

A Framework for Order-Sorted Algebra

John G. Stell

School of Computing
University of Leeds
LEEDS, LS2 9JT, U. K.
email: jgs@comp.leeds.ac.uk

Abstract. Order-sorted algebra is a generalization of many-sorted algebra obtained by having a partially ordered set of sorts rather than merely a set. It has numerous applications in computer science. There are several variants of order sorted algebra, and some relationships between these are known. However there seems to be no single conceptual framework within which all the connections between the variants can be understood. This paper proposes a new approach to the understanding of order-sorted algebra. Evidence is provided for the viability of the approach, but much further work will be required to complete the research programme which is initiated here.

The programme is based on the investigation of two topics. Firstly an analysis of the various categories of order-sorted sets and their relationships, and, secondly, the development of abstract notions of order-sorted theory, as opposed to presentations given by a signature of operation symbols. As a first step, categories of order-sorted sets are described, adjunctions between the categories are obtained, and results on order-sorted theories as categories, in the sense of Lawvere, are obtained.

1 Introduction

1.1 Order-Sorted Algebra in Computer Science

Formal systems in Computer Science often make use of sorts or types to structure the universe of discourse. In what is usually called ‘many-sorted’ algebra, the sorts form a set, but do not support additional structure. Many-sorted algebra and logic has been widely applied in Computer Science; some of these applications are described in the book [MT93].

A generalization of many-sorted algebra is obtained by taking the set of sorts to be partially ordered. This generalization is known as order-sorted algebra. Order-sorted algebra is generally applied where the the universe of discourse supports an hierarchical structure. Order-sorted concepts have been widely used in Computer Science. The following list gives a few examples of applications.

Computer Algebra. Hearn and Schrufer [HS95] have used order-sorted algebra as the underlying model in the design of a computer algebra system.

Object Orientation. Several approaches [Wie91,AG94] to a formal semantics for the object-oriented programming paradigm rely on a partially ordered set of sorts to express inheritance between classes.

Spatial Databases. Erwig and Güting’s data model for spatial data [EG94] is specified in a multi-level order-sorted algebra.

Formal Methods Algebraic Specification with a poset of sorts, to deal with partially defined operations, was one of the original motivations for order-sorted algebra [Gog78]. Subsequent research has included work on parameterization [Poi90,Qia94], and on the incorporation of higher-order functions [Hax95].

Logic Programming. The language PROTOS(L), has an operational semantics based on polymorphic order-sorted resolution [BM94].

1.2 Variants of Order-Sorted Algebra

The early paper by Oberschelp [Obe62] defines sorts by unary predicates on an unsorted universe, and defines subsorts by implications among these predicates. However, it was the work of Goguen [Gog78] which introduced the concept of order-sorted algebra and applied it to problems in computation. Since then the theory has been developed by several authors, and the key papers include [Wal92,GM92,SS89,Poi90]. Note that Robinson [Rob94] uses ‘order-sorted’ in a quite different sense.

There are several distinct versions of order-sorted algebra, and individual variants have been studied in detail, there is no satisfactory coherent account of all the relationships between the various approaches. The multiplicity of forms is illustrated in Figure 1, which is taken from Goguen and Diaconescu’s survey [GD94]. This survey is useful, but the relationships shown in Figure 1 rely on descriptions which take the form of a particular syntactic presentation, and then a semantics based on this syntax for each of the variants. Taking such an approach does not provide presentation independent descriptions, which is an important aspect of the framework in the present paper. Another approach to understanding order-sortedness is via membership equational logic [BJM00], which allows the representation of various versions of order-sorted algebra within its formalism. The following section proposes a programme for understanding order-sorted algebra which is alternative to both these approaches.

2 A Proposed Framework for Order-Sorted Algebra

It is the contention of this paper that to provide a framework for understanding all the various forms of order-sorted algebra two issues need to be addressed.

1. It is necessary to understand the various possible categories of order-sorted sets and their relationships.
2. It is necessary to have a notion of theory, in the presentation independent sense, for order-sorted algebra.

These two issues are discussed in more detail below. Later sections of the paper make a substantial contribution to carrying out the proposed programme.

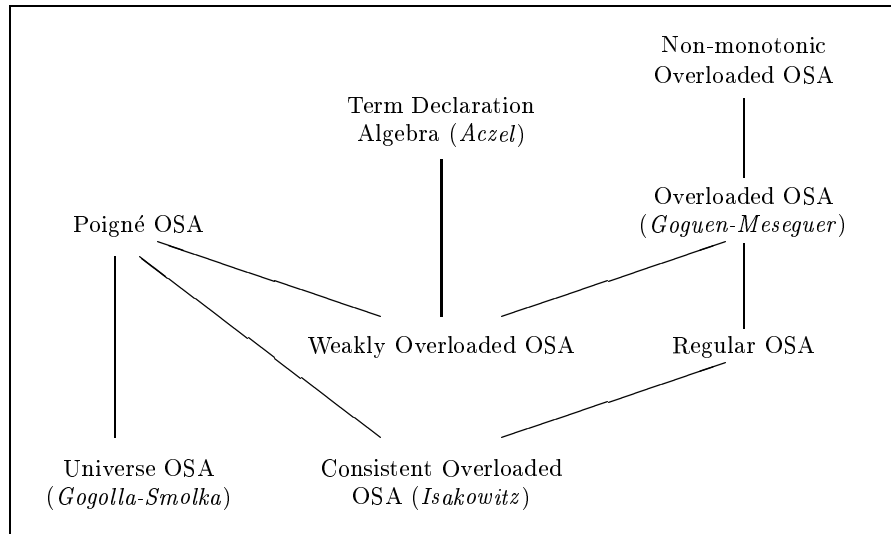


Fig. 1. Goguen and Diaconescu's View of Order-sorted Algebra

2.1 Categories of Order-Sorted Sets

In unsorted, or single-sorted, algebra, the interpretation of a signature consists of a carrier set for the algebra, and functions corresponding to the function symbols in the signature. This situation has a straightforward generalization to the many-sorted case, but the extension to order-sorted algebra is substantially more complex. The complications are due to the fact that there are several different choices for a category of order-sorted sets whose objects are carriers for order-sorted algebras. This multiplicity of categories may explain why several variants of order-sorted algebra have arisen, and section 4 below provides a comprehensive treatment of the various categories.

