

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

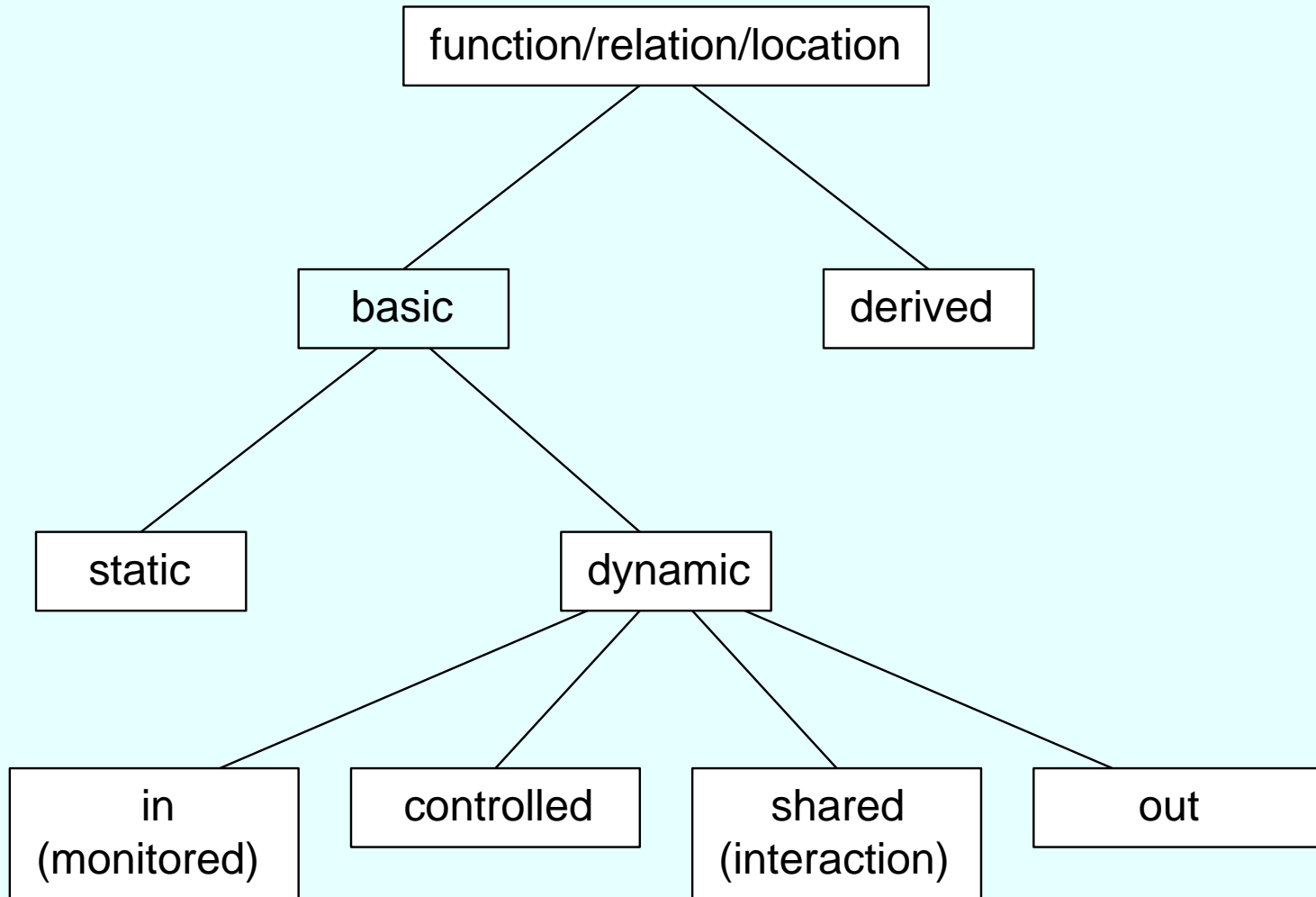
Signatures

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

- If f is an n -ary function name of Σ , then $f^{\mathfrak{A}}: X^n \rightarrow X$.
- If c is a constant of Σ , 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.

Locations

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 of \mathfrak{A} and α 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 α 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 α 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 α 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 Σ be a signature.

Definition. The *terms* of Σ are syntactic expressions generated as follows:

- Variables x, y, z, \dots are terms.
- Constants c of Σ are terms.
- If f is an n -ary function name of Σ , $n > 0$, 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 \mathfrak{A} be a state.

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

- We write $\zeta[x \mapsto a]$ for the variable assignment which coincides with ζ 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 \mathfrak{A} be a state of Σ .

Let ζ be a variable assignment for \mathfrak{A} .

Let t be a term of Σ such that all variables of t are defined in ζ .

