Membership Structures

Roger Bishop Jones

Date: 2007/06/11 20:48:54

Abstract

A exploration of a way of doing set theory in HOL without using axioms. I thought that locales in Isabelle-HOL would make this kind of approach workable, and this document is the result of my investigating this hope. However, it quickly became apparent that the limitations on the implementation of locales. particularly in relation to the forms of definition which you can use in a locale, were too severe. Consequently I don't get very far. I had hoped to do the construction of the PolySets on this kind of set theory, instead of using an axiomatisation of set theory in HOL, but I reverted to the latter course.

http://www.rbjones.com/rbjpub/isar/HOL/Membership.pdf
http://www.rbjones.com/rbjpub/isar/HOL/Membership.tgz

Id: root.tex,v 1.5 2007/06/11 20:48:54 rbj01 Exp

Contents

1 Introduction

This document is concerned with what is usually known as “set theory”. The approach adopted is intended to be distinguished from usual treatmente of set theory in the following ways:

- The subject matter is taken to be *membership structures* rather than sets. A membership structure is, of course, some collection together with a binary relation over that collection, i.e. an interpretation of the first order language of set theory.
- No specific set of axioms is adopted. Instead the interrelationships between various properties of membership structures are studied.
- The distinction is drawn between those properties of membership structures which are expressible as sentences in the language of first order set theory (“first-order” for short), and those which are not (“higher order”).
- Particular interest is shown in certain properties which are not first order and in the kinds of membership structure posessing these properties.
- The informal language of set theory is made more precise by noting that certain properties (notably cardinality) normally thought of as properties of sets, are perhaps better seen as relationships between membership structures and their elements (in what may be called their internal use) unless an external criterion of equipollence is applied

(yielding an "external" notion of cardinality). Some theorems of set theory are explicated using this distinction, for example the result that V=L is incompatible with the existence of measurable cardinals.

The theory is substantially but not exclusively concerned with well-founded membership structures, in which aspect its motivation is largely philosophical. In practical terms, the main application of the well-founded structures is to the construction of non-well-founded structures which are to be used elsewhere.

I regard the use of locales in this to have been a failure and the set theory has not been progressed. I have added material on Boolean Valued Models, motivated by different considerations.

2 Membership Structures

theory *MembershipStructures*
imports *Main*
begin

[1]

First, type abbreviations are introduced for membership structures (i.e. interpretations of a set theory) and for properties of membership structures (the kind of thing expressed by an axiom of set theory).

Normally a membership structure is considered as a set with a binary relation. However, there is in most cases some redundancy in this, since the set is the field of the relatioin, and can be recovered from it. In informal mathematics this is of no consequence, but in formal mathematics redundancy causes clutter. The cases where the set is not the same as the field of the relationship are sufficiently strange that, (like the possibility of an empty domain in a first order interpretation) it is unlikely to be of interest in practice and it is worthwhile to exclude it.

I therefore begin with a notion of membership structure in which there may be no isolated elements and which therefore can be represented simply by the relation, and this theory therefore begins as an extension to the theory of relations.

types
  $'a\ MS = ('a \times 'a)set$
  $'a\ PMS = 'a\ MS \Rightarrow bool$

2.1 Extensionality

We will normally be working with extensional relationships. In that case the extension of a set is usually all we need to know about it, and an extension

[1]Id: MembershipStructures.thy,v 1.3 2006/11/13 07:12:37 rbj01 Exp

suffices to identify a set. We therefore define two functions which will normally provide an injection from sets into extensions, connecting sethood in some membership structure with Isabelle HOL sethood. The main purpose of this is to enable us to use the set theoretic language already available to us while talking about a different kind of set.

```
constdefs
  Ext :: 'a MS ⇒ 'a ⇒ ('a set)
  Ext r x == {y . (y,x):r}

  Co :: 'a MS ⇒ ('a set) ⇒ 'a
  Co r s == (THE x. x:Field r ∧ Ext r x = s)
```

My concern here is exclusively with extensional concepts of set, extensionality being considered in some circles the quintessence of sethood. There are non-extensional set theories but they will not be investigated here,

Though our representation for membership structures excludes isolated elements, it does not rule out urelements, but a full axiom of extensionality would rule out urelements with no elements (though not "Quine atoms" (and possibly other schemes). Though in principle NFU could be done within this framework, in practice this would be awkward, and it would complicate the development unduly to organised the work in a manner sympathetic to NFU. I therefore propose to confine this study to fully extensional structures.

