Solutions to Exercises

2.2.1 See [104]. Hint: See the UML activity diagram rule UmlJoin on p. 280.

2.2.2 See [105].

2.2.3 For a solution DoublyLinkedList see `AsmMethod` (⇝ CD). Hint: Define a macro Link(x, y) which links two nodes x and y together. It is also convenient to use the n-fold application of the function *next* applied to node x, $next^n(x)$.

2.2.4 See Sect. 4.1.1 and define a turbo ASM which for any argument simply outputs the value of a static non-recursive function on that argument.

2.2.5 Distinguish the cases whether the alternation of the conditions is true or not.

2.3.1 Use an induction on walks through the diagram of Fig. 2.14.

2.3.2 Use Lemma 2.3.1. Note that every application of Depart, Continue, Stop counts as one step and that in case of more than one lift ($n > 1$), the notion of "point of attraction" is dynamically determined by Attracted(d, L) refined below.

2.3.3 Use Lemma 2.3.2.

2.3.7 Hint: an object-oriented language will allow us to naturally reflect the parameterization of the Lift ASM by the elements of an abstract domain Lift.

2.3.8 See Sect. 6.4 for a definition of durative actions.

2.4.1 Let l be a location. If l has the same content in $\mathfrak{A}$ and $\mathfrak{B}$, then there is no update for l in the difference $\mathfrak{B} - \mathfrak{A}$ and $(\mathfrak{A} + (\mathfrak{B} - \mathfrak{A}))(l) = \mathfrak{A}(l) = \mathfrak{B}(l)$. Otherwise, if $\mathfrak{A}(l) \neq \mathfrak{B}(l)$, then the update $(l, \mathfrak{B}(l))$ is in $\mathfrak{B} - \mathfrak{A}$ and hence $(\mathfrak{A} + (\mathfrak{B} - \mathfrak{A}))(l) = \mathfrak{B}(l)$.

2.4.2 Let (l, v) be an update in $\mathfrak{B} - \mathfrak{A}$. Then $\mathfrak{A}(l) \neq \mathfrak{B}(l) = v$. According to Def. 2.4.8, we know that $\alpha(\mathfrak{A}(l)) = \mathfrak{A}'(\alpha(l))$ and $\alpha(v) = \alpha(\mathfrak{B}(l)) = \mathfrak{B}'(\alpha(l))$. Since α is injective, it follows that $\mathfrak{A}'(\alpha(l)) \neq \mathfrak{B}'(\alpha(l)) = \alpha(v)$, hence the update $(\alpha(l), \alpha(v))$ is in $\mathfrak{B}' - \mathfrak{A}'$.

Since α is surjective, every update in $\mathfrak{B}' - \mathfrak{A}'$ has the form $(\alpha(l), \alpha(v))$ for a location l of $\mathfrak{A}$ and an element v in $|\mathfrak{A}|$, where $\mathfrak{A}'(\alpha(l)) \neq \mathfrak{B}'(\alpha(l)) = \alpha(v)$. Since α is an isomorphism, $\mathfrak{A}(l) \neq \mathfrak{B}(l) = v$. Hence (l, v) is an update in $\mathfrak{B} - \mathfrak{A}$.

2.4.3 1. That $\oplus$ is associative follows directly from Def. 2.4.10.

Assume that (l, v) belongs to $(U \oplus V) \oplus W$. If (l, v) in W, then $(l, v) \in V \oplus W$ and hence $(l, v) \in U \oplus (V \oplus W)$. If l does not have an update in W but $(l, v) \in V$, then $(l, v) \in V \oplus W$ and hence also $(l, v) \in U \oplus (V \oplus W)$. Otherwise, l does not have an update neither in V nor W but $(l, v) \in U$. Then l does not have an update in $V \oplus W$ and hence $(l, v) \in U \oplus (V \oplus W)$.

Assume that (l, v) belongs to $U \oplus (V \oplus W)$. If $(l, v) \in V \oplus W$, then either $(l, v) \in W$ or l is not updated in W and $(l, v) \in V$. If $(l, v) \in W$, then (l, v) belongs also to $(U \oplus V) \oplus W$. If l is not updated in W and $(l, v) \in V$, then $(l, v) \in U \oplus V$ and hence $(l, v) \in (U \oplus V) \oplus W$. If l is not updated in $V \oplus W$, then $(l, v) \in U$ and there is no update for l neither in V nor in W. Hence, $(l, v) \in U \oplus V$ and thus $(l, v) \in (U \oplus V) \oplus W$.

2. Assume that U and V are consistent. Assume that $(l, v) \in U \oplus V$ and $(l, w) \in U \oplus V$. If l has an update in V, then $(l, v) \in V$ and $(l, w) \in V$, hence $v = w$, since V is consistent. If l is not updated in V, then $(l, v) \in U$ and $(l, w) \in U$, and since U is consistent, it follows that $v = w$. Hence, $U \oplus V$ is consistent.

3. Assume that U and V are consistent. By 2, we know that $U \oplus V$ is consistent. We have to show that $\mathfrak{A} + (U \oplus V) = (\mathfrak{A} + U) + V$.

Let l be a location. If l is not updated in $U \oplus V$, then $(\mathfrak{A} + (U \oplus V))(l) = \mathfrak{A}(l)$ and, since this implies that l is neither updated in U nor in V, also $((\mathfrak{A} + U) + V)(l) = \mathfrak{A}(l)$. Otherwise, there is an update for l in $U \oplus V$, say (l, v), and $(\mathfrak{A} + (U \oplus V))(l) = v$. If $(l, v) \in V$, then $((\mathfrak{A} + U) + V)(l) = v$. If l is not updated in V and $(l, v) \in U$, then $(\mathfrak{A} + U)(l) = v$ and hence $((\mathfrak{A} + U) + V)(l) = v$.

2.4.4 Equation 1 is not true in general.

Take $U = \{(l, 0)\}$, $V = \{(l, 1)\}$ and $W = \emptyset$. Then $U \oplus (V \cup W) = \{(l, 1)\}$, whereas $(U \oplus V) \cup (U \oplus W) = \{(l, 0), (l, 1)\}$.

If $(U \oplus V) \cup (U \oplus W)$ is consistent, then $U \oplus (V \cup W) = (U \oplus V) \cup (U \oplus W)$.

Equation 2 is true, i.e. $(U \cup V) \oplus W = (U \oplus W) \cup (V \oplus W)$.

Assume that $(l, v) \in (U \cup V) \oplus W$. If $(l, v) \in W$, then $(l, v) \in U \oplus W$. If l is not updated in W then $(l, v) \in U \cup V$. If $(l, v) \in U$, then $(l, v) \in U \oplus W$. If $(l, v) \in V$, then $(l, v) \in V \oplus W$.

