Scope-taking and presupposition satisfaction

Julian Grove

September 4, 2019

Contents

List of figures

List of tables

Acknowledgments

Thanks first and foremost to Chris Kennedy, my dissertation advisor, who first suggested that I turn these ten pages of scrambled notes into a dissertation. There is much too much to thank Chris for here beyond that, but I'll mention a few things. Thanks for the hours (upon hours) of discussion on this project over the last couple of years, which helped immensely in guiding my goals, questions, and presentation. Thanks for taking a genuine interest in my unformed intuitions and helping me work them out into something useable. Thanks, not least of all, for teaching me so much about semantics.

Thanks to Itamar Francez, who helped me turn this seemingly abstract enterprise into a semantics dissertation. I have learned much from talking to Itamar about the issues I report on here, both about what it contributes to semantic theory and about how to better explain the basics. Thanks also for your unwavering support over the years.

Thanks to Greg Kobele. I could probably (attempt to) write an entirely separate dissertation about what I have learned from Greg. Thanks for teaching me so much about what linguistics can be and for guiding me in the right direction. Greg's influence is on every page of this thesis.

Thanks to Malte Willer, who joined the committee only at the very end, but who has offered great insight and help with the project. Malte gave great feedback as I presented earlier versions of this work at the Linguistics and Philosophy workshop at the University of Chicago, which has greatly improved my thoughts about and presentation of the material.

Many others have played an important role for me during graduate school and the dissertation writing process. Thanks to Karlos Arregi for the many conversations and good times. Thanks to Jason Merchant for teaching me about ellipsis, and for the wonderful seminars. Thanks to Anastasia Giannakidou for the support and for taking an interest in my work. Thanks to Alan Yu for the support. A very very special thanks to Chris Barker for reading versions of the work presented here, and for offering extremely helpful comments that led to many improvements.

Ming Xiang has been such an important force during my time in Chicago. Ming is partly responsible for bringing me here in the first place as her lab manager. Thank you, Ming, for all of the opportunities you have given me, all of the various ways you have supported me over the years, and for teaching me so much.

Thanks to the many friends who have had this experience with me. To name just a few: Helena Aparicio, Andrea Beltrama, Gallagher Flinn, Katie

Franich, Emily Hanink, Asia Pietraszko, Eszter Rónai, Adam Singerman, and Mike Tabatowski. Special thanks to Matt Teichman, who was the best reading group partner/fellow functional programming fanatic I could ask for. Another thanks to Matt for helping me improve the presentation of chapters one and two. Many thanks to Peet Klecha for the friendship, for the semantics conversations, and for telling me to be cool.

My interest in linguistics started some years ago. Thanks to Kyle Rawlins, from whom I first learned about left downward entailment, and to Geraldine Legendre, from whom I first learned about Principle C. Many thanks to Paul Smolensky, because of whom I even ever wanted to apply to grad school.

Thanks, finally, to my family, and to Ashley.

Abstract

This dissertation argues that presupposition projection and filtering are best understood as arising from certain expressions taking scope over expressions that contain them. According to Barker (2015), an expression takes scope over another containing it "when the larger [(containing)] expression serves as the smaller [(contained)] phrase's semantic argument." Specifically, it is argued that presupposition projection occurs when a presupposition trigger takes scope over the larger phrase containing it, while presupposition filtering occurs when a presupposition trigger fails to take scope over the larger phrase containing it.

Building on the satisfaction account of presupposition put forward by Karttunen (1973) and Heim (1983), this perspective on presupposition is couched within the theory of graded monads, which have recently been proposed, in functional programming languages such as Haskell, as a means to structure the composition of functions and arguments which incur side effects. According to this perspective, presupposition projection is regarded as a side effect to the composition of at-issue meaning. A presupposition trigger is then regarded as taking scope when it incurs side effects.

The graded monadic perspective is applied to a general dynamic semantic account of presupposition and anaphora. Specific topics treated include presupposition projection in conditionals, the complements of propositional attitude verbs, and the scopes of indefinites and distributive quantifiers, like *every*, as well as null complement anaphora and verb phrase ellipsis. The perspective is shown to offer natural accounts that make good linguistic predictions in each case. One particular advantage of the graded monadic perspective is that, despite coinciding with a satisfaction account of presupposition projection, it is unsusceptible to the "proviso problem", which has been thought to present a problem for satisfaction accounts for at least the last two decades. Ultimately, what is offered is a comprehensive, yet simple and modular approach to presupposition in natural language.

Chapter 1

Introduction

I argue in this dissertation that presupposition projection and filtering are best understood as resulting from the way in which certain expressions take scope over expressions that contain them. According to Barker (2015), an expression takes scope over another containing it "when the larger [(containing)] expression serves as the smaller [(contained)] phrase's semantic argument." The proposal that projection and filtering are the effect of scope-taking can thus be boiled down to the following two claims.

> **First claim**: the presuppositions of an expression project to an expression containing it when the former takes the latter as its semantic argument. That is, presupposition projection results from an expression with presuppositions taking scope over some other expression.
>
> **Second claim**: the presuppositions of an expression fail to project to (e.g., they are filtered by) an expression containing it when the former fails to take the latter as its semantic argument. That is, presupposition filtering results from an expression with presuppositions failing to take scope over some other expression (the filter).

These two claims can be succinctly summarized by the super-claim that presupposition projection *just is* scope-taking.

Before launching into a more detailed version of this claim, let us first get clear about the phenomenon that presupposition presents us with, descriptively speaking, as well as how we might go about characterizing this phenomenon, linguistically speaking; then, let's ask why this phenomenon might have anything to do with scope. Take the sentence in (1).

(1) Karlos stopped smoking.

Due to the presence of the aspectual verb *stopped*, (1) presupposes that Karlos smoked. We may thus consider presupposition to be a relation between sentences—a relation which (1) bears to the sentence *Karlos smoked*. The goal of a linguistic account of presupposition is thus to characterize this (apparently infinite) relation. Meanwhile, evidence for the truth of any particular characterization is provided by examples like (1). How do we know that (1) presupposes the sentence *Karlos smoked*? We generally notice that utterances of sentences with presuppositions tend to have a certain pragmatic effect. One who utters (1), for example, takes for granted, as background information, that the sentence *Karlos smoked* is true.[1] Thus a sentence presupposed has the special pragmatic status of an inference which is taken for granted.

Thankfully, empirical linguistics has furnished us with linguistic diagnostics of instances of this relation. Prominent among these diagnostics are family-of-sentence tests (Chierchia and McConnell-Ginet, 1990). To test whether or not one sentence presupposes another, we may negate it, embed it as the antecedent of a conditional, or embed it into a question, in order to observe whether or not the hypothesized presupposition projects.

(2) a. Karlos didn't stop smoking.

b. If Karlos stopped smoking, he'll win the lottery.

c. Did Karlos stop smoking?

Indeed, we still infer, from each of the sentences in (2), that the sentence *Karlos smoked* is true. We may take this result to support our intuition that this sentence is a presupposition of (1). Hence, we may capitalize on one of the characteristic properties of presuppositions—that they project—in order to diagnose them. Another well-known diagnostic for this relation is the "hey, wait a minute" test (Shannon, 1976; von Fintel, 2004). While one may not generally challenge the presuppositions of an interlocutor's utterance through a direct denial (as in B's first answer), one may generally challenge them by prefacing their denial with *hey wait a minute* (as in B's second answer).

[1] Or they take for granted that the interlocutors of the discourse in which (1) is uttered take for granted that *Karlos smoked* is true (Stalnaker, 1998).

(3) A: Karlos stopped smoking.
B: # That's not true—Karlos didn't smoke!
B: Hey, wait a minute! Karlos didn't smoke!

We therefore have further evidence that unnegated version of B's response (*Karlos smoked*) is a presupposition of the sentence A used in their assertion. Both of these diagnostics allow us to home in on this relation, as they provide the data in view of which we may measure the success of a given characterization of it.

As linguists, we are ultimately interested in which grammatical properties of an expression give rise to its presuppositions. It is common for linguistic accounts of presupposition to be guided by the Fregean notion of compositionality, according to which full characterizations of presupposition consist in two parts:

- an account of how presuppositions are grammatically encoded in simple expressions (i.e., presupposition triggers)
- an account of how presuppositions of complex expressions are determined by the meanings of their parts

The second part provides a solution to what is known as the "projection problem" for presupposition (Langendoen and Savin, 1971): to say how presuppositions of complex expressions are determined compositionally is, in part, to say how complex expressions inherit the presuppositions of their parts (or don't). In keeping with most of the semantic and pragmatic literature on presupposition, the main empirical problem of this dissertation is the projection problem. The main thrust of the answer I give is that projection behavior is determined by the way certain expressions (presupposition triggers) take scope (or fail to take scope) over the expressions that contain them.

The literature dealing with the projection problem has generally taken one of two explanatory strategies in addressing it: one strategy (the semantic approach) is to motivate principles of semantic composition in order to characterize presupposition projection, thus regarding it as a grammatical phenomenon; the other one (the pragmatic approach) is to characterize it in terms of extra-grammatical principles which govern linguistic behavior. Pragmatic approaches to the projection problem include, for example, Stalnaker 1972, 1973, 1974, 1998, the cancellation models of Gazdar 1979 and Soames 1979, and the Gricean approaches of Atlas 1976, 1977, 1979, Kempson 1975, Wilson 1975, Atlas and Levinson 1981, and, more recently,

Schlenker 2007, 2008. Recent pragmatic approaches include, for example, the accounts in Schlenker 2009, 2011 based on pragmatic reasoning about a presupposition trigger's possible right contexts (what Schlenker calls "good finals"), and the approach of Lassiter 2012, in which presuppositions are modeled as statements of the inferences which one takes to be probable, given an utterance.

In this dissertation, I will defend a particular semantic approach to characterizing presupposition. The account is compositional in the sense, reminiscent of Montague (1970), that it defines a homomorphic relation between an algebera of syntactic expressions (of English) and semantic ones (of the simply typed λ-calculus). Presupposition projection will thus be regarded as an effect of semantic composition. My approach has salient features in common with certain other semantic accounts. In particular:

- It regards presuppositions as definedness conditions on expression meanings—a defining feature of the accounts of presupposition of Heim 1983 and Heim and Kratzer 1998.

- It models presupposed meaning and at-issue meaning as two aspects of a single meaning which the basic compositional repertoire manages differently. In this way, it is reminiscent of the two-dimensional accounts of presupposition and conventional implicature of Karttunen and Peters (1979) and Potts (2005) (respectively). Approaches to conventional implicature which are similar to these in a way my proposal is reminiscent of include Kubota and Uegaki 2009 and Barker et al. 2010, based on continuations.

- It models semantic definedness by reifying it as a kind of degenerate semantic value representing failure. It is thus reminiscent of multivalent logics and their associated approaches to presupposition (Kleene, 1938; Bochvar, 1939; Kleene, 1959; Herzberger, 1973; Martin, 1979; Beaver, 1999, 2001; Beaver and Krahmer, 2001; Coppock and Beaver, 2015, i.a.).

- It models anaphoric presuppositions syntactically in the metalanguage, and thus also at the level of types. The approach is therefore reminiscent of discourse representation theory based approaches to presupposition (van der Sandt, 1989; van der Sandt and Geurts, 1991; van der Sandt, 1992) and approaches within dependent type semantics (Bekki, 2014, i.a.)

Although it has features in common with a wide variety of semantic approaches to presupposition, the account I will propose can be most fruitfully thought of as a *satisfaction account*, in the terminology of Geurts (1996). Crucially, the account identifies the presupposition of an expression with a statement of the definedness conditions of its meaning, which, within dynamic semantics, corresponds to a context-change potential. Unlike the satisfaction accounts which follow Heim (1982, 1983), however, the one proposed here is not subject to what Geurts calls the "proviso problem", which I will, in turn, claim to be an instance of a more general problem (that of *trapped presupposition triggers*). It is the scopal nature of presupposition triggers, in particular, which allows the account to avoid such problems in the first place.

Let us take a closer look at the notion of scope, and thus how it might figure into a semantic approach to presupposition. Invoking Russell (1905)'s sentence in (4) as an illustration,

(4) The present king of France is bald.

the proposal, simply put, is that the expression *the present king of France* can be expressed as a program written in the simply typed λ-calculus, which, when executed with some argument, satisfies its presupposition to produce the present king of France, and then feeds this value to its argument. If we think of the meaning of the subject as a function f which behaves as described, and if we think of the semantic context as a function g, then the meaning of the sentence as a whole can be expressed as $f\,g$, the result of executing f with g as its argument. From this perspective, an expression's presuppositions project because the expression denotes a program which (a) satisfies its presuppositions when executed, and (b) takes its semantic context as an argument. As a result, (4) has the presupposition that there is a present king of France; the subject noun phrase has taken scope.

As mentioned, such a scope-based perspective on presupposition avoids the proviso problem that normally tarnishes satisfaction accounts. The following pair of sentences illustrates this problem.

(5) a. If Theo is a scuba diver, he will bring his wetsuit.

b. If Theo has a brother, he will bring his wetsuit.

While (5a) presupposes something conditional—that if Theo is a scuba diver, then he has a wetsuit—(5b) has the more in-your-face presupposition that

he has a wetsuit. As Geurts points out, accounts of presupposition projection, like Heim (1983)'s, in which the presuppositions of a sentence are identified with the contexts it may update, predict that both (5a) and (5b) have a conditional presupposition, contrary to native speakers' judgments.

From the scope-based perspective, the issue is easily rectified, even while remaining within an account of presupposition based, like Heim (1983)'s, on the identification of a sentence's presuppositions with the contexts it may update. It is as though the consequent of the conditional of (5b) has decided to ignore the antecedent in order to satisfy its presupposition somewhere outside. In the jargon of this dissertation, the consequent has scoped out.

With this perspective on hand, the following pages are geared toward answering some basic questions about presupposition. First, how are presuppositions semantically encoded, and what is at stake if they are not satisfied (i.e., what does their encoding encode?)? Given an answer to these questions, how does presupposition affect semantic composition: how can a sentence's at-issue content be systematically disentangled from its presupposed content, and how is this disentangling arrived at compositionally in terms of more basic meanings? Finally, why do different linguistic constructions differ in their presuppositional behavior: why do presuppositions project in some environments and get filtered in others?

The answers to these questions I provide will emerge in the context of the general proposal that the compositional scheme most appropriate for deriving meanings with presuppositions has the structure of a graded monad. What this means is that there is a functor that lifts semantic types into domains enriched with presuppositional meanings, along with certain operators that allow for the composition of meanings with presuppositions. Presuppositions are thus encoded in terms of meanings in this enriched domain, while semantic composition may be viewed as composition of at-issue meanings, but which has been lifted via the operators. Finally, the operators associated with a graded monad allow expressions with presuppositions to either have their presuppositions evaluated immediately (in which case, they may be filtered) or to delay evaluation of their presuppositions (in which case, they take scope). It is the last feature which prevents a version of the proviso problem from cropping up in this setting.

It is important to note that my proposal is *not* to take Heim's satisfaction account and combine it with a technical device (e.g., that of scope-taking), in order to allow it to avoid the otherwise inevitable proviso problem. Rather, following Barker and Shan (2014) and Charlow (2014), my proposal is to recognize that a feature *already present* in Heim's account is prof-

itably viewed as invoking *side effects*—i.e., processes which go above and beyond the mere application of functions to their arguments in order to produce values. The feature of note is its attribution of context-dependent definedness conditions to presupposition triggers. In other words, the Heimian approach to presupposition projection already supports a form of semantic composition in the face of side effects. What is added here is a lens through which we may view the satisfaction account as having effectful composition, and thus in terms of which we may make the compositional scheme a little more flexible. I hope to make clear that a Heimian approach to presupposition projection does not (inherently) have a proviso problem any more than an account of quantifiers whereon they have semantic type $\langle\langle e,t\rangle,t\rangle$ has a quantifier-in-subject-position-only problem. In the latter case, we may invoke, e.g., type lifts or Quantifier Raising in order to flexibly deal with such noun phrase meanings. Likewise, when we deal with presuppositional side effects, we may enrich the basic compositional system with mechanisms to describe an array of projection behaviors.

This chapter presents a historical and conceptual backdrop for the claims made in the rest of the dissertation. The historical part aims to set these claims within the context of the satisfaction account of presupposition originating in Karttunen 1974 and later built upon by Heim and others. The conceptual part of the backdrop attempts to isolate exactly which principles invoked by the satisfaction account are responsible for its analytical limitations. Previous authors, including van der Sandt (1992) and Geurts (1996), have responded to the apparent limitations of the satisfaction account tradition by presenting syntactic alternatives that build on Kamp (1981)'s discourse representation theory, thus forgoing the Montagovian approach based on the simply typed λ-calculus. This dissertation aims to show how the Montagovian approach to presupposition (i.e, in which expressions are interpreted homomorphically into either higher-order logic or some model thereof) embodied by the satisfaction account may be enriched in order to overcome the analytical difficulties it has faced in the past. We need only make the satisfaction theoretic notion of "local context" a little more flexible; we need to allow an expression's presuppositional requirements to take scope.

A variant of the traditional Montagovian approach to meaning, dependent type semantics (Bekki, 2014; Bekki and McCready, 2014; Tanaka et al., 2014, 2015; Bekki and Satoh, 2015), has been able to avoid the technical difficulties observed in the satisfaction account by providing meaning representations that carry information about an expression's presuppositions in their types. Presupposition satisfaction can then be viewed as proof search,

so that there may be multiple avenues along which a presupposition is satisfied, as there are often many proofs inhabiting a given type. The approach relies on a dependent type system: Martin-Löf (1984)'s intuitionistic type theory. One of its important features is that expressions are flexibly assigned rich types, unlike in most other Montagovian approaches to dynamic semantics. In fact, both the current approach and dependent type semantics represent the anaphoric requirements of presuppositional expressions at the level of types. But, there are two main differences between dependent type semantics and the approach advanced in this dissertation. The first is that dependent type semantics is a proof-theoretic approach whose underlying type theory is both polymorphic and dependent, while the the approach of this dissertation is model-theoretic and situated squarely within the simply typed λ-calculus. The second is that propositional meanings in dependent type semantics are not Stalnaker's information states, but types. Thus they do not deliver sets of worlds, or *truth conditions*, but types which may be conveniently thought of as sets of proofs. Propositional meanings in the approach advanced in this dissertation, in contrast, are sets of worlds, thus keeping to a more standard assumption within model theoretic semantics in the Montagovian tradition.

One other recent approach to presupposition is that of de Groote and Lebedeva 2010 and Lebedeva 2012, which studies presupposition accommodation within an extension of the simply typed λ-calculus enriched with an exception-handling mechanism. This work builds on the Montagovian presentation of discourse representation theory of de Groote 2006, in which the semantic effects that expressions may have on an evolving discourse are studied using *continuations*. Given a type γ of *contexts* (which may be considered to be, e.g., lists of individuals), expressions are taken to denote functions on their continuations, given a result of type $\gamma \to (\gamma \to t) \to t$. de Groote and Lebedeva (2010) and Lebedeva (2012) add to this approach a mechanism for throwing and handling exceptions, which is proposed to be the mechanism underlying presupposition accommodation. As in the approach of this dissertation, presupposition triggers are there analyzed as registering side effects; unlike this dissertation, the main task of these works is to give an account of presupposition accommodation, rather than to study the dynamics of presupposition projection. Given the progress made by this work in understanding presupposition accommodation, it is worth considering whether or not an approach is possible which integrates the two perspectives. I will briefly note a possible way forward in chapter 6.

1.1 Presupposition in dynamic semantics

1.1.1 Contemporary satisfaction accounts

Contemporary incarnations of the satisfaction account of presupposition have their seed in Karttunen 1974, which identifies a variety of syntactic construction types that differ according to the conditions under which the presuppositions of expressions of those types are satisfied. Karttunen (p. 185) states the following rule for determining the conditions under which the presuppositions of a conditional sentence are satisfied (using the notation 'X ∪ A' to mean X ∪ {A}).

> Context X satisfies-the-presuppositions-of "If A then B" just in case (i) X satisfies-the-presuppositions-of A, and (ii) X ∪ A satisfies-the-presuppositions-of B.

It is defined that a context—regarded as a set of sentences—satisfies a sentence's presuppositions if it entails them. Recall the conditional sentence in (5a).

(5a) If Theo is a scuba diver, he will bring his wetsuit.

If the pronoun *he* is taken to refer to Theo, then the consequent of (5a) has the presupposition that Theo has a wetsuit. The principle stated by Karttunen predicts that a context will satisfy the presupposition of (5a) if augmenting it with the sentence *Theo is a scuba diver* results in a new context that entails *Theo has a wetsuit*. A context thus satisfies the presuppositions of (5a) if it entails *if Theo is a scuba diver, he has a wetsuit*. Such a context may, for example, entail the stronger sentence *Theo has a wetsuit*.

A notable feature of Karttunen's presentation of the satisfaction account is its agnosticism about the presuppositions incurred by any given sentence. As discussed in Francez 2018, Karttunen limits his account of presupposition to describing the constraints that a linguistic context should satisfy in order to satisfy the presuppositions of a given construction type. In general, the constraints that Karttunen describes are meant to determine not an expresssion's presuppositions, but, rather, whether or not its presuppositions are satisfied, given some prior discourse. Such constraints therefore yield a definition of possible prior discourse, given the presuppositions of the atomic sentences that a construction involves. Karttunen does not take the additional position that these constraints are themselves identical to an expression's presuppositions, as Heim does.

There is an obvious loss incurred by an account which remains agnostic about the identity of an expression's presuppositions, however: it fails to actually characterize an expression's presuppositions. The satisfaction account is therefore amended in Heim 1983 in order to provide an account, not only of the conditions presuppositions impose on prior discourse, but of the identity of the presuppositions themselves. Heim accomplishes this by identifying presuppositions with the constraints they impose (p. 399):

> S presupposes p iff all contexts that admit S entail p.

For example, what it means for (5a) to presuppose that if Theo is a scuba diver, he has a wetsuit is for it to be admissible in only the contexts entailing that if Theo is a scuba diver, he has a wetsuit. By identifying a sentence's presuppositions with the contexts in which the sentence is admissible, Heim is able to deliver an account of the presuppositions of complex sentences by stating how they give rise to local contexts for the sentences which they contain. The Heimian rule for interpreting sentences coordinated by *and*, for example, is the following (where $\mathcal{M}$ is a model for interpretation).

$$c + [\![\phi_r \textit{ and } \psi_s]\!]_{\mathcal{M}} = c + [\![\phi_r]\!]_{\mathcal{M}} + [\![\psi_s]\!]_{\mathcal{M}}$$

Idealizing a bit, we take ϕ and ψ to be the at-issue contributions of ϕ_r and ψ_s, respectively, and r and s to be the sentences they presuppose. Given some context c which is updated with the sentence ϕ_r *and* ψ_s, c must satisfy the presupposition r of ϕ_r, while $c + [\![\phi_r]\!]_{\mathcal{M}}$ must satisfy the presupposition s of ψ_s, in order for the update to be successful. Thus, as a result of Heim's identification of a sentence's presuppositions with the constraints it imposes on the contexts it updates, we may determine the presupposition of the complex sentence ϕ_r *and* ψ_s to be $r \wedge \phi \rightarrow s$.

It is common in contemporary satisfaction accounts to encode the presuppositions of atomic sentences as partial functions from possible worlds to truth values, following Heim and Kratzer 1998. Thus the sentence *Karlos stopped smoking* might be assigned the following function from worlds to truth values as its meaning, given some evaluation time t_0 (here described using Heim and Kratzer's λ-notation).

λw_s: $\exists t' < t_0$: Karlos smokes in w at t' . Karlos doesn't smokes in w at t_0

Thus *Karlos stopped smoking* denotes a partial function from worlds to truth values which is defined only at worlds in which Karlos used to smoke, and

which is true at such a world just in case he no longer smokes. The presupposition of the sentence can thus be regarded as a proposition: the set of worlds at which the sentence's meaning, qua function, is defined.

The sentence may be assigned this function as its meaning by means of a procedure which composes the meanings of individual morphemes by functional application. If the meanings of the individual words in the sentence are as follows:

$$\begin{aligned}
[\![\textit{Karlos}]\!]_{\mathcal{M}} &= \text{Karlos} \\
[\![\textit{stopped}]\!]_{\mathcal{M}} &= \lambda P_{\langle s,\langle \tau,et\rangle\rangle}\lambda x_e \lambda w_s \lambda t_\tau : \exists t' < t\text{: } P(w)(t')(x)\ .\ \neg P(w)(t)(x) \\
[\![\textit{smoking}]\!]_{\mathcal{M}} &= \lambda w_s \lambda t_\tau \lambda x_e\ .\ x \text{ smokes in } w \text{ at } t
\end{aligned}$$

we may compose the meanings of the individual words using forward and backward functional application (FA and BA) to arrive at the meaning of the entire sentence.

$$\begin{aligned}
& [\![\textit{Karlos stopped smoking}]\!]_{\mathcal{M}} \\
=\ & \mathsf{BA}([\![\textit{Karlos}]\!]_{\mathcal{M}})(\mathsf{FA}([\![\textit{stopped}]\!]_{\mathcal{M}})([\![\textit{smoking}]\!]_{\mathcal{M}})) \\
=\ & [\![\textit{stopped}]\!]_{\mathcal{M}}([\![\textit{smoking}]\!]_{\mathcal{M}})([\![\textit{Karlos}]\!]_{\mathcal{M}}) \\
=\ & \lambda w_s, t_\tau : \exists t' < t\text{: Karlos smokes in } w \text{ at } t'\ .\ \neg\text{Karlos smokes in } w \text{ at } t
\end{aligned}$$

Choosing t_0 as the relevant evaluation time delivers the meaning presented for the sentence above.

