Satisfaction without provisos

Julian Grove

DRAFT: DECEMBER 15, 2019

Abstract Heim's satisfaction theory of presupposition projection has paved the way for a successful research tradition within dynamic semantics that has been able to give compositional analyses of a variety of presupposition projection behaviors. Since Geurts 1996, however, the promise of this research program has been tarnished by the observation that it systematically leads to incorrect predictions in cases where a presupposition ends up filtered which should not have been. I show that the satisfaction account of presupposition projection is in good stead, after all, by revealing that Geurts's observations are valid only under certain basic assumptions about semantic composition. To illustrate, I present a satisfaction account of presupposition projection that builds on the work of Charlow (2014, 2019a,b) in the context of a semantics based on alternatives, where, indeed, the proviso problem crops up. I slightly sophisticate the compositional repertoire, leading to a setting in which the proviso problem disappears. I additionally demonstrate some extensions of the proposed system for binding and presupposition accommodation.

1 Introduction

Ever since Geurts 1996, the influential "satisfaction theory" of presupposition projection introduced by Heim (1983) has been known to suffer from certain descriptive inadequacies: specifically, it attributes weak, conditional presuppositions to many sentences which are felt by native speakers to have stronger, unconditional presuppositions.

(1) If Theo has a brother, he'll bring his wetsuit.
    $\rightsquigarrow$ Theo has a wetsuit.

While (1) is usually judged to have the presupposition that Theo has a wetsuit, Heim's account predicts it to have the conditional presupposition that Theo has a wetsuit *if he has a brother*, due to the semantics she attributes to the conditional. Geurts dubs this general difficulty of Heim's account, which pertains to a variety of operators, the "proviso problem".

The central aim of this paper is to explore the proviso problem, and, more generally, what I call the problem of "trapped presupposition triggers", within a particular variant of Heim's satisfaction account, but couched within a theory of dynamic semantics that regards meanings as giving rise to alternatives. The account I present is largely faithful to Heim's formulation of dynamic semantics, and more generally, the Stalnakerian program

that regards utterance contexts as sets of worlds to be pruned by new assertions. Like Heim, I will regard a sentence's presuppositions to be given by its definedness conditions: sentence S_1 presupposes sentence S_2 iff the set of worlds at which S_1 is defined is (or entails) the proposition denoted by S_2.[1] My basic claim is that contemporary satisfaction accounts within the Heimian program suffer from the proviso problem because they rely on a view of meaning composition and discourse update that invokes a particular mathematical structure: that of an *applicative functor*. I show that once this is recognized, it is straightforward to upgrade the composition scheme so that it gives rise to the more powerful notion of a *monad*. Once composition is viewed in terms of a monad, the proviso problem disappears from the satisfaction account.

One result of this new picture of presupposition is that sentences like (1) will be regarded as semantically ambiguous. (1), for example, will be taken to presuppose *either* that Theo has a wetsuit if he has a brother, *or* that he has a wetsuit, depending on how the ambiguity is resolved. Indeed, as we will see, this ambiguity will be regarded as a kind of scope ambiguity. From this point of view, the challenge of relating semantic presuppositions to the presuppositions which are actually observed is a problem of describing how pragmatic context drives the resolution of scope ambiguity.

The paper will go as follows. In §2, I present my version of the satisfaction account. While it shares its basic tenets with previous accounts falling within the "satisfaction theory" initated by Heim (1983), my system is developed on top of the dynamic semantics introduced in Charlow 2014: it assumes that indefinite noun phrases give rise to *alternatives*, which are composed by a kind of pointwise functional application. The main addition over Charlow's account is the introduction of (first) intensionality and (then) presupposition. In §2.2, I use a trivalent logic to introduce a means of encoding presuppositions, and in §2.3, I illustrate the limitations placed on the resulting system by its applicative nature. In §3, I introduce a notion of *scope-taking* into the compositional framework by reconsidering it as a monad. As we will see, the proviso problem and, more generally, the problem of trapped presupposition triggers disappears in this new setting. In §4, I provide extensions of the framework for binding and presupposition accommodation. We will see here that it is straightforward to give compositional analyses of binding by indefinite noun phrases introduced inside intensional contexts, like the complements of propositional attitude verbs. Finally, §5 concludes.

2 The satisfaction account

2.1 A grammar for propositions

In my presentation of the satisfaction account, I make the standard Stalnakerian assumption that propositions are sets of possible worlds, but with a minor twist, the motivation for which will soon become clear: propositions here are sets of *pairs of possible worlds and truth values*. For example, the sentence in (2)

[1] I mostly sidestep anaphoric presuppositions in this paper, though see §4.1 for some discussion.

(2) A dolphin swam.

denotes the following proposition (where $\top$ and $\bot$ are the truth values 'true' and 'false'):

$$\{\langle w, \top\rangle \mid (\exists x.\textbf{dolphin}wx \wedge \textbf{swam}wx)\}$$
$$\cup\ \{\langle w, \bot\rangle \mid (\exists x.\textbf{dolphin}wx \wedge \neg\textbf{swam}wx)\}$$

Thus (2) denotes a set of pairs containing, for any world w, $\langle w, \top\rangle$ if there is a dolphin who swam in w, and $\langle w, \bot\rangle$ if there is a dolphin who didn't swim in w. (2) is *true* in some world w if $\langle w, \top\rangle \in [\![(2)]\!]$ and false otherwise. Thus given a proposition ϕ, we may recover its truth conditions as $\{w \in \mathcal{W} \mid \langle w, \top\rangle \in \phi\}$.

This way of looking at things is essentially alternative in nature: indefinites give rise to alternative meanings, which are ultimately reflected in the truth values with which a given world is paired by the denotation of a sentence. This encoding of propositions allows for a simple scheme for semantic composition which invokes an *intensional* form of pointwise functional application. The reader may recognize this view of indefinites as similar to that found in Charlow 2014, 2019a,b, i.a.; as we shall see, my current assumptions about basic meanings (though not composition) are essentially an import of Charlow's, up to the addition of intensionality. In the meantime, we may encode our propositions into the simply typed λ-calculus by regarding them as functions of type $s \to t \to t$, or, functions from worlds to functions from truth values to truth values. On this strategy, the set described above receives the following encoding:

$$(\lambda w, a.(\exists x.\textbf{dolphin}wx \wedge (a = \textbf{swam}wx)))$$

It regards a pair of a world and truth value as in the set if (a) there is a dolphin in the world and (b) the truth value is $\top$ if that dolphin swam and $\bot$ if not.

Here is how our assumption about propositions lends itself to a simple compositional scheme for interpreting sentences. While sentences denote sets of alternative pairs of worlds and truth values, noun phrases will denote sets of alternative pairs of worlds and individuals, while predicates denote sets of alternative pairs of worlds and functions of type $e \to t$. Thus the indefinite noun phrase *a dolphin* denotes the set

$$\{\langle w, x\rangle \mid \textbf{dolphin}wx\}$$

or, the set of pairs whose first component is some or other possible world and whose second component is some or other individual which is a dolphin in that world. Its semantic type is thus $s \to e \to t$. Meanwhile, *swam* denotes the set

$$\{\langle w, (\lambda x.\textbf{swam}wx)\rangle \mid w \in \mathcal{W}\}$$

or, the set of pairs whose first member is a possible world and whose second member is the function from an individual to $\top$ iff the individual swam in the world. The semantic type of *swam* is therefore $s \to (e \to t) \to t$.

To compose the meaning of *a dolphin* with the meaning of *swam*, we may define the following intensional, pointwise variants of forward and backward functional application, $(\overset{\odot}{\triangleright})$ and $(\overset{\odot}{\triangleleft})$:

$$m \overset{\odot}{\triangleright} n \coloneqq \{\langle w, fx\rangle \mid \langle w, f\rangle \in m \wedge \langle w, x\rangle \in n\}$$
$$m \overset{\odot}{\triangleleft} n \coloneqq \{\langle w, fx\rangle \mid \langle w, x\rangle \in m \wedge \langle w, f\rangle \in n\}$$

Note that the notation used in these definitions is really just syntactic sugar for the following, more explicit definitions:

$$m \overset{\odot}{\triangleright} n \coloneqq (\lambda w, o.(\exists f, x.mwf \wedge nwx \wedge o = fx))$$
$$m \overset{\odot}{\triangleleft} n \coloneqq (\lambda w, o.(\exists f, x.mwx \wedge nwf \wedge o = fx))$$

While the casual set-theoretic notation is currently harmless, it will be important to be explicit about exactly what functions it stands for once we've added presupposition to the grammar. The derivation of (2) can now be given as in Figure 1, which involves a single instance of backward functional application, as just defined. The result is exactly the set which was stated above to be the denotation of (2): for any world w, $\langle w, \top\rangle$ is in this set if there is a dolphin in w who swam in w, while $\langle w, \bot\rangle$ is in the set if there is a dolphin in w who didn't swim in w.

$$\{\langle w, \mathbf{swam}wx\rangle \mid \mathbf{dolphin}wx\}$$
$$|=$$
$$\{\langle w, x\rangle \mid \mathbf{dolphin}wx\} \overset{\odot}{\triangleleft} \{\langle w, (\lambda x.\mathbf{swam}wx)\rangle \mid w \in \mathcal{W}\}$$

$\{\langle w, x\rangle \mid \mathbf{dolphin}wx\}$ — *a dolphin*

$\{\langle w, (\lambda x.\mathbf{swam}wx)\rangle \mid w \in \mathcal{W}\}$ — *swam*

Figure 1: Deriving (2)

What about sentences without indefinites? Consider (3), for example.

(3) Flippy swam.

We would like to be able to assign (3) the following denotation, which pairs each world with exactly one alternative (either $\top$ or $\bot$):

$$\{\langle w, \mathbf{swam}w\mathbf{f}\rangle \mid w \in \mathcal{W}\}$$

That is, the set of pairs of possible worlds and truth values containing $\langle w, \top\rangle$ if Flippy swam in w and $\langle w, \bot\rangle$ if Flippy didn't swim in w. To countenance single-alternative-invoking noun phrases, like *Flippy*, we may provide the following type shift, $(\cdot)^{\uparrow\overset{\odot}{}}$, a polymorphic, intensional variant of Partee (1986)'s ident; given something of type α, $(\cdot)^{\uparrow\overset{\odot}{}}$ lifts

it into a set of pairs of possible worlds and α's.

$$(\cdot)^{\uparrow}_{\odot} : \alpha \to s \to \alpha \to t$$
$$a^{\uparrow}_{\odot} = \{\langle w, a\rangle \mid w \in \mathcal{W}\}$$

This type shift may then be applied to the denotation of *Flippy*, as in Figure 2.

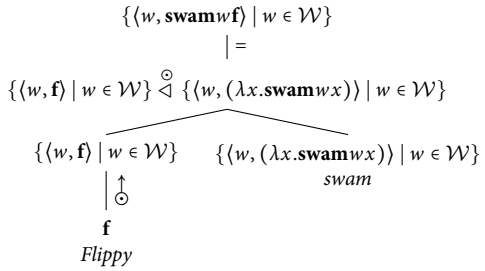

Figure 2: Deriving (3)

In summary, we have lifted variants of forward and backward functional application ($(\overset{\odot}{\triangleright})$ and $(\overset{\odot}{\triangleleft})$), along with an operator $(\cdot)^{\uparrow}_{\odot}$, to compose propositional meanings in a way that handles the semantics of indefinite noun phrases. In the meantime, note that there is a generalization about how the semantics of expressions in this miniature grammar relate to the first-order, extensional meanings they are usually assigned: when an expression has semantic type α in the first-order, extensional setting, it has type $s \to \alpha \to t$ in the current setting. Sentences, normally of semantic type t, now have type $s \to t \to t$, while noun phrases, normally of semantic type e, now have type $s \to e \to t$. The operator $(\cdot)^{\uparrow}_{\odot}$ is designed to reflect this general fact: it merely lifts first-order, extensional values into the enriched meanings of the current setting, changing what they do "as little as possible".

Before we extend our current compositional setting to encode presuppositions, let us remark on an important property it has. It gives rise to a certain mathematical structure familiar to many functional programmers: that of an *applicative functor* (McBride and Paterson, 2008). An applicative functor is a certain map A on semantic types (given a type α, $\mathsf{A}(\alpha)$ is some new type), equipped with two operators, $(\cdot)^{\uparrow}_{\mathsf{A}}$ (called 'unit') and $(\cdot)^{\mathsf{A}}_{\updownarrow}$ (called 'apply'), satisfying certain laws. These operators have the following type signatures, in terms of A:[2]

$$(\cdot)^{\uparrow}_{\mathsf{A}} : \alpha \to \mathsf{A}(\alpha)$$
$$(\cdot)^{\mathsf{A}}_{\updownarrow} : \mathsf{A}(\alpha \to \beta) \to \mathsf{A}(\alpha) \to \mathsf{A}(\beta)$$

[2] I use the notation for applicative functors introduced in Kobele 2018 for graded applicative functors. The original notation of McBride and Paterson (2008) treats $(\cdot)^{\uparrow}_{\mathsf{A}}$ as a prefix 'ι' and $(\cdot)^{\mathsf{A}}_{\updownarrow}$ as an infix '$(\circledast)$'.

The motivation for viewing composition in terms of an applicative functor is that it allows a way to view *enriched* composition between meanings of type $\mathsf{A}(\alpha)$ (for some α) as composition of ordinary meanings of type α that give rise to *side effects*. Side effects are effects of composition that go above and beyond the application of functions to arguments; e.g., by invoking alternatives. Given an applicative functor A, we may compose meanings with side effects by using $(\cdot)^{\overset{\mathsf{A}}{\uparrow}}$ and $(\cdot)^{\overset{\mathsf{A}}{\updownarrow}}$. In these terms, $(\cdot)^{\overset{\mathsf{A}}{\uparrow}}$ injects an ordinary value into the applicative functor by endowing it with trivial side effects—those which do not add or subtract from the side effects with which they are sequenced. $(\cdot)^{\overset{\mathsf{A}}{\updownarrow}}$ exposes the argument of a function with side effects by turning it into a function *between* values with side effects.

If A is an applicative functor, then it is also a *functor*, meaning that it is associated with an operation $\mathsf{map}_{\mathsf{A}}$, obeying certain functor laws, with the following type signature:

$$\mathsf{map}_{\mathsf{A}} : (\alpha \to \beta) \to \mathsf{A}(\alpha) \to \mathsf{A}(\beta)$$

When A is also an *applicative* functor, $\mathsf{map}_{\mathsf{A}}$ may be identified simply with $(\lambda f.f^{\overset{\mathsf{A}}{\uparrow}\overset{\mathsf{A}}{\updownarrow}})$. I henceforth generally use the '$\mathsf{map}_{\mathsf{A}}$' notation when applicable.

In our case, the applicative functor of interest is $\odot$, defined as $\odot(\alpha) = s \to \alpha \to t$. $(\cdot)^{\overset{\mathsf{A}}{\uparrow}}$ is, of course, $(\cdot)^{\overset{\odot}{\uparrow}}$; meanwhile, the operator $(\cdot)^{\overset{\odot}{\downarrow}}$ may be identified with intensional pointwise functional application:

$$u^{\overset{\odot}{\downarrow}} = (\lambda v.\{\langle w, fx\rangle \mid \langle w, f\rangle \in u \wedge \langle w, x\rangle \in v\})$$

Correspondingly, $\mathsf{map}_{\odot}$ receives the following definition:

$$\begin{aligned}\mathsf{map}_{\odot} &: (\alpha \to \beta) \to \odot(\alpha) \to \odot(\beta)\\ \mathsf{map}_{\odot} f &= f^{\overset{\odot}{\uparrow}\overset{\odot}{\downarrow}}\\ &= (\lambda m.\{\langle w, fx\rangle \mid \langle w, x\rangle \in m\})\end{aligned}$$

Thus it maps a function *over* an intensional meaning that invokes alternatives by ignoring the presence of both the intensionality and the structure of the alternatives, cutting straight to the values underneath.

