Algebraic Specification of Abstract Data Types

Lecture 1

Constantin Enea

LIAFA, University Paris 7 & CNRS
E-mail: cenea@liafa.jussieu.fr
1 Motivations
2 Membership algebras
3 Homomorphisms, initiality
4 Abstract data types
5 Terms, congruence relations
6 Congruence relations
Data Types

- **Data types** appear everywhere in the context of computing.

  A data type

  represents

  A collection of data, sharing a similar structure, providing similar services

- Programs make use of data types:
  - provided by the programming language: bool, int, float
  - user definable in the programming language: stack, queue, tree
Abstract data types

- abstract away from the implementation of the data type
- (sometimes) even abstract away from a single data type, referring to a whole class of similar data types

An axiomatic formal specification style:
- formulas ("axioms") relate different operations of the data type to each other
The data type stack (not precise syntax or semantics yet)

\[
\begin{align*}
\text{IsEmpty}(\text{EmptyStack}) &= True \\
\text{IsEmpty}(\text{Push}(n, s)) &= False \\
\text{Pop}(\text{Push}(n, s)) &= s \\
\text{Top}(\text{Push}(n, s)) &= n
\end{align*}
\]

Intuitively: \textit{EmptyStack} is a constant, \\
\textit{IsEmpty} is a predicate, \\
\textit{Push}, \textit{Pop}, and \textit{Top} are operations
1 Motivations

2 Membership algebras

3 Homomorphisms, initiality

4 Abstract data types

5 Terms, congruence relations

6 Congruence relations
Motivations

- **Data types** modeled by *membership algebras*. Why?
  - mathematical precision
  - independence of implementation
  - axiomatic definition of operations
  - overloading, error handling, etc.

- **Abstract data types** modeled by *classes of membership algebras closed under isomorphism*. Why?
  - the closure under isomorphism corresponds to the similarity concept
Membership signatures: operations

- a set of kinds $K \rightsquigarrow$ interpreted to sets of elements
- an operation is a $(n+2)$-tuple $(f, k_1, \ldots, k_n, k)$, written $f : k_1 \times \ldots \times k_n \rightarrow k$, with $k_1, \ldots, k_n, k \in K$ and $n \geq 0$
  - $f$ is called a operation (function) symbol
  - $k_1 \times \ldots \times k_n \rightarrow k$ is called arity
  - if $n = 0$, an operation $f : \rightarrow k$ is called a constant of kind $k$
- $\Sigma_{k_1 \ldots k_n,k}$ is the set of operation symbols of arity $k_1 \times \ldots \times k_n \rightarrow k$
- $\Sigma$ is a $K^* \times K$-indexed family of sets:
  $$\Sigma = \{\Sigma_{w,k} | w \in K^*, k \in K\}.$$
K-kind membership signature: \( \Omega = (\Sigma, \pi) \)

- \( \Sigma \) is the set of operation (function) symbols
- \( S \) a set of sorts of kinds associated by \( \pi : S \rightarrow K \)
Examples of Signatures: booleans

- $K = \{\text{Bool}\}$
- $\Omega_{\text{Bool}} = (\Sigma_{\text{Bool}}, \pi)$ with:

  \[
  \Sigma_{\text{Bool}} = \{ \text{True} : \rightarrow \text{Bool}, \text{False} : \rightarrow \text{Bool}, \neg : \text{Bool} \rightarrow \text{Bool}, \wedge : \text{Bool} \times \text{Bool} \rightarrow \text{Bool} \}
  \]

  \[S = \{\text{bool}\} \text{ and } \pi(\text{bool}) = \text{Bool}\]
Examples of Signatures: natural numbers with addition

- \( K = \{Nat\} \)

- \( \Omega_{Nat} = (\Sigma_{Nat}, \pi) \) with:

  \[
  \Sigma_{Nat} = \{ \text{Zero} : \rightarrow Nat, \text{Succ} : Nat \rightarrow Nat, + : Nat \times Nat \rightarrow Nat \}
  \]

