A}} \cup \sigma^{\mathsf{C}} \in [\![r]\!]^{\mathcal{G}} \wedge \tau = \Pi^{\mathsf{C}}(\sigma^{\mathsf{C}}) \right) \\
\Rightarrow & \quad \langle \text{logic} \rangle \\
 & \Pi^{\mathsf{A}}(\sigma^{\mathsf{A}}) = \tau \\
\Leftrightarrow & \quad \langle \text{def. of } AF \rangle \\
 & (\sigma^{\mathsf{A}}, \tau) \in AF
\end{array}
$$

which concludes the proof of Theorem 4. ■