Generally speaking, let $(*)$ be some binary operation of type $\alpha \to \beta \to \gamma$, for some choice of α, β, and γ (it may, for example, be ordinary functional application). Then given an applicative functor A, we may lift $(*)$ into its applicative variant as follows:

$$\begin{aligned}(*) &: \alpha \to \beta \to \gamma\\ (\overset{\mathsf{A}}{*}) &: \mathsf{A}(\alpha) \to \mathsf{A}(\beta) \to \mathsf{A}(\gamma)\\ m \overset{\mathsf{A}}{*} n &\coloneqq (\mathsf{map}_{\mathsf{A}}(*)m)^{\overset{\mathsf{A}}{\updownarrow}} n\end{aligned}$$

Both forward and backward intensional, pointwise functional application ($(\overset{\odot}{\triangleright})$ and $(\overset{\odot}{\triangleleft})$) can be viewed in this way, where $(\triangleright)$ and $(\triangleleft)$ are ordinary forward and backward func-

tional application, respectively. In other words, it is straightforward to verify that $(\overset{\odot}{\triangleright})$ is just $(\lambda m, n.(\mathsf{map}_{\odot}(\triangleright)m)\overset{\odot}{\downarrow}n)$ and that $(\overset{\odot}{\triangleleft})$ is just $(\lambda m, n.(\mathsf{map}_{\odot}(\triangleleft)m)\overset{\odot}{\downarrow}n)$.

When considering examples of presupposition projection later on, we'll see that the applicative nature of our compositional scheme is responsible for the proviso problem. Although we will be able to compose meanings that register presuppositions, we lack a means of scope-taking, the mechanism by which presuppositions project past potential filters. Without scope-taking, in other words, we'll end up forcing the presuppositions of expressions to be evaluated as these expressions are composed with other expressions. Let us now investigate this limitation.

2.2 *Adding presupposition*

At the moment, ordinary values of type α are being encoded as sets of pairs of worlds and objects of type α; that is, functions of type $s \to \alpha \to t$. We have no way, as things stand, of encoding the meanings of expressions with presuppositions, but it is an easy transition to a new setting in which presuppositions take effect. Following the tradition of trivalent logics and their associated approaches to presupposition (see, e.g., Kleene, 1938; Herzberger, 1973; Martin, 1979; Beaver, 1999, 2001; Beaver and Krahmer, 2001; Coppock and Beaver, 2015, among many others), we will add a third truth value, #, to accompany the two truth values $\top$ and $\bot$. We do this by introducing a new type, $t_{\#}$, inhabited by all three truth values:

$$\top : t_{\#}$$
$$\bot : t_{\#}$$
$$\# : t_{\#}$$

is the "undefined" truth value of trivalent logics; we will use it to encode undefinedness, which in turn represents that a given expression's presuppositions are not satisfied. In the meantime, we will also hold onto the type t of classical truth values $\top$ and $\bot$.[3]

In order to encode meanings which may possibly be undefined, we may use the δ-operator of Beaver 1999.[4] δ is defined as follows:

$$\delta : t \to t_{\#}$$
$$\delta\top = \top$$
$$\delta\bot = \#$$

$\ulcorner\delta\urcorner$ takes a formula representing a truth condition and turns it into one representing a definedness condition; e.g., where $\ulcorner\mathbf{dolphin}x\urcorner$ may have been false, $\ulcorner\delta(\mathbf{dolphin}x)\urcorner$ is now undefined.[5]

[3] In order not to change the underlying type theory, we can add a constant $\mathbf{id} : t \to t_{\#}$ with the behavior $\mathbf{id}\top = \top$ and $\mathbf{id}\bot = \bot$, which is then used implicitly in the promotion of formulae from t to $t_{\#}$.

[4] And of work following it: see, e.g., Beaver 2001; Beaver and Krahmer 2001, i.a..

[5] In order to maintain clarity, mentions of metalanguage symbols and expressions are wrapped in corner quotes ($\ulcorner\cdot\urcorner$) to distinguish them from uses.

Since it leads to a concise and perpsicuous presentation of the upcoming proposals about presupposition projection behavior, I will assume the semantics for the logical connectives ⌜∧⌝, ⌜→⌝, ⌜¬⌝ of Table 1.

$\wedge$	$\top$	$\bot$	#
$\top$	$\top$	$\bot$	#
$\bot$	$\bot$	$\bot$	#
#	#	#	#

$\rightarrow$	$\top$	$\bot$	#
$\top$	$\top$	$\bot$	#
$\bot$	$\top$	$\top$	#
#	#	#	#

$\neg$	
$\top$	$\bot$
$\bot$	$\top$
#	#

Table 1: Semantics of the connectives

Note that this semantics forces the definedness conditions of formulae to automatically project. For example, regardless of the classical truth value associated with one of the conjuncts of ⌜∧⌝, if the other conjunct is valued as #, then so is the conjunction as a whole. Likewise for ⌜→⌝. Meanwhile, ⌜¬⌝ preserves the definedness of its scope.[6]

To interpret existentially quantified formulae of the metalanguage, I will assume the following semantics. Given a model $\mathcal{M}$ and an assignment g, $[\![\ulcorner(\exists x.\phi)\urcorner]\!]_{\mathcal{M},g}$ is true if there is an assignment g' that differs from g at most on what it assigns to x, such that $[\![\phi]\!]_{\mathcal{M},g'}$ is true. Meanwhile, $[\![\ulcorner(\exists x.\phi)\urcorner]\!]_{\mathcal{M},g}$ is false if there is at least one assignment g' that differs from g at most on what it assigns to x, such that $[\![\phi]\!]_{\mathcal{M},g'}$ is false and $[\![\phi]\!]_{\mathcal{M},g'}$ is either false or undefined for every other such assignment g'. Hence, $[\![\ulcorner(\exists x.\phi)\urcorner]\!]_{\mathcal{M},g}$ is undefined only if $[\![\phi]\!]_{\mathcal{M},g'}$ is undefined for every such assignment g'. Table 2 summarizes this interpretation scheme. (The notation '$g[x]g'$' means that assignment g differs from assignment g' at most on what it assigns to x.)

$\{[\![\phi]\!]_{\mathcal{M},g'} \mid g[x]g'\}$	$[\![\ulcorner(\exists x.\phi)\urcorner]\!]_{\mathcal{M},g}$
$\{\top\}$	$\top$
$\{\bot\}$	$\bot$
$\{\#\}$	#
$\{\top,\bot\}$	$\top$
$\{\top,\#\}$	$\top$
$\{\bot,\#\}$	$\bot$
$\{\top,\bot,\#\}$	$\top$

Table 2: Semantics of ⌜∃⌝

This semantics for existentially quantified formulae gives rise, for example, to the following logical truths:

$$(\exists x, y.x = y) = \top$$
$$(\exists x, y.\delta(x = y)) = \top$$
$$(\exists x.\neg(\exists y.x = y)) = \bot$$
$$(\exists x.\delta(\neg(\exists y.x = y))) = \#$$

[6] This semantics corresponds to the "internal" logic of Bochvar (1939); it is also known as "weak Kleene".

Given this addition to the types and interpretation conventions, we can begin to set up a system for encoding presuppositions. Rather than the applicative functor $\odot$, we instead use a new applicative functor $\circledast$, defined as follows:

$$\circledast(\alpha) = s \to \alpha \to t_{\#}$$

$$(\cdot)^{\overset{\uparrow}{\circledast}} : \alpha \to \circledast(\alpha)$$
$$a^{\overset{\uparrow}{\circledast}} = (\lambda w, x.\delta(x = a))$$

$$(\cdot)^{\overset{\circledast}{\downarrow}} : \circledast(\alpha \to \beta) \to \circledast(\alpha) \to \circledast(\beta)$$
$$u^{\overset{\circledast}{\downarrow}} = (\lambda v, w, r.(\exists f, x.uwf \wedge vwx \wedge \delta(r = fx)))$$

If we ignore the possiblity of a # arising, $\circledast$ behaves exactly as does $\odot$, the applicative functor for intensional, pointwise composition of alternative sets. What $\circledast$ adds is the possibility for membership in a set to be *undefined*! Thus while we may say, for any $x : \alpha$, world w, and $m : \odot(\alpha)$, that either $\langle w, x\rangle \in m$ or $\langle w, x\rangle \notin m$, this no longer holds for m of type $\circledast(\alpha)$. It may be that either $\langle w, x\rangle \in m$ or $\langle w, x\rangle \notin m$, but it may also happen that such a decision cannot be made; i.e., if $mwx = \#$. Such a strategy of encoding presuppositions as definedness conditions on membership in the alternative set an expression denotes renders a natural reinterpretation of the satisfaction account of presupposition within our alternative-based setting. Note that a *proposition* in this setting is a set of type $\circledast(t)$. Thus given a world w, we may say (as above) that a proposition ϕ is true at w if $\langle w, \top\rangle \in \phi$ and, moreover, that it is *undefined* at w if it cannot be determined whether or not $\langle w, \top\rangle \in \phi$; i.e., if $\phi w \top = \#$.

Before introducing English examples, let us first remark on a couple of important properties of this new applicative functor. First, note that we have defined $(\cdot)^{\overset{\uparrow}{\circledast}}$ so that, for any value a, membership of $\langle w, x\rangle$ in the set $a^{\overset{\uparrow}{\circledast}}$ is defined only when $x = a$. Thus $a^{\overset{\uparrow}{\circledast}}$ is like an ordinary characteristic function of a set, but which returns #'s in place of $\bot$'s for non-elements. This feature is ultimately harmless, though necessary in order to ensure that $(\cdot)^{\overset{\uparrow}{\circledast}}$ works together with $(\cdot)^{\overset{\circledast}{\downarrow}}$ to make $\circledast$ an applicative functor (see Appendix B).

Second, $(\cdot)^{\overset{\circledast}{\downarrow}}$ is defined so that, given two sets $u : \circledast(\alpha \to \beta)$ and $v : \circledast(\alpha)$, possibly with definedness conditions on membership, these definedness conditions *project* in the new set $u^{\overset{\circledast}{\downarrow}} v$. Note that, for any possible value r (of the right type) and possible world w, $\langle w, r\rangle \in u^{\overset{\circledast}{\downarrow}} v$ iff r is the result of applying some f, such that $\langle w, f\rangle \in u$ to some x, such that $\langle w, x\rangle \in v$. But in order for membership of $\langle w, r\rangle$ in $u^{\overset{\circledast}{\downarrow}} v$ to be defined, there must be a way of determing membership of some $\langle w, f\rangle$ in u and of some $\langle w, x\rangle$ in v so that $r = fx$. Given a collection of pairs $\langle f, x\rangle$ such that (a) $r = fx$, (b) either $\langle w, f\rangle \in u$ or $\langle w, f\rangle \notin u$, and (c) either $\langle w, x\rangle \in v$ or $\langle w, x\rangle \notin v$, then $r \in u^{\overset{\circledast}{\downarrow}} v$ if there is *some* member $\langle f', x'\rangle$ of the collection such that $f' \in u$ and $x' \in v$. If there is no such member of the

collection, then $r \notin u \overset{\circledast}{\downarrow} v$. The important point is that there must be such a collection in the first place to ensure definedness: given the result of sequentially applying set u to set v, membership of r is undefined just in case the only way of generating it is from "undefined members" of u and v.

Such an applicative helps us make sure that presuppositions project in the normal course of events (i.e., when no filters are present). Given a sentence a sentence like (4)

(4) The girl saw the dolphin.

we want to ensure that it denotes a proposition for which membership of $\langle w, \top \rangle$ is defined only if there is both a girl in w and a dolphin in w. Let us now see how the desired projection behavior is achieved through some examples.

Recall, first, that the meaning of the indefinite noun phrase *a dolphin* is the following set:

$$\{\langle w, x \rangle \mid \mathbf{dolphin} w x\}$$

In our current setting, we may encode this set as the following function of type $\circledast(e)$, i.e., $s \to e \to t_{\#}$:

$$(\lambda w, x.\mathbf{dolphin} w x)$$

Now, rather than the indefinite noun phrase *a dolphin*, let us consider the definite description *the dolphin*. Since it is a noun phrase, the semantic type of *the dolphin* is still $\circledast(e)$, but we will take it to denote the following function instead:

$$(\lambda w, x.\delta(\mathbf{dolphin} w x))$$

Given a world and an individual, this function returns $\top$ if the individual is a dolphin in the world, and # otherwise. Unlike an indefinite, whose truth condition is that its referent satisfies the property denoted by its restriction, a definite description demotes this truth condition to a definedness condition. In keeping with the notation above, we may represent the meaning of this definite description as follows:

$$\{\langle w, x \rangle \mid \delta(\mathbf{dolphin} w x)\}$$

Let us now take a look at the following sentence featuring a definite description.

(5) The dolphin was fast.

Given the definitions of applicative forward and backward functional application, we may derive the interpretation of (5) as in Figure 3. The set comprehension shown at the top of the derivation tree is just syntactic sugar for the following function:

$$\begin{aligned} & (\lambda w, a.(\exists f, x.\delta(\mathbf{dolphin} w x) \wedge f = (\lambda y.\mathbf{fast} w y) \wedge \delta(a = f x))) \\ = \; & (\lambda w, a.(\exists x.\delta(\mathbf{dolphin} w x) \wedge a = \mathbf{fast} w x)) \end{aligned}$$

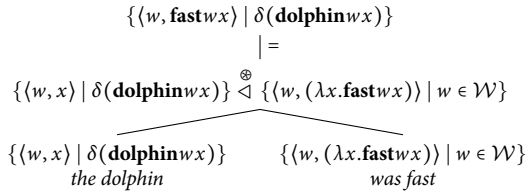

Figure 3: Deriving (5)

Given a world w, $\langle w, \top\rangle$ is in the resulting set if there is a dolphin who was fast in w, while $\langle w, \bot\rangle$ is in the resulting set if there is a dolphin who wasn't fast in w. If there are no dolphins in w, however, $\langle w, \top\rangle$ (as well as $\langle w, \bot\rangle$) gets mapped to #; that is, it cannot be determined whether or not (5) is true at w! We have thus captured the existence presupposition of a sentence with a definite description in terms of the sentence's definedness conditions. More generally, we have identified a sentence's presuppositions with the set of worlds at which it is defined: a sentence is "defined" at w iff it can be determined whether or not the sentence is true at w; i.e., whether or not $\langle w, \top\rangle$ is an element of the proposition it denotes.

Let's now consider the discourse in (6).

(6) A dolphin swam. The dolphin was fast.

To analyze (6), we must introduce some mechanism for discourse update. Note that (6), as a whole, does not have presuppositions; the existence presupposition of the second sentence has been filtered by the at-issue content of the first sentence. To aid in the definition our update operator, we can introduce a new connective, ⌜&⌝, with the semantics shown in Table 3.[7]

&	⊤	⊥	#
⊤	⊤	⊥	#
⊥	⊥	⊥	⊥
#	#	#	#

Table 3: Semantics of ⌜&⌝

Rather than allow the definedness conditions of its two conjuncts to project automatically (as ⌜∧⌝ does), ⌜&⌝ first checks whether or not its left conjunct is true, before paying any attention to its right conjunct, inducing a kind of left-to-right bias. If the left conjunct is either ⊥ or #, the right conjunct is ignored; hence, the definedness conditions of the right conjunct feature in the conjunction as a whole only if the left conjunct is ⊤.

[7] Peters (1979) introduces this semantics for conjunction in order to model the projection behavior described in Karttunen 1973. It has also been called "middle Kleene" after the "strong" trivalent logic of Kleene 1952.

Indeed, the truth-functional behavior of ⌜&⌝ models the presupposition projection behavior of discourse update on a world-by-world basis, suggesting that we should define our operator as follows:

$$(+) : \circledast(t) \to \circledast(t) \to \circledast(t)$$
$$\phi + \psi \coloneqq \{\langle w, a\rangle \mid \langle w, \top\rangle \in \phi \;\&\; \langle w, a\rangle \in \psi\}$$

To update ϕ with ψ, we take the subset of ψ containing worlds in which ϕ is true. Moreover, due to the left-to-right bias of ⌜&⌝, membership of a pair $\langle w, a\rangle$ in the proposition gotten as a result of the update will only be checked if the discourse itself is true in w. As a result, the presupposition of the new sentence is filtered: if pw is the definedness condition of ϕ at some world w and sw is the definedness condition of ψ, then the definedness conditions of $\phi + \psi$ at w are $pw \wedge (\langle w, \top\rangle \in \phi \to sw)$. The presuppositions of the sentence with which a discourse is updated thus become *conditionalized* on the truth conditions of the discourse once the update is performed.