  \( S = \{nat^+\} \) and \( \pi(nat^+) = Nat \)
Examples of Signatures: Peano arithmetic

- $K = \{Nat, Bool\}$

- $\Omega_{Peano} = (\Sigma_{Peano}, \pi)$ with:

  $$\Sigma_{Peano} = \Sigma_{bool} \cup \Sigma_{nat}$$

  $$\cup \{ * : Nat \times Nat \rightarrow Nat, \leq : Nat \times Nat \rightarrow Bool \}$$

  $$S = \{nat^+, nat_{even}\} \text{ and } \pi(nat^+) = \pi(nat_{even}) = Nat$$
Examples of Signatures: Stacks of natural numbers

- $K = \{\text{Nat}, \text{Bool}, \text{Stack}\}$

- $\Omega_{\text{NatStack}} = (\Sigma_{\text{NatStack}}, \pi)$ with:

  \[
  \Sigma_{\text{NatStack}} = \Sigma_{\text{nat}} \\
  \cup \{ \text{EmptyStack} : \rightarrow \text{Stack}, \text{Push} : \text{Stack} \times \text{Nat} \rightarrow \text{Stack}, \text{Pop} : \text{Stack} \rightarrow \text{Stack}, \text{Top} : \text{Stack} \rightarrow \text{Nat}, \text{IsEmpty} : \text{Stack} \rightarrow \text{Bool} \}
  \]

- $S = \{\text{stack}^+\}$ and $\pi(\text{stack}^+) = \text{Stack}$
A membership algebra assigns a meaning to a signature by assigning

- to each kind a set of elements
- to each operation symbol a function over these sets
- to each sort a subset of the set associated to its corresponding kind
Membership algebras, formally

- let $\Omega = (\Sigma, \pi)$ be a membership signature
- an $\Omega$-algebra is a triple $A = (A, \Sigma^A, \Pi_A)$, where
  - each kind $k$ is interpreted to a set of elements $A_k$
    $$A = (A_k \mid k \in K)$$
  - each function symbol $f \in \Sigma_{k_1\ldots k_n}$ is interpreted to a function
    $$f^A : A_{k_1} \times \ldots \times A_{k_n} \rightarrow A_k$$
    $$\Sigma^A_{k_1\ldots k_n} = \{ f^A \mid f \in \Sigma_{k_1\ldots k_n} \}$$
    $$\Sigma^A = (\Sigma^A_{w,k} \mid (w, k) \in K^* \times K)$$
  - $\Pi_A$ is a function which assigns to each sort $s \in S$ a subset $A_s \subseteq A_{\pi(s)}$. 
Examples of Algebras: $\Omega_{\text{Bool}}$-algebras

$\mathcal{A} = (A, \Sigma^A_{\text{Bool}}, \Pi_A)$ with:

- $A_{\text{Bool}} = \{\text{true}, \text{false}\}$
- $\text{True}^A = \text{true}$
- $\text{False}^A = \text{false}$
- $\neg^A(\text{true}) = \text{false}$ and $\neg^A(\text{false}) = \text{true}$
- $\land^A(\text{true}, \text{true}) = \text{true}$, $\land^A(\text{true}, \text{false}) = \text{false}$, $\land^A(\text{false}, \text{true}) = \text{false}$, and $\land^A(\text{false}, \text{false}) = \text{false}$
- $\Pi_A(\text{bool}) = A_{\text{Bool}}$
Examples of Algebras: $\Omega_{Nat}$-algebras

$\mathcal{A} = (A, \Sigma^A_{Nat}, \Pi_A)$ with:

- $A_{Nat} = \mathbb{N}$
- $Zero^A = 0$
- $Succ^A(n) = n + 1$, for any $n \in \mathbb{N}$
- $+^A(m, n) = m + n$, for any $m, n \in \mathbb{N}$
- $\Pi_A(nat^+) = \mathbb{N}^+$
Examples of Algebras: $\Omega_{\text{Nat}}$-algebras

