Frege's Concept Horse Problem in the Simply-Typed λ-calculus

David Titarenco (group: Jon Ben & Jose Trujillo)

University of California, Los Angeles

Abstract

In On Function and Concept, G. Frege makes a distinction between concepts and objects. Specifically, objects are *saturated object-expressions* while concepts are *unsaturated function-expressions*. The goal of this paper is to illuminate what the *object-concept* distinction entails in the context of **simply-typed lambda calculus** ($\lambda^{\rightarrow}$) and whether or not this entailment is succeptible to problems like the concept horse paradox.

I. INTRODUCTION

Frege's On Function and Concept (1892) claims that there is a substantive difference between concepts and objects. This distinction is described on pg. 147:

> Statements in general, just like equations or inequalities or expressions in Analysis, can be imagined to be split up into two parts; one complete in itself, and the other in need of supplementation, or 'unsaturated'. Thus, e.g., we split up the sentence 'Caesar conquered Gaul' into 'Caesar' and 'conquered Gaul'.

Definition 1. Complete parts of statements are 'saturated' while incomplete ones are 'unsaturated'.

Example 1. Consider the following statement:

$$\text{`Fido jumped over the fence'} \tag{1}$$

By **Definition 1**, we can see that (1) is split up into the saturated 'Fido' object-expression and 'jumped over the fence' unsaturated function-expression.

Remark 1. These two categories, as it turns out, have simple analogues in $\lambda^{\rightarrow}$. Unsaturated function-expressions are terms with functional types, e.g. $N^{\sigma\rightarrow\tau}$ or $N : \sigma \rightarrow \tau$, while saturated object-expressions are terms with non-functional types, e.g. M^{σ} or $M : \sigma$.

Definition 2. We say that if M gets type σ and N gets type $\sigma \rightarrow \tau$ then the application NM is **legal** (as N is considered a function from terms of type σ to terms of type τ) and gets type τ.

Definition 3. The set of types of $\lambda^{\rightarrow}$, denoted by Type($\lambda^{\rightarrow}$) is inductively defined as follows. We write $\mathbb{T} = \text{Type}(\lambda^{\rightarrow})$ where

$$\begin{array}{ll} \alpha, \alpha', \alpha'', \ldots \in \mathbb{T} & \text{(type variables);} \\ \sigma, \tau \in \mathbb{T} \Rightarrow (\sigma \rightarrow \tau) \in \mathbb{T} & \text{(function space variables).} \end{array}$$

Definition 4. We will now define a new variant of $\mathbb{T}$ that more closely models Frege's natural language notions of *concept* and *object*. This new variant, $\mathbb{T}_{\mathcal{F}}$, has the following restrictions:

(a) There are three and *only* three types: $\begin{cases} \alpha \in \mathbb{T}_{\mathcal{F}} & (\textit{objects}); \\ \mathbf{H} \in \mathbb{T}_{\mathcal{F}} & (\text{truth-values}); \\ (\alpha \to \mathbf{H}) \in \mathbb{T}_{\mathcal{F}} & (\textit{concepts}). \end{cases}$

(b) *Objects* have an unrestricted domain.

(c) A truth-value is a boolean type where $\mathbf{H} = \{\text{'true'}, \text{'false'}\}$.

(d) A *concept* is a functional type that takes an *object* as an input and returns a truth value. This mapping is done by some valuation function v where v maps to 'true' if the function-expression applied to the argument(s) is true, e.g., in (1), if Fido *did* jump over a fence then v('Fido') = 'true'; otherwise, v maps to 'false'.

(e) Note that $(\mathbf{H} \to \alpha) \notin \mathbb{T}_{\mathcal{F}}$.

Remark 2. Until otherwise noted, we will work only with types in $\mathbb{T}_{\mathcal{F}}$.

Definition 5. Let M be an untyped $\lambda^{\to}$-term. Given the non-functional type α, we say that $M : \alpha$ is a *saturated object-expression*, or simply an *object*.