The present account thus emulates the pattern of projection behavior typically found in satisfaction accounts in the tradition of Stalnaker 1973, 1974, Karttunen 1973, and (most pertinently) Heim 1983.[8] These accounts typically assume that a sentence may felicitously update the common ground only if its presuppositions are satisfied by the latter.[9] At the same time, they tend to identify the presuppositions of a sentence or a piece of discourse with the contexts it may felicitously update. These assumptions lead to the prediction of filtration of a sentence's presuppositions when that sentence occurs in a larger piece of discourse: if S_1 and S_2 update the discourse in turn, the felicity of S_2 is evaluated in a new context in which S_1 is true; thus the presuppositions of $S_1; S_2$ as a piece of discourse are weakened to those of S_1 and the implication that S_1 entails those of S_2. Rather than identify a sentence's presuppositions with the conditions it imposes on discourses it may update, the present account identifies them with the definedness conditions of the proposition it denotes, but to the same effect. The semantics of discourse update here has it that propositional definedness conditions are *weakened* to depend on the truth conditions of prior sentences.[10]

The derivation of (6) in Figure 4 illustrates our definition of discourse update. The meaning derived at the root of the tree shows that the existence presupposition has been filtered. Why? Note that the function encoding the resulting proposition is the following:

$$(\lambda w, a.(\exists x.\mathbf{dolphin}wx \wedge \mathbf{swam}wx) \;\&\; (\exists y.\delta(\mathbf{dolphin}wy) \wedge a = \mathbf{fast}wy))$$

The pair $\langle w, \top\rangle$ is thus in this set if there is a dolphin who swam in w and a dolphin who was fast in w. Because the conjunct ⌜$\mathbf{dolphin}wx$⌝ occurs to the left of the conjunct ⌜$\delta(\mathbf{dolphin}wy)$⌝, the latter conjunct is only evaluated if x is a dolphin in w, and hence,

[8] As pointed out in Francez 2019, Heim was really the first to predict the projection behavior just described, as she was the first to explicitly identify a sentence's presuppositions with the contexts it may felicitously update.

[9] von Fintel (2008) refers to this principle as "Stalnaker's bridge".

[10] This change in perspective is innocent and could be recast in more traditional terms (see, e.g., Rothschild, 2011). For example, propositions in the current setting could be made relations between contexts of type $\circledast(t) \to \circledast(t) \to t_\#$ via a type shift $(\cdot)^{\Uparrow}$, defined as $\psi^{\Uparrow} \coloneqq (\lambda\phi, \phi'.(\forall w.\phi w\top \to \psi w\top \neq \#) \wedge \phi' = \phi + \psi)$.

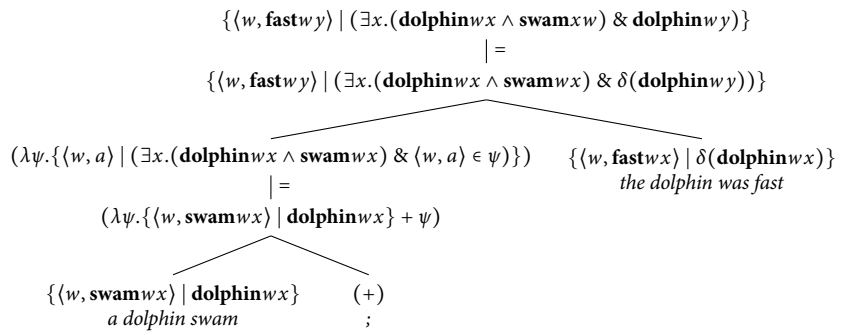

Figure 4: Deriving (6)

if there is some dolphin in w. As a result, some or other witness that verifies the formula $\ulcorner$**dolphin**$yw\urcorner$ will be found in all possible worlds w for which $\ulcorner\delta($**dolphin**$yw)\urcorner$ is evaluated in the first place. And because of the semantics of existential quantification in Table 2, such a witness is all it takes. A rather glaring aspect of the interpretation assigned to (6) is that the dolphin who swam need not be the same as the dolphin who was fast! This result obtains despite the fact that the existence presupposition of the definite description in the second sentence of (6) is filtered. The issue of binding is deferred to subsection 4.1, where this problem is overcome.

It is time now to explore some problems associated with the present compositional scheme, beginning with the proviso problem.

2.3 Trapped presupposition triggers

The compositional system just explored has a structure that limits its empirical adequacy. The relevant structure is that of an applicative functor. In order to avoid the limitations, we will have to make the compositional scheme a little more sophisticated.

The first limitation can be observed in conditional sentences like (1), repeated from the introduction.

(1) If Theo has a brother, he'll bring his wetsuit.

While (1) is most easily understood as presupposing that Theo has a wetsuit, the satisfaction account of Heim 1983 predicts it to have the presupposition that Theo has a wetsuit if he has a brother. Geurts (1996) argues that sentences like (1) exemplify the proviso problem, which he takes to be a difficulty generally associated with satisfaction accounts in the tradition established by Heim. As we will now see, the applicative account proposed here analogously leads to the same prediction.

To see why, let us first assign a semantics to conditional sentences. To do so, we need a notion of propositional negation, in terms of which we can state the semantics of the

conditional. To negate propositions, we introduce the following operator not:

$$\begin{aligned} \mathsf{not} &: \circledast(t) \to \circledast(t) \\ \mathsf{not}\phi &\coloneqq \{\langle w, \top\rangle \mid \langle w, \top\rangle \notin \phi\} \\ &= (\lambda w, a.\neg\phi w \top \wedge a = \top) \end{aligned}$$

Given a proposition ϕ (possibly with presuppositions), $\mathsf{not}\phi$ is a new proposition, such that for any possible world w, $\langle w, \top\rangle \in \mathsf{not}\phi$ just in case $\langle w, \top\rangle \notin \phi$. Meanwhile, if membership of $\langle w, \top\rangle$ in ϕ is undefined, then so is membership of either $\langle w, \top\rangle$ or $\langle w, \bot\rangle$ in $\mathsf{not}\phi$. Thus indefinites in the scope of not lose their scopal potential. At the same time, however, presuppositions in the scope of not retain their scopal potential, projecting upwards.

We may use the operator not to define a semantics for conditional sentences, as follows:

$$\begin{aligned} \mathsf{if} &: \circledast(t) \to \circledast(t) \to \circledast(t) \\ \mathsf{if}\phi\psi &\coloneqq \mathsf{not}(\phi + \mathsf{not}\psi) \end{aligned}$$

According to this semantics, $\mathsf{if}\phi\psi$ is true if updating ψ with the negation of ϕ is false. Thus any possible world in which ϕ is true but ψ is false is filtered out, so that only worlds in which the truth of ϕ guarantees the truth of ψ remain.[11] The derivation for (6) in Figure 5 uses this semantics (setting aside the resolution the embedded subject pronoun for this example).

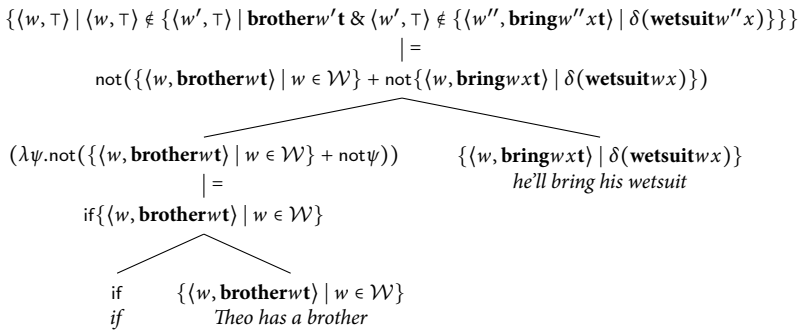

Figure 5: Deriving (1)

One can see by inspecting the specification of the proposition derived that the following formula must be true in any possible world w, in order for membership of $\langle w, \top\rangle$ in this proposition to be defined:

$$\mathbf{brother}wt \to (\exists x.\mathbf{wetsuit}wx)$$

[11] This semantics is only for the material conditional. See Heim 1983, Groenendijk and Stokhof 1991, Dekker 1994, Barker and Shan 2008, and Charlow 2014, among many others, for similar dynamic formulations.

This is more easily observed by first noticing that the derived proposition is encoded by the following function:

$$(\lambda w, a.\neg(\mathbf{brother}w\mathbf{t}\ \&\ \neg(\exists x.\delta(\mathbf{wetsuit}wx) \wedge \mathbf{bring}wx\mathbf{t})) \wedge a = \top)$$

Because $\ulcorner\neg\urcorner$ preserves undefinedness, the definedness condition of the second conjunct of $\ulcorner\&\urcorner$ is that Theo has a wetsuit. Since the first conjunct states that Theo has a brother, the semantics of $\ulcorner\&\urcorner$ guarantees that the presupposition of the second conjunct will only be evaluated in those worlds in which *Theo has a brother* is true. Finally, again because $\ulcorner\neg\urcorner$ preserves undefinedness, the definedness condition of the t-type body of the innermost abstraction is that Theo has a wetsuit if he has a brother.

This result exemplifies the proviso problem in the current setting. It is generally taken for granted that (1) presupposes not the weak presupposition that Theo has a wetsuit if he has a brother, but the stronger presupposition that Theo has a wetsuit. Examples like (1) contrast with (7), in which the weaker presupposition is, in fact, observed.

(7) If Theo is a scuba diver, he'll bring his wetsuit.

(7) does appear to have the presupposition that if Theo is a scuba diver, he has a wetsuit. This is the presupposition that would correctly be derived by the present account (and the account of Heim).

The proviso problem arises for conditional sentences in the present account for two reasons. First, the semantics of the conditional is stated in terms of the update operator, which has the filtering behavior characteristic of discourse update in satisfaction accounts. Second, the presuppositional side effects of the sentence *he'll bring his wetsuit* are forced to undergo evaluation right when the sentence composes with the rest of the conditional. This order of evaluation of side effects is forced in the present account because of the type of the conditional operator: $\circledast(t) \rightarrow \circledast(t) \rightarrow \circledast(t)$. Both the propositions denoted by the consequent and antecedent sentences are consumed by the operator, and their side effects may be tinkered with at its will. In this sense, the presupposition trigger *his wetsuit* is trapped in its local context—that of the consequent of the conditional.

We now investigate another manifestation of this problem, having to do with propositional attitude verbs. Given a reasonably straightforward semantics for propositional attitude verbs, the problem again arises for many examples in which a presupposition trigger is embedded in a propositional attitude verb's complement, as (8) illustrates.[12]

(8) Mary believes the dog is a cat.

The most natural reading of (8) is a *de re* one, according to which it presupposes the existence of a dog. A straightforward semantics for propositional attitude verbs, however, attributes to it a *de dicto* reading and thus predicts it to presuppose that Mary believes

[12] See Heim 1992 for her discussion of an analogous problem for her semantics of propositional attitude verbs which regards them as filters, following suggestions in Karttunen 1974.

that there is a dog. On the predicted reading, Mary is entailed to have inconsistent beliefs; i.e., that there is a dog which is also a cat. Let us see why this erroneous prediction would be made.

First, in order to give a semantics to verbs, like *believe*, which have some kind of universal modal force attached to them, it is useful to introduce a semantics for the universal quantifier. Thus, given a model $\mathcal{M}$ and an assignment g, we take $[\![\ulcorner(\forall x.\phi)\urcorner]\!]_{\mathcal{M},g}$ to be true if every assignment g' which differs from g at most on what it assigns to x is such that $[\![\phi]\!]_{\mathcal{M},g'}$ is true. We take $[\![\ulcorner(\forall x.\phi)\urcorner]\!]_{\mathcal{M},g}$ to be false if at least one assignment g' which differs from g at most on what it assigns to x is such that $[\![\phi]\!]_{\mathcal{M},g'}$ is false, while all other such assignments g' are such that $[\![\phi]\!]_{\mathcal{M},g'}$ is either true or false. Finally, $[\![\ulcorner(\forall x.\phi)\urcorner]\!]_{\mathcal{M},g}$ is undefined just in case there is an assignment g' which differs from g at most on what it assigns to x such that $[\![\phi]\!]_{\mathcal{M},g'}$ is undefined. Table 4 summarizes this interpretation scheme.

$\{[\![\phi]\!]_{\mathcal{M},g'} \mid g[x]g'\}$	$[\![\ulcorner(\forall x.\phi)\urcorner]\!]_{\mathcal{M},g}$
$\{\top\}$	$\top$
$\{\bot\}$	$\bot$
$\{\#\}$	$\#$
$\{\top,\bot\}$	$\bot$
$\{\top,\#\}$	$\#$
$\{\bot,\#\}$	$\#$
$\{\top,\bot,\#\}$	$\#$

Table 4: Semantics of $\ulcorner\forall\urcorner$

Note that, given such a semantics for the universal quantifier, we lose the equivalence between $\ulcorner(\forall x.\phi)\urcorner$ and $\ulcorner\neg(\exists x.\neg\phi)\urcorner$ (and thus between $\ulcorner(\exists x.\phi)\urcorner$ and $\ulcorner\neg(\forall x.\neg\phi)\urcorner$). Given a formula $\ulcorner(\forall x.\phi)\urcorner$, evaluating it involves checking the truth of ϕ against every possible value of x; hence, it returns an undefined result if one of those values leads to an undefined result. If the universal quantifier were defined in terms of the existential quantifier, in contrast, then making a universally quantified formula false would only be a matter of finding some value of x that falsified ϕ, regardless of whether or not other values of x lead ϕ to be undefined.

Given this semantics for universal quantification, let us also introduce a new operator $\ulcorner\Rightarrow\urcorner$, which has filtering behavior analogous to that of $\ulcorner\&\urcorner$. The semantics for this operator is shown in Table 5.[13] Like $\ulcorner\&\urcorner$, $\ulcorner\Rightarrow\urcorner$ has a left-to-right bias: if its left conjunct is $\bot$, it ignores the right conjunct and returns $\top$; only if the left conjunct is $\top$ does it bother checking the right conjunct in the first place.

We may now define the following abbreviation $\ulcorner\mathsf{believe}\urcorner$, in order to provide a seman-

[13] $\ulcorner\Rightarrow\urcorner$ is thus one of Peters's connectives, i.e., middle Kleene.

$\Rightarrow$	$\top$	$\bot$	#
$\top$	$\top$	$\bot$	#
$\bot$	$\top$	$\top$	$\top$
#	#	#	#

Table 5: Semantics of $\ulcorner\Rightarrow\urcorner$

tics for the verb *believes*:[14]

$$\mathsf{believe} : \circledast(t) \to \circledast(e) \to \circledast(t)$$
$$\mathsf{believe}\phi m \coloneqq \{\langle w, \top\rangle \mid (\exists x.\langle w, x\rangle \in m \wedge (\forall w'.\mathbf{dox}_{x,w} w' \Rightarrow \langle w', \top\rangle \in \phi))\}$$

Thus (8) above will be true at a possible world w just in case, for every world w' that is doxastically accessible to Mary at w, the pair $\langle w', \top\rangle$ is in the proposition denoted by *the dog is a cat*.

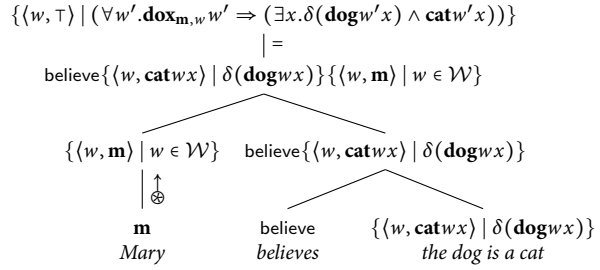

Figure 6: Deriving (8)

The derivation tree in Figure 6 illustrates how we arrive at this result. By inspecting the root, we can see that membership in the set denoted by (8) is defined for a pair $\langle w, \top\rangle$ just in case the following formula is true at w:

$$(\forall w'.\mathbf{dox}_{\mathbf{m},w} w' \to (\exists x.\mathbf{dog} w' x))$$

This can be seen clearly by observing that the function encoding the set derived is the following one:

$$(\lambda w, a.(\forall w'.\mathbf{dox}_{\mathbf{m},w} w' \Rightarrow (\exists x.\delta(\mathbf{dog} w' x) \wedge \mathbf{cat} w' x)) \wedge a = \top)$$

The semantics of implication provided in Table 5 (along with the foregoing semantics for $\ulcorner\forall\urcorner$) dictates that membership in this set of a pair whose first component is w is defined only if some dog can be found in every world w' which is doxastically accessible to Mary at w. Assuming that the semantics for the sentence *there is a dog* is (or is entailed by) the proposition

[14] We follow here the standard formulation of the possible-world semantics of this verb which universally quantifies over worlds doxastically accessible to the agent in the world of evaluation (see, e.g., Hintikka, 1969).

$$(\lambda w, a.a = (\exists x.\mathbf{dog}wx))$$

(8) itself will presuppose the sentence *Mary believes there is a dog*. The natural *de re* reading of (8) according to which it entails that there is a dog and does not ascribe inconsistent beliefs to Mary is not derivable given our present compositional scheme. This is because the semantics of the verb *believes* ensures that it checks the definedness of its complement at each doxastically accessible world. At the same time, a sentence's presuppositions are encoded as definedness conditions on membership in the set the sentence denotes; thus the sentence's presuppositions are filtered by the verb.

