$(\forall\iota\sigma\alpha\Theta\exists\iota\sigma\alpha\Pi)$

sTs

HOL Extended with MFOL-ZFS

(The Mostly First-Order Language of ZF Sets)

Gottfried Barrow

v.2012.11.28.20:24:26

Short Contents

Contents

1 Header Chapter Title Test

1.1 The Axiom of Extension

$\big(\iota\sigma\big)$ (Isar) 1.1.1. (Theory name, imports, and begin.)

```
theory sTs01
imports Complex_Main "i"
begin
```

1.1.2 Test of inline, equation, multline, eqnarray, label, ref, cite

I start off with the bibliography citations [Gol96, 59] and [JH99, Bil03].

Next is an index entry on the word “entry’’, and an index entry where “markup’’ is sub-indexed by using the option `!entry` in the `\index` command.

Footnotes? What I really want to say can only be said below. [1]

I now go on to those very important inline equations and equation environments, so please consider the difficulty of the inline equation $x^1 + \lfloor 1.2 \rfloor = 1$, and now the equation

$$\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2), \tag{1}$$

along with the very ugly `multline` environment,

$$\begin{multline}(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow \\ Px \longleftrightarrow \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)), \tag{2}\end{multline}$$

at least when I stick that middle line in there.

The `eqnarray` environment with 3 columns specified by `[rcl]` is frequently used:

$$\begin{aligned} a &= b \\ &= c \\ &= d. \tag{3}\end{aligned}$$

Having to tweak LaTeX raises its ugly head, so I replace Equation (2) with a two column `eqnarray`, and change the default column alignment from `[rcl]` to `[rl]`. I do that with my equation array environment prefix ⟨`rl`⟩. The ⟨`rl`⟩ is a prefix instead of a suffix, because the suffix position is taken up by the optional reference label.

$$\begin{aligned} (\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow& \\ Px \longleftrightarrow& \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)).& \tag{4}\end{aligned}$$

I could have used the three column default, and left one column empty, but this is for demonstration.

Here, I ask the question, “Are what I’ve been calling equations really equations?’’ If we consider that the HOL $\longleftrightarrow$ operator is merely notation for =, then I suppose they are equations, but as a matter of style, I would rather use “Formula’’, and I have set it up to where “Formula’’ is a synonym for “Equation’’ when used with `\label` and `\eqref`.

[1] Some people could be annoyed by all of this.

After experimenting, I decided I can get rid of the `multline` environment. The environment `eqnarray`, from the `mathenv` package, replaces the standard `eqnarray` and it is a very versatile equation environment which can be used in place of many other environments.

Here's the `multline` environment:

$$(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)). \tag{5}$$

And here's the same equation using the `eqnarray` environment, set for one right justified column:

$$(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)). \tag{6}$$

Personally, I think Formula (6) is formatted better than Formula (5). And there are others ways to format Formula (5) with `eqnarray`, like

$$(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)), \tag{7}$$

or

$$(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2)) \longleftrightarrow \\ ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)), \tag{8}$$

This is why LaTeX can be a bad thing. I can spend way too much time tweaking the formatting of equations.

1.1.3 The primitive set type and membership predicate

ZF sets is a first-order language which requires an infinite set of variables, and it generally goes unsaid in the formalization of a first-order language that the variables provided are of a single type.

However, in HOL there are a multitude of variable types, and additionally, we are allowed to define a new type of variable so we can have a new variable type that exists in its own domain.

For ZF sets, I define the primitive variable type `sT`, where the non-ASCII notation for `sT` is σ_ι. The subscripted character for σ_ι is the Greek letter iota.

$\left(\tau\upsilon\right)$ (Type) 1.1.4. (The primitive set type `sT`: everything is a set.)

```
typedecl sT ("σι")
```

ZF sets is specified to have one predicate, which is membership. The ASCII and non-ASCII notation for membership are `inS` and $\in_\iota$, along with negation of membership, which is notated by `niS` and $\notin_\iota$.

