Semantyka i weryfikacja programów

Bartosz Klin

(slajdy Andrzeja Tarleckiego)

Instytut Informatyki
Wydział Matematyki, Informatyki i Mechaniki
Uniwersytet Warszawski

http://www.mimuw.edu.pl/~klin pok. 5680
klin@mimuw.edu.pl

Strona tego wykładu:

http://www.mimuw.edu.pl/~klin/sem18-19.html

Program Semantics & Verification

Bartosz Klin

(slides courtesy of Andrzej Tarlecki)

Institute of Informatics
Faculty of Mathematics, Informatics and Mechanics
University of Warsaw

`http://www.mimuw.edu.pl/~klin` office: 5680

`klin@mimuw.edu.pl`

This course: `http://www.mimuw.edu.pl/~klin/sem18-19.html`

Universal algebra

Basics of universal algebra:

- signatures and algebras
- homomorphisms, subalgebras, congruences
- equations and varieties
- Birkhoff's theorem

Plus some hints on applications in

foundations of software semantics, verification, specification, development. . .

Signatures

Algebraic signature: *a set of operation names, classified by arities*:

$$\boxed{\Sigma = \langle \Sigma_n \rangle_{n \in \mathbb{N}}}$$

Alternatively:

$$\boxed{\Sigma = (|\Sigma|, arity)}$$

with *operation names* $|\Sigma|$ and *arity function* $arity\colon |\Sigma| \to \mathbb{N}$.

- We write $f \in \Sigma_n$ if $arity(f) = n$,
- and $f \in \Sigma$ if $f \in \Sigma_n$ for some n.

Compare the two notions

Names in Σ_0 are called *constants*.

Fix a signature Σ for a while.

Algebras

- Σ-*algebra*:

$$\boxed{A = (|A|, \langle f^A \rangle_{f \in \Sigma})}$$

  - *carrier set* $|A|$
  - *operations*: $f^A \colon |A|^n \to |A|$, for $f \in \Sigma_n$

- the class of all Σ-algebras:

$$\boxed{\mathbf{Alg}(\Sigma)}$$

Can $\mathbf{Alg}(\Sigma)$ be empty? Finite?

Can $A \in \mathbf{Alg}(\Sigma)$ have an empty carrier?

Multi-sorted setting

Algebraic signature:

$$\boxed{\Sigma = (S, \Omega)}$$

- *sort names*: S
- *operation names, classified by arities and result sorts*: $\Omega = \langle \Omega_{w,s} \rangle_{w \in S^*, s \in S}$

  $f : s_1 \times \ldots \times s_n \to s$ stands for $s_1, \ldots, s_n, s \in S$ and $f \in \Omega_{s_1 \ldots s_n, s}$
- Σ-*algebra*:

$$\boxed{A = (|A|, \langle f_A \rangle_{f \in \Omega})}$$

  - *carrier sets:* $|A| = \langle |A|_s \rangle_{s \in S}$
  - *operations*: $f_A : |A|_{s_1} \times \ldots \times |A|_{s_n} \to |A|_s$, for $f : s_1 \times \ldots \times s_n \to s$

...and so on...

The algebra of TINY

Its *signature* Σ (syntax):

$$
\begin{array}{ll}
\textbf{sorts} & Int, Bool; \\
\textbf{opns} & 0, 1 \colon Int; \\
& plus, times, minus \colon Int \times Int \to Int; \\
& false, true \colon Bool; \\
& lteq \colon Int \times Int \to Bool; \\
& not \colon Bool \to Bool; \\
& and \colon Bool \times Bool \to Bool;
\end{array}
$$

and Σ-*algebra* $\mathcal{A}$ (semantics):

$$
\begin{array}{ll}
\textbf{carriers} & \mathcal{A}_{Int} = \mathbf{Int}, \mathcal{A}_{Bool} = \mathbf{Bool} \\
\textbf{operations} & 0_{\mathcal{A}} = 0, 1_{\mathcal{A}} = 1 \\
& plus_{\mathcal{A}}(n, m) = n + m, times_{\mathcal{A}}(n, m) = n * m \\
& minus_{\mathcal{A}}(n, m) = n - m \\
& false_{\mathcal{A}} = \mathbf{ff}, true_{\mathcal{A}} = \mathbf{tt} \\
& lteq_{\mathcal{A}}(n, m) = \mathbf{tt} \textbf{ if } n \leq m \textbf{ else } \mathbf{ff} \\
& not_{\mathcal{A}}(b) = \mathbf{tt} \textbf{ if } b = \mathbf{ff} \textbf{ else } \mathbf{ff} \\
& and_{\mathcal{A}}(b, b') = \mathbf{tt} \textbf{ if } b = b' = \mathbf{tt} \textbf{ else } \mathbf{ff}
\end{array}
$$

