

# Detailed definition of ASMs

- Part 1: Abstract states and update sets
- Part 2: Mathematical Logic
- Part 3: Transition rules and runs of ASMs
- Part 4: The reserve of ASMs

# Part 1

Abstract states and update sets

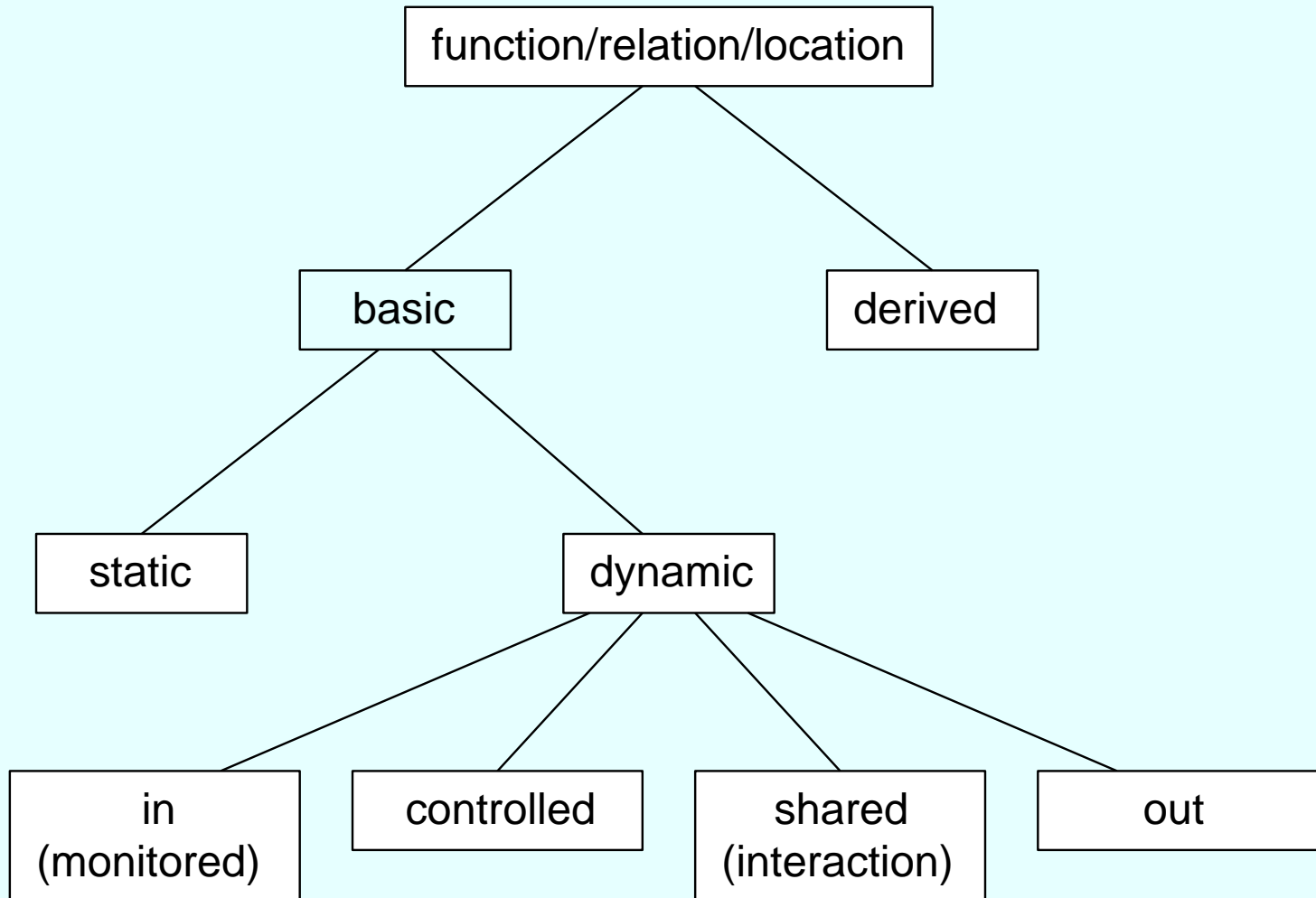
# Signatures

**Definition.** A *signature*  $\Sigma$  is a finite collection of function names.

- Each function name  $f$  has an *arity*, a non-negative integer.
- Nullary function names are called *constants*.
- Function names can be *static* or *dynamic*.
- Every ASM signature contains the static constants *undef*, *true*, *false*.