Full extensionality is defined as follows.

```
constdefs
   FullExt :: 'a PMS
     FullExt ms == ALL x y. x:(Field ms) ∧ y:(Field ms)
     ⟶ (ALL z. (z, x):ms = ((z,y):ms)) ⟶ x = y


lemma FullExt1:
   ⟦FullExt ms; x:(Field ms); y:(Field ms); (ALL z. (z, x):ms = ((z,y):ms))⟧
    ⟹ x = y
⟨proof⟩

locale Ms =

fixes
    ms :: ('a * 'a)set

locale ExtMs = Ms +

fixes
    X  :: 'a ⇒ 'a set
and C  :: 'a set ⇒ 'a

assumes
  FullExt: FullExt ms
```

defines *X-def*: $X == Ext\ ms$
and *C-def*: $C == Co\ ms$

lemma (**in** *ExtMs*) *ExtMs1* :
$[\![x{:}(Field\ ms);\ y{:}(Field\ ms);\ (ALL\ z.\ (z,\ x){:}ms = ((z,y){:}ms))]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *ExtMs*) *ExCo-inv* :
assumes x : *Field ms*
shows $Co\ ms\ (Ext\ ms\ x) = x$
⟨*proof*⟩

lemma (**in** *ExtMs*) *XC-inv* :
assumes x : *Field ms*
shows $C\ (X\ x) = x$
⟨*proof*⟩

2.2 Well-Foundedness

locale *WfMs* = *Ms* +
assumes *wf* : *wf ms*

locale *WfExtMs* = *ExtMs* + *WfMs*

end

3 Filters

theory *Filter*
imports *MembershipStructures*
begin

The theory *Filters* [2] is a partial replica of the material on filters ideals and ultrafilters from Chapter 7 of Thomas Jech's *Set Theory* [1].

This work is undertaken because it has occurred to me that the right kind of interpretation of first order set theory (for my purposes) might possibly be obtainable via "Boolean Valued Models". At present I am simply attempting to formalise the relevant material from Jech in order to get a better understanding of these things in the hope that I will then have a clearer idea of whether and how they will help.

This material belongs in a document on membership structures, because it is primarily concerned with (a rather different kind of) membership structure.

It would is possible to recover the domain of a filter from the set of sets in the filter (it is the union of them), but it is not possible for ideals. It looks

[2]Id: Filter.thy,v 1.1 2006/12/11 12:14:51 rbj01 Exp

like Jech is mentioning ideals *en passant*, the subsequent development seems to be all filters. There are lots of other kinds of ideals about, but maybe not much use of this kind of ideal. Therefore I propose to do the filters without carrying a separate domain and ignore the ideals for now.

Furthermore, this will be formalisation on demand. demand in this case coming from the formalisation of Boolean Valued Models.

constdefs
$\quad$ *filtr*:: *'a set set* $\Rightarrow$ *bool*
$$\begin{aligned}&\mathit{filtr}\ f == \bigcup f \in f \wedge \{\} \notin f\\ &\quad \wedge (\forall x\ y.\ x \in f \wedge y \in f \longrightarrow x \cap y \in f)\\ &\quad \wedge (\forall x\ y.\ x \subseteq \bigcup f \wedge y \subseteq \bigcup f \wedge x \in f \wedge x \subseteq y \longrightarrow y \in f)\end{aligned}$$

lemma *ex1*: $s \neq \{\} \Longrightarrow \mathit{filtr}\ \{s\}$
⟨*proof*⟩

constdefs
$\quad$ *fip*:: *'a set set* $\Rightarrow$ *bool*
$$\mathit{fip}\ G == \forall\ H.\ H \subseteq G \wedge \mathit{finite}\ H \longrightarrow \bigcap H \neq \{\}$$

lemma *lemma7-2i*:
$$(F \neq \{\} \wedge (\forall f.\ f \in F \longrightarrow \bigcup f = S \wedge \mathit{filtr}\ f))$$
$$\Longrightarrow \mathit{filtr}(\bigcap F) \wedge \bigcup(\bigcap F){=}S$$
⟨*proof*⟩

lemma *l7-2i1*: $(\forall f.\ f \in F \longrightarrow \bigcup f = S \wedge \bigcup f \in f) \Longrightarrow S \in \bigcap F$
⟨*proof*⟩

lemma *l7-2i1*: $(\forall f.\ f \in F \longrightarrow \bigcup f = S \wedge \bigcup f \in f) \Longrightarrow \bigcup\bigcap F = S$
⟨*proof*⟩

end

4 PolySets

theory *PolySets*
imports *MembershipStructures*
begin

The theory *PolySets* [3] first provides a construction of a model of a set theory with a universal set (a membership relation) by construction from some other membership relation and then proceeds to establish properties of the model suitable for use in a higher order axiomatisation.