Subalgebras

- for $A \in \mathbf{Alg}(\Sigma)$, a Σ-*subalgebra* $A_{sub} \subseteq A$ is given by subset $|A_{sub}| \subseteq |A|$ closed under the operations:
  – for $f \in \Sigma_n$ and $a_1, \ldots, a_n \in |A_{sub}|$, we require $f^A(a_1, \ldots, a_n) \in |A_{sub}|$ then define
    $f^{A_{sub}}(a_1, \ldots, a_n) = f^A(a_1, \ldots, a_n)$
- for $A \in \mathbf{Alg}(\Sigma)$ and $X \subseteq |A|$, the *subalgebra of* A *generated by* X, $\langle A \rangle_X$, is the least subalgebra of A that contains X.
- $A \in \mathbf{Alg}(\Sigma)$ is *reachable* if $\langle A \rangle_\emptyset$ coincides with A.

Fact: *For any* $A \in \mathbf{Alg}(\Sigma)$ *and* $X \subseteq |A|$, $\langle A \rangle_X$ *exists.*

Proof (idea):

- generate the generated subalgebra from X by closing it under operations in A; or
- the intersection of any family of subalgebras of A is a subalgebra of A.

Homomorphisms

- for $A, B \in \mathbf{Alg}(\Sigma)$, a Σ-*homomorphism* $h\colon A \to B$ is a function $h\colon |A| \to |B|$ that preserves the operations:
  - for $f \in \Sigma_n$ and $a_1, \ldots, a_n \in |A|$,
$$h(f^A(a_1, \ldots, a_n)) = f^B(h(a_1), \ldots, h(a_n))$$

Fact: *Given a homomorphism $h\colon A \to B$ and subalgebras A_{sub} of A and B_{sub} of B, the image of A_{sub} under h, $h(A_{sub})$, is a subalgebra of B, and the coimage of B_{sub} under h, $h^{-1}(B_{sub})$, is a subalgebra of A.*

Fact: *Given a homomorphism $h\colon A \to B$ and $X \subseteq |A|$, $h(\langle A \rangle_X) = \langle B \rangle_{h(X)}$.*

Fact: *Identity function on the carrier of $A \in \mathbf{Alg}(\Sigma)$ is a homomorphism $id_A\colon A \to A$. Composition of homomorphisms $h\colon A \to B$ and $g\colon B \to C$ is a homomorphism $h;g\colon A \to C$.*

Isomorphisms

- for $A, B \in \mathbf{Alg}(\Sigma)$, a Σ-*isomorphism* is any Σ-homomorphism $i\colon A \to B$ that has an *inverse*, i.e., a Σ-homomorphism $i^{-1}\colon B \to A$ such that $i;i^{-1} = id_A$ and $i^{-1};i = id_B$.
- Σ-algebras are *isomorphic* if there exists an isomorphism between them.

Fact: *A Σ-homomorphism is a Σ-isomorphism iff it is bijective ("1-1" and "onto").*

Fact: *Identities are isomorphisms, and any composition of isomorphisms is an isomorphism.*

Congruences

- for $A \in \mathbf{Alg}(\Sigma)$, a Σ-*congruence on* A is an equivalence $\equiv\ \subseteq |A| \times |A|$ that is closed under the operations:
  - for $f \in \Sigma_n$ and $a_1, b_1 \ldots, a_n, b_n \in |A|$,
    if $a_1 \equiv b_1, \ldots, a_n \equiv b_n$ then $f^A(a_1, \ldots, a_n) \equiv f^A(a'_1, \ldots, a'_n)$.

Fact: *For any relation* $R \subseteq |A| \times |A|$ *on the carrier of a* Σ*-algebra* A*, there exists the least congruence on* A *that contains* R*.*

Fact: *For any* Σ*-homomorphism* $h\colon A \to B$*, the kernel of* h*,* $K(h) \subseteq |A| \times |A|$*, where* $a\ K(h)\ a'$ *iff* $h(a) = h(a')$*, is a* Σ*-congruence on* A*.*

Quotients

- for $A \in \mathbf{Alg}(\Sigma)$ and Σ-congruence $\equiv \subseteq |A| \times |A|$ on A, the *quotient algebra* $A/\!\equiv$ is built in the natural way on the equivalence classes of $\equiv$:
  - for $|A/\!\equiv| = \{[a]_\equiv \mid a \in |A|\}$, with $[a]_\equiv = \{a' \in |A| \mid a \equiv a'\}$
  - for $f \in \Sigma_n$ and $a_1, \ldots, a_n \in |A|$,

$$f_{A/\equiv}([a_1]_\equiv, \ldots, [a_n]_\equiv) = [f_A(a_1, \ldots, a_n)]_\equiv$$