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

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

Evaluation of terms (continued)

Lemma (Coincidence). If ζ and η 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 α 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 Σ be a signature.

Definition. The *formulas* of Σ are generated as follows:

- If s and t are terms of Σ , then $s = t$ is a formula.
- If φ is a formula, then $\neg\varphi$ is a formula.
- If φ and ψ are formulas, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ and $(\varphi \rightarrow \psi)$ are formulas.
- If φ 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 φ .
- 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 φ by the term t . (Bound variables are renamed.)

Semantics of formulas

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

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

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

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

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

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

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

Coincidence, Substitution, Isomorphism

Lemma (Coincidence). If ζ and η are two variable assignments for φ such that $\zeta(x) = \eta(x)$ for all free variables x of φ , 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 α 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 φ (written $\mathfrak{A} \models \varphi$), if $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true}$ for all variable assignments ζ for φ .

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 φ then P else Q

Meaning: If φ 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 φ do P

Meaning: Execute P in parallel for each x satisfying φ .

Choose Rule:

choose x with φ do P

Meaning: Choose an x satisfying φ 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

if φ then P else Q endif	if φ then P else Q
[do in-parallel] P_1 \vdots P_n [enddo]	P_1 par \dots par P_n
$\{P_1, \dots, P_n\}$	P_1 par \dots par P_n

Variations of the syntax (continued)

do forall $x: \varphi$ P enddo	forall x with φ do P
choose $x: \varphi$ P endchoose	choose x with φ do P
step P step Q	P seq 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 .

$$\text{let } x = t \text{ in } \underbrace{P}_{\text{scope of } x}$$
$$\text{forall } x \underbrace{\text{with } \varphi \text{ do } P}_{\text{scope of } x}$$
$$\text{choose } x \underbrace{\text{with } \varphi \text{ do } P}_{\text{scope of } x}$$

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 Σ ,
- a set of initial states for Σ ,
- 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:

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

The predicate

$$\text{yields}(P, \mathfrak{A}, \zeta, U)$$

means:

The transition rule P yields the update set U in state \mathfrak{A} under the variable assignment ζ .

Semantics of transition rules (continued)

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

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

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

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

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

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

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

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

if $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{true}$

if $\llbracket \varphi \rrbracket_{\zeta}^{\mathfrak{A}} = \text{false}$

where $a = \llbracket t \rrbracket_{\zeta}^{\mathfrak{A}}$

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

Semantics of transition rules (continued)

$\frac{\text{yields}(P, \mathfrak{A}, \zeta[x \mapsto a], U)}{\text{yields}(\text{choose } x \text{ with } \varphi \text{ do } P, \mathfrak{A}, \zeta, U)}$	if $a \in \text{range}(x, \varphi, \mathfrak{A}, \zeta)$
$\frac{}{\text{yields}(\text{choose } x \text{ with } \varphi \text{ do } P, \mathfrak{A}, \zeta, \emptyset)}$	if $\text{range}(x, \varphi, \mathfrak{A}, \zeta) = \emptyset$
$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U) \quad \text{yields}(Q, \mathfrak{A} + U, \zeta, V)}{\text{yields}(P \text{ seq } Q, \mathfrak{A}, \zeta, U \oplus V)}$	if U is consistent
$\frac{\text{yields}(P, \mathfrak{A}, \zeta, U)}{\text{yields}(P \text{ seq } Q, \mathfrak{A}, \zeta, U)}$	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 ζ , then P yields U in \mathfrak{A} under η .

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 ζ iff P yields U in \mathfrak{A} under $\zeta[x \mapsto a]$.

Lemma (Isomorphism). If α is an isomorphism from \mathfrak{A} to \mathfrak{B} and P yields U in \mathfrak{A} under ζ , 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 α 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 Σ .

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

- \mathfrak{A}_0 is an initial state of M
- for each n ,
 - either M can make a move from \mathfrak{A}_n into the next internal state \mathfrak{A}'_n and the environment produces a consistent set of external or shared updates U such that $\mathfrak{A}_{n+1} = \mathfrak{A}'_n + U$,
 - or M cannot make a move in state \mathfrak{A}_n and \mathfrak{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 $x = \text{new}(X)$ **in** P

abbreviates

import x **do**
 $X(x) := \text{true}$
 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 ζ , 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

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

- $\text{El}(U)$ 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. ζ and P yields a consistent update set U in \mathfrak{A} under ζ , then

- the sequel $\mathfrak{A} + U$ satisfies the reserve condition wrt. ζ ,
- $Res(\mathfrak{A} + U) \setminus ran(\zeta)$ 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. ζ . If α 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 ζ , then α is an isomorphism from \mathfrak{A} to \mathfrak{A} .

Independence of the choice of reserve elements

Lemma (Independence).

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

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

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