2.2 Order-Sorted Theories

In unsorted algebra, the notion of a theory as opposed to a particular presentation of a theory has proved useful. Category theory provides two related approaches to this viewpoint: monads and Lawvere theories. Neither has been fully explored for all the variants of order-sorted algebra. Moving to this level of abstraction loses the concrete signature. However, this is appropriate when the aim, as in this paper, is to study the nature of order-sortedness rather than to apply it. For programming or specification a concrete syntax is necessary.

Lawvere Theories In Lawvere's approach, a theory is a category with certain structure, and its algebras are structure preserving functors to a suitable category. For equational logic the theory category has products, moving beyond

equational logic leads to a hierarchy of categories with more structure [AR94] which provide presentation independent descriptions of theories for richer logics.

Section 6 provides a generalization of algebraic theories by using a suitably enriched notion of product. These poset enriched theories, having total local products in the sense of Jay [Jay90] are adequate for capturing at least one of the forms of order-sorted algebra identified by Goguen and Diaconescu. The algebras for the weakly overloaded case are functors to the category of sets and partial functions. The strongly overloaded approach is more complicated, and some further work is needed.

There has been some other work which is relevant to the quest for the appropriate notion of algebraic theory in the order-sorted case. It is known [GM92] that any variety of strongly overloaded order-sorted algebras is equivalent to a quasi-variety of many-sorted algebras. This might appear to suggest that presentation-independent approaches to order-sorted algebra are already known via such approaches to quasi-varieties of many-sorted algebras, for example [Kea75]. However, this analysis can only be applied immediately to the strongly overloaded case. It also fails to provide a sharp characterization of varieties of order-sorted algebras, as not every quasi-variety of many-sorted algebras is equivalent to a variety of order-sorted algebras.

In Aczel's paper on term declaration logic [Acz91], finitary generalized composita are suggested as the appropriate analogue of algebraic theories for this form of order-sorted algebra, but it is admitted that further work is needed to verify this. The question of whether finitary generalized composita can be used for other forms of order-sorted algebra has not been considered.

Monads An alternative approach to presentation independent theories is provided by the notion of a monad. In [Ste91], I showed how the versions of order-sorted algebra used in Schmidt-Schaß [SS89] and in [MGS89] could be related by mapping each to a morphism between two monads. One of the monads is on **Set**, and the other in the lax slice category $\mathbf{Set} // S$, which is defined in section 4.2 below. Poigné [Poi90] does mention that one category of order-sorted algebras is algebraic over the category denoted **Ext** below, but it is not clear whether all forms of order-sorted algebra can be described as monads.

3 Three Forms of Order-Sorted Algebra

Before commencing the proposed framework, we review details of some forms of order-sorted algebra. The aim of this section is to identify certain categories which arise in order-sorted algebra. We will use (S, \leq) , or just S , to denote a partially ordered set of sorts.

3.1 The Categories **Ext** and **Comp**

An (S, \leq) -sorted set is defined to be a family of sets A_s indexed by S such that $s_1 \leq s_2 \Rightarrow A_{s_1} \subseteq A_{s_2}$. If A is an S -sorted set, \overline{A} will denote $\bigcup_{s \in S} A_s$. For $a \in \overline{A}$, the set of all sorts of a , i.e. $\{s \in S \mid a \in A_s\}$, will be denoted by $\text{sorts}(a)$.

Morphisms between S -sorted sets can be defined in various ways. A **compatible morphism** $\varphi : A \rightarrow B$ is a family of functions $\varphi_s : A_s \rightarrow B_s$ such that if $s \leq t$, then φ_s and φ_t agree on A_s . This gives the category **Comp**, of (S, \leq) -sorted sets and compatible morphisms. A compatible morphism $\varphi : A \rightarrow B$ is **extensible** if whenever $a \in A_s \cap A_t$ then $\varphi_s(a) = \varphi_t(a)$. The category **Ext** of (S, \leq) -sorted sets and extensible morphisms is a subcategory of **Comp**.

3.2 Signatures and Algebras

An (S, \leq) -sorted signature Σ is a family of sets $\Sigma_{w,s}$ indexed by $S^* \times S$. This one notion of signature gives rise to two notions of algebra. These differ in the semantics given to overloaded function symbols. For example, suppose we have a function symbol $\sigma : v \rightarrow s$ and $\sigma : w \rightarrow t$. In **extensible overloading** we require that $A_{\sigma_{v,s}}$ and $A_{\sigma_{w,t}}$ agree on $A_v \cap A_w$. In **compatible overloading** we require that if $v \leq w$ and $s \leq t$ then $A_{\sigma_{v,s}}$ and $A_{\sigma_{w,t}}$ agree on A_v . Goguen and Diaconescu [GD94] use the terms ‘weak’ and ‘strong’ rather than ‘extensible’ and ‘compatible’ respectively.

