Processos e Concorrência 2015/16
Bloco de acetatos 7

Alexandre Madeira
HASLab INESC TEC, DI UMINHO

April 6, 2017

Algebraic specification is everywhere

Definition A (Monoid)

A *monoid* is an algebraic structure with a single associative binary operation and an identity element.

Algebraic specification is everywhere

Definition A (Monoid)

A *monoid* is an algebraic structure with a single associative binary operation and an identity element.

Definition B (Monoid)

A *monoid* is a model of the following specification:

Sorts s;

Op $\cdot : s \times s \to s$

$e :\to s$

Ax $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

$a \cdot e = a$

$e \cdot a = a$

Algebraic specification is everywhere

Specification of a computer memory

Sorts state;

id;

elt;

Algebraic specification is everywhere

Specification of a computer memory

Sorts state;

id;

elt;

Op update: id $\times$ elt $\times$state $\rightarrow$ state;

lookup: id $\times$ state $\rightarrow$ elt;

Algebraic specification is everywhere

Specification of a computer memory

Sorts state;

id;

elt;

Op update: id $\times$ elt $\times$state $\rightarrow$ state;

lookup: id $\times$ state $\rightarrow$ elt;

Ax $lookup(i, update(i, e, s)) = e$

$i \neq j \rightarrow lookup(i, update(j, e, s)) = lookup(i, s)$

Algebraic specification of data and processes

Algebraic specification

- rooted in universal algebra
- Original aims: **modelling software systems as algebras**
- specialised along the times: **definition of ADT (abstract data types)**

Algebraic specification of data and processes

Algebraic specification

- rooted in universal algebra
- Original aims: **modelling software systems as algebras**
- specialised along the times: **definition of ADT (abstract data types)**

Process algebra

- rooted in automata and languages theory
- aims to formal modelling and analysis of concurrent systems

Algebraic specification in this course

AS and PA are highly related:

Micro perspective

- actions can be parametrized by data – we need a rigorous way to define new suitable ADT

Macro perspective

- a process algebra can be seen, itself, as an algebraic specification

Revisiting mCRL2

e.g., the fragment

$$
\begin{array}{lcl}
A1 & x + y & = y + x \\
A2 & (x + y) + z & = x + (y + z) \\
A3 & x + x & = x \\
A4 & (x + y) \cdot z & = x \cdot z + y \cdot z \\
A5 & (x \cdot y) \cdot z & = x \cdot (y \cdot z) \\
A6 & x + \delta & = x \\
A7 & \delta \cdot x & = \delta
\end{array}
$$

- actually mCRL2 can be seen as an algebraic specification
- the mCRL2 axiomatics can be understood as an algebraic specification for the "generic shape of processes"

Revisiting CCS

e.g., the fragment

$$\frac{}{a.p \xrightarrow{a} p}\,(act)$$

$$\frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'}\,(sum - l) \qquad \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}\,(sum - r)$$

- the axiomatics of CCS is now given by a set of inference rules (against the equational axiomatization of mCRL2)
- the previous analogy remains valid

Universal Algebra

Many-sorted sets

Definition (Many-sorted set)

An S-sorted set is an S-family of sets $X = (X_s)_{s \in S}$.

Many-sorted sets

Definition (Many-sorted set)

An S-sorted set is an S-family of sets $X = (X_s)_{s \in S}$.

Given two S-sorted sets $X = (X_s)_{s \in S}$ and $Y = (Y_s)_{s \in S}$:

$X \cup Y = (X_s \cup Y_s)_{s \in S}$

$X \times Y = (X_s \times Y_s)_{s \in S}$

$X \subseteq Y$ iff for each $s \in S$, $X_s \subseteq Y_s$

$\ldots$

Many-sorted sets

Definition (Many-sorted set function)

Let $X = (X_s)_{s \in S}$ and $Y = (Y_s)_{s \in S}$ two S-sorted sets. An S-sorted function $f : X \to Y$ is an S-family of functions $f_s : X_s \to Y_s$.

Many-sorted sets

Many-sorted relation

Let $X = (X_s)_{s \in S}$ and $Y = (Y_s)_{s \in S}$ two S-sorted sets. An S-sorted relation $R \subseteq X \times Y$ is an S-family of relations $R_s \subseteq X_s \times Y_s$.

Many-sorted equivalence relation

An S-sorted relation $X \subseteq R \times R$ is an equivalence on X if, for each $s \in S$:

- R_s is reflexive, i.e., for any $x \in X_s$, $(x, x) \in R_s$;
- R_s is simetric, i.e., for any $x, y \in X_s$, $(x, y) \in R_s$ implies $(y, x) \in R_s$;
- R_s is transitive, i.e., for any $x, y, z \in X_s$, $(x, y) \in R_s$ and $(y, z) \in R_s$ implies $(x, z) \in R_s$;.