Assume that $(l, v) \in (U \oplus W) \cup (V \oplus W)$. If $(l, v) \in W$, then (l, v) belongs also to $(U \cup V) \oplus W$. If l is not updated in W, then (l, v) is in U or in V and hence also in $(U \cup V) \oplus W$.

2.4.5 The proof of Lemma 2.4.5 goes by induction on the size of the term t.
Case 1, t is a variable x: $\alpha([\![x]\!]^{\mathfrak{A}}_{\zeta}) = \alpha(\zeta(x)) = (\alpha \circ \zeta)(x) = [\![x]\!]^{\mathfrak{B}}_{\alpha\circ\zeta}$
Case 2, t is a constant c: $\alpha([\![c]\!]^{\mathfrak{A}}_{\zeta}) = \alpha(c^{\mathfrak{A}}) = c^{\mathfrak{B}} = [\![c]\!]^{\mathfrak{B}}_{\alpha\circ\zeta}$
Case 3. t is the term $f(s_1, \ldots, s_n)$.

$$\begin{array}{rcll} \alpha([\![f(s_1,\ldots,s_n)]\!]^{\mathfrak{A}}_{\zeta}) & = & \alpha(f^{\mathfrak{A}}([\![s_1]\!]^{\mathfrak{A}}_{\zeta},\ldots,[\![s_n]\!]^{\mathfrak{A}}_{\zeta})) & \text{[Def. 2.4.13]} \\ & = & f^{\mathfrak{B}}(\alpha([\![s_1]\!]^{\mathfrak{A}}_{\zeta}),\ldots,\alpha([\![s_n]\!]^{\mathfrak{A}}_{\zeta})) & \text{[Def. 2.4.8]} \\ & = & f^{\mathfrak{B}}([\![s_1]\!]^{\mathfrak{B}}_{\alpha\circ\zeta},\ldots,[\![s_n]\!]^{\mathfrak{B}}_{\alpha\circ\zeta}) & \text{[Ind. hyp.]} \\ & = & [\![f(s_1,\ldots,s_n)]\!]^{\mathfrak{B}}_{\alpha\circ\zeta} & \text{[Def. 2.4.13]} \end{array}$$

2.4.6 1. The coincidence Lemma 2.4.4 for terms is proved by induction on the size of the term t using Def. 2.4.13 of the interpreation of terms.

2. The coincidence Lemma 2.4.7 for formulas is proved by induction on the size of the formula φ using the definition of the semantics of formulas in Table 2.1.

3. The coincidence Lemma 2.4.10 for transition rules is proved by induction on the definition of the "yields" relation in Table 2.2. In the case of **forall** and **choose** one has to use the following fact which is an immediate consequence of Lemma 2.4.7: $\mathit{range}(x, \varphi, \mathfrak{A}, \zeta) = \mathit{range}(x, \varphi, \mathfrak{A}, \eta)$, if $\zeta(y) = \eta(y)$ for every variable in $\mathrm{FV}(\varphi) \setminus \{x\}$ (see also Def. 2.4.16). In the case of a rule call one has tu use the fact that, by definition, in a rule declaration $r(x_1, \ldots, x_n) = R$ the body R contains no free variables except $x_1, \ldots, x_n$ (Def. 2.4.18).

2.4.7 The substitution Lemma 2.4.6 for terms is proved by induction on the size of the term t. The substitution Lemma 2.4.8 formulas is proved by induction on the size of the formula φ. The substitution Lemma 2.4.11 for transition rules is proved by induction on the definition of the "yields" relation in Table 2.2. We consider the case of the sequential composition **seq**, since in this case, the assumption is used that the substituted term is static.
Assume that t is a static term and $a = [\![t]\!]^{\mathfrak{A}}_{\zeta}$. Assume that

– yields$(P\frac{t}{x}, \mathfrak{A}, \zeta, U)$ and U is consistent
– yields$(Q\frac{t}{x}, \mathfrak{A} + U, \zeta, V)$
– yields$((P \textbf{ seq } Q)\frac{t}{x}, \mathfrak{A}, \zeta, U \oplus V)$

By the induction hypothesis applied to yields$(P\frac{t}{x}, \mathfrak{A}, \zeta, U)$ we obtain that yields$(P, \mathfrak{A}, \zeta[x \mapsto a], U)$.
Since t is a static term, it has the same value in the states $\mathfrak{A}$ and $\mathfrak{A} + U$, i.e. $[\![t]\!]^{\mathfrak{A}}_{\zeta} = [\![t]\!]^{\mathfrak{A}+U}_{\zeta} = a$.

Hence, by the induction hypothesis applied to yields$(Q\frac{t}{x}, \mathfrak{A} + U, \zeta, V)$ we obtain yields$(Q, \mathfrak{A} + U, \zeta[x \mapsto a], V)$.

This means that yields$(P \textbf{ seq } Q, \mathfrak{A}, \zeta[x \mapsto a], U \oplus V)$.

The other direction is shown in a similar way.

2.4.8 Let t be the term $f(0)$ and P be the transition rule

$$f(0) := 1 \textbf{ seq } f(1) := x.$$

Then $P\frac{t}{x}$ is the transition rule

$$f(0) := 1 \textbf{ seq } f(1) := f(0).$$

Let $\mathfrak{A}$ be a state in which $f(0)$ is equal to 0, i.e. $[\![f(0)]\!]^{\mathfrak{A}} = 0$.

Then $P\frac{t}{x}$ yields the update set

$$\{((f, 0), 1), ((f, 1), 1)\}$$

in $\mathfrak{A}$ under ζ, whereas P yields the update set

$$\{((f, 0), 1), ((f, 1), 0)\}$$

in $\mathfrak{A}$ under $\zeta[x \mapsto 0]$.

2.4.9 The isomorphism Lemma 2.4.9 formulas is proved by induction on the size of the formula φ. In the base case, Lemma 2.4.5 is used for terms. The isomorphism Lemma 2.4.12 for transition rules is proved by induction on the definition of the "yields" relation in Table 2.2.

2.4.10 The equivalences between the rules are proved using the definition of the "yield" relation in Table 2.2.

1. $(P \textbf{ par skip}) \equiv P$

   Assume that P **par skip** yields U. This implies that there exist V and W such that P yields V, **skip** yields W, and $U = V \cup W$. Since the **skip** rule always yields the empty set, it follows that $W = \emptyset$ and thus $U = V$. Hence, P yields U.

   If P yields U, then also P **par skip** yields U.

2. $(P \textbf{ par } Q) \equiv (Q \textbf{ par } P)$

   Assume that P **par** Q yields U. This implies that there exist V and W such that P yields V and Q yields W, and $U = V \cup W$. But then Q **par** P yields $W \cup V$, hence Q **par** P yields U.

   Hence the commutativity of **par** follows from the commutativity of the set union ($\cup$).

3. $((P \textbf{ par } Q) \textbf{ par } R) \equiv (P \textbf{ par } (Q \textbf{ par } R))$

   The associativity of **par** follows from the the associativity of the set union.