Fact: *The above is well-defined; moreover, the natural map that assigns to every element its equivalence class is a Σ-homomorphisms $[\_]_\equiv \colon A \to A/\!\equiv$.*

Fact: *Given two Σ-congruences $\equiv$ and $\equiv'$ on A, $\equiv \subseteq \equiv'$ iff there exists a Σ-homomorphism $h \colon A/\!\equiv \to A/\!\equiv'$ such that $[\_]_\equiv;h = [\_]_{\equiv'}$.*

Fact: *For any Σ-homomorphism $h \colon A \to B$, $A/K(h)$ is isomorphic with $h(A)$.*

Products

- for $A_i \in \mathbf{Alg}(\Sigma)$, $i \in \mathcal{I}$, the *product of* $\langle A_i \rangle_{i \in \mathcal{I}}$, $\prod_{i \in \mathcal{I}} A_i$ is built in the natural way on the Cartesian product of the carriers of A_i, $i \in \mathcal{I}$:
  - $|\prod_{i \in \mathcal{I}} A_i| = \prod_{i \in \mathcal{I}} |A_i|$
  - for $f \in \Sigma_n$ and $a_1 \in |\prod_{i \in \mathcal{I}} A_i|, \ldots, a_n \in |\prod_{i \in \mathcal{I}} A_i|$, for $i \in \mathcal{I}$,
    $f_{\prod_{i \in \mathcal{I}} A_i}(a_1, \ldots, a_n)(i) = f_{A_i}(a_1(i), \ldots, a_n(i))$

Fact: *For any family $\langle A_i \rangle_{i \in \mathcal{I}}$ of Σ-algebras, projections $\pi_i(a) = a(i)$, where $i \in \mathcal{I}$ and $a \in \prod_{i \in \mathcal{I}} |A_i|$, are Σ-homomorphisms $\pi_i \colon \prod_{i \in \mathcal{I}} A_i \to A_i$.*

> Define the product of the empty family of Σ-algebras.
> When the projection π_i is an isomorphism?

Terms

Consider a set X of variables.

- *terms* $t \in |T_\Sigma(X)|$ are built using variables X, constants and operations from Σ in the usual way: $|T_\Sigma(X)|$ is the least set such that
  - $X \subseteq |T_\Sigma(X)|$
  - for $f \in \Sigma_n$ and $t_1, \ldots, t_n \in |T_\Sigma(X)|$, $f(t_1, \ldots, t_n) \in |T_\Sigma(X)|$
- for any Σ-algebra A and valuation $v \colon X \to |A|$, *the value* $t^A[v]$ *of a term* $t \in |T_\Sigma(X)|$ *in* A *under* v is determined inductively:
  - $x^A[v] = v(x)$, for $x \in X$
  - $(f(t_1, \ldots, t_n))^A[v] = f_A((t_1)^A[v], \ldots, (t_n)^A[v])$, for $f \in \Sigma_n$ and $t_1, \ldots, t_n \in |T_\Sigma(X)|$

Above and in the following: assuming unambiguous "parsing" of terms!

Term algebras

Consider a set X of variables.

- The *term algebra* $T_\Sigma(X)$ has the set of terms as the carrier and operations defined "syntactically":
  - for $f \in \Sigma_n$ and $t_1, \ldots, t_n \in |T_\Sigma(X)|$, $f^{T_\Sigma(X)}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$.

Fact: *For any set X of variables, Σ-algebra A and valuation $v \colon X \to |A|$, there is a unique Σ-homomorphism $v^\# \colon T_\Sigma(X) \to A$ that extends v. Moreover, for* $t \in |T_\Sigma(X)|$, $v^\#(t) = t^A[v]$.

$$\mathbf{Set}^S \qquad \begin{tikzcd} X \arrow[r, "id_{X \hookrightarrow |T_\Sigma(X)|}"] \arrow[dr, "v"'] & {|T_\Sigma(X)|} \arrow[d, "|v^\#|"] \\ & {|A|} \end{tikzcd} \qquad \begin{tikzcd} T_\Sigma(X) \arrow[d, "\exists!"', "v^\#"] \\ A \end{tikzcd} \qquad \mathbf{Alg}(\Sigma)$$

Equations

- *Equation*:

$$\boxed{\forall X.t = t'}$$

  where:

  – X is a set of variables, and

  – $t, t' \in |T_\Sigma(X)|$ are terms.