Many-sorted sets

Quotient set

Given an S-sorted equivalence relation $R \subseteq X \times X$:

- **the equivalence class of $x \in X_s$ modulo R:** $[x]_{R_s} = \{y \in X_s \mid (x, y) \in R_s\}$,
- **the quotient of X modulo R:** $X/R = (X_s/R_s)_{s \in S}$, where $X_s/R_s = \{[x]_{R_s} | x \in X_s\}$.

Exercise E1

1. Show that, in any S-sorted equivalence relation, $[x]_{\equiv_s} = [y]_{\equiv_s}$ iff $x \equiv_s y$.
2. Let consider the sorted set $X = (X_s)_{s \in \{s_1, s_2\}}$, with $X_{s_1} = \{1, 2\}$ and $X_{s_2} = \{a, b, c\}$. Identify, in the following relations, the equivalence relations. Define, for these cases, the respective quotient set.
   a) $R_1 = \{(1,1), (2,2)\}$ and $R_2 = \{(a,a), (b,b), (c,c)\}$
   b) $S_1 = \{(1,1), (2,2), (1,2), (2,1)\}$ and $S_2 = \{(a,a), (b,b), (c,c)\}$
   c) $T_1 = \{(1,1), (2,2)\}$ and $T_2 = \{(a,a), (b,b), (b,c), (c,b), (c,c)\}$
   d) $Q_1 = \{(1,2), (2,1)\}$ and $Q_2 = \{(a,a), (b,b), (b,c), (c,b), (c,c)\}$

Signatures and Algebras

Definition (Many-sorted signature)

A signature is a pair $\Sigma = (S, \Omega)$, where:

- S is a **set of sort names**
- Ω is a $(S^* \times S)$-sorted **set of operation names**

Notation - as usual we may denote

- operation names $f \in \Omega_{s_1 \ldots s_n, s}$ by $f : s_1 \times \cdots \times s_n \to s \in \Sigma$
- (in particular) constants names $c \in \Omega_{\epsilon, s}$ by $c : s \in \Sigma$

Signatures and Algebras

Signature example: $\Sigma = (S, \Omega)$, where

- $S = \{s_1, s_2\}$
- $\Omega_{\epsilon,s_1} = \{c_1\}$, $\Omega_{\epsilon,s_2} = \{c_2\}$
- $\Omega_{s_1,s_1} = \{f\}$, $\Omega_{s_2 s_1,s_1} = \{g\}$
- $\Omega_{\omega,s} = \emptyset$ for other $\omega \in S^*$, $s \in S$

Signatures and Algebras

Signature example: $\Sigma = (S, \Omega)$, where

- $S = \{s_1, s_2\}$
- $\Omega_{\epsilon,s_1} = \{c_1\}$, $\Omega_{\epsilon,s_2} = \{c_2\}$
- $\Omega_{s_1,s_1} = \{f\}$, $\Omega_{s_2 s_1,s_1} = \{g\}$
- $\Omega_{\omega,s} = \emptyset$ for other $\omega \in S^*$, $s \in S$

A more standard presentation

$$
\begin{array}{rl}
\text{sorts} & s_1 \\
 & s_2 \\
\text{op} & c_1 :\rightarrow s_1 \\
 & c_2 :\rightarrow s_2 \\
 & f : s_1 \rightarrow s_1 \\
 & g : s_2 \times s_1 \rightarrow s_1
\end{array}
$$

Signatures and Algebras

Exercise E2

1. define a signature for groups
2. define a signature for natural numbers
3. define a signature for stacks of elements of a set A
4. define a signature for boolean algebra

Signatures and Algebras

Exercise E2

1. define a signature for groups
2. define a signature for natural numbers
3. define a signature for stacks of elements of a set A
4. define a signature for boolean algebra
5. a signature for a simple process algebra?

Signatures and Algebras

Exercise E2

1. define a signature for groups
2. define a signature for natural numbers
3. define a signature for stacks of elements of a set A
4. define a signature for boolean algebra
5. a signature for a simple process algebra?
   - e.g., for the fragment of CCS

$$P := 0 \mid a.P \mid P + P \mid P|P$$

for a fixed set of actions A

Signatures and Algebras

Definition (Many-sorted algebra)

Let $\Sigma = (S, \Omega)$ be a many-sorted signature. A **Σ-algebra** A consists of:

- an S-sorted set $|A|$, i.e., for each $s \in S$, $|A|_s$ is a set; and
- for each $f : s_1 \times \cdots \times s_n \to s \in \Sigma$, a function $f^A : |A|_{s_1} \times \cdots \times |A|_{s_1} \to |A|_s$.

