Typed Object Theory*

Edward N. Zalta
Center for the Study of Language and Information
Stanford University
zalta@stanford.edu

The theory of abstract objects, hereafter 'object theory', is a system that axiomatizes both ordinary and abstract individuals, on the one hand, and properties, relations, and propositions, on the other. It has been applied in a variety of ways, for example:

- in the analysis of mathematical objects and mathematical relations,[1]
- in the analysis of (a) the failures of substitution in intensional contexts, and (b) the denotations of names of fictions,[2] and
- in the analysis of possible worlds and situations, Plato's Forms, Leibnizian concepts, Frege numbers, truth-values and the logical conception of sets, and impossible worlds.[3]

Many of the above applications can be framed within second-order, quantified modal object theory. However, others require *typed object theory*, in which object theory is formulated within a background of relational type theory. Typed object theory has been developed in a number of previous works and I henceforth assume some basic familiarity with it.[4] Its most important features are: (1) there are two atomic forms of predication that may be asserted of relations and objects with the correct types, *exemplification* formulas of the form $F^n x_1 \ldots x_n$ and *encoding* formulas of the form xF[1], (2) the domains of higher-order types are populated with *primitive* relations, which are axiomatized by comprehension and identity principles, (3) every type is partitioned into ordinary and abstract objects of that type, and (4) the abstract objects of each type are axiomatized by a comprehension principle which asserts, for any condition φ on properties of objects of a given type, that there is an abstract object of that type which encodes all and only the properties that satisfy φ.[5] I shall further sketch and document these and other features of typed object theory as the occasion arises below.

In this paper, typed object theory will be examined within the context of the history of *relational* type theory, including such works as Russell 1908, Carnap 1929, Orey 1959, Schütte 1960, Church 1974, and Gallin 1975. Along the way, we shall have cause to briefly compare relational type theory with functional type theory (Church 1940, Montague 1973, and others). As part of the discussion, I examine some recent work in light of typed object theory. In particular, I examine Muskens 2007, Liefke 2014 and Liefke & Werning forthcoming, and Williamson 2013. In each case, I try to show that conclusions drawn in those works are not inevitable if one considers how the target data may be analyzed in typed object theory. I hope to show how typed object theory provides a (possibly more) natural understanding of the phenomena that these other type theories were designed to explain, and I attempt to undermine conclusions that might call into question the way typed object theory is formulated and applied.

1 Relational vs. Functional Type Theory

I begin with a question raised by Partee (2009 [2007], 37):

> If I am asked why we take e and t as the two basic semantic types, I am ready to acknowledge that it is in part because of tradition, and in part

 This paper was written for the volume *Abstract Objects: For and Against*, Concepción Martínez Vidal and José Luis Falguera López (eds.), under review by Synthese Library. I'd like to thank Paul Oppenheimer for carefully reading a draft of this paper and for his valuable suggestions for improvement.

[1] See Zalta 1983 (VI), 2000a, 2006a; Linsky & Zalta 1995, and Nodelman & Zalta 2014.

[2] See Zalta 1983 (VI), 1988a (9–12), and 2000b.

[3] See Zalta 1993, 1997, 1999, 2000c; Pelletier & Zalta 2000; and Anderson & Zalta 2004.

[4] See Zalta 1982, 1983 (V, VI), 1988a (9–12, Appendix), 2000a (§3); Linsky & Zalta 1995; and Nodelman & Zalta 2014 (47–49).

[5] The key idea underlying object theory is that abstract objects *encode* rather than exemplify the properties by which we conceive of them and that such objects may encode *only* those properties and no others. As such abstract objects may be incomplete with respect to the properties they encode: there are properties F and abstract objects x such that neither xF nor $x\overline{F}$ (where $\overline{F}$ is the negation of F). But abstract objects are complete with respect to the properties they exemplify: for every property F, every abstract object x (and indeed every object whatsoever) is such either Fx or $\overline{F}x$.

The notion of encoding didn't appear in object theory *ex nihilo*. Pelletier & Zalta 2000 show how Meinwald 1992 traces a similar idea in Plato's distinction between two kinds of predication; Anderson & Zalta 2004 show show how Boolos 1987 (3) finds the idea in Frege's 'two instantiation relations'; and Zalta 2006 shows how Kripke discusses 'a confusing double usage of predication' in his 1973 [2013] lectures. The idea also appears in other works, as documented in some of the publications on object theory cited thus far.

> because doing so has worked well. … In a certain sense Montague had a third basic type, the type of possible worlds; in Gallins Ty2 (Gallin 1975) this is explicit. But that is not essential, since on some alternatives the basic type t is taken to be the type of propositions, inherently intensional.

I'd like to put forward another answer to Partee's opening question. It starts by pointing out that e and t are the two basic semantic types only if we start with a *functional type theory* (FTT) of the kind developed by Church and Montague. By contrast, *relational type theory* (RTT) uses a single semantic type. RTT starts with a single type i, and then where $\sigma_1, \ldots, \sigma_n$ are any types, $\langle \sigma_1, \ldots, \sigma_n \rangle$ ($n \geq 0$) is the derived type for relations among objects having types $\sigma_1, \ldots, \sigma_n$. When $n = 0$, the type $\langle\ \rangle$ is the type for propositions. So, Partee's question is posed within a background of FTT instead of RTT.

Even starting with FTT instead of RTT, I think Partee's question has an alternative answer, namely, that it starts with the two types e and t because FTT *requires* entities (individuals) and truth values for the analysis of predication in terms of function application. This requirement traces back, I think, to the fact that Frege assumed two primitive (mutually exclusive) domains (*functions* and *objects*) and basic formulas of the form $f(x) = y$, which combine the primitive notions of *function application* and *identity*. Frege's two primitive domains classified entities into two types and in order to analyze predication, which he understood as object *x falls under* concept F, Frege had to introduce two distinguished objects, the truth values **T** and **F**. Thus, a concept F could be analyzed as a function whose values are always one of the two truth values and so *x falls under F* (i.e., predication), in effect, becomes analyzed as: $F(x) = \mathbf{T}$.

Church (1940) generalized Frege's logic to allow for functions of higher type, thereby developing the first FTT. Church used α, β, and γ as variables ranging over types, and in 1940 (56), we find:

- ι and o are type symbols.
- if α and β are type symbols, then $(\alpha\beta)$ is a type symbol.

Here ι is the type for individuals, o is the type of propositions, and $(\alpha\beta)$ is the type of functions with arguments of type β and values of type α. So, for example, a property becomes a function with type $(o\iota)$, i.e., a function from individuals to truth values, and a 2-place relation between individuals has type $((o\iota)\iota)$, i.e., a function from individuals to properties. Note that Church couldn't have rested with just a single primitive type ι for individuals and the single derived type $(\alpha\beta)$. That would only yield higher-order *mappings* and not anything that would

permit him to adopt Frege's analysis of predication in terms of functional application. So he added the primitive type o as the type for a truth bearer (e.g., truth values or propositions). Thus, for Church, simple predications and more complex assertions are expressions of type o, i.e., terms that denote truth values. To assert that individual x exemplifies the property F in Church's system, F must be a variable of type $(o\iota)$ and x a variable of type ι, and the expression (Fx), which represents predication, is an expression of type o (1940, 57).

Montague (1973) extended Church's typing scheme to include what appears to be a third primitive type, namely s, for possible worlds. In 1973 (256), we find Montague defining the set of *types* as the smallest set Y such that:

- $e, t \in Y$
- whenever $a, b \in Y$, $(a, b) \in Y$
- whenever $a \in Y$, $(s, a) \in Y$.

So, for Montague, (e, t) is the type for functions from individuals to truth values, i.e., the characteristic function for a set of individuals; $(e, (e, t))$ is the type for extensional relations between individuals, i.e., the characeristic function for a set of ordered pairs; $(s, (e, (e, t)))$ is the type of functions from possible world to extensional relations between individuals, i.e., intensional relations; and propositions have type (s, t), i.e., functions from possible worlds to truth values. Gallin 1975, Cresswell 1975, 1985, Thomason 1980, Fox & Lappin 2001, and Pollard 2005, 2008 all employ variations of this scheme.[6]

[6] In Gallin (1975, 58), sytem Ty_2 (Two-Sorted Type Theory), we find:

> e, t, s $\in \mathrm{T}_2$
>
> $\alpha, \beta \in \mathrm{T}_2$ imply $(\alpha, \beta) \in \mathrm{T}_2$.

Cf. Cresswell 1975, where there is a categorial language and semantics based on functions. In Thomason 1980 (48–9), we find:

> Basic types: e, t, and p, where p is for propositions
> If σ and τ are any types, then $\langle \sigma, \tau \rangle$ is a type.

Thomasons says we may "think of $\langle \sigma, \tau \rangle$ as the type of functions from the domain of type σ to that of type τ" (49). Cresswell (1985, 69) explicitly uses $\mathrm{D}_{(\tau/\sigma_1,\ldots,\sigma_n)}$ "to indicate a class of n-place functions whose domains are taken from $\mathrm{D}_{\sigma_1}, \ldots, \mathrm{D}_{\sigma_n}$, respectively, and whose range is in D_τ". Fox & Lappin 2001 (176–7) use:

> Basic Types: e for individuals and Π for propositions
> Exponential Types: If A, B are types, then A^B is a type.

Here, the type A^B is a functional type. Finally, in Pollard' (2008, 273) hyperintensional type theory, there are 3 basic types (Ent, Ind, Prop) and then a variety of functional and other complex types.

Semanticists like Montague naturally started with the best mathematical framework they could find, and this turned out to be a version of Church's FTT. Two basic types are needed in FTT to preserve Frege's analysis of predication in terms of function application. Without the type for truth values or propositions, one can't represent the bearers of *truth*.

1.1 Why Relations in RTT Were Interpreted as Functions

By contrast, the bearers of truth come *for free* as 0-place relations when you formulate relational type theory (RTT). This lends it a natural elegance. RTT follows Russell's somewhat different understanding of logic, in which relations are more basic than functions. Functions become defined for Russell as 2-place relations R such that $\forall x \forall y \forall z(Rxy \,\&\, Rxz \rightarrow y = z)$. RTT doesn't need a separate primitive type for truth values since *propositions* can be analyzed as 0-place relations. The reason semanticists haven't used RTT no doubt stems from Quine's concern about the identity of intensional entities like properties and relations. The theory of relations hasn't seemed as mathematically precise as the theory of functions, since the identity conditions for relations (conceived intensionally) aren't as clearcut as those of sets or the functions definable in terms of sets. But, as we'll see, this worry doesn't apply to the version of RTT defended in this paper.