Signatures are also called *vocabularies*.

# Classification of functions



**Definition.** A *state*  $\mathfrak{A}$  for the signature  $\Sigma$  is a non-empty set  $X$ , the *superuniverse* of  $\mathfrak{A}$ , together with an *interpretation*  $f^{\mathfrak{A}}$  of each function name  $f$  of  $\Sigma$ .

- If  $f$  is an  $n$ -ary function name of  $\Sigma$ , then  $f^{\mathfrak{A}}: X^n \rightarrow X$ .
- If  $c$  is a constant of  $\Sigma$ , then  $c^{\mathfrak{A}} \in X$ .
- The superuniverse  $X$  of the state  $\mathfrak{A}$  is denoted by  $|\mathfrak{A}|$ .

- The superuniverse is also called the *base set* of the state.
- The *elements* of a state are the elements of the superuniverse.

## States (continued)

- The interpretations of *undef*, *true*, *false* are pairwise different.
- The constant *undef* represents an undetermined object.
- The *domain* of an  $n$ -ary function name  $f$  in  $\mathfrak{A}$  is the set of all  $n$ -tuples  $(a_1, \dots, a_n) \in |\mathfrak{A}|^n$  such that  $f^{\mathfrak{A}}(a_1, \dots, a_n) \neq \text{undef}^{\mathfrak{A}}$ .
- A *relation* is a function that has the values *true*, *false* or *undef*.
- We write  $a \in R$  as an abbreviation for  $R(a) = \text{true}$ .
- The superuniverse can be divided into *subuniverses* represented by unary relations.

**Definition.** A *location* of  $\mathfrak{A}$  is a pair

$$(f, (a_1, \dots, a_n))$$

where  $f$  is an  $n$ -ary function name and  $a_1, \dots, a_n$  are elements of  $\mathfrak{A}$ .

- The value  $f^{\mathfrak{A}}(a_1, \dots, a_n)$  is the *content* of the location in  $\mathfrak{A}$ .
- The *elements* of the location are the elements of the set  $\{a_1, \dots, a_n\}$ .
- We write  $\mathfrak{A}(l)$  for the content of the location  $l$  in  $\mathfrak{A}$ .

**Notation.** If  $l = (f, (a_1, \dots, a_n))$  is a location for  $\mathfrak{A}$  and  $\alpha$  is a function defined on  $|\mathfrak{A}|$ , then  $\alpha(l) = (f, (\alpha(a_1), \dots, \alpha(a_n)))$ .

# Updates and update sets

**Definition.** An *update* for  $\mathfrak{A}$  is a pair  $(l, v)$ , where  $l$  is a location of  $\mathfrak{A}$  and  $v$  is an element of  $\mathfrak{A}$ .

- The update is *trivial*, if  $v = \mathfrak{A}(l)$ .
- An *update set* is a set of updates.

**Definition.** An update set  $U$  is *consistent*, if it has no clashing updates, i.e., if for any location  $l$  and all elements  $v, w$ , if  $(l, v) \in U$  and  $(l, w) \in U$ , then  $v = w$ .

# Firing of updates

**Definition.** The result of *firing* a consistent update set  $U$  in a state  $\mathfrak{A}$  is a new state  $\mathfrak{A} + U$  with the same superuniverse as  $\mathfrak{A}$  such that for every location  $l$  of  $\mathfrak{A}$ :

$$(\mathfrak{A} + U)(l) = \begin{cases} v, & \text{if } (l, v) \in U; \\ \mathfrak{A}(l), & \text{if there is no } v \text{ with } (l, v) \in U. \end{cases}$$

The state  $\mathfrak{A} + U$  is called the *sequel* of  $\mathfrak{A}$  with respect to  $U$ .