Given a sentence denoting a partial function from worlds to truth values, a satisfaction account must provide a means of updating a discourse—a conversation already in progress—with the sentence. Given a discourse (δ) and a sentence (ϕ), both of which denote functions from worlds to truth values, we may make use of an operator (+) to update the discourse with the sentence.

$$[\![\delta\ ;\ \phi]\!]_{\mathcal{M}} = [\![\delta_r]\!]_{\mathcal{M}} + [\![\phi]\!]_{\mathcal{M}}$$

But the overarching thesis of the satisfaction account of presupposition is that a discourse may only be updated with a particular sentence if the sentence's presuppositions are satisfied by the discourse.[2] Thus the operator (+) ought to be defined in the following way:[3]

[2] This requirement is what von Fintel (2008) calls 'Stalnaker's Bridge', alluding to its discussion in Stalnaker 1973.

[3] An associative definition of (+), which therefore gives it weaker definedness conditions, can be given as:

$$(+) := \lambda\delta_{\langle s,t\rangle}\text{: } \delta \text{ is total } .\ \lambda\phi_{\langle s,t\rangle}\text{: } \phi \text{ is defined at all worlds } w \text{ at which } \delta \text{ is true } .$$

$$\lambda w_s\ .\ \begin{cases} \text{true} & \delta \text{ and } \phi \text{ are true at } w \\ \text{false} & \text{either } \delta \text{ or } \phi \text{ is false at } w \end{cases}$$

Hence, updating δ with ϕ using $(+)$ causes the presuppositions of δ to project, while the presuppositions of ϕ are weakened so that they need only be satisifed at worlds at which δ is true; in other words, the presuppositions of $[\![\delta]\!]_{\mathcal{M}} + [\![\phi]\!]_{\mathcal{M}}$, considered as a proposition, are

$$\lambda w_s\ .\ r(w) \text{ is true and, if } \delta \text{ is true at } w, \text{ then } s(w) \text{ is true}$$

where r and s are $\langle s,t\rangle$-type functions encoding the definedness conditions of δ and ϕ, respectively.

By way of exposition, let's assume the meaning of the sentence *Karlos smoked* is as follows, where t_0 is the evaluation time and t_1 is some other time:

$$\lambda w_s\ .\ t_1 < t_0 \text{ and Karlos smokes at } t_1 \text{ in } w$$

Then the sentence *Karlos smoked and Karlos stopped smoking* receives the following analysis, according to Heim's account:[4]

$$\begin{aligned} & c + [\![\textit{Karlos smoked and Karlos stopped smoking}]\!]_{\mathcal{M}} \\ = \ & c + [\![\textit{Karlos smoked}]\!]_{\mathcal{M}} + [\![\textit{Karlos stopped smoking}]\!]_{\mathcal{M}} \\ = \ & c + \lambda w_s.t_1 < t_0 \text{ and Karlos smokes at } t_1 \text{ in } w \text{ and Karlos doesn't smoke at } t_0 \text{ in } w \end{aligned}$$

The definition of $[\![\phi]\!]_{\mathcal{M}} + [\![\psi]\!]_{\mathcal{M}}$ is such that it is defined at whatever worlds at which both $[\![\phi]\!]_{\mathcal{M}}$ is defined and either $[\![\phi]\!]_{\mathcal{M}}$ is false or $[\![\psi]\!]_{\mathcal{M}}$ is defined. Thus the presuppositions of *Karlos stopped smoking* need only be evaluated at worlds at which *Karlos smoked* is true; that is, at which they are satisfied. As a result, the sentence *Karlos smoked and Karlos stopped smoking* is predicted by Heim's account to always be defined and, hence, presupposition-free.

In the next section, we will briefly observe some general problems for the satisfaction account, as presented in this particular guise.

$$(+) := \lambda\delta_{\langle s,t\rangle}\ .\ \lambda\phi_{\langle s,t\rangle}\ .\ \lambda w_s\text{: } \delta \text{ is defined at } w \text{ and either } \delta \text{ is false at } w \text{ or } \phi \text{ is defined at } w\ .$$

$$\begin{cases} \text{true} & \delta \text{ and } \phi \text{ are true at } w \\ \text{false} & \text{either } \delta \text{ or } \phi \text{ is false } w \end{cases}$$

Both definitions of $(+)$ return the same value in case the first argument is total and second argument is defined at all worlds at which the first argument is true.

[4] See Rothschild 2011 for a categorematic rendering of Heim's proposals.

1.2 Problems for the satisfaction account

The main proposal of this dissertation is that presupposition triggers are best analyzed as taking scope. By this, I do not mean that certain presupposition triggers also happen to be what are normally regarded as scopal expressions, i.e., quantifiers. Rather, I mean that presupposition triggers systematically take their contexts as their semantic argument. In order to motivate this proposal, I here argue that problems for the satisfaction account of presupposition projection in its usual guise crop up specifically because presupposition triggers are *unable* to take scope.

1.2.1 Trapped presupposition triggers

Let us briefly take another look at sentence (5b), examples similar to which Geurts argues demonstrate a general weakness of satisfaction accounts of presupposition projection.

(5b) If Theo has a brother, he will bring his wetsuit.

We would like to be able to analyze this sentence as presupposing that Theo has a wetsuit. That is, we would like its definedness conditions to be the following:

$$\lambda w_s \text{ . Theo has a unique wetsuit in } w$$

The proviso problem is that, on Heim's satisfaction account, the sentence is analyzed as having the weaker presupposition that Theo has a wetsuit if he has a brother. That is, within the satisfaction account of presupposition, (5b) would normally be analyzed has having the following definedness conditions:

$$\lambda w_s \text{ . if Theo has a brother in } w \text{ then he has a unique wetsuit in } w$$

The analysis of the satisfaction account attributes such a presupposition to (5b) because one of the consequences of the account is that an expression's presuppositions are evaluated in that expression's local context. The local context of the presupposition trigger *his wetsuit* in (5b) is determined by its syntactic position in the consequent of a conditional whose antecedent is *Theo has a brother*. Because of the semantics commonly attributed to conditionals under the satisfaction account, i.e.,

$$c + [\![\textit{if } \phi \textit{ then } \psi]\!]_{\mathcal{M}} = c - ((c + [\![\phi]\!]_{\mathcal{M}}) - (c + [\![\phi]\!]_{\mathcal{M}} + [\![\psi]\!]_{\mathcal{M}})),$$

the presupposition trigger *his brother* is evaluated in the local context $c + [\![\textit{Theo has a brother}]\!]_{\mathcal{M}}$; that is, the global context (c) *updated* with the proposition that Theo has a brother. In other words, the presupposition trigger *his brother* is *trapped* in its local context, forced to undergo evaluation there, with the result that its presupposition is filtered. In order to rescue the example, while remaining within a satisfaction account of presupposition projection, it would be useful to have a mechanism to allow the presupposition trigger to be evaluated in its global context, rather than its local one. Such a mechanism would allow the satisfaction account to attribute the correct presuppositions to (5b).

Although the proviso problem is generally discussed with attention to conditional sentences, it is more general (as Geurts notes). Consider the following sentence.

(6) It is not the case that Theo is coming to town and that he is bringing his brother.

Like (5b), (6) has the clear presupposition that Theo has a brother. But the satisfaction account attributes to (6) a different presupposition: that Theo has a brother if he is coming to town. Here is why. If the meaning of *Theo is coming to town is*

$$\lambda w_s \text{ . Theo is coming to town in } w$$

and the meaning of *he is bringing his brother is*

$$\lambda w_s\text{: Theo has a unique brother in } w \text{ . Theo is bringing his unique brother in } w$$

then the effect of (6) itself on an arbitrary context c is given by

$$\begin{aligned} & c + [\![(6)]\!]_{\mathcal{M}} \\ = \; & c - (c + [\![\textit{Theo is coming to town}]\!]_{\mathcal{M}} + [\![\textit{he is bringing his brother}]\!]_{\mathcal{M}} \\ = \; & c - (c + \lambda w_s\text{: if Theo is coming to town in } w \text{ then he has a unique brother in } w \text{ .} \\ & \text{Theo is coming to town in } w \text{ and he is bringing his unique brother in } w) \end{aligned}$$

The result is thus a conditional presupposition, just as is predicted for (5b) above. Likewise, the presupposition trigger *his brother* is required to be evaluated in its local context, which contains the first conjunct of (6), *Theo is coming to town*.

Another problematic case of presupposition triggers which appear to be trapped (and thus evaluated) in their local contexts is provided by examples

similar to those remarked upon by Heim (1992), but for which a presupposition accommodation story is *prima facie* less appealing. The example in (7) is provided by Heim herself (ex. 71, p. 209).

(7) John: I am already in bed.
Mary: My parents think I am also in bed.

Although Heim's satisfaction account predicts that Mary's utterance in (7) presupposes that Mary's parents believe that John is in bed, this presupposition is not observed: the discourse (7) itself is felt to be presupposition-free, while the only inference made about the beliefs of Mary's parents is the at-issue inference that they include that Mary is in bed. Hence, what Mary's utterance in (7) actually appears to presuppose is that John is in bed. Heim suggests that, in order to analyze this example, a *de re* interpretation of the presupposition trigger *also in bed* may be required. The framework presented in this dissertation regards such a suggestion as on the right track by endowing the satisfaction account that Heim presents with the necessary flexibility to follow up on it.

A pervasive feature of satisfaction accounts of presupposition is their reliance on a pragmatic mechanism known as "presupposition accommodation", in order to explain instances of apparent utterance acceptability in cases in which unsatisfied presuppositions would otherwise be predicted to render some sentence or discourse unacceptable (Lewis, 1979; Karttunen, 1974; Heim, 1983, 1992; von Fintel, 2008). Take the following sentence as an illustration.

(8) John's sister is in town.

(8) presupposes that John has a sister. Out-of-the-blue utterances of (8), however, tend not to be conversation-enders due to a break-down in communication leading to unacceptability; rather, if it has not already been explicitly recognized in some discourse that John has a sister, then the proposition that John has a sister is generally accommodated by the interlocutors so that the discourse may continue. If (8) is uttered in a discourse δ in which it hasn't been established that John has a sister, then rather than attempt to understand the discourse as updated with (8) to yield the new, undefined discourse

$$[\![\delta]\!]_{\mathcal{M}} + [\![(8)]\!]_{\mathcal{M}}$$

a hearer of (8) will generally first accommodate the presupposition that John has a sister. The result of accommodation is the new discourse δ *; John has a sister*, which is then updated with (8) to yield

$$[\![\delta]\!]_{\mathcal{M}} + [\![\textit{John has a sister}]\!]_{\mathcal{M}} + [\![(8)]\!]_{\mathcal{M}}$$

Accommodation thus appears to be a necessary mechanism, in order to understand the actual discourses in which people participate. It is common among satisfaction accounts, however, to attempt to also understand what otherwise appear to be ill-behaved patterns of presupposition projection—e.g., those giving rise to the proviso problem—in terms of the pragmatics of presupposition accommodation.

As an example, Beaver (1999, 2001) makes use of this strategy of explanation, in order to provide a response to the proviso problem. Given the satisfaction account's identification of sentence presuppositions with semantic definedness conditions, the sentence (5b)

(5b) If Theo has a brother, he will bring his wetsuit.

semantically presupposes that Theo has a wetsuit if he has a brother. What is likely to be accommodated upon hearing an utterance of (5b) made out of the blue, however, is the stronger proposition that Theo has a wetsuit. According to the account of presupposition accommodation provided by Beaver, this stronger proposition is accommodated because what conversational participants accommodate upon the utterance of a sentence with presuppositions is information about what the speaker is likely to take for granted, given that they uttered the sentence; moreover, it is more likely than not, based on the assumed independence between having a wetsuit and having a brother, that a speaker takes for granted that Theo has a wetsuit if the speaker also takes for granted the weaker proposition that Theo has a wetsuit if he has a brother.[5] Finally, because utterances of (5b) coincide with accommodating the proposition that Theo has a wetsuit, this stronger proposition is what (5b) is felt to presuppose.

Another example of a reliance on presupposition accommodation as a strategy of explanation for otherwise recalcitrant presupposition projection behavior is found in Heim 1992, where it is suggested that accommodation may be responsible for the felt presupposition of a sentence like (9).

[5] An account of presupposition strengthing based on the apparent independence of a conditional's antecedent and the presupposition of its consequent is further fleshed out in van Rooij 2007.

(9) John believes that it stopped raining.

What (9) appears to presuppose is that it rained. In contrast, what the satisfaction account (as presented by Heim) analyzes it as semantically presupposing is that John believes that it rained. As Heim says (pp. 211–212):

> ...we actually accommodate at once *both* the presupposition that it has been raining *and* the presupposition that John believes so. For without the latter, the [context change potential] of [(9)] just wouldn't be defined.

If we accommodate both, according to Heim, then we have an explanation for why (9) appears to presuppose that it rained. Moreover, we might accommodate both, according to Heim, because if it is presupposed in a conversation that John believes that it rained, then this may be because "...it was in fact raining and John was in an appropriate position to find out" (p. 212).

Thus a common strategy for understanding a sentence's presuppositions within satisfaction accounts is to render first a basic semantic analysis of the sentence, in which it is expressed what a sentence's semantic presuppositions are. Once such a semantic analysis has been obtained, presupposition accommodation is recruited as an explanatory strategy, in order to somehow relate the semantic presuppositions to the observed presuppositions, thus providing an understanding of what the sentence's observed presuppositions are. In general, however, an answer to the proviso problem which is based on accommodation must first establish some way of delineating a set of presuppositions any of which may, in principle, be accommodated upon the utterance of a given sentence. Singh (2007) describes the provision of such a set, given a sentence, as the establishment of a hypothesis space of possible accommodations. Once this hypothesis space is established, pragmatic factors may be invoked in order to say how we come down on any particular one of these accommodated presuppositions in a given context. For example, take sentence (5b), again.

(5b) If Theo has a brother, he will bring his wetsuit.

While the semantic presupposition of (9), according to the satisfaction account, is *if Theo has a brother he has a wetsuit*, a hypothesis space of possible accommodations might consist of the meaning of both this sentence, as well as the stronger *Theo has a wetsuit*.[6] Thus a hypothesis space of possible

[6]Such a hypothesis space is argued to be made available in the accounts of Singh 2007, 2008.

accommodations may consist in strengthenings of the semantic presupposition delivered by the satisfaction account. Notably, however a hypothesis space is generated based on a semantic presupposition delivered by the satisfaction account, the procedure to generate it will have to be sophisticated enough to account for the presuppositions of examples like the following.

(10) If Theo is a scuba diver, his brother will bring his wetsuit.

(10) appears to presuppose neither (11a) nor (11b)

(11) a. If Theo is a scuba diver, he has a brother and a wetsuit.

b. Theo has a brother and a wetsuit.

Rather, (10) presupposes (12).

(12) Theo has a brother, and if he is a scuba diver, he has a wetsuit.

An appropriately sophisticated account of accommodation must recognize the presupposition of the consequent (that Theo has both a brother and a wetsuit) as consisting of the conjunction of two other presuppositions—namely, that Theo has a brother and that Theo has a wetsuit—in order to weaken them differentially.

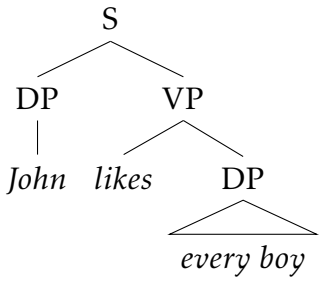
1.2.2 What if presupposition triggers could "QR"?

A metaphor which will prove useful in understanding the main proposals of this dissertation is that of Quantifier Raising (May, 1978; Chomsky, 1981; Heim and Kratzer, 1998). Within the Government and Binding framework of Chomsky 1981, Quantifier Raising is invoked as a syntactic mechanism to transform syntactic surface structures into their associated logical forms, at which the scopes of quantificational noun phrases are fixed.[7] The following example may illustrate.

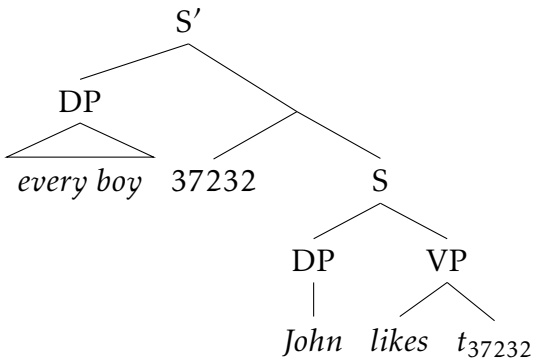
(13) John likes every boy.

Noting facts about constituency in the sentence in (13), one may decide to give it the following syntactic analysis:

[7] QR has, of course, remained an important fixture of Chomsky (1995)'s Minimalist Program.



However, this analysis of the sentence only provides it with a surface structure. In order to yield a sensible interpretation for the quantifier *every boy* and its surrounding syntactic context, Quantifier Raising may be invoked to transform the above phrase marker into the following one:



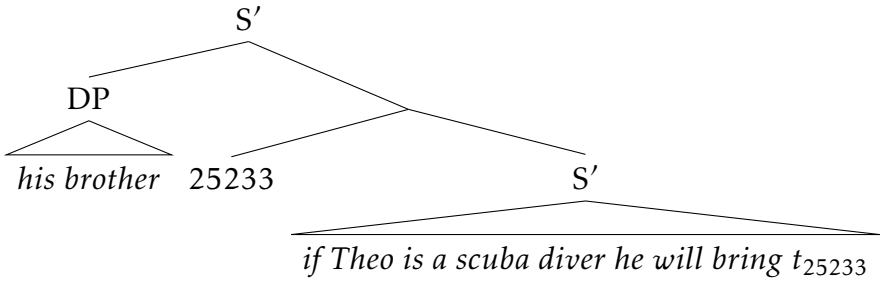
This phrase marker may then be interpreted using the rules of Functional Application and Predicate Abstraction (Heim and Kratzer, 1998). Relevant to the current discussion is that Quantifier Raising is used to give syntactic accounts of quantifier scope ambiguities. Consider the following ambiguous sentence, for example.

(14) Some girl likes every boy.

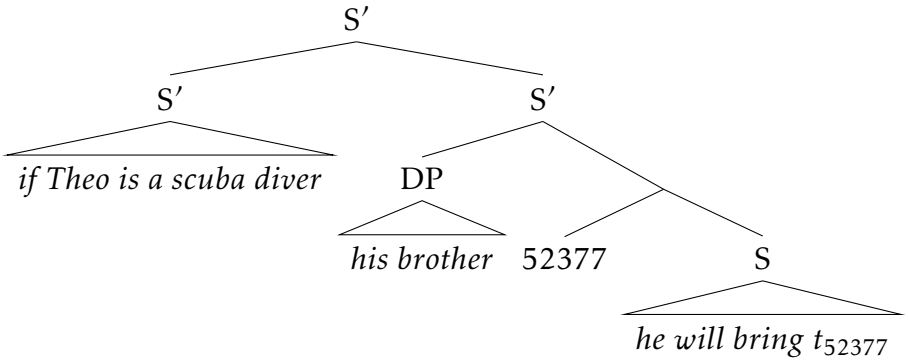
According to its surface-scope interpretation, (14) is true if there is some girl who likes every boy, while according to its inverse-scope interpretation, it is true if every boy is liked by some (possibly different) girl. These two interpretations of (14) may be analyzed as involving either Quantifier Raising of first the object noun phrase, followed by the subject noun phrase (thus yielding surface scope), or Quantifier Raising of first the subject noun phrase, followed by the object noun phrase (thus yielding inverse scope).

Imagine that presupposition triggers were like quantificational noun phrases, in that they were analyzed as taking as their syntactic scope a particular context; e.g., a sentence. The presupposition trigger *his brother* of

(5b) could then undergo something like Quantifier Raising—call it "Presupposition Trigger Raising"—in order to yield a phrase marker like the following one:



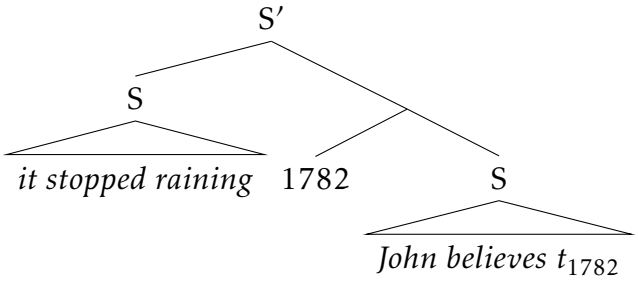
The position to which the presupposition trigger raises, moreover, can be understood as determining the context in which its presupposition is evaluated. For example, the phrase marker analyzing (5b) just given would correspond to the observed reading of (5b), according to which the presupposition that Theo has a brother is evaluated in the global context. Note that the process responsible for delaying the evaluation of the presupposition trigger must therefore be quite different from Quantifier Raising, in a crucial respect: this process is not clause bounded! Meanwhile, the following phrase marker would attribute to (5b) the unobserved, weaker presupposition which results from evaluating the same presupposition in the trigger's local context:



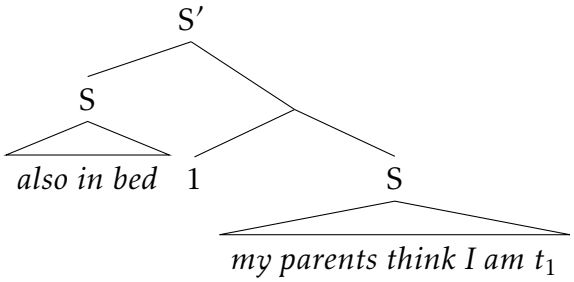
Such an analysis, i.e., according to which the presupposition trigger has raised to a more local position, would be appropriate for the contrasting example (5a), for which the weaker conditional presupposition is, in fact, observed.

What about examples (9) and (7), in which the relevant presupposition trigger is found in an intensional context provided by an individual's be-

liefs? (9), we ought to analyze as follows; that is, by Presupposition Trigger Raising the sentence *it stopped raining* out of the intensional context in which it would otherwise be trapped:

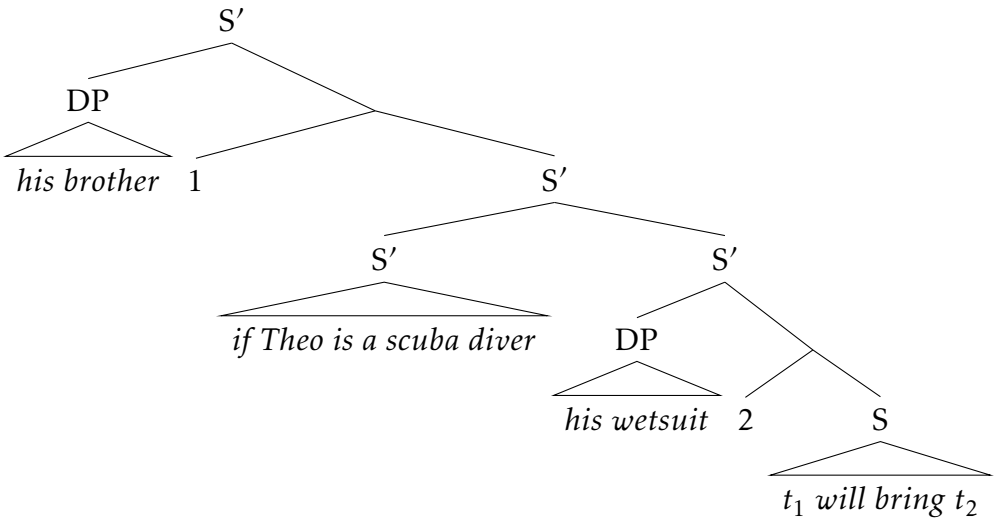


Thus the presupposition that it was raining undergoes evaluation in the global context, rather than the local context provided by John's beliefs. What is reconstructed into the position surrounded by the local context—that of the trace—is mere at-issue meaning: that the event which was, in fact, raining ceased. To (7) we may assign the following phrase marker:



The presupposition-laden *also in bed* has Presupposition Trigger Raised to the global context, so that its presupposition may undergo evaluation there. Meanwhile, its at-issue content—the mere property of being in bed—is reconstructed to the position of the trace.

Finally, how would we derive the appropriate presupposition for (10)? In this case, we ought to give *his brother* scope over the entire conditional, while we give *his wetsuit* scope only over the consequent.



