Freely Generated Exercise :-)

Exercise 6.4.
Reconsider the specification you developed for exercise 1.1. Are the 'intended' algebras (freely) generated? If yes: by which subset of the operations?

Freely Generatedness and Unique Homomorphisms

Theorem 6.7.
Let A and B be two Σ-Algebras for some $\Sigma = (S, \Omega)$.
If A is freely generated, then there exists a unique homomorphism from A to B. $\star$

Term Algebras

A term algebra is a particular freely generated algebra:

The *term algebra* for a signature $\Sigma = (S, \Omega)$ is the Σ-Algebra $T(\Sigma)$, defined by:

$$\begin{array}{lcl} T(\Sigma)(s) & = & T_{\Sigma,s}, \text{ for any } s \in S, \\ T(\Sigma)(\omega) & = & n, \text{ for any } \omega = (n :\rightarrow s) \in \Omega, \\ T(\Sigma)(\omega)(t_1, \ldots, t_k) & = & n(t_1, \ldots, t_k), \text{ for any } \omega = (n : s_1 \times \ldots \times s_k \rightarrow s) \in \Omega, \\ & & k \geq 1, t_i \in T(\Sigma)(s_i), 1 \leq k \end{array}$$

distinguish between T_Σ and $T(\Sigma)$

Term Algebras

Fact 6.8.
Let Σ be a signature. Then $T(\Sigma)(t) = t$ for any ground term $t \in T_\Sigma$. $\star$

Fact 6.9.
A term algebra is freely generated. $\star$

Exercise 6.5.
Prove these facts.

Term Algebra Example

the term algebra $T(\textbf{NAT})$ is defined by:

$$
\begin{array}{lcl}
T(\textbf{NAT})(s) & = & \{0, Succ(0), Succ(Succ(0)), \ldots\} \\
T(\textbf{NAT})(0) & = & 0 \\
T(\textbf{NAT})(Succ)(t) & = & Succ(t), \text{ for any } t \in T(\textbf{NAT})(nat)
\end{array}
$$

Term Evaluations 'as' Homomorphisms

Viewing the terms itself as 'an algebra', we can view the 'value function' $A(t)$ on terms (in any algebra A) as a homomorphism from the term algebra to A.

This is the *only* homomorphism from $T(\Sigma)$ to A, due to fact 6.9 and theorem 6.7.

The homomorphism $h : T(\Sigma) \longrightarrow A$ is called *evaluation homomorphism of A* w.r.t. $T(\Sigma)$, defined by:

$$h_s(t) = A(t), \text{ for any ground term } t \in T_{\Sigma,s},\ s \in S$$

Generated Algebra vs. Term Algebra

It is shown that, if we are *interested in generated algebras only*,
we can even further *restrict attention to quotients of the term algebra*!

Lemma 6.10.
Let Σ be a signature and A be any generated Σ-Algebra.
Then $A \simeq T(\Sigma)_{\equiv_h}$, where $\equiv_h$ is the congruence relation induced by the evaluation homomorphism $h : T(\Sigma) \longrightarrow A$. ⋆

Theorem 6.11.
Let Σ be a signature and A by any Σ-Algebra. Then:

$$A \text{ is generated}$$
$$\Longleftrightarrow$$
$$\text{there exists a congruence relation } Q \text{ on } T(\Sigma) \text{ such that } A \simeq T(\Sigma)/_Q$$

⋆

Term Algebra Quotients are Sufficient

Theorem 6.11 guarantees that

1. every generated algebra 'is' a quotient of the term algebra (up to isomorphism)

2. every other algebras is not

Therefore: only looking at the various quotients of the term algebra does

1. 'capture' all generated algebras,

2. not 'capture' ungenerated algebras.

isn't that great ?

Exercise

Exercise 6.6.
Prove lemma 6.10 and theorem 6.11.
(The proofs are short, if you apply the right theorems.)

Appropriate Congruences

given a class of algebras

Idea: If certain terms are *identified by all algebras* of the class, we make them 'equal' in the quotient term algebra.

The resultung quotient posseses all equational properties common to the algebras.

formally:

Let Σ be a signature and $\mathcal{C} \subseteq Alg(\Sigma)$ be a class of algebras.