# Homomorphisms and isomorphisms

Let  $\mathfrak{A}$  and  $\mathfrak{B}$  be two states over the same signature.

**Definition.** A *homomorphism* from  $\mathfrak{A}$  to  $\mathfrak{B}$  is a function  $\alpha$  from  $|\mathfrak{A}|$  into  $|\mathfrak{B}|$  such that  $\alpha(\mathfrak{A}(l)) = \mathfrak{B}(\alpha(l))$  for each location  $l$  of  $\mathfrak{A}$ .

**Definition.** An *isomorphism* from  $\mathfrak{A}$  to  $\mathfrak{B}$  is a homomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$  which is a one-to-one function from  $|\mathfrak{A}|$  onto  $|\mathfrak{B}|$ .

**Lemma (Isomorphism).** Let  $\alpha$  be an isomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$ . If  $U$  is a consistent update set for  $\mathfrak{A}$ , then  $\alpha(U)$  is a consistent update set for  $\mathfrak{B}$  and  $\alpha$  is an isomorphism from  $\mathfrak{A}+U$  to  $\mathfrak{B}+\alpha(U)$ .

# Composition of update sets

$$U \oplus V = V \cup \{(l, v) \in U \mid \text{there is no } w \text{ with } (l, w) \in V\}$$

**Lemma.** Let  $U, V, W$  be update sets.

- $(U \oplus V) \oplus W = U \oplus (V \oplus W)$
- If  $U$  and  $V$  are consistent, then  $U \oplus V$  is consistent.
- If  $U$  and  $V$  are consistent, then  $\mathfrak{A} + (U \oplus V) = (\mathfrak{A} + U) + V$ .

# Part 2

## Mathematical Logic

# Terms

Let  $\Sigma$  be a signature.

**Definition.** The *terms* of  $\Sigma$  are syntactic expressions generated as follows:

- Variables  $x, y, z, \dots$  are terms.
- Constants  $c$  of  $\Sigma$  are terms.
- If  $f$  is an  $n$ -ary function name of  $\Sigma$ ,  $n > 0$ , and  $t_1, \dots, t_n$  are terms, then  $f(t_1, \dots, t_n)$  is a term.

- A term which does not contain variables is called a *ground term*.
- A term is called *static*, if it contains static function names only.
- By  $t \frac{s}{x}$  we denote the result of replacing the variable  $x$  in term  $t$  everywhere by the term  $s$  (*substitution* of  $s$  for  $x$  in  $t$ ).

# Variable assignments

Let  $\mathcal{A}$  be a state.

**Definition.** A *variable assignment* for  $\mathcal{A}$  is a finite function  $\zeta$  which assigns elements of  $|\mathcal{A}|$  to a finite number of variables.

- We write  $\zeta[x \mapsto a]$  for the variable assignment which coincides with  $\zeta$  except that it assigns the element  $a$  to the variable  $x$ :

$$\zeta[x \mapsto a](y) = \begin{cases} a, & \text{if } y = x; \\ \zeta(y), & \text{otherwise.} \end{cases}$$

- Variable assignments are also called *environments*.

# Evaluation of terms

**Definition.** Let  $\mathcal{A}$  be a state of  $\Sigma$ .

Let  $\zeta$  be a variable assignment for  $\mathcal{A}$ .

Let  $t$  be a term of  $\Sigma$  such that all variables of  $t$  are defined in  $\zeta$ .

The *value*  $\llbracket t \rrbracket_{\zeta}^{\mathcal{A}}$  is defined as follows:

- $\llbracket x \rrbracket_{\zeta}^{\mathcal{A}} = \zeta(x)$
- $\llbracket c \rrbracket_{\zeta}^{\mathcal{A}} = c^{\mathcal{A}}$
- $\llbracket f(t_1, \dots, t_n) \rrbracket_{\zeta}^{\mathcal{A}} = f^{\mathcal{A}}(\llbracket t_1 \rrbracket_{\zeta}^{\mathcal{A}}, \dots, \llbracket t_n \rrbracket_{\zeta}^{\mathcal{A}})$

## Evaluation of terms (continued)

**Lemma (Coincidence).** If  $\zeta$  and  $\eta$  are two variable assignments for  $t$  such that  $\zeta(x) = \eta(x)$  for all variables  $x$  of  $t$ , then  $\llbracket t \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket t \rrbracket_{\eta}^{\mathfrak{A}}$ .