Such a phrase marker corresponds to the attested reading of (10), according to which it presupposes both that Theo has a brother and that if he is a diver, he has a wetsuit.

1.3 What is to come

The analogy with Quantifier Raising is meant to be kept in the back of one's mind while encountering the main proposals of this dissertation. Rather than literally undergo syntactic movement to the contexts in which they are evaluated, however, presupposition triggers will do something else: incur *side effects*. Prior to making a contribution to the at-issue meaning of some sentence, a presupposition trigger may need to engage in some ancillary process; it may need to satisfy its presuppositions. The exact context in which a presupposition trigger's side effects are evaluated, moreover, coincides with where that presupposition trigger can be understood as taking scope. Side effects will be encoded as part of the meaning of the presupposition trigger, but they will be managed as something in the background to semantic composition. Importantly, the particular compositional repertoire used to manage side effects will come in the form of a *graded monad* (Katsumata, 2014; Orchard et al., 2014; Fujii et al., 2016). This choice follows the work of Shan (2001) and, more recently, Charlow (2014), who analyze linguistic side effects in terms of monads. The result of choosing to analyze the side effects incurred by presupposition triggers in terms of a graded monad is that their behavior is much like the behavior we might expect if they literally underwent Presupposition Trigger Raising. The graded

monadic combinatorial scheme I adopt will allow presupposition triggers to choose the contexts at which they register their presuppositions; these will be the contexts within which they undergo evaluation and thus where their side effects are visible.

An important point of clarification is in order before introducing the main proposals. This dissertation does not offer a *solution* to the proviso problem; rather, in the approach I argue for, the problem just does not arise. Conditional sentences whose consequents contain a presupposition trigger give rise to a scope ambiguity. Either the presupposition trigger takes narrow scope with respect to the conditional (thus generating a weak, conditional presupposition) or it takes wide scope (thus generating a strong, unconditional presupposition). The proposal thus renders the account somewhat similar to the discourse representation theory account of van der Sandt (1992), insofar as trigger-containing expressions can be ambiguous. The main contribution of this dissertation is therefore to show that the satisfaction theory does not suffer from the proviso problem *inherently*—once its compositional repertoire is updated, it is capable of delivering predictions quite similar to those of the discourse representation theory account.[8]

A proviso is also in order. Although the proviso problem does not arise for the particular version of the satisfaction theory that I present, this dissertation still leaves open why we observe sentences with semantically ambiguous presuppositions to have the particular presuppositions that they do. Thus while I argue that there are two semantically available readings of (5b)

(5b) If Theo is a scuba diver, he will bring his brother.

I do not say why the stronger presupposition is preferred. Rather, this dissertation may be seen has contributing to our understanding of the hypothesis space of possible accommodations. The role of pragmatic competence in relating semantic presuppositions to observed presuppositions can therefore be understood as to assist in the resolution of scope ambiguity.

1.3.1 Static semantics and presupposition projection

The next chapter lays the basic groundwork for the rest of the dissertation. Here, a static semantics for presupposition projection is introduced, from which it will become clear in what sense presupposition projection is to be regarded as scope-taking. This static semantics is merely meant to illustrate

[8] As has been argued (Zeevat, 1992).

how presupposition can be encoded as a graded monadic side effect, as well as how semantic composition can proceed, given a graded monad.

1.3.2 Dynamic semantics, presupposition satisfaction, and filtering

The following chapters—chapter 3 and chapter 4—enrich the basic proposals of chapter 2, in order to provide mechanisms for presupposition satisfaction and filtering. These chapters apply the framework to standard dynamic semantic phenomena, including indefiniteness, anaphora, and discourse update, in addition to more intricate phenomena, like intensionality and modal subordination. In chapter 5, it is shown how the basic framework may be further leveraged to provide analyses of null complement anaphora, verb phrase ellipsis, and the semantics of distributive quantifiers.

1.3.3 Why the monadic approach?

Ultimately, the picture of presupposition that we end up with is a modular one, due to our choice to regard it as having the structure of a graded monad. The graded monadic approach will allow us to view at-issue meaning as *basic* in an important sense: from the graded monadic perspective, at-issue meaning is synonymous with values, while presupposition is synonymous with side effects, i.e., computations. As stated at the beginning of this chapter, a meaning with presuppositions will be regarded as a program written in the simply typed λ-calculus. We can now be a little bit more explicit: the computation performed when such a program is executed coincides with the side effects it registers; that is, its presupposition. The value returned by the program once it has run is its at-issue meaning. Moreover, computations and ordinary semantic composition of values can be *interleaved*. To say that a presupposition trigger has taken scope outside of a particular semantic context is just to say that its associated program has been executed *prior* to composing with the context.

But not only is there modularity in the separation of values from side effects in the meanings of expressions with presuppositions; there is also modularity in the structure of the side effects themselves. In the next chapter, we introduce a compositional scheme for presuppositional expressions in a static semantics, but in the remaining chapters, we update the scheme to countenance dynamic phenomena. The relevant updates coincide with the actions of various *graded monad transformers*. Given a graded monad, one may often transform it into a new graded monad which retains all of

the functionality of the former, but which gains some new functionality. Thus when dynamic side effects are introduced, they are introduced on top of the static side effects already present. The modularity distinguishing values from side effects, in addition to the modularity structuring the side effects themsleves, allows one to construct models of different grammatical phenomena which, to an extent, live independently of one another.

There is a certain methodological benefit gotten, in part, from the modularity of the approach. It will be possible for the reader to learn from, build upon, and even endorse some of the proposals I put forward without buying them all wholesale. In other words, there can be degrees of (dis)agreement with the approach I take in this dissertation. Here are the basic claims, organized in a way that reflects this structure:

- Presupposition projection is best construed as the effect of presupposition triggers taking scope.
- The best formalization of presuppositional scope-taking is in terms of the operators of a (graded) monad.
- The best choice of (graded) monad for the formalization of presuppositional scope-taking is the Heimian, satisfaction-theoretic one defined in chapter 2 and chapter 3.
- The values produced by semantic composition are context-change potentials, defined in terms of discourse referents, using a framework reminiscent of the compositional discourse representation theory of Muskens (1996).

If my primary goal is achieved, the following pages will convince the reader of the plausibility of the first and second points, as well as that there is at least a nice payoff for throwing in with the third. The analyses presented in the following chapters are stated in the context of some basic assumptions about the representation of truth conditions, context-change potentials, and the semantic contribution of indefinite noun phrases. Given this basic setup (which, in principle, is flexible), the meat of the proposal is that adding context-sensitivity and definedness, along with a graded monadic approach to semantic composition, allows one to describe the attested patterns of presupposition projection with a very tight fit.

Chapter 2

Presupposition as semantic definedness

2.1 Introduction

This chapter sets things up for the rest of the dissertation. The main goal right now is just to have a clean, systematic way of representing an expression's presuppositions. Then, once the basic scaffolding is erected, there will be further ways of decorating it. In light of the work of Shan (2001), and, more recently, Charlow (2014), I will argue that this scaffolding is best construed as having a particular shape—that of a monad.[1] A monad provides the structure needed to handle two aspects of an expression's meaning in a flexible, systematic way. We wish to deal with theories of both an expression's at-issue content and its presupposed content, but we don't want these two theories to muddy or get in the way of each other.

Once monads are introduced, however, one might wonder if other similar abstractions might have worked just as well. Indeed, functional programming is becoming rife with abstractions for handling what one might call "doing two things at once", and it is thus worth addressing whether or not one of these other abstractions could have worked for presupposition. I will argue that what we really need for our purposes is a monad. Presupposition is subject to the generalization that sometimes at-issue content determines presupposed content (think of the meaning of the noun phrase complement of a definite determiner deciding what is registered by

[1] Other authors have also used monads in the analysis of various semantic phenomena, including, e.g., Giorgolo and Unger (anaphora) and Giorgolo and Asudeh (conventional implicature).

the determiner's existence presupposition). Monads, in contrast to, say, applicative functors, are suited to this particular type of boundary-crossing.

Finally, it will turn out that a monad is not quite enough to capture all the flexibility that presuppositional meanings exhibit. Something a bit more expressive will be needed; in particular, a graded monad. A graded monad generalizes an ordinary monad with an extra parameter which will determine the types of the objects presupposed. Presupposition, after all, does not present itself as a uniform phenomenon. But, as we will see, a graded monad can be used to expose its underlying uniformity.

2.2 Monads in linguistic semantics

Monads first became influential in linguistic semantics in Shan 2001, in which it was recognized that many of the obstacles to composing programs faced by functional programmers share features with certain obstacles that pertain to the compositional semantics of natural language.[2] For the programmer, some of these obstacles are provided by the need to read from and update the state of the environment, handle errors and exceptions, and abort a computation provided a certain condition is met. For the semanticist, such obstacles are provided by, for example, the presence of assignment functions, intensionality, and quantification. In each of the phenomena mentioned, there is an apparent tension between two goals. One goal is to be able to call a function whenever one wants, applying it to its argument and producing a value. A second goal is to interact with the environment (broadly speaking) in which the function is called in some designated way.

Let's take intensionality as an illustration of this tension. In the case of intensionality, the enviroment is defined by a state consisting of the world of evaluation, and one relevant goal is to have values be sensitive to the identity of this world. If we consider only this goal, we would assign expressions like *sleep* and *dogs* meanings like

$$\textbf{sleep} : s \to e \to t$$
$$\textbf{dogs} : s \to e$$

respectively. Both meanings, which we represent as λ-terms, encode a world-dependency as their first argument. Having encoded it, it is however no

[2] For the introduction of monads into functional programming, see the series of papers Wadler 1992a,b, 1994, 1995. Monads, a construction from category theory, were first introduced to computer scientists in Moggi 1989 as a way of studying computational side effects.

longer easy to apply the meaning of *sleep* to the meaning of *dogs*—the world arguments are in the way! A monad can help in this situation. Given a set $\mathcal{T}$ of types, a monad is a function $\mathbf{M} : \mathcal{T} \to \mathcal{T}$, together with three operators, $(\cdot)^{\overset{\uparrow}{\mathbf{M}}}$ (pronounced 'pure', 'unit', or 'return'), $(\cdot)^{\overset{\mathbf{M}}{\downarrow}}$ (pronounced 'apply'), and $\mu_{\mathbf{M}}$ (pronounced 'join'), which, at each pair of types α and β, are assigned the following type signatures.[3]

$$(\cdot)^{\overset{\uparrow}{\mathbf{M}}} : \alpha \to \mathbf{M}\alpha \qquad \text{('pure')}$$
$$(\cdot)^{\overset{\mathbf{M}}{\downarrow}} : \mathbf{M}(\alpha \to \beta) \to \mathbf{M}\alpha \to \mathbf{M}\beta \qquad \text{('apply')}$$
$$\mu_{\mathbf{M}} : \mathbf{M}(\mathbf{M}\alpha) \to \mathbf{M}\alpha \qquad \text{('join')}$$

Together, these components of a monad provide a kind of casing for ordinary values, letting them be composed with one another while maintaining some amount of distance from their side effects. Shan (2001) shows how the Reader monad, in particular, can be used to handle intensionality as a side effect. The Reader monad (**R**) is given by the following definition.

Definition 1 (**R**)**.**

$$\begin{aligned}
\mathbf{R}\alpha &= r \to \alpha \\
a^{\overset{\uparrow}{\mathbf{R}}} &= (\lambda s^r.a) \\
m^{\overset{R}{\downarrow}} &= (\lambda n^{r\to\alpha}, s^r.ms(ns)) \\
\mu_{\mathbf{R}}\, m &= (\lambda s^r.mss)
\end{aligned}$$

The function **R**, in this case, contributes a state dependency to any non-monadic type, while $(\cdot)^{\overset{\uparrow}{\mathbf{R}}}$, $(\cdot)^{\overset{\mathbf{R}}{\downarrow}}$, and $\mu_{\mathbf{R}}$ somehow affect the way in which an object deals with state dependencies. Specifically, $(\cdot)^{\overset{\uparrow}{\mathbf{R}}}$ injects a value into the Reader monad by giving it a trivial state dependency; $(\cdot)^{\overset{\mathbf{R}}{\downarrow}}$ allows a function trapped underneath a state dependency to expose its first argument, thus making it a function from state-sensitive arguments to state-sensitive values; and $\mu_{\mathbf{R}}$ squashes two state dependencies into a single state depen-

[3] In this dissertation, I work with the presentation of monads that brings out their structure as applicative functors (McBride and Paterson, 2008) and use the notation introduced in Kobele 2018.

dency.[4] Given our meanings above for *sleep* and *dogs*, we can get the world argument out of the way by evaluating the following term, for which we've identified our state type r with the type of possible worlds.

$$\mathbf{sleep}^{\overset{\mathbf{R}}{\downarrow}}\mathbf{dogs} = (\lambda w^{s}.\mathbf{sleep}\,w(\mathbf{dogs}\,w))$$

That is, we expose the first argument of the meaning of *sleep* with $(\cdot)^{\overset{\mathbf{R}}{\downarrow}}$ and immediately apply the result to the meaning of *dogs*.

Why couldn't we have just done what semanticists normally do by using world-sensitive functional application?

$$(\lambda f^{s\to\alpha\to\beta}, x^{s\to\alpha}, w^{s}.f\,w(x\,w))$$

In other words, a monad was not necessary to tell us what to do: all the monad has provided is a new name ('$(\cdot)^{\overset{\mathbf{R}}{\downarrow}}$') for something. But we will use monads, for two reasons. The first reason is that allowing ourselves access to the basic operators defined by a monad lets us recast the more common operations (like world-sensitive functional application) monadically. As a result, other monads, for which the operators are defined differently, can be understood as involving the same basic operation, but while managing different side effects. Recasting operations monadically thus allows us to view them as basic operations on values.

The second reason is that, once we have decided to encode certain compositional operations in terms of the operators associated with a monad, we have allowed ourselves a way of studying different semantic phenomena independently of each other. If we're committed to handling intensionality in this way, the question, "What features should intensionality have?" becomes "What features should this monad have—that is, how do we define $\mathbf{M}$, $(\cdot)^{\overset{\mathbf{M}}{\uparrow}}$, $(\cdot)^{\overset{\mathbf{M}}{\downarrow}}$, and $\mu_{\mathbf{M}}$?" We then may have a completely separate theory of phenomena that might have nothing to do with intensionality; like binding, for example. This dissertation is about accomplishing, at least partially, a goal of this form. The goal thus stated is to study presupposition independently of other aspects of semantics, like at-issue meaning.

In order to be useful for its intended purpose—that of separating side effects from the values with which they coincide—a monad must satisfy

[4] Thus $(\cdot)^{\overset{\mathbf{R}}{\uparrow}}$ has an effect equivalent to that of the $(\cdot)^{\wedge}$ ('up') operator of Montague's intensional logic. In the same vein, $m^{\overset{\mathbf{R}}{\downarrow}}$ may be expressed as $(\lambda n.(m^{\vee}n^{\vee})^{\wedge})$, while $\mu_{\mathbf{R}}\,m$ may be expressed as $m^{\vee\vee\wedge}$, where $(\cdot)^{\vee}$ is Montague's "down" operator.

$$\mathtt{id}^{\uparrow^{\mathbf{M}}\downarrow^{\mathbf{M}}} = \mathtt{id} \qquad \text{(Identity)}$$

$$u^{\mathbf{M}\downarrow} \circ v^{\mathbf{M}\downarrow} = (((\circ)^{\uparrow^{\mathbf{M}}\downarrow^{\mathbf{M}}} u)^{\mathbf{M}\downarrow} v)^{\mathbf{M}\downarrow} \qquad \text{(Composition)}$$

$$(f\,a)^{\uparrow^{\mathbf{M}}} = f^{\uparrow^{\mathbf{M}}\downarrow^{\mathbf{M}}} a^{\uparrow^{\mathbf{M}}} \qquad \text{(Homomorphism)}$$

$$u^{\mathbf{M}\downarrow} a^{\uparrow^{\mathbf{M}}} = (\lambda f.f\,a)^{\uparrow^{\mathbf{M}}\downarrow^{\mathbf{M}}} u \qquad \text{(Interchange)}$$

Figure 2.1: The Applicative Functor Laws

the Monad Laws. Part of this requirement is that the functions $(\cdot)^{\uparrow^{\mathbf{M}}}$ and $(\cdot)^{\mathbf{M}\downarrow}$, along with $\mathbf{M}$, constitute an applicative functor (McBride and Paterson, 2008). The Applicative Functor Laws are shown in Fig. 2.1. These laws ensure that $(\cdot)^{\uparrow^{\mathbf{M}}}$ and $(\cdot)^{\mathbf{M}\downarrow}$ together allow functions to apply to values in a way that sequences and preserves their side effects.

The laws specific to monads are in Fig. 2.2. These laws say that returning an object already in the monad using $(\cdot)^{\uparrow^{\mathbf{M}}}$ and then squashing the extra side effects incurred with the original ones using $\mu_{\mathbf{M}}$ has no effect (Left Identity); that returning a value with side effects already present using $(\cdot)^{\uparrow^{\mathbf{M}}}$ and then squashing them together with those just incurred by $(\cdot)^{\uparrow^{\mathbf{M}}}$ using $\mu_{\mathbf{M}}$ again does nothing (Right Identity); that when squashing together three layers of side effects into a single layer using $\mu_{\mathbf{M}}$, the order of the squashing is irrelevant (Associativity); and, finally, that squashing together the side effects of a computation and then applying some function to the resulting value is no different from first applying the function to the value and then squashing together the side effects (Naturality).[5] The two sets of laws are illustrated for the Reader monad in Figure 2.3 and Figure 2.4, respectively.

Together, these laws ensure that $(\cdot)^{\uparrow^{\mathbf{M}}}$, $(\cdot)^{\mathbf{M}\downarrow}$, and $\mu_{\mathbf{M}}$ can affect a term's type, but neither its side effects nor its value, with the caveat that $\mu_{\mathbf{M}}$ consolidates two layers of side effects into a single layer; hence, the informa-

[5]Naturality gaurantees that $\mu_{\mathbf{M}}$ is a natural transformation in the sense of category theory.

$$\mu_{\mathbf{M}}\, m^{\uparrow_{\mathbf{M}}} = m \qquad \text{(Left Identity)}$$

$$\mu_{\mathbf{M}}((\lambda x.x^{\uparrow_{\mathbf{M}}})^{\uparrow_{\mathbf{M}}\downarrow^{\mathbf{M}}}\, m) = m \qquad \text{(Right Identity)}$$

$$\mu_{\mathbf{M}}(\mu_{\mathbf{M}}\, m) = \mu_{\mathbf{M}}(\mu_{\mathbf{M}}{}^{\uparrow_{\mathbf{M}}\downarrow^{\mathbf{M}}}\, m) \qquad \text{(Associativity)}$$

$$f^{\uparrow_{\mathbf{M}}\downarrow^{\mathbf{M}}}(\mu_{\mathbf{M}}\, m) = \mu_{\mathbf{M}}(f^{\uparrow_{\mathbf{M}}\downarrow_{\mathbf{M}}\uparrow_{\mathbf{M}}\downarrow^{\mathbf{M}}}\, m) \qquad \text{(Naturality)}$$

Figure 2.2: The Monad Laws

tion about how the resulting layer was originally separated is lost. In short, these functions are present to ensure a certain convenience, i.e., that one can compose values while ignoring their side effects. Additionally, because of the squashing capacity of $\mu_{\mathbf{M}}$, one can have side effects depend on values: when extra layers of side effects result, simply squash them together. This extra feature, provided by $\mu_{\mathbf{M}}$, is what moves **M** from an applicative functor into a monad. We will soon see these features play out in the context of presupposition.

2.3 A monad for semantic definedness

We are now in good position to see how the phenomena of presupposition could be treated by regarding them as the side effects of a monad. Strawson (1950)'s view of the presuppositions of definite descriptions was that they were constituted by felicity conditions on utterances. In other words, if the presupposition of a sentence like (15)—that there is a king of France—is not met on some occasion of its utterance, then the utterance is infelicitous.

(15) The king of France is bald.

The intention behind such an utterance might be to update one's interloctor about some state of affairs. Whatever the intention, though, the failure results from the fact that (15) fails to have whatever kind of basic meaning is conventionally associated with sentences. We can honor the intuition that the failure is one of denotation by allowing expressions, some of the time, to not have any meaning at all.

Identity

$$
\begin{aligned}
& \mathtt{id}^{\uparrow^{\mathbf{R}}_{\downarrow}} \\
= \; & (\lambda v, s.(\lambda s'.\mathtt{id})s(vs)) \\
= \; & (\lambda v, s.\mathtt{id}(vs)) \\
= \; & (\lambda v, s.vs) \\
= \; & \mathtt{id}
\end{aligned}
$$

Composition

$$
\begin{aligned}
& f^{\uparrow^{\mathbf{R}}_{\downarrow}} \circ g^{\uparrow^{\mathbf{R}}_{\downarrow}} \\
= \; & (\lambda v, s.fs(vs)) \circ (\lambda v', s.gs(v's)) \\
= \; & (\lambda v', s.(fs \circ gs)(v's)) \\
= \; & (\lambda v', s.(\lambda s'.(\circ))s(fs)(gs)(v's)) \\
= \; & (\lambda v'', s'''.(\lambda v', s''. \\
& \qquad (\lambda v, s.(\lambda s'.(\circ))s(vs))fs''(v's'')) \\
& \qquad\quad gs'''(v''s''')) \\
= \; & (((\circ)^{\uparrow^{\mathbf{R}}_{\downarrow}} f)^{\mathbf{R}}_{\downarrow} g)^{\mathbf{R}}_{\downarrow}
\end{aligned}
$$

Homomorphism

$$
\begin{aligned}
& (fa)^{\uparrow}_{\mathbf{R}} \\
= \; & (\lambda s.fa) \\
= \; & (\lambda u, s'.(\lambda s.f)s'(us'))(\lambda s.a) \\
= \; & f^{\uparrow^{\mathbf{R}}_{\downarrow}} a^{\uparrow}_{\mathbf{R}}
\end{aligned}
$$

Interchange

$$
\begin{aligned}
& f^{\mathbf{R}}_{\downarrow} a^{\uparrow}_{\mathbf{R}} \\
= \; & (\lambda s.fs((\lambda s'.a)s)) \\
= \; & (\lambda s.fsa) \\
= \; & (\lambda s.(\lambda g.ga)(fs)) \\
= \; & (\lambda v, s.(\lambda s'.(\lambda g.ga))s(vs))f \\
= \; & (\lambda g.ga)^{\uparrow^{\mathbf{R}}_{\downarrow}} f
\end{aligned}
$$

Figure 2.3: Illustration of the Applicative Functor Laws

In fact, functional programming languages already come with devices suited to dealing with what, in a programming context, corresponds to computational failure. For instance, let's say we have a list of values—integers, perhaps—and we would like to extract the third value on the list. In the programming language Haskell (in which it is custom to start counting from zero), we would write

```
l !! 2
```

where `l` is our list. But if `l` doesn't have at least three values—which it plausibily doesn't, as lists can be of any length—an error will arise. Hence, while

```
[3, 4, 5] !! 2
```

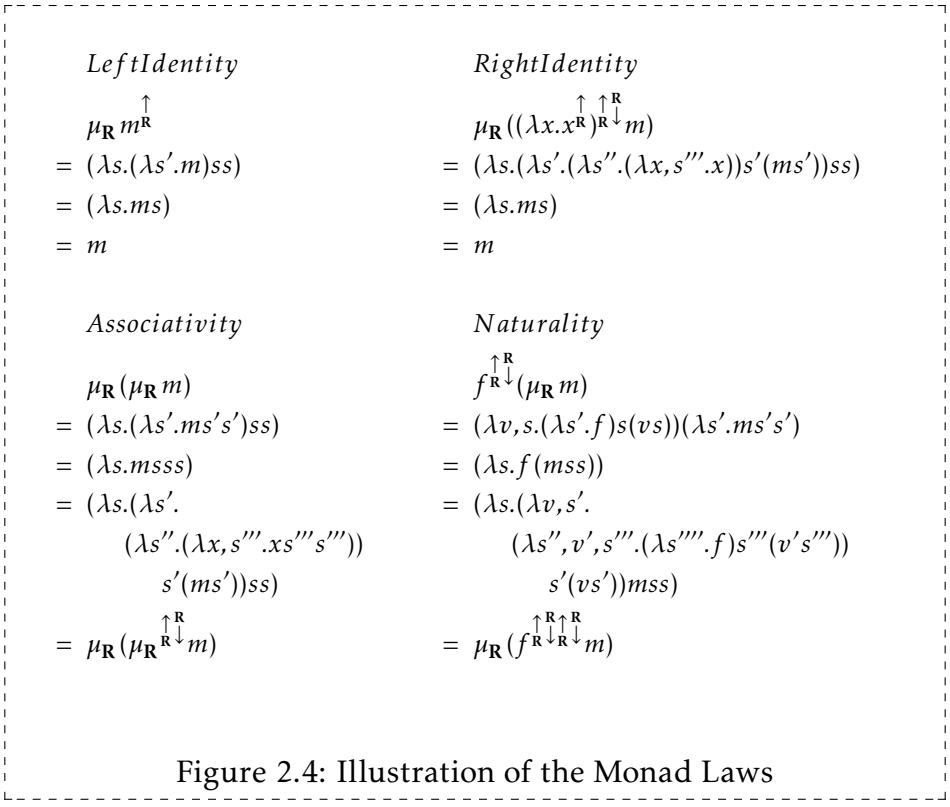


Figure 2.4: Illustration of the Monad Laws

will terminate in `5`,

```
[3, 4] !! 2
```

will fail to terminate at all. We might even say that using the expression `l !! 2` presupposes that `l` has a length of at least three, insofar as its meaning is undefined otherwise. Indeed, we may want to capture presuppositions of this sort in the programming language itself, and a monad provides a means of doing so. In particular, Haskell programmers sometimes use the "Maybe" monad, in order to allow for computations that may or may not terminate. Then, for any reasonable value `a` of type `t`, `Just a`, of type `Maybe t`, means 'the computation has terminated in `a`', while, `Nothing`, also of type `Maybe t`, means 'the computation has failed to terminate'.

