プログラミング－言語・基礎・実践－ 5－18
(1991. 11. 22)

OBJ のサブソートの Martin-Löf の部分集合論における解釈
Interpreting OBJ subsorting in Martin-Löf's subset theory with a universe

木下佳樹
Kinoshita Yoshiki
電子技術総合研究所
Electrotechnical Laboratory

Nov. 22, 1991

Abstract

OBJ の重要な特徴の一つであるサブソートを、ユニバースを持つ Martin-Löf の部分集合論 $\mathbf{ITT}_1^{sub}$ によって意味づけする。このため、OBJ の部分言語で、サブソートについての機能を完全に継承している言語 OBJ-S を定義し、OBJ-S の抽象構文木を $\mathbf{ITT}_1^{sub}$ の項に移す写像を定義する。この写像による行き先の項の $\mathbf{ITT}_1^{sub}$ における操作的および論理的意味は、もとの抽象構文木の直感的意味を反映している。

We intereprete subsorting, one of the important features of OBJ, in Martin-Löf's subset theory $\mathbf{ITT}_1^{sub}$ with a universe. To that end, we set up a sublanguage OBJ-S of OBJ, which inherits the subsorting mechanism of OBJ, and whose intuitive meaning should be understood by the intuitive meaning of OBJ. Then, we define a mapping which takes an abstract syntax tree of OBJ-S to a term in Martin-Löf's subset theory with a universe, whose operational and logical meaning reflects the intuitive meaning of OBJ-S.

Id is a set of identifiers.
Obj ::= `obj` *Id* `is` *Body* `endo` |
`obj` *Id* `[` *Id* `::` *E* `]` `is` *Body* `endo` |
Theory ::= `th` *Id* `is` *Body* `endth` |
`th` *Id* `[` *Id* `::` *E* `]` `is` *Body* `endth` |
View ::=`view` *Id* `from` *E* `to` *E* `is` *Vbody* `endv`
Body ::= *EmptyBody* |
`using` *E* `.` *Body* |
`sort` *Id* `.` *Body* |
`subsort` *Id* `<` *Id* `.` *Body* |
`op` *Id* `:` *SortList* `->` *Sort* `.` *Body* |
`var` *Id* `:` *Sort* `.` *Body* |
`eq` *Term* `=` *Term* `.` *Body*
E ::=*Id* | *Id* `[` *View* `]`
Sort ::=*Id* | *Id* `.` *E*
SortList ::=*EmptySortList* | *Sort SortList*
Term ::=*Id* | *Id* `(` *TermSeq* `)` | *Term* `.` *E*
TermSeq ::=*Term* | *Term* `,` *TermSeq*
VBody ::=*EmptyVBody* |
VBody `.` `sort` *Id* `to` *Sort* |
VBody `.` `op` *Id* `to` *Term* |

Table 1: Abstract syntax of OBJ-S

1 Introduction

In [2], we showed that the intuitive meaning of parameterized modules in OBJ can be given in Martin-Löf's polymorphic set theory with a universe[1]. Here we further show that the subsorting mechanism can naturally be interpreted in Martin-Löf's subset theory with a universe, which is also explained in [1], whose notation we will closely follow hereafter, thus avoiding duplicating the explanation of the Martin-öf's theory.

In the next section, we define the abstract syntax of OBJ-S, which only extends the abstract syntax of OBJ-P in [2] by adding `subsort` declaration. In section **4**, we define the mapping taking the OBJ-S abstract syntax tree to terms in Martin-Löf's subset theory with a universe. Semantics given here to OBJ-S is a type theoretical semantics which is explained briefly in [2].

2 Abstract syntax of OBJ-S

The abstract syntax of OBJ-S is defined in **Table 1**. It is exactly the same as the abstract syntax of OBJ-P except that OBJ-S has `subsort` declaration.

We use italic identifiers beginning with upper case letter for names of phylums (syntactic classes). For variables ranging over abstract syntax trees of a phylum, we use identifiers made by converting all letters of the name of the phylum to lower case. Thus, we use *e*, *id*, *theory* for variables ranging over AST's of phylums *E*, *Id*, *Theory*, respectively.

3 Subsort and subset

A set A in $\mathbf{ITT}_1^{sub}$ is interpreted as an informal pair (A', A''), where A' is a set in $\mathbf{ITT}_1$ and A'' is a predicate in $\mathbf{ITT}_1$ over A'. Then the fact in $\mathbf{ITT}_1^{sub}$ that A is a subset of B is expressed by two judgements in $\mathbf{ITT}_1$: $A' = B'$ and $(\Pi x \in A')\, A''(x) \supset B''(x)$. A' is called the *base set* of A and A'' is called the *characterization predicate* of A.