Signatures and Algebras

Let consider the signature Σ:

sorts s_1, s_2

op $c_1 :\rightarrow s_1,\ c_2 :\rightarrow s_2$

$f : s_1 \rightarrow s_1$

$g : s_2 \times s_1 \rightarrow s_1$

Two example of Σ-algebras:

$|A|_{s_1} = \{a, b\},\ |A|_{s_2} = \{1, 2, 3\}$

$c_1^A = a\ c_2^A = 3$

$f^A(a) = a,\ f^A(b) = a$

$g^A = \{(1, a) \mapsto 1, (1, b) \mapsto$
$1, (2, a) \mapsto 2, (2, b) \mapsto 2, (3, a) \mapsto$
$3, (3, b) \mapsto 3\}$

Signatures and Algebras

Let consider the signature Σ:

sorts s_1, s_2

op $c_1 :\rightarrow s_1,\ c_2 :\rightarrow s_2$

$f : s_1 \rightarrow s_1$

$g : s_2 \times s_1 \rightarrow s_1$

Two example of Σ-algebras:

$|A|_{s_1} = \{a, b\},\ |A|_{s_2} = \{1, 2, 3\}$

$c_1^A = a\ c_2^A = 3$

$f^A(a) = a,\ f^A(b) = a$

$g^A = \{(1, a) \mapsto 1, (1, b) \mapsto$
$1, (2, a) \mapsto 2, (2, b) \mapsto 2, (3, a) \mapsto$
$3, (3, b) \mapsto 3\}$

$|B|_{s_1} = \{\bullet\},\ |B|_{s_2} = \{\heartsuit, \spadesuit\}$

$c_1^B = \bullet,\ c_2^B = \spadesuit$

$f^B(\bullet) = \bullet$

$g^B = \{(\heartsuit, \bullet) \mapsto \heartsuit, (\spadesuit, \bullet) \mapsto \spadesuit\}$

Signatures and Algebras

Exercise E3

Define two distinct algebras for each signature introduced in Exercise E2.

Morphisms and Congruences

Morphism

Let A and B two Σ-algebras. A **Σ-morphism** $h : A \to B$ is an S-sorted function $h : |A| \to |B|$ such that, for each $f : s_1, \ldots, s_n \to s \in \Sigma$, and for any $a_1 \in |A|_{s_1}, \ldots, a_n \in |A|_{s_n}$,

$$h_s(f^A(a_1, \ldots, a_n)) = f^B(h_{s_1}(a_1), \ldots, h_{s_n}(a_n))$$

Morphisms and Congruences

Morphism

Let A and B two Σ-algebras. A **Σ-morphism** $h : A \to B$ is an S-sorted function $h : |A| \to |B|$ such that, for each $f : s_1, \ldots, s_n \to s \in \Sigma$, and for any $a_1 \in |A|_{s_1}, \ldots, a_n \in |A|_{s_n}$,

$$h_s(f^A(a_1, \ldots, a_n)) = f^B(h_{s_1}(a_1), \ldots, h_{s_n}(a_n))$$

$$\begin{array}{ccc} |A|_{s_1} \times \cdots \times |A|_{s_n} & \xrightarrow{h_{s_1} \times \cdots \times h_{sn}} & |B|_{s_1} \times \cdots \times |B|_{s_n} \\ \Big\downarrow{\scriptstyle f^A} & & \Big\downarrow{\scriptstyle f^B} \\ |A|_s & \xrightarrow[h_s]{} & |B|_s \end{array}$$

Morphisms and Congruences

Exercise E4

Let consider the algebras A and B of previous example. Define, if possible, two morphisms $h : A \to B$ and $h' : B \to A$.

Morphisms and Congruences

Definition (Congruence)

Let A be a Σ-algebra and $\equiv \subseteq |A| \times |A|$ an *equivalence relation on* A. The relation $\equiv$ is a congruence if

- for all $a_1, a_1' \in |A|_{s_1}, \ldots, a_n, a_n' \in |A|_{s_1}$, if $a_{s_1} \equiv a'_{s_1}$ and $\cdots$ and $a_{s_n} \equiv a'_{s_n}$, then,

$$f^A(a_1, \ldots, a_n) \equiv f^A(a_1', \ldots, a_n')$$

Morphisms and Congruences

Definition (Quotient Algebra)

Let A be a Σ-algebra and $\equiv \subseteq |A| \times |A|$ a *congruence on* A. The *quotient algebra* A *modulo* $\equiv$ is the Σ-algebra $A/\equiv$ defined by:

- $|A/\equiv| = |A|/\equiv$, and
- for each $f : s_1 \times \cdots \times s_2 \to s \in \Sigma$, $f^{A/\equiv}([a_1]_{\equiv_{s_1}}, \ldots, [a_n]_{\equiv_{s_n}}) = [f^A(a_1, \ldots, a_n)]_{\equiv_s}$ for all $a_1 \in |A|_{s_1}, \ldots, a_n \in |A|_{s_n}$.