A **compatible algebra** for Σ is an (S, \leq) -sorted set, A , and for each $\sigma \in \Sigma_{w,s}$, a function $A_{\sigma_{w,s}} : A_w \rightarrow A_s$ such that

$$\left. \begin{array}{l} \sigma \in \Sigma_{v,s} \cap \Sigma_{w,t} \\ (v, s) \leq (w, t) \\ \mathbf{a} \in A_v \end{array} \right\} \Rightarrow A_{\sigma_{v,s}}(\mathbf{a}) = A_{\sigma_{w,t}}(\mathbf{a}).$$

An **extensible algebra** for Σ is an (S, \leq) -sorted set, A , and for each $\sigma \in \Sigma_{w,s}$, a function $A_{\sigma_{w,s}} : A_w \rightarrow A_s$ such that

$$\left. \begin{array}{l} \sigma \in \Sigma_{v,s} \cap \Sigma_{w,t} \\ \mathbf{a} \in A_v \cap A_w \end{array} \right\} \Rightarrow A_{\sigma_{v,s}}(\mathbf{a}) = A_{\sigma_{w,t}}(\mathbf{a}).$$

By defining a homomorphism $h : A \rightarrow B$ to be a morphism in the appropriate category such that $h_t(A_{\sigma_{w,t}}(\mathbf{a})) = B_{\sigma_{w,t}}(h_w(\mathbf{a}))$, we obtain the categories **AlgComp** $_{\Sigma}$ and **AlgExt** $_{\Sigma}$.

3.3 Least Sorts

A third category of order-sorted sets appears implicitly in the literature. An S -sorted set A **has least sorts** if for any $a \in \overline{A}$ the set **sorts**(a) has a least element. The category **Least** has least sort S -sorted sets for its objects, and compatible morphisms between these. A compatible morphism between least sort S -sorted sets is necessarily extensible, so the least sort S -sorted sets determine the same full subcategory of **Ext** and of **Comp**.

The signatures considered in [MGS89] satisfy a technical condition which implies that every term has a least sort. Thus the category **Least** is the appropriate setting here.

4 Relating Categories of Order Sorted sets

The previous section identified three categories where the objects are order-sorted sets which can be used as carriers for order-sorted algebras. The relationships between these categories, and other related ones, have not been identified before. The following diagram summarizes the results reported in this section.

$$\begin{array}{ccccc}
 \mathbf{Set}^{\mathbf{S}} & \xrightarrow{\quad} & \mathbf{Comp} & \xrightarrow{\quad} & \mathbf{Ext} & \xrightarrow{\cong} & \mathbf{Set} // \mathcal{U}^+ S \\
 & \xleftarrow{\perp} & & \xleftarrow{\perp} & & & \\
 & & & & \downarrow \perp & & \\
 & & & & \mathbf{Conn} & \xrightarrow{\cong} & \mathbf{Set} // \mathcal{C} S \\
 & & & & \downarrow \perp & & \\
 & & & & \mathbf{Least} & \xrightarrow{\cong} & \mathbf{Set} // S
 \end{array}$$

$\swarrow \approx$

In the diagram, $\xrightarrow{\perp}$ indicates an inclusion having a left adjoint. In two cases, indicated by dashed arrows, the left adjoint exists only when S satisfies certain conditions. These conditions are detailed in theorems 4 and 5. The symbol \cong denotes an isomorphism of categories, and \approx an equivalence.

Informally, the diagram can be interpreted as follows. It is well known that sets, or other objects, indexed by some kind of structure admit two types of descriptions: one in a pre-sheaf fashion and one in a bundle like way. For order-sorted sets these two extremes appear in the top line of the diagram at the left hand end and the right hand end respectively. The right-hand column of the diagram shows that three different categories of order-sorted sets can be described in a bundle-like way as lax slice categories over different categories.

4.1 Relationships between Categories

The poset (S, \leq) can be seen as a category, \mathbf{S} , having objects the elements of S , and there being a morphism from s to t iff $s \leq t$. With this viewpoint, \mathbf{Comp} is the full subcategory of the functor category $\mathbf{Set}^{\mathbf{S}}$, determined by those $F : \mathbf{S} \rightarrow \mathbf{Set}$ where if $s \leq t$ then the function $Fs \rightarrow Ft$ is an inclusion.

Theorem 1 *The inclusion of \mathbf{Comp} in $\mathbf{Set}^{\mathbf{S}}$ has a left adjoint.*

Proof The left adjoint, $\Gamma : \mathbf{Set}^{\mathbf{S}} \rightarrow \mathbf{Comp}$ is defined on an object F by taking the colimiting cocone $i : F \rightarrow \text{colim } F$ and defining $(\Gamma F)s = \{i_s(x) \in \text{colim } F \mid x \in Fs\}$. It is routine to verify that Γ is left adjoint to the inclusion. \square

Definition 1 *An S -sorted set, A , has connected sorts if for all $a \in \overline{A}$, $\text{sorts}(a)$ is a connected subset of S . The category of S -sorted sets having connected sorts and compatible morphisms will be denoted \mathbf{Conn} .*

A compatible morphism between S -sorted sets having connected sorts is necessarily extensible. Thus there are inclusions $\mathbf{Conn} \hookrightarrow \mathbf{Ext} \hookrightarrow \mathbf{Comp}$. The first of these is full but not the second.

For any object A of \mathbf{Comp} , ΓA has connected sorts. Using this fact, we can obtain the following result.

Theorem 2 *There is an equivalence between \mathbf{Comp} and \mathbf{Conn} .* □

By considering the composite $\mathbf{Comp} \xrightarrow{\cong} \mathbf{Conn} \hookrightarrow \mathbf{Ext}$, we can establish the next theorem.