Both conditionals and propositional attitude verbs provide contexts in which a sentence's presuppositions may end up "trapped" within their immediate environment, forcing filtration and delivering a semantics for these constructions inconsistent with the facts. In the case of conditionals, a trapped presupposition trigger results in presuppositions which are too weak (the proviso problem), while in the case of propositional attitude verbs, it may force certain unwanted presuppositions about the attitude-holder's beliefs. An appropriate semantics for bouletic verbs like *want* and *hope* is likely to give rise to a similar issue.

(9) a. Mary wants the dog to come inside.

b. Mary hopes that John knows it's raining.

On their most natural readings out of the blue, (9a) presupposes that there is a dog, while (9b) presupposes that it's raining. In the right contexts, however, (9a) can be understood to presuppose that Mary believes that there is a dog, while (9b) can be understood to presuppose that Mary believes that it's raining (Karttunen, 1974). For example, the presuppositions of (9a) and (b) are nullified when these sentences are preceded by (10a) and (b), respectively.

(10) a. Mary believes that there is a dog.

b. Mary believes that it's raining.

Thus an adequate semantics for the verbs *want* and *hope* ought to render them filters, along with the verb *believe*. But if they are filters in the same vein as *believe*, then we would lack an account of (9), in which these verbs do not act as filters, but as *holes* to presupposition projection.[15] In general, there are options about what a presupposition trigger takes to be its local context, i.e., where its presuppositions are evaluated: sometimes, its local context is its immediate environment, as when the presupposition it gives rise to is filtered; others, it is a more global environment, as when the presupposition projects unmodified (as in (1) and (9) uttered out of the blue).

The assumptions made above about semantic composition lead to the problem of trapped presupposition triggers. Our applicative functor $\circledast$ does not equip us with the technology to let an expression's presuppositions *escape* the expression's local context,

[15] Karttunen (1973) suggests that propositional attitude verbs may behave this way in general.

thus projecting past filters. What we would like is to have a mechanism of presuppositional scope-taking. Presupposition triggers ought to be able to stay in their local contexts and be filtered, but they should also have the option to scope out and project.

3 A grammar with scope-taking

The compositional scheme developed above is not powerful enough to allow presuppositional scope-taking, since it merely gives rise to an applicative functor. What we need, instead, is a monad.[16] A monad M equips us not only with the operators $(\cdot)^{\uparrow_{\mathsf{M}}}$ and $(\cdot)^{\mathsf{M}\downarrow}$ determining an applicative functor, but a third operator μ_{M} (called 'join') with the following type signature, in terms of M:

$$\mu_{\mathsf{M}} : \mathsf{M}(\mathsf{M}(\alpha)) \to \mathsf{M}(\alpha)$$

In order for M to be a monad, μ_{M}, $(\cdot)^{\uparrow_{\mathsf{M}}}$, and $(\cdot)^{\mathsf{M}\downarrow}$ are required to satisfy certain monad laws, in addition to the applicative functor laws already satisfied by $(\cdot)^{\uparrow_{\mathsf{M}}}$ and $(\cdot)^{\mathsf{M}\downarrow}$. What μ_{M} allows us to do, which $(\cdot)^{\uparrow_{\mathsf{M}}}$ and $(\cdot)^{\mathsf{M}\downarrow}$ do not on their own, is collapse a meaning with two layers of side effects (that is, a meaning with side effects, which, *itself*, has side effects) into one, resulting in a mere meaning with side effects. On top of our applicative functor $\circledast$, we add the operator $\mu_{\circledast}$, making it a monad:

$$\mu_{\circledast} m = \{\langle w, x\rangle \mid (\exists n.\langle w, n\rangle \in m \wedge \langle w, x\rangle \in n)\}$$

Given some m which is a set of world-set pairs, where both the main set and the contained paired sets potentially have definedness conditions on membership, $\mu_{\circledast}$ transforms m into a set containing all the members of the (paired) sets in m which preserve the world with which they are paired. Thus membership of any pair $\langle w, x\rangle$ in $\mu_{\circledast} m$ is *positive* iff there is at least one pair $\langle w, n\rangle \in m$ such that $\langle w, x\rangle \in n$. Meanwhile, membership of any $\langle w, x\rangle$ in $\mu_{\circledast} m$ is *undefined* just in case membership of $\langle w, x\rangle$ in n is undefined for every n such that membership of n in m is defined (i.e., such that either $\langle w, n\rangle \in m$ or $\langle w, n\rangle \notin m$). For any pair $\langle w, x\rangle$, $\langle w, x\rangle \notin \mu_{\circledast} m$ just in case there is no n such that $\langle w, x\rangle \in n$ and $\langle w, n\rangle \in m$, but at least one n is such that membership of $\langle w, x\rangle$ of in n and membership of $\langle w, n\rangle$ in m is defined.

This definition of $\mu_{\circledast}$ ensures that $\circledast$ constitutes a monad. Importantly, it has the consequence that, given a set m containing a set n, *both of which* have definedness conditions on membership, $\mu_{\circledast} m$ preserves the definedness conditions of both. For example, if n

[16] For the introduction of monads into functional programming, see the series of papers Wadler 1992a,b, 1993, 1994. Monads, a concept from category theory, were first introduced to computer scientists in Moggi 1989 as a way of assigning semantics to computer programs that exhibit side effects.

Monads were introduced into linguistic semantics in Shan 2001. Since then, many authors have used them in the analysis of various semantic phenomena, including, e.g., Giorgolo and Unger (anaphora) and Giorgolo and Asudeh (conventional implicature), and more recently Charlow (2014, 2019a,b, i.a.) in the study of the exceptional scope of indefinites and anaphora. See also Grove 2019, which studies presupposition in terms of *graded monads*.

is one of a collection of sets, membership of a pair $\langle w, x\rangle$ in which is defined only if x is a dog in w, and m is the set of some subset of these n's, membership of a pair $\langle w, n\rangle$ in which is defined only if the members of n are fluffy in w, then $\mu_{\circledast} m$ will be a set the definedness conditions on whose membership relation is such that membership of a pair $\langle w, x\rangle$ is defined iff x is a fluffy dog in w.

Given our way of regarding $\circledast$ as a monad, we may introduce a slightly more advanced compositional scheme that allows for presuppositional scope-taking. As we will see, $(\cdot)^{\overset{\uparrow}{\circledast}}$ and $(\cdot)^{\overset{\circledast}{\downarrow}}$ can work together to free the scope of any given presupposition trigger, allowing it to escape from its local context, while $\mu_{\circledast}$ may be used to fix it. Here is how. We start with the meaning of the noun phrase *the dolphin*, given, again, as follows:

$$\{\langle w, x\rangle \mid \delta(\mathbf{dolphin} wx)\}$$

Recall that this meaning is of type $\circledast(e)$. In order to free the scope of this presupposition-triggering noun phrase, we push up the outer presuppositional layer by applying $(\cdot)^{\overset{\uparrow}{\circledast}}$ *internally*; that is, by *mapping* it over the meaning of *the dolphin* via $\mathsf{map}_{\circledast}$:

$$\begin{aligned}
&\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}\{\langle w, x\rangle \mid \delta(\mathbf{dolphin} wx)\}\\
=\ &\{\langle w, x^{\overset{\uparrow}{\circledast}}\rangle \mid \delta(\mathbf{dolphin} wx)\}\\
=\ &\{\langle w, \{\langle w', x\rangle \mid w' \in \mathcal{W}\}\rangle \mid \delta(\mathbf{dolphin} wx)\}\\
=\ &(\lambda w, r.(\exists x.\delta(\mathbf{dolphin} wx) \wedge \delta(r = (\lambda w', y.\delta(y = x)))))
\end{aligned}$$

What we now have is a meaning of type $\circledast(\circledast(e))$. We can compose this high-typed noun phrase with a verb phrase, like *swam*, by lifting the verb phrase meaning itself using $(\cdot)^{\overset{\uparrow}{\circledast}}$ and composing the two meanings using a doubly-lifted backward functional application, as in Figure 7.[17]

Finally, we may fix the scope of *the dolphin* by lowering the outer presuppositional layer back down using $\mu_{\circledast}$.

So far, it is not really apparent that we have gained anything by adding internal lift and $\mu_{\circledast}$ to our compositional scheme; we could have achieved the same result by composing the normal (unlifted) meanings for *the dolphin* and *swam* via $(\overset{\circledast}{\triangleleft})$. What we will now show is that the monadic nature of $\circledast$ makes all the difference in accounting for presupposition triggers that scope past presupposition filters.

3.1 Conditionals

Recall the problem of trapped presupposition triggers, as manifest in the context of conditional sentences. When a presupposition trigger cannot escape from its local context in an example like (1), it leads to the proviso problem.

[17] $(\overset{\circledast}{\circledast}\triangleleft)$ is denoted by '$\overset{\circledast^2}{\triangleleft}$'.

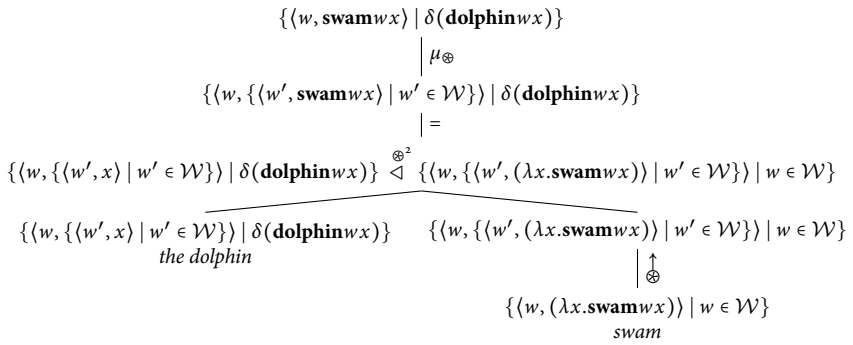

Figure 7: Deriving *the dolphin swam*

(1) If Theo has a brother, he'll bring his wetsuit.

Due to the semantics of the conditional, the presupposition predicted of (1) ends up too weak, viz., that Theo has a wetsuit if he has a brother. To obtain the correct, unconditional presupposition, we simply apply an internal lift to *his wetsuit* before composing it with the rest of the conditional and then lower the resulting meaning via $\mu_{\circledast}$. The internal lift has the effect of *freeing* the presupposition triggered by the consequent of the conditional, only for it to finally be fixed by $\mu_{\circledast}$ once the whole conditional is composed. The new derivation of (1) is illustrated in Figure 8.[18]

As the reader can verify, the proposition derived in Figure 8 is encoded by the following function:

$$(\lambda w, a.(\exists x.\delta(\textbf{wetsuit}wx) \wedge \neg(\textbf{brother}w\textbf{t} \,\&\, \neg\textbf{bring}wx\textbf{t}) \wedge a = \top))$$

It can be seen that, for any world w, membership of $\langle w, \top\rangle$ in the encoded set is defined just in case the following formula is true:

$$(\exists x.\textbf{wetsuit}wx)$$

Thus the presupposition attributed to (1) by Figure 8 is that Theo has a wetsuit, rather than the weaker conditional presupposition derived earlier.

Two ingredients to the analysis were crucial in obtaining this result, both of which are illustrated in Figure 8. The first was the application of $\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}$ to the meaning of the presupposition trigger, represented by '$\circledast(\overset{\uparrow}{\circledast})$' in the derivation tree. This step had the effect of moving the noun phrase's presupposition to an outer effectual layer, allowing it to slide past composition with the rest of the conditional. The second ingredient was the application of $\mu_{\circledast}$ to the conditional meaning that resulted, which squashed a higher-order proposition of type $\circledast(\circledast(t))$ into an ordinary proposition of type $\circledast(t)$. Crucially,

[18] '$\circledast(\overset{\uparrow}{\circledast})$' denotes internal lift, i.e., $\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}$.

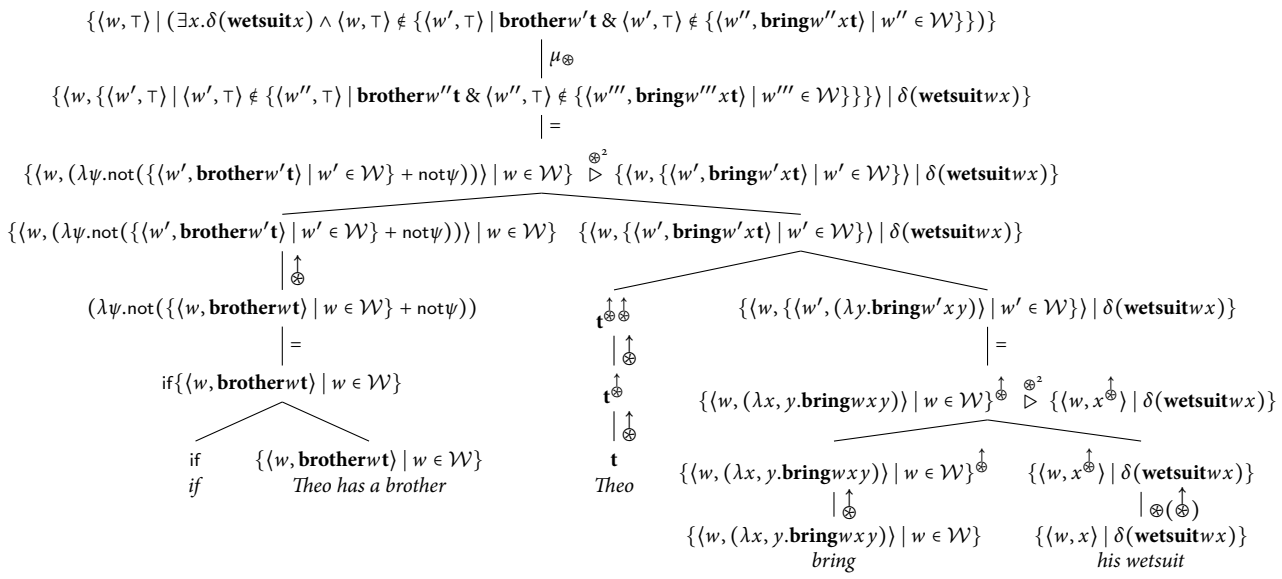

Figure 8: Deriving (1)

applying $\mu_{\circledast}$ has the effect that the presupposition of the outer set takes precedence over either any at-issue content or presuppositions contributed by any inner sets. In general, higher-order propositions are encoded by functions of type $s \to (s \to t \to t_{\#}) \to t_{\#}$; in order to check which objects $\langle w, x\rangle$ are elements of the set of type $s \to t \to t_{\#}$ that results from applying $\mu_{\circledast}$, objects $\langle w, n\rangle$ (for n of type $s \to t \to t_{\alpha}$) must be checked for membership in the higher-order proposition; it is at this step (so to speak) that the presupposition which had previously scoped outward takes effect. In general, this is our solution to the problem of trapped presupposition triggers. Let us now see how we may avoid the usual satisfaction-theoretic pitfall for propositional attitude verbs.

3.2 Propositional attitude verbs

Recall the problem for presupposition triggers embedded in the complements of propositional attitude verbs: their presuppositions always end up relativized to the accessibility relation associated with the verb, whereas we'd often like them to simply escape the propositional attitude context altogether. (8) illustrated this problem.

(8) Mary believes the dog is a cat.

The applicative grammar presented in §2.2 led to a meaning for (8) on which it presupposes that Mary believes that there is a dog. We can now see how the monadic grammar just presented allows us to attribute to this sentence the correct, belief-free presupposition.