Definition 2 (The Maybe monad)**.**

```
pure :: t -> Maybe t
pure a = Just a
```

```
(<*>) :: Maybe (a -> b) -> Maybe a -> Maybe b
Just f  <*> Just a  = Just (f a)
m       <*> Nothing = Nothing
Nothing <*> n       = Nothing

join :: Maybe (Maybe t) -> Maybe t
join (Just m) = m
join Nothing  = Nothing
```

In the above definitions, `pure` symbolizes the $(\cdot)^{\uparrow^{\mathbf{M}}}$ of the Maybe monad, while `(<*>)` symbolizes $(\cdot)^{\mathbf{M}\downarrow}$ and `join` symbolizes $\mu_{\mathbf{M}}$. Above each operator's definition, its type signature is declared. Hence, the first line in the definition means that lifting a value into the Maybe monad is simply wrapping it in a `Just`. The next three lines effectively define `(<*>)`: applying one terminating computation that produces a function to another producing an argument yields a computation that terminates in a value gotten by applying the function to the argument; meanwhile, encountering a `Nothing` (a non-terminating computation) forces the whole computation not to terminate. Finally, `join` simply preserves (non-)termination of the computation to which it applies, either by peeling off an outer `Just`, or by taking `Nothing` onto `Nothing` of a smaller type.

In view of the above example, we can say that the function `(!!)` that takes a list and an integer n to return the $(n+1)^{th}$ element of the list ought to be redefined so that it always terminates. That is, we should define a new function `(//)` that evaluates as follows.

```
[3, 4, 5] // 2 ⇒ Just 5
   [3, 4] // 2 ⇒ Nothing
```

Rather than fail to terminate, the computation terminates in `Nothing`, an object defined only in the Maybe monad.

If we think of semantic composition in these terms, we allow ourselves the possibility of managing semantic undefinedness by reifying it as a semantic object (`Nothing`) representing a failure to denote. If there is no king of France, then *the king of France* in (15) denotes `Nothing`, as does (15) itself. To accomplish this, we can propose the following meaning (`the`) for the definite determiner, which takes as its argument a list of individuals (of type `[Entity]`) to produce a computation that may or may not terminate in a single individual (of type `Maybe Entity`).

```
the :: [Entity] -> Maybe Entity
the [a] = Just a
the _   = Nothing
```

This meaning for *the* encodes the uniqueness and existence presuppositions which are usually attributed to it: if the list it takes as its argument has a single element `a`, it returns `Just a`; otherwise, it returns `Nothing`. Given the following meanings for *king of France* and *is bald*, respectively (where `Bool` is the type of truth values),

```
koF :: [Entity]
bald :: Entity -> Bool
```

we can assign the following meaning to (15) as a whole.

```
(pure bald) <*> (the koF)
```

If `koF` is a list with one element, this computation will terminate in `Just True` or `Just False`, depending on whether `bald` is `True` or `False` of that element. Otherwise, the computation terminates in `Nothing`. In other words, the presupposition projects.

One might wonder what's gained by using something like the Maybe monad over a more traditional approach to semantic undefinedness. Such an approach can be seen in the Heim and Kratzer 1998 treatment of presupposition, wherein expressions with presuppositions are analyzed in terms of partial functions or denotations that are defined only in some models. A Strawsonian denotation for a definite determiner, for example, might be written as in (16), using Heim and Kratzer's λ-notation.

(16) $\lambda P : P \in \mathrm{D}_{\langle \mathrm{e},\, \mathrm{t} \rangle}$ & P is uniquely satisfied . $(\iota x.P(x))$

(16) describes the partial function that maps a property true of only one individual to the individual that uniquely satisfies it. Such a denotation is the counterpart of the one hypothesized above wherein the sub-domain of the function denoted by *the* that it maps to `Nothing` is now just the sub-domain on which it is undefined. Just like in a programming context, incorporating a device for error handling in the language itself (or as a model-theoretic object) allows one to leverage a certain amount of control over error-prone behavior of the kind exhibited. In an account of presupposition, this feature is particularly important, as there often arise cases of what appear to be local failure in the face of a global success. Take (17), for instance.

(17) There is definitely a king of France, and the king of France is bald.

The usual judgment about (17) is that it lacks presuppositions and is therefore simply false. The presupposition appears somehow to be swallowed up by the antecedent so that, when the antecedent is true, the presupposition is satisfied, and when the antecedent is false, the entire text is false. Given the Haskell type `Bool` of truth values, a meaning for *and* that enforces such a pattern of behavior is

```
and :: Bool -> Maybe Bool -> Maybe Bool
and True n  = n
and False n = Just False
```

where `n` is either `Just True`, `Just False`, or `Nothing`. Now, the meaning of *and* checks the value— `True` or `False` —associated with its first conjunct, giving back the `Maybe Bool`-type value of its second conjunct in the former case, and `Just False` in the latter.

What if the value associated with the first conjunct is undefined, i.e., `Nothing`? In that case, `and` must be lifted via `pure` in order to compose it with the meaning of the first conjunct using `(<*>)`, and then lowered again, via `join`, after it has similarly composed with the meaning of its second conjunct, in order to get a term of the desired type: `Maybe Bool`.

```
join ((pure and) <*> Nothing <*> (pure n)) ⇒ Nothing
```

As illustrated, the resulting value is invariably undefined. Since *and* lexically composes with terms of type `Bool`, rather than terms of type `Maybe Bool`, lifting its meaning to compose with `Nothing` forces Maybe monadic functional application `(<*>)` to take control, passing `Nothing` along until the end of the computation. In general, we can observe that the presuppositions of a coordinate structure headed by *and* are satisfied—that is, in which case its meaning terminates in a value of either `Just True` or `Just False` — whenever the presuppositions of its first conjunct are satisfied, as well the implication that if its first conjunct is true, then so are the presuppositions of its second conjunct. This rendering may sound familiar from satisfaction-based conceptions of presupposition, as advanced in Karttunen 1973, Stalnaker 1974, and Heim 1983 (i.a.). It is common among satisfaction-based accounts to render the presuppositions of a coordinate structure headed by *and* satisfied whenever the presuppostions of the first conjunct are satisfied, along with the presuppositions of the second conjunct, given the truth

of the first conjunct. Indeed, the analysis of coordinate structures we will see later in the dissertation has exactly this shape (see Appendix A, subsection A.3.6).

Is there any reason not to simply use the Maybe monad from here on out? For handling the diverse variety of presuppositional phenomena, the Maybe monad turns out not to be enough. This is because presupposition has two aspects: a propositional aspect, and an anaphoric aspect, whereas the Maybe monad can only be used to deal with the former. Maybe allows one to check a condition at any point during the process of semantic composition and default to failure if the condition is not met; but sometimes the satisfaction of a presupposition involves not ensuring a truth-condition, but establishing a referential dependency. So, while the presupposition of (18) may, in fact, be encoded as a condition, i.e., truth-functionally,

(18) The sailor sat down.

it is more difficult to see how the same can be said for (19).

(19) He sat down.

(18) may be analyzed to carry a uniqueness presupposition, so that its truth conditions can be guaranteed by having it presuppose that there is a unique sailor and having it assert that a sailor sat down. Then, in virtue of uniqueness, any previously introduced sailor is bound to be the witness to the sentence's asserted content. (19), whose presupposition arises from a pronoun, must only have some antecedent, whether the property of being male is uniquely satisfied or not. Hence, (19) presupposes existence but not uniquness, and so there is no guarantee that the pronoun in (19) will corefer with its antecedent simply in virtue of its presupposition. Accomplishing coreference requires going outside of the Maybe monad. It is worth trying to state the conditions under which the presupposition of the pronoun is satisfied using Maybe. Using an *if then else* statement, one could condition the definedness of the pronoun meaning on whether or not some potential referent (which we may call `a`) satisfies the predicate `masc`.

```
masc :: Entity -> Bool

he :: Maybe Entity
he = if (masc a) then (Just a) else Nothing
```

One can even relativize the interpretation function to an assignment and vary the a depending on the assignment chosen, thereby establishing coreference via an index on the pronoun. Thus assignments are incorporated to encode presuppositions of coreference, while the Maybe monad is used to enforce the propositional conditions imposed on them. The point is that adding such a mechanism goes outside the Maybe monad, which isn't equipped, by itself, to handle coreference, but which requires extra mechanisms or devices, like assignment functions. Let us instead opt for a slightly more sophisticated device that can do both at once, while, at the same time, providing enough flexibility to handle anaphoric dependencies of various types. If we turn to graded monads, referential dependencies and conditions on how they are resolved can be handled under the same hood.[6]

2.4 Graded monads

The kinds of referential dependencies expressed as presuppositions appear to be fairly heterogeneous. For instance, while the definite description in (18) refers to an individual, in the following example, the presupposition might, for the sake of argument, be said to involve reference to a previous smoking event.

(21) Emily stopped smoking.

Moreover, if we combine expressions with either type of presupposition into a single sentence, we end up with a sentence, like (22), which has two kinds of presuppositions.

[6]The need for a treatment of anaphora that explicitly mentions the intended referents of anaphors has been motivated, by both Heim (1982) and Kamp (1988), using examples from Barbara Partee involving marbles. From Heim's dissertation:

(20) a. I dropped ten marbles and found all of them, except for one. It is probably under the sofa.

b. ?I dropped ten marbles and found only nine of them. It is probably under the sofa.

Stalnaker (1998) has responded to such examples by suggesting a notion of context that would render such a treatment unnecessary. Hence, notwithstanding the possibility that there is a Stalnakerian treatment of all the forms of anaphora to be discussed, I am proposing a mechanism for the representation of anaphora to be made compatible with discourse representation theory.

(22) The sailor stopped smoking.

(22) presupposes reference to an individual *and* an event. The individual presupposition is contributed by *the sailor*, and the event presupposition is contributed by *stopped smoking*. We can further complicate things by adding to the set of presuppositions reference to a relation between individuals and events.

(23) The sailor stopped.

Thus (23) presupposes reference to an individual, reference to an event, and reference to a relation between individuals and events which the individual and the presupposed event are presupposed to satisfy, all at once.

What we would like from a semantic analysis in order to manage all types of presupposition together is that it provide us with a graded monad. A graded monad has the beneficial features of a monad we saw above, but it allows for a controlled amount of uncertainty in the types associated with its side effects. As a result, we can presuppose reference to an individual, an event, and a relation simultaneously. Graded monads were first proposed in Katsumata 2014 and Orchard et al. 2014 for generalizing monads to handle computational effects of varying type, and a formal categorical theory of them is presented in Fujii et al. 2016.[7] The crucial innovation they provide is the addition to monads of a collection of effects—a monoid—in view of which the type of a program can be modified as it is sequenced with other programs. In the case at hand, the types associated with an expression's presuppositions can be modified as it is composed with other expressions that contribute their own presuppositions. The monoid of effects is crucial for determining how types change, as we'll see. Let's start by defining the set of types.

Definition 3 (Types)**.**

$$\mathcal{T} ::= \mathsf{At} \mid \mathcal{T} \to \mathcal{T}$$
$$\mathsf{At} ::= e \mid v \mid t \mid \#$$

What counts as a type ($\mathcal{T}$) is either an atom ($e, v, t, \#$), or an implication type (a type of the form $t_1 \to t_2$). The effects, over which we determine a monoid, will give us a means of keeping track of the types associated

[7] Strictly speaking, Katsumata 2014 and Orchard et al. 2014 introduced monads parameterized by a preordered monoid of types. Such "parametric effect monads" generalize graded monads, whose parameters are trivially preordered.

$$\epsilon + f = f \qquad \text{(Left Identity)}$$
$$e + \epsilon = e \qquad \text{(Right Identity)}$$
$$(e + f) + g = e + (f + g) \qquad \text{(Associativity)}$$

Figure 2.5: The Monoid Laws

with presuppositions. In practice, they give us a kind of list of types of referential dependencies that an expression might have registered. Following the strategy of Kobele 2018, we determine the set $\mathcal{E}$ of effects to be the free monoid over $\mathcal{T}$ (which we write $\mathcal{T}^*$): thus the "empty" effect is the sequence of types of length zero (which we refer to as ϵ); meanwhile, we write a sequence of types by separating the types with commas and surrounding them by curly brackets. Thus ϵ is the effect associated with not having any referential dependencies at all, and $\{t_1, \ldots, t_n\}$ is the effect associated with having referential dependencies of types $t_1, \ldots, t_n$. In the example above, if we take e to be the type of individuals and v to be the type of events, then the expression *the sailor stopped* would register its referential dependencies via the effect $\{e, e \to v \to t, v\}$; that is, its first referential dependency, introduced by the determiner, is of type e, its second, introduced by the absent verbal complement, is of type $e \to v \to t$, and its third, introduced by the main verb, is of type v. The monoidal structure of $\mathcal{T}^*$—taking the monoidal addition $(+)$ to be concatenation of sequences and ϵ to the the identity element—becomes important when we consider how to combine effects into new effects. Importantly, the monoid of effects satisfies the Monoid Laws of Fig. 2.5. Because $\mathcal{T}^*$ is a monoid, we can ensure that programs associated with different effects can be sequenced, while guaranteeing that the effect of the super-program that results is still in $\mathcal{T}^*$: it is gotten by adding the effects of the two programs together using $(+)$.

Given a monoid $\mathcal{E}$ of effects, a graded monad is a function $\mathbf{E} : \mathcal{E} \to \mathcal{T} \to \mathcal{T}$, together (like ordinary monads) with three operators—$(\cdot)^{\uparrow_{\mathbf{E}}}$, $(\cdot)^{\overset{\mathbf{E}}{\downarrow}}$, and $\mu_{\mathbf{E}}$, where the latter two are each parameterized by two effects in $\mathcal{E}$—which we

$$\mathtt{id}^{\overset{\uparrow}{\mathbf{E}}}\overset{\mathbf{E}}{\underset{\epsilon,f}{\downarrow}} = \mathtt{id} \qquad \text{(Identity)}$$

$$u\overset{\mathbf{E}}{\underset{e,f+g}{\downarrow}} \circ v\overset{\mathbf{E}}{\underset{f,g}{\downarrow}} = (((\circ)^{\overset{\uparrow}{\mathbf{E}}}\overset{\mathbf{E}}{\underset{\epsilon,e}{\downarrow}} u)\overset{\mathbf{E}}{\underset{e,f}{\downarrow}} v)\overset{\mathbf{E}}{\underset{e+f,g}{\downarrow}} \qquad \text{(Composition)}$$

$$(f\,a)^{\overset{\uparrow}{\mathbf{E}}} = f^{\overset{\uparrow}{\mathbf{E}}}\overset{\mathbf{E}}{\underset{\epsilon,\epsilon}{\downarrow}} a^{\overset{\uparrow}{\mathbf{E}}} \qquad \text{(Homomorphism)}$$

$$u\overset{\mathbf{E}}{\underset{e,\epsilon}{\downarrow}} a^{\overset{\uparrow}{\mathbf{E}}} = (\lambda f.f\,a)^{\overset{\uparrow}{\mathbf{E}}}\overset{\mathbf{E}}{\underset{\epsilon,e}{\downarrow}} u \qquad \text{(Interchange)}$$

Figure 2.6: The Graded Applicative Functor Laws

assign the following type signatures (where α and β are arbitrary).

$$(\cdot)^{\overset{\uparrow}{\mathbf{E}}} : \alpha \to \mathbf{E}\epsilon\alpha$$
$$(\cdot)\overset{\mathbf{E}}{\underset{e,f}{\downarrow}} : \mathbf{E}e(\alpha \to \beta) \to \mathbf{E}f\,\alpha \to \mathbf{E}(e+f)\beta$$
$$\mu_{\mathbf{E}e,f} : \mathbf{E}e(\mathbf{E}f\,\alpha) \to \mathbf{E}(e+f)\alpha$$

Analogous to ordinary monads, graded monads must satisfy specific laws. The first set of laws are those pertaining to graded applicative functors, introduced in Kobele 2018, which defines and then uses them to implement compositional Cooper storage, wherein the parameters of the monoid are determined by the types of the stored quantifier meanings. The laws for graded applicative functors are presented in Fig. 2.6. The Graded Monad Laws are presented in Fig. 2.7. Notice that both these sets of laws are identical in form to those of their corresponding non-graded variants. The crucial difference between the new laws and the old laws is the added constraints on the manipulation of effects e and f that results from the type signatures of $(\cdot)^{\overset{\uparrow}{\mathbf{E}}}$, $(\cdot)\overset{\mathbf{E}}{\underset{e,f}{\downarrow}}$, and $\mu_{\mathbf{E}e,f}$; these constraints ensure that the effects form a monoid, as already stipulated.

Let's now see what a graded monad for presupposition would look like.

2.5 Presuppositions in the metalanguage

The rest of this chapter introduces, through examples, a formal metalanguage for the model-theoretic analyses that will be presented in the rest of

$$\mu_{\mathbf{E}e,e}\overset{\uparrow}{m}^{\mathbf{E}} = m \qquad \text{(Left Identity)}$$

$$\mu_{\mathbf{E}e,\epsilon}((\lambda x.\overset{\uparrow}{x}^{\mathbf{E}})\overset{\uparrow}{}^{\mathbf{E}}_{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}_{\epsilon,e} m) = m \qquad \text{(Right Identity)}$$

$$\mu_{\mathbf{E}e+f,g}(\mu_{\mathbf{E}e,f}\, m) = \mu_{\mathbf{E}e,f+g}(\mu_{\mathbf{E}f,g}\overset{\uparrow}{}^{\mathbf{E}}_{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}_{\epsilon,e} m) \qquad \text{(Associativity)}$$

$$f\overset{\uparrow}{}^{\mathbf{E}}_{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}_{\epsilon,e+f}(\mu_{\mathbf{E}e,f}\, m) = \mu_{\mathbf{E}e,f}(f\overset{\uparrow}{}^{\mathbf{E}}_{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}_{\epsilon,f}\overset{\uparrow}{}^{\mathbf{E}}_{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}_{\epsilon,e} m) \qquad \text{(Naturality)}$$

Figure 2.7: The Graded Monad Laws

the dissertation. This metalanguage is nothing beyond the simply typed λ-calculus. I hope to reveal that the core claims of the dissertation—that presuppositional phenomena are the result of expressions exploiting mechanisms of scope-taking, that there is uniformity underlying the apparent heterogeneity of presupposition triggers, and that presupposition satisfaction is a combination of anaphora resolution and linguistic contexts taking control of the presupposition triggers that occur within them—can be given such a formal treatment. We will also see there that these features can be added to a directly compositional dynamic semantics similar to the one proposed by Muskens (1996).

2.5.1 Introducing sequents

The first ingredient of the metalanguage is what I will call a "sequent".[8] Sequents in our notation will be used to express some aspect of an expression's at-issue content, which will go to the right of a turnstile ($\Vdash$), given its presuppositions, which will be stated to the left of the same turnstile. As a small subscript underneath the turnstile itself will go a list of variables of various types, each indicating something that the current expression is

[8] Sequents have their origin in Frege (1967 [1879])'s *Begriffsschrift* as judgments, which used a vertical and a horizontal stroke together resembling a turnstile. They were recruited later by Gentzen (1964 [1934]) in the presentation of the sequent calculus, in which some disjunction of consequent formulae to the right of a turnstile is judged to hold, given the conjunction of some antecedent set of assumptions to the left. Sequents and turnstiles have since had a long history in logic, computer science, philosophy, and linguistics. I call the notation introduced here a "sequent", alluding to its superficial resemblance to the metalogical sequent.

missing. Say we want to represent the meaning of the definite description *the dog*. This expression denotes an individual, and it presupposes that this individual is a dog; moreover, exactly which individual it denotes is unknown, absent an antecedent of some kind. Thus we write the following sequent.

$$(\mathbf{dog}x \Vdash_x x)$$

This formula is to be read as follows: "Let's say we have some individual x, and x is a dog. Then, x!" In other words, we need an individual, as indicated by the e-type variable x, and, granted that the individual is a dog, we can use x however we'd like. Most likely, we'd like to use x to construct the meaning of a sentence, like that in (24).

(24) The dog barked.

We can construct this sentence's meaning by combining the term **bark** with the variable on the right side of the turnstile in the representation above to get the following representation.

$$(\mathbf{dog}x \Vdash_x \mathbf{bark}x)$$

This sequent will read, "Let's say we have some individual x, and x is a dog. Then, x barked!" The "some x" uttered as part of this paraphrase (and the one above) is meant to indicate our uncertainty about what x is.

At the moment, I am just hoping to pump intuitions. In the meantime, we can see what we want an expression with multiple presuppositions to look like. Let's take another look at the example from above, repeated here.

(8) The sailor stopped.

We would like to register three presuppositions: a sailor, a relation between individuals and events, and an event. We can do this by attributing the individual presuppositions to the lexical items that trigger them, starting with the definite determiner. Thus we would like the determiner to denote a function from terms of type e $\rightarrow$ t to sequents.

$$[\![the]\!] = (\lambda P.(Px \Vdash_x x))$$

We can then apply this term to the term **sailor** to get a full definite description meaning, which we can β-reduce.

$$\begin{aligned} & (\lambda P.(Px \Vdash_x x))\mathbf{sailor} \\ = \ & (\mathbf{sailor}x \Vdash_x x) \end{aligned}$$

Now, let's take care of the predicate. We'll have *stopped* denote as follows.

$$[\![stopped]\!] = (\lambda x.(Rxe \Vdash_{R,e} (\exists e'.\mathbf{stop}exe')))$$

The verb thus denotes a function from individuals to sequents. Given some individual, the resulting sequent ought to read, "Let's say we have a relation R between individuals and events, along with an event e, and R holds of our individual and e. Then, there's an event e' of the individual stopping e!" We now need a way to compose the meaning of the definite description and the meaning of the verb. Let us use the following magic.

$$\begin{aligned} & (\mathbf{sailor}x \Vdash_x (\lambda y.(Rye \Vdash_{R,e} (\exists e'.\mathbf{stop}eye')))x) \\ = \ & (\mathbf{sailor}x \Vdash_x (Rxe \Vdash_{R,e} (\exists e'.\mathbf{stop}exe'))) \end{aligned}$$

That is, let us apply the meaning of the verb to the meaning of the definite description *inside of the sequent corresponding to the definite*. Alas, we have ended up with something funny: a sequent embedded inside of another sequent. In a sense, we have two layers of presuppositions, and we would like to be able to collapse them into a single layer. Let's do that.

$$(\mathbf{sailor}x,\ Rxe \Vdash_{x,R,e} (\exists e'.\mathbf{stop}exe'))$$

This (pretend) analysis appears to have given us what we want. The resulting sequent can be read, "Let's say we have a sailor x, a relation R between individuals and events, and an event e such that R holds of x and e. Then, there's an event e' of x stopping e!"

2.5.2 A semantics of English

Importantly, the discussion above tracks operations that can be handled in a framework manifesting the methods of a graded monad. In particular, we define a function $\mathbf{P} : \mathcal{T}^* \to \mathcal{T} \to \mathcal{T}$ as follows, where o is an arbitrary result type.

Definition 4 ($\mathbf{P}$).

$$\begin{aligned} \mathbf{P}\epsilon\alpha &= (\alpha \to o) \to (\# \to o) \to o \\ \mathbf{P}(ae)\alpha &= a \to \mathbf{P}e\alpha \end{aligned}$$

We additionally introduce a constant ($\Vdash$), which is assigned the following type signature.

$$(\Vdash) : t \to \alpha \to \mathbf{P}\epsilon\alpha$$

Thus given M of type t and N of type α, $M \Vdash N$ is the term of type $\mathbf{P}\epsilon\alpha$ (i.e., $(\alpha \to o) \to (\# \to o) \to o$) representing either a "successful" value of type α or a "failed" value of type #. We thus use the standard (Böhm-Berarducci) embedding of coproducts into the simply typed λ-calculus using implication; in this case, the coproduct is the type of either a successful α or a failed #.