As we've noted, RTT begins with a single primitive type for individuals. This goes back to Russell 1908 (237), who took individuals, relations, and predication as basic. Though Russell didn't introduce explicit notation for types, he clearly thought of *individuals* as forming the lowest type.[7] Carnap (1929) was the first to introduce notation for types, but his notation isn't one we currently use.[8] Rather, it seems that Orey (1959) developed the now standard definition of relational types, though he didn't regard the entities in the domains of relational types as primitive relations. He defined (1959, 73) a set of *type symbols* to be the smallest set T such that:

- $\iota \in$ T, and
- if $\tau_1, \ldots, \tau_n \in$ T, then $(\tau_1 \ldots \tau_n) \in$ T $\qquad (n \geq 1)$

[7] I shall not be discussing ramified type theory in what follows. Our attention shall be restricted entirely to the simple theory of (relational) types. See Anderson 1989 for a discussion of a ramified type theory and intensional logic.

[8] Carnap (1929, 30–32) used type `t0` for individuals, `t1` for classes of individuals, `t2` for classes of classes of individuals, and so on. He used type `t(00)` for relations among individuals, etc. There was no empty type for propositions, however type `t1` could be rewritten as `t(0)`.

Note here that Orey starts with a single primitive type ι for individuals, and has a single derived type $(\tau_1 \ldots \tau_n)$, which is is the type for relations that relate arguments of type $\tau_1, \ldots, \tau_n$, respectively.

However, Orey interprets expressions of type $(\tau_1 \ldots \tau_n)$ as denoting sets of n-tuples. He uses $G(\tau)$ to denote the domain of type τ, takes $G(\iota)$ to be a nonempty set, and then takes $G(\tau_1 \ldots \tau_n)$ to be some "nonvoid subset of the set of all subsets of the cartesian product of $G(\tau_1) \times \ldots \times G(\tau_n)$" (1959, 73). So, semantically, relations are interpreted as sets of n-tuples.[9]

Church contributed to RTT as well as originating FTT. He extended Orey's notation for relational types by allowing (1974, 25) the empty relational type ( ). He starts with a single primitive type i for individuals and says that where $\beta_1, \beta_2, \ldots, \beta_m$ are any types, then $(\beta_1, \beta_2, \ldots \beta_m)$ is a type. He allows for $m = 0$, and so ( ) is the type for propositions. Church agrees that one should, in some sense, take relations seriously. He says (1974, 22):

> Russell's logic must be understood intensionally if some of its significant features are to be preserved. This means in the first place that the values of the propositional-functional (for short "functional") variables are understood to be properties, in the case of singulary functional variables, or binary relations in intension in the case of binary functional variables, ... To go with this the values of Russell's propositional variables must also be taken as intensional,[...] that is, as propositions in the abstract sense rather than either sentences on the one hand or truth-values on the other.

Nevertheless, Church interprets RTT in terms of functions. Church (1974, 26) starts with two primitive domains of propositions, namely $\mathfrak{T}$ (the domain of true propositions) and $\mathfrak{F}$ (the domain of false propositions), and analyzes relations as m-ary *functions* that take values in $\mathfrak{T} \cup \mathfrak{F}$. He interprets the domain for expressions of type $(\beta_1, \beta_2, \ldots \beta_m)$ as consisting of m-ary *functions* whose m arguments have type $\beta_1, \beta_2, \ldots, \beta_m$ and whose values are (in the domain of) propositions. So, semantically, relations are understood as functions.

Gallin (1975) preserves this understanding of relations, but extends it to a language with a modal operator. Gallin first defines a set of *types* as follows

[9] Schütte (1960, 306) uses types 0 (for individuals) and 1 (for truth values) and then introduces relational types. But he analyzes predication *syntactically* as set membership. He sets up the language (1960, 307) so that in item (1.3.3) we find:

> If $e_1, \ldots, e_n$ are expressions of types $\tau_1, \ldots, \tau_n$ and e is an expression of type $(\tau_1, \ldots, \tau_n)$, then $(e_1, \ldots, e_n \,\epsilon\, e)$ is an expression of type 1

So it is clear why Schütte interprets relational terms (predicates) and λ-expressions as denoting sets of n-tuples.

(1975, 68):

> Let $\mathtt{e}$ be any symbol which is not a finite sequence. The set $\mathtt{P}$ of *predicate types* is the smallest set such that:
>
> (i) $\mathtt{e} \in \mathtt{P}$, and
>
> (ii) $\sigma_0, \sigma_1, \ldots, \sigma_{n-1} \in \mathtt{P}$ imply $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1}) \in \mathtt{P}$.
>
> ... Objects of type $\mathtt{e}$ will be individuals, and objects of type $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$ will be relations of n arguments, of which the first is an object of type σ_0, the second an object of type σ_1, etc.

He then defines a modal language $\mathtt{ML_P}$ and reinterprets the objects of type $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$. He says (1975, 71):

> In $\mathtt{ML_P}$, objects of type $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$ will be predicates (relations-in-intension) of n arguments, of which the first is an object of type σ_0, the second an object of type σ_1, etc.

Gallin then defines the semantic interpretation of a predicate of type $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$ as a function from an index (i.e., a possible world) to a set of n-tuples drawn from the appropriate types (1975, 72–3). Gallin also allowed for Henkin-style *general* models of $\mathtt{ML_P}$, and in those models, the domain of each type is some *subset* of the domain used for standard models. In other words, in a general model, the domain of type $(\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$ is some *subset* of the set of all those n-tuples $\langle a_0, \ldots a_{n-1}\rangle$ such that a_0 is an entity of type σ_0, o_1 is an entity of type σ_1, ..., and a_{n-1} is an entity of type σ_{n-1}. So no matter whether you consider Gallin's standard models or general models, relational predicates are still interpreted as entities that obey the principle of extensionality: they are identical when they map the same arguments (indices) to the same values (sets of n-tuples). Thus, relations are again understood semantically in terms of functions.

It is worth mentioning, at this point in the exposition, that the *intensions* discussed by Carnap, Montague, and Gallin constitute a theoretical model of the *relations-in-intensions* discussed by Russell, Church, and others. Russell and Church do not have a theory of relations-in-intensions on which they become identical when necessarily equivalent. But the intensions of Carnap, Montague, and Gallin exhibit this feature. Montague and Gallin explicitly represent intensions as functions from possible worlds to extensional entities such as truth-values, sets of individuals, or sets of n-tuples. But the primitive relations that populate the domains of relational type theory, as developed in typed object theory, are more fine-grained than intensions as conceived by Carnap, Montague, and Gallin. Such relations may be distinct even if necessarily equivalent and they may offer a better understanding of the notion of *relation-in-intension* as understood by Russell and Church.

In the remainder of this paper, we shall follow Gallin in using $\sigma_1, \sigma_2, \ldots$ as variables ranging over types. The foregoing brief history shows that the main works on RTT took relations seriously only in the syntax; semantically, they either preserved Frege's method of reducing relations to functions or took relations to be sets of n-tuples.[10] The RTT developed in Zalta 1982, 1983, 1988a, and 1988b stands in contrast: in these works, the members of the domain for each derived type are *primitive relations*. Though the notation for types used in 1982, 1983, and 1988b was not as elegant as the notation used in 1988a, nevertheless, in all four works, no attempt was made to reduce relations to functions or sets in the semantics.[11] Using the more elegant formulation of Zalta 1988a (231), a *type* was defined as follows:

- 'i' is a *type*
- Whenever $\sigma_1, \ldots, \sigma_n$ are any *types*, $\ulcorner\langle\sigma_1, \ldots, \sigma_n\rangle\urcorner$ is a *type* $\quad (n \geq 0)$

In the usual way, i is the type for *individuals* and $\langle\sigma_1, \ldots, \sigma_n\rangle$ is the type for *n-place relations*, the arguments of which have types $\sigma_1, \ldots, \sigma_n$, respectively. When $n = 0$, $\langle\ \rangle$ is the type for *propositions*. Thus, if we use the expression F as a typical term of type $\langle\sigma_1, \ldots, \sigma_n\rangle$ and use the expressions $x_1, \ldots, x_n$ as

[10] van Benthem & Doets 1983, and Muskens 1989, followed Orey in this regard. Van Benthem & Doets define (1983, 269):

$$D_{(\tau_1,\ldots,\tau_n)}(A) = \mathcal{P}(D_{\tau_1}(A) \times \cdots \times D_{\tau_n}(A))$$

i.e., they define the domain of relations among entities with types $\tau_1, \ldots, \tau_n$ to be the set containing all the sets of n-tuples with elements drawn, respectively, from the domains of type $\tau_1, \ldots \tau_n$. Muskens also interprets relational expressions as sets of n-tuples rather than as functions. See his definition of Orey frames (1989, 2). But later in this paper, in his system $\mathrm{TT}^{\eta,2}$, we discover that he doesn't take *predications* as basic formulas of the language. Instead, function application is basic (1989, 12, Definition 10, ii), as it is in Muskens 1995 (p. 15, Definition 9, iv).

[11] In 1982 (298), and 1983 (109), the relational types were defined basically as follows:

- i is a *type*
- p is a *type*
- Whenever $\sigma_1, \ldots, \sigma_n$ are any types, $\ulcorner(\sigma_1, \ldots, \sigma_n)/p\urcorner$ is a type $\quad (n \geq 1)$

So in these early works, $(\sigma_1, \ldots, \sigma_n)/p$ is the derived type for relations. Since the primitive type p in these works corresponds to the new derived type $\langle\ \rangle$ in 1988a, the structure of the types is the same. But whereas Zalta 1988a clearly uses a single primitive type, Zalta 1982, 1983, and 1988b suggest that there are two primitive types (see, e.g., 1988b, 69). In these latter works, I was still under the influence of the Montague Grammar I had learned as a graduate student, and hadn't yet recognized that one could eliminate p as a second primitive type by defining type p as the empty derived type $\langle\ \rangle$.

typical terms with types $\sigma_1, \ldots, \sigma_n$, the language of typed object theory includes (atomic) *exemplification* formulas of the form $Fx_1 \ldots x_n$. And where where F has type $\langle\sigma\rangle$ and x has type σ, the language includes (atomic) *encoding* formulas of the form xF.