Definition 6. Let N be an untyped $\lambda^{\to}$-term. Given the functional type $\alpha \to \mathbf{H}$, we say that $N : \alpha \to \mathbf{H}$ is an *unsaturated function-expression*, or simply a *concept*.

Definition 7. We will now make a semantic distinction between two kinds of uses of the token 'is' in natural language. To illustrate the distinction, consider:

(a) 'Mark Twain is Samuel Clemens': here, we have the *'is' of identity*, or $x = x$. The equality operator $=$ is used as a mathematical formalism and may not necessarily be computable in $\lambda^{\to}$. So, v('Mark Twain') = 'true' iff 'Mark Twain' = 'Samuel Clemens'.

(b) 'Mark Twain is dead': here, we have the *'is' of predication* which, if combined with some property P will yield a functional type of form $\alpha \to \mathbf{H}$ for some object α. As in **Definition 4**, if α has property P, $v(\alpha)$ = 'true'; otherwise, v maps to 'false'.

II. SIMPLE NATURAL LANGUAGE SENTENCES IN $\lambda^{\to}$

Theorem 1. Consider the same sentence from **Example 1**:

$$\text{`Fido jumped over the fence'} \tag{2}$$

This sentence can be formulated in $\lambda^{\to}$ like so:

$$\Gamma \vdash J : \alpha \to \mathbf{H}, \Gamma \vdash F : \alpha \Rightarrow \Gamma \vdash (JF) : \mathbf{H} \tag{3}$$

Proof. We will set the *object-expression* 'Fido' = F and the (one-place) *function-expression* '(x) jumped over the fence' = $J\,(x)$. Note that we will drop the (x) in practice for λ-calculus wff satisfiability. By **Definition 5**, F must be of type α. And to make (JF) legal, by **Definition 2**, J must be of type $\alpha \to \mathbf{H}$. Therefore, a correctly typed (JF) from $J : \alpha \to \mathbf{H}$ and $F : \alpha$ looks like

$$(JF) : \mathbf{H} \tag{4}$$

which is precisely what we wanted to show in (3). □

Proof. We can also derive a more formally-rigorous proof. Consider:

$$\frac{\Gamma \vdash J : \alpha \to \mathbf{H} \quad \Gamma \vdash F : \alpha}{\Gamma \vdash JF : \mathbf{H}}{\scriptstyle (\to e)} \tag{5}$$

where we get the same result: JF is of type $\mathbf{H}$. □

Remark 3. Natural language predicates (e.g. 'is blue', 'jumped over the fence', 'is a concept', 'conquered Gaul', 'may have stolen my wallet', etc.) are *concepts* in $\mathbb{T}_{\mathcal{F}}$ while subjects, be they proper nouns, improper nouns, pronouns, etc. (e.g. 'the car', 'Fido', 'he', 'the tall man', 'the Brooklyn Bridge', etc.) are *objects*. We will now look at more nontrivial examples of natural language propositions and attempt to model them in $\lambda^{\to}$ under $\mathbb{T}_{\mathcal{F}}$.

Example 2. Consider the statement:

$$\text{`The concept horse is a concept.'} \tag{6}$$

Claim 1. We will set the *object-expression* 'The concept horse' = T_{ch} and the (one-place) *function-expression* '(x) is a concept' = C. **Example 2** is witness to a semantic ambiguity between natural language and our type system ($\mathbb{T}_{\mathcal{F}}$). More specifically, (6) evaluates to 'true' in natural language, but 'false' under $\mathbb{T}_{\mathcal{F}}$, i.e. CT_{ch} : $\boldsymbol{H}$ has an ambiguous truth-value. So, if $T_{ch} : \alpha$ is the *saturated object-expression* 'the concept horse' and $C : \alpha \rightarrow \boldsymbol{H}$ is the *unsaturated function-expression* 'is a concept', then the mapping $v : \alpha \rightarrow \mathbf{H}$ is ambiguous.