$\left(\omega\pi\right)$ (Operator) 1.1.5. (Membership predicate `inS`: axiomatized by subsequent axioms.)

```
consts inS :: "σι ⇒ σι ⇒ bool" (infixl "inS" 55)
  notation
    inS                       (infixl "∈ι"  55)
  abbreviation
    niS :: "σι ⇒ σι ⇒ bool"    (infixl "niS" 55) where "x niS y == ¬(x ∈ι y)"
  notation
    niS                       (infixl "∉ι"  55)
```

1.1.6 Axiomatizing the two forms of the Axiom of Extension

The standard Axiom of Extension is the formula

$$\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2). \tag{9}$$

The question is whether in Isabelle/HOL, this standard formula is equivalent to the free variable form

$$(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2), \tag{10}$$

where q_1 and q_2, because they are not in the scope of any quantifier, are free variables.

Naively, the following counterexample seems to indicate that they are not equivalent.

$\left(\kappa\xi\right)$ (CounterX) 1.1.7. (A counterexample is found for the naive equivalence.)

```
theorem
  "(∀q₁.∀q₂.(∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂)) ⟷
               ((∀x. x ∈ι q₃ ⟷ x ∈ι q₄) ⟷ (q₃ = q₄))"
--"nitpick[sat_solver=SAT4J,timeout=60,verbose,user_axioms]"
--"Nitpick found a counterexample for card σι = 2:
   Free variables: (q₃::σι) = s1
                   (q₄::σι) = s2"
  oops
```

The problem is that the prover engine implicitly quantifies all free variables in the statement of a theorem, so the formula in the above counterexample is not the equivalence that we need to prove or disprove.

This quantification is done at the outermost level with universal quantifiers. For example, the formula

$$\begin{aligned}&(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2))\\ &\quad\longleftrightarrow ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4))\end{aligned} \tag{11}$$

can be considered to be equivalent to the quantified formula

$$\begin{aligned}&\forall q_3.\forall q_4.(\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2))\\ &\quad\longleftrightarrow ((\forall x.x \in_\iota q_3 \longleftrightarrow x \in_\iota q_4) \longleftrightarrow (q_3 = q_4)).\end{aligned} \tag{12}$$

I say "considered'' because the quantification is actually being done at the meta-logic level. To see that free variables are quantified by the meta-logic quantifier $\bigwedge$, we can look at the output of `thm` in the following example.

$\left(\xi\pi\right)$ (Example) 1.1.8. (Free variables are quantified at the meta-logic level.)

```
theorem
  free‚variable‚conjecture:
  "(∀x::'a. P x) ⟷ (P x)"
  sorry

theorem
  And‚quantified‚conjecture:
  "⋀x.(∀x::'a. P x) ⟷ (P x)"
  sorry

thm
  free‚variable‚conjecture
```

```
  And·quantified·conjecture

--"thm output for both: (∀(x::?'a). ?P::(?'a ⇒ bool) x) = (?P (?x::?'a))."

theorem
  forall·quantified·conjecture:
  "∀x.(∀x::'a. P x) ⟷ (P x)"
  sorry

thm
  forall·quantified·conjecture

--"thm output: ∀(x::?'a). (∀(x::?'a). ?P::(?'a ⇒ bool) x) = (?P x)."
```

If we were to prove the first two conjectures in the above example, then the resulting theorems would be the same, as shown by the output of thm. (Example 1.1.8 also shows how free variables and $\bigwedge$ quantified variables are tied together by means of schematic variables.)

As shown by Example 1.1.8, free variables are quantified by $\bigwedge$, and so I use the fact that because free variable Formula (13), shown here,

$$(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2), \tag{13}$$

is equivalent to Formula (14),

$$\bigwedge q_1. \bigwedge q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2), \tag{14}$$

then it is equivalent to this fully quantified Formula (15),

$$\forall q_1.\forall q_2.(\forall x.x \in_\iota q_1 \longleftrightarrow x \in_\iota q_2) \longleftrightarrow (q_1 = q_2). \tag{15}$$

This is the desired equivalence, that is, that the standard form of the Axiom of Extension is equivalent to the free variable form.

To prove this, because $\bigwedge$ is a meta-logic operator, we must resort to meta-logic, which I now do in the proof of Theorem 1.1.9.

