Composition under distributive natural transformations: Or, When Predicate Abstraction is impossible

Dylan Bumford, UCLA

published at: http://dx.doi.org/10.1007/s10849-022-09361-2

Abstract Natural language semanticists have often found it useful to assume that all expressions denote sets of values. The approach is most prominent in the study of questions and prosodic focus, but also common in work on indefinites, disjunction, negative polarity, and scalar implicature. However, the most popular compositional implementation of this idea is known to face technical obstacles in the presence of object-language binding constructs, including, chiefly, lambda abstraction. The problem has been well-described on several occasions in the literature, and in fact several solutions have been explored. This paper seeks to formalize the challenge of defining an indeterminate semantics for binding operators, and to formally establish the intuition that the challenge is in fact insurmountable. The primary benefit to this exercise is that it offers an abstract characterization of what it means to lift an operation from one semantic space to another, a notion which may be applied to domains having nothing to do with sets of alternatives.

1 Introduction

Compositional theories of natural language semantics often take the form of inductive interpretations of simply-typed lambda terms. In much early work following Montague 1973 these terms comprise the metalanguage in which denotations are expressed. Semantic analysis consists in translating natural language constituents into formal terms with variable-binding constructs, and then assigning meanings to constituents by interpreting their formal translations. In much contemporary work since the publication of Heim & Kratzer's (1998) influential textbook, object languages are themselves assumed to be structured as lambda terms, with variables and variable-abstractions appearing directly in natural language abstract syntax.

Canonically, these expressions are interpreted in the familiar manner of Henkin 1950. Basic types τ are associated with domains $\boldsymbol{D}_\tau$, and functional types $(\sigma\tau)$ with functions $f : \boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau$ between domains. Terms are evaluated relative to assignments that map typed variables v_τ to objects in the relevant domain $\boldsymbol{D}_\tau$. Abstractions denote functions, modifying the assignments at which their bodies are evaluated.

For some purposes, however, these object- or meta-language lambda terms may be interpreted differently. Perhaps most prominently, Rooth 1985 defined a semantics for Montague's higher-order logic in which terms of type τ are interpreted not as functions from assignments to objects in $\boldsymbol{D}_\tau$, but rather as *sets* of such functions. This, he argued, provides a formal strategy for comparing the meaning of an expression to the *alternative* meanings it evokes, crucial to understanding the effects of prosodic focus. Later work in the spirit and shadow of Rooth has extended this idea to explicate the semantic force of question particles (Hagstrom 1998), indeterminate pronouns (Kratzer & Shimoyama 2002), disjunctions and indefinites (Alonso-Ovalle 2006), among other things. But in many of these sequels, syntactic terms of type τ are not interpreted as sets of functions from assignments to $\boldsymbol{D}_\tau$, as in Rooth 1985, but as functions from assignments to sets of $\boldsymbol{D}_\tau$ values. Somewhat unfairly, but in keeping with what terminological consistency there is in the literature, I will call interpretations in this latter type signature *Hamblin denotations*, after Hamblin's (1973) pioneering analysis of questions. In particular, where Rooth interprets abstraction terms $\lambda v_\sigma.\, \phi_\tau$ as sets of functions from assignments to elements of $[\boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau]$,[1] the Hamblin semantics defined in, e.g., Hagstrom 1998 interprets them as functions from assignments to elements of $\wp\,[\boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau]$, the powerset of $[\boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau]$.

Shan 2004 points out that these respective semantic spaces are far from equivalent, and moreover that the denotations assigned to abstraction terms by Hagstrom and the others are inadequate to the empirical tasks they

1 I will use $[\boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau]$ to pick out the set of functions from $\boldsymbol{D}_\sigma$ to $\boldsymbol{D}_\tau$, in lieu of the more traditional $\boldsymbol{D}_\tau^{\boldsymbol{D}_\sigma}$, as the arrow notation is better suited to higher-order signatures.

are designed for. Worse, Shan contends that it is in fact *impossible* to define a suitable Hamblin denotation for abstraction. Charlow 2019a observes that the same difficulties arise in trying to define other binding constructs, including existential closure, if the denotations of their prejacents are functions from assignments into sets instead of sets of functions from assignments.

The demonstrations in Shan 2004 and Charlow 2019a,b leave no doubt that the definition of abstraction in Hagstrom 1998 and Kratzer & Shimoyama 2002 is empirically defective; the reader is referred to these works for concrete illustrations of its failure, and illuminating discussion of the general nature of the problem. In this note, I wish to contribute two small things to the discussion. First, I hope to offer a somewhat more abstract criterion of adequacy for a Hamblin semantics of abstraction, or any other syntactic construct that has an *ordinary* interpretation of the sort discussed in the second paragraph above. The goal here is to lay out a few algebraic laws that we should expect to hold between the intended ordinary denotation of an operator and its Hamblin denotation in the more structured semantic space. Second, I will prove that for a class of such operators, including lambda abstraction, such a definition is indeed impossible, as Shan ordained.[2]

In short, I will propose that an adequate semantics for abstraction depends on the existence of a **natural transformation** between Roothian denotations and Hamblin denotations satisfying two **monad distributivity laws**. These are notions from Category Theory, though I will attempt to justify them on conceptually and linguistically intuitive grounds (see Asudeh & Giorgolo 2020 for a more general linguistic introduction to the mathematics). The strategy is general enough that it should extend to analyses that locate denotations in other sorts of regimented structures, including values paired with propositions (Potts 2005, Koev 2017), domains augmented with undefined elements (Beaver & Coppock 2015, Grove 2019), and values paired with continuations (Krifka 1991, Barker 2016). I discuss these extensions briefly in Section 6.

2 Denotations in Hamblin space

I assume the simplest canonical definition of natural language logical forms consists in at least the following syntax-driven interpretive clauses:[3]

$$[\![c]\!] \coloneqq \lambda g.\, \mathcal{I}\, c \tag{1}$$

$$[\![t_n]\!] \coloneqq \lambda g.\, g_n \tag{2}$$

$$[\![E_{\sigma\tau}\ K_\sigma]\!] \coloneqq \lambda g.\, [\![E]\!]\, g\, ([\![F]\!]\, g) \tag{3}$$

Expressions of type σ denote functions from assignments to values in $\boldsymbol{D}_\sigma$. Lexical items c are looked up in a lexicon $\mathcal{I}$. Variables t_n correspond to projection functions on assignments. And appropriately typed binary constituents are combined by function application.

To this bare-bones formalism, any number of unary **variable-binding** and **type-shifting** constructs and/or alternative binary **modes of combination** may be added. Most fragments since Montague 1973 include a "lambda" of some sort that denotes a function parameterized on the value assigned to a particular variable, as in (4). Less selective binders have also been proposed (Lewis 1975, Heim 1982), along the lines of the universal closure operator in (5). Linguistic type-shifters have come in all shapes and sizes; Partee's (1986) *Lift* and *Be*, in (6) and (7), are paradigmatic examples of the genre. Alternative binary modes of combination oriented around Boolean operations and function composition are commonly entertained, as in (8) and (9).

2 Shan reported in a footnote that the issue with lambda abstraction could be "formalized using *logical relations* and *parametricity* (Reynolds 1983) as for programming languages." I can only presume Shan had something like the proof presented here in mind.

3 This presentation follows the currently common linguistic practice of using undecorated numbers as abstractions and indexed ts as variables. The lambdas on the right-hand sides of the denotation equations are intended to signify actual functions, whose domains and codomains are given in the obvious way by the types of the constituents being interpreted. Application of a function to an argument is indicated by (left-associated) juxtaposition.

$$[\![n_\sigma\ E]\!] := \lambda g \lambda a.\, [\![E]\!]\, g^{n \mapsto a} \tag{4}$$

$$[\![\text{⸸}\ E]\!] := \lambda g.\, \forall h.\, [\![E]\!]\, h \tag{5}$$

$$[\![\mathscr{L}\ E_\sigma]\!] := \lambda g \lambda k.\, k\, ([\![E]\!]\, g) \tag{6}$$

$$[\![\mathscr{B}\ E_{(\sigma\mathrm{t})\mathrm{t}}]\!] := \lambda g \lambda a.\, [\![E]\!]\, g\, (\lambda b.\, b = a) \tag{7}$$

$$[\![E_{\sigma\mathrm{t}}\ K_{\sigma\mathrm{t}}]\!] := \lambda g \lambda a.\, [\![E]\!]\, g\, a \wedge [\![K]\!]\, g\, a \tag{8}$$

$$[\![E_{\rho\tau}\ K_{\sigma\rho}]\!] := \lambda g \lambda a.\, [\![E]\!]\, g\, ([\![K]\!]\, g\, a) \tag{9}$$

Early in the Montagovian era, Hamblin 1973 sketched an analysis of questions that sought to maintain as much of the canonical fragment above as possible while assuming that 'wh'-words like 'who' and 'what' ought to behave for all compositional purposes like names. Yet of course such words cannot be assumed to refer to specific individuals. Hamblin's solution was to reanalyze all argument terms as denoting *sets* of individuals. Names then played the special role of terms whose denotations were **determinate**, or singleton. In fact all non-'wh'-words were taken to denote determinate sets of values.