$A = (A, \Sigma^A_{\text{Nat}}, \Pi_A)$ with:

- $A_{\text{Nat}} = \{ \text{true}, \text{false} \}$
- $\text{Zero}^A = \text{false}$
- $\text{Succ}^A(\text{false}) = \text{true}$ and $\text{Succ}^A(\text{true}) = \text{false}$
- $+^A(\text{true}, \text{true}) = \text{false}$, $+^A(\text{true}, \text{false}) = \text{true}$, $+^A(\text{false}, \text{true}) = \text{true}$, and $+^A(\text{false}, \text{false}) = \text{false}$
- $\Pi_A(\text{nat}^+) = \{ \text{true} \}$
Examples of Algebras: $\Omega_{\text{Nat}}$-algebras

$A = (A, \Sigma^A_{\text{Nat}}, \Pi_A)$ with:

- $A_{\text{Nat}} = \{\#\}$
- $Zero^A = \#$
- $Succ^A(\#) = \#$
- $+^A(\#, \#) = \#$
- $\Pi_A(nat^+) = \emptyset$
1. Motivations
2. Membership algebras
3. Homomorphisms, initiality
4. Abstract data types
5. Terms, congruence relations
6. Congruence relations
Relate different algebras

Two algebras can be related by mappings which "respect" their functions

"respecting" the functions roughly means:
it should not matter if we apply the mapping
  • before or
  • after
we apply the corresponding function

⇝ Homomorphisms
Homomorphisms

Let $\mathcal{A} = (A, \Sigma^A, \Pi_A)$ and $\mathcal{B} = (B, \Sigma^B, \Pi_B)$ be two $\Omega$-Algebras. $\Omega = (\Sigma, \pi)$. An $\Omega$-homomorphism $h : A \to B$ from $\mathcal{A}$ to $\mathcal{B}$ is a family $(h_k)_{k \in K}$ of functions $h_k : A_k \to B_k$ such that for any $f \in \Sigma$ with $f : k_1 \times \ldots \times k_n \to k$,

$$h_k(f^A(a_1, \ldots, a_n)) = f^B(h_{k_1}(a_1), \ldots, h_{k_n}(a_n))$$

for all $(a_1, \ldots, a_k) \in A_{k_1} \times \ldots \times A_{k_n}$, and

$$h_{\pi(s)}(\Pi_A(s)) \subseteq \Pi_B(s)$$

for all $s \in S$

- if $f : \to k$ is a constant then $h_k(f^A) = f^B$
A bijective homomorphism is called isomorphism

Two \( \Omega \)-algebras \( \mathcal{A} \) and \( \mathcal{B} \) are isomorphic (written \( \mathcal{A} \simeq \mathcal{B} \)) if and only if there exists an isomorphism from \( \mathcal{A} \) to \( \mathcal{B} \).

Intuition: \( \mathcal{A} \) and \( \mathcal{B} \) are "identical up to renaming"
Example: Relate different boolean algebras

Remind:

- $K = \{Bool\}$
- $\Omega_{Bool} = (\Sigma_{Bool}, \pi)$ with:

$$
\Sigma_{bool} = \{ True : \rightarrow Bool, \\
False : \rightarrow Bool, \\
\neg : Bool \rightarrow Bool, \\
\wedge : Bool \times Bool \rightarrow Bool \}
$$

$S = \{bool\}$ and $\pi(bool) = Bool$
A classical boolean algebra

\[ A = (A, \Sigma^A_{\text{Bool}}, \Pi_A) \text{ with:} \]

- \( A_{\text{Bool}} = \{ \text{true}, \text{false} \} \)
- \( \text{True}^A = \text{true} \)
- \( \text{False}^A = \text{false} \)
- \( \neg^A(\text{true}) = \text{false} \) and \( \neg^A(\text{false}) = \text{true} \)
- \( \land^A(\text{true}, \text{true}) = \text{true}, \land^A(\text{true}, \text{false}) = \text{false}, \land^A(\text{false}, \text{true}) = \text{false}, \) and \( \land^A(\text{false}, \text{false}) = \text{false} \)
- \( \Pi_A(\text{bool}) = A_{\text{Bool}} \)
Another boolean algebra

\[ \mathcal{B} = (B, \Sigma^B_{Bool}, \Pi_B) \] with:

