Monad defined by a signature with bindings[1]

Vladimir Voevodsky[2,3]

July 2014

Abstract

???

1 Introduction

We first recall the notion of a signature with bindings (see [**?**] where it is simply called a signature). This is one of short papers based on the material of [**?**].

2 Signatures and expressions with bindings

Let $\mathbf{A}$ be the set of finite sequences $(n_1, \ldots, n_k)$ where $n_i \in \mathbf{N}$. Elements of $\mathbf{A}$ will be called *arities*.

Definition 2.1 *[***2014.07.14.def1***] A signature (with bindings) is a pair $\mathbf{S} = (S, a)$ where S is a set and a is a function from S to $\mathbf{A}$.*

Elements of S are called symbols or operations of the signature and for $s \in S$, $a(s)$ is called the arity of s.

Goal: define $Exp(\mathbf{S}, X)$ for $X \in FSets$ in such a way as to make the proof that it is a monad (in X) as simple as possible.

In the case when all arities are of the form $(0, \ldots, 0)$ we get the usual notion of a (single-sorted) algebraic signature.

3 Systems of expressions

Note: [**?**], [**?**].

Free systems of expressions. Let M be a set and let $T(M)$ be the set of finite rooted trees whose vertices (including the root) are labeled by elements of M and such that for any vertex the set of edges leaving this vertex is ordered. Note that such ordered trees have no symmetries. We will use the following notations. For $T \in T(M)$ let $Vrtx(T)$ be the set of vertices of T and for $v \in Vrtx(T)$ let $lbl(v) = lbl(v)_T \in M$ be the label on v. We will sometimes write $v \in T$ instead of $v \in Vrtx(T)$. For $v \in Vrtx(T)$ let $[v] = [v]_T \in T(M)$ be the subtree in T which consists of v and all the vertices under v. Let $val(v)$ be the valency of v i.e. the number of edges leaving v

[1] *2000 Mathematical Subject Classification: 03B15, 03B22, 03F50, 03G25*

[2] School of Mathematics, Institute for Advanced Study, Princeton NJ, USA. e-mail: vladimir@ias.edu

[3] Work on this paper was supported by NSF grant 1100938.

and $ch_1(v), \ldots, ch_{val(v)}(v) \in Vrtx(T)$ be the "children" of v i.e. the end points of these edges. Let further $br_i(v) = [ch_i(v)]$ be the branches of $[v]$. We write $v \leq w$ (resp. $v < w$) if $v \in [w]$ (resp. $v \in [w] - w$). We say that two vertices v and w are independent if $v \notin [w]$ and $w \notin [v]$.

For three sets A, B and $Cont$ let

$$AllExp(A, B; Con) = T(A \amalg B \amalg (Con \times (\amalg_{n \geq 0} B^n)))$$

Elements of $AllExp(A, B; Con)$ are called expressions over the alphabet Con (or with a set of constructors Con), free variables from A and bound variables from B.

An expression is called unambiguous if it satisfies the following conditions:

1. if $lbl(v) \in A \amalg B$ then $val(v) = 0$,
2. (a) if $v < v'$, $lbl(v) = (c; x_1, \ldots, x_n)$ and $lbl(v') = (c'; x'_1, \ldots, x'_{n'})$ then $\{x_1, \ldots, x_n\} \cap \{x'_1, \ldots, x'_{n'}\} = \emptyset$,
   (b) if $lbl(v) = (c; x_1, \ldots, x_n)$ then $x_i \neq x_j$ for $i \neq j$,
3. if $lbl(v) = (c; x_1, \ldots, x_n)$ and $lbl(v') \in \{x_1, \ldots, x_n\}$ then $v' \in [v]$.

The first conditions says that a vertex labeled by a variable is a leaf. The second one is equivalent to saying that if the same variable is bound at two different vertices v, v' then these vertices are independent i.e. $[v] \cap [v'] = \emptyset$ and that a vertex can not bind the same variable twice. The third one says that all the leaves labeled by a bound variable lie under the vertex where it is boud. We let $UAExp(A, B; Con)$ denote the subset of unambiguous expressions in $AllExp(A, B; Con)$. Note that for for any $T \in UAExp(A, B; Con)$ and $v \in Vrtx(T)$ there is a subset $Ext(v) \subset B$ such that

$$[v] \in UAExp(A \amalg Ext(v), B \backslash Ext(v); Con)$$

Any triple of maps $f_{Con} : A \to A'$, $f_B : B \to B'$, $f_{Con} : Con \to Con'$ define a map

$$f_* = (f_A, f_B, f_{Con})_* : AllExp(A, B; Con) \to AllExp(A', B'; Con')$$

which changes labels in the obvious way. If f_B is injective then f_* maps unambiguous expressions to unambiguous ones.