Let $\{\!|E|\!\}$ signify the **Hamblin denotation** of E. At any assignment, 'wh'-phrases denote sets of entities, suitably sortably restricted. Variables and non-'wh'-constants of type σ are interpreted, relative to assignments, as elements of $\wp\, \boldsymbol{D}_\sigma$, that is, as sets of $\boldsymbol{D}_\sigma$ objects.

$$\{\!|\text{who}|\!\} := \lambda g.\, \{x \in \boldsymbol{D}_\mathrm{e} \mid \mathsf{person}\, x\} \tag{10}$$

$$\{\!|c|\!\} := \lambda g.\, \{\mathcal{I}\, c\} \tag{11}$$

$$\{\!|t_n|\!\} := \lambda g.\, \{g_n\} \tag{12}$$

The question then is how to restore compositional, recursive operations like those in (4)–(9), keeping in mind that their inventory may be open-ended and depend on analyses of phenomena completely independent of questions. Hamblin himself suggested that function application ought to be executed **pointwise**; the combination of an expression ϕ of type $(\sigma\tau)$ with an expression ψ of type σ should be the set of all possible applications of a function in $\{\!|\phi|\!\}$ to an argument in $\{\!|\psi|\!\}$.

$$\{\!|\phi_{\sigma\tau}\ \psi_\sigma|\!\} = \lambda g.\, \left\{ f\, a \;\middle|\; \begin{array}{l} f \in \{\!|\phi|\!\}\, g, \\ a \in \{\!|\psi|\!\}\, g \end{array} \right\} \tag{13}$$

There is an obvious sense in which this definition generalizes the ordinary applicative mode of combination. Let $f \bullet a := f\, a$ be the semantic operation of function application. Then the relevant clauses of both $[\![\cdot]\!]$ and $\{\!|\cdot|\!\}$ can be written in terms of $\bullet$:

$$f \bullet a := f\, a \tag{14a}$$

$$[\![\phi_{\sigma\tau}\ \psi_\sigma]\!] = \lambda g.\, ([\![\phi]\!]\, g) \bullet ([\![\psi]\!]\, g) \tag{14b}$$

$$\{\!|\phi_{\sigma\tau}\ \psi_\sigma|\!\} = \lambda g.\, \left\{ f \bullet a \;\middle|\; \begin{array}{l} f \in \{\!|\phi|\!\}\, g, \\ a \in \{\!|\psi|\!\}\, g \end{array} \right\} \tag{14c}$$

This much is trivial, but it provides a clear schema for other modes of combination, as in (15) and (16).

$$f \circ f' := \lambda a.\, f\, (f'\, a) \tag{15a}$$

$$[\![\phi_{\rho\tau}\ \psi_{\sigma\rho}]\!] = \lambda g.\, ([\![\phi]\!]\, g) \circ ([\![\psi]\!]\, g) \tag{15b}$$

$$\{\!|\phi_{\rho\tau}\ \psi_{\sigma\rho}|\!\} = \lambda g.\, \left\{ f \circ f' \;\middle|\; \begin{array}{l} f \in \{\!|\phi|\!\}\, g, \\ f' \in \{\!|\psi|\!\}\, g \end{array} \right\} \tag{15c}$$

$$p \sqcap q := \lambda a.\, p\, a \wedge q\, a \tag{16a}$$

$$[\![\phi_{\sigma t}\ \psi_{\sigma t}]\!] = \lambda g.\, ([\![\phi]\!]\, g) \sqcap ([\![\psi]\!]\, g) \tag{16b}$$

$$\{\!|\phi_{\sigma t}\ \psi_{\sigma t}|\!\} = \lambda g.\, \left\{ p \sqcap q \;\middle|\; \begin{array}{l} p \in \{\!|\phi|\!\}\, g, \\ q \in \{\!|\psi|\!\}\, g \end{array} \right\} \tag{16c}$$

Likewise, various unary syntactic operators can be associated with semantic functions. Hamblin denotations may then be generated by mapping these functions over the set of values returned by $\{\!|\cdot|\!\}$.

$$\mathbf{L} := \lambda a \lambda k.\, k\, a \tag{17a}$$
$$[\![\mathscr{L}\ E]\!] = \lambda g.\, \mathbf{L}\, ([\![E]\!]\, g) \tag{17b}$$
$$\{\!|\mathscr{L}\ E|\!\} = \lambda g.\, \{\mathbf{L}\, a \mid a \in \{\!|E|\!\}\, g\} \tag{17c}$$

$$\mathbf{B} := \lambda Q \lambda a.\, Q\, (\lambda b.\, b = a) \tag{18a}$$
$$[\![\mathscr{B}\ E]\!] = \lambda g.\, \mathbf{B}\, ([\![E]\!]\, g) \tag{18b}$$
$$\{\!|\mathscr{B}\ E|\!\} = \lambda g.\, \{\mathbf{B}\, Q \mid Q \in \{\!|E|\!\}\, g\} \tag{18c}$$

However, this general strategy is no help in defining Hamblin denotations for the binding constructs. What the rules above have in common is that the operations $\bullet$, $\circ$, $\sqcap$, $\mathbf{B}$, and $\mathbf{L}$ are all assignment-independent. They are *extensional*. Binders, on the other hand, are definitionally assignment-modifying. In other words, there is no function $\mathbf{A}$ such that $[\![n_\sigma\ \phi]\!] = \lambda g.\, \mathbf{A}\, ([\![\phi]\!]\, g)$, and so no function to stick into the translational template in (17c) and (18c).

In light of this, a Hamblin denotation for abstraction terms must be *ad-hoc*. Given the typing regime, the goal for any expression E of type τ is to find for $\{\!|n_\sigma\ E|\!\}$ a function from assignments to sets of $\boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau$ functions. For instance, Kratzer & Shimoyama 2002, following Hagstrom 1998, define Hamblin abstraction as in (19).

$$\{\!|n_\sigma\ E|\!\} := \lambda g.\, \{f : \boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau \mid \forall a \in \boldsymbol{D}_\sigma.\, f\, a \in \{\!|E|\!\}\, g^{n\mapsto a}\} \tag{19}$$

This definition succeeds in picking out, relative to an assignment, a set of functions in the desired signature in terms of the Hamblin denotation of its prejacent $\{\!|E|\!\}$. It is therefore well-typed and compositional. However, Shan 2004, Romero & Novel 2013, and Charlow 2019a,b argue that this definition yields sets of alternatives that include many conceptually suspect functions, and demonstrate that it leads directly to a variety of empirical inadequacies.

Could we do better? There are certainly other ways to assemble a set of functions with the right domain and codomain from $\{\!|\phi|\!\}$. It is hard to say *a priori* whether any of them would be adequate to empirical purposes, that is, whether they would "do the job" that abstractions are supposed to do. The same goes for any other binding construct. Given the lack of a systematic mapping procedure from ordinary space to Hamblin space for such constructs, how should we determine which possible function from assignments to sets of the relevant shape is appropriate for an operation, given its ordinary meaning, or whether any such appropriate denotation even exists?

In the next section I formalize this question and offer a criterion of adequacy for any Hamblin denotation of a compositional operation. I then prove that no denotation for an *intensional* operation, including variable binding, can meet this criterion.

3 The formal problem

Let υ be a unary syntactic operator, so that for some class of expressions $\mathcal{E}$, if $E \in \mathcal{E}$, then $[\upsilon\ E]$ is a well-formed constituent. Assume also that the interpretation of υ is compositional, so that for any E, $[\![\upsilon\ E]\!]$ is a function of $[\![E]\!]$. As usual, we identify $[\![\upsilon]\!]$ with this function. Finally, assume that the expressions in $\mathcal{E}$ take denotations (relative to assignments) in some set $\mathcal{A}$, and that the expressions $[\upsilon\ E]$ take denotations (relative to assignments) in a set $\mathcal{R}$. Then $[\![\upsilon]\!]$ itself is a function from $[\mathcal{G} \to \mathcal{A}]$ to $[\mathcal{G} \to \mathcal{R}]$, where $\mathcal{G}$ is the domain of assignments.

Call any such υ **extensional** if $[\![\upsilon]\!]$ is such that for any $g \in \mathcal{G}$ and $\alpha : \mathcal{G} \to \mathcal{A}$:

$$[\![\upsilon]\!]\, \alpha\, g = [\![\upsilon]\!]\, (\lambda g'.\, \alpha\, g)\, g \tag{20}$$

That is, at any assignment g, $[\![\upsilon]\!]$ cares only about the value that its argument α takes at g. For instance, the Partee type-shifters $\mathscr{L}$ and $\mathscr{B}$ are extensional, given the definitions in (6) and (7):

$$[\![\mathscr{L}]\!] \coloneqq \lambda\alpha\lambda g\lambda k.\, k\,(\alpha\, g) \tag{21}$$

$$\begin{aligned}[\![\mathscr{L}]\!]\,(\lambda g'.\,\alpha\, g)\, g &= \lambda k.\, k\,((\lambda g'.\,\alpha\, g)\, g) && \text{(22a)}\\ &= \lambda k.\, k\,(\alpha\, g) && \text{(22b)}\\ &= [\![\mathscr{L}]\!]\,\alpha\, g && \text{(22c)}\end{aligned}$$

$$[\![\mathscr{B}]\!] \coloneqq \lambda\alpha\lambda g\lambda x.\, \alpha\, g\,(\lambda y.\, y = x) \tag{23}$$

$$\begin{aligned}[\![\mathscr{B}]\!]\,(\lambda g'.\,\alpha\, g)\, g &= \lambda x.\,(\lambda g'.\,\alpha\, g)\, g\,(\lambda y.\, y = x) && \text{(24a)}\\ &= \lambda x.\,\alpha\, g\,(\lambda y.\, y = x) && \text{(24b)}\\ &= [\![\mathscr{B}]\!]\,\alpha\, g && \text{(24c)}\end{aligned}$$

Call any υ **intensional** if it doesn't satisfy the equation in (20). In general, any sort of binding or context-shifting operator will be intensional. For instance, the abstraction and closure constructs in (4) and (5) are intensional, as seen in (26) and (28).

$$[\![n]\!] \coloneqq \lambda\alpha\lambda g\lambda x.\, \alpha\, g^{n\mapsto x} \tag{25}$$

$$\begin{aligned}[\![n]\!]\,(\lambda g'.\,\alpha\, g)\, g &= \lambda x.\,(\lambda g'.\,\alpha\, g)\, g^{n\mapsto x} && \text{(26a)}\\ &= \lambda x.\,\alpha\, g && \text{(26b)}\\ &\neq [\![n]\!]\,\alpha\, g && \text{(26c)}\end{aligned}$$

$$[\![\text{⸸}]\!] \coloneqq \lambda\alpha\lambda g.\, \forall h.\, \beta\, h \tag{27}$$

$$\begin{aligned}[\![\text{⸸}]\!]\,(\lambda g'.\,\alpha\, g)\, g &= \forall h.\,(\lambda g'.\,\alpha\, g)\, h && \text{(28a)}\\ &= \forall h.\,\alpha\, g && \text{(28b)}\\ &\neq [\![\text{⸸}]\!]\,\alpha\, g && \text{(28c)}\end{aligned}$$

Similarly, when μ is a compositional binary mode of combination $[\![\phi\ \psi]\!]_\mu$, we write $[\![\cdot]\!]_\mu$ for that function which combines any appropriately typed $[\![\phi]\!]$ and $[\![\psi]\!]$ to produce $[\![\phi\ \psi]\!]_\mu$. So if (at an assignment) ϕ denotes a value in $\mathcal{A}$, ψ a value in $\mathcal{B}$, and $[\phi\ \psi]$ a value in $\mathcal{R}$, then $[\![\cdot]\!]_\mu : [\mathcal{G} \to \mathcal{A}] \times [\mathcal{G} \to \mathcal{B}] \to [\mathcal{G} \to \mathcal{R}]$.