Morphisms and Congruences

Exercise E5

Given a S-function $f : X \to Y$, let us consider the S-relation $Ker(f)$, defined for each $s \in S$ as

$$Ker(f_s) = \{(x, y) | x, y \in X_s \text{ and } f_s(x) = f_s(y)\}$$

1. show that $Ker(f)$ is an equivalence relation
2. let us assume an homomorphism $h : A \to B$. Show that $Ker(h)$ is a congruence.
3. Using the morphism $h : A \to B$ of exercise E4, define the algebra $A/Ker(h)$.

Term Algebras

Definition (Σ-terms)

Let Σ be a signature and $X = (X_s)_{s \in S}$ a S-sorted set of variables for Σ. The **set Σ-terms over X** is the smallest S-set $\mathrm{T}(\Sigma, X)$ such that:

- $X_s \subseteq \mathrm{T}(\Sigma, X)_s$;
- $\Omega_{\epsilon,s} \subseteq \mathrm{T}(\Sigma, X)_s$;
- For any $f : s_1 \times \cdots \times s_n \to s \in \Sigma$ and $t_1 \in \mathrm{T}(\Sigma, X)_{s_1}, \ldots, t_n \in \mathrm{T}(\Sigma, X)_{s_n}$, $f(t_1, \ldots, t_n) \in \mathrm{T}(\Sigma, X)_s$;

Term Algebras

Definition (Σ-terms)

Let Σ be a signature and $X = (X_s)_{s \in S}$ a S-sorted set of variables for Σ. The **set Σ-terms over X** is the smallest S-set $\mathrm{T}(\Sigma, X)$ such that:

- $X_s \subseteq \mathrm{T}(\Sigma, X)_s$;
- $\Omega_{\epsilon,s} \subseteq \mathrm{T}(\Sigma, X)_s$;
- For any $f : s_1 \times \cdots \times s_n \to s \in \Sigma$ and $t_1 \in \mathrm{T}(\Sigma, X)_{s_1}, \ldots, t_n \in \mathrm{T}(\Sigma, X)_{s_n}$, $f(t_1, \ldots, t_n) \in \mathrm{T}(\Sigma, X)_s$;

Exercise

Enumerate the terms of each signature of Exercise E2.

Term Algebras

Ground terms

- The set of terms $\mathrm{T}(\Sigma, \emptyset)$ is called the set of **ground terms**.

Term Algebras

Definition (Term Algebra)

If $\mathrm{T}(\Sigma, X)$ is non empty, the term algebra over X is the Σ-algebra $\mathcal{T}(\Sigma, X)$ such that

- $|\mathcal{T}(\Sigma, X)| = \mathrm{T}(\Sigma, X)$
- for any $f : s_1 \times s_n \to s \in \Sigma$ and every $t_1 \in \mathrm{T}(\Sigma, X)_{s_1}, \ldots, t_n \in \mathrm{T}(\Sigma, X)_{s_n}$,

$$f^{\mathcal{T}(\Sigma,X)}(t_1, \ldots, t_n) := f(t_1, \ldots, t_n)$$

Term algebra

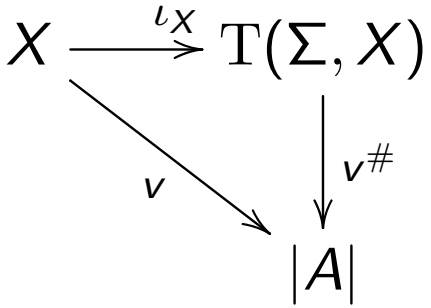
Fact.

For any Σ-algebra A and for any S-function $v : X \to |A|$, there is exactly one Σ-morphism $v^{\#} : \mathrm{T}(\Sigma, X) \to A$ that extends v, i.e., such that $v^{\#}(\iota_X(x)) = v(x)$, where $\iota_X : X \to \mathrm{T}(\Sigma, X)$ maps each variable to its corresponding term.

Term algebra

Fact.

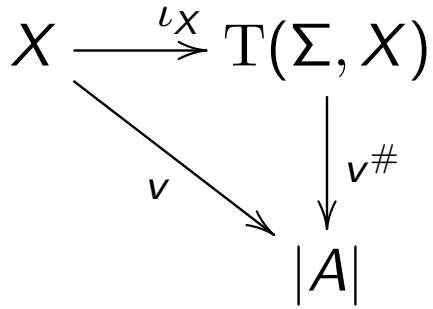
For any Σ-algebra A and for any S-function $v : X \to |A|$, there is exactly one Σ-morphism $v^{\#} : \mathrm{T}(\Sigma, X) \to A$ that extends v, i.e., such that $v^{\#}(\iota_X(x)) = v(x)$, where $\iota_X : X \to \mathrm{T}(\Sigma, X)$ maps each variable to its corresponding term.



Term algebra

Fact.