One of the problem in using $\mathbf{ITT}_1^{sub}$ to interprete subsorting of OBJ-S is that A' cannot be determined at the time of evaluating *Obj* syntax tree; even if A is not a subsort of any other sort at the time of evaluation, it might become a subsort of a sort which will be declared in the future. For example, consider the following *Obj* X:

```
obj X is
  sort A .
  op e: -> A .
endo
```

When this *Obj* is evaluated, `A` is a subsort of no other sorts. Now, assume the following *Obj* $XLIST$ is evaluated afterwards:

```
obj XLIST  is
  using X .
  sort List .  sort NeList .
  subsort A < NeList .  subsort NeList < List .
  op nil: -> List .
  op append: List List -> List .
  var x: List .  var y: List .  var z: List .
  eq append(append(x, y), z) = append(x, append(y, z)) .
  eq append(nil, x) = x .
  eq append(x, nil) = x .
endo
```

Then, the sort `A.X` suddenly becomes a subsort of `List.XLIST`. At the first glance, this may seem to indicate that $[\![\texttt{A.X}]\!]'$ cannot be defined at the time of evaluating X. But we can define $[\![\texttt{A.X}]\!]'$ at that time. Our solution is to add the judgement

$$[\![\texttt{A.X}]\!]' = [\![\texttt{List.XLIST}]\!]'$$

as well as the implication

$$(\Pi x \in [\![\texttt{A.X}]\!]')\, [\![\texttt{A.X}]\!]''(\texttt{x}) \supset [\![\texttt{List.XLIST}]\!]''(\texttt{x})$$

in the base set theory $\mathbf{ITT}_1$ at the time of evaluating $XLIST$. According to this method, the set of values (normal forms) of sort `A.X` is the singleton $\{\,\texttt{e}\,\}$ just after evaluating X, but evaluating Y changes it to

$$\{\,\texttt{e}, \texttt{nil}, \texttt{append(e, e)}, \texttt{append(e, e, e)}, \ldots\}$$

In our interpretation of OBJ-S in $\mathbf{ITT}_1^{sub}$, the equality between the base set is expressed as an equality set $\mathsf{Eq}(\mathsf{U}, [\![\texttt{A.X}]\!]', [\![\texttt{List.XLIST}]\!]')$ in $\mathbf{ITT}_1$. It is possible because all sorts have their coding in U. Specifically, an OBJ-S phrase

$$\texttt{subsortX < Y}$$

has the term

$$\mathsf{Eq}(\mathsf{U}, [\![X]\!]', [\![Y]\!]') \& (\Pi x \in [\![X]\!]')\, [\![X]\!]''(x) \supset [\![Y]\!]''(x)$$

in $\mathbf{ITT}_1$ as its the denotation.

4 Semantic functions

A semantic function, which takes an AST to a set or an element of a set in $\mathbf{ITT_1}$, is provided for each phylum. As indicated in [2], definitions of semantic functions are given relative to a collection R of constants and axioms. In the following, we will explain **Table 2**, which gives the definitions in detail.

Notations First, we fix some notations. We use $\langle\langle\rangle\rangle$ to delimit meta level function applications. [] is an empty context. **Bold face** is used for mata level functions between expressions of $\mathbf{ITT_1}$. These functions (e.g., **M** and **unpack**) are auxiliary functions for use in the definition of semantic functions. AST's are enveloped by $[\![\ ]\!]$, as are in conventional denotational semantics.

All terms in lefthand side of $\stackrel{\text{def}}{=}$ are terms in the subset theory $\mathbf{ITT}_1^{sub}$ except for the case of

$$\mathcal{B}[\![\texttt{subsort}\ sort_1\ \texttt{<}\ sort_2\ .\ \ body]\!] \ldots \stackrel{\text{def}}{=} \ldots$$

in which case the term is in the base set theory $\mathbf{ITT_1}$. Since all judgements in $\mathbf{ITT}_1^{sub}$ are interpreted in $\mathbf{ITT_1}$, we can very well expand the judgements and terms of $\mathbf{ITT}_1^{sub}$ in $\mathbf{ITT_1}$. We did not do so just for readability of the definitions.

We assume that all elements of the set of identifiers *Id* can be introduced to $\mathbf{ITT_1}$ as a constant; i.e., constants and identifiers are taken from the same collection of symbols.