Moreover, the term $(\lambda x.(\mathbf{dog}x \Vdash x))$ is of type

$$\begin{aligned} & \mathbf{P}\{e\}e \\ = \ & e \to \mathbf{P}\epsilon e \\ = \ & e \to (e \to o) \to (\# \to o) \to o \end{aligned}$$

It is useful to introduce a notational convention: when regarding a term $(\lambda x_1, \ldots, x_n.M \Vdash N)$ to be of type $\mathbf{P}\{\alpha_1, \ldots, \alpha_n\}\beta$, we write the term $(M \Vdash_{x_1,\ldots,x_n} N)$; i.e., with the variables underneath the turnstile. Finally, we introduce a single term # : # representing failure to denote, and we introduce the following axiom schemata, in order to regulate the interpretation of terms of type $\mathbf{P}\epsilon\alpha$. $\top$ ('true') and $\bot$ ('false') are the type t terms which are always true and always false, respectively.

$$(\forall x^{\alpha}, s^{\alpha \to o}, f^{\# \to o}.(\top \Vdash x)sf = sx) \qquad \text{(Succeed)}$$

$$(\forall x^{\alpha}, s^{\alpha \to o}, f^{\# \to o}.(\bot \Vdash x)sf = f\#) \qquad \text{(Fail)}$$

In other words, if ϕ is true, then $\phi \Vdash M$ effectively returns M; if ϕ is false, then $\phi \Vdash M$ effectively returns #, giving rise to a Strawsonian implementation of presupposition failure.

At this point, we may see $\mathbf{P}$ to be a graded monad, given the monoid $\mathcal{T}^*$ of effects.

Definition 5 ($\mathbf{P}$).

$$\begin{aligned} a^{\overset{\uparrow}{\mathbf{P}}} &= (\lambda s, f.sa) \\ m^{\overset{\mathbf{P}}{\downarrow}}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}} &= (\lambda n, x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ &\qquad mx_1, \ldots, x_m(\lambda u.ny_1, \ldots, y_n(\lambda v.s(uv))f)f) \\ \mu_{\mathbf{P}\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}}m &= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ &\qquad mx_1, \ldots, x_m(\lambda n.ny_1, \ldots, y_n sf)f) \end{aligned}$$

Figure 2.8: Static inference rules

We have the following theorem, proved in Appendix B.

Theorem 1. **P** is a graded monad.

Given the notational convention introduced, along with the axiom schemata (Succeed) and (Fail), it can be seen that the definitions of these operators may be summarized in the following way (where $\top : t$ is the term which is always true, and ϕ, ψ is to be understood as the conjunction of ϕ and ψ). This equivalence is proved in Appendix B.

Fact 1.

$$a^{\overset{\uparrow}{\mathbf{P}}} = (\top \Vdash a)$$

$$(\phi \Vdash_{x_1,\ldots,x_m} M)\overset{\mathbf{P}}{\downarrow}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}}(\psi \Vdash_{y_1,\ldots,y_n} N) = (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN)$$

$$\mu_{\mathbf{P}\{\alpha_1 \ldots \alpha_m\},\{\beta_1,\ldots,\beta_n\}}(\phi \Vdash_{x_1,\ldots,x_m} (\psi \Vdash_{y_1,\ldots,y_n} M)) = (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M)$$

Finally, we introduce some static inference rules in Figure 2.8, which we will take to be applicable to meanings at any point during a syntactic derivation.[9]

2.6 Grammar

The syntax we present is just Applicative Categorial Grammar (Bar-Hillel, 1953), consisting of two rules: Foward Application and Backward Application. This choice is a matter of convenience (given the context-freeness of the patterns described), and the proposals about semantic composition are meant to be compatible with a variety of grammatical frameworks, in principle. The semantic interpretation of both Forward and Backward Application is functional application or its effect-sensitive variant. We thus

[9]The sequents used in the statements of these inference rules (using $\vdash$) are not to be confused with the constant ($\Vdash$) of the metalanguage. The former are metalogical sequents which allow the inference rules to be stated with reference to the types of the metalanguage expressions they involve.

define the following sets (in Backus-Naur notation), where $(\triangleright)$ and $(\triangleleft)$ abbreviate forward and backward application, respectively.

$$(\triangleright^*) ::= (\triangleright) \mid (\lambda u, v.((\triangleright^*)^{\uparrow \mathbf{P}}_{\epsilon,e} u)^{\mathbf{P}}_{\epsilon,f} v)$$
$$(\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v.((\triangleleft^*)^{\uparrow \mathbf{P}}_{\epsilon,e} u)^{\mathbf{P}}_{\epsilon,f} v)$$

The operations $(\triangleright^*)$ and $(\triangleleft^*)$ are what we use to interpret forward and backward syntactic functional application, respectively.

In general, any given lexical entry will be presented as a pair of a form and a logical expression from the metalanguage, annotated by a category. For category judgements, we use the symbol ::. For example, given that *dog* is a noun, its lexical entry (along with its category judgment) is

$$\langle dog, \mathbf{dog} \rangle :: \mathsf{n},$$

where **dog** is an expression of type $e \to t$.

To start off, we need a set of categories. These will be primitive categories for noun phrases, determiner phrases, verb phrases, and sentences, as well as implicational categories formed from those.

Definition 6 (Syntactic categories)**.**

$$\mathcal{C} ::= \mathtt{Prim} \mid (\mathcal{C}/\mathcal{C}) \mid (\mathcal{C}\backslash\mathcal{C})$$
$$\mathtt{Prim} ::= \mathsf{n} \mid \mathsf{d} \mid \mathsf{a} \mid \mathsf{v} \mid \mathsf{vPROG} \mid \mathsf{vINF} \mid \mathsf{s}$$

The two syntactic rules of the grammar are given in Definition 7.

Definition 7 (Forward and Backward Application)**.**

$$\frac{\langle w, m \rangle :: \mathsf{x/y} \qquad \langle v, n \rangle :: \mathsf{y}}{\langle wv, m \triangleright^* n \rangle :: \mathsf{x}} \, (FA)$$
$$\frac{\langle w, m \rangle :: \mathsf{y} \qquad \langle v, n \rangle :: \mathsf{y\backslash x}}{\langle wv, m \triangleleft^* n \rangle :: \mathsf{x}} \, (BA)$$

We can now do some derivations.

2.7 Warm-up exercises

Let's start out with the definite description from above. This description has two pieces, *the* and *dog*, to which we can attribute the following two interpretations, respectively.

$$(\lambda P.(Px \Vdash_x x)) : (e \to t) \to \mathbf{P}\{e\}e$$
$$\mathbf{dog} : e \to t$$

The lexical entries for *the* and *dog* are then as follows.

$$\langle \mathit{the}, (\lambda P.(Px \Vdash_x x)) \rangle :: \mathsf{d/n}$$
$$\langle \mathit{dog}, \mathbf{dog} \rangle :: \mathsf{n}$$

We can use Foward Application to combine these meanings.

$$\frac{\langle \mathit{the}, (\lambda P.(Px \Vdash_x x)) \rangle :: \mathsf{d/n} \qquad \langle \mathit{dog}, \mathbf{dog} \rangle :: \mathsf{n}}{\langle \mathit{the\ dog}, (\lambda P.(Px \Vdash_x x)) \triangleright^* \mathbf{dog} \rangle :: \mathsf{d}} (FA)$$

The resulting term—a meaning we can attribute to the entire definite description—is of type $\mathbf{P}\{e\}e$. It β-reduces as follows.

$$\begin{aligned} (\lambda P.(Px \Vdash_x x)) \triangleright^* \mathbf{dog} = \ & (\lambda P.(Px \Vdash_x x)) \triangleright \mathbf{dog} \\ \rightarrow_\beta^* \ & (\mathbf{dog}x \Vdash_x x) \end{aligned}$$

We can thus view this term as representing an individual which is missing an individual that it presupposes to be a dog.

Let us revisit our other example from above, repeated here.

(8) The sailor stopped.

To *sailor*, we can assign a type $(e \rightarrow t)$ constant, as we did to *dog*. To *stopped*, we assign the following lexical entry, making use of the denotation from above.[10]

$$\langle \mathit{stopped}, (\lambda x.(Rxe \Vdash_{R,e} (\exists e'.\mathbf{stop}exe'))) \rangle :: \mathsf{d\backslash s}$$

The type of the expression assigned to *stop* is thus $e \rightarrow \mathbf{P}\{e \rightarrow v \rightarrow t, v\}t$. It takes an individual to give back a sequent which requires both a relation between individuals and events of type $e \rightarrow v \rightarrow t$ and an event of type v. Assuming the presupposition is satisfied, a truth value is returned. We compose the meaning of this sentence via the following derivation, in which we've inferred uses of $(\cdot)^{\uparrow_{\mathbf{P}}}$, and $\mu_{\mathbf{P}\{e\},\{e\rightarrow v\rightarrow t,v\}}$.

$$\dfrac{\dfrac{\dfrac{\langle \mathit{the}, (\lambda P.(Px \Vdash_x x)) \rangle :: \mathsf{d/n} \qquad \langle \mathit{sailor}, \mathbf{sailor} \rangle :: \mathsf{n}}{\langle \mathit{the\ sailor}, (\mathbf{sailor}x \Vdash_x x) \rangle :: \mathsf{d}} (FA) \qquad \dfrac{\langle \mathit{stopped}, (\lambda x.(Rxe \Vdash_R (\exists e'.\mathbf{stop}exe'))) \rangle :: \mathsf{d\backslash s}}{\langle \mathit{stopped}, (\lambda x.(Rxe \Vdash_{R,e} (\exists e'.\mathbf{stop}exe'))) \rangle^{\uparrow_{\mathbf{P}}} :: \mathsf{d\backslash s}} \uparrow}{\langle \mathit{the\ sailor\ stopped}, (\mathbf{sailor}x \Vdash_x (Rxe \Vdash_{R,e} (\exists e'.\mathbf{stop}exe'))) \rangle :: \mathsf{s}} (BA)}{\langle \mathit{the\ sailor\ stopped}, (\mathbf{sailor}x, Rxe \Vdash_{x,R,e} (\exists e'.\mathbf{stop}exe')) \rangle :: \mathsf{s}} \mu$$

[10] Giving this denotation to *stopped* appears to ignore its alternant which takes a verb phrase complement. It is shown in chapter 4 how the two can be related by a type shift.

The result is a proposition with presuppositions, denoted by a term of type $\mathbf{P}\{e, e \to v \to t, v\}t$. It seeks an individual, a relation, and an event, and, so long as the individual is a sailor and the relation holds of the individual and the event, the proposition is true if the individual ended the event. This is a start.

2.8 Interim conclusions

So far, we have introduced a system whose applications to the phenomena of presupposition are still left mostly unexplored. Nevertheless, important progress has been made. In particular, we now have the following two things:

- A logic equipped for reasoning about semantic definedness as a side effect. Side effects, moreover, are encoded in "sequents".
- A graded monad determined by this logic which facilitates semantic composition in a grammar based on Functional Application via inference rules (i.e., type shifts).

The rest of the dissertation is geared toward showing how we can do more interesting things with presuppositions (like satisfy them!). In particular, while the next chapter extends the work of this chapter into a dynamic setting, chapter 5 gives analyses of various kinds of local satisfaction—the kinds that, in satisfaction-based theories of presupposition are used to explain apparent filtering behavior.

Before moving on, let's address a lingering question: why graded monads? Why not a more general and less powerful abstraction, as provided by the graded applicative functors defined in Kobele 2018? Indeed, if we kept only the operations $(\cdot)^{\overset{\uparrow}{\mathbf{P}}}$ and $(\cdot)^{\overset{\mathbf{P}}{\downarrow}}_{e,f}$, eliminating $\mu_{\mathbf{P}_{e,f}}$, then we would go from having a graded monad to having a graded applicative functor. What even some of the most run-of-the-mill examples of presupposition show is that presupposition, at its heart, really is constituted by a (graded) monad. Consider, again, the meaning we gave to the definite determiner:

$$(\lambda P.(Px \Vdash_x x))$$

Descriptively speaking, the determiner uses the meaning of its complement to decide the content of the presupposition born by the resulting definite description. Indeed, this is a fact about the lexical content of the determiner; it is not predictable, in any obvious way (for example, based on the type of the resulting value), when a particular argument to a function serving as a lexical meaning will have a role in determining any resulting side effects, and when it will not. Hence, it seems we need to let lexical meanings run their course in semantic derivations and then consolidate loose layers of side effects afterwards.

An example can help illustrate. Take the noun phrase *picture of the dog*, which we derive as follows, given a relational $e \to e \to t$-type meaning for *picture*, and wherein *of* is taken to denote Backward Functional Application.

$$\dfrac{\dfrac{\langle \textit{picture}, \mathbf{picture}\rangle :: \mathsf{n}}{\langle \textit{picture}, \mathbf{picture}^{\mathbf{P}}\rangle :: \mathsf{n}}\uparrow \quad \dfrac{\dfrac{\langle \textit{of}, (\triangleleft)\rangle :: (\mathsf{n}\backslash\mathsf{n})/\mathsf{d}}{\langle \textit{of}, (\triangleleft)^{\mathbf{P}}\rangle :: (\mathsf{n}\backslash\mathsf{n})/\mathsf{d}}\uparrow \quad \dfrac{\langle \textit{the}, (\lambda P.(Px \Vdash_x x))\rangle :: \mathsf{d}/\mathsf{n} \qquad \langle \textit{dog}, \mathbf{dog}\rangle :: \mathsf{n}}{\langle \textit{the dog}, (\mathbf{dog}x \Vdash_x x)\rangle :: \mathsf{d}}(FA)}{\langle \textit{of the dog}, (\mathbf{dog}x \Vdash_x (\lambda f.fx))\rangle :: \mathsf{n}\backslash\mathsf{n}}(FA)}{\langle \textit{picture of the dog}, (\mathbf{dog}x \Vdash_x \mathbf{picture}x)\rangle :: \mathsf{n}}(BA)$$

This meaning is of type $\mathbf{P}\{e\}(e \to t)$; it seeks an individual term and returns a predicate meaning. In our monadic framework, we ought to give the full definite description *the picture of the dog* the following analysis.

$$\dfrac{\dfrac{\dfrac{\langle \textit{the}, (\lambda P.(Py \Vdash_y y))\rangle :: \mathsf{d}/\mathsf{n}}{\langle \textit{the}, (\lambda P.(Py \Vdash_y y))^{\mathbf{P}}\rangle :: \mathsf{d}/\mathsf{n}}\uparrow \qquad \langle \textit{picture of the dog}, (\mathbf{dog}x \Vdash_x \mathbf{picture}x)\rangle :: \mathsf{n}}{\langle \textit{the picture of the dog}, (\mathbf{dog}x \Vdash_x (\mathbf{picture}xy \Vdash_y y))\rangle :: \mathsf{d}}(FA)}{\langle \textit{the picture of the dog}, (\mathbf{dog}x, \mathbf{picture}xy \Vdash_{x,y} y)\rangle :: \mathsf{d}}\mu$$

Now we have a term of type $\mathbf{P}\{e, e\}e$, which seeks two individual terms and gives back the second one.

In contrast, if we had only a graded applicative functor, we could indeed compose the meaning of the determiner with that of its complement.

$$\dfrac{\langle \textit{the}, (\lambda P.(Py \Vdash_y y))^{\mathbf{P}}\rangle :: \mathsf{d}/\mathsf{n} \qquad \langle \textit{picture of the dog}, (\mathbf{dog}x \Vdash_x \mathbf{picture}x)\rangle :: \mathsf{n}}{\langle \textit{the picture of the dog}, (\mathbf{dog}x \Vdash_x (\mathbf{picture}xy \Vdash_y y))\rangle :: \mathsf{d}}(FA)$$

But, now we have $\mathbf{dog}x$ stuck in an outer layer of presupposition, and without the availability of $\mu_{\mathbf{P}\{e\},\{e\}}$, it will stay there indefinitely—an awkward

state of affairs. At the moment, there may not be any obvious empirical cost to the baggage incurred by such an elaborate term, but there is also no obvious empirical benefit to maintaining a separation between layers of presuppositions registered during a derivation. As we will see in chapter 4, consolidating layers becomes quite useful in the treatment of more complex presuppositional phenomena.

Chapter 3

Dynamic presupposition satisfaction

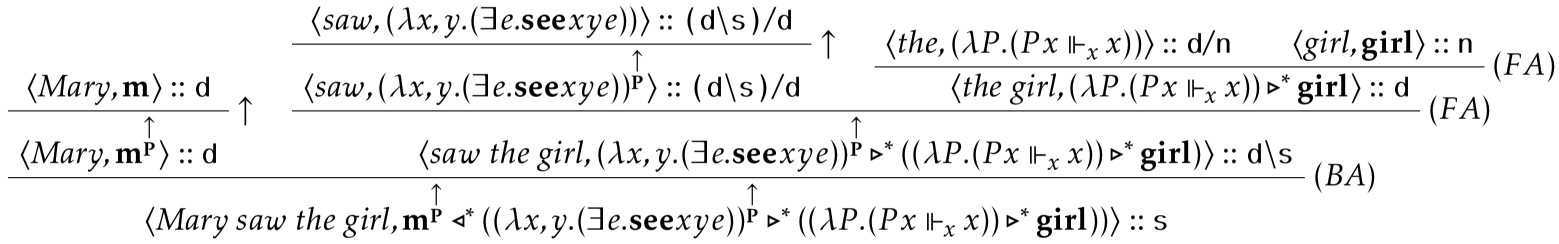
3.1 Introduction

The framework developed in the last chapter has yet to exhibit most of its analytical potential. The task now is to develop it into a full-fledged dynamic approach to those phenomena manifesting the generalization that presuppositions *change* as discourse unfolds. But let's do a quick recap.

According to the last chapter, when composition proceeds, presuppositions project.

(25) Mary saw the girl.

Since *the girl* presupposes reference to a girl, so does (25), which contains it. Projection, in the framework of chapter 2, is an automatic result of semantic composition, which proceeds independently of any incurred presuppositional side effects. Let's quickly see, again, how this works in a derivation for (25), assuming the obvious meanings for the expressions *Mary*, *saw*, and *girl*.


⟨Mary, m⟩ :: d
⟨Mary, m^P⟩ :: d
↑
⟨saw, (λx, y.(∃e.seexye))⟩ :: (d\s)/d
⟨saw, (λx, y.(∃e.seexye))^P⟩ :: (d\s)/d
↑
⟨the, (λP.(Px ⊩_x x))⟩ :: d/n
⟨girl, girl⟩ :: n
⟨the girl, (λP.(Px ⊩_x x)) ▷* girl⟩ :: d
(FA)
⟨saw the girl, (λx, y.(∃e.seexye))^P ▷* ((λP.(Px ⊩_x x)) ▷* girl)⟩ :: d\s
(FA)
⟨Mary saw the girl, m^P ◁* ((λx, y.(∃e.seexye))^P ▷* ((λP.(Px ⊩_x x)) ▷* girl))⟩ :: s
(BA)

If we β-reduce the resulting meaning, we obtain the following.

$$
\begin{aligned}
& \mathbf{m}^{\uparrow^{\mathbf{P}}} \triangleleft^{*} ((\lambda x, y.(\exists e.\mathbf{see} x y e))^{\uparrow^{\mathbf{P}}} \triangleright^{*} ((\lambda P.(P x \Vdash_x x)) \triangleright^{*} \mathbf{girl})) \\
\rightarrow^{*}_{\beta} \ & ((\triangleleft)^{\uparrow^{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,\epsilon}} \mathbf{m}^{\uparrow^{\mathbf{P}}})^{\downarrow^{\mathbf{P}}_{\epsilon,e}} (((\triangleright)^{\uparrow^{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,\epsilon}} (\lambda x, y.(\exists e.\mathbf{see} x y e))^{\uparrow^{\mathbf{P}}})^{\downarrow^{\mathbf{P}}_{\epsilon,e}} ((\lambda P.(P x \Vdash_x x)) \triangleright \mathbf{girl})) \\
\rightarrow^{*}_{\beta} \ & ((\triangleleft)^{\uparrow^{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,\epsilon}} \mathbf{m}^{\uparrow^{\mathbf{P}}})^{\downarrow^{\mathbf{P}}_{\epsilon,e}} (((\triangleright)^{\uparrow^{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,\epsilon}} (\lambda x, y.(\exists e.\mathbf{see} x y e))^{\uparrow^{\mathbf{P}}})^{\downarrow^{\mathbf{P}}_{\epsilon,e}} (\mathbf{girl} x \Vdash_x x)) \\
\rightarrow^{*}_{\beta} \ & ((\triangleleft)^{\uparrow^{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,\epsilon}} \mathbf{m}^{\uparrow^{\mathbf{P}}})^{\downarrow^{\mathbf{P}}_{\epsilon,e}} (\mathbf{girl} x \Vdash_x (\lambda y.(\exists e.\mathbf{see} x y e))) \\
\rightarrow^{*}_{\beta} \ & (\mathbf{girl} x \Vdash_x (\exists e.\mathbf{see} x \mathbf{m} e))
\end{aligned}
$$

As we see, to compose side-effect-free *saw* with the effectful definite description meaning, we must lift it via $(\cdot)^{\uparrow^{\mathbf{P}}}$ and then expose its first argument for functional application via $(\cdot)^{\downarrow^{\mathbf{P}}_{\epsilon,e}}$. Since the lifting of *saw* endows it with trivial side effects, the side effects of the definite description are passed up, as seen on line four. Lifting *Mary* and exposing the second argument of *saw* allows us to combine them, but, again, in a way that passes up the side effects registered by the definite description. Thus we see that the perspective offered by chapter 2 is one which rests upon the following principle:

(**PPMS**) Presupposition projection is just monadic scope-taking.

This principle is just a more refined statement of the claims made in the introduction to this thesis. The last chapter can therefore be seen as having presented an adequate theory of holes to presuppositions: a hole is just an expression whose meaning composes with values—that is, at-issue content. What we need now is a theory of non-holes.

The framework presented in this chapter takes after its dynamic satisfaction-theoretic precursors in the broad strategy it uses to filter presuppositions. The explanation of phenomena showing presupposition filtering will essentially run as follows. Take (26), a modification of (25) that provides an antecedent for the definite description.

(26) A girl walked in. Mary saw the girl.

The important reading of (26) is the one on which the girl that walked in is the same as the girl that Mary saw. Strikingly, unlike (25), (26) lacks presuppositions on this reading. Following the explanatory strategy—originating

in Karttunen 1974—of satisfaction theory, let's conjecture that a presupposition associated with a particular presupposition trigger projects out of a linguistic context if it isn't satisfied in that linguistic context; otherwise, the presupposition disappears. In (26), the presupposition triggered by the definite description in the second sentence isn't satisfied in that sentence, which is why it can still be observed there. However, the same presupposition is satisfied in the linguistic context of the definite description that contains both sentences in (26) as a result of the fact that the first sentence entails the presupposition. Thus (26) itself appears not to have presuppositions. The presupposition triggered inside of it doesn't project because it is also satisfied inside of it. Though, if it weren't satisfied, it would project. In this chapter, we'll investigate a dynamic framework that can be related to this strategy of explanation.

3.2 A dynamic framework

3.2.1 Representing information states

The purpose of any dynamic semantic framework is to provide an account of the effect that utterances have on a discourse, which, in frameworks of the kind originating in Stalnaker 1974, is analyzed as an information state—a representation of the informational content of the discourse as it has so far unfolded. Information states will represent two kinds of information about a discourse. These are: (1) information about elements of a set of points—which we call "possible worlds"—at which a given discourse can be considered to be true; (2) information about whether or not, at any given juncture in the discourse, a particular "register" is a live candidate for anaphora in later pieces of discourse. To represent anaphoric potential, along with the non-determinism introduced by indefinites, we will use a system akin to the compositional discourse representation theory introduced in Muskens 1996. In Muskens's system, individuals are stored in registers, i.e., discourse referents, and any given continuous piece of a discourse is taken to denote a relation between an input assignment of individuals to registers and an output assignment of individuals to registers. Each assignment of individuals to registers, Muskens calls a "state"; we will simply encode states as functions from registers to individuals. To represent a discourse, we thus allow ourselves access to a model which has at least a set D of individuals, a set W of possible worlds, and a denumerably infinite set R of registers.

Similar to the theories of Groenendijk and Stokhof (1991), and Muskens,

we regard discourses as relating input states to output states. However, we will relate any input state to an output state *across a world*. To see what this means, let's first introduce an updated set of types.

Definition 8 (Types)**.**

$$\mathcal{T} ::= \mathsf{At} \mid \mathcal{T} \to \mathcal{T}$$
$$\mathsf{At} ::= e \mid v \mid t \mid s \mid i \mid \#$$

Thus we have types for individuals (e), events (v), truth values (t), worlds (s), and registers (i). We will require a denumerably infinite set of terms for registers, which we obtain via the following two constants.

$$\mathbf{i} : i$$
$$(\cdot)' : i \to i$$

We'll thus have $\mathbf{i}$ as our first register, $\mathbf{i}'$ as the second, etc. As an abbreviatory convention, I will refer to the first register $\mathbf{i}$ as 0, the second register $\mathbf{i}'$ as 1, etc.