4. $(P \textbf{ par } P) \equiv P$ [if P is deterministic (without **choose**)]

   Assume that P is a deterministic rule of an ASM without **choose**. Assume that P **par** P yields U. This implies that there exist V and W such that P yields V, P yields W, and $U = V \cup W$. Since P is deterministic, the update sets V and W are equal, hence $U = V = W$ and P yields U.

   If P yields U, then P **par** P yields $U \cup U = U$. (This is true even for non-deterministic rules P.)

5. $(\textbf{if } \varphi \textbf{ then } P \textbf{ else } Q) \textbf{ par } R \equiv \textbf{if } \varphi \textbf{ then } (P \textbf{ par } R) \textbf{ else } (Q \textbf{ par } R)$

   Assume that (**if** φ **then** P **else** Q) **par** R yields U. This implies that there exist V and W such that **if** φ **then** P **else** Q yields V, R yields W, and $U = V \cup W$. If φ is true, then P yields V and hence P **par** R yields $V \cup W$. If φ is false, then Q yields V and hence Q **par** R yields $V \cup W$. In both cases **if** φ **then** (P **par** R) **else** (Q **par** R) yields U.

6. $(P \textbf{ seq skip}) \equiv P$

   Assume that P **seq skip** yields U in $\mathfrak{A}$. There are two cases.

   Case 1: P yields U in $\mathfrak{A}$ and U is inconsistent.

   Case 2: P yields V in $\mathfrak{A}$, V is consistent, **skip** yields W in $\mathfrak{A} + V$, and $U = V \oplus W$. But then W is the empty set and $U = V$. Hence, in both cases P yields U in $\mathfrak{A}$.

   If P yields U in $\mathfrak{A}$, then P **seq skip** yields also U in $\mathfrak{A}$.

7. $(\textbf{skip seq } P) \equiv P$

   If **skip seq** P yields U in $\mathfrak{A}$, then, since **skip** yields $\emptyset$ and $\mathfrak{A} + \emptyset = \mathfrak{A}$, it follows that P yields U in $\mathfrak{A}$.

8. $((P \textbf{ seq } Q) \textbf{ seq } R) \equiv (P \textbf{ seq } (Q \textbf{ seq } R))$

   The associativity of **seq** follows using the properties of $\oplus$ in Lemma 2.4.3.

   Assume that P **seq** (Q **seq** R) yields U in $\mathfrak{A}$. Then there are several cases.

   Case 1. P yields U in $\mathfrak{A}$ and U is inconsistent. In this case (P **seq** Q) **seq** R also yields U in $\mathfrak{A}$

   Case 2. P yields V in $\mathfrak{A}$, V is consistent, (Q **seq** R) yields W in $\mathfrak{A} + V$ and $U = V \oplus W$.

   Case 2.1. Q yields W in $\mathfrak{A} + V$ and W is inconsistent. Then $V \oplus W$ is also inconsistent, P **seq** Q yields $V \oplus W$ in $\mathfrak{A}$ and hence (P **seq** Q) **seq** R yields U in $\mathfrak{A}$.

   Case 2.2. Q yields X in $\mathfrak{A} + V$, X is consistent, R yields Y in $(\mathfrak{A} + V) + X$ and $W = X \oplus Y$. In this case P **seq** Q yields $V \oplus X$ in $\mathfrak{A}$. By Lemma 2.4.3, $(\mathfrak{A} + V) + X$ is equal to $\mathfrak{A} + (V \oplus X)$. Therefore R yields Y in $\mathfrak{A} + (V \oplus X)$

and $(P \textbf{ seq } Q) \textbf{ seq } R$ yields $(V \oplus X) \oplus Y$ in $\mathfrak{A}$. By the associativity of $\oplus$, $(V \oplus X) \oplus Y = V \oplus (X \oplus Y) = V \oplus W = U$.

The converse direction is proved in a similar way.

9. $(\textbf{if } \varphi \textbf{ then } P \textbf{ else } Q) \textbf{ seq } R \equiv \textbf{if } \varphi \textbf{ then } (P \textbf{ seq } R) \textbf{ else } (Q \textbf{ seq } R)$

   Assume that $(\textbf{if } \varphi \textbf{ then } P \textbf{ else } Q) \textbf{ seq } R$ yields U in $\mathfrak{A}$. Then there are two cases:

   Case 1. U is inconsistent and $\textbf{if } \varphi \textbf{ then } P \textbf{ else } Q$ yields U in $\mathfrak{A}$. If φ is true in $\mathfrak{A}$, then P yields U in $\mathfrak{A}$ and also $P \textbf{ seq } R$ yields U in $\mathfrak{A}$. If φ is false in $\mathfrak{A}$, then Q yields U in $\mathfrak{A}$ and also $Q \textbf{ seq } R$ yields U in $\mathfrak{A}$. In both cases, $\textbf{if } \varphi \textbf{ then } (P \textbf{ seq } R) \textbf{ else } (Q \textbf{ seq } R)$ yields U in $\mathfrak{A}$.

   Case 2. $\textbf{if } \varphi \textbf{ then } P \textbf{ else } Q$ yields V in $\mathfrak{A}$, V is consistent, R yields W in $\mathfrak{A} + V$, and $U = V \oplus W$. If φ is true in $\mathfrak{A}$, then P yields V in $\mathfrak{A}$ and hence $P \textbf{ seq } R$ yields $V \oplus W$ in $\mathfrak{A}$. If φ is false in $\mathfrak{A}$, then Q yields V in $\mathfrak{A}$ and hence $Q \textbf{ seq } R$ yields $V \oplus W$ in $\mathfrak{A}$. In both cases $\textbf{if } \varphi \textbf{ then } (P \textbf{ seq } R) \textbf{ else } (Q \textbf{ seq } R)$ yields $V \oplus W = U$ in $\mathfrak{A}$.

   The converse direction is proved in a similar way.

Counter examples are:

1. $((P \textbf{ par } Q) \textbf{ seq } R) \not\equiv ((P \textbf{ seq } R) \textbf{ par } (Q \textbf{ seq } R))$

   Let $\mathfrak{A}$ be a state with $p = 0$ and $q = 0$. The rule

   $$(p := 1 \textbf{ par } q := 1) \textbf{ seq } r := p + q$$

   updates r to 2 in $\mathfrak{A}$. The rule

   $$(p := 1 \textbf{ seq } r := p + q) \textbf{ par } (q := 1 \textbf{ seq } r := p + q)$$

   updates r to 1 in $\mathfrak{A}$. Hence, the two transition rules are not equivalent.