In all the works on typed object theory, however, the semantics included, for each type σ, a non-empty domain $\mathcal{D}_\sigma$ consisting of primitive entities, as well as a separate non-empty domain $\mathcal{W}$ of possible worlds. Relations in the domains of type $\mathcal{D}_{\langle\sigma_1,\ldots,\sigma_n\rangle}$ were then clearly distinguished from set-theoretic representations thereof by introducing a world-relative *exemplification extension* function that essentially maps each relation to a Montagovian intension. Specifically, this function mapped each pair consisting of an n-place primitive relation $\boldsymbol{r}$ in $\mathcal{D}_{\langle\sigma_1,\ldots,\sigma_n\rangle}$ and a possible world $\boldsymbol{w}$ in $\mathcal{W}$ to a set of n-tuples drawn from the power set of $\mathcal{D}_1 \times \ldots \times \mathcal{D}_n$.[12] So the exemplification extension function helps one give a semantic representation of the world-relative truth conditions of atomic exemplification formulas: where F has type $\langle\sigma_1, \ldots, \sigma_n\rangle$ and $x_1, \ldots, x_n$ have types $\sigma_1, \ldots, \sigma_n$, respectively, the formula '$Fx_1 \ldots x_n$' is true at a world w just in case the n-tuple of objects denoted by $x_1, \ldots, x_n$ is an element of the exemplification extension at $\boldsymbol{w}$ of the relation denoted by F.

Thus, the relational predicates of typed object theory *do not denote* Montagovian intensions; the semantics allows for distinct relations that have the same exemplification extension at every possible world. The semantics of typed object theory also includes an *encoding extension* function that maps each property in each domain $\mathcal{D}_{\langle\sigma\rangle}$ to a set of objects drawn from the domain $\mathcal{D}_\sigma$. The encoding extension function is used to give the truth conditions for atomic encoding formulas: where x has type σ and F has type $\langle\sigma\rangle$, then the encoding formula 'xF' is true at a world $\boldsymbol{w}$ just in case the object in $\mathcal{D}_\sigma$ denoted by x is an element of the *encoding extension* of the property in $\mathcal{D}_{\langle\sigma\rangle}$ denoted by F. Since the truth conditions of 'xF' are independent of the possible worlds, the formula $xF \rightarrow \Box xF$ will be valid.

The reason typed object theory doesn't follow the tradition of interpreting the relations in $\mathcal{D}_{\langle\sigma_1,\ldots,\sigma_n\rangle}$ as functions or sets is that it comes with a precise theory of relations. With a precise theory of relations in hand, one can argue that *no mathematical model* of relations in set theory is needed to represent them in the semantics. Indeed, one shouldn't use a mathematical model that collapses necessarily equivalent relations, given that object theory doesn't require the collapse. This stands in contrast to the versions of RTT in Orey, Church,

[12] See Zalta 1982 (299); 1983 (114); and 1988a (236). The exemplification function also maps each proposition $\boldsymbol{r}$ in $\mathcal{D}_{\langle\,\rangle}$ and a possible world $\boldsymbol{w}$ in $\mathcal{W}$ to a truth value.

and Gallin. Each of these yields artifactually valid statements, i.e., statements that are valid because of the way relations have been represented. For example, either $\forall x(Fx \equiv Gx) \rightarrow F = G$ or $\Box\forall x(Fx \equiv Gx) \rightarrow F = G$ is artifactually valid in other versions of RTT. But object theory isn't committed to either claim.

Before we discuss how typed object theory approaches identity for relations, we first review how identity for relations is defined in *second-order* object theory. In second-order object theory, one can state a precise comprehension principle (i.e., a schema that yields general conditions under which relations exist) and identity principles for relations. Moreover, the latter principles offer *extensional* conditions for the identity of relations, notwithstanding our conception of them as intensional entities. Readers familiar with the second-order version of object theory will recall that the language includes two modes of predication, *exemplification* formulas of the form $\Pi\kappa_1 \ldots \kappa_n$ (where Π is any n-place relation term and $\kappa_1, \ldots, \kappa_n$ are any individual terms), and *encoding* formulas of the form $\kappa\Pi$ (where κ is any individual term and Π any 1-place relation term, i.e., any property term). In the second-order version of object theory, properties F and G are semantically assigned not only an exemplification extension that can vary from world to world, but also an *encoding* extension. Thus, the language of second-order object theory distinguishes the following conditions on properties F and G:

(A) $\Box\forall x(Fx \equiv Gx)$

(B) $\Box\forall x(xF \equiv xG)$

We may then reject the idea that (A) provides identity conditions for F and G. But we accept (B) as providing correct identity conditions for properties even when we conceive of properties as intensional entities. That is, we define:

$$F = G \;=_{df}\; \Box\forall x(xF \equiv xG)$$

Moreover, given the logical principle that $xF \rightarrow \Box xF$, we may infer $\Box\forall x(xF \equiv xG)$ whenever $\forall x(xF \equiv xG)$. Hence, to prove that properties are identical in object theory, one need only prove $\forall x(xF \equiv xG)$. Thus, the foregoing definition and the logic of encoding gives us extensional identity conditions for intensional entities, since they intuitively imply that properties are identical when their encoding extensions are identical.

Moreover, in second-order object theory, identity conditions for propositions and for n-place relations ($n \geq 2$) can be defined in terms of the definition of property identity. Consider propositions first. Let x be a variable ranging over

individuals, and let p and q be variables ranging over propositions. Thus, in second-order object theory, expressions such as $[\lambda x\ p]$ ('*being such that p*') , $[\lambda x\, q]$ ('*being such that q*'), etc., denote properties of individuals; the properties denoted depend on the value assigned to p, q, etc. So we may say:

> Propositions p and q are identical if and only if the properties *being such that p* and *being such that q* are identical, i.e.,
>
> $p = q\ =_{df}\ [\lambda x\ p] = [\lambda x\, q]$

Here, proposition identity has been reduced to property identity. Now, for n-place relations where $n \geq 2$, second-order object theory allows us to define identity as follows. Let F and G both be n-place relation variables, for some $n \geq 2$. Then, we may say:

> F and G are identical just in case every way of 'plugging' $n-1$ individuals into F and G (plugging them in the same order into F and G) results in identical properties.

We can make this precise by using the following formal definition, which is well-formed in second-order object theory, where the arity of both F and G is some n such that $n \geq 2$:

$$\begin{aligned} F = G\ =_{df}\ & \forall x_1 \ldots \forall x_{n-1}([\lambda y\, Fyx_1 \ldots x_{n-1}] = [\lambda y\, Gyx_1 \ldots x_{n-1}]\ \&\\ & [\lambda y\, Fx_1 yx_2 \ldots x_{n-1}] = [\lambda y\, Gx_1 yx_2 \ldots x_{n-1}]\ \&\ \ldots\ \&\\ & [\lambda y\, Fx_1 \ldots x_{n-1}y] = [\lambda y\, Gx_1 \ldots x_{n-1}y]) \end{aligned}$$

These definitions make it clear that second-order object theory doesn't automatically collapse necessarily equivalent (in the classical sense) properties, relations, or propositions.

In particular, second-order object theory doesn't collapse properties that are necessarily equivalent in the sense of (A). There are lots of examples of properties F and G that are distinct despite being necessarily exemplified by the same objects. For example, the property *being a barber who shaves all and only those who don't shave themselves*, i.e.,

$$[\lambda x\, Bx\ \&\ \forall y(Sxy \equiv \neg Syy)]$$

is clearly distinct from the property *being a dog that is both white and not white*:

$$[\lambda x\, Dx\ \&\ Wx\ \&\neg Wx]$$

While these properties clearly satisfy (A), we may assert that they fail to satisfy (B) by asserting that there are abstract objects encode the one without encoding the other. Second-order object theory, however, stipulates that properties that satisfy (B) are identical. Examples of identical properties include: *being a brother* and *being a male sibling*, *being a circle* and *being a closed plane figure every point of which lies equidistant from some given point*, etc.

These features of second-order object theory are, in part, a consequence of the comprehension principle for abstract individuals that was asserted as part of that theory. That principle was formulated, in second-order object theory, as follows, where $A!$ denotes the property of *being abstract*:

$\exists x(A!x\ \&\ \forall F(xF \equiv \varphi))$, provided x doesn't occur free in φ

In other words, for any condition φ on properties F, there is an abstract object that encodes exactly those properties F such that φ. Thus, once we add the assertion that, for some given property pair P and Q that $P \neq Q$, it follows that there is an abstract object that encodes the one without encoding the other.[13] Moreover, if F and G are identical, there couldn't be an abstract object that encodes one without encoding the other.

1.2 Identity in Typed Object Theory

Typed object theory, in contrast to second-order object theory, gives us more flexibility in defining identity. In the discussions of typed-object theory in Zalta 1983 and 1988, the definitions for relation identity essentially followed the foregoing discussion. As a result, the definitions in those works were more 'type-specific' in that they defined identity for type i one way and defined identity for all relational types using principles analogous to those used in second-order object theory.[14] However, in what follows, we adopt the definitions in Zalta

[13] By comprehension, we have, using the defined notion of identity:

$\exists x(A!x\ \&\ \forall F(xF \equiv F = P))$

Clearly, this object encodes P without encoding Q.