For any Σ-algebra A and for any S-function $v : X \to |A|$, there is exactly one Σ-morphism $v^{\#} : \mathrm{T}(\Sigma, X) \to A$ that extends v, i.e., such that $v^{\#}(\iota_X(x)) = v(x)$, where $\iota_X : X \to \mathrm{T}(\Sigma, X)$ maps each variable to its corresponding term.



Definition (Term evaluation)

The interpretation of a term $t \in \mathrm{T}(\Sigma, X)_s$ in a Σ-algebra A under the valuation $v : X \to |A|$, denoted by $t^A(v)$, is $v^{\#}(t)$.

Exercise

Exercise E6 - Consider the following signature:

sorts nat

op $0 : nat$

$suc : nat \to nat$

$+ : nat \times nat \to nat$

1. Consider its set of terms;
2. Consider its set of ground terms;
3. Consider 2 models for this signature;

Equational Specification

Equations and satisfaction

Equations

A **Σ-equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Equations and satisfaction

Equations

A **Σ-equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Satisfaction

Let A be a Σ-algebra and $v : X \to |A|$ be a valuation. Then

$$A, v \models t_1 = t_2 \text{ iff } t_1^A(v) = t_2^A(v)$$

Equations and satisfaction

Equations

A **Σ-equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Satisfaction

Let A be a Σ-algebra and $v : X \to |A|$ be a valuation. Then

$$A, v \models t_1 = t_2 \text{ iff } t_1^A(v) = t_2^A(v)$$

Notation

- $A \models t_1 = t_2$ iff for any $v : X \to |A|$, $A \models t_1 = t_2$.

Equations and satisfaction

Equations

A Σ-**equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Satisfaction

Let A be a Σ-algebra and $v : X \to |A|$ be a valuation. Then

$$A, v \models t_1 = t_2 \text{ iff } t_1^A(v) = t_2^A(v)$$

Notation

- $A \models t_1 = t_2$ iff for any $v : X \to |A|$, $A \models t_1 = t_2$.
- $A \models \Phi$ iff for any $t_1 = t_2 \in \Phi$, $A \models t_1 = t_2$.

Equations and satisfaction

Equations

A **Σ-equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Satisfaction

Let A be a Σ-algebra and $v : X \to |A|$ be a valuation. Then

$$A, v \models t_1 = t_2 \text{ iff } t_1^A(v) = t_2^A(v)$$

Notation

- $A \models t_1 = t_2$ iff for any $v : X \to |A|$, $A \models t_1 = t_2$.
- $A \models \Phi$ iff for any $t_1 = t_2 \in \Phi$, $A \models t_1 = t_2$.
- $K \models t_1 = t_2$ iff for any $A \in K$, $A \models t_1 = t_2$.

Equations and satisfaction

Equations

A **Σ-equation** (of sort s) is an expression $t_1 = t_2$, for $t_1, t_2 \in \mathrm{T}(\Sigma, X)_s$.

Satisfaction

Let A be a Σ-algebra and $v : X \to |A|$ be a valuation. Then

$$A, v \models t_1 = t_2 \text{ iff } t_1^A(v) = t_2^A(v)$$

Notation

- $A \models t_1 = t_2$ iff for any $v : X \to |A|$, $A \models t_1 = t_2$.
- $A \models \Phi$ iff for any $t_1 = t_2 \in \Phi$, $A \models t_1 = t_2$.
- $K \models t_1 = t_2$ iff for any $A \in K$, $A \models t_1 = t_2$.
- $\Phi \models t_1 = t_2$ iff, for any Σ-algebra A, if $A \models \Phi$, then $A \models t_1 = t_2$

Specifications

(Flat) Specification

A flat specification consists of a pair $SP = (\Sigma, \Phi)$, where

- Σ is a signature
- Φ is a set of Σ-equations

Specifications

(Flat) Specification

A flat specification consists of a pair $SP = (\Sigma, \Phi)$, where

- Σ is a signature
- Φ is a set of Σ-equations

Specification models

For $SP = (\Sigma, \Phi)$,

$$\mathrm{Mod}[SP] = \{A | A \text{ is } \Sigma\text{-algebra and } A \models \Phi\}$$

Exercise

Exercise E7

Consider the signature of exercise E6.

- Suggest an axiomatisation for natural numbers (at least 5 equations)
- Identify 3 models for your specification and a counter-example (in the same signature)

Some Examples of the base data types of mCRL2

printed from the tool's reference:
Jan Friso Groote, Mohammad Reza Mousavi. Modeling and Analysis of Communicating Systems. MIT Press, 2008.