**Lemma (Homomorphism).** If  $\alpha$  is a homomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$ , then  $\alpha(\llbracket t \rrbracket_{\zeta}^{\mathfrak{A}}) = \llbracket t \rrbracket_{\alpha \circ \zeta}^{\mathfrak{B}}$  for each term  $t$ .

**Lemma (Substitution).** Let  $a = \llbracket s \rrbracket_{\zeta}^{\mathfrak{A}}$ .  
Then  $\llbracket t \frac{s}{x} \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket t \rrbracket_{\zeta[x \mapsto a]}^{\mathfrak{A}}$ .

# Formulas

Let  $\Sigma$  be a signature.

**Definition.** The *formulas* of  $\Sigma$  are generated as follows:

- If  $s$  and  $t$  are terms of  $\Sigma$ , then  $s = t$  is a formula.
- If  $\varphi$  is a formula, then  $\neg\varphi$  is a formula.
- If  $\varphi$  and  $\psi$  are formulas, then  $(\varphi \wedge \psi)$ ,  $(\varphi \vee \psi)$  and  $(\varphi \rightarrow \psi)$  are formulas.
- If  $\varphi$  is a formula and  $x$  a variable, then  $(\forall x \varphi)$  and  $(\exists x \varphi)$  are formulas.

- A formula  $s = t$  is called an *equation*.
- The expression  $s \neq t$  is an abbreviation for  $\neg(s = t)$ .

# Formulas (continued)

symbol	name	meaning
$\neg$	negation	not
$\wedge$	conjunction	and
$\vee$	disjunction	or (inclusive)
$\rightarrow$	implication	if-then
$\forall$	universal quantification	for all
$\exists$	existential quantification	there is

## Formulas (continued)

$\varphi \wedge \psi \wedge \chi$  stands for  $((\varphi \wedge \psi) \wedge \chi)$ ,

$\varphi \vee \psi \vee \chi$  stands for  $((\varphi \vee \psi) \vee \chi)$ ,

$\varphi \wedge \psi \rightarrow \chi$  stands for  $((\varphi \wedge \psi) \rightarrow \chi)$ , etc.

- The variable  $x$  is *bound* by the quantifier  $\forall$  ( $\exists$ ) in  $\forall x \varphi$  ( $\exists x \varphi$ ).
- The *scope* of  $x$  in  $\forall x \varphi$  ( $\exists x \varphi$ ) is the formula  $\varphi$ .
- A variable  $x$  occurs *free* in a formula, if it is not in the scope of a quantifier  $\forall x$  or  $\exists x$ .
- By  $\varphi \frac{t}{x}$  we denote the result of replacing all free occurrences of the variable  $x$  in  $\varphi$  by the term  $t$ . (Bound variables are renamed.)

# Semantics of formulas

$$\llbracket s = t \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket s \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket t \rrbracket_{\zeta}^{\mathfrak{A}}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \neg\varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{false}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \varphi \wedge \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true} \text{ and } \llbracket \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \varphi \vee \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true} \text{ or } \llbracket \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \varphi \rightarrow \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{false} \text{ or } \llbracket \psi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \forall x \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if } \llbracket \varphi \rrbracket_{\zeta[x \mapsto a]}^{\mathfrak{A}} = \text{true} \text{ for every } a \in |\mathfrak{A}|; \\ \text{false}, & \text{otherwise.} \end{cases}$$

$$\llbracket \exists x \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \begin{cases} \text{true}, & \text{if there exists an } a \in |\mathfrak{A}| \text{ with } \llbracket \varphi \rrbracket_{\zeta[x \mapsto a]}^{\mathfrak{A}} = \text{true}; \\ \text{false}, & \text{otherwise.} \end{cases}$$

# Coincidence, Substitution, Isomorphism

**Lemma (Coincidence).** If  $\zeta$  and  $\eta$  are two variable assignments for  $\varphi$  such that  $\zeta(x) = \eta(x)$  for all free variables  $x$  of  $\varphi$ , then  $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket \varphi \rrbracket_{\eta}^{\mathfrak{A}}$ .