In this case, we say μ is extensional if $[\![\cdot]\!]_\mu$ is such that for any $g \in \mathcal{G}$, $\alpha : \mathcal{G} \to \mathcal{A}$, and $\beta : \mathcal{G} \to \mathcal{B}$:

$$[\![\cdot]\!]_\mu\,(\alpha, \beta)\, g = [\![\cdot]\!]_\mu\,(\lambda g'.\,\alpha\, g,\ \lambda g'.\,\beta\, g)\, g \tag{29}$$

And otherwise, we say μ is intensional. The modes of combination in (3), (8) and (9) are all extensional.

Finally, as discussed in Section 2, for any expression E of type σ, assume that the Hamblin denotation $\{\![E]\!\}$ is (relative to an assignment) a subset of $\boldsymbol{D}_\sigma$. That is, $\{\![E]\!\} : \mathcal{G} \to \wp\, \boldsymbol{D}_\sigma$. Here then is the question:

$$\boxed{\begin{array}{l}\text{For an arbitrary operator } \upsilon\text{, is it possible to define a sensible Hamblin denotation } \{\![\upsilon\ E]\!\} \text{ in terms of}\\ [\![\upsilon]\!] \text{ and } \{\![E]\!\}\text{? Or, likewise, for an arbitrary mode of combination } \mu\text{, is it possible to define a sensible}\\ \text{Hamblin mode } \{\![\phi\ \psi]\!\}_\mu \text{ in terms of } [\![\cdot]\!]_\mu,\ \{\![\phi]\!\}\text{, and } \{\![\psi]\!\}\text{?}\end{array}} \tag{30}$$

Say for concreteness that E has type τ and $[\upsilon\ E]$ has type ρ. Then $\{\![\upsilon\ E]\!\}$ should be a function from $\mathcal{G}$ to $\wp\, \boldsymbol{D}_\rho$, assembled somehow from $\{\![E]\!\} : \mathcal{G} \to \wp\, \boldsymbol{D}_\tau$ and $[\![\upsilon]\!] : [\mathcal{G} \to \boldsymbol{D}_\tau] \to \big[\mathcal{G} \to \boldsymbol{D}_\rho\big]$. There are two basic obstacles here. One is that $[\![\upsilon]\!]$ *expects* a function from $\mathcal{G}$ to $\boldsymbol{D}_\tau$, but $\{\![E]\!\}$ is a function from $\mathcal{G}$ to $\wp\, \boldsymbol{D}_\tau$. The other is that $[\![\upsilon]\!]$ *returns* a function from $\mathcal{G}$ to $\boldsymbol{D}_\rho$, but the Hamblin denotation $\{\![\upsilon\ E]\!\}$ should be a function from $\mathcal{G}$ to $\wp\, \boldsymbol{D}_\rho$.

The same issue arises with binary combination. Say ϕ has type σ, ψ has type τ and $[\phi\ \psi]$ has type ρ. Then $\{\![\phi\ \psi]\!\}_\mu$ ought to be a function from $\mathcal{G}$ to $\wp\, \boldsymbol{D}_\rho$. But the mismatches now are doubled. On the one hand, $[\![\cdot]\!]_\mu$ expects a pair of functions $\mathcal{G} \to \boldsymbol{D}_\sigma$ and $\mathcal{G} \to \boldsymbol{D}_\tau$, though $\{\![\phi]\!\}$ and $\{\![\psi]\!\}$ deliver a pair of functions $\mathcal{G} \to \wp\, \boldsymbol{D}_\sigma$ and $\mathcal{G} \to \wp\, \boldsymbol{D}_\tau$. On the other hand, $\{\![\phi\ \psi]\!\}$ is supposed to deliver a function $\mathcal{G} \to \wp\, \boldsymbol{D}_\rho$, but $[\![\cdot]\!]_\mu$ provides only a function $\mathcal{G} \to \boldsymbol{D}_\rho$.

What is needed then is a map $\Upsilon : [\mathcal{G} \to \wp\, \boldsymbol{D}_\tau] \to \wp\, [\mathcal{G} \to \boldsymbol{D}_\tau]$ to transform a function into sets into a set of functions. With that, the Hamblin denotations could be defined as follows:

$$\{\![\upsilon\ E]\!\} \coloneqq \lambda g.\, \{[\![\upsilon]\!]\, \varphi\, g \mid \varphi \in \Upsilon\, \{\![E]\!\}\} \tag{31}$$

$$\{\![E\ F]\!\}_\mu \coloneqq \lambda g.\, \left\{ [\![\cdot]\!]_\mu\,(\varphi, \psi)\, g \;\middle|\; \begin{array}{l}\varphi \in \Upsilon\, \{\![E]\!\},\\ \psi \in \Upsilon\, \{\![F]\!\}\end{array} \right\} \tag{32}$$

For extensional rules, it's easy enough to concoct such a map. The following would work (note the similarity with (19)), as would others.

$$\Upsilon \coloneqq \lambda\Phi.\, \{\varphi : \mathcal{G} \to \boldsymbol{D}_\tau \mid \forall h \in \mathcal{G}.\, \varphi\, h \in \Phi\, h\} \tag{33}$$

Since an extensional $\llbracket \upsilon \rrbracket$ only ever evaluates its argument at its own input g, the set determined by (31) and (33) depends only on the image of $\lBrace E \rBrace$ at g. For example,

$$\begin{aligned} \lBrace \mathscr{L}\ E \rBrace &= \lambda g.\, \{\llbracket \mathscr{L} \rrbracket\ \varphi\ g \mid \varphi \in \Upsilon\ \lBrace E \rBrace\} && \text{(34a)} \\ &= \lambda g.\, \{\llbracket \mathscr{L} \rrbracket\ \varphi\ g \mid \forall h \in \mathcal{G}.\, \varphi\ h \in \lBrace E \rBrace\ h\} && \text{(34b)} \\ &= \lambda g.\, \{\lambda k.\, k\ (\varphi\ g) \mid \forall h \in \mathcal{G}.\, \varphi\ h \in \lBrace E \rBrace\ h\} && \text{(34c)} \\ &= \lambda g.\, \{\lambda k.\, k\ a \mid a \in \lBrace E \rBrace\ g\} && \text{(34d)} \\ &= \lambda g.\, \{\mathbf{L}\ a \mid a \in \lBrace E \rBrace\ g\} && \text{(34e)} \end{aligned}$$

The set described in (34c) is determined by the collection of functions φ which have the property that at any assignment h, they pick out an element of $\lBrace E \rBrace\ h$. There are a great many such functions. But since for each one, $\lambda k.\, k\ (\varphi\ g)$ simply projects out the value that it takes at g, almost all of the variation among the possible φs is irrelevant. All that matters is what value they choose at g. And as (34c) guarantees, the values available at g are all and only the values in $\lBrace E \rBrace\ g$. Thus the intimidating expression in (34c) is reduced to the familiar (34d), which is exactly in accordance with the schema in (17c).

But for intensional operators, this Υ is clearly inappropriate. Consider for instance the closure operator $\dagger$. Perhaps the simplest cases to inspect are those in which $\dagger$'s prejacent contains no free variables at all. The ordinary denotation of such an expression is equivalent to the ordinary denotation of the prejacent, since the quantification over assignments is vacuous:

$$\begin{aligned} \llbracket \dagger\ [\text{John left}] \rrbracket &= \lambda g.\, \forall h.\, \llbracket \text{John left} \rrbracket\ h && \text{(35a)} \\ &= \lambda g.\, \forall h.\, \mathsf{left}\ \mathsf{j} && \text{(35b)} \\ &= \lambda g.\, \mathsf{left}\ \mathsf{j} && \text{(35c)} \end{aligned}$$

But things are quite different for the Hamblin denotation of such a vacuously closed expression. Assume, following the discussion in Section 2, that $\lBrace \text{who left} \rBrace = \lambda g.\, \{\mathsf{left}\ z \mid z \in \boldsymbol{D}_{\mathrm{e}}\}$. Then (31)–(33) predict:

$$\begin{aligned} \lBrace [\dagger\ [\text{who left}]] \rBrace &= \lambda g.\, \{\llbracket \dagger \rrbracket\ \varphi\ g \mid \varphi \in \Upsilon\ \lBrace \text{who left} \rBrace\} && \text{(36a)} \\ &= \lambda g.\, \{\llbracket \dagger \rrbracket\ \varphi\ g \mid \forall h \in \mathcal{G}.\, \varphi\ h \in \lBrace \text{who left} \rBrace\ h\} && \text{(36b)} \\ &= \lambda g.\, \{\llbracket \dagger \rrbracket\ \varphi\ g \mid \forall h \in \mathcal{G}.\, \varphi\ h \in \{\mathsf{left}\ z \mid z \in \boldsymbol{D}_{\mathrm{e}}\}\} && \text{(36c)} \\ &= \lambda g.\, \{\forall h.\, \varphi\ h \mid \forall h \in \mathcal{G}.\, \varphi\ h \in \{\mathsf{left}\ z \mid z \in \boldsymbol{D}_{\mathrm{e}}\}\} && \text{(36d)} \\ &= \lambda g.\, \{\forall x \in D.\, \mathsf{left}\ x \mid D \subseteq \boldsymbol{D}_{\mathrm{e}},\ D \neq \emptyset\} && \text{(36e)} \end{aligned}$$

Any φ that meets the comprehension condition in (36d) is a function that sends each $h \in \mathcal{G}$ to a proposition of the form $(\mathsf{left}\ z)$, where z is an entity. Some such functions map every assignment to $\mathsf{left}\ \mathsf{j}$, others to $\mathsf{left}\ \mathsf{m}$; others may map some assignments to $\mathsf{left}\ \mathsf{j}$ and other assignments to $\mathsf{left}\ \mathsf{m}$; etc. For any of these φs, to say that $\varphi\ h$ is true at every h is to say that every proposition in the range of φ is true. In other words, at any input g, the alternatives in $\lBrace \dagger\ [\text{who left}] \rBrace\ g$ are all propositions of the form "everyone in D left", for arbitrary settings of who's in D. This is certainly an interesting set of propositions, but not at all what you'd expect when trying to bind all of the free variables in this sentence with no free variables.[4] Things get only less predictable when there really are variables to bind, as the reader may verify.