- \( B_{Bool} = \{\#\} \)
- \( \text{True}^B = \# \)
- \( \text{False}^B = \# \)
- \( \neg^B(\#) = \# \)
- \( \land^B(\#, \#) = \# \)
- \( \Pi_B(\text{bool}) = B_{Bool} \)
Another boolean algebra

\[ C = (C, \Sigma_{Bool}^C, \Pi^C) \] with:

- \( C_{Bool} = \{0, 1\} \)
- \( \text{True}^C = 1 \)
- \( \text{False}^C = 0 \)
- \( \neg^C(1) = 0 \) and \( \neg^C(0) = 1 \)
- \( \land^C(m, n) = m \ast n \)
- \( \Pi_C(bool) = C_{Bool} \)
Another boolean algebra

\[ D = (D, \Sigma^D_{\text{Bool}}, \Pi_D) \]

- \( D_{\text{Bool}} = \mathbb{N} \)
- \( \text{True}^D = 1 \)
- \( \text{False}^D = 0 \)
- \( \neg^D(n) = \begin{cases} n + 1 & \text{if } n \text{ is even} \\ n - 1 & \text{otherwise} \end{cases} \)
- \( \wedge^D(m, n) = m \ast n \)
- \( \Pi_D(\text{bool}) = D_{\text{Bool}} \)
Another boolean algebra

\[ \mathcal{E} = (E, \Sigma^E_{\text{Bool}}, \Pi_E) \] with:

- \( E_{\text{Bool}} = \mathbb{N} \)
- \( \text{True}^E = 1 \)
- \( \text{False}^E = 0 \)
- \( \neg^E(n) = n + 1 \)
- \( \land^E(m, n) = m + n \)
- \( \Pi_E(\text{bool}) = E_{\text{Bool}} \)
Example: facts

Exercises:
- The function $h : A \to B$ with $h(true) = h(false) = \#$ is a homomorphism.
- The function $g : A \to C$ with $g(true) = 1$ and $g(false) = 0$ is an isomorphism.
- There exists a homomorphism $k : A \to D$.
- There exists a homomorphism $l : D \to A$.
- (But) $A$ and $D$ are not isomorphic.
- There exists no homomorphism from $A$ to $E$, nor from $E$ to $A$. 
Properties of homomorphisms

Theorem

The composition of two $\Omega$-homomorphisms yields a $\Omega$-homomorphism

Theorem

Let $h : A \rightarrow B$ be an $\Omega$-isomomorphism from $A$ to $B$. Then $h^{-1} = (h_k^{-1})_{k \in K}$ is an $\Omega$-isomomorphism from $B$ to $A$.

Theorem

The relation $\simeq$ on Alg($\Omega$) is an equivalence relation.
Let $\mathcal{C} \subseteq \text{Alg}(\Omega)$ be a class of $\Omega$-algebras.

An algebra $\mathcal{A} \in \mathcal{C}$ is initial in the class $\mathcal{C}$ \iff

for each $\mathcal{B} \in \mathcal{C}$ there exists exactly one homomorphism from $\mathcal{A}$ to $\mathcal{B}$.

**Theorem**

Let $\mathcal{C} \subseteq \text{Alg}(\Omega)$ be a class of $\Omega$-algebras. Assume $\mathcal{A}$ is initial in $\mathcal{C}$. Then:

$\mathcal{B}$ is initial in $\mathcal{C}$ \iff $\mathcal{A} \simeq \mathcal{B}$
1. $A$ is initial in the class $\{A, B\}$

2. How many initial algebras does the set $\{A, B, C, D\}$ have?
1 Motivations

2 Membership algebras

3 Homomorphisms, initiality

4 Abstract data types

5 Terms, congruence relations

6 Congruence relations
Observation: Isomorphic algebras are ”similar” in the sense that they only differ in how the carrier elements are ”called”. The ”structure” of isomorphic algebras is the same.