- *Satisfaction relation*: Σ-algebra A *satisfies* $\forall X.t = t'$

$$\boxed{A \models \forall X.t = t'}$$

  when for all $v\colon X \to |A|$, $t^A[v] = t'^A[v]$.

Semantic entailment

$$\boxed{\Phi \models_\Sigma \varphi}$$

> Σ-*equation* φ *is a semantic consequence of a set of* Σ-*equations* Φ *if* φ *holds in every* Σ-*algebra that satisfies* Φ.

BTW:

- *Models* of a set of equations: $Mod(\Phi) = \{A \in \mathbf{Alg}(\Sigma) \mid A \models \Phi\}$
- *Theory* of a class of algebras: $Th(\mathcal{C}) = \{\varphi \mid \mathcal{C} \models \varphi\}$
- $\Phi \models \varphi \iff \varphi \in Th(Mod(\Phi))$
- Mod and Th form a *Galois connection*

Equational calculus

$$\frac{}{\forall X.t = t} \qquad \frac{\forall X.t = t'}{\forall X.t' = t} \qquad \frac{\forall X.t = t' \quad \forall X.t' = t''}{\forall X.t = t''}$$

$$\frac{\forall X.t_1 = t_1' \quad \ldots \quad \forall X.t_n = t_n'}{\forall X.f(t_1 \ldots t_n) = f(t_1' \ldots t_n')} \qquad \frac{\forall X.t = t'}{\forall Y.t[\theta] = t'[\theta]} \text{for } \theta\colon X \to |T_\Sigma(Y)|$$

Mind the variables!

$a = b$ does *not* follow from $a = f(x)$ and $f(x) = b$, unless...

Proof-theoretic entailment

$$\boxed{\Phi \vdash_\Sigma \varphi}$$

Σ-equation φ is a proof-theoretic consequence of a set of Σ-equations Φ if φ can be derived from Φ by the rules.

How to justify this?

Semantics!

Soundness & completeness

Fact: *The equational calculus is sound and complete:*

$$\boxed{\Phi \models \varphi \iff \Phi \vdash \varphi}$$

- soundness: "all that can be proved, is true" ($\Phi \models \varphi \Longleftarrow \Phi \vdash \varphi$)
- completeness: "all that is true, can be proved" ($\Phi \models \varphi \Longrightarrow \Phi \vdash \varphi$)

Proof (idea):

- soundness: easy!
- completeness: not so easy!

One motivation

Software systems (data types, modules, programs, databases. . .):

sets of data with operations on them

- Disregarding: code, efficiency, robustness, reliability, . . .
- Focusing on: CORRECTNESS

Universal algebra
from rough analogy:

module interface $\leadsto$ signature

module $\leadsto$ algebra

module specification $\leadsto$ class of algebras

Example

$$
\begin{aligned}
\textbf{spec } \textsc{Stack} = \textbf{sorts } & Elem, Stack \\
\textbf{opns } & empty : Stack; \\
& \quad push : Elem \times Stack \to Stack; \\
& \quad pop : Stack \to Stack; \\
& \quad top : Stack \to Elem \\
\textbf{axioms } & \forall s{:}Stack.\forall e{:}Elem.top(push(e,s)) = e; \\
& \quad \forall s{:}Stack.\forall e{:}Elem.pop(push(e,s)) = s; \\
& \quad \ldots
\end{aligned}
$$

Problem:

There are models $M \in Mod(\textsc{Stack})$ such that $M \models empty = push(empty, e)$, or even:

$$M \models \forall s, t{:}Stack.s = t$$

Equational specifications

$$\boxed{\langle \Sigma, \Phi \rangle}$$

- signature Σ, to determine the static module interface
- axioms (Σ-equations), to determine required module properties

Birkhoff's HSP Theorem:

Fact: *A class of Σ-algebras is equationally definable iff it is closed under subalgebras, products and homomorphic images.*

Solution: allow more powerful specification formalisms

Wrapping up

Message to take home

- Programming languages have a lot in common
- Some basic semantic notions that keep popping up:
  - state vs. environment
  - static vs. dynamic scope
  - parameter passing modes
  - Continuations!
- We may try to prove that programs are correct
  - Very little can be done!
  - But so much it at stake that making even tiny progress is very useful

Your (near) future

- Jezyki i Paradygmaty Programowania (JiPP):
  - very cool programming languages totally unlike TINY
  - writing an interpreter (in Haskell) for a language of your own design
  - Hint: just recall your denotational semantics!
- Metody Realizacji Jezykow Programowania (MRJP):
  - writing a full-fledged compiler
  - lexing, parsing, code generation, the works
- But first...
  - Good luck at the exam!