So the question in (30) comes down to whether there exists an $\Upsilon : [\mathcal{G} \to \wp\ \boldsymbol{D}_{\tau}] \to \wp\ [\mathcal{G} \to \boldsymbol{D}_{\tau}]$ that could deliver sensible results in (31) and (32).

4 As it happens, when propositions are identified with simple Boolean values, the denotation in (36e) is in fact equivalent to $\lBrace \text{who left} \rBrace = \lambda g.\, \{\mathsf{left}\ z \mid z \in \boldsymbol{D}_{\mathrm{e}}\}$, due to the coarseness of nonempty sets of bits (there are only three!). If instead $\mathsf{left}\ z$ is the set of worlds in which z left, for instance, the function in (36e) is clearly different from the Hamblin denotation of the question.

4 Proposal: Distributive Natural Transformations

I would like to suggest that any transformation Υ worth considering ought to satisfy the following three constraints. For any types σ and τ, and any $P \subseteq \boldsymbol{D}_\sigma$, $f : \boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau$, $\varphi : \mathcal{G} \to \boldsymbol{D}_\sigma$, and $\Phi : \mathcal{G} \to \wp\, \boldsymbol{D}_\sigma$:

$$\Upsilon\,(\lambda g.\, P) = \{\lambda g.\, x \mid x \in P\} \qquad \textbf{Left}$$

$$\Upsilon\,(\lambda g.\, \{\varphi\, g\}) = \{\varphi\} \qquad \textbf{Right}$$

$$\Upsilon\,(\lambda g.\, \{f\, x \mid x \in \Phi\, g\}) = \{\lambda g.\, f\,(\varphi\, g) \mid \varphi \in \Upsilon\, \Phi\} \qquad \textbf{Nat}$$

These are the natural transformation and distributivity laws when assignment-dependence and indeterminacy are considered as monads. More on this in Section 5. But first, let's see what they amount to for practical linguistic purposes.

Imagine that an operator's prejacent contains no variables, traces, pronouns, etc. The Hamblin denotation of such a constituent will be a constant function from assignments to some set $P \subseteq \boldsymbol{D}_\sigma$. What set of assignment-dependent values should Υ return? **Left** guarantees that in this case, the set we get back is just the set of constant functions into the elements of P.

Assume, as is common, that **focus alternatives** are calculated using Hamblin denotations. That is, prosodically focused expressions contribute ordinary denotations to ordinary truth conditions, but at the same time raise the specter of *alternative* denotations from the same domain. These alternative values contribute to calculations of alternative propositions, ideas evoked but not uttered. For instance, the alternatives elicited by placing stress on 'John' in 'John smiled' are just propositions about other people smiling (Mary smiled, Bill smiled, etc.). **Left** ensures the result in (37). Upon repackaging, the alternatives of 'John$_\mathsf{F}$ smiled' are just the ordinary denotations 'z smiled' for other type-e expressions z.

$$\Upsilon\, \{\!|\text{John}_\mathsf{F}\text{ smiled}|\!\} = \Upsilon\,(\lambda g.\, \{\text{smiled}\, z \mid z \in \boldsymbol{D}_\mathsf{e}\}) \qquad (37a)$$

$$= \{\lambda g.\, \text{smiled}\, z \mid z \in \boldsymbol{D}_\mathsf{e}\} \qquad (37b)$$

$$= \{[\![\text{z smiled}]\!] \mid \text{z} :: \mathsf{e}\} \qquad (37c)$$

More generally, if $[\cdots\ \mathrm{x}_\mathsf{F}\ \cdots]$ is variable-free, then **Left** guarantees the identity in (38), where $a \sim b$ means that a has the same type as b. In other words, in purely extensional contexts where assignments are irrelevant, focus alternatives can be computed from ordinary denotations by simply replacing focused constituents with expressions of the same type.

$$\Upsilon\, \{\!|[\cdots \mathrm{x}_\mathsf{F} \cdots]|\!\} = \{[\![[\cdots \mathrm{z} \cdots]]\!] \mid \mathrm{z} \sim \mathrm{x}\} \qquad (38)$$

Note that this immediately ensures that variable closure (or any other intensional operation) over a closed formula is vacuous, in contrast to (33).

$$\{\!|\forall\, E|\!\} = \lambda g.\, \{[\![\forall]\!]\, \varphi\, g \mid \varphi \in \Upsilon\, \{\!|E|\!\}\} \qquad (39a)$$

$$= \lambda g.\, \{[\![\forall]\!]\, \varphi\, g \mid \varphi \in \{\lambda g'.\, p \mid p \in \{\!|E|\!\}\, g\}\} \qquad (39b)$$

$$= \lambda g.\, \{[\![\forall]\!]\, (\lambda g'.\, p)\, g \mid p \in \{\!|E|\!\}\, g\} \qquad (39c)$$

$$= \lambda g.\, \{\forall h.\, p \mid p \in \{\!|E|\!\}\, g\} \qquad (39d)$$

$$= \lambda g.\, \{p \mid p \in \{\!|E|\!\}\, g\} \qquad (39e)$$

$$= \{\!|E|\!\} \qquad (39f)$$

Now imagine that an operator's prejacent contains no interesting alternative-generating language; no 'wh'-words, focused constituents, etc. Its Hamblin denotation φ, at any assignment, is just the singleton set of its ordinary denotation at that assignment. Then **Right** guarantees that when this denotation is passed to Υ, the result contains exactly the one function that maps any g to the single element in $\varphi\, g$.[5]

For instance, assume that the Hamblin denotation of a vanilla declarative sentence, like 'he smiled', is just the function $\lambda g.\, \{[\![t_1 \text{ smiled}]\!]\, g\}$, mapping assignments to singleton ordinary denotations. Then **Right** ensures:

$$\begin{aligned} \Upsilon \{\!\!\{ [t_1 \text{ smiled}] \}\!\!\} &= \Upsilon\, (\lambda g.\, \{\text{smiled}\, g_1\}) && \text{(40a)}\\ &= \{\lambda g.\, \text{smiled}\, g_1\} && \text{(40b)} \end{aligned}$$

More generally, if E contains no alternative-generating expressions (is focus-free, etc.), then **Right** guarantees:

$$\Upsilon \{\!\!\{ E \}\!\!\} = \{[\![E]\!]\} \tag{41}$$

At the very least, this ensures that abstraction satisfies η-reduction in the absence of focus. Assume that E contains no focus-marking and no free occurrence of t_n (the latter is only important to justifying the step from (42f) to (42g)).

$$\begin{aligned} \{\!\!\{ n\ [E\ t_n] \}\!\!\} &= \lambda g.\, \{[\![n]\!]\, \varphi\, g \mid \varphi \in \Upsilon \{\!\!\{ E\ t_n \}\!\!\}\} && \text{(42a)}\\ &= \lambda g.\, \{[\![n]\!]\, \varphi\, g \mid \varphi \in \{[\![E\ t_n]\!]\}\} && \text{(42b)}\\ &= \lambda g.\, \{[\![n]\!]\, [\![E\ t_n]\!]\, g\} && \text{(42c)}\\ &= \lambda g.\, \{\lambda a.\, [\![E\ t_n]\!]\, g^{n\mapsto a}\} && \text{(42d)}\\ &= \lambda g.\, \{\lambda a.\, [\![E]\!]\, g^{n\mapsto a}\, a\} && \text{(42e)}\\ &= \lambda g.\, \{[\![E]\!]\, g^{n\mapsto a}\} && \text{(42f)}\\ &= \lambda g.\, \{[\![E]\!]\, g\} && \text{(42g)}\\ &= \{\!\!\{ E \}\!\!\} && \text{(42h)} \end{aligned}$$

Nat is more complicated, but guarantees that Υ is not *ad-hoc*, in the sense that it should perform "the same" operation no matter what size or type of denotation it is passed. To see this, let $\circledast_{\text{R}}$ and $\circledast_{\text{H}}$ be defined as in (43) and (44).

$$\begin{aligned} \circledast_{\text{H}} &: [\mathcal{G} \to \wp\, [\mathcal{A} \to \mathcal{B}]] \times [\mathcal{G} \to \wp\, \mathcal{A}] \to [\mathcal{G} \to \wp\, \mathcal{B}] && \text{(43a)}\\ \Phi \circledast_{\text{H}} \xi &:= \lambda g.\, \{f\, x \mid f \in \Phi\, g,\ x \in \xi\, g\} && \text{(43b)} \end{aligned}$$

$$\begin{aligned} \circledast_{\text{R}} &: \wp\, [\mathcal{G} \to [\mathcal{A} \to \mathcal{B}]] \times \wp\, [\mathcal{G} \to \mathcal{A}] \to \wp\, [\mathcal{G} \to \mathcal{B}] && \text{(44a)}\\ \Phi \circledast_{\text{R}} \xi &:= \{\lambda g. f\, g\, (x\, g) \mid f \in \Phi,\ x \in \xi\} && \text{(44b)} \end{aligned}$$

Both of these operations have a claim to the name **Pointwise Function Application**. The former is just Hamblin's version of application, introduced in (13), reified as a combinator. It combines an assignment-dependent set of functions from $\mathcal{A}$ to $\mathcal{B}$ with an assignment-dependent set of values in $\mathcal{A}$ to produce an assignment-dependent set of results in $\mathcal{B}$. The latter is the Roothian analog of this. It combines a set of assignment-dependent functions from $\mathcal{A}$ to $\mathcal{B}$ with a set of assignment-dependent values in $\mathcal{A}$ to produce a set of assignment-dependent results in $\mathcal{B}$. In other words, (43) would combine two denotations before Υ has gone to work on them, (44) after.