2. $(P \textbf{ seq } (Q \textbf{ par } R)) \not\equiv ((P \textbf{ seq } Q) \textbf{ par } (P \textbf{ seq } R))$

   Let P be the rule $c := 0 \textbf{ par } d := 0$. Then the rule

   $$P \textbf{ seq } (c := 1 \textbf{ par } d := 1)$$

   yields the update set $\{(c, 1), (d, 1)\}$. The rule

   $$(P \textbf{ seq } c := 1) \textbf{ par } (P \textbf{ seq } d := 1)$$

   yields the inconsistent update set $\{(d, 0), (c, 1), (c, 0), (d, 1)\}$.

3. $(\textbf{let } x = t \textbf{ in } P) \not\equiv P\frac{t}{x}$

   Let $\mathfrak{A}$ be a state with $f(0) = 0$. Then

   $$\textbf{let } x = f(0) \textbf{ in } (f(0) := 1 \textbf{ seq } f(1) := x)$$

   updates f at the argument 1 to the value 0, whereas

   $$f(0) := 1 \textbf{ seq } f(1) := f(0)$$

   udates f at the argument 1 to the value 1.

2.4.11 Using the definition of the "yields" relation in Table 2.2 the following property is proved by induction on the size of R: If $P \equiv Q$ and $\mathrm{yields}(R[P], \mathfrak{A}, \zeta, U)$, then $\mathrm{yields}(R[Q], \mathfrak{A}, \zeta, U)$. In the case that the occurrence of P in $R[P]$ is $R[P]$ itself, $R[Q]$ is equal to Q and, since $P \equiv Q$, nothing has to be shown. Otherwise one can assume that the occurrences of P are in the components of R.

2.4.12 1. Consider the following two transition rules:

$$\underbrace{\textbf{if } c \neq 0 \textbf{ then } c := 0}_{P} \qquad \underbrace{c := 0}_{Q}$$

Then P and Q are extensionally equal but not equivalent in sense of Exercise 2.4.10.

2. Consider the following context:

$$\underbrace{c := 1 \textbf{ par } *}_{R[*]}$$

Then $R[Q]$ yields in every state the inconsistent update set $\{(c, 0), (c, 1)\}$. In a state $\mathfrak{A}$ with $c = 0$, the rule $R[P]$, however, yields the update set $\{(0, 1)\}$. Hence P and Q are extensionally equal but $R[P]$ and $R[Q]$ are not.

2.4.13 Let α be an isomorphism from $\mathfrak{A}$ to $\mathfrak{B}$. Then α maps the reserve of $\mathfrak{A}$ onto the reserve of $\mathfrak{B}$, i.e. $\alpha(\mathit{Res}(\mathfrak{A})) = \mathit{Res}(\mathfrak{B})$. We have to show that the side conditions for the modified rules for "yields" in Table 2.3 are also true for $\mathfrak{B}$.

Case **import**: If $a \in \mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta)$, then $\alpha(a) \in \mathit{Res}(\mathfrak{B}) \setminus \mathit{ran}(\alpha \circ \zeta)$.

Case **par**: If $\mathit{Res}\mathfrak{A} \cap \mathit{El}(U) \cap \mathit{El}(V) \subseteq \mathit{ran}(\zeta)$, then
$\mathit{Res}(\mathfrak{B}) \cap \mathit{El}(\alpha(U)) \cap \mathit{El}(\alpha(V)) \subseteq \mathit{ran}(\alpha \circ \zeta)$.

2.4.14 In the case of **skip** or a basic update, there is nothing to show. We can simply take the identity function which permutes $\mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta)$.

In the case of **if** φ **then** P **else** Q we apply the induction hypothesis either to P or to Q depending on whether the guard φ is true or false in $\mathfrak{A}$ under ζ.

In the case of **let** assume that $a = [\![t]\!]^{\mathfrak{A}}_{\zeta}$ and

$$\frac{\mathrm{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U)}{\mathrm{yields}(\textbf{let } x = t \textbf{ in } P, \mathfrak{A}, \zeta, U)} \qquad \frac{\mathrm{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U')}{\mathrm{yields}(\textbf{let } x = t \textbf{ in } P, \mathfrak{A}, \zeta, U')}$$

Since x is not in the domain of ζ, we have $\mathit{ran}(\zeta[x \mapsto a]) = \mathit{ran}(\zeta) \cup \{a\}$ and $\mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta[x \mapsto a])$ is contained in $\mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta)$. Hence, $\mathfrak{A}$ satisfies the reserve condition with respect to the extended environment $\zeta[x \mapsto a]$. By the induction hypothesis, there exists a permuation α of $\mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta[x \mapsto a])$ such that $\alpha(U) = U'$. The function α is also a permuation of $\mathit{Res}(\mathfrak{A}) \setminus \mathit{ran}(\zeta)$.

In the case of **forall** assume that $I = range(x, \varphi, \mathfrak{A}, \zeta)$ and

$$\frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U_a) \quad \text{for each } a \in I}{\text{yields}(\textbf{forall } x \textbf{ with } \varphi \textbf{ do } P, \mathfrak{A}, \zeta, \bigcup_{a \in I} U_a)}$$

$$\frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U'_a) \quad \text{for each } a \in I}{\text{yields}(\textbf{forall } x \textbf{ with } \varphi \textbf{ do } P, \mathfrak{A}, \zeta, \bigcup_{a \in I} U'_a)}$$

and for all $a, b \in I$, if $a \neq b$, then

$$Res(\mathfrak{A}) \cap El(U_a) \cap El(U_b) \subseteq ran(\zeta),$$

$$Res(\mathfrak{A}) \cap El(U'_a) \cap El(U'_b) \subseteq ran(\zeta).$$

By the inductions hypothesis, there exists for each $a \in I$ a permutation α_a of $Res(\mathfrak{A}) \setminus ran(\zeta[x \mapsto a])$ such that $\alpha_a(U_a) = U'_a$. Let γ be the union of the permuations α_a restricted to the set $\bigcup_{a \in I} El(U_a)$. As in the case of **par** the function γ is well-defined and one-one. Since the set $\bigcup_{a \in I} El(U_a)$ is finite, there exists a permutation of $Res(\mathfrak{A}) \setminus ran(\zeta)$ that agrees with γ on $\bigcup_{a \in I} El(U_a)$ and therefore maps $\bigcup_{a \in I} U_a$ to $\bigcup_{a \in I} U'_a$.
In the case of a rule call, we can simply apply the induction hypothesis.

2.4.15 Let $\mathfrak{A}$ be a state that contains two copies of the set of natural numbers, one copy $0, 1, 2, \ldots$ for a subuniverse *Nat* and one copy $\hat{0}, \hat{1}, \hat{2}, \ldots$ for the *Reserve*. Consider the following transition rule:

forall $x \in Nat$ **do**
  import y **do** $f(x) := y$

Two possible update sets of the transition rule in state $\mathfrak{A}$ are:

$$\begin{aligned} U &= \{((f, n), \hat{n}) \mid n \in \mathbb{N}\} \cup \{((Reserve, \hat{n}), false) \mid n \in \mathbb{N}\} \\ U' &= \{((f, n), \widehat{n+1}) \mid n \in \mathbb{N}\} \cup \{((Reserve, \hat{n}), false) \mid n > 0\} \end{aligned}$$

In the first case, the reserve is fully exhausted after firing the updates of U. In the second case, the reserve still contains the element $\hat{0}$ after firing U'. A map α that maps U to U' has to map $\hat{n}$ to $\widehat{n+1}$. The map α, however, cannot be a permutation of the reserve, since $\hat{0}$ is not met by α.