**Lemma (Substitution).** Let  $t$  be a term and  $a = \llbracket t \rrbracket_{\zeta}^{\mathfrak{A}}$ . Then  $\llbracket \varphi \frac{t}{x} \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket \varphi \rrbracket_{\zeta[x \mapsto a]}^{\mathfrak{A}}$ .

**Lemma (Isomorphism).** Let  $\alpha$  be an isomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$ . Then  $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \llbracket \varphi \rrbracket_{\alpha \circ \zeta}^{\mathfrak{B}}$ .

# Models

**Definition.** A state  $\mathfrak{A}$  is a *model* of  $\varphi$  (written  $\mathfrak{A} \models \varphi$ ), if  $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = true$  for all variable assignments  $\zeta$  for  $\varphi$ .

# Part 3

## Transition rules and runs of ASMs

# Transition rules

*Skip Rule:*

**skip**

Meaning: Do nothing

*Update Rule:*

$f(s_1, \dots, s_n) := t$

Meaning: Update the value of  $f$  at  $(s_1, \dots, s_n)$  to  $t$ .

*Block Rule:*

**$P$  par  $Q$**

Meaning:  $P$  and  $Q$  are executed in parallel.

*Conditional Rule:*

**if  $\varphi$  then  $P$  else  $Q$**

Meaning: If  $\varphi$  is true, then execute  $P$ , otherwise execute  $Q$ .

*Let Rule:*

**let  $x = t$  in  $P$**

Meaning: Assign the value of  $t$  to  $x$  and then execute  $P$ .

## Transition rules (continued)

*Forall Rule:*

**forall  $x$  with  $\varphi$  do  $P$**

Meaning: Execute  $P$  in parallel for each  $x$  satisfying  $\varphi$ .

*Choose Rule:*

**choose  $x$  with  $\varphi$  do  $P$**

Meaning: Choose an  $x$  satisfying  $\varphi$  and then execute  $P$ .

*Sequence Rule:*

**$P$  seq  $Q$**

Meaning:  $P$  and  $Q$  are executed sequentially, first  $P$  and then  $Q$ .

*Call Rule:*

**$r(t_1, \dots, t_n)$**

Meaning: Call transition rule  $r$  with parameters  $t_1, \dots, t_n$ .

# Variations of the syntax

<b>if</b> $\varphi$ <b>then</b> $P$ <b>else</b> $Q$ <b>endif</b>	<b>if</b> $\varphi$ <b>then</b> $P$ <b>else</b> $Q$
[ <b>do in-parallel</b> ] $P_1$ $\vdots$ $P_n$ [ <b>enddo</b> ]	$P_1$ <b>par</b> ... <b>par</b> $P_n$
$\{P_1, \dots, P_n\}$	$P_1$ <b>par</b> ... <b>par</b> $P_n$

## Variations of the syntax (continued)

<b>do forall</b> $x: \varphi$ $P$ <b>enddo</b>	<b>forall</b> $x$ <b>with</b> $\varphi$ <b>do</b> $P$
<b>choose</b> $x: \varphi$ $P$ <b>endchoose</b>	<b>choose</b> $x$ <b>with</b> $\varphi$ <b>do</b> $P$
<b>step</b> $P$ <b>step</b> $Q$	$P$ <b>seq</b> $Q$

# Free and bound variables

**Definition.** An occurrence of a variable  $x$  is *free* in a transition rule, if it is not in the scope of a **let**  $x$ , **forall**  $x$  or **choose**  $x$ .

**let**  $x = t$  **in**  $P$   
scope of  $x$

**forall**  $x$  **with**  $\varphi$  **do**  $P$   
scope of  $x$

**choose**  $x$  **with**  $\varphi$  **do**  $P$   
scope of  $x$

# Rule declarations

**Definition.** A *rule declaration* for a rule name  $r$  of arity  $n$  is an expression

$$r(x_1, \dots, x_n) = P$$

where

- $P$  is a transition rule and
- the free variables of  $P$  are contained in the list  $x_1, \dots, x_n$ .