[14] In Zalta 1983 (121, 124) and 1988 (241–2), we first defined identity for individuals as follows, where x and y are variables of type i, F is a variable of type $\langle i\rangle$, and $O!$ and $A!$ are the predicates of *being ordinary* and *being abstract*, respectively, both of type $\langle i\rangle$:

$x = y\ =_{df}\ (O!x\ \&\ O!y\ \&\ \Box\forall F(Fx \equiv Fy)) \vee (A!x\ \&\ A!y\ \&\ \Box\forall F(xF \equiv yF))$

Then, for any type σ, let x is a variable of type σ, and let F and G be variables of type $\langle\sigma\rangle$. Then we defined:

$F = G\ =_{df}\ \Box\forall x(xF \equiv xG)$

1982 (301–2) and 2000a (228), where we find identity defined for all types using a single formula scheme. First, recall that the domain of every type σ is partitioned into *ordinary* and *abstract* objects of that type. We use the typically ambiguous predicates $O!^{\langle\sigma\rangle}$ and $A!^{\langle\sigma\rangle}$, for every type σ, to denote the properties of *being ordinary* and *being abstract*, respectively.[15] Consequently, there are both ordinary and abstract properties of individuals, ordinary and abstract relations among individuals, etc.[16]

It is axiomatic that ordinary objects of type σ only exemplify properties, whereas abstract objects of type σ both exemplify and encode properties. Moreover, the abstract objects of each type σ are governed by a typed version of the comprehension principle discussed earlier. The typed version of this principle can be stated generally as follows. For any type σ, let x be a variable of type σ, F be a variable of type $\langle\sigma\rangle$, and $A!$ be a predicate (mentioned earlier) of type $\langle\sigma\rangle$. Then we take the instances of the following to be axioms, for any type σ:

$$\exists x(A!x \,\&\, \forall F(xF \equiv \varphi)), \text{ provided } x \text{ doesn't occur free in } \varphi$$

In other words, for any condition φ that places a (possibly empty) condition on the variable F, there is an abstract object x of type σ that encodes all and only the properties F such that φ.

Now since the domain of each type is partitioned into ordinary and abstract objects of that type, we can state identity conditions for objects of any type σ as follows. Let σ be any type; let x and y be variables of type σ; let F be a variable of type $\langle\sigma\rangle$; and let $O!$ and $A!$ be the predicates of *being ordinary* and *being abstract*, respectively, with type $\langle\sigma\rangle$. Then we define:

$$\text{(C)}\quad x = y \;=_{df}\; (O!x \,\&\, O!y \,\&\, \Box\forall F(Fx \equiv Fy)) \vee (A!x \,\&\, A!y \,\&\, \Box\forall F(xF \equiv yF))$$

Finally, identity for propositions and n-place relational types were defined along the lines used for second-order object theory. See the references cited at the beginning of this note for further details.

[15] Actually, *being ordinary* and *being abstract* are defined technical terms in typed object theory; we used a typically-ambiguous primitive predicate, $E!^{\langle\sigma\rangle}$ (for every σ), where this predicate intuitively picks out the concrete objects of type σ. Now let x be a variable of type σ, $O!$ be the predicate for *being ordinary*, with type $\langle\sigma\rangle$, and $A!$ be the predicate for *being abstract*, also with type $\langle\sigma\rangle$. Then we may say x exemplifies *being ordinary*, written $O!x$, just in case $\Diamond E!x$, and x exemplifies *being abstract*, written $A!x$, just in case $\neg\Diamond E!x$. This holds for every type σ. Thus, the domain of every type is partitioned into the ordinary and abstract objects of that type.

[16] The standard primary and secondary qualities count as good examples of ordinary properties of individuals, and we may include empty properties, such as *being a giraffe in the Arctic Circle*, *being round and square*, as ordinary properties of individuals. But fictional properties (e.g., *being a hobbit*, *being composed of phlogiston*, etc.) and mathematical properties (e.g., *being a Peano number*, *being a ZF set*, etc.) have been analyzed as abstract properties. These latter encode just the properties of properties attributed to them in their respective story or theory. *Absolute simultaneity*, the *membership relation* of ZF, etc., are examples of abstract relations.

In other words, for any type σ, ordinary objects of type σ are identical whenever they necessarily exemplify the same type-$\langle\sigma\rangle$ properties, and abstract objects of type σ are identical whenever they necessarily encode the same type-$\langle\sigma\rangle$ properties. Clearly, given the theorem $O!x \vee A!x$, we can derive $x = x$ from (C).[17] So, by taking the substitution of identicals as an axiom, typed object theory has a theory of identity in which substitution of identicals holds in any context.

To see an example of (C) in action, consider any type σ; let F and G be variables of type $\langle\sigma\rangle$; let $O!$ and $A!$ be the predicates *being ordinary* and *being abstract* with type $\langle\langle\sigma\rangle\rangle$; and let $\mathcal{F}$ be a variable of type $\langle\langle\sigma\rangle\rangle$. Then the following is an instance of (C) (where we've reduced the font size of 'F' and 'G' for readability):

$$\text{(D)}\quad F = G \;=_{df}\; (O!F \,\&\, O!G \,\&\, \Box\forall\mathcal{F}(\mathcal{F}F \equiv \mathcal{F}G)) \vee (A!F \,\&\, A!G \,\&\, \Box\forall\mathcal{F}(F\mathcal{F} \equiv G\mathcal{F}))$$

This definition governs any properties F and G with type $\langle\sigma\rangle$, for any σ. And, clearly, it yields the theorem $F = F$.

Note that (D) still allows necessarily equivalent properties to be distinct even when necessarily equivalent, i.e., (C) doesn't imply the typed version of (A). Moreover, the definition of property identity used in second-order object theory now becomes a theorem. That is, where x is a variable of type σ, and F and G have the types assigned in the previous paragraph, we may prove:

$$F = G \equiv \Box\forall x(xF \equiv xG)$$

We leave the proof to a footnote.[18]

[17] This follows by disjunction syllogism from the theorem $O!x \vee A!x$. Suppose $O!x$. Then since it is a modal theorem that $\Box\forall F(Fx \equiv Fx)$, we have $O!x \,\&\, O!x \,\&\, \Box\forall F(Fx \equiv Fx)$. By $\vee$I, this gives us that right-side of (C). So $x = x$. On the other hand, suppose $O!x$. Then since it is a modal theorem that $\Box\forall F(xF \equiv xF)$, we have $A!x \,\&\, A!x \,\&\, \Box\forall F(xF \equiv xF)$. By $\vee$I, this gives us that right-side of (C). So $x = x$.

[18] ($\rightarrow$) This direction is trivial, by the substitution of identicals and the modal theorem $\Box\forall x(xF \equiv xF)$. ($\leftarrow$). Assume $\Box\forall x(xF \equiv xG)$. Then by the T schema, $\forall x(xF \equiv xG)$. Now where $A!$ is the predicate *being abstract* of type $\langle\sigma\rangle$ and H is a variable of the same type, then using the defined notion of identity in (D), we have the following instance for comprehension for abstract individuals:

$$\exists x(A!x \,\&\, \forall H(xH \equiv H = F))$$

Call such an individual a, so that we know both $A!a$ and $\forall H(aH \equiv H = F)$. Instantiating the latter to F and G, respectively, we have both:

$$(\vartheta)\quad aF \equiv F = F$$

$$(\xi)\quad aG \equiv G = F$$

Since $F = F$, it follows from (ϑ) that aF. But, it follows from a previously established fact, namely $\forall x(xF \equiv xG)$, that $aF \equiv aG$. Hence aG. Now, for reductio, assume $F \neq G$, i.e., by symmetry of identity, $G \neq F$. Then by (ξ), $\neg aG$. Contradiction. $\bowtie$

With a precise theory of relations in hand, we have a foundational framework that allows one to represent classical predication in relational terms. We can regard 'x loves y', 'x worships y', '$x \in y$', '$x \leq y$', etc., as exemplification predications of the form Rxy, where x and y are individuals and R is a relation of type $\langle i, i\rangle$. From the point of view of typed object theory, there is nothing more basic than individuals and relations. We have no need of (a) the Fregean tradition of interpreting relations as functions and interpreting predication as functional application, or (b) the set-theoretic tradition of interpreting relations as sets of n-tuples and interpreting predication as set membership.[19] From the point of view of typed object theory, such traditions *fail to capture* the essential fact about predication, namely, that in a true exemplification predication of the form Fx, the property F *characterizes* x; it doesn't merely classify x or correlate (i.e., map) x to a truth-value.

For the remainder of this paper, we therefore assume it is a mistake to regard relations as functions or sets; as noted previously, such an interpretation collapses necessarily equivalent relations and validates principles to which typed object theory is not committed. Instead, it is clear that general, Henkin models in which the domain of each type consists of primitive entities of that type gives a more accurate picture of the ontology that underlies RTT in general and typed object theory in particular.[20] Moreover, as we shall see, typed object theory has some theoretical virtues when compared to other recent intensional interpretations of RTT.

2 Intensional Type Theory: I

It may be of interest to see what typed object theory accomplishes when compared to the framework developed in Muskens 2007. Muskens states clearly that he is not so much interested in the question of what the intensional entities that populate the domains of RTT *are*, but rather interested in the general features of any RTT in which the relational types denote *intensions* that can be distinguished from extensional entities such as sets of n-tuples. He traces a two-stage

[19] See Bueno, Menzel, & Zalta 2014 for a discussion of how the theory of propositions and possible worlds is completely 'set free' in object theory.

[20] From the present standpoint, the semantics of the language of typed object object doesn't provide any further theoretical understanding of the primitive relations that populate the domains $\mathcal{D}_\sigma$, for $\sigma \neq i$. Indeed, metaphysically, the language of set theory used in a typical model-theoretic semantics can be analyzed within typed object theory. But, of course, if we allow ourselves some set theory and urelements, we can develop a model-theoretic semantics for the language of typed object theory. See Zalta 1983 and 1988a.

pattern of semantically distinguishing intensions and extensions back to Frege's distinction between sense and reference. He writes (2007, 101):

> Thus, while opinions about the nature of intensions radically diverge, all proposals follow a simple two-stage pattern. The aim of this paper is not to add one more theory of intension to the proposals that have already been made, but is an investigation of their common underlying logic. The idea will be that the two-stage set-up is essentially all that is needed to obtain intensionality. For the purposes of logic it suffices to consider intensions as abstract objects; the question what intensions are, while philosophically important, can be abstracted from.

His general system for studying RTT under intensional interpretations, which he calls ITL ('Intensional Type Logic'), has some very interesting properties.

One basic difference between ITL and typed object theory concerns the language. ITL doesn't have a primitive form of *predication*, whereas typed object theory has two. Instead, ITL has primitive function application of the form (AB); where A is a relational term of type $\langle \sigma_1, \ldots, \sigma_n\rangle$ and B is a term of type σ_1, (AB) is a term of type $\langle \sigma_2, \ldots, \sigma_n\rangle$. So, in effect, ITL doesn't treat statements of the form x loves y, $x \in y$, $x \leq y$, etc., as instances of the *primitive* form of predication Rxy. Instead, the semantics shows that relations are treated functionally. Muskens explains the semantic clause that assigns a value to terms of the form (AB) as follows:[21]

> To better understand the motivation behind the second and third clauses of this definition, it may help to consider that any $n + 1$ place relation R can be thought of as a unary function F such that $F(d) = \{\langle \vec{d}\rangle \mid \langle d, \vec{d}\rangle \in R\}$.