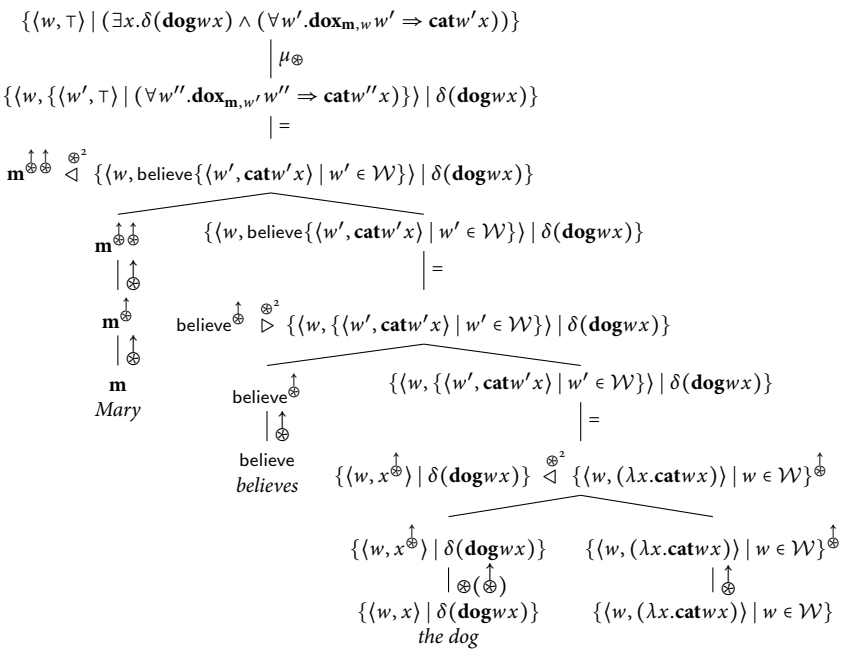

Figure 9: Deriving (8)

A new derivation for this sentence is given in Figure 9. Note how internal lift is invoked in a way analogous to that of Figure 8: the meaning of the presupposition trigger is internally lifted so that its presupposition floats to an outer effectual layer, which allows it to slide past composition with *believes*. Once the full higher-order proposition is composed, $\mu_{\circledast}$ is applied, fixing the scope of the floated presupposition trigger. Once its scope is fixed, moreover, we obtain an ordinary, $\circledast(t)$-type proposition, in which the presupposition from the embedded clause appears untouched. We may therefore see that membership of $\langle w, \top \rangle$ (for some w) in the resultant proposition is defined just in case the following formula is true:

$$(\exists x.\mathbf{dog}wx)$$

At the end of the day, Figure 9 ascribes to (8) the presupposition that there is some dog, along with the truth condition that Mary believes that that dog is a cat.

3.3 Applicative functors, monads, and scope-taking

In summary, we went from a grammar that invoked $\circledast$ as an applicative functor (equiping us with $(\cdot)^{\uparrow\circledast}$ and $(\cdot)^{\circledast\downarrow}$) to a grammar that invokes $\circledast$ as a monad (additionally, giving us $\mu_{\circledast}$). The applicative grammar suffered from the proviso problem, and more generally,

the problem of trapped presupposition triggers, because it lacked the tools to generate propositions (meanings of type $\circledast(t)$) whereof the semantic contribution of presupposition triggers is unaffected by the presence of filters that outscope them. What this general problem illustrates is that the semantic contribution of a presupposition trigger should be made *flexible*: its presupposition may be affected by a filter, and in fact, filtration is the very mechanism by which presuppositions become satisfied in the satisfaction account; however, presupposition triggers often give rise to readings in which their presuppositions appear unaffected. The latter kind of situation is illustrated by the often unconditional presuppositions of conditional sentences and the non-intensional presuppositions often associated with intensional verbs. The argument of the foregoing discussion has been that these presuppositions are generated by a kind of presuppositional scope-taking, and that $\mu_{\circledast}$ was the ingredient necessary to accomplish it. In this guise, the satisfaction theory is, perhaps, somewhat reminiscent of the alternative views of presupposition projection developed on top of Kamp's DRT, insofar it gives rise to a syntactically flexible projection mechanism (van der Sandt, 1989; van der Sandt and Geurts, 1991; van der Sandt, 1992).

The reader may still wonder exactly what role $\mu_{\circledast}$ plays in the account of presuppositional scope-taking. After all, is it not a purely applicative notion—that of internal lift—which frees a presupposition trigger in the first place, allowing it to take scope? In other words, the recipe for internal lift requires only applicative ingredients: $\mathsf{map}_{\circledast}$ and $(\cdot)^{\overset{\uparrow}{\circledast}}$, while all $\mu_{\circledast}$ does is lower higher-order propositions into ordinary propositions, thus fixing the scope of presupposition triggers which have floated free, reining them back in again.

Perhaps unintuitively, the scope-taking mechanism provided by the monadic grammar illustrated above does not map onto a single compositional operation; rather, the work of scope-taking is distributed across the actions of both $\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}$ and $\mu_{\circledast}$. This point may be more easily seen by considering $\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}$ and $\mu_{\circledast}$ to be the "what" and the "where" of scope-taking, respectively; $\mathsf{map}_{\circledast}(\cdot)^{\overset{\uparrow}{\circledast}}$ determines *that* a given presupposition trigger takes scope (i.e., by freeing up its presupposition), while $\mu_{\circledast}$ determines *where* it takes scope (i.e., by lowering it back down again).

This distinction is, perhaps, made more convincing by noting an alternative to the mechanism for scope invoked in our monadic grammar—one pursued in Charlow 2019a and Charlow 2019b. Charlow gives an extensional treatment of the exceptional scope-taking capabilities of indefinites, but within a monadic framework in which ordinary values of type α are upgraded to monadic meanings of type $i \to \alpha \to t$, where i is the type of assignment functions; that is, assignment-relative sets of α's. However, rather than present monads M in terms of $(\cdot)^{\overset{\uparrow}{\mathsf{M}}}$, $(\cdot)^{\overset{\mathsf{M}}{\downarrow}}$, and μ_{M}, as I have, he gives an equivalent presentation of them in terms of $(\cdot)^{\overset{\uparrow}{\mathsf{M}}}$, along with a genuine scope-taking operator: $(\cdot)^{\gg=_{\mathsf{M}}}$ (called 'bind'). Given a particular monad M, this operator has the following type signature and definition in terms of our more familiar monadic operators $(\cdot)^{\overset{\uparrow}{\mathsf{M}}}$, $(\cdot)^{\overset{\mathsf{M}}{\downarrow}}$, and

μ_{M}:

$$(\cdot)^{\gg=_{\mathsf{M}}} : \mathsf{M}(\alpha) \to (\alpha \to \mathsf{M}(\beta)) \to \mathsf{M}(\beta)$$
$$m^{\gg=_{\mathsf{M}}} = (\lambda k.\mu_{\mathsf{M}}(\mathsf{map}_{\mathsf{M}} k)m)$$

Thus given a presupposition trigger, say, *the dog*, applying this operator to its $\circledast(e)$-type meaning gives us a new meaning $\{\langle w, x\rangle \mid \delta(\mathbf{dog}wx)\}^{\gg=_{\circledast}}$ of type $(e \to \circledast(\beta)) \to \circledast(\beta)$. If we evaluate this new meaning, we get:

$$\begin{aligned}\{\langle w, x\rangle \mid \delta(\mathbf{dog}wx)\}^{\gg=_{\circledast}} &= (\lambda k.\mu_{\circledast}(\mathsf{map}_{\circledast} k\{\langle w, x\rangle \mid \delta(\mathbf{dog}wx)\})) \\ &= (\lambda k.\mu_{\circledast}\{\langle w, kx\rangle \mid \delta(\mathbf{dog}wx)\})\end{aligned}$$

As such, the presupposition trigger *the dog* may take syntactic scope over a context of type $e \to \circledast(\beta)$ (which may be generated by a rule like Predicate Abstraction (Heim and Kratzer, 1998)), in order to give back something of type $\circledast(\beta)$. If we identify β with t, the presupposition trigger takes scope over and returns propositions. In that case, the semantic type of *the dog* is $(e \to \circledast(t)) \to \circledast(t)$, so that nominal presupposition triggers become the present framework's anolog of quantifiers! Figure 10 gives a Charlow-style alternative to the derivation in Figure 9—one featuring literal syntactic scope-taking, along with Predicate Abstraction.[19] The meaning ultimately delivered is identical.

Note the two labels in the derivation tree highlighted in blue: these correspond to the $(\cdot)^{\gg=_{\circledast}}$ shift, which renders the presupposition trigger a syntactic scope-taker, and the $(\cdot)^{\overset{\uparrow}{\circledast}}$ shift, which lifts its trace position into the monad. In comparing this derivation to the derivation of Figure 9, it is convenient to think of the $(\cdot)^{\overset{\uparrow}{\circledast}}$ shift of Figure 10 as corresponding to one half of Figure 9's internal lift (the "lift" part), and the $(\cdot)^{\gg=_{\circledast}}$ of Figure 10 as corresponding to Figure 9's $\mu_{\circledast}$, composed with the other half of its internal lift (the "internal" part $\mathsf{map}_{\circledast}$). Thus the Charlow-style alternative derivation of Figure 10 brings out the scopal nature of presupposition triggers within the original monadic grammar by foregrounding it. By making scope-taking manifest, it reveals how presupposition triggers take scope latently within the $\langle(\cdot)^{\overset{\uparrow}{\circledast}}, (\cdot)^{\overset{\circledast}{\downarrow}}, \mu_{\circledast}\rangle$ grammar; i.e., by drawing on all three operators: in Figure 9, the $(\cdot)^{\overset{\uparrow}{\circledast}}$ of the internal lift and the $\mu_{\circledast}$ at the top of the derivation tree work together to demarcate the semantic context over which the presupposition trigger takes scope, while the $\mathsf{map}_{\circledast}$ of the internal lift works to *feed* this semantic context to the presupposition trigger as its semantic argument. In the alternative derivation, the scope of the presupposition trigger is similarly demarcated by $(\cdot)^{\overset{\uparrow}{\circledast}}$ and $\mu_{\circledast}$; it's just that the scope is represented as a syntactic constituent and the feeding is accomplished by $\mathsf{map}_{\circledast}$ (in the definition of $(\cdot)^{\gg=_{\circledast}}$) and functional application.

[19] The reader may find it odd that covert syntactic movement of *the dog* is not clause-bounded, as covert movement is thought to be. To allow indefinites to take exceptional scope without violating clause-boundedness, Charlow invokes a kind of roll-up pied-piping, in which the mover moves to its minimal clause's edge, after which the clause itself moves to *its* smallest-containing-clause's edge, etc., until the desired scope-position is reached. This has the effect of percolating up indefinite side effects without letting through bona fide quantifiers, as desired. Such a clause-bounded mechanism is dropped here for perspicuity.

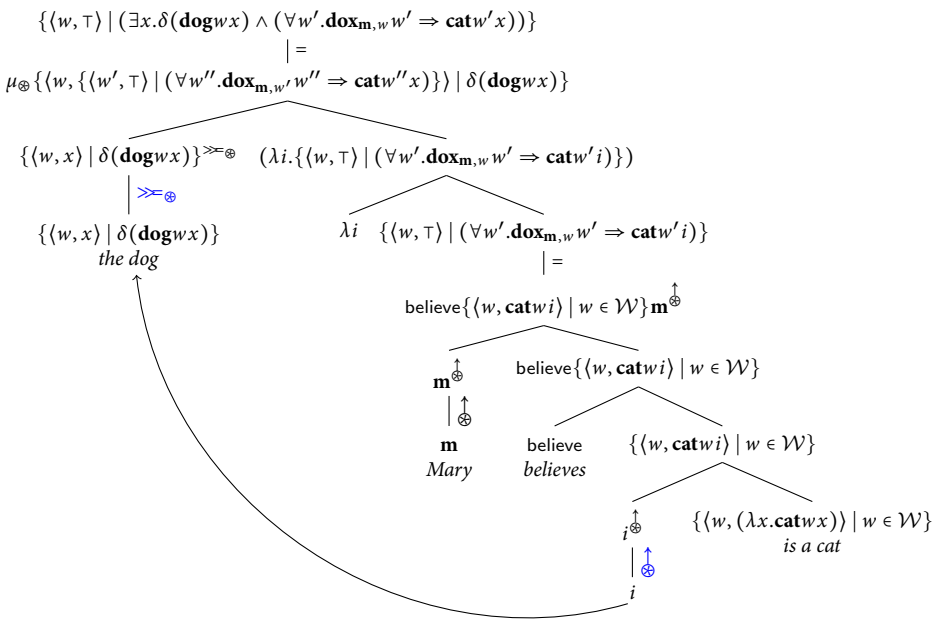

Figure 10: Deriving (8), Charlow-style

4 Some extensions

The discussion above has been aimed at establishing a preliminary framework for analyzing presuppositional phenomena, in particular the scopal behavior of presupposition triggers. The rest of this paper aims to add some more advanced features to this framework. First, we add a mechanism for binding and coreference, in order to give a more adequate analysis of discourses like (6). This mechanism relies on a monad transformer and is essentially based on the strategy used in Charlow 2014 for analyzing discourse referents. Finally, I will briefly illustrate a strategy for analyzing presupposition accommodation and explore some of the analyses it makes available for definites and indefinites.

4.1 Binding and coreference

Recall the discourse in (6), which featured a definite description that was anaphoric to an indefinite.

(6) A dolphin swam. The dolphin was fast.

While the analysis of §2.2 was able to describe the presupposition projection behavior of (6), we currently lack a mechanism to explain its binding behavior; i.e., the fact that

the swimming dolphin and the fast dolphin are undestood to be the same dolphin. To describe this behavior, we need a notion of 'discourse referent' and a mechanism for keeping track of the discourse referents that expressions introduce so that they may serve as antecedents for later anaphora. To accomplish this, I will follow in the footsteps of Charlow (2014) by introducing a State monad transformer (Liang et al., 1995). A State monad transformer s takes a monad M and returns a new monad s(M) which keeps track of some aspect of the state of the discourse. Given a type σ, the type of the state, the state monad transformer s has the following definition:

$$\begin{aligned}
\mathsf{s}(\mathsf{M})(\alpha) &= \sigma \to \mathsf{M}(\langle \alpha, \sigma \rangle) \\
a^{\overset{\uparrow}{\mathsf{s}(\mathsf{M})}} &= (\lambda s.\langle a, s\rangle^{\overset{\uparrow}{\mathsf{M}}}) \\
u^{\overset{\mathsf{s}(\mathsf{M})}{\downarrow}} &= (\lambda v, s.\mu_{\mathsf{M}}(\mathsf{map}_{\mathsf{M}}(\lambda\langle f, s'\rangle.\mathsf{map}_{\mathsf{M}}(\lambda\langle x, s''\rangle.\langle f x, s''\rangle)(v s'))(u s))) \\
\mu_{\mathsf{s}(\mathsf{M})} m &= (\lambda s.\mu_{\mathsf{M}}(\mathsf{map}_{\mathsf{M}}(\lambda\langle n, s'\rangle.n s')(m s)))
\end{aligned}$$

Given an underlying monad M, its State-transformed variant yields functions which read in an input state s (of type σ) and give back values in M consisting of pairs of ordinary values and an updated state. $(\cdot)^{\overset{\uparrow}{\mathsf{s}(\mathsf{M})}}$, when applied to a value, returns a function which takes a state and simply returns that value paired with that state, then lifted in M. $(\cdot)^{\overset{\mathsf{s}(\mathsf{M})}{\downarrow}}$, when applied to a stateful computation returning a function, sequences that computation with another stateful computation returning an argument by feeding the state which is output by the first computation as the input to the second computation. Meanwhile, the resulting value is the function applied to the argument. Finally, $\mu_{\mathsf{s}(\mathsf{M})}$ takes a stateful computation which is two layers deep (i.e., which itself returns a stateful computation) and collapses the layers by feeding the state which is output by the outer computation to the inner computation and then collecting up the resulting value-state pair.