My first thought was to establish, in the theory *MembershipStructures*,

[3]Id: PolySets.thy,v 1.2 2006/11/13 07:12:37 rbj01 Exp

locales for well-founded extensional membership structures with sufficient other properties to yield under the construction a domain with the required properties. However, I find that the locales are not as helpful as I hoped. The assumption of well-foundedness does not facilitate the use of transfinite induction in the construction, since the definitional facilities available in locales are too limited.

The construction is therefore defined relative to an arbitrary membership structure, and the properties of the original membership structure are considered only after the construction when they are required to deduce desirable properties of the resulting model.

The first stage is to construct the PolySet ordinals for which purpose we need an ordered pair constructor.

4.1 Pairs

Pairs and ordered pairs are defined as relationships over an arbitrary binary relationship. This does not mean that a representation is chosen which works in any relation. The representation is the usual Wiener-Kuratovski construction $(x, y) = \{x, \{x, y\}\}$, and the relationship we define tells us whether one element is an ordered pair of two others, without prejudging or depending whether such pairs in fact exist in any given relationship.

constdefs
 $Wkp :: 'a\ MS \Rightarrow ('a * 'a) \Rightarrow 'a \Rightarrow bool$
  $Wkp\ ms == \lambda(x,y)\ z.\ \exists\ v.\ Ext\ ms\ v = \{x,y\} \wedge Ext\ ms\ z = \{x,v\}$

4.2 PolySet Ordinals

There is an injection from the well-founded sets into the PolySets. The set we are about to define is the image under that injection of the Von Neumann ordinals (the injection is not itself defined).

This is done by defining the successor function, and then the intersection of the sets which contain the empty set and are closed under taking the strict supremum of a set of ordinals.

The set defined will be well-ordered by PolySet membership (to be defined) if there is exactly one zero in the original relationship. Otherwise it will either be the empty set or the universal set.

constdefs
 $Zero :: 'a\ MS \Rightarrow 'a \Rightarrow bool$
  $Zero\ ms\ x == x \in Field\ ms \wedge Ext\ ms\ x = \{\}$
 $Succ :: 'a\ MS \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
  $Succ\ ms\ x\ y == \exists\ v\ w.\ Wkp\ ms\ (v,w)\ y \wedge Zero\ ms\ v \wedge Ext\ ms\ w = (Ext\ ms\ x) \cup \{x\}$
 $UnionSuccs :: 'a\ MS \Rightarrow 'a\ set \Rightarrow 'a \Rightarrow bool$

$\quad\quad UnionSuccs\ ms\ s\ t == Ext\ ms\ t = \{u\ .\ (\exists\ \ v\ w.\ v \in s \wedge (w,v) \in ms \wedge Succ\ ms\ w\ u)\}$

consts $PolySetOrds :: 'a\ MS \Rightarrow 'a\ set$
inductive $PolySetOrds\ ms$
intros
$\quad zPsoI : Zero\ ms\ z \Longrightarrow z{:}\ PolySetOrds\ ms$
$\quad PsoI : (\forall\, psos.\ UnionSuccs\ ms\ psos\ pso$
$\quad\quad\quad\quad \wedge (\forall\, y.\ y \in psos \longrightarrow y{:}\ PolySetOrds\ ms))$
$\quad\quad\quad \Longrightarrow pso{:}\ PolySetOrds\ ms$

constdefs
$\quad RelRestr :: 'a\ MS \Rightarrow 'a\ set \Rightarrow 'a\ MS$
$\quad\quad\quad RelRestr\ ms\ s == \{(x,y)\ .\ x \in s \wedge y \in s \wedge (x,y) \in ms\}$
$\quad PolySetOrdMs :: 'a\ MS \Rightarrow 'a\ MS$
$\quad\quad\quad PolySetOrdMs\ ms == RelRestr\ ms\ (PolySetOrds\ ms)$

4.3 PolySets

The PolySets are now definable inductively.