Any given piece of discourse will be taken to denote an "information state". An information state is denoted by a term of type $(i \to e) \to s \to (i \to e) \to t$. In other words, given an input assignment and a world, an information state returns a set of output assignments that satisfy some number of constraints. Some information states may be "tests" (in the terminology of Groenendijk and Stokhof), in that they may return either an empty set, given some input assignment and some world, or they may return the singleton set containing the given assignment. Tests will sometimes be seen to be contributed by information states denoted by sentences containing pronouns, while, sometimes, by sentences that have felicitous propositional logic paraphrases. As a notational convention, we write T for the type of information states, and additionally, E and V for the types $(i \to e) \to s \to (i \to e) \to e$ and $(i \to e) \to s \to (i \to e) \to v$, respectively, of expressions denoting functions from assignments to functions from worlds to functions from assignments to individuals and events, respectively.

One more preliminary. We will often specify an information state using set-comprehension and pairing notations. For example, we may write the information state in which Mary slept as follows.

$$(\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sleep}w\mathbf{m}e)\})$$

The actual term specifying this information state would be an abstraction like the following.

$$(\lambda g, w, g'.g = g' \wedge (\exists e.\mathbf{sleep} w \mathbf{m} e))$$

Given this basic machinery, we can begin to build information states. We assign the following meanings to the expressions *Mary* and *slept*, respectively.

$$(\lambda g, w, g'.\mathbf{m}) : E$$
$$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sleep} w (x g w g) e)\}) : E \to T$$

We can then derive *Mary slept* as follows.

$$\frac{\langle \mathit{Mary}, (\lambda g, w, g'.\mathbf{m})\rangle :: \mathtt{d} \qquad \langle \mathit{slept}, (\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sleep} w (x g w g) e)\})\rangle :: \mathtt{d\backslash s}}{\langle \mathit{Mary\ slept}, (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sleep} w \mathbf{m} e)\})\rangle :: \mathtt{s}}\ (BA)$$

In fact, by using a general lexical type shift, such higher-order dynamic meanings can be constructed out of the more basic static intensional meanings one ought to assign to expressions. The type shift we use is modeled after the intensionality transformation of de Groote and Kanazawa 2013. First, at each type α, we define a function `lift` which lifts a function of type $s \to \alpha$ into a function of type $(i \to e) \to s \to (i \to e) \to \alpha$.

Definition 9 (`lift`)**.**

$$\mathtt{lift} : (s \to \alpha) \to (i \to e) \to s \to (i \to e) \to \alpha$$
$$\mathtt{lift} := (\lambda x, g, w, g'.xw)$$

Following de Groote and Kanazawa (2013), we define a function $\overline{(\cdot)}$.

$$\overline{e} = E$$
$$\overline{v} = V$$
$$\overline{t} = T$$
$$\overline{\alpha \to \beta} = \overline{\alpha} \to \overline{\beta}$$

The type shift is presented in terms of two functions `dyn` and `sta`, each indexed by a type. At any given type a, these functions have the following type signatures (where A is defined as $(i \to e) \to s \to (i \to e) \to a$).

$$\mathsf{dyn}_a : A \to \overline{a}$$
$$\mathtt{sta}_a : \overline{a} \to A$$

At all atomic types a, except for t, we then have that $\mathsf{dyn}_a x = \mathtt{sta}_a x = x$, as well as that $\mathtt{sta}_t \phi = \phi$. dyn_t, we define as

$$\mathsf{dyn}_t \phi = (\lambda g, w, g'.g = g' \wedge \phi g w g')$$

At complex types, we define the functions using mutual recursion as follows.

$$\mathtt{dyn}_{\alpha\to\beta}M = (\lambda x^{\overline{\alpha}}.\mathtt{dyn}_{\beta}(\lambda g,w,g'.Mgwg'(\mathtt{sta}_{\alpha}xgwg')))$$
$$\mathtt{sta}_{\alpha\to\beta}M = (\lambda g,w,g',x^{\alpha}.\mathtt{sta}_{\beta}(M(\mathtt{dyn}_{\alpha}(\lambda h,w',h'.x)))gwg')$$

Given an ordinary lexical meaning a of type $\mathrm{s}\to\alpha$, we then dynamize it by doing

$$\mathtt{dyn}(\mathtt{lift}a)$$

As an example, take the basic meaning of *slept*.

$$(\lambda w,x.(\exists e.\mathbf{sleep}wxe))$$

To get the shifted interpretation of *slept* we do the following.

$$\begin{aligned}
&\mathtt{dyn}_{e\to t}(\mathtt{lift}(\lambda w,x.(\exists e.\mathbf{sleep}wxe)))\\
=\ &\mathtt{dyn}_{e\to t}(\lambda g,w,g',x.(\exists e.\mathbf{sleep}wxe))\\
=\ &(\lambda x.\mathtt{dyn}_{t}(\lambda g,w,g'.(\exists e.\mathbf{sleep}w(\mathtt{sta}_{e}xgwg')e)))\\
=\ &(\lambda x.\mathtt{dyn}_{t}(\lambda g,w,g'.(\exists e.\mathbf{sleep}w(xgwg')e)))\\
=\ &(\lambda x,g.\{\langle w,g\rangle \mid (\exists e.\mathbf{sleep}w(xgwg')e)\})
\end{aligned}$$

This result is just the meaning for *slept* from above.

The introduction of information states allows a fairly simple notion of discourse update. Let's define the following abbreviation (+) for update on information states.

Definition 10 (+)**.**

$$\begin{aligned}
(+) &: T\to T\to T\\
\phi+\psi &:= (\lambda g,w,g''.(\exists g'.\phi gwg'\wedge\psi g'wg''))
\end{aligned}$$

For the kinds of examples we've been considering, discourse update amounts to boolean conjunction. To see this, take the following discourse as an example.

(27) Mary slept. John slept.

To analyze this discourse, we can add the following lexical entry for ;.

$$\langle ;,(+)\rangle :: \mathrm{s}\backslash(\mathrm{s}/\mathrm{s})$$

λ-term	Shorthand	Type
$(\lambda g, w, g'.\mathbf{m})$	$\mathtt{m}$	E
$(\lambda g, w, g'.\mathbf{j})$	$\mathtt{j}$	E
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{dog}w(xgwg)\})$	$\mathtt{dog}$	$E \to T$
$(\lambda P, Q, g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g', (P(\lambda g, w, g'.g_i) + Q(\lambda g, w, g'.g_i))g'wg'')\})$	$\mathtt{a}_i$	$(E \to T) \to (E \to T) \to T$

Table 3.1: Abbreviations

At this point, it is convenient to introduce abbreviations for our meanings; these are given in Table 3.1. We can then give (27) the following derivation, given the lexical entries that we have adopted.

$$\dfrac{\dfrac{\langle \textit{Mary slept}, \mathtt{sleptm}\rangle :: \mathtt{s} \qquad \langle ;, (+)\rangle :: \mathtt{s}\backslash(\mathtt{s}/\mathtt{s})}{\langle \textit{Mary slept} ;, (\lambda\psi.(\mathtt{sleptm}) + \psi)\rangle :: \mathtt{s}/\mathtt{s}}\,(BA) \qquad \langle \textit{John slept}, \mathtt{slept\,j}\rangle :: \mathtt{s}}{\langle \textit{Mary slept ; John slept}, (\mathtt{sleptm}) + (\mathtt{slept\,j})\rangle :: \mathtt{s}}\,(FA)$$

If we replace each abbreviation with its definition and β-reduce, we obtain the following result.

$$(\mathtt{sleptm}) + (\mathtt{slept\,j}) = (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sleep}w\mathbf{m}e) \wedge (\exists e.\mathbf{sleep}w\mathbf{j}e)\})$$

Hence, we obtain a new information state which is the boolean conjunction of the two olds ones. We get more interesting effects, however, when discourse update interacts with the semantic contribution of indefinites.

To indefinites, we give a meaning that allows them to modify the assignments related by information states (see, i.a., Groenendijk and Stokhof, 1991; Dekker, 1994; Muskens, 1996). For now, we have the indefinite article denote as follows, thus allowing it to occur only in subject position. We also associate the indefinite article with an index, indicating with respect to which register it modifies the output assignments of the information state it produces. Thus there are (countably) infinitely many indefinite articles, each with a different denotation. In general, we write $g[x_1, \ldots, x_n]g'$ to mean that g and g' differ at most in the individuals they assign to registers $x_1, \ldots, x_n$.

$$\langle a_i, (\lambda P, Q, g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g', (P(\lambda g, w, g'.g_i) + Q(\lambda g, w, g'.g_i))g'wg'')\})\rangle :: (\mathtt{s}/(\mathtt{d}\backslash\mathtt{s}))/\mathtt{n}$$

That is, the indefinite article non-deterministically feeds a single individual—generated by a non-deterministic variant of the input assignment—to both its restriction and its scope, updates the former with the latter, and collects up the resulting assignments. Let's look at an example: *a dog slept*.

We ascribe to *dog* the following lexical entry (with a meaning which we can derive from the dynamization type shift defined above from a more basic intensional meaning **dog**).

$$\langle dog, (\lambda x, g.\{\langle w, g\rangle \mid \mathbf{dog}w(xgwg)\})\rangle :: \mathsf{n}$$

We can then derive the full sentence as follows (using the appropriate abbreviations from Table 3.1).

$$\dfrac{\dfrac{\langle a_1, \mathsf{a}_1\rangle :: (\mathsf{s}/(\mathsf{d}\backslash\mathsf{s}))/\mathsf{n} \qquad \langle dog, \mathtt{dog}\rangle :: \mathsf{n}}{\langle a_1\ dog, \mathtt{a dog}\rangle :: \mathsf{s}/(\mathsf{d}\backslash\mathsf{s})}(FA) \qquad \langle slept, \mathtt{slept}\rangle :: \mathsf{d}\backslash\mathsf{s}}{\langle a_1\ dog\ slept, \mathsf{a}_1\,\mathtt{dog slept}\rangle :: \mathsf{s}}(FA)$$

Expanding the abbreviations and β-reducing yields the following result.

$$\mathsf{a}_1\,\mathtt{dog slept} = (\lambda g.\{\langle w, g'\rangle \mid g[1]g' \wedge \mathbf{dog}wg'_1 \wedge (\exists e.\mathbf{sleep}wg'_1e)\})$$

Thus we end up with an information state that relates an assignment across a world to any assignment that varies from it at register 1 by some dog who slept in the world.

3.2.2 Abstracting over the context

Information states are what we will take to be the semantic values associated with discourses, while presuppositions are analyzed as their side effects. As we saw at the beginning of this chapter, presuppositions may be modified; thus we require a way of allowing contexts to modify the presuppositions incurred during discourse.

In order to allow presuppositions to be modified, we need to make them sensitive to the identity of the semantic environment in which they are triggered within the discourse as a whole. The proposal we'll look at in this chapter is one wherein semantic contexts leverage a certain amount of control over the presuppositions of expressions that occur within them. This will be done by weakening presuppositions from ultimatums about what's true in, say, a model to mere constraints on a given input context. To start, let's recall the static graded monad for presupposition introduced in the last chapter (now presented in terms of sequents).

Definition 11 (**P**)**.**

$$\begin{aligned}
\mathbf{P} &: \mathcal{T}^* \to \mathcal{T} \to \mathcal{T} \\
\mathbf{P}\epsilon\alpha &= (\alpha \to o) \to (\# \to o) \to o \\
\mathbf{P}(ae)\alpha &= a \to \mathbf{P}e\alpha \\
a^{\overset{\uparrow}{\mathbf{P}}} &= (\top \Vdash a) \\
(\phi \Vdash_{x_1,\ldots,x_m} M)^{\overset{\mathbf{P}}{\downarrow}}_{\alpha_1 \ldots \alpha_m, \beta_1 \ldots \beta_n}(\psi \Vdash_{y_1,\ldots,y_n} N) &= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) \\
\mu_{\mathbf{P}\alpha_1 \ldots \alpha_m, \beta_1 \ldots \beta_n}(\phi \Vdash_{x_1,\ldots,x_m} (\psi \Vdash_{y_1,\ldots,y_n} M)) &= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M)
\end{aligned}$$

In order to incorporate contextual control over an expression's presuppositions, we update this graded monad with Reader functionality, so that an expression's presuppositions may be made subject to change. We can do this by wrapping this functionality around the graded monad we already have. Given some type r, we introduce r as a dependency at every monadic type in **P** via a graded Reader monad transformer $\mathbf{RT} : \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T}$, where $\mathcal{E}$ is any monoid of parameters.

Definition 12 (**RT**)**.**

$$\begin{aligned}
\mathbf{RT}(r)(\mathbf{E_0})e\alpha &= r \to \mathbf{E_0}e\alpha \\
a^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} &= (\lambda s.a^{\overset{\uparrow}{\mathbf{E_0}}}) \\
u^{\overset{\mathbf{RT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} &= (\lambda v, s.(us)^{\overset{\mathbf{E_0}}{\downarrow}}_{e,f}(vs)) \\
\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}} m &= (\lambda s.\mu_{\mathbf{E_0}e,f}((\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\overset{\mathbf{E_0}}{\downarrow}}_{\epsilon,e}(ms)))
\end{aligned}$$

For our purposes, we will choose r to be $T \to T$. This is because we want expressions with presuppositions to be able to take a given "context changer"—something of type $T \to T$—and modify their presuppositions accordingly.

Importantly, we have the following theorem, which is proved in Appendix B.

Theorem 2. If $\mathbf{E_0}$ is a graded monad with monoid of parameters $\mathcal{E}$, then so is $\mathbf{RT}(r)(\mathbf{E_0})$, for any type r.

This functionality is not yet quite enough. We would also like to keep track of which registers have already been used in a given discourse containing indefinites. In order to do so, we may add further Reader functionality, i.e., to read in a register. Thus we apply the graded Reader monad

λ-term	Shorthand	Type
$(\lambda x, y, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w(xgwg)(ygwg)e)\})$	saw	$E \to E \to T$
$(\lambda P, i, c.(c(Px) = \mathtt{true} \Vdash_x x))$	the	$(E \to T) \to \mathbf{D}\{E\}E$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{girl} w(xgwg)\})$	girl	$E \to T$

Table 3.2: Abbreviations

transformer again, but this time choosing the r to be i. The result of applying the graded Reader monad transformer to **P** twice is a graded monad **D** for dynamic presupposition. In particular, we define a function $\mathbf{D} : \mathcal{T}^* \to \mathcal{T} \to \mathcal{T}$ as $\mathbf{D} = \mathbf{RT}(i)(\mathbf{RT}(T \to T)(\mathbf{P}))$. This delivers the following new graded monad **D**.

Definition 13 (D).

$$\begin{aligned}
\mathbf{D}e\alpha &= i \to (T \to T) \to \mathbf{P}e\alpha \\
\overset{\uparrow}{a}^{\mathbf{D}} &= (\lambda i, c.\overset{\uparrow}{a}^{\mathbf{P}}) \\
\overset{\mathbf{D}}{\underset{\downarrow}{u}}_{e,f} &= (\lambda v, i, c.(uic)\overset{\mathbf{P}}{\underset{\downarrow}{}}_{e,f}(vic)) \\
\mu_{\mathbf{D}e,f}m &= (\lambda i, c.\mu_{\mathbf{P}e,f}((\lambda n.nic)\overset{\uparrow\mathbf{P}}{\underset{\downarrow}{}}_{e,f}(mic)))
\end{aligned}$$

Let's now go back to the example sentence in (25) as an illustration of what has been proposed. First, we introduce two abbreviatory conventions. Rather than the term

$$(\lambda g, w, g'.g = g'),$$

we will write

$$\mathtt{true},$$

as it denotes a relation true of all worlds, only requiring that an input assignment be sent out unmodified. To the words *Mary*, *saw*, *the*, and *girl*, we assign the following lexical entries, using the abbreviations defined in Table 3.2. Note that only the meaning of the definite determiner registers presuppositional side effects associated with a type belonging to the graded monad **D**.

$$\begin{aligned}
\langle \mathit{Mary}, \mathtt{m}\rangle &:: \mathtt{d} \\
\langle \mathit{saw}, \mathtt{saw}\rangle &:: \mathtt{(d\backslash s)/d} \\
\langle \mathit{the}, \mathtt{the}\rangle &:: \mathtt{d/n} \\
\langle \mathit{girl}, \mathtt{girl}\rangle &:: \mathtt{n}
\end{aligned}$$

Note, moreover, that the presuppositional side effects are themselves taken to depend on an input context; the definite description has yielded its presuppositions to the local context to say, "do with me what you will." The derivation of (25), given this graded monad, looks a lot like the derivation given for it above. The only change is the particular graded monad determining the type that effectual meanings inhabit. This is because the way values compose is not affected by this choice of type: in particular, functional application is defined equivalently.

$$(\triangleright^*) ::= (\triangleright) \mid (\lambda u, v.((\triangleright^*)^{\uparrow^{\mathbf{D}}_{\epsilon,e}} u)^{\downarrow^{\mathbf{D}}_{e,f}} v)$$
$$(\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v.((\triangleleft^*)^{\uparrow^{\mathbf{D}}_{\epsilon,e}} u)^{\downarrow^{\mathbf{D}}_{e,f}} v)$$

Using these definitions of $(\triangleright^*)$ and $(\triangleleft^*)$, we obtain the following derivation for (25), given correspondingly updated inference rules in Figure 3.1.

$$\dfrac{\dfrac{\langle \mathit{Mary}, \mathtt{m}\rangle :: \mathtt{d}}{\langle \mathit{Mary}, \mathtt{m}^{\uparrow^{\mathbf{D}}}\rangle :: \mathtt{d}} \uparrow \quad \dfrac{\dfrac{\langle \mathit{saw}, \mathtt{saw}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}}{\langle \mathit{saw}, \mathtt{saw}^{\uparrow^{\mathbf{D}}}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}} \uparrow \quad \dfrac{\langle \mathit{the}, \mathtt{the}\rangle :: \mathtt{d}/\mathtt{n} \qquad \langle \mathit{girl}, \mathtt{girl}\rangle :: \mathtt{n}}{\langle \mathit{the\ girl}, \mathtt{the} \triangleright^* \mathtt{girl}\rangle :: \mathtt{d}}\,(FA)}{\langle \mathit{saw\ the\ girl}, \mathtt{saw}^{\uparrow^{\mathbf{D}}} \triangleright^* (\mathtt{the} \triangleright^* \mathtt{girl})\rangle :: \mathtt{d}\backslash\mathtt{s}}\,(FA)}{\langle \mathit{Mary\ saw\ the\ girl}, \mathtt{m}^{\uparrow^{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow^{\mathbf{D}}} \triangleright^* (\mathtt{the} \triangleright^* \mathtt{girl}))\rangle :: \mathtt{s}}\,(BA)$$

We then expand the result as follows.

$$\begin{aligned}
& \mathtt{m}^{\uparrow^{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow^{\mathbf{D}}} \triangleright^* (\mathtt{the} \triangleright^* \mathtt{girl}) \\
= \ & \mathtt{m}^{\uparrow^{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow^{\mathbf{D}}} \triangleright^* (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{girl} w (xgwg)\}) = \mathtt{true} \Vdash_x x))) \\
= \ & \mathtt{m}^{\uparrow^{\mathbf{D}}} \triangleleft^* (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{girl} w (xgwg)\}) = \mathtt{true} \\
& \qquad \Vdash_x (\lambda y, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w (xgwg)(ygwg)e)\}))) \\
= \ & (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{girl} w (xgwg)\}) = \mathtt{true} \\
& \qquad \Vdash_x (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w (xgwg)\mathbf{m}e)\})))
\end{aligned}$$

3.2.3 Taking stock

In the next section, we will look at some real dynamic semantic phenomena, starting with discourse update. But let's, for a moment, take stock of where we are by codifying our progress in some inference rules. Our current stash of rules is given in Figure 3.1. Note that, as far as the monad is concerned,

$$\frac{\Gamma \vdash a : \alpha}{\Gamma \vdash a^{\uparrow\mathbf{D}} : \mathbf{D}\epsilon\alpha}\uparrow \qquad \frac{\Gamma \vdash m : \mathbf{D}e(\mathbf{D}f\alpha)}{\Gamma \vdash \mu_{\mathbf{D}e,f}m : \mathbf{D}(e+f)\alpha}\mu$$

Figure 3.1: Dynamic inference rules (first take)

λ-term	Shorthand	Type
$(\lambda g, w, g'.\mathbf{e})$	e	E
$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{smoke}w(xgwg)e)\})$	smoked	$E \to T$
$(\lambda g, w, g'.\mathbf{smkg}w)$	smkg	E
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{healthy}(xgwg)\})$	healthy	$E \to T$

Table 3.3: Abbreviations

these inference rules are exactly the same as those of the last chapter! All that has changed is that we have sophisticated the monad slightly to accommodate a more complex array of phenomena. Importantly, we have done all of this without affecting ordinary semantic values in any way.

3.3 Discourse update

Let us first consider a simple discourse.

(28) Emily smoked. Smoking is healthy.

Our goal is to update the meaning of the first sentence with that of the second. Within our current grammar, we can derive these sentences as follows (given the abbreviations in Table 3.3, where **smoke** : $s \to e \to v \to t$, **smkg** : $s \to e$, and **healthy** : $s \to e \to t$).

$$\frac{\langle \textit{Emily}, \mathtt{e}\rangle :: \mathtt{d} \qquad \langle \textit{smoked}, \mathtt{smoked}\rangle :: \mathtt{d\backslash s}}{\langle \textit{Emily smoked}, \mathtt{smokede}\rangle :: \mathtt{s}}\,(BA)$$

$$\frac{\langle \textit{smoking}, \mathtt{smkg}\rangle :: \mathtt{d} \qquad \langle \textit{healthy}, \mathtt{healthy}\rangle :: \mathtt{d\backslash s}}{\langle \textit{smoking is healthy}, \mathtt{healthy}\rangle :: \mathtt{s}}\,(BA)$$

In order to define a rule for discourse update, let's first define two things. First, let's have a function ($\rightrightarrows$) for relating information states by a dynamic version of implication (see Groenendijk and Stokhof, 1991; Dekker, 1994,

i.a., for equivalently defined variants). The role of such a relation is to nullify the dynamic contribution of any indefinites, except those contributed by the antecedent information state to the evaluation of the consequent information state.

Definition 14 ($\rightrightarrows$)**.**

$$(\rightrightarrows) : T \to T \to T$$
$$(\rightrightarrows) \coloneqq (\lambda\phi, \psi, g.\{\langle w, g\rangle \mid (\forall g'.\phi g w g' \to (\exists g''.\psi g' w g''))\})$$

Second, let's define a graded monadic operator ($\gg\!=_E$) (whose name we pronounce 'bind'). When defining ordinary monads, ($\gg\!=_{\mathbf{M}}$) and $(\cdot)^{\overset{\uparrow}{\mathbf{M}}}$, along with **M**, are together sufficient; in fact, they constitute the typical rendering of monads in programming languages like Haskell. Within our graded monadic setting, we give to ($\gg\!=_{\mathbf{E}}$), which takes two effects, the following type signature and definition, where **E** is the type constructor associated with **E**.