So it seems clear that Muskens, like Orey, Church, and Gallin, is not taking relations in RTT as primitive entities.

Putting this aside, the general logical framework Muskens develops for RTT has a number of virtues. As he explains, the framework allows us to distinguish formulas that are 'co-entailing' (2007, 113), allows us to represent the sense/reference distinction (2007, 114), and allows for a construction of possible worlds (2007, 115).

[21] The semantic clause in question is (2007, 104):

$$V(a, AB) = \{\langle \vec{d}\rangle \mid \langle I(a, B), \vec{d}\rangle \in V(a, A)\}$$

Since $V(a, X)$ is generally defined to be the extension of the intension of X, I would gloss the above as follows: the extension of the intension of (AB) is the set of $n - 1$ tuples obtained by removing the first member of each n-tuple in the extension of the intension of A whose first member is the intension of B.

However, I think that typed object theory offers somewhat more general analyses of the same phenomena precisely because it provides a theory of the intensions that populate the domains of relational types. First, as noted earlier, typed object theory distinguishes properties and relations from their Montagovian intensions. Let us return to the examples of the two properties that are distinct but necessarily equivalent: $[\lambda x\, Bx \,\&\, \forall y(Sxy \equiv \neg Syy)]$ ('being a barber who shaves all and only those who don't shave themselves') is a property of type $\langle i\rangle$, as is $[\lambda x\, Dx \,\&\, Wx \,\&\neg Wx]$ ('being a dog that is white and non-white'). In these expressions, 'B', 'D', and 'W' are of type $\langle i\rangle$, and 'S' is of type $\langle i, i\rangle$. Now let:

- s ('Sally') denote an individual of type i,
- j ('John') denote an individual of type i,
- $Bel(\ ,\ )$ ('believes') denote a relation of type $\langle i, \langle\,\rangle\rangle$,
- $[\lambda\ \varphi]$ ('that φ') denote a proposition of type $\langle\,\rangle$, provided φ has no encoding subformulas

Then, in typed object theory, the claim:

> Sally believes that John is a barber who shaves all and only those who don't shave themselves, i.e.,
> $Bel(s, [\lambda\, [\lambda x\, Bx \,\&\, \forall y(Sxy \equiv \neg Syy)]\, j])$

doesn't imply:

> Sally believes that John is a dog that is both white and not white, i.e.,
> $Bel(s, [\lambda\, [\lambda x\, Dx \,\&\, Wx \,\&\neg Wx]\, j])$

Typed object theory has had this feature from its inception, but with principles that articulate existence and identity conditions for the intensions that play a role in the above representations, namely, relations *Bel* and S, properties B, D, and W, and propositions $[\lambda[\lambda xBx\&\forall y(Sxy \equiv \neg Syy)]\, j]$ and $[\lambda[\lambda xDx\&Wx\&\neg Wx]\, j]$.

Moreover, to avoid the problems of hyperintensionality, which requires us to explain why the fact that John believes that Cicero is a Roman doesn't imply that John believes that Tully is a Roman, Muskens has to interpret proper names as (higher-order) properties of properties, and justify this by appeal to a principle of the 'primacy of properties' (2007, 114). By contrast, typed object theory treats 'Cicero' and 'Tully' as names that denote individuals. It then

uses a completely general analysis of Fregean senses to explain the problems of hyperintensionality. We sketch this analysis briefly.

In typed object theory, the sense of a natural language expression with type σ isn't an entity of higher type. Rather the sense is of the very same type. The sense of an expression of type σ is an abstract object of type σ, i.e., one that encodes properties with type $\langle\sigma\rangle$. By encoding properties of σ-type objects, the sense can *represent* an object that exemplifies the properties in question, though object theory doesn't require that sense determines reference! Indeed, object theory allows the sense of an expression to vary from person to person and that for many expressions, the sense of that expression for a person can encode properties that the object denoted by the expression fails to exemplify. But, in what follows, we suppress this feature of the theory.

Consider type i expressions 'Samuel Clemens' ('c') and 'Mark Twain' ('t'), which are learned in different contexts. The denotation (extension) and the senses (intensions) of 'c' and 't' are of type i:

- 'c' and 't' denote the same ordinary individual
- The sense of 'c' and the sense of 't' are distinct abstract individuals.

In typed object theory, we may represent the sense of 'c' and 't' as '$\underline{c}$' and '$\underline{t}$', respectively.[22] We then have a way to model Frege's solution to the problem of cognitive significance of identity statements for proper names. 'Cicero is Cicero' ($c = c$) is knowable *a priori*, whereas 'Cicero is Tully' ('$c = t$') is true but not knowable *a priori*: 'Cicero' and 'Tully' are expressions that have the same denotation (namely, Cicero) and different senses (namely $\underline{c}$ and $\underline{t}$).

Moreover, this analysis generalizes to all higher types. The sense of an expression of type $\langle\sigma_1, \ldots, \sigma_n\rangle$ is an abstract object of that very type. There is no type-raising. Consider, for example, the type $\langle i\rangle$ expressions 'woodchuck' ('W') and 'groundhog' ('G'), which are learned in different contexts. The denotation (extension) and the senses (intensions) of 'W' and 'G' are of type $\langle i\rangle$:

- 'W' and 'G' denote the same property of individuals
- The sense of 'W' and the sense of 'G' are distinct abstract properties – they encode different properties of properties.

[22] These can be indexed to persons and times or contexts, but as noted previously, we'll omit this relativization. The important point is that, as abstract objects, $\underline{c}$ and $\underline{t}$ can encode properties that Cicero exemplifies. (Or not, if one really wants to have a better understanding of how language works, as opposed to simply following Frege in discussing an ideal language.)

We may represent the sense of 'W' and 'G' in typed object theory as '$\underline{W}$' and '$\underline{G}$', respectively, again suppressing possible indices to persons, times or contexts. This provides a Fregean solution to the problem of the cognitive significance of identities: whereas 'being a woodchuck is identical to being a woodchuck' ('$W = W$') is knowable *a priori*, 'being a woodchuck is identical to being a groundhog' ('$W = G$') is not; the expressions 'being a woodchuck' and 'being a groundhog' have the same denotation but different senses.

Thus, we may explain hyperintensionality both at the level of individuals and at every higher type. At the level of individuals, we represent belief reports in terms of an ambiguity: the expressions 'Cicero' and 'Tully' contribute their denotations on the *de re* readings, but contribute their senses on the *de dicto* readings:

- John believes that Cicero is a Roman.
  (1) $B(j, [\lambda\ Rc])$ (*de re*)
  (2) $B(j, [\lambda\ R\underline{c}])$ (*de dicto*)

- John doesn't believe that Tully is a Roman.
  (3) $\neg B(j, [\lambda\ Rt])$ (*de re*)
  (4) $\neg B(j, [\lambda\ R\underline{t}])$ (*de dicto*)

- Cicero is Tully.
  (5) $c = t$

We explain the hyperintensionality by the fact that the *de dicto* readings (2) and (4) are consistent, even given (5).

The very same explanation can be given for hyperintensionality of the woodchuck/groundhog case. The expressions 'woodchuck' and 'groundhog' contribute their denotations on the *de re* readings, but contribute their senses on the *de dicto* readings:

- John believes that Woody is a woodchuck.
  (6) $B(j, [\lambda\ Ww])$ (*de re*)
  (7) $B(j, [\lambda\ \underline{W}w])$ (*de dicto*)

- John doesn't believe that Woody is a groundhog.
  (8) $\neg B(j, [\lambda\ Gw])$ (*de re*)
  (9) $\neg B(j, [\lambda\ \underline{G}w])$ (*de dicto*)

- Being a woodchuck just is being a groundhog.
  (10) $W = G$

Again, the *de dicto* readings (7) and (9) are consistent, even given (10). Note also that object theory even offers the reading $B(j, [\lambda\ \underline{W}\,\underline{w}])$, in which both the sense of the individual term and the sense of the predicate are combined in the proposition that is the object of belief. The consequences of these readings were developed in other works on object theory; see Zalta 1988a (166–172); and 2001 (337–341).

One final point of comparison with Muskens 2007 is in order. Muskens suggests that possible worlds can be constructed in ITL as properties of propositions.[23] Using the ITL variables w of type $\langle\langle\ \rangle\rangle$ (i.e., the type for properties of propositions) to range over possible worlds, he extends the system to include (2007, 115):

- a new primitive predicate Ω ('is a world') with type $\langle\langle\langle\ \rangle\rangle\rangle$,

- axioms that assert (1) if w is a world, then the false proposition ($\bot$) is not true at w, and (2) if w is a world, then if a conditional $A \rightarrow B$ is true at w, then for any objects, if it is true at w that A characterizes those objects, then it is true at w that B does too,

- a new primitive constant w_0 to designate the *actual world*, and

- axioms that govern w_0, which stipulate that w_0 is a possible world and that all and only true propositions are true at w_0

These basic features come with the nice feature that the notion, proposition *p is true at* world w, is just defined as wp (i.e., the result of applying w to p).

By constrast, object theory doesn't need a new primitive predicate for possible worlds. Possible worlds are defined as *situations*, which are in turn defined as abstract objects that encode propositions by encoding only propositional properties (Zalta 1983 (IV); Zalta 1993; and Menzel & Zalta 2014). A *possible world* is defined as any abstract object that might be such that it encodes all and only true propositions. Moreover, *truth at a world* is defined in terms of encoding: p is true at w (written $w \models p$) if and only if w encodes *being such that p*, i.e., if and

[23] Some of the following observations, suitably adjusted, apply to the reconstruction of possible worlds in the FTTs articulated in Fox & Lappin 2001 (184–7), and Pollard 2005 (41–3), 2008 (276–7). These authors assume a domain of primitive propositions structured as a pre-Boolean algebra or prelattice and then define possible worlds as ultrafilters (maximal prime filters) on this domain. This is clearly a *model* of possible worlds and *truth at a world*, not a theory of these notions. The propositions in a set do not characterize the set in any way. By contrast, possible worlds that *encode* propositional properties are characterized by these properties, since encoding is a mode of predication.

only if $w[\lambda y p]$. An actual world is then defined to be a possible world w such that $\forall p((w \models p) \equiv p)$. I won't rehearse these definitions in detail here, but merely assert that the *axioms* Muskens asserts to construct possible worlds are *theorems* of object theory. It is provable in object theory that: (1) no contradiction is true at any world, (2) that if $w \models (p \rightarrow q)$ and $w \models p$, then $w \models q$, (3) every world is maximal (i.e., for any w and any proposition p, either $(w \models p) \vee (w \models \neg p)$, and (4) there is a unique actual world (see, e.g., Zalta 1993).