Look ahead: Moreover, isomorphic algebras cannot be distinguished by logic.

A class $C \subseteq \text{Alg}(\Omega)$ is closed under isomorphism, if the following condition is satisfied: $A \in C$ and $A \cong B$ implies $B \in C$. 
Abstract Data Types

An abstract data type for a signature $\Omega$ is any class of $\Omega$-algebras that is closed under isomorphism.

An abstract data type is called monomorphic, if its algebras are all isomorphic to each other. Otherwise, it is called polymorphic.

Informally:

- A monomorphic abstract data type stands for a ”single” data type.
- A polymorphic abstract data type stands for ”several” data types, which typically correspond to an incomplete specification.

Example

- $\{ \mathcal{X} \in \text{Alg}(\Omega_{\text{Bool}}) \mid \mathcal{X} \simeq A \}$ is an abstract data type which is monomorphic and contains $\mathcal{C}$
- $\{ \mathcal{X} \in \text{Alg}(\Omega_{\text{Bool}}) \mid \mathcal{X} \simeq A \text{ or } \mathcal{X} \simeq B \}$ is a polymorphic abstract data type
Motivations

Membership algebras

Homomorphisms, initiality

Abstract data types

Terms, congruence relations

Congruence relations
With each signature $\Omega = (\Sigma, \pi)$ is associated a family $X = (X_k)_{k \in K}$ of disjoint infinite sets. The elements of $X_k$ are called variables of kind $k$.

It is assumed that the variables of $X$ and the operation names of $\Sigma$ are different.

$T_\Omega(X) = (T_{\Omega,k}(X))_{k \in K}$ is the family of minimal sets satisfying:

- $X_k \subseteq T_{\Omega,k}(X)$
- if $f : \rightarrow k \in \Sigma$, then $f \in T_{\Omega,k}(X)$
- if $f : k_1 \times \ldots \times k_n \rightarrow k \in \Sigma$ (with $n \geq 1$), and $t_i \in T_{\Omega,k_i}(X)$, for any $1 \leq i \leq n$, then $f(t_1, \ldots, t_n) \in T_{\Omega,k}(X)$
Occurring variables, ground terms

Var(t) is the set of variables occurring in t

If Var(t) = ∅, then t is a ground term

\( T_{\Omega,k} = T_{\Omega,k}(\emptyset) \) denotes the set of all ground terms of kind \( k \)

\( T_{\Omega} = ( T_{\Omega,k} )_{k \in K} \) is called the set of ground terms of \( \Omega \).
Reconsider the signature $\Omega_{\text{Peano}}$ and let $X_{\text{Nat}} = \{m, n\}$, $X_{\text{Bool}} = \{b, c\}$.

Terms of kind $\text{Bool}$ are:

- $\text{False}$
- $c$
- $\land(\land(\text{True}, b), \text{False})$
- $\leq (0, +(m, \text{Succ}(n)))$
Assignments

Let $\Omega = (\Sigma, \pi)$ be a signature and $X$ a set of variables for $\Omega$, and $\mathcal{A} = (A, \Sigma^A, \Pi_A) \in \text{Alg}(\Omega)$.

A family $\gamma = (\gamma_k)_{k \in K}$ with $\gamma_k : X_k \rightarrow A_k$ is called an assignment of $X$ into $\mathcal{A}$.

One writes $\gamma : X \rightarrow A$

If $a \in A_k$ and $x \in X_k$, then $\gamma[x/a]$ is the assignment obtained from $\gamma$ by replacing the value $\gamma_k(x)$ by $a$. 
Let $\Omega = (\Sigma, \pi)$ be a signature and $X$ a set of variables for $\Omega$, and $A = (A, \Sigma^A, \Pi_A) \in \text{Alg}(\Omega)$.