Example: generic useful auxiliary stuff

map $\approx, \not\approx, <, \leq, \geq, > : S \times S \to \mathbb{B};$
$if : \mathbb{B} \times S \times S \to S;$
var $x, y : S;$
$b : \mathbb{B};$

eqn $x \approx x = true;$
$x \not\approx y = \neg(x \approx y);$
$if(true, x, y) = x;$
$if(false, x, y) = y;$
$if(b, x, x) = x;$
$if(x \approx y, x, y) = y;$
$x < x = false;$
$x \leq x = true;$
$x > y = y < x;$
$x \geq y = y \leq x;$

Example: Booleans

sort $\mathbb{B}$;
cons $true, false : \mathbb{B}$;
map $\neg : \mathbb{B} \to \mathbb{B}$;
$\wedge, \vee, \Rightarrow : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$;

var $b : \mathbb{B}$;
eqn $\neg true = false$;
$\neg false = true$;
$\neg\neg b = b$;
$b \wedge true = b$;
$b \wedge false = false$;
$true \wedge b = b$;
$false \wedge b = false$;
$b \vee true = true$;
$b \vee false = b$;
$true \vee b = true$;
$false \vee b = b$;
$b \Rightarrow true = true$;
$b \Rightarrow false = \neg b$;

$true \Rightarrow b = b$;
$false \Rightarrow b = true$;
$true \approx b = b$;
$false \approx b = \neg b$;
$b \approx true = b$;
$b \approx false = \neg b$;
$false < b = b$;
$true < b = false$;
$b < false = false$;
$b < true = \neg b$;
$false \leq b = true$;
$true \leq b = b$;
$b \leq false = \neg b$;
$b \leq true = true$;

Example: positive naturals

sort $\mathbb{N}^+$;
cons $@c1 : \mathbb{N}^+$;
$@cDub : \mathbb{B} \times \mathbb{N}^+ \to \mathbb{N}^+$;
map $max, min : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$;
$succ, @pospred : \mathbb{N}^+ \to \mathbb{N}^+$;
$+ : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$;
$@addc : \mathbb{B} \times \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$;
$* : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$;
var $b, c : \mathbb{B}$;
$p, q, r : \mathbb{N}^+$;

Example: positive naturals

eqn $@c1 \approx @cDub(b,p) = false;$

$@cDub(b,p) \approx @c1 = false;$

* $succ(p) \approx c1 = false;$

* $@c1 \approx succ(q) = false;$

* $succ(p) \approx @cDub(c,q) = p \approx @pospred(@cDub(c,q));$

* $@cDub(b,p) \approx succ(q) = @pospred(@cDub(b,p)) \approx q;$

$@cDub(b,p) \approx @cDub(c,q) = b \approx c \wedge p \approx q;$

$p < @c1 = false;$

$@c1 < @cDub(b,p) = true;$

$@cDub(b,p) < @cDub(c,q) = if(c \Rightarrow b, p<q, p \leq q);$

* $@c1 < succ(q) = true;$

* $succ(p) < @cDub(c,q) = p < @pospred(@cDub(c,q));$

* $@cDub(b,p) < succ(q) = @cDub(b,p) \leq q;$

Example: lists

sort $List(D)$;
cons $[] : List(D)$;
$\triangleright : D \times List(D) \to List(D)$;
map $in : D \times List(D) \to \mathbb{B}$;
$\# : List(D) \to \mathbb{N}$;
$\triangleleft : List(D) \times D \to List(D)$;
$+\!\!+ : List(D) \times List(D) \to List(D)$;
$. : List(D) \times \mathbb{N} \to D$;
$head : List(D) \to D$;
$tail : List(D) \to List(D)$;
$rhead : List(D) \to D$;
$rtail : List(D) \to List(D)$;
var $d, e : D$;
$s, t : List(D)$;
$p : \mathbb{N}^+$;
eqn $[] \approx d \triangleright s = false$;
$d \triangleright s \approx [] = false$;
$d \triangleright s \approx e \triangleright t = d \approx e \wedge s \approx t$;
$[] < d \triangleright s = true$;
$d \triangleright s < [] = false$;
$d \triangleright s < e \triangleright t = (d \approx e \wedge s < t) \vee d < e$;

$[] \leq d \triangleright s = true$;
$d \triangleright s \leq [] = false$;
$d \triangleright s \leq e \triangleright t = (d \approx e \wedge s \leq t) \vee d < e$;
$in(d, []) = false$;
$in(d, e \triangleright s) = d \approx e \vee in(d, s)$;
$\#[] = @c0$;
$\#d \triangleright s = @cNat(succ(\#s))$;
$[] \triangleleft d = d \triangleright []$;
$(d \triangleright s) \triangleleft e = d \triangleright (s \triangleleft e)$;
$[] +\!\!+ s = s$;
$(d \triangleright s) +\!\!+ t = d \triangleright (s +\!\!+ t)$;
$s +\!\!+ [] = s$;
$(d \triangleright s).@c0 = d$;
$(d \triangleright s).@cNat(p) = s.pred(p)$;
$head(d \triangleright s) = d$;
$tail(d \triangleright s) = s$;
$rhead(d \triangleright []) = d$;
$rhead(d \triangleright (e \triangleright s)) = rhead(e \triangleright s)$;
$rtail(d \triangleright []) = []$;
$rtail(d \triangleright (e \triangleright s)) = d \triangleright rtail(e \triangleright s)$;