Moreover, object theory is developed in a modal setting. So, its theory of worlds also yields the following claims as theorems:

$$\forall p(\Box p \equiv \forall w(w \models p))$$

$$\forall p(\Diamond p \equiv \exists w(w \models p))$$

Thus, the object-theoretic analysis of worlds derives the fundamental facts about possible worlds as theorems: a proposition is necessarily true if and only if it is true in all possible worlds, and proposition is possibly true if and only if it is true in some possible world. These principles draw a deep connection between our pre-theoretical understanding of necessity and possibility and our theoretical understanding of possible worlds. With such principles as theorems, all we have to do to prove the existence of non-actual possible worlds is to assert, for some proposition p, that $\neg p \,\&\, \Diamond p$, for it then follows that there exists a possible world distinct from the actual world where p is true. It is not clear whether this connection between our pre-theoretic understanding of modality and our theoretical understanding of possible worlds is preserved by the analysis of the modal operators we find in Muskens 2007 (116).[24]

I conclude this section with two further observations. The first is that object theory doesn't need any special new axioms to develop the theory of *impossible* worlds: an impossible world i is a situation that is *maximal* and such that it is not possible that every proposition true in i is true, i.e., $\neg\Diamond\forall p((i \models p) \rightarrow p)$. So, in the special cases of hyperintensionality where impossible worlds are needed

[24] I say this because Muskens has to stipulate the axioms he labels W3 and W4, which assert that when *truth at a world* and *being a world* hold of the appropriate objects, they hold by necessity. By constrast, object theory yields these claims as theorems: one can prove in object theory that $(w \models p) \rightarrow \Box(w \models p)$ and that $PossibleWorld(w) \rightarrow \Box PossibleWorld(w)$. These facts hold because both the notions of *truth at a world* and *possible world* are defined in terms of encoding formulas, which are governed by the axiom $xF \rightarrow \Box xF$. Also, it looks like the analysis Muskens offers (2007, 116) has to build the fundamental principles connecting modality and *truth at* a world into the R (accessibility) relation, so that principles like the ones being discussed in the text end up just being *defined into* the modal operators, thereby making the principles definitional truths. In object theory, the principles in question aren't simply true by definition.

(e.g., for counterfactuals with impossible antecedents), the theory provides the background theoretical entities needed for the analysis to proceed.

Second, there may be an issue with Muskens' reconstruction of possible worlds. If worlds are, as he says, properties of propositions, and properties are intensional entities, then he may have *too many* possible worlds. This is clearest in the case of the actual world w_0. if w_0 is a property of propositions, then consider any property of propositions that is distinct from w_0 but necessarily equivalent to it. Then we would have *two* distinct actual worlds. In other words, his definitions and axioms don't guarantee that there exists a unique actual world. By contrast, in object theory, it is provable that there is a unique actual world (i.e., there is a unique abstract object that is a possible world and is actual, namely, the abstract object that encodes all and only the properties of the form $[\lambda y\ p]$, where p is a true proposition. The problem of *too many worlds* affects other well-known attempts to define possible worlds as fine-grained intensional entities such as *states of affairs* (see Zalta 1988a, 72–74, for further discussion).

3 Nominalized Propositions

Recently, some linguists have focused on the fact that, in natural language, expressions that denote propositions can occur in sentence positions where expressions that appear to denote individuals can occur. The expressions in question are referred to as complement phrases (CP) and determiner phrases (DP), respectively, and the sentences that have positions where both CPs and DPs can occur may be called CP/DP-neutral constructions. Liefke (2014) and Liefke & Werning (forthcoming) compile a wide variety of these and other similar constructions. Here are just a few examples:

1. DP/CP neutrality:
   a. Mary noticed [$_{DP}$ Bill].
   b. Mary noticed [$_{CP}$ that Bill waiting for Pat].

2. DP/CP coordinability:
   a. Mary remembered [$_{DP}$ Bill] and [$_{CP}$ that Bill was waiting for Pat].

3. CP nominalization:
   a. [$_{DP}$ Mary] bothered Bill.
   b. [$_{CP}$ That Pat was so evasive] bothered Bill.

4. DP/CP equatability:
   a. [$_{DP}$ The problem] was [$_{DP}$ Pat's dislike of Bill].
   b. [$_{DP}$ The problem] was [$_{CP}$ that Pat did not like Bill.]

5. Proposition-type anaphora:
   a. Mary told John [$_{CP}$ that it was raining]. John did not believe [$_{PRO}$ it].

Whereas Partee (2009) uses such constructions to question the distinction between the primitive types e and t in FTT, Liefke 2014 and Liefke & Werning (forthcoming) conclude that such constructions (and others) provide evidence for developing a semantics for natural language based on single primitive type o, which is nevertheless to be interpreted as a higher Montagovian type $(s,(s,t))$. In Liefke 2014 (18, 86, 97, 163), this higher type is understood to be that of *propositional concepts*, i.e., functions from possible worlds to Montagovian propositions. By contrast, Liefke & Werning (forthcoming) interpret the type $(s,(s,t))$ as the type for functions from contextually specified situations to sets of situations (forthcoming, Section 4).[25] They then (Section 4) interpret both CPs and DPs in the higher type $(s,(s,t))$. But to give this analysis, they must introduce the notions of *situation*, *contextual specification of a situation*, and *situative proposition*, and invoke a rich ontology that includes worlds, times, locations, situations, inhabitants of situations, situative propositions, etc. For the most part, they stipulate the structure that is needed, e.g., a partial ordering $\sqsubseteq$ ('inclusion') on a set of situations, with top and bottom elements, etc. (Section 4).

In typed object theory, one can offer an alternative analysis of the linguistic data, namely, that the constructions involve nominalized propositions, i.e., abstract individuals that are defined by, and so correspond to, propositions. Typed object theory has a natural way to do this: for each proposition p, there is an abstract individual of type i that is the nominalization of p. Let p be any proposition, i.e., entity of type $\langle\,\rangle$, x be a variable ranging over individuals, $A!$ (*being abstract*) be a property of individuals and F a variable ranging over properties of individuals. Then object theory guarantees that the following definition picks out a canonical individual, $\check{}p$, which we may call *the nominalization of p*:

$$\check{}p \;=_{df}\; \iota x(A!x \,\&\, \forall F(xF \equiv F=[\lambda y\, p]))$$

This identifies the nominalization of p as the abstract individual that encodes just the property *being such that p* (i.e., encodes just $[\lambda y\, p]$). Given such a definition, we may interpret sentences like the ones above as giving rise to contexts in which the DPs and CPs both denote individuals. For example, *notice* can be a verb of type $\langle i,i\rangle$, so that (1.a) and (1.b) above can be analyzed, respectively, as follows, where W is the relation *waiting for*:

$$N(m,b)$$

$$N(m,\check{}[\lambda\, Wbp])$$

In the second case, *notice* relates Mary to the nominalization of the proposition *that Bill was waiting for Pat*. Thus, instead of type-raising, object theory *lowers* the relational type for propositions $\langle\,\rangle$ to the type for individuals!

This kind of solution then generalizes to the other cases, though one may have to apply certain operations on individuals, for example, to analyze the compound individuals such as the conjunction of the individual Bill and the nominalization of the proposition *that Bill was waiting for Pat* (to handle examples like 2.a).[26]

This ability to nominalize propositions in typed object theory is similar to its ability to nominalize properties. Suppose G is a property of individuals, i.e., of type $\langle i\rangle$. Then we may define the nominalization of G, written $\check{}G$, as follows:

$$\check{}G = \iota x(A!x \,\&\, \forall F(xF \equiv F=G))$$

In other words, the nominalization of G is the abstract object that encodes just G and no other properties. This allows the semanticist to give a uniform analysis of the sentences:

John is fun.
Fj

Running is fun.
$F\check{}R$

[25] I shall continue to use parenthesis to denote derived, functional types in FTT, and use angled brackets to denote derived, relational types in RTT. But the reader should note that Liefke & Werning use angled brackets for Montagovian functional types.

[26] Typed object theory provides such compound individuals. Where y denotes any individual and $\check{}p$ denotes the individual which is the nominalization of the proposition p, object theory asserts that there is an *intersect* object, $y \wedge \check{}p$, that encodes exactly the properties that y and $\check{}p$ exemplify in common:

$$y \wedge \check{}p \;=_{df}\; \iota x(A!x \,\&\, \forall F(xF \equiv Fy \,\&\, F\check{}p))$$

as well as a *union* object, $y \vee \check{}p$, that encodes all and only the properties exemplified by either y or $\check{}p$:

$$y \vee \check{}p \;=_{df}\; \iota x(A!x \,\&\, \forall F(xF \equiv Fy \vee F\check{}p))$$

In these representations, both 'John' and 'Running' denote individuals, though the latter is an abstract individual. Similarly, if G is a 3-place relation among individuals, e.g., *x gives y* to *z*, then we can identify its nominalization (*giving*) as the nominalization of the property that results by existentially projecting G to $[\lambda x\ \exists y \exists z Gxyz]$, i.e., as $^{\vee}[\lambda x\ \exists y \exists z Gxyz]$. So if *being rewarding* (R) is a property of individuals, we have the following analysis:

> Giving is rewarding.
> $R^{\vee}[\lambda x\ \exists y \exists z Gxyz]$

Note that in this analysis, no type-raising is involved.