Theorem 3 *The inclusion of \mathbf{Ext} into \mathbf{Comp} has a left adjoint.* □

Next we turn to conditions under which the inclusions $\mathbf{Conn} \hookrightarrow \mathbf{Ext}$, and $\mathbf{Least} \hookrightarrow \mathbf{Conn}$ have left adjoints. Recall that an **upper set** in S is a subset $X \subseteq S$ for which $x \in X$ and $x \leq s \in S$ imply $s \in X$. The smallest upper set containing X is denoted X^\uparrow . The set of all connected upper sets which contain X will be denoted $\mathbf{upconn}(X)$. If the set $\mathbf{upconn}(X)$ has a least element it will be denoted $X^{\text{e}\uparrow}$. When the set $\mathbf{upconn}(X)$ has a least element for every X such that $\emptyset \neq X \subseteq S$, we will say that S is **up-connected**.

Theorem 4 *The inclusion $\mathbf{Conn} \hookrightarrow \mathbf{Ext}$ has a left adjoint if and only if S is up-connected.*

Proof Suppose S is up-connected. Define $F : \mathbf{Ext} \rightarrow \mathbf{Conn}$ on objects by $\overline{FA} = \overline{A}$, and for all $a \in \overline{A}$, $\text{sorts}_{FA}(a) = (\text{sorts}_A(a))^{\text{e}\uparrow}$. Thus for each s , $A_s \subseteq (FA)_s$.

To define F on morphisms, suppose $\varphi : A \rightarrow B$ is extensible, and define $F\varphi$ by $\overline{F\varphi} = \overline{\varphi}$. To justify this we need to check that if $a \in (FA)_s$, then $\varphi(a) \in (FB)_s$. Note that in general if $X \subseteq Y$ then $X^{\text{e}\uparrow} \subseteq Y^{\text{e}\uparrow}$. Thus from $\text{sorts}_A(a) \subseteq \text{sorts}_B(\varphi(a))$, we get $\text{sorts}_{FA}(a) \subseteq \text{sorts}_{FB}(\varphi(a))$ as required. It is straightforward to verify the remaining details.

Conversely, suppose S is not up-connected. If there is a left adjoint, F , we will have $\mathbf{Ext}(A, K) \cong \mathbf{Conn}(FA, K)$ for any objects A of \mathbf{Ext} and K of \mathbf{Conn} . Let $X \subseteq S$ have the property that the intersection of all the elements of $\mathbf{upconn}(X)$ is not connected. The upward closure of X , X^\uparrow , cannot be connected. Let A be the S -sorted set where $\overline{A} = \{a\}$ and $\text{sorts}(a) = X^\uparrow$. Thus $A_s = \emptyset$ for $s \notin X^\uparrow$.

Suppose first that $\mathbf{upconn}(X)$ is non-empty. For any $Y \in \mathbf{upconn}(X)$ define Y^\dagger to be the object of \mathbf{Conn} where $\overline{Y^\dagger} = \{a\}$ and $\text{sorts}(a) = Y$. There is exactly one morphism in \mathbf{Ext} from A to Y^\dagger , so $\mathbf{Conn}(FA, Y^\dagger)$ also contains exactly one morphism. Thus if $Y_s^\dagger = \emptyset$, we must have $(FA)_s = \emptyset$. Since this holds for all $Y \in \mathbf{upconn}(X)$, $(FA)_s$ can only be non-empty when s belongs to the intersection of all the elements of $\mathbf{upconn}(X)$. Thus either $\overline{FA} = \emptyset$, or $\{s \in S \mid (FA)_s \neq \emptyset\}$ is not connected. Now, as FA is an object of \mathbf{Conn} , if $\overline{FA} \neq \emptyset$ then \overline{FA} contains two distinct elements. In either case we can conclude that $\mathbf{Ext}(A, FA) = \emptyset \neq \mathbf{Conn}(FA, FA)$.

If $\mathbf{upconn}(X)$ is empty, we can find $s, t \in X$ lying in distinct connected components of S . As FA is an object of \mathbf{Conn} , $(FA)_s \cap (FA)_t = \emptyset$. Thus there

can be no extensible morphism from A to FA , so again we have $\mathbf{Ext}(A, FA) \not\cong \mathbf{Conn}(FA, FA)$. \square

The next result is relevant to order-sorted unification. Viewing S as a category, $X \subseteq S$ determines a diagram in this category and a limit for this diagram is exactly a meet for X . Next we see how X gives rise to a diagram in \mathbf{Least} . Regarding X as a category and X^{op} being its opposite, the functor $X^* : X^{op} \rightarrow \mathbf{Least}$ is defined by making $X^*(x)$ the S -sorted set having just the element x having least sort x .

A cocone on X^* is (up to isomorphism) a set containing a lower bound for each connected component of X . A colimit for X^* is a set consisting of a meet for each connected component of X .

Theorem 5 *The inclusion $\mathbf{Least} \hookrightarrow \mathbf{Conn}$ has a left adjoint if and only if every connected subset of S has a meet.*

Proof Suppose S has connected meets. Define $\Delta : \mathbf{Conn} \rightarrow \mathbf{Least}$ by $\overline{\Delta A} = \overline{A}$ and $\text{sorts}_{\Delta A}(a) = (\bigwedge \text{sorts}_A(a))^\dagger$. It is straightforward to check that Δ has the appropriate properties.