Proof. Consider the following natural language statement:

$$\text{`The house is a house.'} \tag{7}$$

Trivially, (7) is true. Per **Definition 7**, the 'is' here seems to be an *'is' of identity*, so v('house') = 'true' since 'house' = 'house'.

Now, let's define a new set of types $\mathbb{T}^*_{\mathcal{F}}$ where $\mathbb{T}^*_{\mathcal{F}}$ is just like $\mathbb{T}_{\mathcal{F}}$ except that $\mathbb{T}^*_{\mathcal{F}}$ has one and only one additional type: $\beta \in \mathbb{T}^*_{\mathcal{F}}$. We will call all terms of type β *houses*. Now, we an ambiguity in (7); it is unclear whether we're asking if the identity 'house' = 'house' holds or whether we're asking if the *saturated object-expression* 'the house' is of type β. The former is true per the first half of this proof. The latter is false, as any *object-expression* is of type α in $\mathbb{T}_{\mathcal{F}}$ and, by extension, also in $\mathbb{T}^*_{\mathcal{F}}$ and $\alpha \neq \beta$. This same kind of ambiguity happens in (7). □

Definition 8. Given the proof of *Claim 1*, we will now denote two variations of v, essentially defining two valuations for $\alpha \rightarrow \mathbf{H}$:

(a) v_0 for meta-language predicates;

(b) $v_{\mathfrak{L}}$ for natural language predicates.

In light of this, we now have two interpretations for statements that involve *objects*, *concepts*, and truth-values. So, in some cases, $v_0(CT_{ch} : \mathbf{H})$ will evaluate to 'false', but $v_{\mathfrak{L}}(CT_{ch} : \mathbf{H})$ evaluates to 'true' or vice-versa. This is known as the concept horse paradox.

Theorem 2. *Given any object-expression,* $\mathbb{O} : \alpha$ *and if we let the function-expression '(x) is a concept'* $= C$, $v_0(C\mathbb{O} : \boldsymbol{H})$ *will always evaluate to 'false'.*

Proof. From the proof of *Claim 1* and **Definition 8**. □

Example 3. Now consider the statement:

$$\text{`Seabiscuit is the concept horse.'} \tag{8}$$

Claim 2. We will set the *object-expression* 'Seabiscuit' $= S$ and the (one-place) *function-expression* '(x) is the concept horse' $= C_h$. **Example 3** avoids the ambiguity in (6) because its truth value happens to be false under both valuations, so $v_0(C_h S : \mathbf{H}) = v_{\mathfrak{L}}(C_h S : \mathbf{H})$.

Proof. By **Theorem 2**, $v_0(C_h S : \mathbf{H})$, will always evaluate to 'false'. In natural language, 'Seabiscuit' is not 'the concept horse' (at best, it is an instance of 'the concept horse'), so $v_{\mathfrak{L}}(C_h S : \mathbf{H})$ will evaluate to 'false' here as well. Since both valuations return 'false', we avoid the paradox. □

Example 4. Consider the statement:

$$\text{'Anna is something Bill is not – namely, a student.'} \tag{9}$$

Remark 4. **Example 4** is not ambiguous as it only uses terms found in natural language. This is trivial to prove (since the only valid valuation is $v_{\mathfrak{L}}$) However, (9) introduces another problem with $\mathbb{T}_{\mathcal{F}}$ (more specifically, with $\lambda^{\rightarrow}$): subtyping. It seems intuitive that 'being a student' should be a member of 'things Bill is not'. As it turns out, $\lambda^{\rightarrow}$ has no way of representing this.