The *congruence relation induced by $\mathcal{C}$* is the family

$\equiv_{\mathcal{C}} = (\equiv_{\mathcal{C},s})_{s \in S}$ of relations on T_Σ defined by:

$$\equiv_{\mathcal{C},s} = \{ (t,u) \in T_{\Sigma,s} \times T_{\Sigma,s} \mid A(t) = A(u) \text{ for each } A \in \mathcal{C} \}$$

Congruence?

Theorem 6.12.
A congruence relation $\equiv_{\mathcal{C}}$ induced by a class $\mathcal{C} \subseteq Alg(\Sigma)$ is indeed a congruence. ⋆

Exercise 6.7.
Prove theorem 6.12.

Appropriate Quotients

Let $\mathcal{C} \subseteq Alg(\Sigma)$ be a non-empty class of Σ-Algebras for some signature Σ.

The *quotient term algebra of the class $\mathcal{C}$*, written $T(\Sigma, \mathcal{C})$, is defined by:

$$T(\Sigma, \mathcal{C}) = T(\Sigma)/_{\equiv_{\mathcal{C}}}$$

The following plays a fundamental role in the definition of the semantics (see below):

Theorem 6.13.
Let $\mathcal{C} \subseteq Alg(\Sigma)$ be a non-empty class of Σ-Algebras for some signature Σ. For each algebra $A \in \mathcal{C}$ there exists a unique homomorphism from $T(\Sigma, \mathcal{C})$ to A, with $h([t]) = A(t)$ for each ground term $t \in T_\Sigma$ and called the evaluation homomorphism. ⋆

Exercise

Exercise 6.8.
Prove theorem 6.13.

Part 7: Logic

Logics with equality

Finally, we define logics, i.e. syntax and semantics for the language of our axioms.

We distinguish between three logics:

- *(purely) equational logic EL*
- *conditional equational logic CEL*
- *first-order predicate logic PL*

Equational Logic

For each signature Σ, the set of *formulas of the equational logic EL* is:

$$EL(\Sigma) = \{t = u \mid t, u \in T_{\Sigma(X),s}, \text{ for some } s \in S, X \text{ a set of variables for } \Sigma\}$$

A formula $\varphi \in EL(\Sigma)$ is also called equation.

$Var(t = u) = Var(t) \cup Var(u)$ is the *set of variables occurring in φ*.

Equational Logic
‘Satisfies’

Let Σ be a signature. *‘An algebra A satisfies an equation’* $t = u$, written $A \models_\Sigma t = u$, is defined by:

$$A \models_\Sigma t = u$$
$$:\Longleftrightarrow$$
$$A(\alpha)(t) = A(\alpha)(u)$$
$$\text{f.a. assignments } \alpha : X \longrightarrow A \text{ with } X \supseteq Var(t = u)$$

Example: Be A the ‘classical’ **NATBOOL** algebra:

$A \models x + 1 = 1 + x$
$A \not\models x = y$
$A \models x = x$

Non-Distinguishable Algebras

Theorem 7.1.
For any signature Σ, and any equality $\varphi \in EL(\Sigma)$, and for any Σ-Algebras A, B with $A \simeq B$:

$$A \models_\Sigma \varphi \quad \Longleftrightarrow \quad B \models_\Sigma \varphi$$

⋆

Exercise 7.1.
Prove theorem 7.1.

Conditional Equational Logic

For each signature Σ, the set of *formulas of the equational logic EL* is:

$$CEL(\Sigma) \quad = \quad \{ t_1 = u_1 \wedge \ldots \wedge t_k = u_k \rightarrow t_{k+1} = u_{k+1} \mid t_i, u_i \in T_{\Sigma(X),s_i} \text{ for some } s_i \in S, k \geq 0, X \text{ a set of variables for } \Sigma \}$$

A formula $\varphi \in CEL(\Sigma)$ is also called a conditional equation.

$Var(\varphi) = Var(t_1) \cup Var(u_1) \cup \ldots \cup Var(t_{k+1}) \cup Var(u_{k+1})$ is the *set of variables occurring in φ*.