**Remark:** Recursive rule declarations are allowed.

# Abstract State Machines

**Definition.** An *abstract state machine*  $M$  consists of

- a signature  $\Sigma$ ,
- a set of initial states for  $\Sigma$ ,
- a set of rule declarations,
- a distinguished rule name of arity zero called the *main rule name* of the machine.

# Semantics of transition rules

The semantics of transition rules is defined in a calculus by rules:

$$\frac{Premise_1 \cdots Premise_n}{Conclusion} \textit{Condition}$$

The predicate

$$\textit{yields}(P, \mathcal{A}, \zeta, U)$$

means:

The transition rule  $P$  yields the update set  $U$  in state  $\mathcal{A}$  under the variable assignment  $\zeta$ .

# Semantics of transition rules (continued)

$$\frac{}{\text{yields}(\mathbf{skip}, \mathfrak{A}, \zeta, \emptyset)}$$

$$\frac{}{\text{yields}(f(s_1, \dots, s_n) := t, \mathfrak{A}, \zeta, \{(l, v)\})}$$

where  $l = (f, ([s_1]_{\zeta}^{\mathfrak{A}}, \dots, [s_n]_{\zeta}^{\mathfrak{A}}))$   
and  $v = [t]_{\zeta}^{\mathfrak{A}}$

$$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U) \quad \text{yields}(Q, \mathfrak{A}, \zeta, V)}{\text{yields}(P \mathbf{par} Q, \mathfrak{A}, \zeta, U \cup V)}$$

$$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U)}{\text{yields}(\mathbf{if} \varphi \mathbf{then} P \mathbf{else} Q, \mathfrak{A}, \zeta, U)}$$

if  $[\varphi]_{\zeta}^{\mathfrak{A}} = \mathit{true}$

$$\frac{\text{yields}(Q, \mathfrak{A}, \zeta, V)}{\text{yields}(\mathbf{if} \varphi \mathbf{then} P \mathbf{else} Q, \mathfrak{A}, \zeta, V)}$$

if  $[\varphi]_{\zeta}^{\mathfrak{A}} = \mathit{false}$

$$\frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U)}{\text{yields}(\mathbf{let} x = t \mathbf{in} P, \mathfrak{A}, \zeta, U)}$$

where  $a = [t]_{\zeta}^{\mathfrak{A}}$

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

where  $I = \mathit{range}(x, \varphi, \mathfrak{A}, \zeta)$

# Semantics of transition rules (continued)

$\frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U)}{\text{yields}(\mathbf{choose } x \mathbf{ with } \varphi \mathbf{ do } P, \mathfrak{A}, \zeta, U)}$	if $a \in \text{range}(x, \varphi, \mathfrak{A}, \zeta)$
$\frac{}{\text{yields}(\mathbf{choose } x \mathbf{ with } \varphi \mathbf{ do } P, \mathfrak{A}, \zeta, \emptyset)}$	if $\text{range}(x, \varphi, \mathfrak{A}, \zeta) = \emptyset$
$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U) \quad \text{yields}(Q, \mathfrak{A} + U, \zeta, V)}{\text{yields}(P \mathbf{ seq } Q, \mathfrak{A}, \zeta, U \oplus V)}$	if $U$ is consistent
$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U)}{\text{yields}(P \mathbf{ seq } Q, \mathfrak{A}, \zeta, U)}$	if $U$ is inconsistent
$\frac{\text{yields}(P \frac{t_1 \dots t_n}{x_1 \dots x_n}, \mathfrak{A}, \zeta, U)}{\text{yields}(r(t_1, \dots, t_n), \mathfrak{A}, \zeta, U)}$	where $r(x_1, \dots, x_n) = P$ is a rule declaration of $M$

$$\text{range}(x, \varphi, \mathfrak{A}, \zeta) = \{a \in |\mathfrak{A}| : \llbracket \varphi \rrbracket_{\zeta[x \mapsto a]}^{\mathfrak{A}} = \text{true}\}$$

# Coincidence, Substitution, Isomorphisms

**Lemma (Coincidence).** If  $\zeta(x) = \eta(x)$  for all free variables  $x$  of a transition rule  $P$  and  $P$  yields  $U$  in  $\mathfrak{A}$  under  $\zeta$ , then  $P$  yields  $U$  in  $\mathfrak{A}$  under  $\eta$ .