Thus, where FTT systems often use type-raising (type-lifting) techniques to unify the analysis of natural language, typed object theory can analyze many constructions without such techniques. We've already seen some examples. Type-raising isn't needed for the analysis of the intensions of natural language expressions, nor for the nominalizations of propositions and properties. Consider also the classic FTT analysis of using generalized quantifiers to unify the noun phrases 'John' and 'every person' by type-raising. Both expressions are often analyzed extensionally in FTT systems as denoting a set of properties of individuals, i.e., as $\{F \mid F j\}$ and $\{F \mid \forall x(Px \rightarrow Fx)\}$, respectively. But in object theory, type-raising isn't needed: both expressions have type i. 'John' denotes an individual and 'every person' can denote the following individual:

$$\iota x(A!x \,\&\, \forall F(xF \equiv \forall y(Py \rightarrow Fy)))$$

So 'every person' would denote the abstract object that encodes all the properties exemplified by every person.

Nor does type-raising help with fictions. In FTT systems, it is suggested that 'Sherlock Holmes' ('h') denotes $\{F \mid Fh\}$, i.e., a set of properties. But if that set is to be something other than the empty set, 'h' must have a denotation. Object theory provides such a denotation:

$$h = \iota x(A!x \,\&\, \forall F(xF \equiv CD \models Fh))$$

This identifies Holmes the abstract object that encodes exactly the properties F such that, in the Conan Doyle novels, Holmes exemplifies F.[27] Thus, Holmes is identified on the basis of the body of story-truths of the form: in the Conan Doyle

[27] Note here that 'In the Conan Doyle novels, p' has been represented as a claim of the form '$s \models p$' ('p is true in s'), where s is a situation and p is a proposition. So truth in a situation is given the same analysis as truth at a world, namely, as $s[\lambda y\ p]$, i.e., s encodes the propositional property *being such that p*.

novels, Holmes is F. By being abstract, Holmes is not a possibly concrete object. As Kripke noted, there are too many complete, possible objects consistent with the novels (supposing the novels are consistent). Holmes is an individual that is *incomplete* with respect to his encoded properties, but complete with respect to his exemplified properties. Given that the English copula is ambiguous between encoding and exemplification predication, we may say that Holmes 'is' a detective in the sense of *encodes*, but fails to exemplify detectivehood.

This analysis extends to fictional properties of individuals, such as *being a hobbit* (H). We don't need type-raising to interpret the predicate 'hobbit', for its analysis is similar to the analysis of names of fictional individuals: H denotes an abstract property of individuals, i.e., an abstract property with type $\langle i \rangle$. An abstract property of individuals encodes properties of properties of individuals. So where H is of type $\langle i \rangle$, x is a variable of type $\langle i \rangle$, and $A!$ is a constant and F a variable of type $\langle\langle i \rangle\rangle$, we may identify *being a hobbit* as follows:

$$H = \iota x(A!x \,\&\, \forall F(xF \equiv \mathit{LordOfTheRings} \models FH))$$

That is, *being a hobbit* is the abstract property of individuals that encodes exactly the properties of properties of individuals that *being a hobbit* exemplifies in *The Lord of the Rings*.[28]

In summary, then, typed object theory avoids type-lifting by taking advantage of the abstract objects that exist at each type. It is based on RTT with a single primitive type and offers a natural way to define situations, possible worlds, fictional entities, etc. These entities have precise definitions and the main principles governing them can be derived. Propositions and properties both, no matter whether simple or complex, have nominalizations, and we need not interpret sentence positions that are neutral with respect to CPs and DPs as positions requiring a higher type.

4 Intensional Type Theory: II

We now turn to a discussion that compares typed object theory to the intensional framework developed in Williamson 2013. We begin by showing that an argument Williamson raises against general (Henkin) models can be undermined.

[28] Strictly speaking, in this analysis of 'hobbit' and in the analysis of the name 'Holmes', we need to index the term being analyzed to the story in question. So we should use H_{LOTR} and h_{CD} on both sides of the identity symbol in the respective principles.

4.1 Response to an Argument Against Henkin Models

In 2013 (226–230), Williamson develops an extended argument that is designed to show the superiority of *standard* models of higher-order logic to *general* (Henkin) models. In this argument he distinguishes the standard notions of logical consequence and validity from the analogous notions, *g-logical consequence* and *g-validity*, that apply to general models. Williamson begins by asserting (2013, 226):

> Despite the formal tractability of g-logical consequence, general models are more complex and less natural than standard models. Why have arbitrary restrictions on the permissible intensions of the appropriate type for a predicate?

Though Williamson goes on to give an example, the second sentence in this opening statement betrays a presupposition that is rejected by typed object theory, namely, that the *permissible* intensions are those that are given by possible world semantics, in which relations are identified as set-theoretically defined functions from worlds to sets of n-tuples. This presupposes that set theory and set membership offer a more fundamental account of relations than a direct, axiomatic theory of relations and predication. But object theory has no such presupposition. From the point of view of typed object theory, we should reverse the order: the permissible intensions are those that are given by a mathematically precise theory of relations, such as the one offered by (typed) object theory. If there is nothing more fundamental than individuals, relations, and predication, why suppose that set theory with possible worlds as urelements gives us a greater insight as to what relations or intensions exist?

Once we recognize this presupposition in Williamson's argument, it becomes easy to undermine the other reasons Williamson gives for preferring standard models to general models. He notes, for example, that the Comprehension Principle is standardly valid, but not g-valid (2013, 228):

> By contrast, in some general models, $\mathrm{dom}(\langle t_1, \ldots, t_n \rangle)$ omits the intensions of $\lambda \nu_1 \ldots \nu_n(A)$ needed as a value of of V to verify an instance of Comp, so Comp is not g-valid.

But this offers no reason why standard models should be preferred. Why let the semantics drive a precise theory of relations? Instead, the comprehension principle for relations, which is derivable in object theory (suitably restricted to exclude encoding subformulas from allowable matrices), should drive the semantics. This comprehension principle tells us the conditions under which

relations exist. I suspect that the reason Williamson doesn't consider it definitive is the same one we encountered before in trying to understand why FTT rather than RTT became standard in linguistics: without encoding formulas to give precise identity conditions for relations, Williamson has no theory of relation identity to fall back on, other than the set-theoretic reconstruction of them as Montagovian intensions. But, from the point of view of typed object theory, a general model is sufficient if it makes the comprehension principle for relations valid. Such general models would then include everything needed to show that the theory of relations is consistent.

Williamson discusses this option (2013, 229):

> We could add Comp as an extra principle to Gallin's axiomatic system presented earlier, and restrict the general models to those in which it is valid. Of course, the resulting logic would still have a recursively axiomatizable set of theorems, and so be weaker than the standard logic. Even a general model that validates Comp may have highly restricted intensions for most types because many intensions correspond to no formula of the language, relative to any values of its parameters.

But again, Williamson assumes that general models would have 'highly restricted intensions', because he supposes that the *intensions* are given by set-theoretic functions from possible worlds to sets of n-tuples. We should not, however, accept such a prior characterization of intensions. That is to give the conception of intensions derived from set-theory preference over the conception of intensions derived from metaphysical considerations. Object theory starts with a primitive, fine-grained notion of relations; these are *more fine-grained* than set-theoretic functions from worlds to sets of n-tuples: we saw in Sections 1.1 and 1.2 that while Montagovian intensions collapse necessarily equivalent relations, the identity conditions for relations in typed object theory do not.

Williamson next charges that the structure of standard models, but not g-models, is what our metaphysics *should* characterize (2013, 229–230):

> Thus non-standard models also differ from standard ones in respects relevant to the evaluation of claims about purely logical structure, in the sense of claims expressed by formulas without non-logical constants. But logical structure is what the logical core of our metaphysics is supposed to characterize. ... Hence a g-logic is less informative than standard logic about purely logical structure. A metaphysical theory based on g-logic rather than standard logic is neutral on many of the very questions it is supposed to answer.

But I would reply that it is exactly the neutrality of *g*-models that prevents it from falling into the obvious error of standard models, namely, the error of collapsing relations, properties, and propositions that are necessarily equivalent. A *g*-logic *should* remain neutral on many questions that should be decided on the basis of theory, not on the basis of the *set-theoretic artifacts* of a standard model.

Finally, Williamson claims (2013, 230):

> Moreover, to the extent to which we take models for $\mathtt{ML_P}$ seriously, the standard ones are more faithful than the non-standard ones to our intended interpretation.

This strikes me as rather controversial. Given the precise theory of relations offered in (typed) object theory, how could standard models based on Montagovian intensions, which collapse relations that can be kept apart in *g*-models of object theory, be more faithful?

4.2 Comparison of the Ontologies

If we put his argument against *g*-models aside, though, there are some interesting points of comparison between the typed object theory and Williamson's intensional logic. One is that typed object theory uses one set of types for both its syntax and semantics. Williamson, by contrast, uses one set of types for the syntax of his language and a different set of types for its semantics. For the syntax, he uses the standard RTT types, though using e as the primitive type for individuals. For the semantics, he adds w as a second base type, and then defines a new type $\tau\sigma$ from each σ in the syntactic hierarchy, as follows (2013, 236–7):

> Each type t of $\mathtt{ML_P}$ corresponds to a type τt of the metalanguage by the rule τe and $\tau\langle t_1, \ldots, t_n\rangle$ is $\langle \tau t_1, \ldots, \tau t_n, w\rangle$. But we add a cumulative infinite limit type λ to the metalanguage: the expressions of type λ are exactly those of any finite type. Thus expressions of type λ belong to some more specific type, but expressions of type $\langle\lambda\rangle$ do not.

If I'm understanding this correctly, then this typing scheme, unlike that of typed object theory, essentially takes the entities denoted by n-place relational predicates to be $n + 1$-place relations and requires the metaphysician to regard relations essentially as world-indexed entities. This fundamentally changes the way in which relations are to be conceived and such a change is not required by typed object theory. Williamson would no doubt justify the proposal by citing the advantages of the semantics he goes on to give (237–8) (i.e., a kind of homophonic semantics in which quantification can be conceived without domains

and as unrestricted). But that semantics, as Williamson admits, requires a *plural* conception of higher-order quantifiers, something that doesn't easily generalize to relations, given that there seems to be no natural way to render quantification over relations in the plural idiom.

Moreover, Williamson's world-indexed relations leave open a variety of questions. If F and G are variables for properties with type $\langle\sigma\rangle$ and x a variable for an object of type σ, do the world-indexed relations obey the law: $\forall x \forall w(Fxw \equiv Gxw) \rightarrow F = G$? What is the denotation of complex λ-expressions in the semantics that Williamson develops? His semantics (2013, 238) doesn't say.