Conversely, if there is a left adjoint then \mathbf{Least} has colimits since \mathbf{Set}^S has colimits and these are preserved by the left adjoints defined earlier. But then for any connected set X we have a colimit for X^* and hence a meet for X . \square

In [MGS89] conditions are given under which an order-sorted signature has a minimal set of unifiers. In the single sorted case there is a well known [RS87] description of a most general unifier as a coequalizer. In [Ste92] I showed how one form of order-sorted unification could be described as a generalization of the notion of coequalizer. Theorem 5 leads to conditions under which \mathbf{Least} may lack coequalizers, and with further work, should provide an account of the generalizations of coequalizers which describe minimal sets of order-sorted unifiers.

4.2 Description as Lax Slice Categories

The categories \mathbf{Ext} , \mathbf{Conn} , and \mathbf{Least} all admit alternative descriptions as lax slice categories. If (R, \leq) is any poset, then we can construct the lax slice category $\mathbf{Set} // R$. An object of this category consists of a set, X , and a function $\alpha : X \rightarrow R$. Such an object will be denoted X_α . A morphism $f : X_\alpha \rightarrow Y_\beta$ is a function $f : X \rightarrow Y$ such that $\beta f x \leq \alpha x$ for all $x \in X$. The three categories just mentioned are all (up to isomorphism) categories of this form for suitable choice of the poset R .

It is straightforward to verify that there is an isomorphism $\mathbf{Least} \rightarrow \mathbf{Set} // S$, which sends a least-sorts S -sorted set, A , to \overline{A}_α where α takes each $a \in \overline{A}$ to its least sort. To describe \mathbf{Ext} as a lax slice category, note that the set of sorts of an arbitrary S -sorted set, A , will be a non-empty upper set of S . Let $\mathcal{U}^+ S$ denote the poset of non-empty upper sets of S ordered oppositely to inclusion. Then

there is an isomorphism of categories $\mathbf{Ext} \cong \mathbf{Set} // \mathcal{U}^+ S$. Using \mathcal{CS} to denote the subposet of $\mathcal{U}^+ S$ determined by the connected non-empty upper sets, we obtain the isomorphism $\mathbf{Conn} \cong \mathbf{Set} // \mathcal{CS}$.

Note that the poset of all upper sets of S ordered oppositely to inclusion is actually the free meet semilattice on the poset S . Thus \mathbf{Ext} can be described as the category of least sort S^\wedge -sorted sets, where S^\wedge is the completion of S to allow for non-empty meets of sorts. There are also some interesting connections with sheaves and topology. Recall that a poset S can be given the Alexandroff topology by taking the open sets to be the upper sets of S . The functor category \mathbf{Set}^S is known [Gol84, p366] to be the category of sheaves over this topological space. The full picture of how this relates to the usual connection between sheaves and bundles needs some further work.

5 The Connection with Partial Algebras

This section shows how extensible algebras are a kind of partial algebra. Some basic facts about the category of sets and partial functions which we need are reviewed.

5.1 Extensible Algebras as Partial Algebras

An order sorted signature Σ can be forgotten to an unsorted signature $|\Sigma|$. An extensible Σ -algebra, A , can be forgotten to a partial $|\Sigma|$ -algebra with carrier $\bar{A} = \bigcup_{s \in S} A_s$. From this viewpoint, the elements of S serve to name subsets of \bar{A} , and the partial order in S specifies certain inclusions between these named subsets. The extensible Σ -algebras are thus partial $|\Sigma|$ -algebras where the domains of the operations are related by certain specified inclusions.

5.2 The Category of Sets and Partial Functions

Recall that an ordered category is a category enriched in the category of posets and monotone maps. Thus the morphisms between any two objects form a poset, and composition of morphisms respects this order. A key example of an ordered category is \mathbf{PFn} , the category of sets and partial functions. Given partial functions f and g from X to Y we put $f \leq g$ iff for all $x \in X$ if $f(x)$ is defined then $g(x)$ is defined, and $f(x) = g(x)$. In diagrams it is common to indicate that $f \leq g$ by drawing an arrow \Rightarrow from f to g . In \mathbf{PFn} a subset, $A \subseteq X$ is just a morphism $A : X \rightarrow X$ such that $A \leq \text{id}_X$.

The next observation plays an important role in expressing the sorting information in an order-sorted signature in a categorical way. The condition that f be a partial function defined on a superset of A and producing results in B from arguments in A is equivalent to the existence of a 2-cell as in the following diagram, where $\{*\}$ is a one-element set, and the morphisms to this set labelled

* are total functions.

$$\begin{array}{ccc}
 X & \xrightarrow{f} & Y \\
 A \downarrow & \Rightarrow & \downarrow B \\
 X & \xrightarrow{*} \{*\} \xleftarrow{*} & Y
 \end{array}$$

5.3 Example of Theory as Category

This simple example explains the basic idea behind the construction of the the next section. The aim is that given a signature Σ , we should be able to construct a category \mathbb{T} . The category should have certain structure, and functors from \mathbb{T} to \mathbf{PFn} which preserve the structure should be essentially the same as extensible algebras for Σ . This category is sometimes called the **classifying category** of Σ [Cro93].

Suppose we have sorts $B \leq A$, and a function symbol f with sortings $f : A \times A \rightarrow A$, and $f : B \times B \rightarrow B$. It appears we need some form of ordered category, which includes the following diagrams.

$$\begin{array}{ccc}
 & & 1 \times 1 \xrightarrow{!} 0 \xleftarrow{!} 1 \\
 & & \uparrow A \times A \Rightarrow A \uparrow \\
 \begin{array}{ccc}
 \xrightarrow{\text{id}_1} & & \\
 \xrightarrow{A} \uparrow & & \\
 \xrightarrow{B} \uparrow & & \\
 \xrightarrow{\quad} & &
 \end{array} & & 1 \times 1 \xrightarrow{f} 1 \\
 & & \downarrow B \times B \Rightarrow B \downarrow \\
 & & 1 \times 1 \xrightarrow{!} 0 \xleftarrow{!} 1
 \end{array}$$

If we consider what structure the category should support, it is clear that, unlike the single sorted case, 0 cannot be a terminal object and \times cannot be a product. The category \mathbf{PFn} does have a terminal object, but it is the empty set. There are also products in \mathbf{PFn} , but again these are not the structure we need. The product of A and B in \mathbf{PFn} is actually $A + (A \times B) + B$, where \times and $+$ are the product and coproduct respectively in \mathbf{Set} .