5 This rule, incidentally, is in line with Charlow's (2019a) interpretation of the abstraction rule proposed in Kotek 2017:

$$\{\!\!\{ n\ E \}\!\!\} := \lambda g.\, \left\{\lambda a.\, \iota p.\, p \in \{\!\!\{ E \}\!\!\}\, g^{n\mapsto a}\right\} \qquad \text{[Charlow 2019a: (16)]}$$

In other words, Kotek can be seen as adopting the special case of **Right** that applies to lambda abstraction, though she also argues, in effect, that when **Right** is inapplicable, abstraction should be undefined. In any case, the point is that **Right** does the only reasonable thing when alternatives are trivial (i.e., when Hamblin denotations are isomorphic ordinary denotations).

What **Nat** guarantees is that when Φ is deeply uninteresting, it won't make any difference which of (43) or (44) you pick. Let $\Phi = \lambda g.\, \{f\}$, for some function $f : \mathcal{A} \to \mathcal{B}$. If this is the Hamblin denotation of an expression, then that expression must be free of any alternative-generating constituents (declarative, unfocused, etc.), since the value at any assignment is a singleton set of alternatives. Likewise, it must be free of any assignment-sensitive constituents (traces, pronouns, etc.), since the value at every assignment is the same. And in fact, every such focus- and pronoun-free expression will have as its Hamblin denotation some function in the shape of Φ. In these circumstances, **Nat** is exactly the requirement that Υ is a homomorphism with respect to the two notions of Pointwise Function Application.

$$\Upsilon\, (\Phi \circledast_{\mathrm{H}} \xi) = \Upsilon\, \Phi \circledast_{\mathrm{R}} \Upsilon\, \xi \tag{45}$$

This, together with **Left** and **Right**, ensures that the alternatives generated by Υ depend only on the anaphoric and indeterminate components of its prejacent. Ordinary content rides free. For instance, consider the sentence 'her mom$_{\mathbf{F}}$ called John', with the referent of 'her' free and prosodic focus on mom. This sentence contains a mixture of anaphoric, indeterminate, and plain language. By **Nat**, we are assured that Hamblin application $\circledast_{\mathrm{H}}$ of the predicate 'called John' to the subject 'her mom' satisfies the following equations:

$$\begin{aligned}
\Upsilon\, \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \text{ called John} \,|\!\} &= \Upsilon\, (\{\!|\,\text{called John}\,|\!\} \circledast_{\mathrm{H}} \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\}) && \text{(46a)}\\
&= \Upsilon\, \{\!|\,\text{called John}\,|\!\} \circledast_{\mathrm{R}} \Upsilon\, \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\} && \text{(46b)}\\
&= \{\lambda g.\, \mathsf{call}\, \mathsf{j}\} \circledast_{\mathrm{R}} \Upsilon\, \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\} && \text{(46c)}\\
&= \{\lambda g.\, \mathsf{call}\, \mathsf{j}\, (\varphi\, g) \mid \varphi \in \Upsilon\, \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\}\} && \text{(46d)}
\end{aligned}$$

Combining this with the general recipe in (31), we are guaranteed that abstracting over n in the constituent $[t_n\text{'s mom}_{\mathbf{F}}$ called John$]$ will generate the set of alternative properties in (47).

$$\{\!|\, n\ [t_n\text{'s mom}_{\mathbf{F}} \text{ called John}] \,|\!\} = \lambda g.\, \{\lambda x.\, \mathsf{call}\, \mathsf{j}\, (\varphi\, g^{n \mapsto x}) \mid \varphi \in \Upsilon\, \{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\}\} \tag{47}$$

For each assignment-dependent entity φ that Υ transforms $\{\!|\, t_n\text{'s mom}_{\mathbf{F}} \,|\!\}$ into, we get a function that is true of x whenever $\varphi\, g^{n \mapsto x}$ called John. The important thing to see here is that the property of John-calling is what each alternative φ is tested for, but by **Nat**, this property can play no role in determining what the alternatives actually are. Those are determined entirely by the constituents containing pronouns and focus.

5 The impossibility of Hamblin abstraction

Here is the main result: the three requirements **Left**, **Right**, and **Nat** are together inconsistent. There cannot be any such Υ that is well-behaved on focus-free and pronoun-free language. And consequently there cannot be any means of interpreting an intensional operator $[\upsilon\ E]$ in terms of its ordinary behavior $[\![\upsilon]\!]$, and its prejacent's Hamblin denotation $\{\!|\, E \,|\!\}$.

The result actually has nothing to do with assignments, variables, or "alternatives" *per se*, as others have deduced (e.g., Rooth 1985: ch. II, pt. 3, Heim 2011: sec. 3, Charlow 2019a: pg. 10). So what is proved here is a more general fact about functions with certain signatures. We start by situating Hamblin's and Rooth's typing assumptions in a more abstract setting. Let $\langle \mathbb{S}, \mathbf{1}_{\mathrm{S}}, \mathbf{S} \rangle$ and $\langle \mathbb{R}, \mathbf{1}_{\mathrm{R}}, \mathbf{R} \rangle$ be the following functors, where $\mathcal{R}$ is any fixed set.

$$\mathbb{S}\,\mathcal{A} \coloneqq \wp\,\mathcal{A} \tag{48}$$

$$\mathbf{1}_{\mathsf{S}} : \mathcal{A} \to \mathbb{S}\,\mathcal{A} \tag{49a}$$

$$\mathbf{1}_{\mathsf{S}} \coloneqq \lambda x.\, \{x\} \tag{49b}$$

$$\mathbf{S} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{S}\,\mathcal{A} \to \mathbb{S}\,\mathcal{B}] \tag{50a}$$

$$\mathbf{S} \coloneqq \lambda f \lambda A.\, \{f\ x \mid x \in A\} \tag{50b}$$

$$\mathbb{R}\,\mathcal{A} \coloneqq [\mathcal{R} \to \mathcal{A}] \tag{51}$$

$$\mathbf{1}_{\mathsf{R}} : \mathcal{A} \to \mathbb{R}\,\mathcal{A} \tag{52a}$$

$$\mathbf{1}_{\mathsf{R}} \coloneqq \lambda x \lambda r.\, x \tag{52b}$$

$$\mathbf{R} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{R}\,\mathcal{A} \to \mathbb{R}\,\mathcal{B}] \tag{53a}$$

$$\mathbf{R} \coloneqq \lambda f \lambda \varphi \lambda r.\, f\ (\varphi\ r) \tag{53b}$$

$\mathbb{S}$ and $\mathbb{R}$ are functions from domains to domains. For any set $\mathcal{A}$, $\mathbb{S}\,\mathcal{A}$ returns the powerset of $\mathcal{A}$, and $\mathbb{R}\,\mathcal{A}$ returns the set of functions from $\mathcal{R}$ to $\mathcal{A}$. These constructors are both equipped with two polymorphic operations, a unit and a map. For any domain $\mathcal{A}$, the unital operation lifts *objects* in $\mathcal{A}$ to *objects* in the relevant domain image. The mapping operation lifts *functions* between domains $f : \mathcal{A} \to \mathcal{B}$ to *functions* between their images, e.g., $\mathbf{S}\ f : \mathbb{S}\,\mathcal{A} \to \mathbb{S}\,\mathcal{B}$. To say that $\mathbb{R}$ and $\mathbb{S}$ are **functors** is to say that these latter operations are homomorphisms for function composition, as the reader may verify:

$$\mathbf{S}\,(\lambda a.\, a) = \lambda A.\, A \tag{54}$$

$$\mathbf{S}\,(f' \circ f) = \mathbf{S}\ f' \circ \mathbf{S}\ f \tag{55}$$

$$\mathbf{R}\,(\lambda a.\, a) = \lambda \varphi.\, \varphi \tag{56}$$

$$\mathbf{R}\,(f' \circ f) = \mathbf{R}\ f' \circ \mathbf{R}\ f \tag{57}$$

From these, define the two composite functors $\langle \mathbb{R}\mathbb{S},\ \mathbf{1}_{\mathsf{RS}},\ \mathbf{RS}\rangle$ and $\langle \mathbb{S}\mathbb{R},\ \mathbf{1}_{\mathsf{SR}},\ \mathbf{SR}\rangle$. These are simply the compositions of the constructors $\mathbb{R}$ and $\mathbb{S}$, together with the compositions of their associated maps.

