

*On the so-called antinomy of the variable*  
Joshua Schwartz

## 1

The creation and use of quantifier-variable notations (e.g., “ $(\forall x)(Fx)$ ”), which launched the rapid development of mathematical logic in the 19<sup>th</sup> century, extinguished deeply entrenched scientific practices that associated variables with indeterminate quantities and varying magnitudes. W. V. Quine alludes to these past traditions at the start of his article on variables:

It used to be necessary to warn against the notion of variable numbers, variable quantities, variable objects, and to explain that the variable is purely a notation, admitting only fixed numbers or other fixed objects as its values. This dissociation now seems to be generally understood.<sup>1</sup>

But for some founders of the new logic, the dissociation that Quine mentions simply forced new problems to surface. Bertrand Russell, for example, in his early account of the subject, explicitly rejects the existence of variable objects yet courts the following quandary: “ $x$  is, in some sense, the object denoted by *any term*; yet this can hardly be strictly maintained, for different variables may occur in a proposition, yet the object denoted by *any term*, one would suppose, is unique.”<sup>2</sup> Russell’s views at the time compel him to postulate a unique object — the variable — denoted by what

---

<sup>1</sup>“The variable” in *The Ways of Paradox*, revised and enlarged edition (Cambridge, MA: Harvard UP, 1976), p. 272.

<sup>2</sup>*The Principles of Mathematics*, 2nd edition (London: Allen & Unwin, 1937), §93.

he calls the denoting concept *any term*. But this assumption evidently conflicts with the occurrence of distinct variables in a proposition.<sup>3</sup>

Contemporary philosophers would likely blame Russell for Russell’s difficulty with variables. How does he understand the phrase “an occurrence of a variable”? Or should we ask about “an occurrence of *the* variable”? Given rigorous treatments of strictly notational uses of “variable”, that is, precise descriptions of the linguistic role of certain letters, like the “*x*” from high school algebra, Russell’s ontological assumption that *the* variable exists seems quaint, needlessly abstruse, and, well, philosophical. As measured by the dearth of literature on the topic, Russellian worries pose no threat to understanding the variable insofar as we implicitly assume that the linguistic role of variables is completely transparent.

Kit Fine in his recent book shatters this complacency and argues that our implicit assumption about variables is unwarranted — a de-ontologized version of Russell’s difficulty threatens the cogency of our best technical treatments of the subject.<sup>4</sup> His case rests on the formulation of an apparent antinomy concerning the semantic role(s) of variables and an argument that three familiar approaches to the semantics of vari-

---

<sup>3</sup>Peter Hylton discusses Russell’s view of the variable (and the nature of denoting concepts) in *Russell, Idealism, and the Emergence of Analytic Philosophy* (New York: Oxford UP, 1990), Chapter 5, pp. 212–221.

<sup>4</sup>Fine, *Semantic Relationalism* (Malden, MA: Blackwell Publishing, 2007). The material on variables that I will discuss was published separately in an earlier article by Fine, “The role of variables” in *The Journal of Philosophy*, vol. C, no. 12 (December 2003), pp. 605–633. Fine opens the article with the following remark:

There are deep problems concerning the role of variables that have never been properly recognized, let alone solved, and once we attempt to solve them we see that they have profound implications not only for our understanding of variables but also for our understanding of other forms of expression and for the general nature of semantics.

He cites the Russell passage on page 607.

ables and quantified sentences fail to solve the problem, namely, what he calls the Tarskian, instantial, and algebraic approaches. In addition, Fine maintains that failure to solve the antinomy about variables is failure to provide a satisfactory semantics for first-order quantification theory. His positive contribution is a new approach to semantics that aims to solve the antinomy and yield a satisfactory semantics for first-order logic.<sup>5</sup>

I agree with Fine that there is more to the variable than meets the eye in standard formulations of first-order logic. But I disagree that we face anything like an antinomy of the variable. My rejection of the antinomy, a chief aim of this paper, will take the form of an interpretation and defense of a neglected strand of thought in Quine's philosophy, namely, his predicate-functor account of the variable. Quine's predicate-functor approach explains the linguistic role of the variable by removing the standard quantificational apparatus of variable-binding from first-order logic and considering the addition of surrogate variables which equal the standard apparatus in expressive power.<sup>6</sup> The surrogate variables which Quine devises are called predicate functors, operators which apply to primitive predicate constants or lexical items true of objects from some domain and which yield complex predicates defined over the same domain. What distinguishes the predicate-functor account from competitors is that it ties the linguistic role of the variable to the predicate. As Quine has it,

---

<sup>5</sup>Fine, "The role of variables", p. 605. Fine dubs his approach semantic relationalism. I will not discuss Fine's semantics in this paper, rather I will focus on his formulation of the antimony and his arguments against proposed solutions.

<sup>6</sup>The notion of a surrogate variable comes from William C. Purdy, "Surrogate variables in natural language" in *Variable-Free Semantics, Articulation and Language*, Michael Boettner and Wolf Thuemmel, eds., (University of Osnabrueck, 2000), Volume 3.

“[w]hat brings the variable, if any, is the predicate itself”, and elsewhere he writes that “the essential services of the variable are the permutation of predicate places and the linking of predicate places.”<sup>7</sup> On Quine’s predicate-functor analysis, variables are a uniform and convenient notational device for expressing properties of the argument structures of predicates. The nature of this connection between variables and predicates carries the burden of my reading of Quine (section **3**) and my argument against Fine (sections **2** and **4**). I show that Fine’s discussion of the variable suppresses the relevant connection, which then generates the appearance of antinomy, and that explicit recognition of the interaction between variables and predicates undermines his formulation of the antinomy. I begin with this formulation in section **2**.

## **2**

Strictly speaking, by Fine’s own lights, there is no antinomy of the variable. Rather, what appears as antinomy or explicit contradiction is a deeper more intractable puzzle about variables. In this section I will closely follow Fine’s exposition and discuss, first, his statement of the antinomy of the variable, second, its immediate resolution, and third, Fine’s formulation of the attendant, more worrisome puzzle about variables. At the close of the section, I will modify Fine’s statement of the puzzle. The changes which I propose stem from Fine’s own discussion but reveal a crucial assumption that underwrites his formulation of the puzzle.

Both the antinomy of the variable and the puzzle about variables rest on the

---

<sup>7</sup>“The variable”, p. 276 and “Algebraic Logic and Predicate Functors” in *The Ways of Paradox*, revised and enlarged edition (Cambridge, MA: Harvard UP, 1976), p. 304.

notion of the semantic role of a linguistic expression. Fine makes three points about his use of the phrase “semantic role”. First, semantic role is a nontechnical notation “whose application may already be taken to be implicit in our intuitive understanding of a given language or symbolism.”<sup>8</sup> Fine distinguishes between conventional and nonconventional properties of expressions, where nonconventional properties concern the “linguistic function” of the expression. “Semantic role”, on Fine’s usage, picks out these nonconventional properties of meaningful expressions. We might balk at this casual appeal to the distinction between conventional and nonconventional properties, especially in relation to linguistic matters, but nothing in Fine’s argument depends on a sharp boundary and its applications in the cases which he considers are clear. If “semantic role” induces cold sweat in the following discussion, read “linguistic function”, “linguistic role”, or “linguistic use” instead.

Fine’s second point about semantic role is that the relevant difficulties with the variable stem from intuitive judgments about *sameness and difference* of semantic role. The antinomy and the puzzle are therefore compatible with different views of semantic role per se.<sup>9</sup> The final point concerning semantic role is that the antinomy of the variable and its related puzzle presuppose that *free* variables have semantic roles.<sup>10</sup> This assumption taps the root of the question What is a variable? since we might think, as Fine writes but himself disavows, “that our understanding of variables is inseparable from their role in quantification or other forms of variable-binding and that any attempt to explain the role of *free* variables, apart from their connection

---

<sup>8</sup>“The role of variables”, p. 606.

<sup>9</sup>“The role of variables”, p. 606.

<sup>10</sup>“The role of variables”, p. 610.

with the apparatus of binding, must therefore fail.”<sup>11</sup> The assumption that free variables have semantic roles and Fine’s remark about binding raise delicate issues whose discussion must await my (re)formulation of the puzzle and my presentation of Quine’s views. I will address the topic in section 4.

The antinomy of the variable is a generalization of the following example. Suppose we have two distinct variables, “ $x$ ” and “ $y$ ”, and that both admit values from the same domain of objects.<sup>12</sup> For concreteness, following Fine, take the set of real numbers as the domain, and consider the semantic roles of our two variables in the expressions “ $x > 0$ ” and “ $y > 0$ ”.<sup>13</sup> Massaging intuitions, Fine inclines us to say that the semantic roles of  $x$  and  $y$  are the same in these expressions, that “this would appear to be as clear a case as any of a mere ‘conventional’ or ‘notational’ difference; the difference is merely in the choice of the symbol and not in its linguistic function.”<sup>14</sup> Fine’s judgement about appearances in this case is sound. Suppose we are interested in definable subsets of the real numbers in the standard model relative to a language that contains our two variables and  $>$ . Then either formula,  $x > 0$  or  $y > 0$ , defines the set of positive real numbers; the choice of  $x$  or  $y$  is immaterial or merely notational. But this means that there is no difference in the semantic roles of  $x$  and  $y$ . Against this case, Fine has readers consider the semantic roles of  $x$  and  $y$  in the expression  $x > y$ . Again, the verdict is obvious enough, viz., that the semantic roles of the variables are distinct since “it is essential to the linguistic function of the expression as a whole

---

<sup>11</sup>“The role of variables”, p. 610.

<sup>12</sup>We may assume that variables are letters or abstract objects individuated by typographical shape and that they are distinct from their occurrences and their putative semantic roles.

<sup>13</sup>For the remainder of the paper (excluding the **Appendix**), I will ignore use-mention distinctions.

<sup>14</sup>Fine, “The role of variables”, p. 606.

that it contains two distinct variables, not two occurrences of the same variable, and presumably this is because the roles of the distinct variable are not the same.”<sup>15</sup>

The choice of  $x$  and  $y$  in Fine’s example is unimportant. For any two distinct variables, supposing at least two exist, the same argument takes shape. Fine therefore formulates two claims that generalize on our intuitive judgments about sameness and difference of semantic role in the example:

*Semantic Sameness* (SS): Any two variables (ranging over a given domain of objects) have the same semantic role; and

*Semantic Difference* (SD): Any two variables (ranging over a given domain of objects) have a different semantic role.<sup>16</sup>

Fine’s example illustrates a so-called antinomy of the variable because apparently we must maintain contradictory claims about the semantic roles of any two variables.