3.1.1 See `GroundModelsSimpleExls` (⇝ CD).

3.1.6 Use a conservative refinement. A robustness-rule usually has the form **if** *Cond′* **then** ..., where *Cond′* selects the relevant ways in which the *Cond* guarding a "normal" rule r is expected to be violated (preventing r from being fired), without other rules of the machine being enabled.

3.1.8 See `GroundModelsSimpleExls` (⇝ CD).

3.1.10 See `GroundModelsSimpleExls` (⇝ CD). Use a conservative extension.

3.1.11 See `GroundModelsSimpleExls` (⇝ CD).

3.1.14 Use iterated **choose** as in [410, Fig. 5.3] and shown on slide 21 in `ComponentModel` (⇝ CD).

3.2.3 See slide 7 of `Backtracking` (⇝ CD).

3.2.4 See [105, Sect. 4].

3.2.6 Consider in corresponding segments of computation from a crash (included) to the next crash (excluded) a) the $(1,1)$-diagrams originating from applications of homonymous rules, and b) the $(0,1)$-diagrams originating from applications of GET in the implementation LATECHOICE where the refined $return_{ref}$ takes the first alternative (read: decides to delete a message from the queue without returning it). To reflect that the same messages get lost you will have to assume that in the crash the choice function $select_{early}$ chooses exactly those elements for deletion from the queue for which $select_{late}$ chooses the first alternative of the refined $return_{ref}$. For a prophecy variable solution see [316, p. 8.14].

3.2.8 Simulate each step of the machine SHORTESTPATH_1 in state t, applied to $frontier_t(\text{SHORTESTPATH}_1)$, by selecting successively all the elements of $frontier_t(\text{SHORTESTPATH}_1)$.

3.2.11 See `Stack` (⇝ CD).

3.2.12 Refine WRITELOG by logging also the before-image of the written location (fetched from stable or cache memory), to be used in UNDO. See [260, Sect. 4.1].

3.2.13 Refine FAIL to initialize an iterator *thisRec* for scanning all records in *stableLog* for recovery. Refine ABORT to first initialize the iterator *thisRec* and then scanning all *log* records of the considered transaction. Refine WRITELOG by linking the new log to the current transaction's last log record. See [260, Sect. 4.2].

3.2.14 See [260, Sect. 4.3].

3.3.2 Use an induction on the number of fetched jump or branch instructions.

3.3.3 See the KIV verification in [217, 407].

3.3.5 See [119, Sect. 5]. First eliminate the insertion of two empty instructions in P^{par}, using a special decoding which permits us to detect jump or branch instructions at the end of the IF-stage, and anticipating the computation of the new PC-value to the ID-stage. Then consider the possible cases for data-dependent jump instructions and refine the rule macros accordingly.

3.3.6 Anticipate the load risk detection (and the stalling) to the *ID*-stage instead of waiting until the *EX*-stage. See [279].

4.1.6 See the AsmGofer program in [390].

5.1.2 See [120].

5.1.3 Introduce an additional control state *MovingToUnloadPos* and use a monitored function *UnloadPosReached*. See Exercise 5.1.1.

5.1.4 See [120].

5.1.5 See [120, 424, 207].

5.1.6 See [120].

5.1.8 See [120].

5.1.9 See [120].

5.2.2 See slide 20 in `GateController` (⇝ CD).

5.2.3 See slide 28 in `GateController` (⇝ CD).

5.2.4 See [253].

6.1.6 See [372, Fig. 32.1].

6.1.8 See [372, Fig. 32.2].

6.1.9 Use a pure data refinement, recording the initiator's identity when informing neighbors and letting the initiator wait for the echo to is own initiative.

6.1.10 See [254].

6.2.1 See [125, p. 612].

6.3.1 Consider in particular the restriction to the case where except for the initially empty sender and receiver interval $[1, 0]$, the windows do not exceed length 1.

6.3.3 For the first part use $msgId = fileNum(m) \bmod 2 \times winsize$ and adapt *Match*. For the second part see [283].

6.3.4 See [258] and the lecture slides `Bakery` (⇝ CD).

6.3.5 Investigate applications of *ReSend* which could be in parallel with applications of $SlideWindow(sender)$.

6.3.6 See [283]

6.3.7 Use the broadcast reliability assumption.

6.3.8 Use the finiteness of *PROCESSOR*, the monotonicity of every $Clock(p)$, the positive heartbeat interval $d_h > d_u$, the lower recovery bound $d_r > d_h + d_u$, the new group timestamp increment $d_n > d_c + d_u$, and the upper delivery bound d_c.

6.3.11 Instead of sending a single copy of m to the entire group, make *InTransit* binary to send one copy to each processor p, to be deleted by the $MsgCarrier(p)$ when put into $InBox(p)$.

6.3.12 Modify CUSTODIAN(p) to delete messages with timestamp $< Clock(p)$ (if p is crashed) or with timestamp $< StartUpTime(p)$ (if p is alive).

6.4.1 See [3, Fig. 3, p. 6].

6.4.2 See [114].

6.4.3 Consider a reader and a writer where for two reads overlapping with a write, the first read gets the later write value and the second read the earlier one.

6.4.4 Adapt the above proofs, using C0–C3, or see [3].

6.5.2 See [266, p. 26].

7.1.3 See STREAMPROCESSINGFSM.

7.1.5 See [177].

7.2.1 Let Σ be the signature with two nullary dynamic functions c and d and a static unary function f. We consider the algorithm A given by the following ASM update rule:

$$c := d$$

Let $T = \{c, d, f(c)\}$. By Lemma 7.2.3 we know that A satisfies the sequential-time, the abstract-state and the uniformly-bounded-exploration postulate. Let $\mathfrak{A}$ and $\mathfrak{B}$ be two states with base set $\{0, 1\}$ and the following properties:

$$\mathfrak{A} \models c = 0 \wedge d = 1 \wedge f(0) = 0 \wedge f(1) = 0$$
$$\mathfrak{B} \models c = 0 \wedge d = 1 \wedge f(0) = 0 \wedge f(1) = 1$$

Since the states $\mathfrak{A}$ and $\mathfrak{B}$ differ only in the interpretation of $f(1)$, they coincide over T. In the successor states, where $c = d = 1$, we have $\tau_A(\mathfrak{A}) \models f(c) = 0$ and $\tau_B(\mathfrak{A}) \models f(c) = 1$. Hence, $\tau_A(\mathfrak{A})$ and $\tau_A(\mathfrak{B})$ do not coincide over T. Note, that the set T is even closed under subterms.