Definition 15 ($\gg\!=_{\mathbf{E}}$)**.**

$$(\gg\!=_{\mathbf{E}e,f}) : \mathbf{E}e\alpha \to (\alpha \to \mathbf{E}f\beta) \to \mathbf{E}(e+f)\beta$$
$$m \gg\!=_{\mathbf{E}e,f} k = \mu_{\mathbf{E}e,f}(k^{\overset{\uparrow}{\mathbf{E}}\overset{E}{\downarrow}}_{e,e} m)$$

We can then define discourse update by the following rule, which makes use of $\rightrightarrows$ and $\gg\!=_{\mathbf{P}}$; i.e., the bind of the graded monad **P**.

$$(;) : \mathbf{D}eT \to \mathbf{D}fT \to \mathbf{D}(e+f)T$$
$$\phi \;;\; \psi \coloneqq \phi \gg\!=_{\mathbf{D}e,f} (\lambda p, i, c.\psi i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}f,e} (\lambda q.(p+q)^{\overset{\uparrow}{\mathbf{P}}}))$$

In other words, to update ϕ with ψ, we bind to ϕ a function that takes an information state p, a register i, and an input context c, runs ψ on i and the context consisting of c composed with the function $(\lambda t.p \rightrightarrows t)$ and binds q to the result; and then, finally, returns p updated with q qua information states. The purpose of composing the context with $(\lambda t.p \rightrightarrows t)$ is to allow the information state contributed by the antecedent discourse ϕ to weaken the presuppositions of the discourse ψ with which it is updated. In particular, if the first discourse entails the presuppositions of the second, the presuppositions of the result will be weakened to $\mathtt{true} = \mathtt{true}$, i.e., $\top$. Let's, therefore, change the meaning assigned to the lexical item ; to reflect this new commitment to the effect of discourse update.

$$\langle ;, (\, ; \,) \rangle :: \mathtt{s} \backslash (\mathtt{s} / \mathtt{s})$$

We may then see what it it gets us for (28).

$$\dfrac{\dfrac{\dfrac{\langle \textit{Emily smoked}, \mathtt{smokede} \rangle :: \mathtt{s}}{\langle \textit{Emily smoked}, (\mathtt{smokede})^{\uparrow \mathbf{D}} \rangle :: \mathtt{s}} \uparrow \quad \langle ;, (\, ; \,) \rangle :: \mathtt{s} \backslash (\mathtt{s} / \mathtt{s})}{\langle \textit{Emily smoked} \; ;, (\lambda \psi . (\mathtt{smokede})^{\uparrow \mathbf{D}} \; ; \psi) \rangle :: \mathtt{s} / \mathtt{s}} (BA) \quad \dfrac{\dfrac{\langle \textit{smoking is healthy}, \mathtt{healthysmkg} \rangle :: \mathtt{s}}{\langle \textit{smoking is healthy}, (\mathtt{healthysmkg})^{\uparrow \mathbf{D}} \rangle :: \mathtt{s}} \uparrow}{\langle \textit{Emily smoked} \; ; \; \textit{smoking is healthy}, (\mathtt{smokede})^{\uparrow \mathbf{D}} \; ; (\mathtt{healthysmkg})^{\uparrow \mathbf{D}} \rangle :: \mathtt{s}} (FA)$$

The result, in this case, is the same as that of simply doing information state update on the values associated with the meanings of the two sentences, and then injecting the result into **D**. Thus we have the following equivalence.

$$\begin{aligned} & (\mathtt{smokede})^{\uparrow \mathbf{D}} \; ; (\mathtt{healthysmkg})^{\uparrow \mathbf{D}} \\ = \; & ((\mathtt{smokede}) + (\mathtt{healthysmkg}))^{\uparrow \mathbf{D}} \end{aligned}$$

This result holds precisely because two sentences' side effects are trivial, however: updating the context for evaluation of the meaning of the second sentence with the value contributed by the meaning of the first makes no difference, as the second sentence lacks presuppositions, and, hence, must be injected into **D** via $(\cdot)^{\uparrow \mathbf{D}}$, causing the argument representing its local context to simply be thrown out. We can state this result as a more general fact, proved in Appendix B.

Fact 2.

$$\phi^{\uparrow \mathbf{D}} \; ; \; \psi^{\uparrow \mathbf{D}} = (\phi + \psi)^{\uparrow \mathbf{D}}$$

This result is by design: we want discourse update to mimic information state update when it handles sentences that lack presuppositions.

Let's now turn to some more complex examples, starting with basic instances of anaphora.

3.3.1 Adding anaphora

Let's first consider an analysis of pronoun meaning, given the current proposal. Take a sentence like (29).

λ-term	Shorthand	Type
$(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{masc}w(xgwg)\}) = \mathtt{true} \Vdash_x x))$	he	E
$(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{fem}w(xgwg)\}) = \mathtt{true} \Vdash_x x))$	she	E
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{exist}w(xgwg)\})$	exists	$E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{nice}w(xgwg)\})$	nice	$E \to T$

Table 3.4: Abbreviations

(29) Mary exists. She is nice.

Our setup suggests an analysis of pronouns as follows, where *he* and *she* have gender presuppositions.

$$\langle he, (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{masc}w(xgwg)\}) = \mathtt{true} \Vdash_x x))\rangle :: \mathtt{d}$$
$$\langle she, (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{fem}w(xgwg)\}) = \mathtt{true} \Vdash_x x))\rangle :: \mathtt{d}$$

Therefore, if we give an analysis to (29) using the meaning for discourse update, we can obtain the following meaning (where we use the abbreviations defined in Table 3.4).

$$\dfrac{\dfrac{\dfrac{\langle \mathit{Mary}, \mathtt{m}\rangle :: \mathtt{d} \qquad \langle \mathit{exists}, \mathtt{exists}\rangle :: \mathtt{d}\backslash\mathtt{s}}{\langle \mathit{Mary\ exists}, \mathtt{existsm}\rangle :: \mathtt{s}}\,(BA)}{\langle \mathit{Mary\ exists}, (\mathtt{existsm})^{\uparrow\mathtt{D}}\rangle :: \mathtt{s} \quad \langle ;, (\,;)\rangle :: \mathtt{s}\backslash(\mathtt{s}/\mathtt{s})}\,\uparrow \quad (BA) \qquad \dfrac{\dfrac{\langle \mathit{she}, \mathtt{she}\rangle :: \mathtt{d} \qquad \dfrac{\langle \mathit{nice}, \mathtt{nice}\rangle :: \mathtt{d}\backslash\mathtt{s}}{\langle \mathit{nice}, \mathtt{nice}^{\uparrow\mathtt{D}}\rangle :: \mathtt{d}\backslash\mathtt{s}}\,\uparrow}{\langle \mathit{she\ is\ nice}, \mathtt{she} \triangleleft^* \mathtt{nice}^{\uparrow\mathtt{D}}\rangle :: \mathtt{s}}\,(BA)}{\langle \mathit{Mary\ exists\ ;\ she\ is\ nice}, (\mathtt{existsm})^{\uparrow\mathtt{D}}\,;(\mathtt{she} \triangleleft^* \mathtt{nice}^{\uparrow\mathtt{D}})\rangle :: \mathtt{s}}\,(FA)$$

If we replace each abbrevation with its definition and β-reduce, we obtain the following result.

$$\begin{aligned} & (\mathtt{existsm})^{\uparrow\mathtt{D}}\,;(\mathtt{she} \triangleleft^* \mathtt{nice}^{\uparrow\mathtt{D}}) \\ = \ & (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \to \mathbf{fem}w(xgwg)\}) = \mathtt{true} \\ & \qquad \Vdash_x (\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \wedge \mathbf{nice}w(xgwg)\}))) \end{aligned}$$

According to this analysis, (29) registers a presupposition: that if Mary exists, then the referent of the pronoun *she* enjoys whatever constraints are imposed on it by the pronoun's gender. There are two things to say about this result. The first is that it makes clear that we need a mechanism for accomplishing anaphora resolution. Ideally, we would like to have x in the

above term substituted for $\mathtt{m}$ on an analysis of (29) in which *Mary* and the pronoun corefer. The second is that, given such a substitution, the resulting meaning doesn't seem quite correct. If anything, what (29) seems to presuppose is that Mary is female, not that she is female if she exists. We'll look at a mechanism for handling anaphora resolution now. The second issue gets deferred until chapter 4, where we look at the current framework's solution to the proviso problem.

Recall that the effect registered by the meaning of an expression with a single missing (dynamically lifted) individual is $\{E\}$. Thus the term translating (29) has the following type.

$$i \to (T \to T) \to \mathbf{P}\{E\}T$$

What we would like to do, intuitively, is plug some function—in particular, the constant one $(\lambda g, w, g'.\mathbf{m})$—in for the effect registered by the interpretation of (29) in order to get back the type

$$i \to (T \to T) \to \mathbf{P}\epsilon T$$

which would be that of the term

$$\begin{aligned}(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \to \mathbf{fem}w\mathbf{m}\}) = \mathtt{true}\\ \Vdash (\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \wedge \mathbf{nice}w\mathbf{m}\})))\end{aligned}$$

To do so, we introduce two families of functions, anaph_n, indexed by a "position" in the effect associated with an expression with presuppositions. To define anaph_n, we first define a family of functions $\mathbf{insert}_n : \mathcal{T} \to \mathcal{T}^* \to \mathcal{T}$, each member of which returns a type.

$$\begin{aligned}\mathbf{insert}_0 ae &= ae\\ \mathbf{insert}_{n+1} a(be) &= b(\mathbf{insert}_n ae)\end{aligned}$$

Members of the family of functions anaph_n take a type and an effect in order to return a term-level function.

Definition 16 (anaph_n)**.**

$$\begin{aligned}\mathsf{anaph}_n ae &: a \to \mathbf{P}(\mathbf{insert}_n ae)\alpha \to \mathbf{P}e\alpha\\ \mathsf{anaph}_0 ae &= (\lambda x^a, m^{a \to \mathbf{P}e\alpha}.mx)\\ \mathsf{anaph}_{n+1} a(be) &= (\lambda x^a, m^{b \to \mathbf{P}(\mathbf{insert}_n ae)\alpha}, y^b.\mathsf{anaph}_n aex(my))\end{aligned}$$

$$\frac{\Gamma \vdash m : \mathbf{D}(\mathbf{insert}_n ae)\alpha \qquad \Gamma \vdash t : a}{\Gamma \vdash \mathsf{anaph}_n aetm : \mathbf{D}e\alpha}\ Anaph$$

Figure 3.2: The anaphora rule

In order to simplify the presentation, we allow the term $\mathsf{anaph}_n ae$ to be ambiguous between the $(\mathbf{P}(\mathbf{insert}_n ae)\alpha \to \mathbf{P}e\alpha)$-type function just defined and its corresponding $(\mathbf{D}(\mathbf{insert}_n ae)\alpha \to \mathbf{D}e\alpha)$-type variant; i.e.,

$$\mathsf{anaph}_n ae \coloneqq (\lambda m, i, c.\mathsf{anaph}_n ae(mic))$$

These additions are all we need in order to accomplish a form of anaphora resolution within our framework. To our dynamic inference rules, we add the rule *Anaph*, presented in Figure 3.2. Note that the second variant of anaph_n is used in the definition of this rule. Given a type a and an effect e, anaph_n takes some value of type a, along with a meaning registering an effect whose n^{th} position is typed a, and rids the effect of that position by (so to speak) applying it to the value.

We now see how this rule works for the case of (29). Referring to the anaphora rule, we can plug $\mathtt{m}$ in to the meaning of (29), thus finishing the derivation from above.

$$\frac{(\mathtt{existsm})^{\uparrow_{\mathbf{D}}};(\mathtt{she} \blacktriangleleft^* \mathtt{nice}^{\uparrow_{\mathbf{D}}})}{\mathsf{anaph}_0 E\epsilon\mathtt{m}((\mathtt{existsm})^{\uparrow_{\mathbf{D}}};(\mathtt{she} \blacktriangleleft^* \mathtt{nice}^{\uparrow_{\mathbf{D}}}))}\ Anaph$$

$$\begin{aligned} & \mathsf{anaph}_0 E\epsilon\mathtt{m}((\mathtt{existsm})^{\uparrow_{\mathbf{D}}};(\mathtt{she} \blacktriangleleft^* \mathtt{nice}^{\uparrow_{\mathbf{D}}})) \\ = \ & (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \to \mathbf{fem}w\mathbf{m}\}) = \mathtt{true} \\ & \qquad \Vdash (\lambda g.\{\langle w, g\rangle \mid \mathbf{exists}w\mathbf{m} \wedge \mathbf{nice}w\mathbf{m}\}))) \end{aligned}$$

The result is exactly the meaning we sought out (modulo the strange presupposition). We now slightly refine our semantic analysis of indefinites, which, as we will see, dovetails with the analysis of anaphora just proposed.

3.4 Indefinites

3.4.1 The semantic contribution of indefinites

Recall the meaning proposed earlier in this chapter for the indefinite article indexed with register i.

$$(\lambda P, Q, g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g' \wedge (P(\lambda g, w, g'.g_i) + Q(\lambda g, w, g'.g_i))g'wg'')\})$$

As we saw, this meaning works well for cases in which the indefinite argument of the verb is a subject, and in which it obligatorily receives narrow scope. What we would like is, on the one hand, for an indefinite argument to able to occur in any position, whether it be subject position, object position, or an oblique position, and, on the other hand, for its interpretational possibilities to be scopally free. To handle both of these features of indefiniteness, we will add a continuation-passing style of semantic composition to the current framework. Continuations were first introduced as a mechanism for handling natural language quantification in semantics in Barker 2002 and have recently been explored more fully in Barker and Shan 2014 and Charlow 2014. I'll use them for narrow purposes in this dissertation; in particular, to analyze the two features of indefinites just mentioned.

To add a continuation-passing style of composition to the analysis, we can use a graded continuation monad transformer. Given our present graded monad, we transform it into one which allows expressions to potentially introduce, as their meanings, functions on their continuations. Given a result type r for the continuation, the definition of this graded monad transformer (**CT**) is as follows, where $\mathcal{E}$ is any monoid of parameters.

Definition 17 (**CT**)**.**

$$\begin{aligned}
\mathbf{CT} &: \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T} \\
\mathbf{CT}(r)(\mathbf{E_0})e\alpha &= (\alpha \to \mathbf{E_0}fr) \to \mathbf{E}_0(e+f)r \\
a^{\overset{\uparrow}{\mathbf{CT}(\mathbf{E_0})}} &= (\lambda k.ka) \\
u^{\mathbf{CT}(\mathbf{E_0})}_{\overset{\downarrow}{e,f}} &= (\lambda v, k.u(\lambda U.v(\lambda V.k(UV)))) \\
\mu_{\mathbf{CT}(\mathbf{E_0})_{e,f}}m &= (\lambda k.m(\lambda k'.k'k))
\end{aligned}$$

The result of applying **CT** to a graded monad is also guaranteed to be a graded monad (as proved in Appendix B).

Theorem 3. If $\mathbf{E_0}$ is a graded monad with monoid of parameters $\mathcal{E}$, then so is $\mathbf{CT}(r)(\mathbf{E_0})$, for any type r.

We may then introduce a new graded monad **K**, defined as $\mathbf{CT}(T)(\mathbf{D})$.

Definition 18 (K).

$$\begin{aligned}
\mathbf{K} &: \mathcal{T}^* \to \mathcal{T} \to \mathcal{T} \\
\mathbf{K}e\alpha &= (\alpha \to \mathbf{D}f\,T) \to \mathbf{D}(e+f)T \\
a^{\overset{\uparrow}{\mathbf{K}}} &= (\lambda k.ka) \\
u^{\overset{\mathbf{K}}{\downarrow}}_{e,f} &= (\lambda v, k.u(\lambda U.v(\lambda V.k(U\,V)))) \\
\mu_{\mathbf{K}e,f}\,m &= (\lambda k.m(\lambda k'.k'k))
\end{aligned}$$

Let's illustrate how the this graded monad works with an analysis of the sentence in (30). The inference rules we use are now to be understood according to the graded monad **K**, rather than **D**.

(30) Mary saw a girl.

We can assign the expressions *Mary*, *saw*, and *girl* the same lexical entries we assigned to them above. To the indefinite, which no longer needs to be associated with an index, and which now receives the same syntactic category as any other determiner, we assign the following lexical entry with a meaning within the monad **K**.

$$\langle a, \mathsf{a} \rangle :: \mathsf{d/n}$$

where we define the abbreviation a as follows.

$$\begin{aligned}
&\mathsf{a} : (E \to T) \to \mathbf{K}e E \\
&\mathsf{a} := (\lambda P, k, i.(\\
&\qquad (\lambda g.\{\langle w, g'' \rangle \mid (\exists g'.g[i]g' \wedge P(\lambda g, w, g'.g_i)g'wg'')\})^{\overset{\uparrow}{\mathbf{D}}} \\
&\qquad\quad ; k(\lambda g, w, g'.g_i))i')
\end{aligned}$$

Thus the meaning of an indefinite noun phrase reads in a register and introduces an information state whose only role is to non-deterministically update the incoming register with an individual satisfying the noun phrase's restriction. It then returns this information state in **D**, updates the result with the indefinite's continuation fed whichever individual occupies the relevant register at the time the continuation is reified, and increments the register by 1 for the next indefinite article meaning appearing in its scope. The following derivation of (30) illustrates the meaning of the indefinite article in action.

$$\dfrac{\dfrac{\langle \mathit{Mary}, \mathtt{m}\rangle :: \mathtt{d}}{\langle \mathit{Mary}, \mathtt{m}^{\uparrow\mathbf{K}}\rangle :: \mathtt{d}}\uparrow \quad \dfrac{\dfrac{\langle \mathit{saw}, \mathtt{saw}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}}{\langle \mathit{saw}, \mathtt{saw}^{\uparrow\mathbf{K}}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}}\uparrow \quad \dfrac{\langle a, \mathtt{a}\rangle :: \mathtt{d}/\mathtt{n} \qquad \langle \mathit{girl}, \mathtt{girl}\rangle :: \mathtt{n}}{\langle \mathit{a\ girl}, \mathtt{agirl}\rangle :: \mathtt{d}}(FA)}{\langle \mathit{saw\ a\ girl}, \mathtt{saw}^{\uparrow\mathbf{K}} \triangleright^* (\mathtt{agirl})\rangle :: \mathtt{d}\backslash\mathtt{s}}(FA)}{\langle \mathit{Mary\ saw\ a\ girl}, \mathtt{m}^{\uparrow\mathbf{K}} \triangleleft^* (\mathtt{saw}^{\uparrow\mathbf{K}} \triangleright^* (\mathtt{agirl}))\rangle :: \mathtt{s}}(BA)$$

If we expand and β-reduce, we obtain the following result.

$$\begin{aligned}
&\mathtt{m}^{\uparrow\mathbf{K}} \triangleleft^* (\mathtt{saw}^{\uparrow\mathbf{K}} \triangleright^* (\mathtt{agirl}))\\
=\ &(\lambda k.(\lambda i.((\lambda g.\{\langle w, g'\rangle | g[i]g' \wedge \mathbf{girl} w g'_i\})^{\uparrow\mathbf{D}}; k(\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w g_i \mathbf{m} e)\}))i'))
\end{aligned}$$

Since we have been composing values within the graded continuation monad $\mathbf{K}$, our result is of type $\mathbf{K}\epsilon T$. In general, what we would like, instead, is an ordinary information state—something of type $\mathbf{D}\epsilon T$—possibly with presuppositions; since the sentence in (30) lacks presuppositions, such an information state ought to simply be returned in the monad $\mathbf{D}$ in the present case. To obtain such a meaning, we feed our current meaning $(\cdot)^{\uparrow\mathbf{D}}$ as its continuation argument, giving us the following result.

$$\begin{aligned}
&(\lambda k.(\lambda i.((\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i\})^{\uparrow\mathbf{D}}\\
&\qquad ; k(\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w g_i \mathbf{m} e)\}))i'))(\cdot)^{\uparrow\mathbf{D}}\\
=\ &(\lambda i.((\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i\})^{\uparrow\mathbf{D}}; (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{see} w g_i \mathbf{m} e)\})^{\uparrow\mathbf{D}})i)\\
=\ &(\lambda i.(\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{see} w g'_i \mathbf{m} e)\})^{\uparrow\mathbf{D}} i)
\end{aligned}$$

Thus we have a function which reads in a register and gives back (returned in $\mathbf{D}$ and then applied to the next register) an information state which transitions from one assignment to another across a world just in case the latter assignment is like the former, yet it fills the next register with a girl that Mary saw in the world.

3.4.2 Anaphora to indefinites

We are now in a position to consider a treatment of discourses like the one exemplified at the beginning of this chapter. Here is (26), again.

λ-term	Shorthand	Type
$(\lambda P,k,i.((\lambda g.\{\langle w,g''\rangle \mid (\exists g'.g[i]g' \wedge P(\lambda g,w,g'.g_i)g'wg'')\})^{\uparrow_{\mathbf{D}}}; k(\lambda g,w,g'.g_i))i')$	a	$(E \to T) \to \mathbf{K}\epsilon E$
$(\lambda x,g.\{\langle w,g\rangle \mid \mathbf{girl}w(xgwg)\})$	girl	$E \to T$
$(\lambda x,g.\{\langle w,g\rangle \mid (\exists e.\mathbf{walk}w(xgwg)e)\})$	walked_in	$E \to T$
$(\lambda g,w,g'.g_i)$	$\mathtt{pro}_i$	E

Table 3.5: Abbreviations

(26) A girl walked in. Mary saw the girl.

To analyze (26), we will treat what appears to be cross-sentential binding of the definite description by the indefinite as an instance of anaphora.

Recall from above that the second sentence of this discourse is assigned the following meaning.

$$(\lambda i,c.(c(\lambda g.\{\langle w,g\rangle \mid \mathbf{girl}w(xgwg)\}) = \mathtt{true} \Vdash_x (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{see}w(xgwg)\mathbf{m}e)\})))$$

We would like to be able to use the anaphora rule to make the definite description anaphoric to the indefinite noun phrase that precedes it in (26). In the present case, the term we plug into the abstraction introduced by the definite description is

$$(\lambda g,w,g'.g_0)$$

whose type is E (and which we will call $\mathtt{pro}_0$, following the convention shown in Table 3.5). If we perform anaphora on this term and the term translating the second sentence of (26), we get the following result.

$$(\lambda i,c.(c(\lambda g.\{\langle w,g\rangle \mid \mathbf{girl}wg_0\}) = \mathtt{true} \Vdash (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{see}wg_0\mathbf{m}e)\})))$$

Let's now analyze the first sentence of (26). As for (30), the inference rules we use are upgraded in order to involve the graded monad **K**. The full set of abbreviations used is given in Table 3.5.

$$\dfrac{\dfrac{\langle a,\mathtt{a}\rangle :: \mathtt{d/n} \qquad \langle girl,\mathtt{girl}\rangle :: \mathtt{n}}{\langle a\ girl,\mathtt{agirl}\rangle :: \mathtt{d}}(FA) \qquad \dfrac{\langle walked\ in,\mathtt{walked\_in}\rangle :: \mathtt{d\backslash s}}{\langle walked\ in,\mathtt{walked\_in}^{\uparrow_{\mathbf{K}}}\rangle :: \mathtt{d\backslash s}}\uparrow}{\langle a\ girl\ walked\ in,(\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}}\rangle :: \mathtt{s}}(BA)$$

$$(\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}}$$

$$= (\lambda k,i.((\lambda g.\{\langle w,g'\rangle \mid g[i]g' \wedge \mathbf{girl}wg'_i\})^{\uparrow_{\mathbf{D}}}; k(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{walk}wg_ie)\}))i')$$

As for the previous example, we ought to feed the resulting meaning the trivial continuation $(\cdot)^{\uparrow\mathbf{D}}$.

$$
\begin{aligned}
& (\lambda k, i.((\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i\})^{\uparrow\mathbf{D}}; k(\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{walk} w g_i e)\}))i')(\cdot)^{\uparrow\mathbf{D}} \\
= \; & (\lambda i.((\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i\})^{\uparrow\mathbf{D}}; (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{walk} w g_i e)\})^{\uparrow\mathbf{D}})i) \\
= \; & (\lambda i.(\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e)\})^{\uparrow\mathbf{D}} i)
\end{aligned}
$$

If we then update the meaning of the first sentence of (26) with that of the second after it has undergone anaphora, we obtain the following result.

$$\dfrac{\dfrac{\langle \textit{a girl walked in}, ((\texttt{agirl}) \blacktriangleleft^{*} \texttt{walked\_in}^{\uparrow_{\mathsf{K}}})(\cdot)^{\uparrow_{\mathsf{D}}} \rangle :: \mathsf{s} \qquad \langle ;, (\,;\,) \rangle :: (\mathsf{s}\backslash\mathsf{s})/\mathsf{s}}{\langle \textit{a girl walked in} ;, (\lambda\psi.((\texttt{agirl}) \blacktriangleleft^{*} \texttt{walked\_in}^{\uparrow_{\mathsf{K}}})(\cdot)^{\uparrow_{\mathsf{D}}} ; \psi) \rangle :: \mathsf{s}/\mathsf{s}}\,(BA) \qquad \dfrac{\langle \textit{Mary saw the girl}, \mathtt{m}^{\uparrow_{\mathsf{D}}} \blacktriangleleft^{*} (\mathtt{saw}^{\uparrow_{\mathsf{D}}} \blacktriangleright^{*} (\mathtt{thegirl})) \rangle :: \mathsf{s}}{\langle \textit{Mary saw the girl}, \mathsf{anaph}_0 E \epsilon \mathtt{pro}_0 (\mathtt{m}^{\uparrow_{\mathsf{D}}} \blacktriangleleft^{*} (\mathtt{saw}^{\uparrow_{\mathsf{D}}} \blacktriangleright^{*} (\mathtt{thegirl}))) \rangle :: \mathsf{s}}\,\textit{Anaph}}{\langle \textit{a girl walked in ; Mary saw the girl}, ((\texttt{agirl}) \blacktriangleleft^{*} \texttt{walked\_in}^{\uparrow_{\mathsf{K}}})(\cdot)^{\uparrow_{\mathsf{D}}} ; (\mathsf{anaph}_0 E \epsilon \mathtt{pro}_0 (\mathtt{m}^{\uparrow_{\mathsf{D}}} \blacktriangleleft^{*} \mathtt{saw}^{\uparrow_{\mathsf{D}}} \blacktriangleright^{*} (\mathtt{the} \blacktriangleright^{*} \mathtt{girl})))) \rangle :: \mathsf{s}}\,(FA)$$

If we expand the result, we obtain the following term.

$$\begin{aligned}
&((\mathtt{a girl}) \triangleleft^* \mathtt{walked_in}^{\uparrow_{\mathsf{K}}})(\cdot)^{\uparrow_{\mathsf{D}}}; (\mathrm{anaph}_0 E \epsilon \mathtt{pro}_0 (\mathtt{m}^{\uparrow_{\mathsf{D}}} \triangleleft^* \mathtt{saw}^{\uparrow_{\mathsf{D}}} \triangleright^* (\mathtt{the} \triangleright^* \mathtt{girl})))\\
=\ &(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e)) \rightarrow \mathbf{girl} w g'_0)\}) = \mathtt{true}\\
&\qquad \Vdash (\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e) \wedge (\exists e.\mathbf{see} w g'_0 \mathbf{m} e)\})))
\end{aligned}$$

Crucially, the rule for discourse update weakened the presuppositions of the second sentence of the discourse to have them depend on the at-issue content of the first sentence. Since the first sentence entails the presuppositions of the second, the presuppositions of the entire discourse have been trivialized to `true`. In particular, the meaning of ; ensures that, given an input context c and some index i, the presupposition of the entire discourse is

$$\begin{aligned}
&c((\lambda g.\{\langle w, g'\rangle \mid g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e)\}) \rightrightarrows (\lambda g.\{\langle w, g\rangle \mid \mathbf{girl} w g_0\})) = \mathtt{true}\\
=\ &c(\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e)) \rightarrow (\exists g''.g' = g'' \wedge \mathbf{girl} w g'_0))\}) = \mathtt{true}\\
=\ &c(\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[i]g' \wedge \mathbf{girl} w g'_i \wedge (\exists e.\mathbf{walk} w g'_i e)) \rightarrow \mathbf{girl} w g'_0)\}) = \mathtt{true}
\end{aligned}$$

At this point, if we wish, we can decide that the discourse is presupposition-free and reset it by feeding our context-changer argument the trivial context-changer: the identity function. Part of this resetting mechanism will also be to value the index associated the indefinite, so as to satisfy the presupposition of the sentence which introduced the definite. We define an abbreviation `reset` to accomplish both tasks.[1]

Definition 19 (`reset`)**.**

$$\begin{aligned}
\mathtt{reset} &: \mathbf{D}e\alpha \rightarrow \mathbf{D}e\alpha\\
\mathtt{reset}\, m &:= (\lambda i, c.(m\, 0\, \mathtt{id}))
\end{aligned}$$

For lack of space, I leave it to the reader to check that running `reset` on the discourse above yields the following result.

$$(\lambda g.\{\langle w, g'\rangle \mid g[0]g', \mathbf{girl} w g'_0 \wedge (\exists e.\mathbf{walk} w g'_0 e) \wedge (\exists e.\mathbf{see} w g'_0 \mathbf{m} e)\})^{\uparrow_{\mathbf{D}}}$$

Thus at the end of the discourse, we have just returned the information state in which a girl walked in and Mary saw the girl in the graded monad **D**. Correctly, we predict that the discourse in (26), which is analyzed as a mere returned information state, does not have presuppositions.