$\Big(\Theta\Big)$ (Theorem) 1.1.9. (Standard and free variable Axiom of Extension equivalence.)

```
theorem extension·equivalence:
  "(Trueprop (∀q₁.∀q₂.(∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂))) ≡
            (⋀q₃.⋀q₄.((∀x. x ∈ι q₃ ⟷ x ∈ι q₄) ⟷ (q₃ = q₄)))"
```

$\Big(\Pi\Big)$ (Proof) 1.1.9.1.

```
proof
  assume "∀q₁.∀q₂.(∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂)"
  then show "⋀q₃.⋀q₄.(∀x. x ∈ι q₃ ⟷ x ∈ι q₄) ⟷ (q₃ = q₄)"
  by simp
next
  assume "⋀q₃.⋀q₄.((∀x. x ∈ι q₃ ⟷ x ∈ι q₄) ⟷ (q₃ = q₄))"
  then show "∀q₁.∀q₂.(∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂)"
  by simp
qed
```

The use of `simp` in Proof 1.1.9.1 [Ste12] shows that the proof of Theorem 1.1.9 is trivial, which is not surprising, since we would expect meta-logic $\bigwedge$ and the HOL operator $\forall$ to both produce the same result. But though the proof is trivial, the theorem is not unimportant, since Formula (13) will be stated as an axiom, and when it comes to extending HOL with axioms, due diligence should be exercised, although what is "due diligence" to one may be "belaboring the point" to another.

Of note is that Theorem 1.1.9 has been resorted to out of necessity, and it is tempting to also want to formally prove that Formula (13) is equivalent to Formula (14), but we would end up in the same situation, that because there are free variables in Formula (13), those variables would get quantified by $\bigwedge$ if we tried to state, in a single formula, that Formula (13) and Formula (14) are equivalent. (In a single Isar hypothesis, there may be a way to state that two formulas are equivalent without using $\equiv$, $\longleftrightarrow$, or $=$, but if there is, I am not aware of it.)

Now, because it has been shown that the two forms of the Axiom of Extension are equivalent, then we can axiomatize either form and get the other. However, rather than just axiomatize one of the formulas, I axiomatize both. The `Nu` form of `Ax⸴x` axiom will be used, whenever possible, for theorems with no free variables. The `Nf` form will be used for theorems which contain free variables. This allows the possibility of separating the axioms and theorems with free variables from the axioms and theorems which contain no free variables.

$\left(\alpha\xi\right)$ (Axiom) 1.1.10. (The Axiom of Extension: set equality.)

```
axiomatization where
  Ax⸴x^Nu: "∀q₁.∀q₂.(∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂)" and
  Ax⸴x^Nf: "          (∀x. x ∈ι q₁ ⟷ x ∈ι q₂) ⟷ (q₁ = q₂)"
```

1.2 Rough LaTeX markup notes and documentation

My rough notes and documentation on how I'm marking things up. They're commented out and don't show up in the PDF.

There's no fool proof method behind my selection of what character to use to create a markup delimiter. I think a little, and then I start using a character until I think of something better or find out why a certain character is not going to work.

1.2.1 Guidelines

1.2.2 Text and text formatting

1.2.3 Equations

1.2.4 cite, index, footnote ref, label

$\left(\iota\sigma\right)$ (Isar) 1.2.5. (Theory end.)

```
end
```

Bibliography

[Bil03] Stefan Bilaniuk. *A Problem Course in Mathematical Logic*. Version 1.6 edition, 2003.

[Gol96] Derek Goldrei. *Classic Set Theory*. Chapman and Hall/CRC, New York, 1996.

[JH99] Thomas Jech and Karel Hrbacek. *Introduction to Set Theory*. Marcel Dekker, Inc., New York, 3rd edition, 1999.

[Ste12] Christian Sternagel. Re: [isabelle] free variable dead horse beat; getting two equiv formulas after either one is axiomatized. http://lists.cam.ac.uk/mailman/htdig/cl-isabelle-users/2012-October/msg00112.html, October 2012.

Index