References

[1] Nordström, Bengt; Petersson, Kent; Smith, Jan M.: Programming in Martin-Löf's Type Theory, Oxford University Press, 1990.

[2] Kinoshita, Y.: OBJ3 has predicative polymorphism, in *Proceedings of the 8th Japan Society for Software Science and Technology conference*, 1991.

$$\mathcal{O}[\![\texttt{obj}\ id\ \texttt{is}\ body\ \texttt{endo}]\!] \stackrel{\text{def}}{=} \lceil id \in \mathbf{M}\langle\!\langle \mathcal{B}[\![body]\!][]\rangle\!\rangle\rceil$$

$$\mathcal{O}[\![\texttt{obj}\ id_1\ \texttt{[}\ id_2\ \texttt{::}\ e\ \texttt{]}\ \texttt{is}\ body\ \texttt{endo}]\!] \stackrel{\text{def}}{=} \lceil id_1 \in (\Pi\, id_2 \in \mathcal{E}_T[\![e]\!])\,\mathbf{M}\langle\!\langle \mathcal{B}[\![body]\!][]\rangle\!\rangle\rceil$$

$$\mathcal{T}[\![\texttt{th}\ id\ \texttt{is}\ body\ \texttt{endth}]\!] \stackrel{\text{def}}{=} \lceil id \equiv \mathbf{M}\langle\!\langle \mathcal{B}[\![body]\!][]\rangle\!\rangle\rceil$$

$$\mathcal{T}[\![\texttt{th}\ id_1\ \texttt{[}\ id_2\ \texttt{::}\ e\ \texttt{]}\ \texttt{is}\ body\ \texttt{endth}]\!] \stackrel{\text{def}}{=} \lceil id_1 \equiv (\Pi\, id_2 \in \mathcal{E}_T[\![e]\!])\,\mathbf{M}\langle\!\langle \mathcal{B}[\![body]\!][]\rangle\!\rangle\rceil$$

$$\mathcal{V}[\![\texttt{view}\ id\ \texttt{from}\ e_1\ \texttt{to}\ e_2\ \texttt{is}\ vbody\ \texttt{endv}]\!] \stackrel{\text{def}}{=} \lceil id \in \mathcal{VB}[\![vbody]\!][\![e_1]\!][\![e_2]\!]\rceil$$

$$\mathcal{B}[\![emptybody]\!]\rho \stackrel{\text{def}}{=} \lceil \mathsf{T}\rceil$$

$$\mathcal{B}[\![\texttt{using}\ e\ .\ \ body]\!]\rho \stackrel{\text{def}}{=} \lceil(\Sigma\, x_1 \in A_1)\ldots(\Sigma\, x_n \in A_n)\,\mathcal{B}[\![body]\!]\langle\!\langle \rho \uplus [x_1 \in A_1, \ldots, x_n \in A_n]\rangle\!\rangle\rceil$$

$$\mathcal{B}[\![\texttt{sort}\ sort\ .\ \ body]\!]\rho \stackrel{\text{def}}{=} \lceil(\Sigma\, sort \in \mathsf{U})\,\mathcal{B}[\![body]\!]\langle\!\langle \rho \uplus [sort \in \mathsf{U}]\rangle\!\rangle\rceil$$

$$\mathcal{B}[\![\texttt{subsort}\ sort_1 \texttt{ < } sort_2\ .\ \ body]\!]\rho \stackrel{\text{def}}{=}$$
$$\lceil \mathsf{Eq}(\mathsf{U}, \mathcal{S}[\![sort_1]\!]', \mathcal{S}[\![sort_2]\!]')\,\&(\Pi\, x \in \mathcal{S}[\![sort_1]\!]')\,\mathcal{S}[\![sort_1]\!]''(x) \supset \mathcal{S}[\![sort_2]\!]''(x)\rceil$$

$$\mathcal{B}[\![\texttt{op}\ id\ \texttt{:}\ \ sort_1\ \ldots\ sort_n\ \texttt{->}\ sort\ .\ \ body]\!]\rho \stackrel{\text{def}}{=}$$
$$\lceil(\Sigma\, id \in s_1 \times \cdots \times s_n \to s)\,\mathcal{B}[\![body]\!]\langle\!\langle \rho \uplus [id \in s_1 \times \cdots \times s_n \to s]\rangle\!\rangle\rceil$$
where $s_i = \mathcal{S}[\![sort_i]\!]$ for $1 \le i \le n$ and $s = \mathcal{S}[\![sort]\!]$