**Lemma (Substitution).** Let  $t$  be a static term and  $a = \llbracket t \rrbracket_{\zeta}^{\mathfrak{A}}$ . Then the rule  $P \frac{t}{x}$  yields the update set  $U$  in state  $\mathfrak{A}$  under  $\zeta$  iff  $P$  yields  $U$  in  $\mathfrak{A}$  under  $\zeta[x \mapsto a]$ .

**Lemma (Isomorphism).** If  $\alpha$  is an isomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$  and  $P$  yields  $U$  in  $\mathfrak{A}$  under  $\zeta$ , then  $P$  yields  $\alpha(U)$  in  $\mathfrak{B}$  under  $\alpha \circ \zeta$ .

# Move of an ASM

**Definition.** A machine  $M$  can make a *move* from state  $\mathfrak{A}$  to  $\mathfrak{B}$  (written  $\mathfrak{A} \xRightarrow{M} \mathfrak{B}$ ), if the main rule of  $M$  yields a consistent update set  $U$  in state  $\mathfrak{A}$  and  $\mathfrak{B} = \mathfrak{A} + U$ .

- The updates in  $U$  are called *internal updates*.
- $\mathfrak{B}$  is called the *next internal state*.

If  $\alpha$  is an isomorphism from  $\mathfrak{A}$  to  $\mathfrak{A}'$ , the following diagram commutes:

$$\begin{array}{ccc} \mathfrak{A} & \xRightarrow{M} & \mathfrak{B} \\ \alpha \downarrow & & \downarrow \alpha \\ \mathfrak{A}' & \xRightarrow{M} & \mathfrak{B}' \end{array}$$

# Run of an ASM

Let  $M$  be an ASM with signature  $\Sigma$ .

A *run* of  $M$  is a finite or infinite sequence  $\mathcal{A}_0, \mathcal{A}_1, \dots$  of states for  $\Sigma$  such that

- $\mathcal{A}_0$  is an initial state of  $M$
- for each  $n$ ,
  - either  $M$  can make a move from  $\mathcal{A}_n$  into the next internal state  $\mathcal{A}'_n$  and the environment produces a consistent set of external or shared updates  $U$  such that  $\mathcal{A}_{n+1} = \mathcal{A}'_n + U$ ,
  - or  $M$  cannot make a move in state  $\mathcal{A}_n$  and  $\mathcal{A}_n$  is the last state in the run.

- In *internal* runs, the environment makes no moves.
- In *interactive* runs, the environment produces updates.

# Part 4

The reserve of ASMs

# Importing new elements from the reserve

*Import rule:*

**import**  $x$  **do**  $P$

Meaning: Choose an element  $x$  from the reserve, delete it from the reserve and execute  $P$ .

**let**  $x = new(X)$  **in**  $P$

abbreviates

**import**  $x$  **do**  
 $X(x) := true$   
 $P$

## The reserve of a state

- New dynamic relation *Reserve*.
- *Reserve* is updated by the system, not by rules.
- $Res(\mathfrak{A}) = \{a \in |\mathfrak{A}| : Reserve^{\mathfrak{A}}(a) = true\}$
- The reserve elements of a state are not allowed to be in the domain and range of any basic function of the state.

**Definition.** A state  $\mathfrak{A}$  satisfies the *reserve condition* with respect to an environment  $\zeta$ , if the following two conditions hold for each element  $a \in Res(\mathfrak{A}) \setminus ran(\zeta)$ :

- The element  $a$  is not the content of a location of  $\mathfrak{A}$ .
- If  $a$  is an element of a location  $l$  of  $\mathfrak{A}$  which is not a location for *Reserve*, then the content of  $l$  in  $\mathfrak{A}$  is *undef*.