$$\mathbb{R}\mathbb{S}\,\mathcal{A} \coloneqq (\mathbb{R} \circ \mathbb{S})\,\mathcal{A} \tag{58}$$

$$= [\mathcal{R} \to \mathbb{S}\,\mathcal{A}] \tag{59}$$

$$\mathbf{1}_{\mathsf{RS}} : \mathcal{A} \to \mathbb{R}\mathbb{S}\,\mathcal{A} \tag{60a}$$

$$\mathbf{1}_{\mathsf{RS}} \coloneqq \mathbf{1}_{\mathsf{R}} \circ \mathbf{1}_{\mathsf{S}} \tag{60b}$$

$$= \lambda x \lambda r.\, \{x\} \tag{60c}$$

$$\mathbf{RS} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{R}\mathbb{S}\,\mathcal{A} \to \mathbb{R}\mathbb{S}\,\mathcal{B}] \tag{61a}$$

$$\mathbf{RS} \coloneqq \mathbf{R} \circ \mathbf{S} \tag{61b}$$

$$= \lambda f \lambda \Phi \lambda r.\, \{f\ x \mid x \in \Phi\ r\} \tag{61c}$$

$$\mathbb{S}\mathbb{R}\,\mathcal{A} \coloneqq (\mathbb{S} \circ \mathbb{R})\,\mathcal{A} \tag{62}$$

$$= \wp\,[\mathcal{R} \to \mathcal{A}] \tag{63}$$

$$\mathbf{1}_{\mathsf{SR}} : \mathcal{A} \to \mathbb{S}\mathbb{R}\,\mathcal{A} \tag{64a}$$

$$\mathbf{1}_{\mathsf{SR}} \coloneqq \mathbf{1}_{\mathsf{S}} \circ \mathbf{1}_{\mathsf{R}} \tag{64b}$$

$$= \lambda x.\, \{\lambda r.\, x\} \tag{64c}$$

$$\mathbf{SR} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{S}\mathbb{R}\,\mathcal{A} \to \mathbb{S}\mathbb{R}\,\mathcal{B}] \tag{65a}$$

$$\mathbf{SR} \coloneqq \mathbf{S} \circ \mathbf{R} \tag{65b}$$

$$= \lambda f \lambda \Phi.\, \{\lambda r.\, f\ (\varphi\ r) \mid \varphi \in \Phi\} \tag{65c}$$

When we fix $\mathcal{R}$ to be the domain of assignments $\mathcal{G}$, the first composition $\mathbb{R}\mathbb{S}$ is the set of possible Hamblin denotations. For any expression E of type σ, the set $\mathbb{R}\mathbb{S}\ \boldsymbol{D}_\sigma$ contains $\{\!|E|\!\}$. The latter composition $\mathbb{S}\mathbb{R}$ is the set of possible denotations à la Rooth.

The question put to us in (30) turns on what kinds of transformations between $\mathbb{R}\mathbb{S}$ and $\mathbb{S}\mathbb{R}$ are possible. For our purposes, let us say that a **transformation** $\Upsilon : \mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$ is a polymorphic function from the space of Hamblin denotations to those of Rooth. That is, for any domain $\mathcal{A}$, Υ determines a function from $\mathbb{R}\mathbb{S}\,\mathcal{A}$ to $\mathbb{S}\mathbb{R}\,\mathcal{A}$.

Then we say $\Upsilon : \mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$ is **natural** iff the following diagram commutes for all sets $\mathcal{A}$ and $\mathcal{B}$.

$$\begin{tikzcd}
\mathbb{R}\mathbb{S}\,\mathcal{A} \arrow[r, "\mathbf{RS} f"] \arrow[d, "\Upsilon"] & \mathbb{R}\mathbb{S}\,\mathcal{B} \arrow[d, "\Upsilon"] \\
\mathbb{S}\mathbb{R}\,\mathcal{A} \arrow[r, "\mathbf{SR} f"] & \mathbb{S}\mathbb{R}\,\mathcal{B}
\end{tikzcd}$$

That is, for any $f : \mathcal{A} \to \mathcal{B}$ and $\Phi : \mathbb{R}\mathbb{S}\,\mathcal{A}$, we have the following equivalence. When $\mathcal{R} = \mathcal{G}$, this is the eponymous law proposed in Section 4.

$$\Upsilon\,(\lambda r.\,\{f\,x \mid x \in \Phi\,r\}) = \{\lambda r. f\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\} \qquad \textbf{Nat}$$

We say a transformation $\Upsilon : \mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$ is **distributive** iff the following diagrams commute.

$$\begin{tikzcd} & \mathbb{S}\,\mathcal{A} \arrow[dl, "\mathbf{1}_{\mathsf{R}}"'] \arrow[dr, "\mathbf{S}\,\mathbf{1}_{\mathsf{R}}"] & \\ \mathbb{R}\mathbb{S}\,\mathcal{A} \arrow[rr, "\Upsilon"'] & & \mathbb{S}\mathbb{R}\,\mathcal{A} \end{tikzcd} \qquad \begin{tikzcd} & \mathbb{R}\,\mathcal{A} \arrow[dl, "\mathbf{R}\,\mathbf{1}_{\mathsf{S}}"'] \arrow[dr, "\mathbf{1}_{\mathsf{S}}"] & \\ \mathbb{R}\mathbb{S}\,\mathcal{A} \arrow[rr, "\Upsilon"'] & & \mathbb{S}\mathbb{R}\,\mathcal{A} \end{tikzcd}$$

That is, for any $A \in \mathbb{S}\,\mathcal{A}$ and $\varphi : \mathbb{R}\,\mathcal{A}$, we have the following equivalences. Again, when $\mathcal{R} = \mathcal{G}$, these are the laws of Section 4.

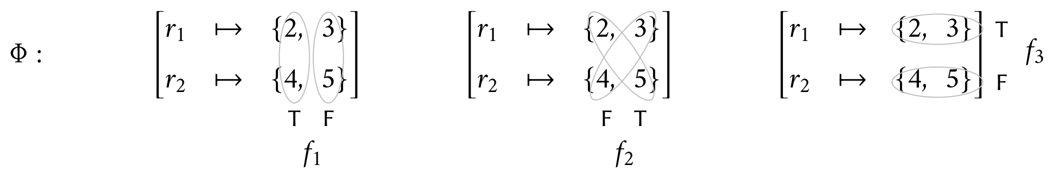
$$\Upsilon\,(\lambda r.\,A) = \{\lambda r.\,x \mid x \in A\} \qquad \textbf{Left}$$
$$\Upsilon\,(\lambda r.\,\{\varphi\,r\}) = \{\varphi\} \qquad \textbf{Right}$$

Claim: No transformation $\Upsilon : \mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$ is both natural and distributive.

Proof: Consider the following model.

$$\begin{array}{lll} \mathcal{R} := \{r_1, r_2\} & \Phi : \mathbb{R}\mathbb{S}\,\mathcal{A} & f_1, f_2, f_3 : \mathcal{A} \to \mathcal{B} \\ \mathcal{A} := \{2,3,4,5\} & \Phi := \begin{bmatrix} r_1 \mapsto \{2,3\} \\ r_2 \mapsto \{4,5\} \end{bmatrix} & \quad f_1\,n := \mathsf{T} \text{ iff } 2 \mid n \\ \mathcal{B} := \{\mathsf{T},\mathsf{F}\} & & \quad f_2\,n := \mathsf{T} \text{ iff } n \mid 10 \\ & & \quad f_3\,n := \mathsf{T} \text{ iff } n < 4 \end{array}$$

Φ is a function from $\mathcal{R}$ to $\wp\,\mathcal{A}$. It sends r_1 to the subset $\{2,3\}$ and r_2 to the subset $\{4,5\}$. The functions f_1, f_2, and f_3 map elements of $\mathcal{A}$ to truth values. The first of these f_1 is the characteristic function of even numbers in $\mathcal{A}$. The second is the characteristic function of the numbers in $\mathcal{A}$ that are factors of 10. And the third the characteristic function of numbers less than 4. As we'll see, the crucial configuration in this model is that the three functions cross-cut $\mathcal{A}$ in three different ways, relative to Φ:



Let Υ be a transformation $\mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$, and assume for the purposes of contradiction that Υ satisfies **Nat**, **Left**, and **Right**. Since Υ maps any object of type $\mathbb{R}\mathbb{S}\,\mathcal{A}$ to an object of type $\mathbb{S}\mathbb{R}\,\mathcal{A}$, it must in particular map Φ to some set of functions $\Upsilon\,\Phi$ in $\wp\,[\mathcal{R} \to \mathcal{A}]$.

Part 1

1. $\lambda r.\,\{f_1\,x \mid x \in \Phi\,r\} = \begin{bmatrix} r_1 \mapsto \{\mathsf{T},\mathsf{F}\} \\ r_2 \mapsto \{\mathsf{T},\mathsf{F}\} \end{bmatrix}$.

2. So by **Left**: $\Upsilon\,(\lambda r.\,\{f_1\,x \mid x \in \Phi\,r\}) = \Upsilon\,(\lambda r.\,\{\mathsf{T},\mathsf{F}\}) = \{\lambda r.\,\mathsf{T},\ \lambda r.\,\mathsf{F}\}$.

3. By **Nat**: $\Upsilon\,(\lambda r.\,\{f_1\,x \mid x \in \Phi\,r\}) = \{\lambda r.\,f_1\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\}$.

4. So $\{\lambda r.\,f_1\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\} = \{\lambda r.\,\mathsf{T},\ \lambda r.\,\mathsf{F}\}$.

5. This means that every function φ in $\Upsilon\,\Phi$ must be such that either (i) $\varphi\,r$ is even at every $r \in \mathcal{R}$, or (ii) $\varphi\,r$ is odd at every $r \in \mathcal{R}$. And $\Upsilon\,\Phi$ must include at least one of each variety.

Part 2

1. $\lambda r.\,\{f_2\,x \mid x \in \Phi\,r\} = \begin{bmatrix} r_1 & \mapsto & \{\mathsf{T},\mathsf{F}\} \\ r_2 & \mapsto & \{\mathsf{F},\mathsf{T}\} \end{bmatrix}$.

2. So by **Left**, $\Upsilon\,(\lambda r.\,\{f_2\,x \mid x \in \Phi\,r\}) = \Upsilon\,(\lambda r.\,\{\mathsf{T},\mathsf{F}\}) = \{\lambda r.\,\mathsf{T},\ \lambda r.\,\mathsf{F}\}$.

3. By **Nat**, $\Upsilon\,(\lambda r.\,\{f_2\,x \mid x \in \Phi\,r\}) = \{\lambda r.\,f_2\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\}$.

4. So $\{\lambda r.\,f_2\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\} = \{\lambda r.\,\mathsf{T},\ \lambda r.\,\mathsf{F}\}$.

5. This means that every function φ in $\Upsilon\,\Phi$ must be such that either (i) $\varphi\,r$ is a divisor of 10 at every $r \in \mathcal{R}$, or (ii) $\varphi\,r$ is a non-divisor of 10 at every $r \in \mathcal{R}$. And $\Upsilon\,\Phi$ must include at least one of each variety.