III. PROPOSED SOLUTIONS

A. Semantic Culling

The most straightforward solution to the semantic problem of the concept horse paradox is a simple one. Instead of using terms like *concept* and *object*, that not only have a meaning in the meta-language, but also in natural languages, we will use terms that have no meaning in natural language. Consider a new type system, $T^{**}_{\mathcal{F}}$.

$$\text{The types of } \mathbb{T}^{**}_{\mathcal{F}}\text{:} \begin{cases} \alpha \in \mathbb{T}^{**}_{\mathcal{F}} & (\textit{foo}); \\ \mathbf{H} \in \mathbb{T}^{**}_{\mathcal{F}} & (\text{baz}); \\ (\alpha \rightarrow \mathbf{H}) \in \mathbb{T}^{**}_{\mathcal{F}} & (\textit{foobaz}). \end{cases}$$

Under such a schema, statements like 'The concept horse is [a] foobaz' are meaningless in natural language, but are semantically well-formed in the meta-language. Thus, we can infer that we need to valuate the statement with v_0 as opposed to $v_{\mathfrak{L}}$ and avoid any ambiguity.

Remark 5. The caveat here is that we lose some expressiveness in natural languages. In **Example 3**, we were able to refer both to the semantics of natural language and the meta-language without ambiguity. This would no longer be possible.

B. Syntactic Sugar

Another proposed solution is syntactic in nature. We previously showed that 'Fido jumped over the fence' (**Example 1**) can be formulated as follows:

$$\frac{\Gamma \vdash J : \alpha \to \mathbf{H} \quad \Gamma \vdash F : \alpha}{\Gamma \vdash JF : \mathbf{H}}\,{\scriptstyle(\to e)} \tag{10}$$

We will now introduce a new type of $\lambda^{\to}$-term that will differentiate between what valuation v one should use to determine the truth-value of some $M : \mathbf{H}$. To do this, consider a new type system $\mathbb{T}^{\omega}_{\mathcal{F}}$.

$$\text{The types of } \mathbb{T}^{\omega}_{\mathcal{F}}\text{:} \begin{cases} \alpha \in \mathbb{T}^{\omega}_{\mathcal{F}} & (\textit{objects}); \\ \mathbf{H} \in \mathbb{T}^{\omega}_{\mathcal{F}} & (\text{truth-values}); \\ (\alpha \to \omega \to \mathbf{H}) \in \mathbb{T}^{\omega}_{\mathcal{F}} & (\text{unrestricted } \textit{concepts}); \\ (\omega \to \mathbf{H}) \in \mathbb{T}^{\omega}_{\mathcal{F}} & (\text{restricted } \textit{concepts}); \\ \omega \in \mathbb{T}^{\omega}_{\mathcal{F}} & (\mathbf{worlds}). \end{cases}$$

A derivation of (10) would now look like the following:

$$\frac{\dfrac{\Gamma \vdash J : \alpha \to \omega \to \mathbf{H} \quad \Gamma \vdash F : \alpha}{\Gamma \vdash JF : \omega \to \mathbf{H}}\,{\scriptstyle(\to e)} \qquad \Gamma \vdash W : \omega}{\Gamma \vdash JFW : \mathbf{H}}\,{\scriptstyle(\to e)} \tag{11}$$

The purpose of W is to pick out one (or several, if there are no ambiguities) valuating function(s). So, in **Example 1**, W picks out $v_{\mathfrak{L}}$, in **Example 3**, W picks out $\{v_{\mathfrak{L}}, v_0\}$, and in **Example 2**, W either picks out $v_{\mathfrak{L}}$ or v_0 but not both. A benefit of $W : \omega$ is that it can pick out the meta-language (v_0), meta-meta-language (v_1), meta-meta-meta-langauge (v_2), etc. (v_n). Furthermore, W can also pick out combinations of languages ($\{v_{\mathfrak{L}}, v_0, v_2, v_n\}$), provided the truth value stays consistent.

Remark 6. A caveat of this syntactic addition is that W tends to be implicit and thus, the concept horse paradox merely shifts (now we have an ambiguity of W) and does not completely disappear. A rather elegant fix is letting W pick out $v_{\mathfrak{L}}$ by default. In such a case, the paradox would dissolve and we would still preserve the truth-value in sentences like **Example 3**.