constdefs
$\quad MakePolySet{::}\ 'a\ MS \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
$\quad\quad\quad MakePolySet\ ms\ ord\ pss\ ps == ord \in (PolySetOrds\ ms) \wedge Wkp\ ms\ (ord, pss)\ ps$

consts
$\quad PolySets :: 'a\ MS \Rightarrow 'a\ set$
inductive $PolySets\ ms$
intros
$\quad zPsI{:}\ Zero\ ms\ z \Longrightarrow z : PolySets\ ms$
$\quad PsI{:}\ (\forall\ \ t\ ord\ u.$
$\quad\quad\quad\quad ord \in (PolySetOrds\ ms)$
$\quad\quad\quad \wedge (Ext\ ms\ u = t)$
$\quad\quad\quad \wedge\ MakePolySet\ ms\ ord\ u\ v$
$\quad\quad\quad \wedge (\forall\, w.\ w \in t \longrightarrow w : PolySets\ ms))$
$\quad\quad\quad \Longrightarrow v \in PolySets\ ms$

4.4 Membership

The PolySets are patterns, in which the ordinals are instantiable pattern variables. The key element in defining the membership relation over the PolySets is the specification of when a PolySet conforms to a pattern. It does do of course, when the pattern can be instantiated according to some assignment of values to the pattern variables, to give the required PolySet. I therefore begin with the definition of this process of instantiation, after which the definition of membership will be straightforward,

4.4.1 Pattern Instantiation

Pattern instantiation is defined by transfinite recursion over the PolySets. The well-founded relation with respect to which the recursion is well-founded is not the intended membership relation over the PolySets, but the presumed membership relation from which the PolySets are constructed. Though this need not itself be well-founded, its restriction to the PolySets is.

The function we require here is total over the intended domains, which are subsets of unspecified types. Our logic is a logic of total functions, and so we have to define instantiation not as a function but as a relation (which will be many one over the target domain).

A PolySet valuation for an ordinal in the PolySets derived from some membership structure is a (HOL, total) function which maps the PolySet ordinals less than the given ordinal to PolySets. It doesn't matter what it does elsewhere.

constdefs
  PsOrdLt :: *'a MS* ⇒ *'a* ⇒ *'a* ⇒ *bool*
    PsOrdLt ms o1 o2 ==
        o1 ∈ *PolySetOrds ms*
      ∧ *o2* ∈ *PolySetOrds ms*
      ∧ (∃ *so z*. *Zero ms z*
      ∧ (*o1*, *so*) ∈ *ms*
      ∧ *MakePolySet ms z so o2*)

  isPsValuation :: *'a MS* ⇒ *'a* ⇒ (*'a* ⇒ *'a*) ⇒ *bool*
    isPsValuation ms ord va ==
        ord ∈ *PolySetOrds ms*
      ∧ (∀ *o2*. *PsOrdLt ms o2 ord* ⟶ (*va o2*) ∈ *PolySets ms*)

consts
  PolySetInstantiate :: *'a MS* ⇒ *'a* ⇒ (*'a* ⇒ *'a*) ⇒ (*'a* ∗ *'a*) *set*
  PsOrdAdd :: *'a MS* ⇒ (*'a* ∗ *'a*) ⇒ *'a* ⇒ *bool*

inductive
  PolySetInstantiate ms ord va
intros
  zPsInI : *Zero ms z*
        ⟹ (*ord*, *z*) : *PolySetInstantiate ms ord va*
  vPsInI : *PsOrdLt ms o2 ord*
        ⟹ (*o2*, *va o2*) : *PolySetInstantiate ms ord va*
  oPsInI : *o1* ∈ *PolySetOrds ms*
        ∧ ¬ *PsOrdLt ms o1 ord*
        ∧ *PsOrdAdd ms* (*ord*,*o2*) *o1*
        ⟹ (*o1*, *o2*) : *PolySetInstantiate ms ord va*

```
  sPsInI : "\<not> s \<in> PolySetOrds ms
```

```
\<and> MakePolySet ms or sps1 s
\<and> MakePolySet ms or sps2 t
\<and> (\<forall>ps2. (ps2,sps2) \<in> ms
  = (\<exists>ps1. (ps1,sps1) \<in> ms \<and> (ps1, ps2) \<in> PolySetI
\<Longrightarrow> (s,t): PolySetInstantiate ms ord va"
```

end

References

[1] Thomas Jech, *Set Theory*, The Third Millenium Edition, Springer 2002