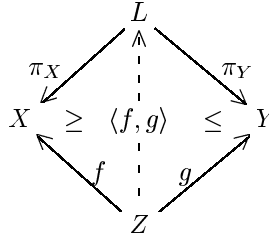
6 The Classifying Category of an Order-Sorted Theory

6.1 Ordered Categories with Total Local Products

We shall see that the appropriate structure on an extensible order-sorted theory *qua* category is that of an ordered category with total local products. In this section we review the definitions, which are due to Jay [Jay90]. The definitions assume the context of an ordered category \mathbb{O} .

Definition 2 A morphism $\delta : X \rightarrow X$ in \mathbb{O} is a **deflation** if $\delta^2 = \delta \leq \text{id}_X$. A morphism $f : Y \rightarrow Z$ is **total** if for every $g : X \rightarrow Y$, and for every deflation $\delta : X \rightarrow X$, the equality $f g \delta = f g$ implies $g \delta = g$.

Definition 3 A *local product* of objects X, Y is an object L and morphisms, π_X, π_Y as in the diagram below. Furthermore, given any f, g as in the diagram, there is a morphism $\langle f, g \rangle$ which is maximal among all morphisms $k : Z \rightarrow L$ such that $\pi_X k \leq f$ and $\pi_Y k \leq g$.



Definition 4 An object L of \mathbb{O} is a *local terminal object* if for any object X there is a morphism $\ell : X \rightarrow L$, and $k : X \rightarrow L \Rightarrow k \leq \ell$.

These forms of local limit are very weak notions. In **PFn** both \emptyset and 1 are locally terminal, and any subset of $X \times Y$ is a local product. However, total local limits are unique up to isomorphism.

Definition 5 A local product is *total* if the projections are total and whenever f and g are total, then $\pi_X \langle f, g \rangle = f$ and $\pi_Y \langle f, g \rangle = g$. A local terminal object is *total* if $\ell : X \rightarrow L$ is always total.

In **PFn**, the total product is $X \times Y$ as in **Set** with $\langle f, g \rangle$ defined on the intersection of domains of definition of f and g . Any one element set is a total terminal object.

6.2 Construction of the Classifying Category from a Signature

In this section the construction of a category from a signature Σ will be considered. The use of presentations consisting of a signature together with equations will be dealt with in the following section.

Essentially, the category we seek will be the free ordered category with total local products, generated diagrams derived from Σ as in the example in section 5.3. However, an explicit syntactic construction can be given, just as for the single sorted and many-sorted cases [Cro93]. Although order-sorted algebra can be seen as an extension of many-sorted algebra, several aspects of the following construction are closer to the single-sorted than the many-sorted case.

In the single sorted case, the objects of the category are the natural numbers $0, 1, 2, \dots$, and a morphism $m \rightarrow n$ is an equivalence class of n -tuples of terms constructed from m variables. In the case of extensible order-sorted algebra the main complication arises from morphisms $m \rightarrow 0$. Since such morphisms are to be interpreted as partial functions, it is not the case that there is a unique morphism $m \rightarrow 0$ which can be represented by the 0-tuple of terms.

First we construct a signature $\tilde{\Sigma}$ which extends the unsorted signature $|\Sigma|$ by the addition of two kinds of function symbol. Each $f : s_1 \times \dots \times s_n \rightarrow s$ in

Σ , where $n \geq 0$, gives rise to $f : n \rightarrow 1$ in $\tilde{\Sigma}$. The signature $\tilde{\Sigma}$ has, function symbols which represent subsets. If S^\wedge denotes the completion of S to allow for meets of non-empty finite subsets of S , then $\tilde{\Sigma}$ has, for each $s \in S^\wedge$ a function symbol $s : 1 \rightarrow 1$.

The signature $\tilde{\Sigma}$ also contains for each $n \geq 0$, a function symbol $*^n : n \rightarrow 0$. The significance of the symbols $*^n$ is that when the category we are constructing is interpreted in **Pfn**, with 1 interpreted as the set H , the morphism $*^n : n \rightarrow 0$ will be interpreted as the unique total function from H^n to the one element set H^0 . Non-total functions to H^0 will arise by composition with $*^n$.

Terms in $\tilde{\Sigma}$ require variables of two types: 0 and 1. It is assumed that variables z, z_1, z_2, \dots , of type 0, and variables x, x_1, x_2, \dots , of type 1 are available. Other terms are formed according to the following rules. In these rules k will be a constant in Σ , and f will either be a function symbol in Σ (the case of $n = 1$), or $*^m$, in which case $n = 0$.

$$\frac{f : m \rightarrow n \quad t_i : 1 \quad i = 1, \dots, m \quad m > 0}{f(t_1, \dots, t_m) : n} \quad \frac{k : 0 \rightarrow 1 \quad t : 0}{k(t) : 1}$$

The appearance of terms which contain variables such as $k(z)$ when k is a constant in Σ may appear strange. This construction is necessary to cope with partially defined functions. For example, suppose that 1 is interpreted as the set H , the sort s is interpreted as $A \subseteq H$, and the constant k as $h \in H$. The term $k(*^1(s(x)))$ is interpreted as the function from H to H sending elements of A to h , and undefined outside A .