Part 3

1. $\lambda r.\,\{f_3\,x \mid x \in \Phi\,r\} = \begin{bmatrix} r_1 & \mapsto & \{\mathsf{T}\} \\ r_2 & \mapsto & \{\mathsf{F}\} \end{bmatrix}$.

2. So by **Right**, $\Upsilon\,(\lambda r.\,\{f_3\,x \mid x \in \Phi\,r\}) = \Upsilon \begin{bmatrix} r_1 & \mapsto & \{\mathsf{T}\} \\ r_2 & \mapsto & \{\mathsf{F}\} \end{bmatrix} = \left\{\begin{bmatrix} r_1 & \mapsto & \mathsf{T} \\ r_2 & \mapsto & \mathsf{F} \end{bmatrix}\right\}$.

3. By **Nat**, $\Upsilon\,(\lambda r.\,\{f_3\,x \mid x \in \Phi\,r\}) = \{\lambda r.\,f_3\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\}$.

4. So $\{\lambda r.\,f_3\,(\varphi\,r) \mid \varphi \in \Upsilon\,\Phi\} = \left\{\begin{bmatrix} r_1 & \mapsto & \mathsf{T} \\ r_2 & \mapsto & \mathsf{F} \end{bmatrix}\right\}$.

5. This means that every function φ in $\Upsilon\,\Phi$ must be such that $\varphi\,r_1 \in \{2,3\}$ and $\varphi\,r_2 \in \{4,5\}$. And $\Upsilon\,\Phi$ must include at least one such function.

Part 4

1. Given **Part 3**, if φ picks 2 for r_1, it must pick 4 or 5 for r_2. By **Part 1**, since 2 is even, $\varphi\,r_2$ must be even as well, so it must be 4; but by **Part 2**, since 2 divides 10, $\varphi\,r_2$ must also divide 10, so it must be 5. This is impossible, so φ can't pick 2 for r_1.

2. Given **Part 3**, if φ picks 3 for r_1, it must pick 4 or 5 for r_2. By **Part 1**, since 3 is odd, $\varphi\,r_2$ must be odd as well, so it must be 5; but by **Part 2**, since 3 doesn't divide 10, $\varphi\,r_2$ must also not divide 10, so it must be 4. This is impossible, so φ can't pick 3 for r_1.

3. So there can't be any $\varphi \in \Upsilon\,\Phi$, since by **Part 3** any such φ would have to pick either 2 or 3 for r_1, but neither choice is possible. And since $\Upsilon\,\Phi$ must be non-empty (by all three parts), we have a contradiction.

Thus there is no natural, distributive $\curlyvee : \mathbb{R}\mathbb{S} \to \mathbb{S}\mathbb{R}$, since any such function would have to make all three diagrams commute for Φ at f_1, f_2, and f_3, which is impossible. $\square$

It should be clear that I have used numbers and truth values here just to make the functions simple, familiar, and describable. The reasoning is entirely abstract as regards the elements of $\mathcal{R}$, $\mathcal{A}$, and $\mathcal{B}$. In particular, when $\mathcal{R}$ is $\mathcal{G}$, we are assured that there is no natural, distributive transformation between the two denotational spaces for composing alternatives at issue in this paper. Moreover, it follows immediately that *any* parameter to the ordinary denotation function $[\![\cdot]\!]$ will suffer the same fate. This includes worlds, times, indexical contexts, judges, etc. And consequently, we cannot expect to define abstractions or any other modal operators.

6 Outlook

I have argued that the Hamblin denotation of an intensional rule of composition depends on the existence of a transformation $\curlyvee : [\mathcal{G} \to \wp\,\mathcal{A}] \to \wp\,[\mathcal{G} \to \mathcal{A}]$, per (31) and (32), repeated here.

$$\{\!|\upsilon\ E|\!\} := \lambda g.\ \{[\![\upsilon]\!]\ \varphi\ g \mid \varphi \in \curlyvee\ \{\!|E|\!\}\} \tag{66}$$

$$\{\!|E\ F|\!\}_\mu := \lambda g.\ \left\{[\![\cdot]\!]_\mu\ (\varphi, \psi)\ g \;\middle|\; \begin{array}{l}\varphi \in \curlyvee\ \{\!|E|\!\},\\ \psi \in \curlyvee\ \{\!|F|\!\}\end{array}\right\} \tag{67}$$

The narrow formal result established in the previous section is that no such transformation can be both natural and distributive in the sense of **Left**, **Right**, and **Nat**. I've also suggested that these are requirements we should expect any decent Hamblin definition to meet. They guarantee very basic things about the correspondence between a rule's ordinary behavior and its Hamblin behavior.

But strictly speaking, the logic here is very weak. One is free to reject the entire framing in terms of (66) and (67) and/or reject any of the distributivity and naturality laws. Ultimately the question is about when a definition counts as a *reasonable generalization* of an operation. Of course, in analyzing a particular construct, we may assign it an arbitrary Hamblin denotation to see if it makes predictions we like. What the proof here shows is that the hopes of doing this in any *systematic* way, any way which is neutral about the ordinary semantics of the construct, are dashed.

Before closing, let me also point out that the laws here are highly abstract. They depend only on the functorial nature of $\mathbb{S}$ and $\mathbb{R}$. And sets are not the only functors that have been exploited to structure natural language denotations. To give two examples: First, Heim & Kratzer 1998 interpret expressions of type σ as denoting *partial* functions from assignments to $\boldsymbol{D}_\sigma$. The denotation of an expression like 'her$_1$ guitar' is only defined for assignments that map 1 to an individual with a guitar. Let $\mathbb{M}\ \mathcal{A} = \mathcal{A} \cup \{\bot\}$, where $\bot$ is an object not in the domain of any type. Then we may identify the partial denotation of an expression $\lfloor\!\lfloor E_\sigma \rfloor\!\rfloor$ as a function from $\mathcal{G}$ to $\mathbb{M}\ \boldsymbol{D}_\sigma$. If $[\![E_\sigma]\!]$ is defined at g, then the partial denotation and the ordinary denotation coincide; otherwise $\lfloor\!\lfloor E_\sigma \rfloor\!\rfloor\ g = \bot$.

Second, for various purposes ordinary denotations can be paired with *supplemental* content (e.g., Potts 2005, Martin 2013, Koev 2017). For concreteness, say that an expression like 'John, a linguist' denotes an entity paired with the proposition that John is a linguist. Let $\mathbb{W}\ \mathcal{A} = \mathcal{A} \times \boldsymbol{D}_\pi$, where π is the type of propositions. Under such assumptions, the bidimensional denotation of an expression $\langle\!|E_\sigma|\!\rangle$ lies in $\mathcal{G} \to \mathbb{W}\ \boldsymbol{D}_\sigma$; at any assignment, it denotes a pair whose left component is in $\boldsymbol{D}_\sigma$ and whose right component is in $\boldsymbol{D}_\pi$.

Both $\mathbb{M}$ and $\mathbb{W}$ are functors, under the associated maps below (where $\top$ is the denotation of a tautology):

$$\mathbb{M}\,\mathcal{A} \coloneqq \mathcal{A} \cup \{\bot\} \tag{68}$$

$$\mathbf{1}_{\mathsf{M}} : \mathcal{A} \to \mathbb{M}\,\mathcal{A} \tag{69a}$$

$$\mathbf{1}_{\mathsf{M}} \coloneqq \lambda a.\, a \tag{69b}$$

$$\mathbf{M} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{M}\,\mathcal{A} \to \mathbb{M}\,\mathcal{B}] \tag{70a}$$

$$\mathbf{M} \coloneqq \lambda f \lambda a.\, \begin{cases} \bot & \text{if } a = \bot \\ f\,a & \text{otherwise} \end{cases} \tag{70b}$$

$$\mathbb{W}\,\mathcal{A} \coloneqq \mathcal{A} \times \boldsymbol{D}_\pi \tag{71}$$

$$\mathbf{1}_{\mathsf{W}} : \mathcal{A} \to \mathbb{W}\,\mathcal{A} \tag{72a}$$

$$\mathbf{1}_{\mathsf{W}} \coloneqq \lambda a.\, \langle a,\, \top \rangle \tag{72b}$$

$$\mathbf{W} : [\mathcal{A} \to \mathcal{B}] \to [\mathbb{W}\,\mathcal{A} \to \mathbb{W}\,\mathcal{B}] \tag{73a}$$

$$\mathbf{W} \coloneqq \lambda f \lambda \langle a,\, p \rangle.\, \langle f\,a,\, p \rangle \tag{73b}$$