An element T of $UAExp(A, B; Con)$ is said to be strictly unambiguous if for any $v \neq v'$ in $Vrtx(T)$ such that $lbl(v) = (c; x_1, \ldots, x_n)$ and $lbl(v') = (c'; x'_1, \ldots, x'_{n'})$ one has $\{x_1, , \ldots, x_n\} \cap \{x'_1, \ldots, x'_{n'}\} = \emptyset$ i.e. if the names of all bound variables are different. We let $SUAExp(A, B; Con)$ denote the subset of strictly unambiguous expressions in $UAExp(A, B; Con)$.

An element T of $UAExp(A, B; Con)$ is said to be α-equivalent to an element T' of $UAExp(A, B'; Con)$ if there is a set B'' , an element $T'' \in UAExp(A, B''; Con)$ and two maps $f : B'' \to B$, $f' : B'' \to B'$ such that $T = (Id, f, Id)_*(T'')$ and $T' = (Id, f', Id)_*(T'')$. The following lemma is straightforward:

Lemma 3.1 *[**2009.09.08.l1**] For any two sets A and Con one has:*

1. *α-equivalence is an equivalence relation,*
2. *for any set B and any element $T \in UAExp(A, B; Con)$ there exists an element $T' \in UAExp(A, \mathbf{N}; Con)$ such that $T \stackrel{\alpha}{\sim} T'$ and T' is strictly unambiguous,*

3. *fwo strictly unambiguous elements $T, T' \in UAExp(A, B; Con)$ are α-equivalent if and only if there exists a permutation $f : B \to B$ such that $(Id, f, Id)_*(T) = T'$ (cf. swapping).*

We let $Exp_\alpha(A; Con)$ denote the set of α-equivalence classes in $\amalg_B UAExp(A, B; Con)$. In view of Lemma **??** this set is well defined and can be also defined as the set of equivalence classes in $SUAExp(A, \mathbf{N}; Con)$ modulo the equivalence relation generated by the permutations on $\mathbf{N}$.

Note that for two α-equivalent expressions T_1, T_2 and a vertex $v \in V(T_1) = V(T_2)$ the expressions $[v]_{T_1}$ and $[v]_{T_2}$ need not be α-equivalent since some of the variables which are bound in T_1 may be free in $[v]$.

The maps $(f_A, f_B, f_{Con})_*$ respect α-equivalence. Therefore for any $f_A : A \to A'$ and $f_{Con} : Con \to Con'$ there is a well defined map

$$(f_A, f_{Con})_* : Exp_\alpha(A; Con) \to Exp(A'; Con')$$

which make $Exp_\alpha(-; -)$ into a covariant functors from pairs of sets to sets. In addition there is a well defined notion of substitution on $Exp_\alpha(-; Con)$ which may be considered as a collection of maps of the form:

$$Exp_\alpha(A; Con) \times (\prod_{a \in A} Exp_\alpha(X_a; Con)) \to Exp_\alpha(\amalg_{a \in A} X_a; Con)$$

given for all pairs $(A; \{X_a\}_{a \in A})$ where A is a set and $\{X_a\}_{a \in A}$ a family of sets parametrized by A. Alternatively, the substitution structure can be seen as a collection of maps

$$Exp_\alpha(Exp_\alpha(A; Con); Con) \to Exp_\alpha(A; Con)$$

given for all A and Con. These maps make the functor $Exp_\alpha(-; Con)$ into a monad (triple) on the category of sets which functorially depends on the set Con.

Example 3.2 [**lambda**] The mapping which sends a set X to the set of α-equivalence classes of terms of the untyped λ-calculus with free variables from X is a sub-triple of $Exp_\alpha(-; Con)$ where $Con = \{\lambda, ev\}$. Elements T of $UAExp(X, \mathbf{N}; \{\lambda, ev\})$ which belong to this sub-triple are characterized by the following "local" conditions:

1. for each $v \in T$, $lbl(v) \in X \amalg \mathbf{N} \amalg \{ev\} \amalg \{\lambda\} \times \mathbf{N}$
2. if $lbl(v) \in \{\lambda\} \times \mathbf{N}$ then $val(v) = 1$
3. if $lbl(v) = ev$ then $val(v) = 2$.

Example 3.3 *[***propositional***]*The mapping which sends a set X to the set of terms of the propositional calculus with free variables from X is a sub-triple of $Exp_\alpha(-; C_0)$ where $C_0 = \{\vee, \wedge, \urcorner, \Rightarrow\}$. Elements T of $UAExp(X, \mathbf{N}; C_0)$ which belong to this sub-triple are characterized by the following "local" conditions:

1. for all $v \in T$, $lbl(v) \in X \amalg C_0$
2. if $lbl(v) \in \{\vee, \wedge, \Rightarrow\}$ then $val(v) = 2$
3. if $lbl(v) = \urcorner$ then $val(v) = 1$.

