

Subsets in type theory

Bengt Nordström

`bengt@cs.chalmers.se`

ChungAng University

on leave from Chalmers University, Göteborg, Sweden

An example: Linear search

Write a program that, given a natural number n and a function $f \in \mathbf{N} \rightarrow \mathbf{Bool}$ outputs the least index j for which $f[j] = \text{true}$ and $j < n$. If there is no such index the output is n .

A formal specification is the set

$$\prod n \in \mathbf{N}. \prod f \in \mathbf{N} \rightarrow \mathbf{Bool}. \text{First}(n, f)$$

where

$$\text{First}(n, f) \equiv \{j \in \mathbf{N} \mid (\forall k \in \mathbf{N}. k < j \supset f[k] = \text{false}) \\ \& (j \leq n) \& (j < n \supset f[j] = \text{true})\}$$

The base case

Assume that $n \in \mathbb{N}$ and $f \in \mathbb{N} \rightarrow \text{Bool}$. We are going to find an element in $\text{First}(n, f)$ by induction over n . In the base case, $0 \in \text{First}(0, f)$ since

$$\forall k \in \mathbb{N}. k < 0 \supset f[k] = \text{false}$$

$$0 \leq 0$$

$$(0 < 0 \supset f[j] = \text{true})$$

all hold.

The induction step

Assume now that $x \in \mathbb{N}$ and $y \in \text{First}(x, f)$. We must find an element in $\text{First}(\text{succ}(x), f)$. There are two cases.

- If $f[y] = \text{true}$, then y is the index of the first true element in the interval up to x . But then y is also the first index in the interval up to $\text{succ}(x)$.
- If $f[x] = \text{false}$, then there are no true elements in the interval up to x and hence $\text{succ}(x)$ is a solution to the problem.

Combining these two cases we obtain that

if $f[y]$ then y else $\text{succ}(x)$

is an element in $\text{First}(\text{succ}(x), f)$.

This solves the induction step.

Finishing the derivation

We can conclude that

$$\text{natrec}(n, 0, (x, y) \text{ if } f[y] \text{ then } y \text{ else succ}(x)) \in \text{First}(n, f).$$

Notice that the program which is obtained here only contains the computationally interesting parts of the proof. The rest of the proof is in the derivation of the program. In the traditional type theory we identify the derivation of the program with the program itself. Can we change type theory to obtain this?

First attempt

We try to introduce subsets as an addition to the polymorphic type theory.

Let A be a set and B a propositional function (family of sets) defined on the set A , i.e. assume A set and $B(x)$ set $[x \in A]$. From these assumptions and the explanation of what it means to be a set, it follows that the canonical elements and their equality relation is understood for the set A and for the set $B(a)$ whenever $a \in A$.

The subset of A with respect to B is denoted

$$\{|\}(A, B)$$

where $\{|\}$ is a constant of arity $\mathbf{0} \otimes (\mathbf{0} \multimap \mathbf{0}) \multimap \mathbf{0}$. Instead of $\{|\}(A, B)$, we shall use the more lucid notation

$$\{x \in A \mid B(x)\}$$

Definition of the set

If a is a canonical element in the set A and $B(a)$ is true, i.e. if there exists an element $b \in B(a)$, then a is also a canonical element in the set $\{x \in A \mid B(x)\}$.

Subset – introduction 1

$$\frac{a \in A \quad b \in B(a)}{a \in \{x \in A \mid B(x)\}}$$

But it is difficult to define an elimination rule!

Proposed elimination rule

Subset – elimination 1

$$\frac{c \in \{x \in A \mid B(x)\} \quad d(x) \in C(x) \quad [x \in A, y \in B(x)]}{d(c) \in C(c)}$$

where y must not occur free in d nor in C

Salvesen and Smith showed that propositions of the form

$$(\forall x \in \{z \in A \mid P(z)\})P(x)$$

cannot in general be proved. In the intensional theory, not even $(\forall x \in \{z \in \mathsf{T} \mid \perp\})\perp$ can be derived.

The subset theory

We will define a new theory, where sets and propositions are separated.

We have *primitive* constants for the logical constants: $\&$, \vee and \supset of arity $\mathbf{0} \otimes \mathbf{0} \multimap \mathbf{0}$, \perp of arity $\mathbf{0}$, \forall and \exists of arity $\mathbf{0} \otimes (\mathbf{0} \multimap \mathbf{0}) \multimap \mathbf{0}$.

We also need a primitive constant ID of arity $\mathbf{0} \otimes \mathbf{0} \otimes \mathbf{0} \multimap \mathbf{0}$ for forming the proposition that two elements of a certain set are equal. Instead of $\text{ID}(A, a, b)$ we will often write $a =_A b$.

What does it mean to be a set?

To know the judgement

A set

we must have A' and A'' and know

- A' set
- $A''(x)$ prop $[x \in A']$

What does it mean for two sets to be equal

To know that A and B are equal sets

$$A = B$$

in the sense of the subset theory, is to know

- $A' = B'$
- $A''(x) \equiv B''(x) \text{ true } [x \in A']$

as explained in the basic set theory.

What does it mean to be an element in a set

To know the judgement

$$a \in A$$

we must know the judgements

- $a \in A'$
- $A''(a)$ *true*

as explained in the basic set theory.

What does it mean to be a proposition?

To know a proposition P in the subset theory is to know a proposition P^* in the basic set theory.