8.1.1 To verify the validity of the properties D1–D9 in Table 8.2 and U1–U9 in Table 8.3 one has to study the semantics of the predicates "upd" and "def" in Table 8.1 as well as the inductive definition of the "yields" relation in Table 2.2.

8.1.2 The validity of the substitution principle in Lemma 8.1.1 is shown by induction on the length of the modal formula φ using the substitution lemmas 2.4.6, 2.4.8 and 2.4.11.

The substitution principle is in general not true for non-static terms. Let f be a dynamic function and $\mathfrak{A}$ be a state in which $f(0)$ is equal to 0. Let φ be the formula $[f(0) := 1](x = 0)$ and t be the non-static term $f(0)$. Then we have:

$$[\![[f(0) := 1](f(0) = 0)]\!]^{\mathfrak{A}}_{\zeta} = \mathit{false}$$
$$[\![[f(0) := 1](x = 0)]\!]^{\mathfrak{A}}_{\zeta[x \mapsto 0]} = \mathit{true}$$

8.1.3 We show just one direction of Lemma 8.1.2. The converse direction is proved in a similar way.

Assume that $[\![P \simeq Q]\!]^{\mathfrak{A}}_{\zeta} = \mathit{true}$.

That $[\![\mathsf{Con}(P)]\!]^{\mathfrak{A}}_{\zeta} = [\![\mathsf{Con}(Q)]\!]^{\mathfrak{A}}_{\zeta}$ follows from the definition of the formula $P \simeq Q$.

Assume that P yields a consistent update set U in $\mathfrak{A}$ under ζ and Q yields a consistent update set V. We have to show that $\mathfrak{A} + U = \mathfrak{A} + V$. Since $\mathsf{Con}(P)$ and $\mathsf{Con}(Q)$ are true in $\mathfrak{A}$ under ζ, the following two formulas are also true:

$$\bigwedge\!\!\bigwedge_{f \text{ dyn.}} \forall x, y\, (\mathsf{upd}(P, f, x, y) \rightarrow (\mathsf{upd}(Q, f, x, y) \vee f(x) = y))$$

$$\bigwedge\!\!\bigwedge_{f \text{ dyn.}} \forall x, y\, (\mathsf{upd}(Q, f, x, y) \rightarrow (\mathsf{upd}(P, f, x, y) \vee f(x) = y))$$

Let a be a possible argument of f and b be the value of $f(a)$ in state $\mathfrak{A} + U$. We have to show that $f(a)$ is equal to b also in $\mathfrak{A} + V$.

Case 1. The update $((f, a), b)$ is in U: Then $\mathsf{upd}(P, f, x, y)$ is true, if we assign a to x and b to y. Hence, $\mathsf{upd}(Q, f, x, y)$ or $f(x) = y$ is true, which means that the update $((f, a), b)$ is in V or $f(a)$ is equal to b in $\mathfrak{A}$. If $((f, a), b)$ is in V, then $f(a)$ is equal to b in $\mathfrak{A} + V$. Otherwise, $f(a)$ is equal to b in $\mathfrak{A}$. Suppose that there is an update $((f, a), c)$ in V with $c \neq b$. This would imply that $((f, a), c)$ is also U and U would be inconsistent. Since, $f(a)$ is equal to v in $\mathfrak{A}$, it is also equal to v in $\mathfrak{A} + V$.

Case 2. There is no update for (f, a) in U and $f(a)$ is equal to b in $\mathfrak{A}$: If there is no update for (f, a) in V, then $f(a)$ is equal to b in $\mathfrak{A} + V$. Otherwise, there is an update $((f, a), c)$ in V. It follows that $f(a)$ is equal to c in $\mathfrak{A}$. Hence, $b = c$ and we are done.

8.1.4 We have to show that the axioms of the logic are valid. The validity of the restricted quantifier axioms 1 and 2 follows from Lemma 8.1.1.

The modal axiom 3 is valid, since the modal operators have a Kripke semantics (see Table 8.1).

The necessitation rule 4 preserves validity.

Axiom 5 is valid, since if $\mathsf{Con}(R)$ is false, then either R is not defined or yields an inconsistent update set.

Axiom 6 is valid, since we consider only deterministic ASMs that do not contain **choose**. Assume that $[R]\varphi$ is false in $\mathfrak{A}$ under ζ. We want to show that $[R]\neg\varphi$ is true in $\mathfrak{A}$ under ζ. Assume that R yields a consistent update set U in $\mathfrak{A}$ under ζ. Suppose that φ is true in $\mathfrak{A} + U$ under ζ. Then $[R]\varphi$ would be true in $\mathfrak{A}$ under ζ which is not the case by assumption. Hence, φ must be false in $\mathfrak{A} + U$ under ζ.

The Barcan Axiom 7 is valid, since the base set of a state does not change when a rule is fired. Assume that $\forall x[R]\varphi$ is true in $\mathfrak{A}$ under ζ and R yields the consistent update set U in $\mathfrak{A}$ under ζ. We want to show that $\forall x\, \varphi$ is true in $\mathfrak{A} + U$ under ζ. Let $a \in |\mathfrak{A}|$. By the assumption it follows that $[R]\varphi$ is true in $\mathfrak{A}$ under $\zeta[x \mapsto a]$. Since the variable x is not free in rule R, by the Coincidence Lemma 2.4.10, R yields U in $\mathfrak{A}$ under $\zeta[x \mapsto a]$. Hence, φ is true in $\mathfrak{A} + U$ under $\zeta[x \mapsto a]$.

Axiom 8 is valid, since the truth value of a static, pure first-order formula φ depends only on the interpretation of the static functions. Hence, if φ is true in $\mathfrak{A}$ under ζ, then φ is true in every possible sequel $\mathfrak{A} + U$.

For the same reason axiom 9 is valid. If a static, pure first-order formula φ is true in a state $\mathfrak{A} + U$ under ζ, then it is also true in $\mathfrak{A}$ under ζ.

The validity of the axioms D1–D9 in Table 8.2 and U1–U9 in Table 8.3 has to be shown in Exercise 8.1.1.

Axiom 12 is valid, since according to the definition of the semantics of the predicate upd in Table 8.1, if $\mathsf{upd}(R, f, x, y)$ is true, then R yields an update set and hence R is defined.

The validity of Axiom 13 can be seen as follows. Assume that $\mathsf{upd}(R, f, x, y)$ is true in $\mathfrak{A}$ under ζ. This means that R yields a consistent update set U in $\mathfrak{A}$ under ζ such that the update $((f, \zeta(x)), \zeta(y))$ is in U. Hence, $f(x) = y$ is true in $\mathfrak{A} + U$ under ζ.