In our application of this monad transformer, we will choose our state type σ to be the type of *lists* of functions from worlds to individuals, a type we call 'γ'.[20] Sensitivity to worlds will be an important feature of our analysis of anaphora to antecedents within the scope of an intensional predicate, such as a propositional attitude verb. Something of type γ might be a list like the following, which has three elements:

$$[(\lambda w.\textbf{president}w), (\lambda w.\textbf{tallest}w), (\lambda w.\textbf{f})]$$

The first element is the function from worlds to whoever the president (say, of France) is in that world, the second element is the function from worlds to whoever the tallest person in that world is, and the third is the constant function which throws out its world and returns Flippy the dolphin. In general, lists may be of any length (including zero). Given the definition of s above, the type $\mathsf{s}(\circledast)(\alpha)$ is $\gamma \to s \to \langle \alpha, \gamma \rangle \to t_{\#}$. In fact, this type is isomorphic to the *implicational* type $\gamma \to s \to \alpha \to \gamma \to t_{\#}$; in order to keep the

[20] Following the notation of de Groote (2006).

type theory simple, I will present $\mathrm{s}(\circledast)$ in terms of the latter type, instead.[21]

$$\begin{aligned}
\mathrm{s}(\circledast) &= \gamma \to s \to \alpha \to \gamma \to t_{\#} \\
a^{\overset{\uparrow}{\mathrm{s}(\circledast)}} &= (\lambda c.\{\langle w, a, c\rangle \mid w \in \mathcal{W}\}) \\
u^{\overset{\mathrm{s}(\circledast)}{\downarrow}} &= (\lambda v, c.\{\langle w, fx, c''\rangle \mid (\exists c'.\langle w, f, c'\rangle \in uc \wedge \langle w, x, c''\rangle \in vc')\}) \\
\mu_{\mathrm{s}(\circledast)} m &= (\lambda c.\{\langle w, x, c''\rangle \mid (\exists c', n.\langle w, n, c'\rangle \in mc \wedge \langle w, x, c''\rangle \in nc')\})
\end{aligned}$$

The new monad $\mathrm{s}(\circledast)$ describes meanings which are able to read in a *context* (i.e., a list of intensional individuals) and return a set of triples of worlds, values, and updated contexts. In this setting, $(\cdot)^{\overset{\uparrow}{\mathrm{s}(\circledast)}}$ takes a value onto the function which reads in a context and returns the set of triples consisting of a possible world, that value, and the original context. $(\cdot)^{\overset{\mathrm{s}(\circledast)}{\downarrow}}$ sequences a stateful, intensional, non-deterministic function with a stateful, intensional, non-deterministic argument by reading in a context to feed to the former and collecting up the triples which result, then feeding the new contexts generated to the latter and collecting up the triples which result, finally, non-deterministically generating all the triples that result from applying the functions of the first set of triples to the arguments of the second set (meanwhile preserving the contexts generated by the second set). Finally, $\mu_{\mathrm{s}(\circledast)}$ functions in a way reminiscent of $\mu_{\circledast}$, with the caveat that it must manage contexts: it does so by feeding the contexts generated by the outer set to the inner sets in order to produce the sets of triples from which it draws its ultimate values and contexts.

Let us analyze (6) by way of illustration. First, let us assign the following new denotation of type $(e \to t) \to \mathrm{s}(\circledast)(e)$ to the indefinite article, which allows it to update its incoming context ($(::)$ appends a new function of type $s \to e$ to the head of a list):

$$(\lambda P, c.\{\langle w, fw, f :: c\rangle \mid P(fw)\})$$

Thus, given the property denoted by its noun phrase complement, an indefinite reads in a context and returns a set of triples consisting of a possible world, a function which, given that world, returns an individual satisfying the property, and the input context updated with that function. The derivation of the first sentence of (6) is shown in Figure 11.

The derived meaning is a *stateful proposition*: a function from input contexts to sets of triples of worlds, truth values, and output contexts. Given an input context c, this function returns a set which contains the triple $\langle w, \top, f :: c\rangle$ iff fw is a dolphin in w who swam in w; meanwhile, the same set contains the triple $\langle w, \bot, f :: c\rangle$ iff fw is a dolphin in w who didn't swim in w. If there is no dolphin in w, then no triple in the returned set features w. More generally, we may assess the truth of a stateful proposition ϕ at a world w by checking whether or not there is a context c such that the following holds (where ε is the empty context, i.e., of length zero):

$$\langle w, \top, c\rangle \in \phi\varepsilon$$

[21] To generalize this typing scheme, s may be thought of as a partial map, defined on monads whose value type is not the result type.

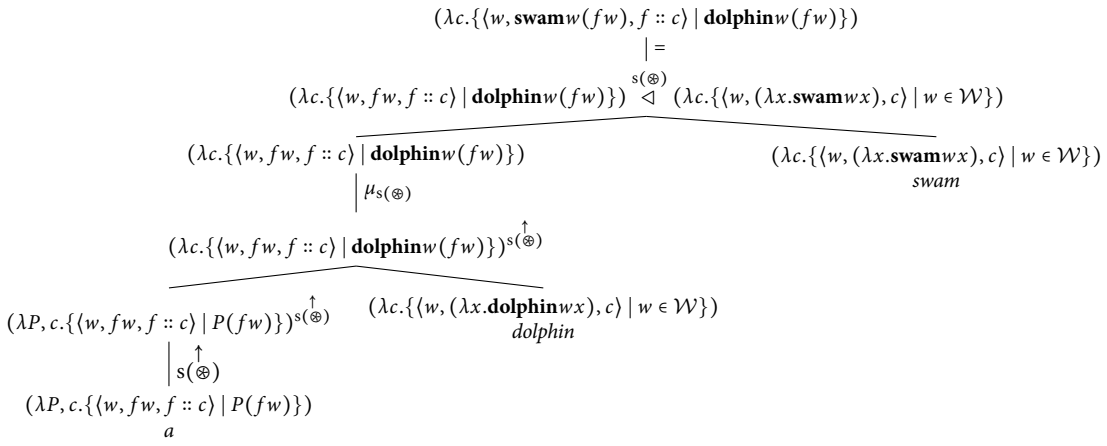

Figure 11: Deriving *a dolphin swam*

If such a triple can be found, then we take ϕ to be true at w, and false otherwise.

Let us now look again at the second sentence of (6). To analyze the definite determiner, we may follow the analysis of pronouns of de Groote (2006) by invoking a *selection function* **sel** whose type is $\gamma \to s \to e$; it accesses a context to return a function from worlds to individuals. To the determiner, we may assign the following denotation (which, like that of the indefinite article, is of type $(e \to t) \to \mathrm{s}(\circledast)(e)$):

$$(\lambda P, c.\{\langle w, \mathbf{sel}cw, c\rangle \mid \delta(P(\mathbf{sel}cw))\})$$

Thus, after composing with its complement, a definite description accesses its input context and returns a set. If the input context is c, any given triple $\langle w, \mathbf{sel}cw, c\rangle$ is a member of this set iff $\mathbf{sel}cw$ satisfies the property denoted by the complement. Hence, while the new definite determiner meaning gives rise to an existence presupposition, it also gives rise to an anaphoric presupposition: that its referent has been established by prior context. Using this denotation, the meaning of the second sentence of (6) is derived in Figure 12.

Finally, we would like a new, stateful notion of discourse update, in order to derive a meaning for (6) itself. Discourse update for our current stateful setting, which we'll write $\ulcorner +^{\mathrm{s}} \urcorner$, may be defined as follows:

$$(+^{\mathrm{s}}) : \mathrm{s}(\circledast)(t) \to \mathrm{s}(\circledast)(t) \to \mathrm{s}(\circledast)(t)$$
$$\phi +^{\mathrm{s}} \psi \coloneqq (\lambda c.\{\langle a, w, c''\rangle \mid (\exists c'.\langle \top, w, c'\rangle \in \phi c \;\&\; \langle a, w, c''\rangle \in \psi c')\})$$

$(+^{\mathrm{s}})$ is just $(+)$, but upgraded with some stateful bookkeeping. Just as $(+)$ does, $(+^{\mathrm{s}})$ acts as a filter: the discourse undergoing update has the effect of removing worlds in which it is false from consideration in the evaluation of the new sentence's presuppositions. Just as before, this is due to the left-to-right asymmetry in the semantics of $\ulcorner \& \urcorner$.

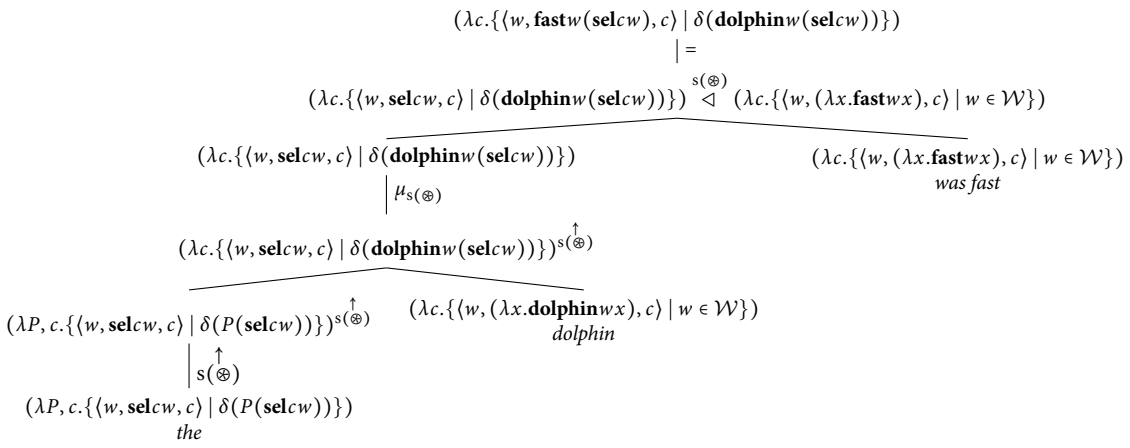

Figure 12: Deriving *the dolphin was fast*

The meaning of the full discourse in (6) is derived in Figure 13. Due to the stateful bookkeeping of $(+^{\mathrm{s}})$, the context output by the first sentence of (6) is given as input to the second sentence. As a result, **sel** is granted access to the intensional individual pushed into the context by the indefinite—the penultimate step of the derivation (highlighted in blue) shows what happens if this intensional individual is, in fact, selected (as on the most salient reading of (6)). The presupposition triggered by the definite description *the dolphin* is also filtered, just as in the derivation of Figure 4. The filtering behavior of this discourse is witnessed by the equivalence between the final two lines in the derivation, in which the final, presuppositional conjunct associated with the definite description is removed. The crucial difference between the derivation of Figure 4 and that of Figure 13 is that, now, the definite description is also bound by its antecedent!

Thus binding of a definite description by an indefinite is accomplished straightforwardly in terms of our original monad $\circledast$, upgraded with State functionality via s. At the same time, all of the presupposition-filtering behavior that we could describe in terms of $\circledast$, we may just as easily describe in terms of $\mathrm{s}(\circledast)$. Ditto for projection behavior: all of the analyzes of presupposition projection that were presented in §3 will have the same shape in the current setting, up to the addition of contexts.

The reader may wonder, at this point, why contexts were chosen to be lists of functions of type $s \to e$, rather than simply lists of *individuals*. Indeed, the intensionality associated with discourse referents in the foregoing analysis appears to play no role. While this is true, intensionality becomes crucial when we describe the behavior of sentences which give rise to discourse referents in the complements of propositional attitude verbs, like *believes*. Consider the following discourse for illustration.

(11) Mary believes a dolphin swam. She believes the dolphin was fast.

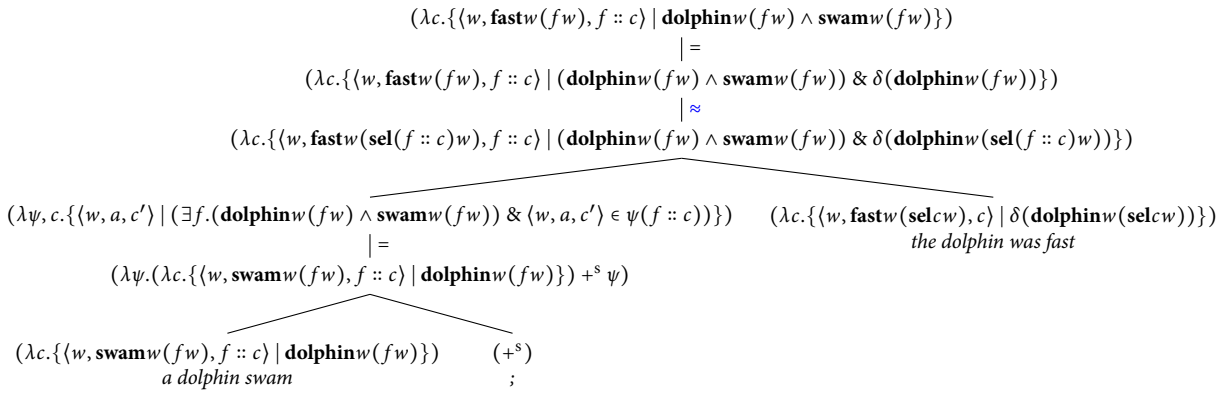

Figure 13: Deriving (6) with State

On one salient reading of this discourse, one infers not that there is a dolphin, but only that Mary believes that there is one. Moreover, on this reading, the second sentence of (11) presupposes only that Mary believes that there is a dolphin, in contrast to (8). To make sense of this discourse, we will define the following new meaning for the verb *believes*:

$$\mathsf{believe} : \mathsf{s}(\circledast)(t) \to e \to \mathsf{s}(\circledast)(t)$$
$$\mathsf{believe}\phi a \coloneqq (\lambda c.\{\langle w, \top, c'\rangle \mid (\forall w'.\mathbf{dox}_{a,w}w' \Rightarrow \langle w', \top, c'\rangle \in \phi c)\})$$

This new denotation is just like the old one, except that it manages contexts by passing the input context to the complement clause and passing up the output contexts that it produces to the matrix clause. We can see the result this produces for the first clause of (11) in Figure 14.

As we can see, the first clause of (11) is true at a world w iff there is some function f from worlds to individuals, such that, at all worlds w' doxastically accessible to Mary at w, f returns a dolphin in w' who swam in w'. For there to be some such function is just for there to be a path from each of Mary's doxastically accessible worlds to some or other swimming dolphin in that world. Indeed, these are exactly the truth conditions of *Mary believes a dolphin swam*. Note that the way we have achieved these truth conditions is by regarding a discourse referent as a kind of "Skolemized" individual. Like Skolemized choice functions, which, in work on indefinites, have been used to confer on them apparent narrow scope with respect to quantifiers, our discourse referents give the illusion of narrow scope by allowing their values to depend on the world of evaluation.

The second sentence of (11) receives an analogous derivation, shown in Figure 15. For simplicity, I analyze the pronoun *she* as denoting Mary, in order to deliver the relevant reading.[22] Crucially, in this analysis, the presupposition trigger *the dolphin* does not take scope over the propositional attitude verb, unlike its counterpart in the embedded

[22] A fully explicit analysis would allow the proper name *Mary* to push a discourse referent into the context;

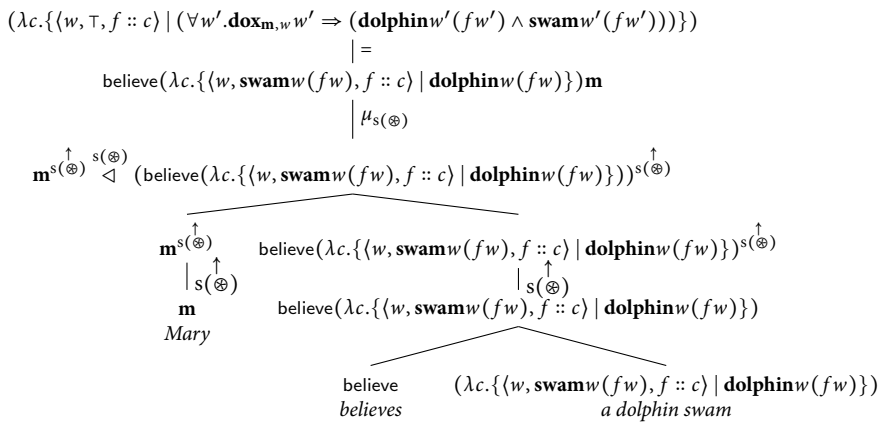

Figure 14: Deriving *Mary believes a dolphin swam*

clause of Figure 9.[23] The scope of the presupposition trigger remains low, so that we may derive an intensionalized presupposition. By inspecting the top of the derivation tree in Figure 15, one can see that, having fed the resulting stateful proposition a context, say c, membership in the returned set of a triple whose first component is w is defined iff the following formula is true:

$$(\forall w'.\mathbf{dox}_{\mathbf{m},w}\, w' \rightarrow \mathbf{dolphin}\, w'(\mathbf{sel}\, cw'))$$

That is, the definedness conditions of the meaning of the second sentence of (11) are (more or less) that some salient discourse referent, when applied to any of Mary's doxastically accessible worlds w', returns something which is a dolphin in w'. The sentence is therefore predicted to presuppose that Mary believes that there is a dolphin.