Consequently, their compositions $\mathbb{MR}$, $\mathbb{RM}$, $\mathbb{WR}$ and $\mathbb{RW}$ are all functors as well. Unsurprisingly, when attempting to extend ordinary compositional rules to rules in these enriched spaces, we run into precisely the same obstacles as in Section 2. Namely, the only way to utilize $[\![\upsilon]\!]$ is to provide it with a function $\varphi : \mathcal{G} \to \mathcal{A}$, but the relevant enriched denotations are functions from $\mathcal{G}$ to $\mathbb{M}\,\mathcal{A}$ or $\mathbb{W}\,\mathcal{A}$. So we find ourselves again in need of a transformations $\Upsilon_{\mathbb{M}} : [\mathcal{G} \to \mathbb{M}\mathcal{A}] \to \mathbb{M}\,[\mathcal{G} \to \mathcal{A}]$ and $\Upsilon_{\mathbb{W}} : [\mathcal{G} \to \mathbb{W}\mathcal{A}] \to \mathbb{W}\,[\mathcal{G} \to \mathcal{A}]$. Outfitted with such transformations, the following general templates would allow operations to be lifted as follows:

$$\lfloor\!\lfloor \upsilon\, E \rfloor\!\rfloor \coloneqq \lambda g.\, \begin{cases} \bot & \text{if } \Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor E \rfloor\!\rfloor = \bot \\ [\![\upsilon]\!]\, \left(\Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor E \rfloor\!\rfloor\right) g & \text{otherwise} \end{cases} \tag{74a}$$

$$\lfloor\!\lfloor E\, F \rfloor\!\rfloor_\mu \coloneqq \lambda g.\, \begin{cases} \bot & \text{if } \bot \in \left\{\Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor E \rfloor\!\rfloor,\, \Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor F \rfloor\!\rfloor\right\} \\ [\![\cdot]\!]_\mu\, \left(\Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor E \rfloor\!\rfloor,\, \Upsilon_{\mathbb{M}}\, \lfloor\!\lfloor F \rfloor\!\rfloor\right) g & \text{otherwise} \end{cases} \tag{74b}$$

$$\langle\!\langle \upsilon\, E \rangle\!\rangle \coloneqq \lambda g.\, \langle [\![\upsilon]\!]\, \varphi\, g,\, p \rangle, \quad \text{where } \langle \varphi,\, p \rangle = \Upsilon_{\mathbb{W}}\, \langle\!\langle E \rangle\!\rangle \tag{75a}$$

$$\langle\!\langle E\, F \rangle\!\rangle_\mu \coloneqq \lambda g.\, \left\langle [\![\cdot]\!]_\mu\, (\varphi, \psi)\, g,\, p \wedge q \right\rangle, \quad \text{where } \langle \varphi,\, p \rangle = \Upsilon_{\mathbb{W}}\, \langle\!\langle E \rangle\!\rangle,\ \langle \psi,\, q \rangle = \Upsilon_{\mathbb{W}}\, \langle\!\langle F \rangle\!\rangle \tag{75b}$$

These transformations have corresponding naturality and distributivity laws, derived by replacing $\mathbb{S}$ with $\mathbb{M}$ or $\mathbb{W}$ in the diagrams above. For any types σ and τ, and any $x \in \boldsymbol{D}_\sigma$, $p \in \boldsymbol{D}_\pi$, $m \in \mathbb{M}\,\boldsymbol{D}_\sigma$, $f : \boldsymbol{D}_\sigma \to \boldsymbol{D}_\tau$, $\varphi : \mathcal{G} \to \boldsymbol{D}_\sigma$, and $\Phi : \mathcal{G} \to \mathbb{M}\,\boldsymbol{D}_\sigma$ (or $\Phi : \mathcal{G} \to \mathbb{W}\,\boldsymbol{D}_\sigma$, as appropriate):

$$\Upsilon_{\mathbb{M}}\, (\lambda g.\, m) = \begin{cases} \bot & \text{if } m = \bot \\ \lambda g.\, m & \text{otherwise} \end{cases} \tag*{$\mathbb{M}.\textbf{Left}$}$$

$$\Upsilon_{\mathbb{M}}\, \varphi = \varphi \tag*{$\mathbb{M}.\textbf{Right}$}$$

$$\Upsilon_{\mathbb{M}} \left(\lambda g.\, \begin{cases} \bot & \text{if } \Phi\, g = \bot \\ f\,(\Phi\, g) & \text{otherwise} \end{cases}\right) = \begin{cases} \bot & \text{if } \Upsilon_{\mathbb{M}}\, \Phi = \bot \\ \lambda g.\, f\,((\Upsilon_{\mathbb{M}}\, \Phi)\, g) & \text{otherwise} \end{cases} \tag*{$\mathbb{M}.\textbf{Nat}$}$$

$$\Upsilon_{\mathbb{W}}\, (\lambda g.\, \langle x,\, p \rangle) = \langle \lambda g.\, x,\, p \rangle \tag*{$\mathbb{W}.\textbf{Left}$}$$

$$\Upsilon_{\mathbb{W}}\, (\lambda g.\, \langle \varphi\, g,\, \top \rangle) = \langle \varphi,\, \top \rangle \tag*{$\mathbb{W}.\textbf{Right}$}$$

$$\Upsilon_{\mathbb{W}}\, \left(\lambda g.\, \left\langle f\,(\Phi\, g)_0,\, (\Phi\, g)_1 \right\rangle\right) = \langle \lambda g.\, f\,((\Upsilon_{\mathbb{W}}\, \Phi)_0\, g),\, (\Upsilon_{\mathbb{W}}\, \Phi)_1 \rangle \tag*{$\mathbb{W}.\textbf{Nat}$}$$

I will mostly leave to future research the investigation of what exactly these constraints impose on transformations. It is clear that there *are* solutions in both cases. For instance, the Υs defined in (76) and (77) satisfy their three respective laws.

$$\Upsilon_{\mathbb{M}} \coloneqq \lambda \Phi.\, \begin{cases} \bot & \text{if } \bot \in \{\Phi\, g \mid g \in \mathcal{G}\} \\ \Phi & \text{otherwise} \end{cases} \tag{76}$$

$$\Upsilon_{\mathbb{W}} \coloneqq \lambda \Phi.\, \left\langle \lambda g.\, (\Phi\, g)_0,\, (\Phi\, g^*)_1 \right\rangle \tag{77}$$

In (76) any total function Φ is returned as is; any (non-total) partial function is sent to $\bot$. This yields a kind of Weak Kleene semantics for binding operators, such that if any assignment would lead the prejacent to failure, then the entire operation is undefined. (77) is considerably more arbitrary. For any $\Phi : \mathcal{G} \to \mathbb{W}\, \boldsymbol{D}_\sigma$, the transformation $\Upsilon_{\mathbb{W}}\, \Phi$ returns a pair whose left component is the function that at any g projects the left component of $\Phi\, g$, but whose right component is whatever supplemental content Φ takes at some fixed assignment g^*.

I take it to be obvious that at least the second of these would yield a completely inadequate semantics for, say, lambda abstraction. Clearly if these laws are relevant to the program of lifting denotations from one setting to another, they are *merely necessary* conditions. But then it is perhaps all the more surprising that no transition from Hamblin to Rooth can meet even this minimal muster.

References

Alonso-Ovalle, Luis. 2006. *Disjunction in alternative semantics*. University of Massachusetts Amherst PhD Dissertation.

Asudeh, Ash & Gianluca Giorgolo. 2020. *Enriched meanings: natural language semantics with category theory*. Vol. 13. Oxford University Press.

Barker, Chris. 2016. Why relational nominals make good concealed questions. *Lingua* 182. 12–29. https://doi.org/https://doi.org/10.1016/j.lingua.2016.01.002.

Beaver, David & Elizabeth Coppock. 2015. Novelty and familiarity for free. In *20th Amsterdam Colloquium*, 50–59.

Charlow, Simon. 2019a. E-closure and alternatives. *Linguistic Inquiry*. 1–18.

Charlow, Simon. 2019b. The scope of alternatives: Indefiniteness and islands. *Linguistics and Philosophy*. 1–46.

Grove, Julian. 2019. *Scope-taking and presupposition satisfaction*. The University of Chicago PhD Dissertation.

Hagstrom, Paul. 1998. *Decomposing questions*. Cambridge, MA: Massachusetts Institute of Technology PhD Dissertation.

Hamblin, C. L. 1973. Questions in Montague English. *Foundations of Language* 10(1). 41–53.

Heim, Irene. 1982. *The semantics of definite and indefinite noun phrases*. Amherst: University of Massachusetts Ph.D. Dissertation.

Heim, Irene. 2011. Compositional semantics of questions and wh-movement. Unpublished class notes: Topics in Semantics.

Heim, Irene & Angelika Kratzer. 1998. *Semantics in Generative Grammar*. Oxford: Blackwell.

Henkin, Leon. 1950. Completeness in the theory of types. *The Journal of Symbolic Logic* 15(2). 81–91.

Koev, Todor. 2017. Quotational indefinites. *Natural Language & Linguistic Theory* 35(2). 367–396.

Kotek, Hadas. 2017. Intervention effects arise from scope-taking across alternatives. In *The 47th annual meeting of the North East Linguistic Society (NELS)*, 153–166.

Kratzer, Angelika & Junko Shimoyama. 2002. Indeterminate pronouns: The view from Japanese. In Yukio Otsu (ed.), *Third Tokyo conference on psycholinguistics*, 1–25. Tokyo. https://doi.org/10.1007/978-3-319-10106-4_7.

Krifka, Manfred. 1991. A compositional semantics for multiple focus constructions. In Steven Moore & Adam Zachary Wyner (eds.), *Semantics and linguistic theory (SALT) 1*, 127–158. Cornell University: Cornell University Working Papers in Linguistics.

Lewis, David. 1975. Adverbs of quantification. In Edward Keenan (ed.), *Formal semantics of natural language*, 3–15. Cambridge, MA: Cambridge University Press.

Martin, Scott. 2013. *The dynamics of sense and implicature*. The Ohio State University PhD Dissertation.

Montague, Richard. 1973. The proper treatment of quantification in ordinary English. In *Approaches to natural language*, 221–242. Dordrecht: D. Reidel Publishing Company.

Partee, Barbara. 1986. Noun phrase interpretation and type-shifting principles. In Jeroen Groenendijk, Dick de Jongh & Martin Stokhof (eds.), *Studies in Discourse Representation Theory and the theory of generalized quantifiers*, 115–144. Dordrecht: Foris. https://doi.org/10.1002/9780470751305.ch10.

Potts, Christopher. 2005. *The logic of conventional implicatures* (Oxford Studies in Theoretical Linguistics). Oxford: Oxford University Press.

Reynolds, John. 1983. Types, abstraction and parametric polymorphism. In *Information processing 83: proceedings of the IFIP 9th world computer congress*, 513–523. Amsterdam.

Romero, Maribel & Marc Novel. 2013. Variable binding and sets of alternatives. In *Alternatives in semantics*, 174–208. Springer.

Rooth, Mats. 1985. *Association with focus*. Amherst, MA: University of Massachusetts, Amherst PhD Dissertation.

Shan, Chung-chieh. 2004. Binding alongside Hamblin alternatives calls for variable-free semantics. In Laurence Horn & Gregory Ward (eds.), *Handbook of pragmatics* (Blackwell Handbooks in Linguistics), 289–304. Blackwell Publishers Ltd.