Substitution is defined in the usual way, though only terms of type i can be substituted for variables of i , where i can be 0 or 1.

Certain equations between terms are required. Any term containing two sub-terms $A(x_1)$ and $B(x_1)$ is equal to the term with both replaced by $A \wedge B(x_1)$, where $A \wedge B$ is the meet of A and B in S^\wedge . It is also necessary to have $A(B(x_1)) = A \wedge B(x_1)$, thus intersection of subsorts appears as composition in the category. In addition to these equalities, and substitution instances of them, certain equalities needed in the unsorted case have to be imposed. The reader should be able to supply the details by following the account of the unsorted case in [Cro93, section 3.8].

Using this equivalence, a morphism $m \rightarrow n$ is an equivalence class of expressions of the form $\Gamma; \mathbf{t}$, where Γ is a context, consisting of a list of variables, and \mathbf{t} is a list of terms. When $m = 0$, Γ will be a single variable of type 0, and when $m > 0$, Γ will be a list of m distinct variables of type 1. When $n = 0$, \mathbf{t} will be a single term of type 0, and when $n > 0$, \mathbf{t} will be an n -tuple of terms of type 1. In both cases, the terms in \mathbf{t} are built from the variables in Γ . Composition of morphisms is defined using substitution.

The order on the hom sets is generated by adding the sort constraints. For a function symbol $f : s_1 \times \dots \times s_n \rightarrow s$ in Σ we require

$$*^n(s_1(x_1), \dots, s_n(x_n)) \leq *^1(s(f(x_1, \dots, x_n))).$$

Note that this is just the general case of one of the diagrams in the example in section 5.3. For a constant $k : s$ in Σ we require $*^0(z) \leq *^1(s(k(z)))$. And for

each subsort relation $s_1 \leq s_2$ between elements of S^\wedge we require $s_1(x) \leq s_2(x)$. Further relations are generated by substitution.

The above construction yields an ordered category with total local products which is denoted $Cl(\Sigma)$. In $Cl(\Sigma)$ the object n is the n -fold total local product of 1 with itself.

6.3 Presentations including Equations

To introduce equations, we need to recall the construction of order-sorted terms from a signature Σ . Given Σ , and an S -sorted set of variables, X , we can construct an extensible algebra, having as carrier the S -sorted set of terms $T_\Sigma(X)$. Details can be found in [Gog78] and in [Poi90, p235]. An equation is then an expression of the form $[X]t =_s t'$ where $t, t' \in (T_\Sigma(X))_s$ for some $s \in S$, and every variable in t or t' occurs in X . An algebra A in \mathbf{AlgExt}_Σ is said to **satisfy** the equation $[X]t =_s t'$, if $h_s(t) = h_s(t')$ for every homomorphism $h : T_\Sigma(X) \rightarrow A$. The category of all extensible algebras satisfying every equation in a presentation P will be denoted \mathbf{AlgExt}_P .

A set of equations of this form can be represented by parallel pairs in the category $Cl(\Sigma)$. By constructing the quotient category we obtain the classifying category $Cl(P)$ for an order-sorted presentation P .

6.4 Extensible Models of $Cl(P)$

Given an order-sorted presentation $P = (\Sigma, E)$, we can construct its classifying category $Cl(P)$ as outlined above. In this section we define models of $Cl(P)$ as certain functors to \mathbf{PFn} , and compare the category of these models with the category \mathbf{AlgExt}_P .

Definition 6 *Let \mathbb{T} be an ordered category with total local products. An **extensible model** of \mathbb{T} is a lax ordered functor to \mathbf{PFn} which preserves the total local products. A **morphism of extensible models** is a lax natural transformation between the functors. The category of extensible models and morphisms will be denoted $\mathbf{ModExt}_\mathbb{T}$.*

An operation symbol can be interpreted in a model by a partial function which is defined more widely than it is required to be. This accounts for the weakened sense of equivalence in the next theorem. Possibly this result is weaker than expected, which might suggest that further work could find an improved notion of functorial model.

Theorem 6 *The categories $\mathbf{ModExt}_{Cl(P)}$ and \mathbf{AlgExt}_P are equivalent in the following weakened sense. There are functors $A : \mathbf{ModExt}_{Cl(P)} \rightarrow \mathbf{AlgExt}_P$, and $M : \mathbf{AlgExt}_P \rightarrow \mathbf{ModExt}_{Cl(P)}$ where AM is the identity on \mathbf{AlgExt}_P . The composite MA is related to the identity on $\mathbf{ModExt}_{Cl(P)}$ by natural transformations $\eta : I \rightarrow MA$ and $\theta : MA \rightarrow I$, where I is the identity functor on $\mathbf{ModExt}_{Cl(P)}$. These transformations satisfy the properties that $\eta\theta = id_{MA}$, and $\theta\eta \leq id_I$, where id_I denotes the identity natural transformation on the functor I .*

7 Conclusions and Further Work

This paper has advocated, and presented some initial results in, a new research programme for understanding the various forms of order-sorted algebra. Specifically, an analysis of the various categories of order-sorted sets has been presented, and an enriched form of algebraic theory to model extensible order-sorted algebra has been identified.

Much more work is required to complete the programme. The next step should be a treatment of compatible algebras along the lines of that for extensible algebras. Another area for investigation is a categorical treatment of order-sorted unification and rewriting which extends the single-sorted case developed in [RS87].