Fine quickly points out a direct solution to the antinomy. We can resolve the apparent conflict between (SS) and (SD) if we disambiguate the phrase “same semantic role”.<sup>17</sup> Considering the example, we said that (SS) obtains for  $x$  and  $y$  in the expressions  $x > 0$  and  $y > 0$ , but that (SS) fails to obtain for these two variables in  $x > y$ . Ambiguity of “semantic role” in our judgments results from omission of the broader linguistic context:  $x$  and  $y$  have the same semantic role *in two distinct contexts*,  $x > 0$  and  $y > 0$  respectively, but they have different semantic roles *within a single context*, i.e.,  $x > y$ . However we refine intuitions about semantic role, sameness and difference of semantic role surely obtain relative to particular linguistic contexts. Given this modification, Fine distinguishes between what he calls crosscontextual and

---

<sup>15</sup>Fine, “The role of variables”, p. 606.

<sup>16</sup>Fine, “The role of variables”, p. 606.

<sup>17</sup>Fine, “The role of variables”, p. 607.

intracontextual senses of “semantic role”, where (SS) holds for  $x$  and  $y$  in the former but not the latter sense. His example therefore supports only the following two generalizations:

(SS<sub>r</sub>): Any two variables (ranging over a given domain of objects) have the same crosscontextual semantic role; and  
(SD<sub>r</sub>): Any two variables (ranging over a given domain of objects) have a different intracontextual semantic role.<sup>18</sup>

Unlike (SS) and (SD), (SS<sub>r</sub>) and (SD<sub>r</sub>) are not incompatible; they are simply generalizations about two distinct notions of semantic role.<sup>19</sup>

Despite the appeal of this direct solution to the antinomy, Fine argues that it merely relocates the worry insofar as intracontextual differences in semantic role generate crosscontextual differences. Given further modifications of (SS) and (SD) with a univocal sense of “(crosscontextual) semantic role”, Fine’s example then generalizes to a problem, if not an antinomy, concerning variables. Fine bases this further transition from the antinomy on the following point. The variables  $x$  and  $y$  have different intracontextual semantic roles in  $x > y$  because replacing the occurrence of  $y$  with a second occurrence of  $x$  naturally changes the semantic role of  $x > y$ . For example, suppose that the semantic roles of variables involve assignments of values from some domain (assignments are typically defined as certain functions or sequences). Then Fine’s point is that  $x > x$  constrains which assignments are relevant to the semantic role of the whole expression in a way that  $x > y$  does not; whereas some assignment may assign the same value to occurrences of  $x$  and  $y$ , no assignment may assign dis-

---

<sup>18</sup>Fine, “The role of variables”, p. 606.

<sup>19</sup>Fine does not explicitly state (SS<sub>r</sub>) and (SD<sub>r</sub>). I have done so only to make his initial solution to the antinomy more vivid.

tinct values to recurrences of  $x$  in an expression. If we suppose that semantic role is role in inference, we may formulate a second, perhaps more vivid illustration of Fine’s point:  $\langle \{(\forall z)(z \geq z)\}, (x \geq x) \rangle$  is an application or instance of the rule UI, but  $\langle \{(\forall z)(z \geq z)\}, (x \geq y) \rangle$  is not. Both examples reveal that the semantic role of the *pair* of distinct variables  $(x, y)$  in  $x > y$  is crosscontextually different from the semantic role of the *pair* of identical variables  $(x, x)$  in the formula  $x > x$ . So, intracontextual differences in semantic role between  $x$  and  $y$  generate crosscontextual differences in semantic role between the pairs  $(x, y)$  and  $(x, x)$ . Fine exploits this relation between notions of intra- and crosscontextual semantic role and reformulates (SS) and (SD) as follows:

- (SS’): There is no crosscontextual difference in semantic role between the variables  $x$  and  $y$ ; and  
 (SD’): There is a crosscontextual difference in semantic role between the pair of variables  $x, y$  and the pair  $x, x$ .<sup>20</sup>

(SS’) and (SD’) constitute Fine’s final modifications of the original generalizations. Like (SS<sub>r</sub>) and (SD<sub>r</sub>), there is no explicit contradiction between (SS’) and (SD’). However, Fine urges that the latter pair pose a residual more problematic worry:

[H]ow can there be a crosscontextual difference in semantic role between the pair of variables  $x, y$  and the pair  $x, x$  unless there is a crosscontextual semantic difference between the variables  $x$  and  $y$  themselves? What else could account for the difference in semantic role between the  $x, y$  and  $x, x$  except a semantic difference in the individual variables  $x$  and  $y$ ? Or to put it another way, if there is a semantic difference between  $x, y$  and  $x, x$ , then there must be a semantic difference between  $x$  and  $y$ ; and it is hard to see why this difference should only be “turned on” or made manifest

---

<sup>20</sup>Fine, “The role of variables”, p. 608.

when the variables are used in the same context and not when they are used in different contexts.<sup>21</sup>

The residual worry concerns the failure of informal semantic compositionality, or at least we may take Fine's questions in this direction. Informal semantic compositionality is the idea that the semantic role of an expression depends only on the semantic roles of its constituent expressions. From (SD') we have that the open sentences  $x > y$  and  $x > x$  differ in crosscontextual semantic role. By compositionality, this difference depends only on the semantic roles of  $(x, y)$ ,  $>$ , and  $(x, x)$ ,  $>$ , respectively. But, from (SS'), there is no crosscontextual difference in the semantic roles of  $x$  and  $y$ . Since the role of  $>$  is presumably constant across contexts, we conclude that there is no crosscontextual semantic difference between  $x > y$  and  $x > x$ . Contradiction.

[**comment/transition**]

Consider Fine's original example and the initial formulation of (SS) and (SD). Although Fine claims that the latter simply generalize on the example, his later remarks imply that (SS) and (SD) generalize only on certain features of the example. In particular, instead of considering the two variables,  $x$  and  $y$ , (SS) and (SD) concern any two variables; likewise, instead of specifying a particular domain for the variables like the set of real numbers, (SS) and (SD) refer to an arbitrary domain (e.g., there is no provision that the domain must be set). But the example also involves certain expressions that contain (occurrences of) the variables, namely, the open sentences  $x > 0$ ,  $y > 0$ , and  $x > y$ , and the two initial generalizations simply fail to incorporate this feature of the example. Of course, this omission motivates Fine's solution to the

---

<sup>21</sup>Fine, "The role of variables", p. 608.

antinomy, which disambiguates “semantic role” as it occurs in (SS) and (SD) precisely by tying semantic role to linguistic context. The apparent antinomy between (SS) and (SD) simply reflects our failure to *correctly* generalize on the role that linguistic context serves in the example. But what is the correct generalization?

Fine’s original example concerns the semantic roles of  $x$  and  $y$  in the open sentences  $x > 0$ ,  $y > 0$ , and  $x > y$ , and not, say, in the following expressions:  $x > 0$ ,  $y \geq 0$ , and  $x = y$  (or  $(\exists x)(x > 0)$  for that matter). Call the text that surrounds free variables in these expressions matrices (e.g.,  $> 0$  and  $>$  are matrices in the open sentences from Fine’s example). Restriction to the same matrix is implicit in any judgment about the crosscontextual semantic roles of variables; otherwise, judgements would not track semantic roles of variables *as opposed to* properties of possibly distinct matrices. We can judge sameness and difference in the semantic roles of variables, if at all, only if we fix the matrix. However, if we acknowledge this condition or constraint on Fine’s notion of crosscontextual semantic role, we may drop “crosscontextual semantic role” from (SS’) and (SD’). Since the linguistic contexts at issue are invariably open sentences and the matrices are invariably predicates (counting  $> 0$  as a one-place predicate), we can reformulate (SS’) and (SD’) as schematic generalizations. Let  $G$  stand for any two-place predicate,  $x$  and  $y$  be any distinct variables, and  $a$  an individual constant:

(SS’): There is no difference in semantic role between  $x$  and  $y$  in  $Gxa$  and  $Gya$ ; and  
 (SD’): There is a difference in semantic role between the pair of variables  $x$ ,  $y$  and the pair  $x$ ,  $x$  in  $Gxy$  and  $Gxx$ .

(SS’’) and (SD’’) explicitly capture the condition on crosscontextual judgments

about variables, viz., that distinct contexts must contain (occurrences of) the same matrix or predicate. Some auxiliary definitions will facilitate defense and discussion of this point. For now, focusing on (SD''), I take predicate and substitution (and open sentence) as primitive notions. An *atomic open schema* is an expression that consists of a schematic predicate letter (e.g.,  $G$ ) followed by a string of one or more variables (with possible repetitions).<sup>22</sup> Say that a schema  $S$  *schematizes* an open sentence  $S'$  if  $S'$  is the result of some substitution in  $S$ .<sup>23</sup> And last, a predicate  $P$  is said to *occur in an open sentence*  $S'$  if there is some schema  $S$  that schematizes  $S'$  and  $S'$  is the result of substituting  $P$  in  $S$ . For example, if we rewrite Fine's sentences  $x > y$  and  $x > x$ , in a Polish-style prefix notation  $>xy$  and  $>xx$ , schematization is straightforward:  $Gxy$  schematizes  $>xy$  because the latter comes by the former upon substitution of the predicate  $>$  for the predicate letter  $G$ . Likewise,  $Gxx$  schematizes  $>xx$  because the latter is the result of putting  $>$  for  $G$  in the former. The open sentences  $x > y$  and  $x > x$  therefore contain occurrences of the same predicate. Given the stipulation that  $G$  stands for a two-place predicate, any two open sentences respectively schematized by the schemata in (SD'') contain occurrences of that predicate. This stipulation therefore captures the constraint on Fine's notion of crosscontextual semantic role mentioned at the start of this paragraph.

Unfortunately, further examples demand definitions of substitution and predicate whose ramifications entangle crosscontextuality. Staying with (SD''),  $Gxy$  and  $Gxx$

---

<sup>22</sup> $Gxy$  and  $Gxx$  are atomic open schemata. Since I will discuss only atomic open schemata, I will refer to them simply as schemata.