Given the results of Figure 14 and Figure 15, let us now sequence the entire discourse by using our stateful discourse update operator $(+^{\mathrm{s}})$. A derivation is given in Figure 16. Crucial to delivering the relevant reading of (11) is the penultimate line in the derivation tree, in which the selection function invoked by the definite description *the dolphin* is resolved to the discourse referent introduced by *a dolphin*. Indeed, once this choice is made, the presupposition that the referent of the definite description is a dolpin in each of Mary's belief worlds is filtered, due to the left-to-right semantics assigned to ⌜&⌝, as witnessed by the equivalence between the final two lines of the derivation tree. As a result, our account of (11) correctly predicts it to be presupposition-free. Additionally, given a suitable meaning for *and*, it correctly predicts the entailments in (12).[24]

i.e., it would denote $(\lambda c.\{\langle w, \mathbf{m}, (\lambda w.\mathbf{m}) :: c\rangle \mid w \in \mathcal{W}\})$, while the pronoun, which could select it, would denote $(\lambda c.\{\langle w, \mathbf{sel}\, cw, c\rangle \mid w \in \mathcal{W}\})$.

[23] Of course, the present framework gives rise to another derivation in which the definite description *does* take scope over the propositional attitude verb. Indeed, if both the definite description and its indefinite antecedent take scope above their respective matrix verbs, filtration goes through just the same, and we obtain the reading of (11) on which the indefinite has wide scope relative to the propositional attitude verb.

[24] In particular, we can have *and* denote property conjunction: $(\lambda P, Q, x.Px \;\&\; Qx)$. For more complex

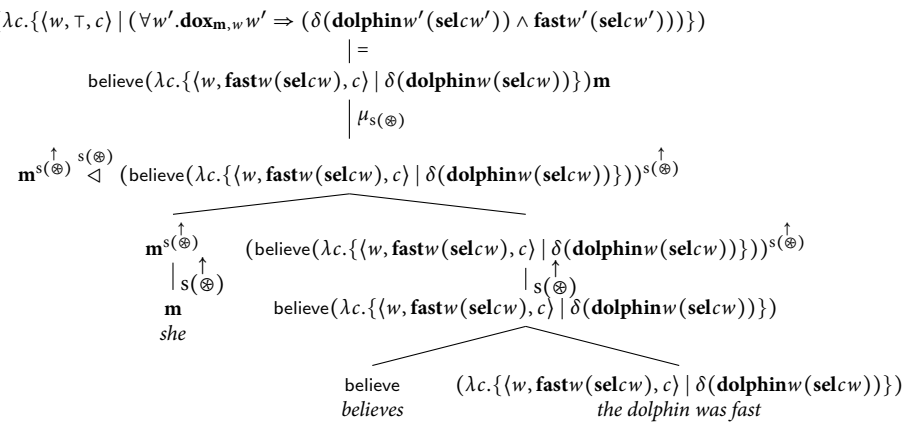

Figure 15: Deriving *she believes the dolphin was fast*

(12) a. Mary believes a dolphin swam. She believes the dolphin was fast.
⇝ Mary believes a dolphin swam and was fast.

b. Mary believes a dolphin swam and was fast.
⇝ Mary believes a dolphin swam. She believes the dolphin was fast.

That is, the meaning assigned to (11) corresponds to the reading on which Mary believes one and the same dolphin to have swum and been fast.

At this point, we may observe the importance of choosing to regard discourse referents as functions of type $s \to e$. When a discourse referent is introduced within the complement of a propositional attitude verb, it appears as if the associated indefinite's scope is narrow relative to the verb, despite the fact that the discourse referent itself is a live candidate for anaphora outside the verbal complement. This appearance is due to the fact that the discourse referent may covary in reference with the world at which it is evaluated, according to the constraints placed on this world by the meaning of the complement clause in which it is introduced. If discourse referents were mere individuals, such covariation would not be permitted.

---

cases, we may want to allow it access to stateful discourse update: $(\lambda P, \mathcal{Q}, x.(Px)^{\mathrm{s}(\uparrow)} +^{\mathrm{s}} \mathcal{Q}^{\ \mathrm{s}(\circledast)}_{\ \downarrow}\ x^{\mathrm{s}(\uparrow)})$ (where it is of type $(e \to t) \to \mathrm{s}(\circledast)(e \to t) \to e \to \mathrm{s}(\circledast)(t)$). For example, *Mary believes that a dolphin swam and that the dolphin was fast.*

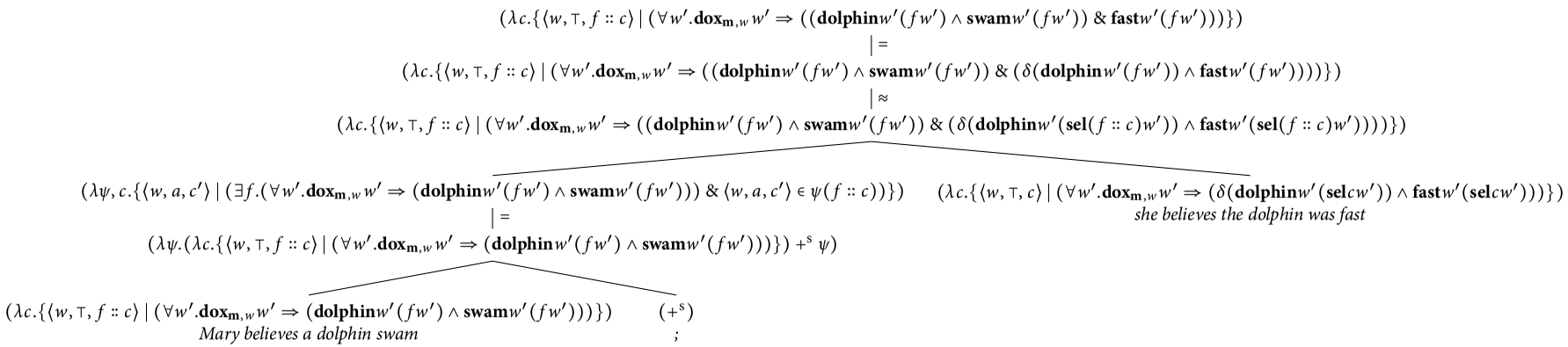

Figure 16: Deriving (12)

4.2 Accommodation

Our way of encoding presuppositions leads to a fairly straightforward strategy for studying *presupposition accommodation* (Lewis, 1979). When a presupposition is accommodated, it acquires a status more like that of at-issue meaning, and, as a result, it is able to interact with operators in a way which would be prohibitted under its presuppositional guise. (13) illustrates this phenomenon.

(13) The dolphin wasn't fast, because there is no dolphin!

On a coherent reading of (13), the existence presupposition normally associated with the definite description does not project; rather it is cancelled. On this reading, the negated auxiliary appears to behave under a metalinguistic guise by targeting a pragmatic inference associated with its prejacent, i.e., that there is a dolphin. Because the presupposition does not project, and yet it interacts with its surrounding context as if it were at-issue, it is said to be *locally* accommodated. One way of analyzing presupposition accommodation, generally, would be to introduce an operator like the following (which I call, suggestively, $\ulcorner\delta^{-1}\urcorner$):[25]

$$\begin{aligned}\delta^{-1} &: t_{\#} \to t\\ \delta^{-1}\top &\coloneqq \top\\ \delta^{-1}\bot &\coloneqq \bot\\ \delta^{-1}\# &\coloneqq \bot\end{aligned}$$

Indeed, δ^{-1} is a left inverse for δ, in that we have $\delta^{-1} \circ \delta = \mathsf{id}$; though it is not a right inverse ($\delta \circ \delta^{-1} \neq \mathsf{id}$), since $\delta(\delta^{-1}\bot) = \#$. δ^{-1} is useful for analyzing accommodation because it allows one to define an operator accom, as follows:

$$\begin{aligned}\mathsf{accom} &: \circledast(\alpha) \to \circledast(\alpha)\\ \mathsf{accom}m &\coloneqq (\lambda w, x.\delta^{-1}(mwx))\end{aligned}$$

Let us take a look at how we might use this operator, in order to accommodate the presupposition of (13). Recall our meaning for *the dolphin* from §2.2:

$$\{\langle w, x\rangle \mid \delta(\mathbf{dolphin}wx)\}$$

This meaning results in an existence presuppostion for the definite description: once composed with the rest of a sentence, membership of $\langle w, \top\rangle$ in the resulting proposition will be defined only if there is a dolphin in w. Now let's see what results when we accommodate this presupposition, using accom:

$$\begin{aligned}\mathsf{accom}\{\langle w, x\rangle \mid \delta(\mathbf{dolphin}wx)\} &= \{\langle w, x\rangle \mid (\delta^{-1} \circ \delta)(\mathbf{dolphin}wx)\}\\ &= \{\langle w, x\rangle \mid \mathbf{dolphin}wx\}\end{aligned}$$

[25] δ^{-1} is equivalent to the A-operator of Beaver and Krahmer (2001).

The result is just the meaning for the corresponding indefinite, *a dolphin*! Indeed, accommodating the definite description in (13) allows for an analysis of its first clause on which it is equivalent to (14), read with the indefinite taking narrow scope.

(14) A dolphin wasn't fast.

Based on this close relationship between indefinites and definites in the current framework, one may wonder whether or not there is a feasible analysis of definite descriptions according to which they arise from indefinites. In other words, me might introduce an operator $\mathsf{presup} : \circledast(\alpha) \to \circledast(\alpha)$ which is just $(\lambda m, w, x.\delta(mwx))$, i.e., a right inverse for accom. Such an operator might mediate between indefinites and definites in languages in which they exhibit identical morphology; alternatively, it might be the denotation of certain morphemes in languages which derive definites from indefinites explicitly. I leave an investigation along these lines for future research.

5 Conclusion

One of the great innovations of the satisfaction account of presupposition projection is that it fits the description of presupposition projection behavior squarely within the program of compositional semantics; presupposition projection is simply an effect—a side effect—of semantic composition. What Geurts showed is that this program may not have been as sure-footed as originally hoped: satisfaction-account analyses of the presuppositions of complex constructions are too deterministic, leading to filtration when it is unwanted. I hope to have shown that this worry is ultimately unfounded. The *classical* satisfaction account is essentially applicative, strapping presupposition triggers down into their local contexts and forcing them to undergo evaluation. The upgraded satisfaction account I have presented is monadic, allowing presupposition triggers to scope freely.

Given the progress made in understanding presupposition within the satisfaction account, it is important to point out that the strategy for upgrading from an applicative to a monad is backwards compatible: analyses couched within a satisfaction-theoretic setting which is applicative may still exist comfortably within a monadic setting, since every monad is also some applicative A, but with the addition of a μ_{A}. Given the promise witnessed above of presuppositional grammars that incorporate a μ, it will be interesting to see what future research shows them to be capable of.

References

Barker, Chris, and Chung-chieh Shan. 2008. Donkey anaphora is in-scope binding. *Semantics and Pragmatics* 1:1–46. URL http://dx.doi.org/10.3765/sp.1.1.

Beaver, David, and Emiel Krahmer. 2001. A partial account of presupposition projection. *Journal of Logic, Language and Information* 10:147–182. URL https://doi.org/10.1023/A:1008371413822.

Beaver, David I. 1999. Presupposition accomodation: A plea for common sense. In *Logic, language, and computation*, ed. Lawrence S. Moss, Jonathan Ginzburg, and Rijke de Maarten, volume 2, 21–44. Stanford: CSLI Publications.

Beaver, David I. 2001. *Presupposition and assertion in dynamic semantics*. Studies in Logic, Language and Information. Stanford: CSLI Publications. URL https://semanticsarchive.net/Archive/jU1MDVmZ.

Bochvar, Dmitry A. 1939. On a three valued calculus and its application to the analysis of contradictories. *Matematicheskii Sbornik* 4:287–308. URL https://doi.org/10.2307/226908.

Charlow, Simon. 2014. On the semantics of exceptional scope. Doctoral Dissertation, NYU, New York. URL https://semanticsarchive.net/Archive/2JmMWRjY.

Charlow, Simon. 2019a. The scope of alternatives: indefinites and islands. *Linguistics and Philosophy* 1–46. URL https://doi.org/10.1007/s10988-019-09278-3.

Charlow, Simon. 2019b. Static and dynamic exceptional scope. URL https://ling.auf.net/lingbuzz/004650, forthcoming in *Journal of Semantics*.

Coppock, Elizabeth, and David Beaver. 2015. Definiteness and determinacy. *Linguistics and Philosophy* 38:377–435. URL https://doi.org/10.1007/s10988-015-9178-8.

Dekker, Paul. 1994. Predicate logic with anaphora. In *Proceedings of the 4th Conference on Semantics and Linguistic Theory*, ed. Mandy Harvey and Lynn Santelmann, 79–95. Ithaca, NY: Cornell. URL http://dx.doi.org/10.3765/salt.v4i0.2459.

von Fintel, Kai. 2008. What is presupposition accommodation, again? *Philosophical Perspectives* 22:137–170. URL https://doi.org/10.1111/j.1520-8583.2008.00144.x.

Francez, Itamar. 2019. How not to project the satisfaction theory of projection (on karttunen) or: Who has a proviso problem? In *okens of meaning: Papers in honor of lauri karttunen*, ed. Cleo Condoravdi and Tracy Holloway King. Stanford: CSLI Publications. URL https://ling.auf.net/lingbuzz/004219.

Geurts, Bart. 1996. Local satisfaction guaranteed: A presupposition theory and its problems. *Linguistics and Philosophy* 19:259–294. URL https://doi.org/10.1007/BF00628201.

Giorgolo, Gianluca, and Ash Asudeh. 2012. $\langle M, \eta, \star \rangle$ monads for conventional implicature. In *Proceedings of Sinn und Bedeutung 16*, ed. Ana Aguilar Guevara, Anna Chernilovskaya, and Rick Nouwen, MITWPL, 265–278. URL http://mitwpl.mit.edu/open/sub16/Giorgolo.pdf.

Giorgolo, Gianluca, and Christina Unger. 2009. Coreference without discourse referents. In *Proceedings of the 19th Meeting of Computational Linguistics in the Netherlands*, ed. Barbara Plank, Erik Tjong Kim Sang, and Tim Van de Cruys, 69–81.

Groenendijk, Jeroen A., and Martin B. J. Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy* 14:39–100. URL https://doi.org/10.1007/BF00628304.

de Groote, Philippe. 2006. Towards a Montagovian account of dynamics. In *Proceedings of the 16th Conference on Semantics and Linguistic Theory*, ed. Masayuki Gibson and Jonathan Howell, 1–16. Ithaca, NY: Cornell. URL https://dx.doi.org/10.3765/salt.v16i0.2952.

Grove, Julian. 2019. Scope-taking and presupposition satisfaction. Doctoral Dissertation, University of Chicago, Chicago. URL https://semanticsarchive.net/Archive/TRmOTkzM.

Heim, Irene. 1982. The semantics of definite and indefinite noun phrases. Doctoral Dissertation, University of Massachussetts, Amherst. URL https://semanticsarchive.net/Archive/Tk0ZmYyY/.

Heim, Irene. 1983. On the projection problem for presuppositions. In *Proceedings of the 2nd West Coast Conference on Formal Linguistics*, ed. Michael D. Barlow, Daniel P. Flickinger, and Nancy Wiegand, 114–125. Stanford: Stanford University Press.

Heim, Irene. 1992. Presupposition projection and the semantics of attitude verbs. *Journal of Semantics* 9:183–221. URL https://doi.org/10.1093/jos/9.3.183.