We extend the function $\gamma$ to terms as follows:

- if $t = x$ with $x \in X_k$ then $\gamma(t) = \gamma(x)$;
- if $t = f$ with $f : \rightarrow k \in \Sigma$ then $\gamma(t) = f^A$;
- if $t = f(t_1, \ldots, t_n)$ then $\gamma(t) = f^A(\gamma(t_1), \ldots, \gamma(t_n))$.

The value of ground terms is the same for any assignment $\gamma$. 
Let $\mathcal{A}$ and $\mathcal{B}$ be two $\Omega$-algebras and let $h : A \to B$ be an $\Omega$-homomorphism. Then:

- $h(\gamma(t)) = \gamma'(t)$, for each assignment $\gamma : X \to A$ and $\gamma' : X \to B$, and ground term $t$;
Term algebra

Let $\Omega = (\Sigma, \pi)$ be a signature, $A = (A, \Sigma^A, \Pi_A) \in \text{Alg}(\Omega)$, and $\rho = (\rho_k)_{k \in K}$ a congruence relation on $A$

$T_\Omega(X) = (T_\Omega(X), \Sigma^{T_\Omega(X)}, \Pi_{T_\Omega(X)})$ is the term algebra:

- $f^{T_\Omega(X)} = f$ if $f : \rightarrow k$ is a constant
- $f^{T_\Omega(X)} = f(t_1, \ldots, t_n)$, for any $f : k_1 \times \ldots k_n \rightarrow k$, and $t_i \in T_{\Omega,k_i}(X)$, for any $1 \leq i \leq n$
- $\Pi_{T_\Omega(X)}(s) \subseteq T_{\Omega,\pi(s)}(X)$, for any $s \in S$.

$T_\Omega = T_\Omega(\emptyset)$ is the ground term algebra
Motivations

Membership algebras

Homomorphisms, initiality

Abstract data types

Terms, congruence relations

Congruence relations
Congruence relations

Let $\Omega = (\Sigma, \pi)$ be a signature and $\mathcal{A} = (A, \Sigma^A, \Pi_A) \in \text{Alg}(\Omega)$.

A congruence relation on $\mathcal{A}$ is a family $\rho = (\rho_k)_{k \in K}$ of equivalence relations $\rho_k$ on $A_k$ such that:

for any $f : k_1 \times \ldots \times k_n \rightarrow k$ in $\Sigma$ with $n \geq 1$:

for any $a_i, a'_i \in A_{k_i}$ with $a_i \rho_{k_i} a'_i$, $1 \leq i \leq n$,

$$f(a_1, \ldots, a_n) \rho_k f(a'_1, \ldots, a'_n)$$

Informally: Equivalent arguments lead to equivalent function values
Consider the $\Omega_{Bool}$-algebra $\mathcal{D}$:

Define: $m \equiv_{\mathcal{D}} n$ iff $n + m$ is even

Then $\rho = (\rho_{\mathcal{D}})$ is a congruence relation on $\mathcal{D}$
Quotient Algebras

Let $\Omega = (\Sigma, \pi)$ be a signature, $A = (A, \Sigma^A, \Pi_A) \in \text{Alg}(\Omega)$, and $\rho = (\rho_k)_{k \in K}$ a congruence relation on $A$.

The quotient algebra (or quotient) of $A$ by $\rho$ is the $\Omega$-algebra $A/\rho = (B, \Sigma^B, \Pi_B)$, defined by:

$$B_k = \{[a]_{\rho_k} \mid a \in A_k\}, \text{ for all } k \in K$$

$$f^B([a_1]_{\rho_{k_1}}, \ldots, [a_n]_{\rho_{k_n}}) = [f(a_1, \ldots, a_n)]_{\rho_k}, \text{ for all } f : k_1 \times \ldots \times k_n \rightarrow k$$

$$\Pi_B(s) = \{[a]_{\rho_{\pi(s)}} \mid [a]_{\rho_{\pi(s)}} \cap \Pi_A(s) \neq \emptyset\}$$

The above definition is consistent.