<sup>23</sup>This is a modification of Goldfarb's definition. See his *Deductive Logic* (Indianapolis: Hackett Publishing Company, Inc., 2003), p. 125.

respectively schematize the following open sentences:

- (i)  $x$  used to work for the man who murdered  $y$  and  $y$  was the second husband of  $x$ 's youngest sister
- (ii)  $x$  used to work for the man who murdered  $x$  and  $x$  was the second husband of  $x$ 's youngest sister<sup>24</sup>

But these examples reveal that schematization is not simply a matter of substituting pat expressions like  $<$  for schematic predicate letters. Evidently, we must substitute for the whole schema  $Gxy$  itself, which raises the basic problem of substitution (in elementary logic): Which expression, when substituted for  $Gxy$  and  $Gxx$ , respectively, yields (i) and (ii)? Here is one approach to substitution that captures the new example.<sup>25</sup> A *predicate* is any expression formed from an open sentence by supplanting occurrences of free variables with Greek letters, where occurrences of the same variable are supplanted by occurrences of the same Greek letter. From sentences (i) and (ii), we obtain the following predicates:

- (i')  $\zeta$  used to work for the man who murdered  $\xi$  and  $\xi$  was the second husband of  $\zeta$ 's youngest sister
- (ii')  $\zeta$  used to work for the man who murdered  $\zeta$  and  $\zeta$  was the second husband of  $\zeta$ 's youngest sister.

The Greek letters are neither variables nor any other kind of singular term; they are meaningless place-holders which mark (otherwise unoccupied) argument places in a predicate. Predicates themselves are neither sentences nor schemata; on the

---

<sup>24</sup>The example is from Quine, *Methods of Logic*, third edition (New York: Holt, Rinehart and Winston, 1972), p. 145.

<sup>25</sup>Quine, *Elementary Logic*, revised edition (Cambridge, MA: Harvard UP, 1980), §§40–42. My account is extremely simplified insofar as it treats only *quantifier-free* open schemata and open sentences. For this reason, I ignore restrictions on substitution which greatly complicate the theory.

present approach, they are strictly auxiliary devices which facilitate substitution.<sup>26</sup> *Introduction of a predicate  $P$  at a given occurrence of a predicate letter* consists in supplanting that occurrence and its attached string of variables by the expression which we get from  $P$  by putting the initial variable of the attached string for all occurrences of the alphabetically first Greek letter in  $P$ , the second variable for the alphabetically second Greek letter, and so on. Then, we may say that *substitution of a predicate  $P$  for a given predicate letter in an open schema  $S$*  introduces  $P$  at all occurrences of the given predicate letter in  $S$ . For example, we may substitute (i') for  $G$  in  $Gxy$  by introducing the former at  $G$  as follows. We supplant  $Gxy$  by the result of putting  $x$  for  $\zeta$  and  $y$  for  $\xi$  in (i'), which yields (i). Likewise, we may substitute (i') for  $G$  in  $Gxx$  by supplanting the latter with the result of putting the initial  $x$  for  $\zeta$  and the second occurrence of  $x$  for  $\xi$  in (i'), which generates (ii). Earlier definitions of schematization and predicate-occurrence simply absorb our new definitions of predicate and substitution; the schemata  $Gxy$  and  $Gxx$  schematize (i) and (ii) (as desired) and the latter contain the same predicate, namely, (i').

The revised account clearly subsumes our original case as well. From Fine's two open sentences,  $x > y$  and  $x > x$ , we have the following predicates:  $\zeta > \xi$  and  $\zeta > \zeta$ . But substitution of  $\zeta > \xi$  in the open schemata  $Gxy$  and  $Gxx$ , yield  $x > y$  and  $x > x$ . So, the two schemata schematize  $x > y$  and  $x > x$ , respectively, and these sentences contain occurrences of the same predicate in accordance with the constraint on crosscontextuality. The revised approach differs from the initial one insofar as predicates are now defined as diagrams of a sort ( $\zeta > \xi$ ) and not implicitly

---

<sup>26</sup>We could formulate the theory of substitution using only schemata and open sentences.

identified with constituents of surface syntax ( $<$ ), and, related, substitutions for entire schemata ( $Gxy$ ) now replace more narrow substitutions for schematic predicate letters alone ( $G$ ). But the differences matter since a sentence like  $x > x$  may now contain occurrences of more than one distinct predicate:  $x > x$  contains  $\zeta > \xi$  because the former results from substitution of the latter in  $Gxx$ . But  $x > x$  is also the result of substituting  $\zeta > \zeta$  in  $Gxx$ ; the second occurrence of  $x$  attached to  $G$  vacuously supplants the alphabetically second Greek letter in  $\zeta > \zeta$  since it contains no such letter. In other words, the predicate  $\zeta > \zeta$ , unlike  $\zeta > \xi$ , is a one-place predicate that occurs in  $x > x$ . A second legitimate schematization of  $x > x$  is therefore  $Hx$ .

Use of  $Hx$  or the vacuous substitution of  $\zeta > \zeta$  in  $Gxx$  violates Fine's constraint on crosscontextuality (in  $(SD'')$  at least), which requires, roughly, that  $G$  means the same in  $Gxy$  and  $Gxx$  or that  $>$  means the same in  $x > y$  and  $x > x$ , namely,  $\zeta > \xi$ .<sup>27</sup> Violation of this condition opens the possibility that apparent differences in semantic role between the pair of variables  $(x, y)$  in  $x > y$  and the pair  $(x, x)$  in  $x > x$  are really differences in semantic role between two distinct predicates:  $\zeta > \xi$  and  $\zeta > \zeta$ . Turned around, respect for Fine's constraint, or appeal to crosscontextual semantic role in formulation of the puzzle about variables, closes this possibility. Fine's formulation of the puzzle simply stipulates that the occurrence of  $\zeta > \zeta$  in  $x > x$  is immaterial to the semantic role of  $(x, x)$  in this context and vice versa. In short, the puzzle about variables presupposes that the semantic roles of predicates and variables are mutually independent — this is a nontrivial consequence of the constraint on crosscontextuality, one that naturally touches the puzzle itself and warps

---

<sup>27</sup>Note that substitution of  $\zeta > \zeta$  in  $Gxy$  yields  $x > x$  not  $x > y$ .

the space of admissible solutions.

We should not, however, underestimate Fine's puzzle about variables despite the deeper assumption about predicates lately mentioned. First, predicates, in the sense defined, are dispensable adjuncts to the theory of substitution; only in a strained and contrived sense may they be said to occur in meaningful expressions. But this concession makes predicates mere ghosts in the machinery of the formalism, the actual sentences and formulas, and we struggle to distinguish between  $\zeta > \xi$  and  $\zeta > \zeta$  without reference to  $Gxy$  and  $Gxx$  (or  $x > y$  and  $x > x$ ). Second, a related point, although  $Hx$  legitimately schematizes  $x > x$ , the depiction is unfaithful and buries crucial inferential links between  $x > x$  and  $x > y$  (schematized by  $Gxy$ ). In the setting of first-order logic, we need dyadic representations of one-place predicates like  $Gxx$ . Since we also have dyadic representations of two-place predicates, where a particular case has both argument places occupied by the same argument, there is an unavoidable redundancy of expression within the usual symbolism for first-order logic. Fine's puzzle springs from this redundancy. If  $x > x$  and  $x > y$  contain occurrences of the same two-place predicate, then any semantic difference between them depends on crosscontextual differences in the semantic roles of the variables. Given the relevant redundancy, we may then simply stipulate that the antecedent in this conditional obtains and ignore any dependency between variables and predicates.

[comment/transition]

### 3

Quine announces the motivation for his development of predicate-functor logic [PFL] in nearly every piece that he published on the topic. Here is a quick roundup:

The interest in [PFL] is that the device of the variable receives, in a sense, its full and explicit analysis.<sup>28</sup>

The purpose of predicate-functor logic is itself theoretical: a deeper understanding of the variable.<sup>29</sup>

It is this deepened understanding of the variable itself, rather than any practical advantages of a logic without variables, that makes [PFL] worthwhile.<sup>30</sup>

The philosophical importance of the predicate functors is the resolution of the variable's magic into its pedestrian components, which are matters purely of order and recurrence of reference.<sup>31</sup>

Although PFL might find further applications in logic, linguistics, and computer science, for Quine, the primary interest of this logic is the analysis of the variable that it affords. But what kind of logical apparatus is PFL? What exactly is the “deeper understanding” of the variable that it provides? What are the “pedestrian components” of the variable? Elsewhere, Quine implies that PFL solves a particular technical problem, namely, the problem of algebrizing quantification and other forms of variable-binding. Quine characterizes this problem by contrasting what he calls the algebraic and analytic styles in mathematics and logic. This contrast concerns

---

<sup>28</sup>“Variables explained away” in *Selected Logic Papers*, enlarged edition (Cambridge, MA: Harvard UP, 1995), p. 229.

<sup>29</sup>“Algebraic Logic and Predicate Functors”, p. 305.

<sup>30</sup>Quine, *Methods of Logic*, fourth edition (Cambridge, MA: Harvard UP, 1982), p. 284.

<sup>31</sup>Quine, *From Stimulus to Science* (Cambridge, MA: Harvard UP, 1995), p. 35.

the availability of certain substitutional principles, but the problem is once again simply the nature of the variable. I think that reflection on Quine's distinction would be fruitful, but in this paper I will bracket his contrast between styles and more general questions about the nature of algebraic logic. For our purposes, a single point concerning the algebraic dimensions of PFL is relevant. The predicate-functor solution to the problem of algebrizing quantification, however we understand this problem, eliminates the use of variables in favor of "discrete operations comparable to simple arithmetic."<sup>32</sup> PFL is thus a variable-free logic that seeks to explain variables by explaining them away with a set of auxiliary operations.

At this level of description, the predicate-functor account of the variable dovetails with Quine's broader view of philosophical analysis or explication, which is essentially a kind of theoretical engineering. As explicators, "we supply lacks."<sup>33</sup> Suppose there are some ideas or concepts whose expression is integral to a theoretical pursuit or body of discourse, yet which pose difficulties by way of expression. The task of explication, per Quine, is to "fix on the particular functions of the unclear expression that make it worth troubling about, and then devise a substitute, clear and couched in terms to our liking, that fills those functions."<sup>34</sup> Good philosophical analysis reveals that the original difficulties, often encapsulated by a question like What is a variable?, stem only from needless usages which we may eliminate from the pursuit.

---

<sup>32</sup>*From Stimulus to Science*, p. 33.

<sup>33</sup>*Word and Object* (Cambridge, MA: The MIT Press, 1960), §53, p. 258.

<sup>34</sup>*Word and Object*, §53, pp. 258–259. Quine's paradigm for philosophical explication is the definition — or definitions — of ordered pair within set theory.