Conditional Equational Logic
‘Satisfies’

Let Σ be a signature. *‘An algebra A satisfies a conditional equation’* is defined by:

$$A \models_\Sigma t_1 = u_1 \wedge \ldots \wedge t_k = u_k \;\rightarrow\; t_{k+1} = u_{k+1}$$
$$:\Longleftrightarrow$$
$$\text{if } A(\alpha)(t_1) = A(\alpha)(u_1) \text{ and } \ldots \text{ and } A(\alpha)(t_k) = A(\alpha)(u_k)$$
$$\text{then } A(\alpha)(t_{k+1}) = A(\alpha)(u_{k+1})$$
$$\text{f.a. assignments } \alpha : X \longrightarrow A \text{ with } X \supseteq Var(\varphi)$$

Conditional Equation Example

Consider **NATBOOL**, augmented with: $- : nat \times nat \to nat$.

Then these are conditional equations:

$$x \leq y = \mathit{True} \quad \to \quad x - y = 0$$
$$x \leq y = \mathit{False} \quad \to \quad (y - x) + x = y$$

First-Order Predicate Logic

For each signature Σ, the set of *formulas of the first-oder predicate logic $PL(\Sigma)$* is the minimal set satisfying:

- $EL(\Sigma) \subseteq PL(\Sigma)$
- if $\varphi_1, \varphi_2 \in PL(\Sigma)$, then $(\varphi_1 \wedge \varphi_2) \in PL(\Sigma)$
- if $\varphi \in PL(\Sigma)$, then $(\neg\varphi) \in PL(\Sigma)$
- if $\varphi \in PL(\Sigma)$, $s \in S$ and $x \in X_s$, then $\forall x : s.\ \varphi \in PL(\Sigma)$

First-Order Predicate Logic

Other connectives can be defined as abbreviations.

$(\varphi_1 \vee \varphi_2)$	stands for	$(\neg((\neg\varphi_1) \wedge (\neg\varphi_2)))$
$(\varphi_1 \to \varphi_2)$	stands for	$((\neg\varphi_1) \vee \varphi_2)$
$(\varphi_1 \leftrightarrow \varphi_2)$	stands for	$((\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1))$
$(\exists x : s.\ \varphi)$	stands for	$(\neg(\forall x : s.\ \neg\varphi))$

The free variables of a formula φ, $\mathit{free}(\varphi)$, are defined like:

$\mathit{free}(t = u) = \mathit{Var}(t = u)$
$\mathit{free}(\varphi_1 \wedge \varphi_2) = \mathit{free}(\varphi_1) \cup \mathit{free}(\varphi_2)$
$\mathit{free}(\neg\varphi) = \mathit{free}(\varphi)$
$\mathit{free}(\forall x : s.\ \varphi) = \mathit{free}(\varphi) - \{x\}$

First-Order Predicate Logic
Value of Formulas

Let Σ be a signature.

The *value $A(\alpha)(\varphi)$* of a formula $\varphi \in PL(\Sigma)$ for an assignment $\alpha : X \longrightarrow A$ with $X \supseteq \mathit{free}(\varphi)$ is defined inductively on the structure of φ:

$$
\begin{aligned}
A(\alpha)(t = u) = \text{true} \quad & \text{iff} \quad A(\alpha)(t) = A(\alpha)(u) \\
A(\alpha)(\varphi_1 \wedge \varphi_2) = \text{true} \quad & \text{iff} \quad (A(\alpha)(\varphi_1) = \text{true and } A(\alpha)(\varphi_2) = \text{true}) \\
A(\alpha)(\neg\varphi) = \text{true} \quad & \text{iff} \quad A(\alpha)(\varphi) = \text{false} \\
A(\alpha)(\forall x : s.\ \varphi) = \text{true} \quad & \text{iff} \quad A(\alpha^a_x)(\varphi) = \text{true for all } a \in A(s)
\end{aligned}
$$

First-Order Predicate Logic
'Satisfies'

Let Σ be a signature. *'An algebra A satisfies a formula φ'* is defined by:

$$A \models_\Sigma \varphi$$
$$:\Longleftrightarrow$$
$$A(\alpha)(\varphi) = \text{true}$$
$$\text{f.a. assignments } \alpha : \mathit{free}(\varphi) \longrightarrow A$$

Non-Distinguishable Algebras

Theorem 7.1 also holds for CEL and PL.