$$\mathcal{B}[\![\texttt{var}\ id\ \texttt{:}\ \ sort\ .\ \ body]\!]\rho \stackrel{\text{def}}{=} \lceil(\Pi\, id \in \mathcal{S}[\![sort]\!]\rho)\,\mathcal{B}[\![body]\!]\rho\rceil$$

$$\mathcal{B}[\![\texttt{eq}\ t_1\ \texttt{=}\ t_2\ .\ \ body]\!]\rho \stackrel{\text{def}}{=} \lceil \mathsf{Eq}(A, t_1, t_2) \times \mathcal{B}[\![body]\!]\rho\rceil,$$
where $t_1 \in A\,[\rho]$ follows R

$$\mathcal{E}_T[\![id]\!] \stackrel{\text{def}}{=} \begin{cases} \lceil id\rceil & \text{if } id \equiv A \text{ follows } R, \\ \lceil A\rceil & \text{if } id \in A \text{ follows } R. \end{cases}$$

$$\mathcal{E}_T[\![id\ \texttt{[}\ view\ \texttt{]}]\!] \stackrel{\text{def}}{=} \lceil B[id_2 := C]\rceil$$
where $\mathcal{E}_T[\![id]\!] = \lceil(\Pi\, id_2 \in A)\,B\rceil$ and $\mathcal{V}[\![view]\!] = \lceil id_3 \in C\rceil$.

$$\mathcal{E}_O[\![id]\!] \stackrel{\text{def}}{=} \lceil id\rceil$$

$$\mathcal{E}_O[\![id\ \texttt{[}\ view\ \texttt{]}]\!] \stackrel{\text{def}}{=} \lceil \mathsf{apply}(id, \mathcal{V}[\![view]\!])\rceil$$

$$\mathcal{S}[\![id]\!] \stackrel{\text{def}}{=} \mathsf{Set}(id)$$

$$\mathcal{S}[\![id\ .\ \ e]\!] \stackrel{\text{def}}{=} \mathsf{Set}(\langle\!\langle \mathcal{E}_O[\![e]\!]\rangle\!\rangle \downarrow id)$$

$$\mathcal{A}[\![id]\!] \stackrel{\text{def}}{=} id$$

$$\mathcal{A}[\![id\ \texttt{(}\ t_1\ \texttt{,}\ \ldots\ \texttt{,}\ t_n\ \texttt{)}]\!] \stackrel{\text{def}}{=} \mathsf{apply}(id, \langle \mathcal{A}[\![t_1]\!], \ldots, \mathcal{A}[\![t_n]\!]\rangle)$$

$$\mathcal{A}[\![term\ .\ \ e]\!] \stackrel{\text{def}}{=} \mathcal{A}'[\![e;\ term]\!]$$

$$\mathcal{A}'[\![e;\ id]\!] \stackrel{\text{def}}{=} \langle\!\langle \mathcal{E}_O[\![e]\!]\rangle\!\rangle \downarrow id$$

$$\mathcal{A}'[\![e;\ id\ \texttt{(}\ t_1\ \texttt{,}\ \ldots\ \texttt{,}\ t_n\ \texttt{)}]\!] \stackrel{\text{def}}{=} \mathsf{apply}(\langle\!\langle \mathcal{E}_O[\![e]\!]\rangle\!\rangle \downarrow id, \langle \mathcal{A}'[\![e;\ t_1]\!], \ldots, \mathcal{A}'[\![e;\ t_n]\!]\rangle)$$

$$\mathcal{A}'[\![e;\ term\ .\ \ e']\!] \stackrel{\text{def}}{=} \mathcal{A}'[\![e';\ term]\!]$$

$$\mathcal{VB}[\![emptyvbody]\!][\![e_1]\!][\![e_2]\!] \stackrel{\text{def}}{=} \mathcal{E}_O[\![e_1]\!]$$

$$\mathcal{VB}[\![vbody\ .\ \ \texttt{sort}\ id\ \texttt{to}\ sort]\!] \stackrel{\text{def}}{=} \langle\!\langle \mathcal{VB}[\![vbody]\!][\![e_1]\!][\![e_2]\!]\rangle\!\rangle\,![id := \mathcal{S}[\![sort\ .\ e_2]\!]]$$

$$\mathcal{VB}[\![vbody\ .\ \ \texttt{op}\ id\ \texttt{to}\ term]\!] \stackrel{\text{def}}{=} \langle\!\langle \mathcal{VB}[\![vbody]\!][\![e_1]\!][\![e_2]\!]\rangle\!\rangle\,![id := \mathcal{A}[\![term\ .\ e_2]\!]]$$

Table 2: Semantic functions