An important if nuanced feature of Quine's view is the discrimination between informative and uninformative elimination:

Explication is elimination but not all elimination is explication. Showing how the useful purposes of some perplexing expression can be accomplished through new channels would seem to count as explication just in case the new channels parallel the old ones sufficiently for there to be a striking if partial parallelism of function between the old troublesome form of expression and some form of expression figuring in the new method.<sup>35</sup>

As an eliminative account, Quine's predicate-functor analysis of the variable respects this idea of a parallelism of function insofar as PFL is *intertranslatable* with first-order logic by means of translation algorithms (i.e., effective procedures for mapping sentences in the language of PFL to equivalent closed formulas in first-order logic and vice versa). A striking property of Quine's translation algorithms is that they are stepwise, eliminating each occurrence of a variable in a (closed) formula of first-order logic separately (or introducing occurrences one at a time if we are translating in the other direction). This self-conscious feature of Quine's technical development of PFL isolates those devices of the predicate-functor notation which assume the burden of variables and variable-binding in first-order logic, and, on this basis, we can judge whether a (partial) parallelism of function between the two formalisms obtains. I will return to this issue at the close of the section.

---

<sup>35</sup> *Word and Object*, §53, p. 261.

I have yet to discuss what Quine means by “the pedestrian components” of the variable or the discrete operations which constitute the core of PFL, in short, the predicate functors. The best path to the basic idea of a predicate functor is naturally through Quine.<sup>36</sup> The variable is ostensibly a device for marking certain positions in a sentence (or formula) which impact the meaning of the sentence. Suppose we have a two-place predicate  $B$ , which is the transitive verb “bites”. How might we say that  $x$  bites something? Using the familiar existence prefix “Something  $y$  is such that”, we can proceed by forming the open sentence  $Bxy$  and applying this formula to the prefix to get: Something  $y$  is such that  $Bxy$ . The point about marking positions in sentences which impact meaning becomes vivid if we replace  $x$  with a second occurrence of  $y$ , which yields the following sentence: Something  $y$  is such that  $Byy$  (Something bites itself). Furthermore, by iterating prefixes and exhausting combinations of positions and markings with our two variables, we can say the following things as well: Something  $x$  is such that something  $y$  is such that  $Bxy$  (Something bites something), Something  $y$  is such that  $Byx$  ( $x$  is bitten by something).

The predicate-functor approach provides alternative means for manipulating certain positions in a sentence which impact the meaning of the sentence. Consider the position occupied by  $y$  in the open sentence  $Bxy$  or the position of the object in  $x$  bites  $y$ . On this alternative approach, we begin with a device that transforms a transitive verb, say, “bites”, into an intransitive verb “bites (something)”. The device that Quine proposes is a predicate functor called cropping, which we may represent by the Hebrew letter “ג” (Gimel). ג is neither a predicate, function, nor a function

---

<sup>36</sup>For the following discussion, see Quine, “Variables explained away”, pp. 229–231.

name; it applies neither to itself nor to other functors, but strictly to predicates.<sup>37</sup> More specific,  $\beth$  is an operator that applies to a two-place predicate  $B$ , and forms a complex one-place predicate  $\beth B$ . Applying  $x$  to this latter predicate yields  $(\beth B)x$ , which says that  $x$  bites (something). But, we now have expressed the latter without making use of “Something  $y$  is such that”, the variable  $y$ , or the open sentence  $Bxy$ . The cropping functor has evidently incorporated the linguistic roles of  $y$  directly into a complex predicate built around  $B$  itself. However,  $\beth$  only manipulates a single position in the argument structure of  $B$ , namely, the object position occupied by  $y$  in  $Bxy$ . What about the additional examples that exploit iterations of prefixes and permutations of variables? As Quine writes, “[t]o make the [cropping functor] suffice in lieu of existence prefixes and variables, what are needed are certain further operators capable of coaxing a variable into the right position”, namely, into a position where the cropping functor can crop it.<sup>38</sup> Fulfilling this need is the general burden of Quine’s work on PFL, which, apart from additional predicate functors, requires a generalization of cropping for application to  $n$ -place predicates. In short, we want a set of predicate functors that eliminates variables from polyadic quantification theory. In the remainder of this section I discuss a particular set of functors that Quine investigates relatively late in his career.<sup>39</sup> I will use “ $\Delta_{LM}$ ” to denote the relevant set.  $\Delta_{LM}$ , like alternative sets that Quine recommends on other occasions, suffices for

---

<sup>37</sup>“A functor is a sign that attaches to one or more expressions of a given grammatical kind or kinds to produce an expression of a given grammatical kind” (*Methods of Logic*, fourth edition, p. 129). A predicate functor is therefore a sign that attaches to a predicate to produce a predicate.

<sup>38</sup>“Variables explained away”, p. 230.

<sup>39</sup>See “Predicate functors revisited” in *The Journal of Symbolic Logic*, vol. 46, no. 3 (Sep., 1981), pp. 649–652; “Predicates, terms, and classes” in *Theories and Things* (Cambridge, MA & London: Harvard UP, 1981), pp. 164–172; and *Methods of Logic*, fourth edition, pp. 283–288.

the general apparatus of quantification theory. I will say more about this sufficiency claim momentarily. But, more important,  $\Delta_{LM}$  possesses a property not shared by other sets that is crucial for the evaluation of PFL as an analysis of the variable and my rejection of Fine’s puzzle about variables:  $\Delta_{LM}$  includes a subset of combinatory predicate functors that is adequate for homogenization.

$\Delta_{LM}$  contains the following seven predicate functors (preceded by their names in italics): *complement*  $\text{—}$ , *homogenous intersection*  $\cap$ , *major inversion*  $\text{Inv}$ , *minor inversion*  $\text{inv}$ , *reflection*  $\text{Ref}$ , *padding*  $\text{Pad}$ , and *cropping*  $\text{J}$ . Apart from a few suggestions, and despite the name predicate-functor *logic*, Quine’s writings on PFL conspicuously lack axioms, proof procedures, model-theoretic semantics, and systems of natural deduction for predicate functors. He focuses exclusively on translation algorithms which expose the linguistic roles of variables and establish the relative expressive strength of a language based on predicate functors.<sup>40</sup> Quine always presents predicate functors informally through schematic equivalences which use a mixed notation that relies on the usual interpretation of formalisms for first-order logic. These equivalences are an accessible vehicle for witnessing the effect of the functors on the argument structures of predicates and conveying the truth-conditions of formulas in a language that contains such operators. In addition, Quine uses the schemata for implementing his translation algorithms. Here are equivalences for  $\Delta_{LM}$  ( $F^n$  and  $G^n$  stand for  $n$ -ary primitive predicate constants):

---

<sup>40</sup>Logicians and linguists have since filled this major theoretical lacunae in work on PFL. Model-theoretic semantics have been devised for various predicate-functor languages, and complete and sound proof procedures in the axiomatic and natural deduction style have been formulated. For example, see Steven T. Kuhn, “An axiomatization of predicate functor logic” in *Notre Dame Journal of Formal Logic*, vol. 24, no. 2 (April 1983), pp. 233–241.

- (a)  $(\neg F^n)x_1x_2\dots x_n \equiv \neg(F^n x_1x_2\dots x_n)$
- (b)  $(F^n \cap G^n)x_1x_2\dots x_n \equiv (F^n x_1x_2\dots x_n \wedge G^n x_1x_2\dots x_n)$
- (c)  $(\text{inv}F^n)x_2x_1x_3\dots x_n \equiv F^n x_1x_2x_3\dots x_n$
- (d)  $(\text{Inv}F^n)x_2\dots x_nx_1 \equiv F^n x_1x_2\dots x_n$
- (e)  $(\text{Ref}F^n)x_2\dots x_n \equiv F^n x_2x_2\dots x_n$
- (f)  $(\text{Pad}F^n)x_0x_1x_2\dots x_n \equiv F^n x_1x_2\dots x_n$
- (g)  $(\exists F^n)x_2\dots x_n \equiv (\exists x_1)(F^n x_1x_2\dots x_n)$ <sup>41</sup>

The use of variables in Quine’s presentation of PFL is wholly expository; they are not contained in the primitive notation of PFL. A pure predicate functor language might contain just the single primitive dyadic predicate constant  $L^2$  and the predicate functor Ref. The formulas of this language would be  $L^2$ ,  $\text{Ref}L^2$ ,  $\text{RefRef}L^2$ , and so on (assuming we allow vacuous applications of Ref).<sup>42</sup>

$\Delta_{LM}$  and (a)–(g) suffice to translate any (closed) first-order schema into an equivalent complex 0-ary predicate, i.e., a sentence, built up solely by applications of elements of  $\Delta_{LM}$  to the primitive predicate letters which occur in the schema. This is Quine’s sufficiency claim for PFL. In order to exhibit the predicate functors at work, I will provide a single instances of Quine’s claim. For this example, we must add another predicate functor to the language, the 0-place functor or constant (dyadic)

---

<sup>41</sup>Note that unlike the earlier example with  $B$  and “bites”, this cropping functor does not operate on the last place in the argument structure of a predicate but the first place.

<sup>42</sup>I describe an impure or mixed notation for PFL in **Appendix**.

predicate of identity  $I^2$ , true of all self-identical objects.  $Q$  is any predicate true of Quine and  $D$  is the predicate “died on anti-Christmas”. Predicate functors are boldfaced to enhance visibility.

**Example:** Quine died on anti-Christmas

$$(1) (\exists y)(\forall x)[(Qx \equiv y = x) \wedge Dy]$$

$$(2) (\exists y) \neg (\exists x) \neg [(\neg (Q^1x \wedge \neg \mathbf{I}^2yx) \wedge \neg (\neg Q^1x \wedge \mathbf{I}^2yx)) \wedge D^1y]$$

$$(3) (\exists y) \neg (\exists x) \neg [(\neg (Q^1x \wedge \neg \mathbf{I}^2yx) \wedge \neg (\neg Q^1x \wedge \mathbf{I}^2yx)) \wedge D^1y]$$

$$(4) (\exists y) \neg (\exists x) \neg [(\neg ((\mathbf{Pad}Q^1)yx \wedge \neg \mathbf{I}^2yx) \wedge \neg ((\mathbf{Pad}\neg Q^1)yx \wedge \mathbf{I}^2yx)) \wedge D^1y]$$

$$(5) (\exists y) \neg (\exists x) \neg [(\neg (\mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2)yx \wedge \neg (\mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2)yx) \wedge D^1y]$$

$$(6) (\exists y) \neg (\exists x) \neg [(\neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2)yx \wedge (\neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2)yx) \wedge D^1y]$$

$$(7) (\exists y) \neg (\exists x) \neg [(\neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2)yx) \wedge D^1y]$$

$$(8) (\exists y) \neg (\exists x) \neg [(\neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2)yx) \wedge (\mathbf{inv} \mathbf{Pad} D^1)yx]$$

$$(9) (\exists y) \neg (\exists x) \neg (\neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2 \cap \mathbf{inv} \mathbf{Pad} D^1)yx$$

$$(10) (\exists y) \neg (\exists x) (\neg \neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2 \cap \mathbf{inv} \mathbf{Pad} D^1)yx$$

$$(11) (\exists y) \neg (\exists x) (\mathbf{inv} \neg \neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2 \cap \mathbf{inv} \mathbf{Pad} D^1)xy$$

$$(12) (\exists y) \neg (\mathbf{I} \mathbf{inv} \neg \neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2 \cap \mathbf{inv} \mathbf{Pad} D^1)y$$

$$(13) (\exists y) (\neg \mathbf{I} \mathbf{inv} \neg \neg \mathbf{Pad}Q^1 \cap \neg \mathbf{I}^2 \cap \neg \mathbf{Pad}\neg Q^1 \cap \mathbf{I}^2 \cap \mathbf{inv} \mathbf{Pad} D^1)y$$

$$(14) \quad \lrcorner \lrcorner \text{inv} \text{---} \text{---} \text{Pad} Q^1 \cap \text{---} I^2 \cap \text{---} \text{Pad} \text{---} Q^1 \cap I^2 \cap \text{inv} \text{Pad} D^1$$