Heim, Irene, and Angelika Kratzer. 1998. *Semantics in generative grammar*. Malden: Blackwell.

Herzberger, Hans G. 1973. Dimensions of truth. *Journal of Philosophical Logic* 2:535–556. URL https://doi.org/10.1007/BF00262954.

Hintikka, Jaakko. 1969. Semantics for propositional attitudes. In *Philosophical logic*, volume 20 of *Synthese Library (Monographs on Epistemology, Logic, Methodology, Philosophy of Science, Sociology of Science and Knowledge, and on the Mathematical Methods of Social and Behavioral Sciences)*, 21–45. Dordrecht: Springer. URL https://doi.org/10.1007/978-94-010-9614-0_2.

Kamp, Hans. 1981. A theory of truth and semantic representation. In *Formal methods in the study of language*, ed. Jeroen A. Groenendijk, Theo M. V. Janssen, and Martin B. J. Stokhof, number 135 in Mathematical Centre Tracts, 277–322. Amsterdam: Mathematisch Centrum.

Karttunen, Lauri. 1973. Presuppositions of compound sentences. *Linguistic Inquiry* 4:169–193. URL https://www.jstor.org/stable/4177763.

Karttunen, Lauri. 1974. Presuppositions and linguistic context. *Theoretical Linguistics* 1:181–194. URL https://doi.org/10.1515/thli.1974.1.1-3.181.

Kleene, Stephen Cole. 1938. On a notation for ordinal numbers. *Journal of Symbolic Logic* 3:150–155. URL https://doi.org/10.2307/2267778.

Kleene, Stephen Cole. 1952. *Introduction to metamathematics*. Amsterdam: North Holland.

Kobele, Gregory M. 2018. The Cooper storage idiom. *Journal of Logic, Language and Information* 27:95–131. URL https://doi.org/10.1007/s10849-017-9263-1.

Lewis, David. 1979. Scorekeeping in a language game. In *Semantics from different points of view*, ed. Rainer Bäurele and Arnim von Stechow, volume 6 of *Springer Series in Language and Communication*, 172–187. Berlin: Springer. URL https://doi.org/10.1007/978-3-642-67458-7_12.

Liang, Shen, Paul Hudak, and Mark Jones. 1995. Monad transformers and modular interpreters. In *POPL '95 Proceedings of the 22nd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, 333–343. New York. URL https://doi.org/10.1145/199448.199528.

Martin, John N. 1979. Some misconceptions in the critique of semantic presupposition. *Theoretical Linguistics* 6:235–282. URL https://doi.org/10.1515/thli.1979.6.1-3.235.

McBride, Conor, and Ross Paterson. 2008. Applicative programming with effects. *Journal of Functional Programming* 18:1–13. URL https://doi.org/10.1017/S0956796807006326.

Moggi, Eugenio. 1989. Computaional lambda-calculus and monads. In *Proceedings of the 4th Annual Symposiom on Logic in Computer Science*, 14–23. Piscataway: IEEE Press. URL https://doi.org/10.1109/LICS.1989.39155.

Partee, Barbara H. 1986. Noun-phrase interpretation and type-shifting principles. In *Studies in discourse representation theory and the theory of generalized quantifiers*, ed. Jeroen Groenendijk, Dick de Jongh, and Martin Stokhof, 115–143. Dordrecht: Foris.

Peters, Stanley. 1979. A truth-conditional formulation of karttunen's account of presupposition. *Synthese* 40:301–316. URL https://doi.org/10.1007/BF00485682.

Rothschild, Daniel. 2011. Explaining presupposition projection with dynamic semantics. *Semantics and Pragmatics* 4:1–43. URL https://doi.org/10.3765/sp.4.3.

van der Sandt, Rob A. 1989. Presupposition and discourse structure. In *Semantics and contextual expression*, ed. Renate Bartsch, Johan van Benthem, and Peter van Emde Boas, 267–294. Dordrecht: Foris.

van der Sandt, Rob A. 1992. Presupposition projection as anaphora resolution. *Journal of Semantics* 9:133–177. URL https://doi.org/10.1093/jos/9.4.333.

van der Sandt, Rob A., and Bart Geurts. 1991. Presupposition, anaphora, and lexical content. In *Text understanding in lilog*, ed. O. Herzog and C.-R. Rollinger, 259–296. Berlin: Springer. URL https://doi.org/10.1007/3-540-54594-8_65.

Shan, Chung-chieh. 2001. Monads for natural language semantics. In *Proceedings of the ESSLLI 2001 Student Session*, ed. Kristina Striegnitz, 285–298. 13th European Summer School in Logic, Language, and Information. URL https://arxiv.org/abs/cs/0205026.

Stalnaker, Robert. 1973. Presuppositions. *Journal of Philosophical Logic* 2:447–457. URL https://doi.org/10.1007/BF00262951.

Stalnaker, Robert. 1974. Pragmatic presuppositions. In *Semantics and philosophy*, ed. Milton K. Munitz and Peter K. Unger, 197–214. New York: New York University Press. URL https://doi.org/10.1093/0198237073.003.0003.

Stalnaker, Robert. 1978. Assertion. In *Pragmatics*, ed. Peter Cole, volume 9, 315–332. New York: Academic Press.

Wadler, Philip. 1992a. Comprehending monads. In *Mathematical structures in computer science*, volume 2, 461–493. Cambridge University Press. URL https://doi.org/10.1017/S0960129500001560.

Wadler, Philip. 1992b. The essence of functional programming. In *Proceedings of the 19th ACM SIGPLAN-SICACT symposium on Principles of programming languages*, 1–14. URL https://doi.org/10.1145/143165.143169.

Wadler, Philip. 1993. Monads for functional programming. In *Program design calculi*, ed. Manfred Broy, volume 118 of *NATO ASI Series (Series F: Computer and Systems Sciences)*. Berlin, Heidelberg: Springer. URL https://doi.org/10.1007/978-3-662-02880-3_8.

Wadler, Philip. 1994. Monads and composable continuations. *Lisp and Symbolic Computation* 7:39–56. URL https://doi.org/10.1007/BF01019944.

A Applicative functors and monads

An applicative functor (McBride and Paterson, 2008) is a map A on types associated with two operators, $(\cdot)^{\overset{\uparrow}{\lambda}}$ ('unit') and $(\cdot)^{\overset{\Uparrow}{\downarrow}}$ ('apply'), having the following type signatures,

$$(\cdot)^{\overset{\uparrow}{\lambda}} : \alpha \to \mathsf{A}(\alpha)$$
$$(\cdot)^{\overset{\Uparrow}{\downarrow}} : \mathsf{A}(\alpha \to \beta) \to \mathsf{A}(\alpha) \to \mathsf{A}(\beta)$$

and satisfying the laws of Figure 17 ($\ulcorner\circ\urcorner$ abbreviates function composition, i.e., $f \circ g \coloneqq (\lambda x.f(gx))$).

$$\mathsf{id}^{\uparrow_{\mathsf{A}}^{\mathsf{A}}\downarrow} = \mathsf{id} \qquad \text{(Identity)}$$

$$(((\circ)^{\uparrow_{\mathsf{A}}^{\mathsf{A}}\downarrow} u)^{\mathsf{A}\downarrow} v)^{\mathsf{A}\downarrow} = u^{\mathsf{A}\downarrow} \circ v^{\mathsf{A}\downarrow} \qquad \text{(Composition)}$$

$$f^{\uparrow_{\mathsf{A}}^{\mathsf{A}}\downarrow} a^{\uparrow_{\mathsf{A}}} = (f\,a)^{\uparrow_{\mathsf{A}}} \qquad \text{(Homomorphism)}$$

$$(\lambda f.f\,a)^{\uparrow_{\mathsf{A}}^{\mathsf{A}}\downarrow} u = u^{\mathsf{A}\downarrow} a^{\uparrow_{\mathsf{A}}} \qquad \text{(Interchange)}$$

Figure 17: The Applicative Functor Laws

Given an applicative functor A, the operator $\mathsf{map}_{\mathsf{A}} : (\alpha \to \beta) \to \mathsf{A}(\alpha) \to \mathsf{A}(\beta)$, which makes A a *functor*, is given as $(\cdot)^{\mathsf{A}\downarrow} \circ (\cdot)^{\uparrow_{\mathsf{A}}}$.[26]

A monad is an applicative functor M with an additional operator μ_{M} ('join') having the following type signature,

$$\mu_{\mathsf{M}} : \mathsf{M}(\mathsf{M}(\alpha)) \to \mathsf{M}(\alpha)$$

and which satisfies the additional laws of Figure 18.

$$\mu_{\mathsf{M}} \circ (\cdot)^{\uparrow_{\mathsf{M}}} = \mathsf{id} \qquad \text{(Left Identity)}$$

$$\mu_{\mathsf{M}} \circ \mathsf{map}_{\mathsf{M}}(\cdot)^{\uparrow_{\mathsf{M}}} = \mathsf{id} \qquad \text{(Right Identity)}$$

$$\mu_{\mathsf{M}} \circ \mu_{\mathsf{M}} = \mu_{\mathsf{M}} \circ \mathsf{map}_{\mathsf{M}}\mu_{\mathsf{M}} \qquad \text{(Associativity)}$$

$$\mu_{\mathsf{M}} \circ \mathsf{map}_{\mathsf{M}}(\mathsf{map}_{\mathsf{M}} f) = \mathsf{map}_{\mathsf{M}} f \circ \mu_{\mathsf{M}} \qquad \text{(Naturality)}$$

Figure 18: The Monad Laws for Applicative Functors

Thus any monad is an applicative functor, but not the reverse; some applicative functors cannot be made to satisfy the laws of Figure 18.

An equivalent presentation of monads associates them with two, rather than three operators. These are the $(\cdot)^{\uparrow_{\mathsf{M}}}$ of the applicative presentation, along with an operator $(\cdot)^{\gg=_{\mathsf{M}}}$ ('bind') having the following type signature:[27]

$$(\cdot)^{\gg=_{\mathsf{M}}} : \mathsf{M}(\alpha) \to (\alpha \to \mathsf{M}(\beta)) \to \mathsf{M}(\beta)$$

On this presentation, $(\cdot)^{\uparrow_{\mathsf{M}}}$ and $(\cdot)^{\gg=_{\mathsf{M}}}$ are required to satisfy the laws of Figure 19.

The $\langle (\cdot)^{\uparrow_{\mathsf{M}}}, (\cdot)^{\mathsf{M}\downarrow}, \mu_{\mathsf{M}} \rangle$ presentation and the $\langle (\cdot)^{\uparrow_{\mathsf{M}}}, (\cdot)^{\gg=_{\mathsf{M}}} \rangle$ presentation are interdefinable as follows. From the first point of view, $(\cdot)^{\gg=_{\mathsf{M}}}$ may be identified as

$$m^{\gg=_{\mathsf{M}}} = (\lambda k.\mu_{\mathsf{M}}(\mathsf{map}_{\mathsf{M}}\, k\, m))$$

[26] Some functors are not applicative; i.e., they only include the operator $\mathsf{map}_{\mathsf{F}}$, satisfying certain functor laws.

[27] The superscript notation for bind was introduced by Simon Charlow in Charlow 2019a. It is usually written as an infix.

$$a^{\overset{\uparrow}{\mathsf{M}} \gg=_{\mathsf{M}}} k = ka \qquad \text{(Left Identity)}$$

$$m^{\gg=_{\mathsf{M}}} (\lambda x.x^{\overset{\uparrow}{\mathsf{M}}}) = m \qquad \text{(Right Identity)}$$

$$(m^{\gg=_{\mathsf{M}}} n)^{\gg=_{\mathsf{M}}} o = m^{\gg=_{\mathsf{M}}} (\lambda z.(nz)^{\gg=_{\mathsf{M}}} o) \qquad \text{(Associativity)}$$

Figure 19: The Monad Laws

From the second point of view, $(\cdot)^{\overset{\mathsf{M}}{\downarrow}}$ may be identified as

$$u^{\overset{\mathsf{M}}{\downarrow}} = (\lambda v.u^{\gg=_{\mathsf{M}}} (\lambda f.v^{\gg=_{\mathsf{M}}} (\lambda x.(fx)^{\overset{\uparrow}{\mathsf{M}}})))$$

while μ_{M} may be identified as

$$\mu_{\mathsf{M}} m = m^{\gg=_{\mathsf{M}} \mathsf{id}}$$

The presentations equivalent in the sense that if the first presentation satisfies the Applicative Functor Laws of Figure 17 and the Monad Laws of Figure 18, then its reconstruction as the second presentation satisfies the Monad Laws of Figure 19, and vice versa.

B Proof that $\circledast$ is a monad

We now take a look at the monad $\circledast$ from the main text. (The proof that $\odot$ is a monad is obtained by recognizing that it is merely a Reader-transformed Powerset monad.) Recall the definition of $\circledast$ from the main text.

$$\circledast(\alpha) = s \to \alpha \to t_{\#}$$

$$(\cdot)^{\overset{\uparrow}{\circledast}} : \alpha \to \circledast(\alpha)$$
$$a^{\overset{\uparrow}{\circledast}} = (\lambda w, x.\delta(x = a))$$

$$(\cdot)^{\overset{\circledast}{\downarrow}} : \circledast(\alpha \to \beta) \to \circledast(\alpha) \to \circledast(\beta)$$
$$u^{\overset{\circledast}{\downarrow}} = (\lambda v, w, r.(\exists f, x.uwf \wedge vwx \wedge \delta(r = fx)))$$

It is useful to draw out some equivalences. In justifying them, it is important to see that because of the semantics assigned to $\ulcorner\exists\urcorner$, the formulae $\ulcorner(\exists v.\phi \wedge \delta(v = x) \wedge \psi)\urcorner$ and $\ulcorner\phi[x/v] \wedge \psi[x/v]\urcorner$ are equivalent. Note, first, the following equivalence for $\mathsf{map}_{\circledast}$:

$$\begin{aligned}\mathsf{map}_{\circledast} f &= f^{\overset{\uparrow}{\circledast}\overset{\circledast}{\downarrow}} \\ &= (\lambda v, w, r.(\exists g, x.\delta(g = f) \wedge vwx \wedge \delta(r = gx))) \\ &= (\lambda v, w, r.(\exists x.vwx \wedge \delta(r = fx)))\end{aligned}$$

In terms of this, we can show the following equivalence for $(\cdot)^{\gg=_\circledast}$:

$$\begin{aligned} m^{\gg=_\circledast} &= (\lambda k.\mu_\circledast(\mathsf{map}_\circledast\, k\, m)) \\ &= (\lambda k.\mu_\circledast(\lambda w, r.(\exists x.mwx \wedge \delta(r = kx)))) \\ &= (\lambda k, w, y.(\exists r.(\exists x.mwx \wedge \delta(r = kx)) \wedge rwy)) \\ &= (\lambda k, w, y.(\exists x.mwx \wedge kxwy)) \end{aligned}$$

Theorem 1. $\circledast$ is a monad.

Proof.

Left Identity

$$\begin{aligned} \overset{\uparrow}{a^{\circledast}}^{\gg=_\circledast} k &= (\lambda w, y.(\exists x.\delta(x = a) \wedge kxwy)) \\ &= (\lambda w, y.kawy) \\ &= ka \end{aligned}$$

Right Identity

$$\begin{aligned} m^{\gg=_\circledast}(\lambda x.\overset{\uparrow}{x^{\circledast}}) &= (\lambda w, y.(\exists x.mwx \wedge \delta(y = x))) \\ &= (\lambda w, y.mwy) \\ &= m \end{aligned}$$

Associativity

$$\begin{aligned} (m^{\gg=_\circledast} n)^{\gg=_\circledast} o &= (\lambda w, y.(\exists x.(m^{\gg=_\circledast} n)wx \wedge oxwy)) \\ &= (\lambda w, y.(\exists x.(\exists z.mwz \wedge nzwx) \wedge oxwy)) \\ &= (\lambda w, y.(\exists x, z.mwz \wedge nzwx \wedge oxwy)) \\ &= (\lambda w, y.(\exists z.mwz \wedge (\exists x.nzwx \wedge oxwy))) \\ &= m^{\gg=_\circledast}(\lambda z, w, y.(\exists x.nzwx \wedge oxwy)) \\ &= m^{\gg=_\circledast}(\lambda z.(nz)^{\gg=_\circledast} o) \end{aligned}$$

□