Acknowledgements

I am grateful to Joseph Goguen for his interest in this work and his suggestions. The anonymous referees also made useful comments. The categorical diagrams have been drawn using Paul Taylor's macros.

References

- [Acz91] P. Aczel. Term declaration logic and generalized composita. In *6th Annual IEEE Symposium on Logic in Computer Science, Amsterdam, July 1991*, pages 22–30. IEEE Computer Society, 1991.
- [AG94] A. J. Alencar and J. A. Goguen. Specification in OOZE with examples. In K. Lano and H. Houghton, editors, *Object-Oriented Specification Case Studies*, chapter 8, pages 158–183. Prentice-Hall, 1994.
- [AR94] J. Adámek and J. Rosický. *Locally Presentable and Accessible Categories*, volume 189 of *London Mathematical Society Lecture Notes Series*. Cambridge University Press, 1994.
- [BJM00] A. Bouhoula, J.-P. Jouannaud, and J. Meseguer. Specification and proof in membership equational logic. *Theoretical Computer Science*, 236:35–132, 2000.
- [BM94] C. Beierle and G. Meyer. Run-time type computations in the Warren abstract machine. *Journal of Logic Programming*, 18(2):123–148, 1994.
- [Cro93] R. L. Crole. *Categories for Types*. Cambridge University Press, 1993.
- [EG94] M. Erwig and R. H. Güting. Explicit graphs in a functional model for spatial databases. *IEEE Transactions on Knowledge and Data Engineering*, 6(5):787–804, 1994.
- [GD94] J. A. Goguen and R. Diaconescu. An Oxford survey of order-sorted algebra. *Mathematical Structures in Computer Science*, 4:363–392, 1994.
- [GM92] J. A. Goguen and J. Meseguer. Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105:217–273, 1992.
- [Gog78] J. A. Goguen. Order sorted algebras: Exceptions and error sorts, coercions and overloaded operators. Semantics and Theory of Computation Report 14, University of California at Los Angeles. Computer Science Department, December 1978.

- [Gol84] R. Goldblatt. *Topoi, The Categorical Analysis of Logic*, volume 98 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, revised edition, 1984. First edition 1979.
- [Hax95] A. E. Haxthausen. Order-sorted algebraic specifications with higher-order functions. In V. S. Alagar and M. Nivat, editors, *Algebraic Methodology and Software Technology, 4th International Conference, AMAST '95, Montreal, July 1995*, volume 936 of *Lecture Notes in Computer Science*, pages 133–151. Springer-Verlag, 1995.
- [HS95] A. C. Hearn and E. Schrufer. A computer algebra system based on order-sorted algebra. *Journal of Symbolic Computation*, 19(1–3):65–77, 1995.
- [Jay90] C. B. Jay. Extending properties to categories of partial maps. Technical Report ECS-LFCS-90-107, Laboratory for Foundations of Computer Science, University of Edinburgh, February 1990.
- [Kea75] O. Keane. Abstract Horn theories. In F. W. Lawvere, C. Maurer, and G. C. Wraith, editors, *Model Theory and Topoi*, volume 445 of *Lecture Notes in Mathematics*, pages 15–50. Springer-Verlag, 1975.
- [MGS89] J. Meseguer, J. A. Goguen, and G. Smolka. Order sorted unification. *Journal of Symbolic Computation*, 8:383–413, 1989.
- [MT93] K. Meinke and J. V. Tucker, editors. *Many-sorted Logic and its Applications*. Wiley, 1993.
- [Obe62] A. Oberschelp. Untersuchungen zur Mehrsortigen Quantoren Logik. *Mathematische Annalen*, 145:297–333, 1962.
- [Poi90] A. Poigné. Parameterization for order-sorted algebraic specification. *Journal of Computer and System Sciences*, 40:229–268, 1990.
- [Qia94] Z. Qian. Another look at parameterization for order-sorted algebraic specifications. *Journal of Computer and System Sciences*, 49:620–666, 1994.
- [Rob94] E. Robinson. Variations on algebra: monadicity and generalisations of equational theories. Technical Report 6/94, School of Cognitive and Computing Sciences, University of Sussex, April 1994.
- [RS87] D. E. Rydeheard and J. G. Stell. Foundations of equational deduction: A categorical treatment of equational proofs and unification algorithms. In D. H. Pitt et al., editors, *Category Theory and Computer Science, Edinburgh, 1987*, volume 283 of *Lecture Notes in Computer Science*, pages 114–139. Springer-Verlag, 1987.
- [SS89] M. Schmidt-Schauß. *Computational Aspects of an Order-Sorted Logic with Term Declarations*, volume 395 of *Lecture notes in Artificial Intelligence*. Springer-Verlag, 1989.
- [Ste91] J. G. Stell. Unique-sort order-sorted theories – a description as monad morphisms. In S. Kaplan and M. Okada, editors, *Proceedings of 2nd International Workshop on Conditional and Typed Rewriting Systems, Montreal, June 1990*, volume 516 of *Lecture Notes in Computer Science*, pages 389–400. Springer-Verlag, 1991.
- [Ste92] J. G. Stell. *Categorical Aspects of Unification and Rewriting*. PhD thesis, University of Manchester, 1992.
- [Wal92] U. Waldmann. Semantics of order-sorted specifications. *Theoretical Computer Science*, 94:1–35, 1992.
- [Wie91] R. J. Wieringa. A formalization of objects using equational dynamic logic. In C. Delobel, M. Kifer, and Y. Masunaga, editors, *Deductive and Object-Oriented Databases. Second International Conference, DOOD '91*, volume 566 of *Lecture Notes in Computer Science*, pages 431–452. Springer-Verlag, 1991.