Quine describes different translation procedures in various writings (depending on the set of primitive and defined functors), but each requires some preliminary steps. Given our example, we first eliminate singular terms other than variables (e.g., “Quine”) through an extension of Russell’s treatment of definite descriptions. Second, we rid the resulting quantificational formula of universal quantifiers and express all sentential connectives in terms of conjunctions and negations. An intermediate step, required for book keeping, involves the introduction of new predicate letters with superscripts, which simply record the number of variables attached to the plain letters (or flanking =). Finally, we exchange all occurrences of = for occurrences of  $I^2$ . These preliminary steps yield a modified quantificational formula, namely, (2). The second phase of translation, steps (3)–(11), uses (a)–(f) to transform the scope of the innermost existential quantifier  $(\exists x)$  into a single open sentence of the form  $\Gamma xy$ , where  $\Gamma$  is a complex predicate built up by application of the predicate functors to the primitive predicate constants in our example. This phase reduces the four recurrences of  $x$  in the scope of  $(\exists x)$  to a single occurrence, which we coax into a position that can be cropped on the basis of equivalence (g). We repeat this second phase on the scope of the quantifier that becomes innermost after cropping, and we repeat this second phase on the scope of the quantifier that becomes innermost after cropping, and so on. The procedure terminates when there are no occurrences of quantifiers left (and no variables since (1) is a closed sentence).

Inspection of steps (3)–(11) in our example reveals an important feature of the translation algorithm. Consider steps (3)–(5). In (3), we have the subformula  $(Q^1x \wedge \neg I^2yx)$ , which is a conjunction of open sentences. In order to reduce this conjunction to a single open sentence, as dictated by the procedure, we must use (b) and intersect the conjuncts. But  $\cap$  only intersects formulas with identical strings of attached variables. This property of  $\cap$  requires the intermediate step (4), which transforms the left conjunct  $Q^1x$  into  $(\mathbf{Pad}Q^1)yx$ . We may then proceed with the reduction indicated by (5). This simple example illustrates a more general point. Given any two open sentences with attached strings of variables, however heterogeneous, Quine’s procedure for translating from first-order logic into PFL requires that the predicate functors have the expressive capacity to endow the sentences with identical strings of (occurrences of) variables. Technically, the set of functors must be adequate for homogenization.<sup>43</sup> Quine describes this property in the following remark:

The two inversion functors, applied in iteration to an  $n$ -place predicate, suffice to effect any desired permutation of the  $n$  arguments . . . In particular then we can permute any duplicate arguments to initial position and then drop duplications by applying “Ref”. By “Pad”, moreover, we can add extraneous arguments at will, afterward permuting them to any desired positions. The upshot is that our four combinatory functors suffice, without outside aid, to *homogenize* any two predications — that is, to endow them with matching strings of arguments — and to leave the arguments in any desired order, devoid of repetitions.<sup>44</sup>

Adequacy for homogenization is the key to Quine’s predicate-functor analysis of the variable and my rejection of Fine’s puzzle. I will end this section with a brief discussion

---

<sup>43</sup>See **Appendix**, especially **Lemma 3** and the preceding definitions.

<sup>44</sup>“Predicate functors revisited”, p. 650.

of the former point and return to Fine’s argument in section 4.

Quine’s description of homogenization presupposes a distinction between combinatory functors,  $\text{inv}$ ,  $\text{Inv}$ ,  $\text{Red}$ ,  $\text{Pad}$ , and alethic or truth-oriented functors, which are the remaining operators in  $\Delta_{LM}$ .<sup>45</sup> Given this distinction, Quine argues that “the distinctively combinatory business of variables is thus isolated and instructively anatomized,” in short, that the subset  $\Delta_c = \{\text{inv}, \text{Inv}, \text{Ref}, \text{Pad}\} \subset \Delta_{LM}$  of combinatory functors provides a discrete analysis of the nature of the variable.<sup>46</sup> But what is the basis for this distinction between  $\Delta_c$  and the set of alethic functors  $\{\text{—}, \cap, \beth\}$  and why associate the variable with the former? Consider the correspondence of idioms between a language for PFL based on  $\Delta_{LM}$  and a language for first-order logic that contains a countable collection of variables and three sentential connectives, the existential quantifier, negation, and conjunction (apart from punctuation and nonlogical constants). Quine’s insight is that alethic predicate functors are generalized sentential connectives which correspond to the quantifier, negation, and conjunction signs. The latter two are simply special cases of  $\text{—}$  and  $\cap$ , in particular, the case where  $\text{—}$  and  $\cap$  apply to 0-ary predicates, and the correspondence between  $\exists$  and  $\beth$  is evident from inspection of (g). But then the remainders on each side pair off:  $\Delta_c$  corresponds to the countable collection of variables in the language.

Adequacy of  $\Delta_c$  for homogenization — without aid from alethic functors — supports Quine’s claim that there is a parallelism of function between the combinatory functors and the variable. This point is clear if we reinterpret and formalize Quine’s

---

<sup>45</sup>Recall that  $I^2$  is not an element of  $\Delta_{LM}$ .

<sup>46</sup>“Predicate functors revisited”, p. 650.

argument that PFL and first-order logic are intertranslatable for a *mixed* notation  $L_M$ , which contains both  $\Delta_{L_M}$  and a minimal but otherwise standard notation for first-order logic (without  $=$ , individual constants, and function symbols).<sup>47</sup>  $L_M$  contains a stock of  $n$ -ary primitive predicate constants and its set of atomic formulas includes the usual (atomic) open sentences, i.e.,  $n$ -ary primitive predicate constants followed by strings of  $n$  variables (with possible repetitions). In this setting, we may represent semantically relevant distinctions between the argument structures of primitive predicates in two distinct fashions. On the one hand, in the usual manner, we can depict these structures through the use of heterogeneous strings of variables attached to the primitive predicate constants. On the other hand, we can homogenize the set of atomic formulas with  $\Delta_c$  and explicitly represent properties of argument structures through syntactic constituents of the predicates themselves, that is, we can express these properties through iterated applications of the combinatory functors and the generation of complex predicates. But then we can simply dispense with the attached strings of variables, which are now identical and idle —  $\Delta_c$  has wholly usurped the office of the variable. So,  $\Delta_{L_F}$  has the following important property. There exists  $\Delta_c \subset \Delta_{L_M}$  that contains only combinatory predicate functors and that is adequate for homogenization. Quine considers alternative sets of predicate functors which are also adequate for homogenization, but the adequacy depends on both combinatory and alethic predicate functors.<sup>48</sup>

---

<sup>47</sup>See **Appendix**.

<sup>48</sup>[this shows that existence of  $\Delta_c$  nontrivial; expand]

## 4

Recall from section 2 that Fine’s puzzle about variables invokes a general notion of the crosscontextual semantic roles of variables, which severs the latter from predicates. More specific, crosscontextual judgments about the semantic role of the pair of variables  $(x, y)$  in  $x > y$  and the pair  $(x, x)$  in  $x > x$  presuppose that  $x > y$  and  $x > x$  contain occurrences of the same predicate:  $\zeta > \xi$ . But this effectively bars from any explanatory role the distinction between this two-place predicate and the one-place predicate  $\zeta > \zeta$ , which also occurs in  $x > x$ . However, at the level of the actual formalism for first-order logic, this putative distinction between predicates proves elusive; and matched with an expressive redundancy deriving from the need for dyadic depictions of one-place predicates, Fine’s puzzle about variables is unavoidable. We simply have no reason to reject the assumption that  $x > y$  and  $x > x$  contain occurrences of the same predicate.

Quine’s predicate-functor analysis of the variable furnishes such a reason. On this approach,  $Gxx$  is coextensive with  $(\text{Ref}G)x$ . This shows that we do not need dyadic representations of one-place predicates. Instead, we may use coextensive, *complex* one-place predicates, and, in general, as a consequence of  $\Delta_c$ ’s adequacy for homogenization, any use of repeating strings of variables is avoidable, a needless usage. Unlike the artificial predicate-expression  $\zeta > \zeta$ ,  $(\text{Ref}G)$  is a piece of the formalism itself. In addition, unlike the use of (say)  $Hx$  to schematize  $x > x$ ,  $(\text{Ref}G)x$  remains inferentially active in all ways appropriate to  $Gxx$ , given a suitable predicate-functor logic. If we adopt Quine’s analysis of the variable, then we may reject the constraint

on Fine’s notion of crosscontextual semantic role. On the predicate-functor approach, the apparent difference in semantic role between  $(x, y)$  in  $x > y$  and  $(x, x)$  in  $x > x$  is also a difference in semantic role between two distinct predicates, namely, the two-place predicate  $>$  in  $>xy$  and the complex one-place predicate  $(\text{Ref}G)$  in  $(\text{Ref}G)x$ . But from this Quinean point of view, despite initial appearances, Fine’s puzzle about variables tastes like chicken.

[**Fine’s arbitrariness objection**]

## Appendix

The purpose of this appendix is to formalize and prove Quine’s claim that predicate-functor logic is precisely adequate for first-order (polyadic) quantification theory.<sup>49</sup> As I discuss in the paper, Quine demonstrates this claim through provision of translation algorithms which effectively pair closed schemata in the notation of first-order logic with equivalent 0-ary predicate schemata in the notation of PFL. The present exposition reinterprets Quine’s adequacy claim as a result about a mixed language  $L_M$ , which contains both the usual connectives and variables from first-order logic and a finite set of predicate functors  $\Delta_{L_M}$  from Quine (1981, 1981a, 1982). In this setting, Quine’s adequacy claim is a corollary to the following (roughly stated) theorem: For any formula  $\phi$  of  $L_M$  that contains no occurrences of predicate functors, there exists a semantically equivalent atomic formula  $\phi'$  of  $L_M$  that contains a complex  $n$ -ary predicate followed by an ordered sequence of  $n$  distinct free variables, where the latter exhausts the distinct variables which occur free (with possible repetitions and

---

<sup>49</sup>Tom Lockhart provided invaluable assistance with the statement and proof of the main theorem in this appendix.

in any order) in  $\phi$ . Apart from philosophical considerations canvassed in the paper, the significance of this appendix is largely expository.