For Axiom 14 assume that $\mathsf{inv}(R, f, x)$ and $f(x) = y$ are true in $\mathfrak{A}$ under ζ. Assume that R yields the consistent update set U in $\mathfrak{A}$ under ζ. Since $\forall y\, \neg\mathsf{upd}(R, f, x, y)$ is true in $\mathfrak{A}$, there is no update for the location $(f, \zeta(x))$ in U and therefore $f(x) = y$ is still true in $\mathfrak{A} + U$ under ζ.

The validity of the extensionality axiom 15 follows from Lemma 8.1.2 and Exercise 8.1.3.

The Axiom 16 is trivial, since **skip** yields the empty update set and $\mathfrak{A} + \emptyset$ is the same as $\mathfrak{A}$.

For the validity of Axiom 17 we use Lemma 2.4.3. Assume that $[P \textbf{ seq } Q]\varphi$ is true in $\mathfrak{A}$ under ζ. Assume that P yields the consistent update set U in $\mathfrak{A}$ under ζ and Q yields the consistent update set V in $\mathfrak{A} + U$. Then, $P \textbf{ seq } Q$ yields $U \oplus V$ in $\mathfrak{A}$ under ζ and, since $U \oplus V$ is consistent, the formula φ is true in $\mathfrak{A} + (U \oplus V)$. By Lemma 2.4.3, the state $\mathfrak{A} + (U \oplus V)$ is the same as $(\mathfrak{A} + U) + V$ and therefore $[Q]\varphi$ is true in $\mathfrak{A} + U$ and $[P][Q]\varphi$ is true in $\mathfrak{A}$. The converse implication of Axiom 17 is shown in a similar way.

8.1.5 Use axiom 5.

8.1.6 That the axiom 8 is derivable from the formulas 73 and 74 can be derived by induction on the length of φ. We can assume that φ is built up from equations $s = t$ or negated equations $s \neq t$ between static terms s and t using $\wedge$, $\vee$, $\forall x$ and $\exists x$. In the case of $\forall x$ one needs the Barcan Axiom 7.

8.1.7 We can derive Axiom 9 from the formula 75 by a case distinction on $\varphi \vee \neg\varphi$ together with the Axiom 8.

8.1.8 The formulas 18–23 and 26 are derivable with the corresponding axioms in the tables 8.2 and 8.3

The equivalence 24 is derivable by a case distinction on $\mathsf{Con}(P) \vee \neg\mathsf{Con}(P)$ in addition to the axioms 5, 9, D7 and U7.

The equivalence 25 is derivable by a case distinction on $\mathsf{Con}(P) \vee \neg\mathsf{Con}(P)$ in addition to the axioms D8 and U8.

8.1.9 For the formula 27 of Lemma 8.1.4 assume $\mathsf{Con}(R)$ and $[R]f(x) = y$.

We make a case distinction on $\mathsf{inv}(R, f, x) \vee \neg\mathsf{inv}(R, f, x)$.

Case 1. $\mathsf{inv}(R, f, x)$: Let z with $f(x) = z$.

By Axiom 14, we obtain $[R]f(x) = z$.

Therefore, $[R]y = z$ and, since $\mathsf{Con}(R)$, by Axiom 9, $y = z$.

Hence, $\mathsf{inv}(R, f, x) \wedge f(x) = y$.

Case 2. $\neg\mathsf{inv}(R, f, x)$: Then $\exists z\, \mathsf{upd}(R, f, x, z)$. Let z with $\mathsf{upd}(R, f, x, z)$.

By Axiom 13, $[R]f(x) = z$ and therefore $[R]y = z$.

Since $\mathsf{Con}(R)$, by Axiom 9, $y = z$.

Hence, $\mathsf{upd}(R, f, x, y)$.

The principle 28 can be derived with Axioms 6 and 9.

The implication $\exists x[R]\varphi \to [R]\exists x\, \varphi$ of the equivalence 29 can easily be derived from $\varphi \to \exists x\, \varphi$ via $[R]\varphi \to [R]\exists x\, \varphi$.

For the converse implication $[R]\exists x\,\varphi \to \exists x[R]\varphi$, assume $[R]\exists x\,\varphi$.
If $\neg\mathsf{Con}(R)$, then we obtain $[R]\varphi$ by Axiom 5 and therefore $\exists x[R]\varphi$.
Otherwise, we have $\mathsf{Con}(R)$. Suppose that $\neg\exists x[R]\varphi$.
Then we can derive:

$$\begin{array}{ll} \forall x \neg[R]\varphi & \\ \forall x[R]\neg\varphi & \text{(Axiom 6)} \\ [R]\forall x\,\neg\varphi & \text{(Axiom 7)} \end{array}$$

Together with the assumption $[R]\exists x\,\varphi$ we obtain $[R]\bot$.
Since we have $\mathsf{Con}(R)$, we can derive $\bot$ using Axiom 9.
Hence, $\exists x[R]\varphi$.

8.1.10 The formulas 30, 32–35, 37 and 38 are derivable with the corresponding axioms in the tables 8.2 and 8.3
The equivalence 31 is derivable with axioms D2 and U2.
The equivalence 36 is derivable with axioms 5 and 6 in addition to D7 and U7 and the property 27.

8.1.11 The property 39 is derivable with axioms U2, 13 and the formulas 19 and 27.
The principle 40 is derivable with Axioms U2, 9, 14 and the formula 19.
By case distinction on $\mathsf{Con}(P \mathbin{\mathbf{par}} Q) \vee \neg\mathsf{Con}(P \mathbin{\mathbf{par}} Q)$ we can derive principles 41 and 42 using axioms 5, 13 and 14 and formula 27.

8.1.12 Let $\mathfrak{A}$ be a state with $a = 0$ and $b = 0$. Let φ be the formula

$$(a = 1 \wedge b = 0) \vee (a = 0 \wedge b = 1).$$

Then, $\mathfrak{A} \models [a := 1]\varphi \wedge [b := 1]\varphi$, but $\mathfrak{A} \not\models [a := 1 \mathbin{\mathbf{par}} b := 1]\varphi$.

8.1.13 Lemma 8.1.7:
Formula 43 can be derived by a case distinction on $\varphi \vee \neg\varphi$:
In case of φ we can derive $\mathbf{if}\ \varphi\ \mathbf{then}\ P\ \mathbf{else}\ Q \mathrel{\dot\simeq} P$.
In case of $\neg\varphi$ we can derive $\mathbf{if}\ \varphi\ \mathbf{then}\ P\ \mathbf{else}\ Q \mathrel{\dot\simeq} Q$.
Then, formula 43 is derivable with the extensionality Axiom 15.
Formula 45 can be derived by a case distinction on

$$(\mathsf{def}(P) \wedge \neg\mathsf{Con}(P)) \vee \neg(\mathsf{def}(P) \wedge \neg\mathsf{Con}(P)).$$

In the first case we can derive $\mathbf{try}\ P\ \mathbf{else}\ Q \mathrel{\dot\simeq} Q$ and in the second case $\mathbf{try}\ P\ \mathbf{else}\ Q \mathrel{\dot\simeq} P$.
Then we use the extensionality Axiom 15 and Axiom 5.