Example 3.4 [**multisorted**] Consider first order logic with several sorts $GS = \{S_1, \ldots, S_n\}$. Let GP be the set of generating predicates and GF the set of generating functions. Let $C_1 = C_0 \amalg \{\forall, \exists\}$ and $C_2 = C_1 \amalg GP \amalg GF \amalg GS$. We can identify the α-equivalence classes of formulas of the first order language defined by GS and GF with free variables from a set X with a subset in $Exp_\alpha(X, \mathbf{N}; C_2)$. Vertices which are labeled by $(\forall; x)$ and $(\exists; x)$ have valency two. For such a vertex v, the first branch of $[v]$ is one vertex labeled by an element of GS giving the sort over which the quantification occurs and the second branch is the expression which is quantified. Now however, these subsets do not form a sub-triple of Exp_α since not all substitutions are allowed. By allowing all substitutions irrespectively of the sort we get (for each X) a subset in $Exp_\alpha(X; C_2)$ whose elements will be called pseudo-formulas.

The following operations on expressions are well defined up to the α-equivalence:

1. If $T_1, \ldots, T_m \in Exp_\alpha(A; Con)$, $a_1, \ldots, a_n$ are pair-wise different elements of A and $M \in Con$ we will write $(M, a_1, \ldots, a_n)(T_1, \ldots, T_m)$ for the expression whose root v is labeled by $(M, a_1, \ldots, a_n)$, $val(v) = n$ and $br_i(v) = T_i$.
2. For $T_1, T_2 \in Exp_\alpha(A; Con)$ and $v \in T_1$ we let $T_1(T_2/[v])$ be the expression obtained by replacing $[v]$ in T_1 with T_2' where T_2' is obtained from T_2 by the change of bound variables such that the bound variables of T_2' do not conflict with the variables of T_1.
3. For $T, R_1, \ldots, R_n \in Exp_\alpha(A; Con)$ and $y_1, \ldots, y_n \in A$ we let $T(R_1/y_1, \ldots, R_n/y_n)$ denote the expression obtained by changing R_i's by α-equivalent R_i' such that $bnd(R_i') \cap bnd(R_j)' = \emptyset$ for $i \neq j$, changing T to an α-equivalent T' such that $bnd(T') \cap (var(R_1') \cup \ldots \cup var(R_n')) = \emptyset$ and then replacing all the leaves of T' marked by y_i by R_i'.

In all the examples considered above, these operations correspond to the usual operations on formulas. The first operation can be used to directly associate expressions in our sense with the formulas. For example, the expression associated with the formula $\forall x : S.P(x, y)$ in a multi-sorted predicate calculus is $(\forall, x)(S, P(x, y))$ where as was mentioned above we use the same notation for an element of $A \amalg B \amalg (Con \times (\amalg_{n \geq 0} B^n))$ and the one vertex tree with the corresponding label.

Note: about representing elements of $AllExp(A, B; Con)$ by linear sequences of elements of $A \amalg B \amalg$??.

Reduction structures. Another component of the structure present in systems of expressions used in formal systems is the reduction relation. It is very important for our approach to type systems that the reduction relation is defined on all pseudo-formulas and is compatible with the substitution structure even when not all psedu-formulas are well formed formulas. In what follows we will consider, instead of a particular syntactic system, a pair $(S, \triangleright)$ where S is a continuous triple on the category of sets and $\triangleright$ is a reduction structure on S i.e. a collection of relations $\triangleright_X$ on $S(X)$ given for all finite sets X satisfying the following two conditions:

1. if $E \in S(\{x_1, \ldots, x_n\})$, $f_1, \ldots, f_n, f_i' \in S(\{y_1, \ldots, y_m\})$ and $f_i \triangleright_{\{y_1, \ldots, y_m\}} f_i'$ then
$$E(f_1/x_1, \ldots, f_i/x_i, \ldots f_n/x_n) \triangleright_{\{x_1, \ldots, x_n\}} E(f_1/x_1, \ldots, f_i'/x_i, \ldots f_n/x_n),$$
2. if $E, E' \in S(\{x_1, \ldots, x_n\})$, $f_1, \ldots, f_n \in S(\{y_1, \ldots, y_m\})$ and $E \triangleright_{\{x_1, \ldots, x_n\}} E'$ then
$$E(f_1/x_1, \ldots, f_n/x_n) \triangleright_{\{x_1, \ldots, x_n\}} E'(f_1/x_1, \ldots, f_n/x_n).$$

The following two results are obvious but important.

Proposition 3.5 *[2009.10.17.prop1] Let S be a continuous triple on Sets and $\triangleright_\alpha$ be a family of reduction structures on S. Then the intersection $\cap_\alpha \triangleright_\alpha : X \mapsto \cap_\alpha \triangleright_{\alpha,X}$ is a reduction structure on S.*

Corollary 3.6 *[2009.10.17.cor1] For any family (X_α, pre_α) of pairs of the form (X, pre) where X is a set and pre is a relation on $S(X)$ (i.e. a subset of $S(X) \times S(X)$) there exists the smallest reduction structure $\triangleright = \triangleright(X_\alpha, pre_\alpha)$ on S such that for each α and each $(f,g) \in pre_\alpha$ one has $f \triangleright g$.*