To facilitate the formulation and proof of the theorem, I adopt the following conventions. Greek letters ‘ $\phi$ ’, ‘ $\psi$ ’,  $\chi$ , and ‘ $\Gamma^n$ ’, ‘ $\Theta^n$ ’, ‘ $\Lambda^n$ ’, are used as syntactic metavariables ranging over formulas in  $L_M$  and  $n$ -ary predicates in  $L_M$  respectively. Similarly, lowercase letters from the beginning of the Greek alphabet, ‘ $\alpha$ ’, ‘ $\beta$ ’,  $\delta$ , and ‘ $\gamma$ ’, with and without indices and accents, range over variables in  $L_M$ . Corner-quotes (‘ $\ulcorner$ ’, ‘ $\urcorner$ ’) are used in the metalanguage to denote unspecified expressions of  $L_M$  which would be obtained by removing the corners and replacing (formerly) enclosed Greek letters by the unspecified expressions which they (temporarily) designate. The set of natural numbers is  $\{0,1,2,\dots\}$ , and lowercase Latin letters, ‘ $l$ ’, ‘ $m$ ’, ‘ $n$ ’, ‘ $p$ ’, ‘ $q$ ’, ‘ $r$ ’, ‘ $s$ ’, ‘ $t$ ’, ‘ $u$ ’, and ‘ $v$ ’, take values from this set.

The language  $L_M$  contains the following symbols, which we assume are distinct:

1. *variables*:  $w, x, y, z$ . Further variables will be formed by means of accents:  $x'$ ,  $y'$ ,  $z'$ ,  $x''$ ,  $\dots$ . The following alphabetic ordering of the set of variables will be assumed throughout the discussion:  $w x y z w' x' y' z' w'' \dots$
2. *primitive predicate constants*: For each  $n > 0$ , there is a countable set of  $n$ -ary primitive predicate constants.
3. *sentential connectives*:  $\neg, \wedge, \exists$ .
4. *predicate functors*:  $—, \cap, \text{inv}, \text{Inv}, \text{Pad}, \text{Ref}, \perp$ .

‘ $\Delta_{L_M}$ ’ denotes the set of predicate functors in  $L_M$ . I will assume that  $L_M$  contains

parentheses and that additional sentential connectives (e.g., ‘ $\equiv$ ’) may be introduced by definition in the usual manner. The identity sign is not a primitive symbol of  $L_M$  and its use in the following discussion is restricted to the metalanguage. Identity may be added to  $L_M$  by expanding the set of sentential connectives and  $\Delta_{L_M}$  with only minimal complications for the proof of the theorem. For  $n \geq 0$ , the set of *n-ary predicates of  $L_M$*  is defined as follows:

1. All primitive predicate constants are *n-ary predicates*.
2. If  $\Gamma^n$  and  $\Theta^n$  are *n-ary predicates*, then  $\lceil \neg \Gamma^n \rceil$ ,  $\lceil \cap \Gamma^n \Theta^n \rceil$ ,  $\lceil \text{inv} \Gamma^n \rceil$ , and  $\lceil \text{Inv} \Gamma^n \rceil$  are *n-ary predicates*.
3. If  $\Gamma^n$  is an *n-ary predicate*, then  $\lceil \text{Pad} \Gamma^n \rceil$  is an  $(n+1)$ -*ary predicate*.
4. If  $\Gamma^n$  is an *n-ary predicate*, then  $\lceil \text{Ref} \Gamma^n \rceil$  and  $\lceil \mathbf{J} \Gamma^n \rceil$  are  $(n-1)$ -*ary predicates*, unless  $n = 0$ , in which case they are *n-ary predicates*.

A finite string of symbols from  $L_M$  is a *predicate of  $L_M$*  if, for some  $n \geq 0$ , it is compelled to be an *n-ary predicate* by 1–4. Note that ‘ $\cap$ ’ is the only binary predicate functor in  $\Delta_{L_M}$ ; the remaining operators are unary. For  $n \geq 0$ , an *atomic formula of  $L_M$*  is  $\lceil \Gamma^n(\alpha_1 \dots \alpha_n) \rceil$ , where  $\Gamma^n$  is an *n-ary predicate* and for all  $i$ ,  $1 \leq i \leq n$ ,  $\alpha_i$  is a variable. A *formula of  $L_M$*  is defined as follows:

1. All atomic formulas are formulas.
2. If  $\phi$  and  $\psi$  are formulas, then  $\lceil (\neg \phi) \rceil$ ,  $\lceil (\phi \wedge \psi) \rceil$ , and  $\lceil (\exists \alpha) \phi \rceil$  are formulas.

I will use ‘ $\text{AtForm}_{L_M}$ ’ and ‘ $\text{Form}_{L_M}$ ’ to denote the set of atomic formulas of  $L_M$  and the set of formulas of  $L_M$  respectively. A given occurrence of a variable  $\alpha$  is said to be

*bound to* a given occurrence of  $\ulcorner(\exists\alpha)\urcorner$  if it stands in a formula that begins with the given occurrence of  $\ulcorner(\exists\alpha)\urcorner$  and stands in no formula beginning with a later occurrence of  $\ulcorner(\exists\alpha)\urcorner$ . An occurrence of a variable  $\alpha$  will be considered *bound in* a formula  $\phi$  if it is bound to an occurrence of  $\ulcorner(\exists\alpha)\urcorner$  in  $\phi$ . Finally, an occurrence of a variable  $\alpha$  will be said to be *free in* a formula  $\phi$  if it is not bound in  $\phi$ .<sup>50</sup> A *sentence of*  $L_M$  is defined in the standard way as a formula of  $L_M$  that contains no free occurrences of variables.

An *interpretation*  $\mathcal{M}$  for  $L_M$  is a pair  $(\mathcal{D}, \mathcal{I})$ , where  $\mathcal{D}$  is a nonempty set of objects and  $\mathcal{I}$  is a function that assigns a subset of  $\mathcal{D}^n$  to each  $n$ -ary primitive predicate constant in  $L_M$ . A *variable-assignment function* is a function  $s$  that assigns to each variable in  $L_M$  an element of  $\mathcal{D}$ . I will use ' $s(\alpha)$ ' to stand for the element of  $\mathcal{D}$  that  $s$  assigns to  $\alpha$  and ' $[s(\alpha|d)]$ ' for an assignment shifted on  $\alpha$  but otherwise identical to  $s$ . A *formula  $\phi$  is satisfied by an interpretation  $\mathcal{M}$  under an assignment  $s$* , in symbols  $\mathcal{M} \models \phi [s]$ , if the following conditions obtain:

1.  $\phi$  is an atomic formula  $\ulcorner\Gamma^n(\alpha_1 \dots \alpha_n)\urcorner$  and
  - (a)  $\Gamma^n$  is a  $n$ -ary primitive predicate constant and  $\langle s(\alpha_1) \dots s(\alpha_n) \rangle \in \mathcal{I}(\Gamma^n)$ ,
  - (b)  $\Gamma^n$  is  $\ulcorner\neg\Theta^n\urcorner$  and  $\mathcal{M} \not\models \ulcorner\Theta^n(\alpha_1 \dots \alpha_n)\urcorner [s]$ ,
  - (c)  $\Gamma^n$  is  $\ulcorner\cap\Lambda^n\Theta^n\urcorner$ ,  $\mathcal{M} \models \ulcorner\Lambda^n(\alpha_1 \dots \alpha_n)\urcorner [s]$  and  $\mathcal{M} \models \ulcorner\Theta^n(\alpha_1 \dots \alpha_n)\urcorner [s]$ ,
  - (d)  $\Gamma^n$  is  $\ulcorner\text{inv}\Theta^n\urcorner$  and  $\mathcal{M} \models \ulcorner\Theta^n(\alpha_2\alpha_1 \dots \alpha_n)\urcorner [s]$ ,
  - (e)  $\Gamma^n$  is  $\ulcorner\text{Inv}\Theta^n\urcorner$  and  $\mathcal{M} \models \ulcorner\Theta^n(\alpha_2 \dots \alpha_n\alpha_1)\urcorner [s]$ ,

---

<sup>50</sup>These definitions are adapted from Quine (1981), pp. 76 and 78.

- (f)  $n \geq 1$ ,  $\Gamma^n$  is  $\ulcorner \text{Pad}\Theta^{n-1} \urcorner$  and  $\mathcal{M} \models \ulcorner \Theta^{n-1}(\alpha_2 \dots \alpha_n) \urcorner [s(\alpha_1|d)] \forall d \in \mathcal{D}$ ,
- (g)  $\Gamma^n$  is  $\ulcorner \text{Ref}\Theta^{n+1} \urcorner$  and  $\mathcal{M} \models \ulcorner \Theta^{n+1}(\alpha_1 \alpha_1 \dots \alpha_n) \urcorner [s]$ , or
- (h)  $\Gamma^n$  is  $\ulcorner \mathbf{J}\Theta^{n+1} \urcorner$  and  $\mathcal{M} \models \ulcorner \Theta^{n+1}(\alpha_0 \alpha_1 \dots \alpha_n) \urcorner [s(\alpha_0|d)]$  for some  $d \in \mathcal{D}$ .

- 2.  $\phi$  is  $\ulcorner (\neg\psi) \urcorner$  and  $\mathcal{M} \not\models \psi [s]$ ,
- 3.  $\phi$  is  $\ulcorner (\psi \wedge \chi) \urcorner$ ,  $\mathcal{M} \models \psi [s]$  and  $\mathcal{M} \models \chi [s]$ , or
- 4.  $\phi$  is  $\ulcorner (\exists\alpha)\psi \urcorner$  and  $\mathcal{M} \models \psi [s(\alpha|d)]$  for some  $d \in \mathcal{D}$ .

A formula  $\phi$  is true in  $\mathcal{M}$  if  $\mathcal{M} \models \phi [s]$  for any assignment function  $s$ . A formula  $\phi$  is valid if it is true in all interpretations of  $L_M$ , in which case we write:  $\models \phi$ .