Example: sets

sort $Set(D)$;

cons $@set : (D \to \mathbb{B}) \times FSet(D) \to Set(D)$;

map $\{\} : Set(D)$;
$\in : D \times Set(D) \to \mathbb{B}$;
$\overline{\ } : Set(D) \to Set(D)$;
$\cup : Set(D) \times Set(D) \to Set(D)$;
$\cap : Set(D) \times Set(D) \to Set(D)$;
$- : Set(D) \times Set(D) \to Set(D)$;
$@false : D \to \mathbb{B}$;
$@true : D \to \mathbb{B}$;
$@not : (D \to \mathbb{B}) \to D \to \mathbb{B}$;
$@and : (D \to \mathbb{B}) \times (D \to \mathbb{B}) \to D \to \mathbb{B}$;
$@or : (D \to \mathbb{B}) \times (D \to \mathbb{B}) \to D \to \mathbb{B}$;

var $e, d : D$;
$s, t : FSet(D)$;
$f, g : D \to \mathbb{B}$;
$x, y : Set(D)$;

eqn $\{\} = @set(@false, @fset\_empty)$;
$e \in @set(f, s) = f(e) \not\approx @fset\_in(e, s)$;
$@set(f, s) \approx @set(g, t) =$
$\forall c{:}D.(f(c) \approx g(c)) \not\approx @fset\_in(c, @fset\_diff(s, t))$;
$@set(f, s) \cup @set(g, t) =$
$@set(@or(f, g), @fset\_union(f, g, s, t))$;
$@set(f, s) \cap @set(g, t) =$
$@set(@and(f, g), @fset\_inter(f, g, s, t))$;

$x \subset y = x \subseteq y \wedge x \not\approx y$;
$x \subseteq y = (x \cap y) \approx x$;
$@set(f, s) = @set(@not(f), s)$;
$x - y = x \cap \overline{y}$;
$@false(e) = false$;
$@true(e) = true$;
$@false \approx @true = false$;
$@true \approx @false = false$;
$@not(f)(e) = \neg f(e)$;
$@not(@false) = @true$;
$@not(@true) = @false$;
$@and(f, g)(e) = f(e) \wedge g(e)$;
$@and(f, f) = f$;
$@and(f, @false) = @false$;
$@and(@false, f) = @false$;
$@and(f, @true) = f$;
$@and(@true, f) = f$;
$@or(f, g)(e) = f(e) \vee g(e)$;
$@or(f, f) = f$;
$@or(f, @false) = f$;
$@or(@false, f) = f$;
$@or(f, @true) = @true$;
$@or(@true, f) = @true$;

Theories

Definition(Theories)

Let Φ be a set of Σ-equations and K be a class of Σ-algebras. Then, we define:

- $\mathrm{Mod}(\Phi) := \{A | A \text{ is } \Sigma\text{-algebra and } A \models \Phi\}$
- $\mathrm{Th}_\Sigma(K) = \{t_1 = t_2 | \text{for any } A \in K, A \models t_1 = t_2\}$

Galois connections

1. $\Phi \subseteq \Psi$ implies $\mathrm{Mod}(\Phi) \supseteq \mathrm{Mod}(\Psi)$;
2. $K \subseteq K'$ implies $\mathrm{Th}_\Sigma(K) \supseteq \mathrm{Th}_\Sigma(K')$;
3. $\Phi \subseteq \mathrm{Th}_\Sigma(\mathrm{Mod}(\Phi))$ and $K \subseteq \mathrm{Mod}(\mathrm{Th}_\Sigma(K))$.

Equational Calculus

$$\frac{}{\Phi \vdash_\Sigma t = t}\,(\textit{reflexivity}) \qquad \frac{}{\Phi \vdash_\Sigma t_1 = t_2}\, , t_1 = t_2 \in \Phi\ (\text{axioms})$$

$$\frac{\Phi \vdash_\Sigma t_1 = t_2}{\Phi \vdash_\Sigma t_2 = t_1}\,(\textit{symmetry}) \qquad \frac{\Phi \vdash_\Sigma t_1 = t_2 \quad \Phi \vdash_\Sigma t_2 = t_3}{\Phi \vdash_\Sigma t_1 = t_3}\,(\textit{transitivity})$$

$$\frac{\Phi \vdash_\Sigma t_1 = t_1' \quad \cdots \quad \Phi \vdash_\Sigma t_n = t_n'}{\Phi \vdash_\Sigma f(t_1, \cdots, t_n) = f(t_1', \cdots, t_n')}\, , f : s_1 \times \cdots \times s_n \to s \in \Sigma\ (\textit{congruence})$$

$$\frac{\Phi \vdash_\Sigma t_1 = t_2}{\Phi \vdash_\Sigma \sigma(t_1) = \sigma(t_2)}\, , \sigma : X \to \mathrm{T}(\Sigma, X) (\textit{replacement})$$