Theorem 7.2.
For any signature Σ, and any formula $\varphi \in CEL(\Sigma)$ resp. $\varphi \in PL(\Sigma)$ and for any Σ-Algebras A, B with $A \simeq B$:

$$A \models_\Sigma \varphi \quad \Longleftrightarrow \quad B \models_\Sigma \varphi$$

⋆

Exercise 7.2.
Prove theorem 7.2.

(Interesting) Universes of Algebras

In algebraic specification:

One typically restricts the class of algebras one is interested in, (in the one or other way).

We call such 'interesting' classes *universes $\mathcal{U}$*.

The only requirement: $\mathcal{U}$ must be closed under isomorphism.

Examples of such universes: $Alg(\Sigma)$, $Gen(\Sigma)$.

Models of Formulas

Let L be a logic, Σ a signature and $\Phi \subseteq L(\Sigma)$ a set of formulas.

A Σ-Algebra A is called a *model of Φ*, if $A \models_\Sigma \varphi$ f.a. $\varphi \in \Phi$.
We write also $A \models \Phi$.

Let $\mathcal{U}$ be a universe.

The class of *algebras of $\mathcal{U}$ that are models of Φ* is denoted $Mod_{\mathcal{U},\Sigma}(\Phi)$.

Two special cases:

- $Mod_\Sigma(\Phi) := Mod_{Alg(\Sigma),\Sigma}(\Phi)$ is the class of *all models of Φ*.
- $Ind_\Sigma(\Phi) := Mod_{Gen(\Sigma),\Sigma}(\Phi)$ is the class of *generated models of Φ*.

Models and Abstract Data Types

Theorem 7.3.
$Mod_{\mathcal{U}}(\Phi)$ is always an abstract data type. ⋆

This the basis to the constructor based approach to semantics.

There, we consider only $\mathcal{U}$'s which are generated (by certain constructors), and intersect this with all models of the specification Φ.

Precise Definition below.

Exercise 7.3.
Prove theorem 7.3.

Models and Quotients

Here comes the basis to the initial approach to semantics.

Let L be a logic, Σ a signature and $\Phi \subseteq L(\Sigma)$ a set of formulas.

The *quotient term algebra of Φ* is defined to be the quotient term algebra of the class $Mod_\Sigma(\Phi)$ (of *all* models of Φ).

We write $\equiv_\Phi$ instead of $\equiv_{Mod_{L,\Sigma}(\Phi)}$.

We write $T(\Sigma, \Phi)$ instead of $T(\Sigma, Mod_\Sigma(\Phi))$.

Fact 7.4.
If the quotient term algebra $T(\Sigma, \Phi)$ is a model of Φ, then it is initial in $Mod_\Sigma(\Phi)$. ⋆

Logical Consequence

We overload the symbol '$\models$'. It not only denotes the 'satisfies' relation between algebras and formulas, but also the '*logical consequence*' relation between formulas and formulas.

Let L be a logic, Σ a signature, $\varphi \in L(\Sigma)$ a formula, and $\Phi \subseteq L(\Sigma)$ a set of formulas. Let $\mathcal{U}$ be a universe of algebras.

φ is a *locial consequence of Φ in $\mathcal{U}$*, written $\Phi \models_{\mathcal{U},\Sigma} \varphi$,
if $A \models \varphi$ f.a. $A \in Mod_{\mathcal{U},\Sigma}(\Phi)$.

φ is a *locial consequence of Φ*, written $\Phi \models_\Sigma \varphi$,
if $\Phi \models_{Alg(\Sigma),\Sigma} \varphi$.

φ is an *inductive consequence of Φ*, written $\Phi \models_{Ind,\Sigma} \varphi$,
if $\Phi \models_{Gen(\Sigma),\Sigma} \varphi$.