Formula 46 is derivable with the extensionality Axiom 15.

Lemma 8.1.8 derives with the definitions of $\dot{\simeq}$ and $\simeq$.

Lemma 8.1.9:

The principle 47 is derivable with axioms D1, D3, U1 and U3.

The formulas 48–50 can be derived with the Axioms D3 and U3.

The principles 51 and 52 are derivable with Axioms D3, D4, U3 and U4.

Lemma 8.1.10:

The property 54 is derivable with the Axioms 9 and 13.

The principle 55 can be derived with axioms 16 and 18

The property 56 is derivable with Axioms D7, U7, 5, 6 and 17 and the properties 24 and 8.1.4

The formula 57 can be derived with Axioms D4, D7, U4, U7 and 21 and principle 43.

Lemma 8.1.11 can be derived with formulas 29 and 39.

Lemma 8.1.12:

By a case distinction on $\varphi_i \vee \neg\varphi_i$ we can derive $\mathsf{def}(\textbf{if } \varphi_i \textbf{ then } f(s_i) \textbf{ else skip})$ for any i and so also $\mathsf{def}(R)$.
Then, principles 58 and 59 follow with axioms D3, D4 and U2, U3, U4.

Like in the previous examples we can derive $\mathsf{def}(R)$.Then, 60 can be derived with 59 and $\neg\exists y\varphi \to \forall y\neg\varphi$.

Formula 61 can be derived from Lemma 8.1.11 using the following principle:

$$\bigwedge_{i<j} \neg(\varphi_i \wedge \varphi_j) \to \Big(\bigwedge_{i=1}^{n}(\varphi_i \to [f(s_i) := t_i]\psi) \wedge (\neg \bigvee_{i=1}^{n} \varphi \to \psi) \leftrightarrow [R]\psi\Big)$$

Lemma 8.1.13:

Formula 62 is derivable with the definition of **while** φ **do** P and the formulas 18 and 21.

Formula 63 is derivable with the definition of **while** φ **do** P and formula 21.

8.1.14 Principle 76 is derivable with the axioms 8 and 13 and property 27.

8.1.15 Principle 77 can be derived by induction on the size of R with the definition of $\dot{\simeq}$.

8.1.16 Property 78 can be derived with the definitions of $\mathsf{Con}(R)$ and $\mathsf{con}(R)$.

Formula 79 follows from Axiom 12.

By a case distinction on $\mathsf{def}(R) \vee \neg\mathsf{def}(R)$ we can derive property 80 using formula 79.

The property 81 is derivable with Axiom 18.

The formula 82 is derivable with Axiom 19.

One direction of the implication 83 can be derived with principle 20 and the other by a case distinction on $\mathsf{def}(P \textbf{ par } Q) \vee \neg\mathsf{def}(P \textbf{ par } Q)$ and the property 79.

The principle 84 is derivable with Axiom U4.

The property 85 is derivable with Axiom U5.

One direction of the implication 86 can be derived with Axiom 23 and the other by a case distinction on $\mathsf{def}(\textbf{forall } x \textbf{ with } \varphi \textbf{ do } P) \vee \neg\mathsf{def}(\textbf{forall } x \textbf{ with } \varphi \textbf{ do } P)$ and the property 79.

One direction of the implication 87 can be derived with Axiom U7 and the other by a case distinction on $\mathsf{def}(P \textbf{ seq } Q) \vee \neg\mathsf{def}(P \textbf{ seq } Q)$ and the property 79.

One direction of the implication 88 can be derived with Axiom 25 and the other by a case distinction on

$$\mathsf{def}(\textbf{try } P \textbf{ else } Q) \vee \neg\mathsf{def}(\textbf{try } P \textbf{ else } Q)$$

and the property 79.

The property 89 is derivable with Axiom U9.

8.1.17 We extend Table 2.2 by

$$\frac{\mathrm{yields}(P, \mathfrak{A}, \zeta, U)}{\mathrm{yields}(\textbf{try } P \textbf{ catch } T\ Q, \mathfrak{A}, \zeta, U)} \qquad \text{if } U \restriction \mathsf{Loc}(T) \text{ is consistent}$$

$$\frac{\mathrm{yields}(P, \mathfrak{A}, \zeta, U) \quad \mathrm{yields}(Q, \mathfrak{A}, \zeta, V)}{\mathrm{yields}(\textbf{try } P \textbf{ catch } T\ Q, \mathfrak{A}, \zeta, V)} \qquad \text{otherwise}$$

It is convenient to define $\mathsf{Con}(R \restriction T)$ as an abbreviation for

$$\mathsf{def}(R) \wedge \bigwedge_{f(t) \in T} \forall y, z(\mathsf{upd}(R, f, t, y) \wedge \mathsf{upd}(R, f, t, z) \to y = z)$$

The following axioms have to be added to the logic:

1. $\mathsf{def}(\textbf{try } R \textbf{ catch } T\ S) \leftrightarrow (\mathsf{def}(R) \wedge \mathsf{def}(S)) \vee \mathsf{Con}(R \restriction T)$
2. $\mathsf{upd}(\textbf{try } R \textbf{ catch } T\ S, f, x, y) \leftrightarrow$
   $(\mathsf{Con}(R \restriction T) \wedge \mathsf{upd}(R, f, x, y)) \vee (\neg\mathsf{Con}(R \restriction T) \wedge \mathsf{def}(R) \wedge \mathsf{upd}(S, f, x, y))$

The following property can then be derived:

$$\mathsf{Con}(\textbf{try } R \textbf{ catch } T\ S) \leftrightarrow \mathsf{Con}(R) \vee (\mathsf{Con}(S) \wedge \mathsf{def}(R) \wedge \neg\mathsf{Con}(R \restriction T))$$

8.1.18 It is enough to proof the Axiom 17 for first-order φ, because in Sect. 8.1.5 it is shown that in the case of hierarchical ASMs for any formula φ

there exists a first-order formula φ' so that $\varphi \leftrightarrow \varphi'$ is derivable (without using Axioms 15, 16 and 17).

Axiom 17 for hierarchical ASMs is then derived by induction on the size of φ. As in Sect. 8.1.5 we can assume that the formula φ is built up from equations $x = y$ and $f(x) = y$ using boolean connectives and quantifiers. In case of an equation $x = y$, Axioms 5, 6, 8 and 9 are used. In case of and equation $f(x) = y$, Axioms 5, 6, 8, 13 and the formula 27 are used.

8.1.19 It is sufficient to derive the extensionality axiom 15 for first-order formulas φ. In the case of an equation $f(x) = y$ one can use one direction of Exercise 8.1.14.

8.1.20 See the hint above.