# Semantics of ASMs with a reserve

$$\begin{array}{l}
 \frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U)}{\text{yields}(\mathbf{import} \ x \ \mathbf{do} \ P, \mathfrak{A}, \zeta, V)} \quad \text{if } a \in \text{Res}(\mathfrak{A}) \setminus \text{ran}(\zeta) \text{ and} \\
 \quad \quad \quad V = U \cup \{((\text{Reserve}, a), \text{false})\} \\
 \\
 \frac{\text{yields}(P, \mathfrak{A}, \zeta, U) \quad \text{yields}(Q, \mathfrak{A}, \zeta, V)}{\text{yields}(P \ \mathbf{par} \ Q, \mathfrak{A}, \zeta, U \cup V)} \quad \text{if } \text{Res}(\mathfrak{A}) \cap \text{El}(U) \cap \text{El}(V) \subseteq \text{ran}(\zeta) \\
 \\
 \frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U_a) \quad \text{for each } a \in I}{\text{yields}(\mathbf{forall} \ x \ \mathbf{with} \ \varphi \ \mathbf{do} \ P, \mathfrak{A}, \zeta, \bigcup_{a \in I} U_a)} \quad \text{if } I = \text{range}(x, \varphi, \mathfrak{A}, \zeta) \text{ and for } a \neq b \\
 \quad \quad \quad \text{Res}(\mathfrak{A}) \cap \text{El}(U_a) \cap \text{El}(U_b) \subseteq \text{ran}(\zeta)
 \end{array}$$

- $\text{El}(U)$  is the set of elements that occur in the updates of  $U$ .
- The elements of an update  $(l, v)$  are the value  $v$  and the elements of the location  $l$ .

# Problem

Problem 1: New elements that are imported in parallel must be different.

```
import  $x$  do  $parent(x) = root$ 
```

```
import  $y$  do  $parent(y) = root$ 
```

Problem 2: Hiding of bound variables.

```
import  $x$  do
```

```
   $f(x) := 0$ 
```

```
  let  $x = 1$  in
```

```
    import  $y$  do  $f(y) := x$ 
```

**Syntactic constraint.** In the scope of a bound variable the same variable should not be used again as a bound variable (**let**, **forall**, **choose**, **import**).

## Lemma (Preservation of the reserve condition).

If a state  $\mathfrak{A}$  satisfies the reserve condition wrt.  $\zeta$  and  $P$  yields a consistent update set  $U$  in  $\mathfrak{A}$  under  $\zeta$ , then

- the sequel  $\mathfrak{A} + U$  satisfies the reserve condition wrt.  $\zeta$ ,
- $Res(\mathfrak{A} + U) \setminus ran(\zeta)$  is contained in  $Res(\mathfrak{A}) \setminus El(U)$ .

# Permutation of the reserve

**Lemma (Permutation of the reserve).** Let  $\mathfrak{A}$  be a state that satisfies the reserve condition wrt.  $\zeta$ . If  $\alpha$  is a function from  $|\mathfrak{A}|$  to  $|\mathfrak{A}|$  that permutes the elements in  $Res(\mathfrak{A}) \setminus ran(\zeta)$  and is the identity on non-reserve elements of  $\mathfrak{A}$  and on elements in the range of  $\zeta$ , then  $\alpha$  is an isomorphism from  $\mathfrak{A}$  to  $\mathfrak{A}$ .

## Lemma (Independence).

Let  $P$  be a rule of an ASM without **choose**. If

- $\mathfrak{A}$  satisfies the reserve condition wrt.  $\zeta$ ,
- the bound variables of  $P$  are not in the domain of  $\zeta$ ,
- $P$  yields  $U$  in  $\mathfrak{A}$  under  $\zeta$ ,
- $P$  yields  $U'$  in  $\mathfrak{A}$  under  $\zeta$ ,

then there exists a permutation  $\alpha$  of  $Res(\mathfrak{A}) \setminus ran(\zeta)$  such that  $\alpha(U) = U'$ .