Exercise

Sort $bool$

Op $true : bool$

$false : bool$

$\neg : bool \to bool$

$\wedge : bool \times bool \to bool$

Ax_{Bool}

- $\neg true = false$
- $\neg false = true$
- $p \wedge true = p$
- $p \wedge false = false$
- $p \wedge \neg p = false$

Using the equational calculus prove that:

1. $Ax_{Bool} \vdash \neg\neg true = true$
2. $Ax_{Bool} \vdash \neg true \wedge \neg false = false$
3. $Ax_{Bool} \vdash (p \wedge \neg true) \wedge false = \neg true$

Equational Calculus

Definition

Let $\equiv_\Phi \subseteq \mathrm{T}(\Sigma, X) \times \mathrm{T}(\Sigma, X)$ the relation defined by

$$\equiv_\Phi = \{(t_1, t_2) | \Phi \vdash t_1 = t_2\}$$

i.e.

$$t_1 \equiv_\Phi t_2 \text{ iff } \Phi \vdash t_1 = t_2$$

Equational Calculus

Lemma

$\equiv_\Phi$ is a congruence in $\mathcal{T}(\Sigma, X)$

Equational Calculus

Lemma

$\Phi \vdash t_1 = t_2$ iff $\mathcal{T}(\Sigma, X)/\equiv_\Phi \models t_1 = t_2$

Equational Calculus

Theorem (Soundness and completeness of equational calculus)

For any set of Σ-equations Φ and for any equation $t_1 = t_2$,

$$\Phi \vdash t_1 = t_2 \text{ iff } \Phi \models t_1 = t_2$$

Equational Calculus

Theorem (Soundness and completeness of equational calculus)

For any set of Σ-equations Φ and for any equation $t_1 = t_2$,

$$\Phi \vdash t_1 = t_2 \text{ iff } \Phi \models t_1 = t_2$$

Hints for the proof:

soundness, i.e., for implication $\Rightarrow$, use induction over the equational calculus rules

completness, i.e., for implication $\Leftarrow$, use the previous Lemma

Initial Models

- The class of Σ-algebras given by loose semantics of a Σ -specification contains too many algebras to be useful in practices. E.g.,
  - if Σ has no constants, the empty algebra is a model of any Σ-specification
  - if Σ has constants, an algebra carried by an S-family of singletons is a model of any Σ-specification

Initial Models

Let $A \in \mathrm{Mod}(\Sigma, \Phi)$. A contains

- **Junk**, if A is not reachable, i.e., there is an $a \in |A|_s$ such that, there is no a ground term $t \in \mathrm{T}(\Sigma, X)$ such that $t^A = a$
- **Confusion**, if A satisfies some ground equation $t_1 = t_2$ such that $\Phi \not\vdash t_1 = t_2$

Initial Models

Let $A \in \mathrm{Mod}(\Sigma, \Phi)$. A contains

- **Junk**, if A is not reachable, i.e., there is an $a \in |A|_s$ such that, there is no a ground term $t \in \mathrm{T}(\Sigma, X)$ such that $t^A = a$
- **Confusion**, if A satisfies some ground equation $t_1 = t_2$ such that $\Phi \nvdash t_1 = t_2$

What should be a good model for a specification?

Exercise

Sort $bool$

Op $true : bool$

$false : bool$

$\neg : bool \to bool$

$\wedge : bool \times bool \to bool$

Ax
- $\neg true = false$
- $\neg false = true$
- $p \wedge true = p$
- $p \wedge q = q \wedge p$
- $p \wedge false = false$
- $p \wedge \neg p = false$

1. Present 3 finite models with 1, 2 and 3 elements.
2. Classify the models with respect to "junk" and "confusion".
3. Build the algebra $\mathcal{T}(\Sigma_{Bool})/\equiv_{\Phi}$, where Φ is the set of equations of the specification.

Initial Models

Lemma

The model $\mathcal{T}(\Sigma)/\equiv_{\Phi}$ has no junk neither confusion

Initial Model

Initial model

An algebra A is initial in a class K if for any algebra $B \in K$, there is an unique morphism $h : A \to B$.

Initial Model

Initial model

An algebra A is initial in a class K if for any algebra $B \in K$, there is an unique morphism $h : A \to B$.

Lemma

Initial models are unique up to isomorphism

Initial Model

Theorem

$\mathcal{T}(\Sigma)/\equiv_{\Phi}$ is an initial model of $\mathrm{Mod}(\langle \Sigma, \Phi \rangle)$