Our reinterpretation and proof of Quine's adequacy claim for PFL depends on the existence of a subset of formulas of  $L_M$  that satisfies the following conditions. Let  $\Sigma_0 = \{\ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner \in \text{AtForm}_{L_M} \mid \Gamma^n \text{ is an } n\text{-ary primitive predicate constant}\}$ . We define  $\Sigma \subset \text{Form}_{L_M}$  as follows:

- 1.  $\Sigma_0 \subseteq \Sigma$
- 2. If  $\phi, \psi \in \Sigma$ , then  $\ulcorner (\neg\phi) \urcorner \in \Sigma$ ,  $\ulcorner (\phi \wedge \psi) \urcorner \in \Sigma$ , and  $\ulcorner (\exists\alpha)\phi \urcorner \in \Sigma$ .

I will assume that a unique such  $\Sigma$  exists.<sup>51</sup> Observe that  $\Sigma$  is precisely the set of formulas in some minimal but otherwise standard language for first-order logic (minimal insofar as it lacks constant symbols, function symbols, and the identity sign). Conditions 1–2 simply state that  $\Sigma$  contains the atomic formulas of such a

---

<sup>51</sup>This should follow from a recursion theorem and unique readability for  $\text{Form}_{L_M}$  (the same point holds for the existence of a satisfaction relation satisfying 1–4 above).

minimal language and is closed under the usual formula-building operations. Proof that  $\Sigma$  coincides exactly with the set of formulas in a pure predicate language for first-order logic is trivial if we stipulate that the latter contains the same set of variables,  $n$ -ary primitive predicate constants, and primitive sentential connectives as  $L_M$ .

Given the preceding characterization of  $L_M$ , I will now formulate a series of definitions and lemmas which are essential for the proof of the theorem.

**Lemma 1** All formulas of the following forms are valid formulas of  $L_M$  (supposing ‘ $\equiv$ ’ replaced by primitive notation).

1.  $\ulcorner \neg \Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n) \equiv (\neg \Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n))^\ulcorner$
2.  $\ulcorner \cap \Gamma^n \Theta^n(\alpha_1 \alpha_2 \dots \alpha_n) \equiv (\Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n) \wedge \Theta^n(\alpha_1 \alpha_2 \dots \alpha_n))^\ulcorner$
3.  $\ulcorner \text{inv} \Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n) \equiv \Gamma^n(\alpha_2 \alpha_1 \dots \alpha_n)^\ulcorner$
4.  $\ulcorner \text{Inv} \Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n) \equiv \Gamma^n(\alpha_2 \dots \alpha_n \alpha_1)^\ulcorner$
5.  $\ulcorner \text{Pad} \Gamma^n(\alpha_0 \alpha_1 \dots \alpha_n) \equiv \Gamma^n(\alpha_1 \dots \alpha_n)^\ulcorner$
6.  $\ulcorner \text{Ref} \Gamma^n(\alpha_2 \alpha_3 \dots \alpha_n) \equiv \Gamma^n(\alpha_2 \alpha_2 \alpha_3 \dots \alpha_n)^\ulcorner$
7.  $\ulcorner \beth \Gamma^n(\alpha_2 \dots \alpha_n) \equiv (\exists \alpha_1) \Gamma^n(\alpha_1 \alpha_2 \dots \alpha_n)^\ulcorner$

1–7 adapt to  $L_M$  the schemata that Quine uses to describe his translation algorithms. Apart from incorporating Quine’s schemata directly into a mixed notation for PFL, our equivalences differ from Quine’s in holding for any  $n$ -ary predicate, whether primitive or complex. It is routine to check that 1–7 are valid given the semantics for  $L_M$ . However, the proof requires a local determination lemma for satisfaction of formulas,

roughly, that for any two variable-assignment functions  $s_1$  and  $s_2$ , which agree on all free occurrences of variables in a formula  $\phi$  (if any), we have that  $\mathcal{M} \models \phi [s_1]$  if and only if  $\mathcal{M} \models \phi [s_2]$ .<sup>52</sup>

For the following definitions, let  $\phi = \ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner$  be an atomic formula of  $L_M$  and let  $S$  be the set of occurrences of variables in  $\phi$ , i.e.,  $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ .

**Definition 1** We define  $\phi_\pi = \ulcorner \Gamma^n(\pi(\alpha_1) \dots \pi(\alpha_n)) \urcorner$ , where  $\pi$  is some permutation of  $S$ . Since there are  $n!$  permutations for any  $n$ -membered set  $S$ , for each such  $\phi \in AtForm_{L_M}$ , there are  $n!$  such  $\phi_\pi \in AtForm_{L_M}$ . If  $S$  is empty, that is, no variables occur in  $\phi$ , then  $\phi_\pi$  is  $\phi$ .

**Definition 2** Let  $\Omega \in \{\text{inv}, \text{Inv}\} \subset \Delta_{L_M}$ . Then  $\ulcorner \Omega \phi \urcorner$  will be said to *induce a permutation of  $\phi$* , in symbols  $\ulcorner \Gamma^n(\pi_\Omega^\phi \alpha_1 \dots \pi_\Omega^\phi \alpha_n) \urcorner$ , where  $\pi_\Omega^\phi$  is a permutation of  $S$  defined as follows:

$$1. \text{ If } \Omega = \text{inv}, \text{ then } \pi_\Omega^\phi \alpha_i = \begin{cases} \alpha_2 & \text{for } i = 1 \\ \alpha_1 & \text{for } i = 2 \\ \alpha_i & \text{for } 2 < i \leq n \end{cases}$$

$$2. \text{ If } \Omega = \text{Inv}, \text{ then } \pi_\Omega^\phi \alpha_i = \begin{cases} \alpha_1 & \text{for } i = n \\ \alpha_{i+1} & \text{for } 1 \leq i < n \end{cases}$$

$\ulcorner \pi_\Omega^\phi \alpha_i \urcorner$  denotes the image of the variable  $\alpha_i$  under the permutation of  $\phi$  induced by the predicate functor  $\Omega$ .

---

<sup>52</sup>See Enderton (2001), p. 86, **Theorem 22A**. The definition of truth for sentences of  $L_M$  follows from the local determination theorem.

**Defintion 3** If  $\ulcorner \Omega \phi \urcorner$  induces a permutation of  $\phi \in AtForm_{L_M}$ , then it is said to be a *permutation-variant* of  $\phi$ . Observe that if  $\psi$  is a permutation-variant of  $\phi$ , we have that  $\models \ulcorner \phi \equiv \psi \urcorner$ . We extend the notion of a permutation-variant of an atomic formula in two directions. First, we allow that every atomic formula is a permutation-variant of itself. Second, if  $\phi_1 \dots \phi_n$  is a (nonempty) finite sequence of atomic formulas such that, for all  $i$ ,  $1 \leq i < n$ ,  $\phi_{i+1}$  is a permutation-variant of  $\phi_i$ , then we say that  $\phi_n$  is a permutation-variant of  $\phi_1$ .

Given  $\phi \in AtForm_{L_M}$ , we can now ask whether, for some  $\phi_\pi$  associated with  $\phi$ , there exists a permutation-variant of  $\phi$  that induces the permutation  $\pi$ . In general, we need to prove the following lemma, which states that iterated applications of the minor and major inversion functors to the predicates in atomic formulas of  $L_M$  suffice to induce all permutations of the (occurrences of) variables in those formulas.

**Lemma 2** Let  $\phi = \ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner$  be an atomic formula of  $L_M$  and  $\phi_{\pi_1} \phi_{\pi_2} \dots \phi_{\pi_{n!}}$  be an enumeration of the  $\phi_\pi$  associated with  $\phi$  as in **Definition 1**. Then for each  $i$ ,  $1 \leq i \leq n!$ , there exists  $\phi_i = \ulcorner \boxtimes \Gamma^n(\beta_1 \dots \beta_n) \urcorner \in AtForm_{L_M}$  that meets the following conditions:

1.  $\phi_i$  is a permutation-variant of  $\phi$ .
2.  $\boxtimes$  is a finite sequence of (occurrences of) elements from  $\{\text{inv}, \text{Inv}\}$ .
3. For all  $j$ ,  $1 \leq j \leq n$ ,  $\beta_j = \pi_i(\alpha_j)$ .

*proof* [I'm working on it but here's a sketch that borrows some ideas from algebra. The minor and major inversion functors induce cyclic permutations on the set of

variables in  $\phi$  (ordered according to their occurrences in  $\phi$ ). In particular,  $\text{inv}$  and  $\text{Inv}$  induce transpositions or cycles which have length 2 (they swap two variables and fix the remaining ones). A theorem from algebra states that all permutations of a (nonempty) finite set can be reached through a succession of transpositions or compositions of 2-cycles of the form  $(a_1 a_k) \dots (a_1 a_3)(a_1 a_2)$ , where  $(a_1 a_2 \dots a_k)$  is a  $k$ -cycle that occurs in the standard representation of a given permutation (of the set of  $a_i$ 's) as a product of disjoint cycles. The proof of **Lemma 2** then reduces to showing that  $\text{inv}$  and  $\text{Inv}$  suffice to induce all transpositions of the relevant form. (Quine's argument)]

**Definition 4** Let  $\phi = \ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner$  and  $\psi = \ulcorner \Theta^m(\beta_1 \dots \beta_m) \urcorner$  be atomic formulas of  $L_M$ .  $\phi$  and  $\psi$  are said to be *homogenous* if and only if  $n = m$  and for all  $i, j$ ,  $1 \leq i, j \leq n = m$ , the following holds:

1.  $\alpha_i = \beta_i$ .
2. If  $i \neq j$ , then  $\alpha_i \neq \alpha_j$ .
3. If  $i < j$ , then  $\alpha_i$  is alphabetically prior to  $\alpha_j$ .

**Definition 5** Let  $\phi, \psi \in \text{AtForm}_{L_M}$ . A set  $\Delta \subseteq \Delta_{L_M}$  of predicate functors is said to *homogenize*  $\phi$  and  $\psi$  if there exist  $\phi', \psi' \in \text{AtForm}_{L_M}$  such that  $\phi' = \ulcorner \boxtimes \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner$ ,  $\psi' = \ulcorner \boxtimes' \Theta^m(\beta_1 \dots \beta_m) \urcorner$  and

1.  $\phi'$  and  $\psi'$  are homogenous.
2.  $\models \ulcorner \phi \equiv \phi' \urcorner$  and  $\models \ulcorner \psi \equiv \psi' \urcorner$ .

3.  $\boxtimes, \boxtimes'$  are finite sequences of (occurrences of) elements of  $\Delta$ .

**Definition 6** A set  $\Delta \subseteq \Delta_{L_M}$  of predicate functors is *adequate for homogenization* if and only if for any  $\phi, \psi \in AtForm_{L_M}$ ,  $\Delta$  homogenizes  $\phi$  and  $\psi$ .