Since P may contain quantifiers ranging over subsets, P^* will depend on the interpretation of subsets. Since propositions are interpreted as sets in the basic set theory, P^* is nothing but a set in the basic theory.

What does it mean for a proposition to be true

To know that the proposition P is true in the subset theory is to know that P^* is true in set theory.

Hypothetical Judgements

The general shape of a context is:

$$[\mathcal{C}_1, \dots, \mathcal{C}_n]$$

where the assumption \mathcal{C}_k is either of the form

- $x_k \in A_k(x_1, \dots, x_{k-1})$ where $A_k(x_1, \dots, x_{k-1})$ is a subset in the context $\mathcal{C}_1, \dots, \mathcal{C}_{k-1}$
- $P(x_1, \dots, x_k)$ true where $P(x_1, \dots, x_k)$ is a proposition in the context $\mathcal{C}_1, \dots, \mathcal{C}_{k-1}$.

I will only deal with contexts of the form

$$x \in C, P(x) \text{ true}, y \in D(x)$$

To be a set under assumptions

To know the judgement

$$A(x, y) \text{ set } [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory is to have a pair (A', A'') such that

$$A'(x, y) \text{ set } [x \in C', y \in D'(x)]$$

and

$$A''(x, y, z) \text{ prop } [x \in C', y \in D'(x), z \in A'(x, y)]$$

both hold in the basic set theory.

Note that being a set under assumptions does not depend on any proposition being true.

To be equal sets under assumptions

To know the judgement

$$A(x, y) = B(x, y) \quad [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory, is to know the judgements

$$A'(x, y) = B'(x, y) \quad [x \in C', y \in D'(x)]$$

and

$$A''(x, y) \equiv B''(x, y) \text{ true} \quad [x \in C', C''(x) \text{ true}, \\ P^*(x) \text{ true}, y \in D'(x), D''(x, y) \text{ true}]$$

in the basic set theory.

To be an element in a set

To know the judgement

$$a(x, y) \in A(x, y) \quad [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory, is to know the judgements

$$a(x, y) \in A'(x, y) \quad [x \in C', y \in D'(x)]$$

and

$$A''(x, y, a(x, y)) \text{ true} \quad [x \in C', C''(x) \text{ true}, \\ P^*(x) \text{ true}, y \in D'(x), D''(x, y) \text{ true}]$$

in the basic set theory.

Two equal elements

To know the judgement

$$a(x, y) = b(x, y) \in A(x, y) \quad [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory, is to know the judgement

$$a(x, y) = b(x, y) \in A'(x, y) \quad [x \in A', y \in B'(x)]$$

in the basic set theory.

To be a proposition

To know the judgement

$$Q(x, y) \text{ prop } [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory is to know the judgement

$$Q^*(x, y) \text{ prop } [x \in C', y \in D'(x)]$$

in the basic set theory.

A proposition is true

To know the judgement

$$Q(x, y) \text{ true } [x \in C, P(x) \text{ true}, y \in D(x)]$$

in the subset theory, where $Q(x, y)$ is a proposition in the context $x \in C, P(x) \text{ true}, y \in D(x)$, is to know the judgement

$$Q^*(x, y) \text{ true } [x \in C', C''(x) \text{ true}, \\ P^*(x) \text{ true}, y \in D'(x), D''(x, y) \text{ true}]$$

in the basic set theory.

Interpretation: logical constants

$$(P \& Q)^* \equiv P^* \times Q^*$$

$$(P \vee Q)^* \equiv P^* + Q^*$$

$$(P \supset Q)^* \equiv P^* \rightarrow Q^*$$

$$\top^* \equiv \top$$

$$\perp^* \equiv \emptyset$$

$$((\forall x \in A) P(x))^* \equiv (\Pi x \in A')(A''(x) \rightarrow P^*(x))$$

$$((\exists x \in A) P(x))^* \equiv (\Sigma x \in A')(A''(x) \times P^*(x))$$

$$(a =_A b)^* \equiv \text{Id}(A', a, b)$$

Interpretation: sets

Remember that $a \in A$ means that $a \in A'$ and $A''(a)$ true.

$$\{x \in A \mid P(x)\}' \equiv A'$$

$$\{x \in A \mid P(x)\}'' \equiv (z)(A''(z) \times P^*(z))$$

$$\{i_1, \dots, i_n\}' \equiv \{i_1, \dots, i_n\}$$

$$\{i_1, \dots, i_n\}'' \equiv (z)\top$$

$$\mathbf{N}' \equiv \mathbf{N}$$

$$\mathbf{N}'' \equiv (z)\top''$$

$$((\Pi x \in A)B(x))' \equiv (\Pi x \in A')B(x)'$$

$$((\Pi x \in A)B(x))'' \equiv (z)((\Pi x \in A')(A''(x) \rightarrow B(x)''(\text{apply}(z, x))))$$

Interpretation: sets, cont'd

Remember that $a \in A$ means that $a \in A'$ and $A''(a)$ true.

$$(A + B)' \equiv A' + B'$$

$$(A + B)'' \equiv (z)((\exists x \in A')(A''(x) \times \text{Id}(A', z, \text{inl}(x))) + (\exists y \in B')(B''(y) \times \text{Id}(B', z, \text{inr}(y))))$$

$$((\Sigma x \in A)B(x))' \equiv (\Sigma x \in A')B'(x)$$

$$((\Sigma x \in A)B(x))'' \equiv (z)(A''(\text{fst}(z)) \times B(\text{fst}(z))''(\text{snd}(z)))$$