Logical Consequence (cont'd)

In words:

- φ is a locial consequence of Φ,
  '$\Phi \models_\Sigma \varphi$',
  if φ is satisfied by *all* models of Φ.

- φ is an inductive consequence of Φ,
  '$\Phi \models_{Ind,\Sigma} \varphi$',
  if φ is satisfied by all *generated* models of Φ.

Logical Consequence Example

$$\begin{aligned}\Phi \;=\; \{\; & x + 0 = x, \\ & x + Succ(y) = Succ(x + y), \\ & (x + y) + z = x + (y + z) \;\}\end{aligned}$$

Then:

- $\Phi \not\models 0 + x = x$

  consider: A with $A(nat) = \mathbb{N}_0$, $A(0) = 0$, $A(Succ)(m) = m$, $A(+)(m, n) = m$

- but:

  $\Phi \models_{Ind} 0 + x = x$

  (note: A above was not generated)

Part 8: Loose Specifications

Genuine Loose Specifications

Let L be a logic.

- A *(genuine) loose specification* in L is a pair $sp = (\Sigma, \Phi)$, where Σ is a signature and $\Phi \subseteq L(\Sigma)$ is a set of formulas.

- The *meaning* $\mathcal{M}(sp)$ is the abstract data type $\mathcal{M}(sp) = Mod_\Sigma(\Phi)$

a possible concrete syntax:

loose spec	**sorts**	list-of-sorts
	opns	list-of-operations
	vars	list-of-vars
	axioms	
endspec		

Loose Specifications with Constructors

Let L be a logic.

- A *loose specification with constuctors* is a 3-tupel $sp = (\Sigma, \Phi, \Omega_c)$, where $\Sigma = (S, \Omega)$ is a signature, $\Phi \subseteq L(\Sigma)$ is a set of formulas, and Ω_c are the constructors.

- The *meaning* $\mathcal{M}(sp)$ is defined to be the abstract data type $\mathcal{M}(sp) = Mod_\mathcal{U}(\Phi)$, where

  $\mathcal{U} = \{\; A \in Alg(\Sigma) \mid A \text{ is generated by } \Omega_c \;\}$

Loose Specifications with Constructors

a possible concrete syntax:

loose spec	**sorts**	s_1 generated by $op_{1,1}, op_{1,2}, \ldots$
		s_2 generated by $op_{2,1}, op_{2,2}, \ldots$
		$\vdots$
	opns	list-of-operations
	vars	list-of-vars
	axioms	
endspec		

Loose Specifications with Free Constructors

$\mathcal{M}(sp) = Mod_{\mathcal{U}}(\Phi)$, where

$\mathcal{U} = \{ A \in Alg(\Sigma) \mid A$ is freely generated by $\Omega_c \}$

a possible concrete syntax:

loose spec	**sorts**	s_1 freely generated by $op_{1,1}, op_{1,2}, \ldots$
		s_2 freely generated by $op_{2,1}, op_{2,2}, \ldots$
		$\vdots$
	opns	list-of-operations
	vars	list-of-vars
	axioms	
endspec		

Part 9: Initial Specifications

Initial Specifications

initial specifications require a particular kind of logic: EL

(contrasting to loose specifications)

Also: initial specifications always define monomorphic data types.

Initial Specifications

- An *initial specification* is a pair $sp = (\Sigma, \Phi)$, where Σ is a signature and $\Phi \subseteq EL(\Sigma)$ is a set of formulas.

- The *meaning* $\mathcal{M}(sp)$ of an initial specification is the monomorphic abstract data type $\mathcal{M}(sp) = \{ A \in Alg(\Sigma) \mid A \simeq T(\Sigma, \Phi) \}$

a possible concrete syntax:

initial spec	**sorts**	list-of-sorts
	opns	list-of-operations
	vars	list-of-vars
	axioms	
endspec		

Initiality Property

Theorem 9.1.
Let Σ ba a signature and $\Phi \subseteq EL(\Sigma)$ a set of equations. Then:

$$T(\Sigma, \Phi) \in Mod_{\Sigma}(\Phi)$$ ⋆

Exercise 9.1.
Prove theorem 9.1.
(Somehow tricky. You might want to use the following lemma.)