**Lemma 3** The set  $\Delta_c = \{\text{inv}, \text{Inv}, \text{Pad}, \text{Ref}\} \subseteq \Delta_{L_M}$  is adequate for homogenization.

*proof* Suppose  $\phi = \ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner$  and  $\psi = \ulcorner \Theta^m(\beta_1 \dots \beta_m) \urcorner$  are atomic formulas of  $L_M$ . For  $p \leq n, q \leq m$ , let  $\gamma_1 \dots \gamma_p \gamma'_1 \dots \gamma'_q$  be an enumeration of the distinct variables which occur (with possible repetitions) in  $\phi$  and  $\psi$  respectively. We consider four cases. (i) Suppose  $p = n$  and  $q = m$ . For each  $\gamma'_i$ , if there is no  $\gamma_j$  such that  $\gamma'_i = \gamma_j$ , add exactly one occurrence of  $\gamma'_i$  to  $\phi$  by a single application of Pad. Let ‘Pad<sup>u</sup>’ stand for a string of  $u$  occurrences of Pad. After we check each  $\gamma'_i$ , we have  $\ulcorner \text{Pad}^u \Gamma^n(\alpha'_1 \dots \alpha'_u \alpha_1 \dots \alpha_n) \urcorner$ , where for each  $l, 1 \leq l \leq u \leq q$ , there is some  $i, 1 \leq i \leq q$ , such that  $\alpha'_l = \gamma'_i$ . By **Lemma 2**, there exists a permutation-variant of  $\ulcorner \text{Pad}^u \Gamma^n(\alpha'_1 \dots \alpha'_u \alpha_1 \dots \alpha_n) \urcorner$  that induces the “alphabetic” permutation of  $\alpha'_1 \dots \alpha'_u \alpha_1 \dots \alpha_n$ , i.e., the induced permutation maps  $\alpha'_1$  to the alphabetically first variable in  $\alpha'_1 \dots \alpha'_u \alpha_1 \dots \alpha_n$ , and  $\alpha'_2$  to the alphabetically second variable, and so forth. Let  $\phi' = \ulcorner \boxtimes \text{Pad}^u \Gamma^n(\delta_1 \dots \delta_s) \urcorner$  be this permutation-variant, where  $s = u + n$ . A similar argument yields a  $\psi' = \ulcorner \boxtimes' \text{Pad}^v \Theta^m(\delta_1 \dots \delta_s) \urcorner$ , where  $s = v + m$ . But  $\phi'$  and  $\psi'$  are homogenous.<sup>53</sup> [**incomplete**]

**Theorem:** For any  $\ulcorner \phi(\alpha_1 \dots \alpha_n) \urcorner \in \Sigma$  with  $n$  free occurrences of  $m$  distinct variables,  $n \geq m$ , there exists an  $m$ -ary predicate  $\Gamma^m$  in  $L_M$  such that the following hold:

<sup>53</sup>We must check that  $\phi$  and  $\phi'$  ( $\psi$  and  $\psi'$ ) are equivalent, which requires proof that  $\ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner \equiv \text{Pad}^u \Gamma^n(\alpha'_1 \dots \alpha'_u \alpha_1 \dots \alpha_n) \urcorner$  and  $\ulcorner \Theta^m(\alpha_1 \dots \alpha_m) \urcorner \equiv \text{Pad}^v \Theta^m(\beta'_1 \dots \beta'_v \beta_1 \dots \beta_m) \urcorner$  are valid.

1.  $\models \ulcorner \phi(\alpha_1 \dots \alpha_n) \equiv \Gamma^m(\beta_1 \dots \beta_m) \urcorner$
2.  $\beta_i \neq \beta_j$ , for all  $i, j$ ,  $1 \leq i \neq j \leq m$
3. If  $i < j$ , then  $\beta_i$  is alphabetically prior to  $\beta_j$
4. For each  $\alpha_i$ ,  $1 \leq i \leq n$ , there exists  $\beta_j$ ,  $1 \leq j \leq m$ , such that  $\alpha_i = \beta_j$ .

The following proof is by induction on the complexity of  $\phi \in \Sigma$ , where the complexity of a formula is the number of occurrences of logical constants it contains.

*proof:* Let  $\phi = \ulcorner \Gamma^n(\alpha_1 \dots \alpha_n) \urcorner \in \Sigma$  be an atomic formula with  $n$  free occurrences of  $m$  distinct variables. We consider two cases. (i) Suppose  $n = m$ . Then each variable from the  $m$  distinct variables occurs exactly once in  $\phi$ . By **Lemma 1**, there exists a formula  $\ulcorner \boxtimes \Gamma^n(\beta_1 \dots \beta_m) \urcorner$  that is equivalent to  $\phi$ , where  $\boxtimes$  is a finite sequence of elements of  $\Delta_c$ , and  $\beta_1 \dots \beta_m$  satisfy 2–4 in the statement of the theorem. (ii) Assume that  $n > m$ . By **Lemma 2** and **Lemma 3**, we can eliminate any repetitions in  $\alpha_1 \dots \alpha_n$  and obtain a formula  $\ulcorner \boxtimes \Gamma^n(\delta_1 \dots \delta_m) \urcorner$  that is equivalent to  $\phi$ , where  $\boxtimes$  is a finite sequence of elements of  $\Delta_c$ ,  $\ulcorner \boxtimes \Gamma^n \urcorner$  is an  $m = (n-k)$ -ary predicate, and  $k$  is the number of occurrences of Ref in  $\boxtimes$ . But this reduces case (ii) to case (i), so we have shown that the theorem holds for any  $\phi \in \Sigma$ , where the complexity  $\phi$  is 0. Next assume that our theorem holds for all formulas in  $\Sigma$  of complexity  $\leq l$ . We must show that the conditions of theorem are satisfied for any  $\phi \in \Sigma$  of complexity  $\leq l+1$ . We divide this step into three cases.

(i') Suppose  $\phi = \ulcorner (\neg\psi(\alpha_1 \dots \alpha_n)) \urcorner \in \Sigma$  contains  $n$  free occurrences of  $m$  distinct variables. By induction, since the complexity of  $\psi$  is at most  $l$ , there exists

an  $m$ -ary predicate  $\Gamma^m$  in  $L_M$  such that  $\models \ulcorner \psi(\alpha_1 \dots \alpha_n) \equiv \Gamma^m(\beta_1 \dots \beta_m) \urcorner$  and that satisfies 2–4 in the statement of the theorem. It follows that  $\models \ulcorner (\neg \psi(\alpha_1 \dots \alpha_n)) \equiv (\neg \Gamma^m(\beta_1 \dots \beta_m)) \urcorner$ . But, by **Lemma 1**, we have that  $\ulcorner \neg \Gamma^m(\beta_1 \dots \beta_m) \equiv (\neg \Gamma^m(\beta_1 \dots \beta_m)) \urcorner$ . So,  $\models \ulcorner (\neg \psi(\alpha_1 \dots \alpha_n)) \equiv \neg \Gamma^m(\beta_1 \dots \beta_m) \urcorner$  as needed.

(ii') Suppose  $\phi = \ulcorner (\psi(\alpha_1 \dots \alpha_p) \wedge \chi(\delta_1 \dots \delta_q)) \urcorner \in \Sigma$  contains  $n$  free occurrences of  $m$  distinct variables,  $p, q \leq m \leq n$ . Let  $\gamma_1 \dots \gamma_m$  be an alphabetic ordering of the  $m$  distinct variables that occur free (with possible repetitions) in  $\phi$ . By induction, since the complexity of  $\ulcorner \psi(\alpha_1 \dots \alpha_p) \urcorner$  ( $\ulcorner \chi(\delta_1 \dots \delta_q) \urcorner$ ) is at most  $l$ , there exists  $r$ -ary and  $t$ -ary predicates,  $\Gamma^r$  and  $\Theta^t$ , such that  $\models \ulcorner \psi(\alpha_1 \dots \alpha_p) \equiv \Gamma^r(\beta_1 \dots \beta_r) \urcorner$  and  $\models \ulcorner \chi(\delta_1 \dots \delta_q) \equiv \Theta^t(\beta'_1 \dots \beta'_t) \urcorner$  and that satisfy conditions 2–4 respectively. It follows that  $\models \ulcorner (\Gamma^r(\beta_1 \dots \beta_r) \wedge \Theta^t(\beta'_1 \dots \beta'_t)) \equiv (\psi(\alpha_1 \dots \alpha_p) \wedge \chi(\delta_1 \dots \delta_q)) \urcorner$ . Insofar as we have no guarantee that  $r = t = m$ , we cannot directly apply **Lemma 1** in order to find a complex predicate that is equivalent to  $\phi$  (and that satisfies 2–4). For a different choice of  $\Delta$ , this step in the proof would be directly analogous to case (i'), but working with the primitive set of predicate functors in  $L_M$ , we must first homogenize  $\Gamma^r$  and  $\Theta^t$ . This requires applications of **Lemma 2** and **Lemma 3** along the following lines. Consider the  $m$  distinct variables that occur free (with possible repetitions) in  $\phi$ :  $\gamma_1 \dots \gamma_m$ . Working with  $\gamma_1$ , scan  $\ulcorner (\Gamma^r(\beta_1 \dots \beta_r)) \urcorner$  from left to right checking for an occurrence of  $\gamma_1$  (recall that each  $\beta_i$  must be identical to some  $\gamma_j$  by induction). If you hit an occurrence of  $\gamma_1$ , restart the procedure with  $\gamma_2$ . If you reach the end of the formula, then  $\gamma_1$  is not among the  $\beta_i$ 's and we must add it to the latter. **Lemma 3** guarantees that we can do this and, by **Lemma 2**, we can arrange the new string of variables in alphabetic order. After repeating this procedure for each

$\gamma_i$ , we obtain a predicate  $\ulcorner \boxtimes \Gamma^r \urcorner$  such that  $\models \ulcorner \boxtimes \Gamma^r(\gamma_1 \dots \gamma_n) \equiv \Gamma^r(\beta_1 \dots \beta_n) \urcorner$ , where  $\boxtimes$  is a finite sequence of elements of  $\Delta_c$ ,  $\ulcorner \boxtimes \Gamma^r \urcorner$  is an  $m = (r + v)$ -ary predicate, and  $v$  is the number of occurrences of in  $\boxtimes$ . [**incomplete**]

**Corollary** For any sentence  $\sigma \in \Sigma$ , there exists a 0-ary predicate  $\Gamma$  such that  $\models \ulcorner \sigma \equiv \Gamma \urcorner$ .