By contrast, typed object theory simply rests with its axiomatic foundations; there are axioms for quantification, axioms for relations, and definitions and theorems governing possible worlds. No set-theoretic model of such a system gives any deeper insight into the nature of the entities being modeled. One should not mistake the entities in such models or the artifactual set-theoretic domains of the models for the entities and notions being modeled.

5 Conclusion

It might be thought that RTT, despite its elegance in having a single primitive type, can be reduced to FTT. But Oppenheimer & Zalta (2011) show that FTT has no straightforward way of representing the logic of typed object theory as the latter is formulated in RTT. This suggests that RTT is the more general framework. Basically, we noted that in FTT, every formula can be converted into a term. The semantics of the quantified formula $\forall x^\sigma \varphi$ is handled by converting φ to $[\lambda x^\sigma \varphi]$, which is a function that maps objects of type σ to a truth value. Then $\forall$ is interpreted as a particular function that maps the expression $[\lambda x^\sigma \varphi]$ to a truth value. In particular, $\forall$ maps $[\lambda x^\sigma\ \varphi]$ to The True, i.e., $\forall x^\sigma \varphi$ is true, just in case the function $[\lambda x^\sigma \varphi]$ maps every object to The True.

But in typed object theory, formulas with encoding subformulas *can't* be converted to terms, on pain of paradox. The formula $xF\ \&\ \neg Fx$ can't be converted to $[\lambda x\, xF\ \&\ \neg Fx]$ since the latter is not even well-formed in object theory. Oppenheimer & Zalta (2011) point out that FTT can't therefore interpret the expression $\forall F(xF\ \&\ \neg Fx)$ by applying the higher order function $\forall$ to the predicate $[\lambda x\ (xF\ \&\ \neg Fx)]$, since the latter isn't in the language.

These considerations, as well as the ones presented earlier in the body of this paper, may prove helpful when comparing the relative merits of FTT and RTT systems for the analysis of natural language, and when comparing foundations

that take relations and predication as basic, instead of functions and function application or sets and set membership.

Bibliography

Anderson, C.A., 1989, "Russellian Intensional Logic", in J. Almog, J. Perry, and H. Wettstein (eds.), *Themes from Kaplan*, New York: Oxford University Press, 67–103.

Anderson, D.J., and E. Zalta, 2004, "Frege, Boolos, and Logical Objects", *Journal of Philosophical Logic*, 33(1): 1–26.

Boolos, G., 1987, "The Consistency of Frege's *Foundations of Arithmetic*", in J. Thomson (ed.), *On Being and Saying*, Cambridge, MA: MIT Press; reprinted in G. Boolos, *Logic, Logic, and Logic*, J. Burgess and R. Jeffrey (eds.), Cambridge, MA: Harvard University Press, 1998, 183–201.

Bueno, O., Menzel, C., and E. Zalta, 2014,"Worlds and Propositions Set Free", *Erkenntnis*, 79: 797–820.

Carnap, R., 1929, *Abriss der Logistik*, Wien: J. Springer.

Church, A., 1940, "A Formulation of the Simple Theory of Types", *The Journal of Symbolic Logic*, 5(2): 56–68.

Church, A., 1974, "Russellian Simple Type Theory", *Proceedings and Addresses of the American Philosophical Association*, 47: 21–33.

Cresswell, M., 1975, "Hyperintensional logic", *Studia Logica*, 34: 25–38.

Cresswell, M., 1985, *Structured Meanings: The Semantics of Propositional Attitudes*, Bradford Books/MIT Press.

Fox, C., and S. Lappin, 2001, "A Framework for the Hyperintensional Semantics of Natural Language with Two Implementations", in P. de Groote, G. Morrill, and C. Restoré (eds.), in *LACL 2001: Proceedings of the 4th International Conference on Logical Aspects of Computational Linguistics, Le Croisic, June 2729, 2001* (Lecture Notes in Artificial Intelligence 2099), Berlin: Springer, 175–92.

Gallin, D., 1975, *Intensional and Higher-Order Modal Logic: With Applications to Montague Semantics*, Amsterdam: North-Holland.

Kripke, S., 1973 [2013], *Reference and Existence: The John Locke Lectures*, Oxford: Oxford University Press, 2013.

Liefke, K., 2014, *A Single-Type Semantics for Natural Language*, Enschede: Ipskamp Drukkers.

Liefke, K., and M. Werning, forthcoming, "Evidence For Single-Type Semantics – An Alternative To *e*/*t*-Based Dual-Type Semantics", *Journal of Semantics*, first online 15 August 2018, doi:10.1093/jos/ffy009

Linsky, B., and E. Zalta, 1995, "Naturalized Platonism vs. Platonized Naturalism", *The Journal of Philosophy*, 92(10): 525–555.

Meinwald, C., 1992, "Good-bye to the Third Man", in *The Cambridge Companion to Plato*, R. Kraut (ed.), Cambridge: Cambridge University Press, 365–396.

Menzel, C., and E. Zalta, 2014, "The Fundamental Theorem of World Theory", *Journal of Philosophical Logic*, 43(2): 333–363.

Montague, R., 1973, "The Proper Treatment of Quantification in Ordinary English", in J. Hintikka, J. Moravcsik, and P. Suppes (eds.), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, Dordrecht: D. Reidel, 221–242; reprinted in R. Thomason (ed.), *Formal Philosophy: Selected Papers of Richard Montague*, New Haven: Yale University Press, 1974, pp. 247–270. [Page reference is to the reprint.]

Muskens, R., 1989, "A Relational Formulation of the Theory of Types", *Linguistics and Philosophy*, 12: 325–346.

Muskens, R., 1995, *Meaning and Partiality*, Stanford: CSLI Publications.

Muskens, R., 2007, "Intensional Models for the Theory of Types", *The Journal of Symbolic Logic*, 72(1): 98–118.

Nodelman, U., and E. Zalta, 2014, "Foundations for Mathematical Structuralism", *Mind*, 123(489): 39–78.

Oppenheimer, P., and E. Zalta, 2011, "Relations Versus Functions at the Foundations of Logic: Type-Theoretic Considerations", *Journal of Logic and Computation*, 21: 351–374.

Orey, S., 1959, "Model Theory for the Higher Order Predicate Calculus", *Transactions of the American Mathematical Society*, 92: 72–84.

Partee, B., 2009 [2007], "Do we need two basic types?, *Snippets*, Issue 20 (Special issue in honor of Manfred Krifka, Sigrid Beck and Hans-Martin Gärtner, eds.), Milan: Edizioni Universitarie di Lettere Economia Diritto, pp. 37–41; see also "Type Theory and Natural Language: Do We Need Two Basic Types?", 2007 presentation handout, 100th Meeting of the Seminar: Mathematical Methods Applied to Linguistics, March 31, Moscow State University, URL = <http://people.umass.edu/partee/docs/TwoTypesHOMarch07.pdf>

Pelletier, F.J., and E. Zalta, 2000, "How to Say Goodbye to the Third Man", *Noûs*, 34(2): 165–202.

Pollard, C., 2005, "Hyperintensional Semantics in a Higher-Order Logic with Definable Subtypes", in M. Fernández, C. Fox, and S. Lappin (eds.), *Lambda Calculus, Type Theory, and Natural Language*, pp. 32-46, URL = https://pdfs.semanticscholar.org/2562/1a82b55c0d34f90c6b15ad83939c0e04ec1c.pdf

Pollard, C., 2008, "Hyperintensions", *Journal of Logic and Computation*, 18: 257–82.

Quine, W.V.O., 1960, "Variables Explained Away", *American Philosophical Society*, 104 (3): 343–347; reprinted in W.V.O. Quine, *Selected Logical Papers*, New York: Random House, 1966, 227–235.

Russell, B., 1908, "Mathematical Logic As Based On the Theory of Types", *American Journal of Mathematics*, 30(3): 222–262.

Schütte, K., 1960, "Syntactical and Semantical Properties of Simple Type Theory", *The Journal of Symbolic Logic*, 25(4): 305–326.

Thomason, R., 1980, "A model theory for propositional attitudes", *Linguistics & Philosophy*, 4: 47–70.

van Benthem, J. and K. Doets, 1983, "Higher-Order Logic", in D. Gabbay and F. Guenthner (eds.), *Handbook of Philosophical Logic* (Volume I), Dordrecht: D. Reidel, pp. 275–329.

Williamson, T., 2013, *Modal Logic as Metaphysics*, Oxford: Oxford University Press.

Zalta, E., 1982, "Meinongian Type Theory and Its Applications", *Studia Logica*, 41(2–3): 297–307.

———, 1983, *Abstract Objects: An Introduction to Axiomatic Metaphysics*, Dordrecht: D. Reidel.

———, 1988a, *Intensional Logic and the Metaphysics of Intentionality*, Cambridge, MA: MIT Press.

———, 1988b, "A Comparison of Two Intensional Logics", *Linguistics and Philosophy*, 11(1): 59–89.

———, 1993, "Twenty-Five Basic Theorems in Situation and World Theory", *Journal of Philosophical Logic*, 22: 385–428.

———, 1997, "A Classically-Based Theory of Impossible Worlds", *Notre Dame Journal of Formal Logic*, 38(4): 640–660.

———, 1999, "Natural Numbers and Natural Cardinals as Abstract Objects: A Partial Reconstruction of Frege's *Grundgesetze* in Object Theory", *Journal of Philosophical Logic*, 28(6): 619–660.

———, 2000a, "Neo-Logicism? An Ontological Reduction of Mathematics to Metaphysics", *Erkenntnis*, 53(1–2): 219–265.

———, 2000b, "The Road Between Pretense Theory and Object Theory", in *Empty Names, Fiction, and the Puzzles of Non-Existence*, A. Everett and T. Hofweber (eds.), Stanford: CSLI Publications, pp. 117–147.

———, 2000c, "A (Leibnizian) Theory of Concepts", *Philosophiegeschichte und logische Analyse / Logical Analysis and History of Philosophy*, 3: 137–183.

———, 2001, "Fregean Senses, Modes of Presentation, and Concepts", *Philosophical Perspectives* (*Noûs* Supplement), 15: 335–359.

———, 2006a, "Essence and Modality", *Mind*, 115(459): 659–693.

———, 2006b, "Deriving and Validating Kripkean Claims Using the Theory of Abstract Objects", *Noûs*, 40(4): 591–622.