Lemma 9.2.
If Q is a congruence relation on A, and $t, u \in T_{\Sigma(X)}$, then:

$$(A/_Q)(\beta)(t) = (A/_Q)(\beta)(u) \text{ for all assignments } \beta : X \longrightarrow A/_Q$$
$$\Longleftrightarrow$$
$$[A(\alpha)(t)] = [A(\alpha)(u)] \text{ for all assignments } \alpha : X \longrightarrow A.$$

⋆

Initiality Property Conclusions

Corollary 9.3.
Let $sp = (\Sigma, \Phi)$ be an initial specification.

Then each algebra in $\mathcal{M}(sp)$ is initial in $Mod_{\Sigma}(\Phi)$. ⋆

Fact 9.4.
Let $sp = (\Sigma, \Phi)$ be an initial specification.

The algebras in $\mathcal{M}(sp)$ are generated. ⋆

Initial Specification Examples

initial spec	**sorts**	nat
	opns	$0 :\rightarrow nat$
		$Succ : nat \rightarrow nat$
		$+ : nat \times nat \rightarrow nat$
	vars	$n, m : nat$
	axioms	$n + 0 = n$
		$n + Succ(m) = Succ(n + m)$
endspec		

discuss $T(\Sigma, \Phi)$

Initial Specification Examples

initial spec	**sorts**	nat, set
	opns	$0 :\longrightarrow nat$
		$Succ : nat \rightarrow nat$
		$\emptyset :\longrightarrow set$
		$Insert : nat \times set \rightarrow set$
	vars	$n, m : nat,\ ns : set$
	axioms	$Insert(n, Insert(n, s)) = Insert(n, s)$
		$Insert(n, Insert(m, s)) = Insert(m, Insert(n, s))$
endspec		

discuss $T(\Sigma, \Phi)$

Loose (here) vs. Initial

loose spec	**sorts**	**freely generated** nat
	opns	**constr** $0 :\longrightarrow nat$
		constr $Succ : nat \rightarrow nat$
		$Pred : nat \rightarrow nat$
	vars	$n : nat$
	axioms	$Pred(Succ(n)) = n$
endspec		

discuss $T(\Sigma, \Phi)$

Loose vs. Initial (here)

initial spec	**sorts**	nat
	opns	$0 :\longrightarrow nat$
		$Succ : nat \rightarrow nat$
		$Pred : nat \rightarrow nat$
	vars	$n : nat$
	axioms	$Pred(Succ(n)) = n$
endspec		

discuss $T(\Sigma, \Phi)$

Towards The Main Theorem

Three properties of equational logic:

Lemma 9.5.
Let A, B be Σ-Algebras and $h : A \longrightarrow B$ a homomorphism. Let $t = u$ be a ground equation ($Var(t = u) =, \emptyset$) and $v = w$ an arbitrary equation in $EL((\Sigma)$. Then:

- *If $A \models t = u$, then $B \models t = u$.*
- *If h is surjective and $A \models v = w$, then $B \models v = w$.*
- *If h is injective and $B \models v = w$, then $A \models v = w$.*

$\star$

The Main Theorem for Initial Specification

Theorem 9.6.
Let $sp = (\Sigma, \Phi)$ *be an initial specification.*
Let $t = u$ *be a ground equation (*$Var(t = u) = \emptyset$*) and* $v = w$ *an arbitrary equation in* $EL(\Sigma)$*. Then:*

- $T(\Sigma, \Phi) \models t = u$ *iff* $\Phi \models t = u$
- $T(\Sigma, \Phi) \models t = u$ *iff* $A \models t = u$ *f.a.* $A \in Mod_\Sigma(\Phi)$
- $T(\Sigma, \Phi) \models v = w$ *iff* $\Phi \models_{Ind} v = w$
- $T(\Sigma, \Phi) \models v = w$ *iff* $A \models v = w$ *f.a.* $A \in Ind_\Sigma(\Phi)$
- $T(\Sigma, \Phi) \models v = w$ *iff* $\Phi \models v\sigma = w\sigma$ *f.a. ground substitutions* $\sigma : X \longrightarrow T_\Sigma$

$\star$