[1] To define `reset`, I introduce an operator $(+) : i \rightarrow i \rightarrow i$ whose behavior is axiomatized in the obvious way; i.e.,

$$\begin{aligned}
\mathbf{i} + m &= m\\
n' + m &= (n + m)'
\end{aligned}$$

3.5 Conclusions

We've now gotten a handle on some basic scaffolding for a dynamic account of indefinite reference, anaphora, and discourse-related presupposition satisfaction. We have an account that can be summarized, more or less, in terms of the following principles.

- An expression's presuppositions are stated relative to that expression's local context, which is a function from information states to information states. This enrichment is given by the graded monad **D**.
- Discourse update updates the local context to which an expression's presuppositions are relativized so that these presuppositions are weakened to depend on prior discourse.
- Anaphora is accomplished by changing the type of an expression with presuppositions, thus shrinking that expression's side effects.
- Indefinite reference is contributed by relations between assignments of individuals to registers. Anaphora to indefinites is implemented in terms of anaphora to some register.

Not to mention that we have also transformed the graded monad **D** into its continuized variant **K** in order to account for the syntactic distribution of indefinites. As we will see in the next chapter, the addition of continuations to the compositional repertoire allows us to give a treatment of the exceptional scoping properties of indefinites. Consider the following example to illustrate.

(31) Mary believes that one of her friends is outside.

This sentence can be understood in one of two ways. Either it is the case that Mary's doxastically accessible worlds are worlds in which one of her friends is outside; or one of Mary's friends is such that Mary's doxastically accessible worlds are worlds in which that friend is outside. Which reading is manifest depends on where the indefinite *one of her friends* takes scope.

Before moving on, it is worth recognizing and correcting the fact that we've been a bit fast and loose with the use of inference rules in semantic derivations without first codifying them in any way. The derivations above have instances of $\uparrow$ and $\downarrow$ in both the graded monads **D** and **K**, for example. So, let's codify them now.

First, we'd like to be able to manage presuppositions within both the graded monads **D** and **K** and, moreover, to be able to transition between them when appropriate. Let's, therefore, define two operations, $(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$, the latter of which is simply the continuation-feeding operation used above.

Definition 20 ($(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$)**.**

$$
\begin{aligned}
&(\cdot)^{\Uparrow} : \mathbf{D}e\alpha \to \mathbf{K}e\alpha \\
&m^{\Uparrow} = (\lambda k.m \gg\!=_{\mathbf{D}e,f} k) \\
&(\cdot)^{\Downarrow} : \mathbf{K}eT \to \mathbf{D}eT \\
&m^{\Downarrow} = m(\lambda x.x^{\overset{\uparrow}{\mathbf{D}}})
\end{aligned}
$$

By inspecting the types, one sees that $(\cdot)^{\Uparrow}$ lifts an object in the graded monad **D** into one in the graded monad **K**, which it does by binding the object to a continuation, over which it then abstracts. $(\cdot)^{\Downarrow}$ does the opposite: it feeds an object in the graded monad **K** waiting for its continuation the trivial continuation $(\cdot)^{\overset{\uparrow}{\mathbf{D}}}$. Indeed, $(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$ allow one to view an object in **D** in two different ways: either in **D** or as its continuized variant in **K**.[2]

Second, as we will see in the next chapter, we would like to be able to apply our monadic operators ($(\cdot)^{\overset{\uparrow}{\mathbf{D}}}$, $(\cdot)^{\overset{\mathbf{D}}{\downarrow}}_{e,f}$, $\mu_{\mathbf{D}e,f}$, $(\cdot)^{\overset{\uparrow}{\mathbf{K}}}$, $(\cdot)^{\overset{\mathbf{K}}{\downarrow}}_{e,f}$, and $\mu_{\mathbf{K}e,f}$) when- and wherever it seems useful. For example, perhaps, we'll have some term

$$(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_n} a))$$

whose side effects we'd like to remain unaffected by some meaning designed to affect side effects (take ;, for instance). Let's call this meaning *op*. In that case, we could apply *op* to our meaning above, in order to get exactly what we don't want.

$$op(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_n} a)) = \text{oops!}$$

In such a case, we would like to be able to trick *op* by getting the side effects out of the way, somehow. If only we could do

$$op^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}}_{\epsilon,\alpha_1\ldots\alpha_n}(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_n} a^{\overset{\uparrow}{\mathbf{D}}})) = (\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_n} opa^{\overset{\uparrow}{\mathbf{D}}})),$$

[2] Exploiting a continuation monad transformer, equipped with lift and lower operators, is heavily inspired by Charlow 2014, which uses similar mechanisms to study the semantic contributions of distributive quantifiers, like *every*, within a monadic framework designed to handle the non-determinism contributed by indefinites.

instead. Let's allow ourselves access to such operations by defining a set of operations Op.

$$Op ::= (\cdot)^{\overset{\uparrow}{\mathbf{D}}} \mid \mu_{\mathbf{D}_{e,f}} \mid (\cdot)^{\overset{\uparrow}{\mathbf{K}}} \mid \mu_{\mathbf{K}_{e,f}} \mid (\cdot)^{\Uparrow} \mid (\cdot)^{\Downarrow} \mid Op^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}_{e,e}} \mid Op^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}_{e,e}}$$

Finally, we ought to redefine graded monadic forward and backward functional application to accommodate both the monads **D** and **K**. They are now defined as follows.

$$(\triangleright^*) ::= (\triangleright) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}_{e,e}} u)^{\overset{\mathbf{D}}{\downarrow}_{e,f}} v) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}_{e,e}} u)^{\overset{\mathbf{K}}{\downarrow}_{e,f}} v)$$

$$(\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}_{e,e}} u)^{\overset{\mathbf{D}}{\downarrow}_{e,f}} v) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}_{e,e}} u)^{\overset{\mathbf{K}}{\downarrow}_{e,f}} v)$$

Our final set of inference rules is, then, as in Figure 3.3. In general, for some operation o in the graded monad **E** (i.e, either **D** or **K**), we'll ambiguously write $o_{\mathbf{E}_1}$ for either $o_{\mathbf{E}}{}^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}}$ or $o_{\mathbf{E}}{}^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}}$; e.g., $(\cdot)^{\overset{\uparrow}{\mathbf{D}_1}}$ for $(\lambda x.x^{\overset{\uparrow}{\mathbf{D}}})^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}_{e,f}}$, given any two effects e and f. And so on for any number of applications of either $(\cdot)^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}}$ or $(\cdot)^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}}$ to an operation; e.g., $(\cdot)^{\overset{\uparrow}{\mathbf{D}_2}}$ for $(\lambda x.x^{\overset{\uparrow}{\mathbf{D}}})^{\overset{\uparrow}{\mathbf{K}}\overset{\mathbf{K}}{\downarrow}_{g,h}}{}^{\overset{\uparrow}{\mathbf{D}}\overset{\mathbf{D}}{\downarrow}_{e,f}}$, etc. The ability to map operations over meanings with side effects by looking past the side effects will allow presuppositions to escape their local contexts. This feature, then, gives us a handle on problems that typically plague model-theoretic dynamic accounts of presupposition, like the proviso problem.

Enough throat clearing—time for some applications to the phenomena of presupposition.

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{D}_n}} : \ldots \mathbf{D} \epsilon \alpha \ldots} \uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{D} e (\mathbf{D} f \alpha) \ldots}{\Gamma \vdash \mu_{\mathbf{D}_n e,f} m : \ldots \mathbf{D}(e+f)\alpha \ldots} \mu$$

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{K}_n}} : \ldots \mathbf{K} \epsilon \alpha \ldots} \uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K} e (\mathbf{K} f \alpha) \ldots}{\Gamma \vdash \mu_{\mathbf{K}_n e,f} m : \ldots \mathbf{K}(e+f)\alpha \ldots} \mu$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{D} e \alpha \ldots}{\Gamma \vdash m^{\Uparrow_n} : \ldots \mathbf{K} e \alpha \ldots} \Uparrow$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{K} e T \ldots}{\Gamma \vdash m^{\Downarrow_n} : \ldots \mathbf{D} e T \ldots} \Downarrow$$

$$\frac{\Gamma \vdash m : \mathbf{D}(\mathbf{insert}_n a e)\alpha \qquad \Gamma \vdash t : a}{\Gamma \vdash \mathsf{anaph}_n a e t m : \mathbf{D} e \alpha} \mathit{Anaph}$$

Figure 3.3: Dynamic inference rules

Chapter 4

Presupposition triggers as scope-takers

4.1 Introduction

Given the developments in the last chapter, we are now in a position to look at some slightly more complex phenomena. One of the explananda most concerning for an account of presupposition is the pattern that governs presupposition projection. Thus in the next section, we look at the general treatment of filtering within the framework elaborated in the last chapter. We pay specific attention to conditionals and propositional attitude verbs, like *believe*, both of which we see to be associated with two patterns of presupposition projection. One of these patterns exhibits behavior wherein presuppositions triggered in an embedded clause of some sort appear to be registered in the construction as a whole. The second pattern shows up in constructions in which the presupposition is modified, or "filtered", in some way by the expression that embeds it. In both patterns, it is argued that the resulting judgment can explained in terms of where the side effects associated with a particular presupposition trigger have decided to take scope.

In the following sections, we look at other applications of the framework developed above. §4.3 takes a look at the phenomenon of modal subordination, as illustrated by examples like the following.

(32) If a dog walked in, it would be hungry. The dog would eat a potato.

Crucially, in the second sentence in (32), we see anaphora to the indefinite *a dog* which appears in the first sentence, even though this indefinite appears

to take scope inside of a conditional. Thus what (32) seems ot mean is that if a dog walked in, that dog would (a) be hungry, and (b) eat a potato. What we'll see is that the dynamic semantic framework for presupposition developed in chapter 3 makes available a compositional analysis of sentences like (32), given an appropriate semantics for the modal auxiliary *would*.

In chapter 5, we will look at other kinds of complex anaphora within the present framework, including null complement anaphora and ellipsis, in addition to quantification.

4.2 Filtering

4.2.1 Conditionals (naïve version)

The following sentence exemplifies the sort of presupposition filtering behavior that we often see exhibited by conditional sentences.

(33) If John came, Mary knows John came.

What it's crucial to explain about (33) is that the sentence is entirely free of presuppositions, despite the fact that we find the presupposition trigger *knows* inside of its consequent. This presupposition trigger has the effect of registering a presupposition—that John came—as part of the meaning of the consequent. However, because the presupposition is entailed by the antecedent of the conditional, the conditional satisfies the presupposition of the consequent, and the presupposition doesn't project.

To explain both the at-issue meaning contributed by conditional sentences and their filtering behavior, we can define a term, abbreviated $\mathtt{if}$, which we may postulate as the meaning of the connective *if*.

$$\mathtt{if} : T \to \mathbf{D}eT \to \mathbf{D}eT$$
$$\mathtt{if} \coloneqq (\lambda p, \psi, i, c.\psi i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}\epsilon,e} (\lambda q.(p \rightrightarrows q)^{\uparrow_{\mathbf{P}}}))$$

$$\langle \mathit{if}, \mathtt{if} \rangle :: (\mathtt{s}/\mathtt{s})/\mathtt{s}$$

Given this meaning, a conditional sentence is true if the meaning of its antecedent dynamically implies the meaning of its consequent. Moreover, its presuppositions are that meaning of its antecedent dynamically implies the presuppositions of its consequent. This semantics for conditionals is similar to that used in Barker and Shan 2008 and Charlow 2014, which involves directly negating information states. (These semantic analyses, in turn, echo

λ-term	Shorthand	Type
$(\lambda g, w, g'.\mathbf{j})$	j	E
$(\lambda g, w, g'.\mathbf{m})$	m	E
$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{come} w (x g w g) e\}))$	came	$E \to T$
$(\lambda p, x, i, c.(cp = \mathtt{true} \Vdash (\lambda g.\{\langle w, g\rangle \mid \mathbf{believe} w p (x g w) g\})))$	knows	$T \to E \to \mathbf{D}\epsilon T$

Table 4.1: Abbreviations

semantic mechansims used in earlier dynamic theories to account for patterns manifest by indefinites and their potential anaphors.) Let's look at what this meaning gives us for the sentence in (33). First, we assign the expressions involved the following lexical entries.

$$\langle \textit{John}, \mathtt{j}\rangle :: \mathtt{d}$$
$$\langle \textit{Mary}, \mathtt{m}\rangle :: \mathtt{d}$$
$$\langle \textit{came}, \mathtt{came}\rangle :: \mathtt{d}\backslash\mathtt{s}$$
$$\langle \textit{knows}, \mathtt{knows}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{s}$$

Important to note is the meaning we assign to *know*. This meaning, which registers a presupposition, is given by a term of type $T \to E \to \mathbf{D}\epsilon T$. It combines with an information state and a lifted individual and produces a new information state in which the individual is in a relation of belief to the information state taken as argument (we spell this out more explicitly in §4.2.2). Moreover, the information state it produces has a presupposition; in particular, that the information state which it takes as its argument is `true`.

Let's first compose the two sentences conjoined by *if*. The first sentence, we derive as follows.

$$\frac{\langle \textit{John}, \mathtt{j}\rangle :: \mathtt{d} \qquad \langle \textit{came}, \mathtt{came}\rangle :: \mathtt{d}\backslash\mathtt{s}}{\langle \textit{John came}, \mathtt{came\ j}\rangle :: \mathtt{s}}\,(FA)$$

Given this meaning, the second sentence gets the following interpretation.

$$\frac{\langle \textit{Mary}, \mathtt{m}\rangle :: \mathtt{d} \qquad \dfrac{\langle \textit{knows}, \mathtt{knows}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{s} \qquad \langle \textit{John came}, \mathtt{came\ j}\rangle :: \mathtt{s}}{\langle \textit{knows John came}, \mathtt{knows(came\ j)}\rangle :: \mathtt{d}\backslash\mathtt{s}}\,(FA)}{\langle \textit{Mary knows John came}, \mathtt{knows(came\ j)m}\rangle :: \mathtt{s}}\,(BA)$$

We can expand the result of the second derivation as follows.

$$\begin{aligned}
&\texttt{knows(came j)m}\\
=\ &(\lambda i,c.(c(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e\})) = \texttt{true}\\
&\qquad \Vdash (\lambda g.\{\langle w,g\rangle \mid \mathbf{believe}w(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e)\})\mathbf{m}\})))
\end{aligned}$$

According to the meaning derived, *Mary knows John came* presupposes that John came. Now, we may derive the meaning of (33) itself.

$$\dfrac{\dfrac{\langle \textit{if}, \texttt{if}\rangle :: (\texttt{s/s})/\texttt{s} \qquad \langle \textit{John came}, (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e\}))\rangle :: \texttt{s}}{\langle \textit{if John came}, \texttt{if(came j)}\rangle :: \texttt{s/s}}(FA) \qquad \langle \textit{Mary knows John came}, \texttt{knows(came j)m}\rangle :: \texttt{s}}{\langle \textit{if John came Mary knows John came}, \texttt{if(came j)(knows(came j)m)}\rangle :: \texttt{s}}(FA)$$

We may then expand the result as follows.

$$\begin{aligned}
&\texttt{if(came j)(knows(came j)m)}\\
=\ &(\lambda i,c.(c\,\texttt{true} = \texttt{true}\\
&\qquad \Vdash (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e) \rightarrow \mathbf{believe}w(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e)\})\mathbf{m}\})))
\end{aligned}$$

Thus the resulting meaning lacks presuppositions, as we can see by applying `reset` to it.

$$\begin{aligned}
&\texttt{reset}(\lambda i,c.(c\,\texttt{true} = \texttt{true}\\
&\qquad \Vdash (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e) \rightarrow \mathbf{believe}w(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e)\})\mathbf{m}\})))\\
=\ &(\lambda i,c.(\top \Vdash (\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e) \rightarrow \mathbf{believe}w(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e)\})\mathbf{m}\})))\\
=\ &(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e) \rightarrow \mathbf{believe}w(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{come}w\mathbf{j}e)\})\mathbf{m}\})^{\uparrow_{\mathbf{D}}}
\end{aligned}$$

This analysis thus predicts that the presupposition triggered by the verb *knows* in the consequent is filtered in the conditional as a whole. Thus we obtain a mere information state (returned in the graded monad **D**) in all worlds of which, if John came in that world, then Mary believes that John came in that world. These truth conditions appear to correspond adequately to those of (33).

Next, we look at cases, known as "donkey anaphora", in which an indefinite noun phrase in the antecedent of a conditional covaries in its reference with an anaphoric expression in the consequent, while, at the same time, it appears to contribute universal quantificational force.

(34) If a girl walked in, Mary saw the girl.

(34) could be paraphrased as, 'Mary saw every girl that walked in', for example. In the analysis of this example, we require an instance of the anaphora

λ-term	Shorthand	Type
$(\lambda p,\psi,i,c.\psi i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}\epsilon,e}(\lambda q.(\lambda k.q(\lambda f,s.k(p \rightrightarrows f)s))^{\uparrow\mathbf{P}}))$	`if`	$T \to \mathbf{D}eT \to \mathbf{D}eT$
$(\lambda P,k,i.((\lambda g.\{\langle w,g''\rangle \mid (\exists g'.g[i]g',P(\lambda g,w,g'.g_i)g'wg'')\})^{\uparrow\mathbf{D}};k(\lambda g,w,g'.g_i))i')$	`a`	$(E \to T) \to \mathbf{K}\epsilon E$
$(\lambda x,g.\{\langle w,g\rangle \mid \mathbf{girl}w(xgwg)\})$	`girl`	$E \to T$
$(\lambda x,g.\{\langle w,g\rangle \mid (\exists e.\mathbf{walk}w(xgwg)e)\})$	`walked_in`	$E \to T$
$(\lambda g,w,g'.\mathbf{m})$	`m`	E
$(\lambda x,y,g.\{\langle w,g\rangle \mid (\exists e.\mathbf{see}w(xgwg)(ygwg)e)\})$	`saw`	$E \to E \to T$
$(\lambda P,i,c.(c(Px) = \mathtt{true} \Vdash_x x))$	`the`	$(E \to T) \to \mathbf{D}\{E\}E$
$(\lambda x,g.\{\langle w,g\rangle \mid \mathbf{girl}w(xgwg)\})$	`girl`	$E \to T$
$(\lambda g,w,g'.g_i)$	$\mathtt{pro}_i$	E

Table 4.2: Abbreviations

rule. The entire sentence is thus given the following derivation, which uses the rule for anaphora. For reference, the abbreviations used in the derivation are defined in Table 4.2.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \mathit{if}, \mathtt{if}\rangle :: (\mathtt{s}/\mathtt{s})/\mathtt{s}}{\langle \mathit{if}, \mathtt{if}^{\uparrow_{\mathbf{D}}}\rangle :: (\mathtt{s}/\mathtt{s})/\mathtt{s}}\uparrow \quad \langle \mathit{a\ girl\ walked\ in}, ((\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}{\langle \mathit{if\ a\ girl\ walked\ in}, \mathtt{if}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}/\mathtt{s}} (FA) \quad \langle \mathit{Mary\ saw\ the\ girl}, \mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow_{\mathbf{D}}} \triangleright^* (\mathtt{thegirl}))\rangle :: \mathtt{s}}{\langle \mathit{if\ a\ girl\ walked\ in\ Mary\ saw\ the\ girl}, (\mathtt{if}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}) \triangleright^* (\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow_{\mathbf{D}}} \triangleright^* (\mathtt{thegirl})))\rangle :: \mathtt{s}} (FA) \quad \mu}{\langle \mathit{if\ a\ girl\ walked\ in\ Mary\ saw\ the\ girl}, \mu_{\mathbf{D}_{\epsilon,\{E\}}}((\mathtt{if}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}) \triangleright^* (\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow_{\mathbf{D}}} \triangleright^* (\mathtt{thegirl}))))\rangle :: \mathtt{s}} \mu$$

$$\dfrac{}{\langle \mathit{if\ a\ girl\ walked\ in\ Mary\ saw\ the\ girl}, (\lambda i, c.\mathsf{anaph}_0 E \epsilon \mathsf{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}((\mathtt{if}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{agirl}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}) \triangleright^* (\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow_{\mathbf{D}}} \triangleright^* (\mathtt{thegirl}))))) ic)\rangle} \mathit{Anaph}$$

$$\begin{aligned}
& \mathtt{reset}(\lambda i, c.\mathsf{anaph}_0 E \epsilon \mathsf{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}((\mathtt{if}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{agirl}) \triangleleft^* \mathtt{walked_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}) \triangleright^* (\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{saw}^{\uparrow_{\mathbf{D}}} \triangleright^* (\mathtt{thegirl}))))) ic) \\
= \ & \mathtt{reset}(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[i]g', \mathbf{girl} w g_i', (\exists e.\mathbf{walk} w g_i' e)) \rightarrow (\exists g''.\mathbf{girl} w g_0))\}) = \mathtt{true} \\
& \qquad \Vdash (\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[i]g', \mathbf{girl} w g_i', (\exists e.\mathbf{walk} w g_i' e)) \rightarrow (\exists g'', e.g' = g'', \mathbf{see} w g_0 \mathbf{m} e)\}))) \\
= \ & (\lambda i, c.(\mathtt{id}(\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[0]g', \mathbf{girl} w g_0', (\exists e.\mathbf{walk} w g_0' e)) \rightarrow (\exists g''.\mathbf{girl} w g_0))\}) = \mathtt{true} \\
& \qquad \Vdash (\lambda g.\{\langle w, g\rangle \mid (\forall g'.(g[0]g', \mathbf{girl} w g_0', (\exists e.\mathbf{walk} w g_0' e)) \rightarrow (\exists g'', e.g' = g'', \mathbf{see} w g_0 \mathbf{m} e)\}))) \\
= \ & (\lambda g.\{\langle w, g\rangle \mid (\forall x.(\mathbf{girl} w x, (\exists e.\mathbf{walk} w x e)) \rightarrow (\exists e.\mathbf{see} w x \mathbf{m} e)\}))^{\uparrow_{\mathbf{D}}}
\end{aligned}$$

As shown underneath the derivation, if we reset this meaning, we obtain the following result.

$$(\lambda g.\{\langle w, g\rangle \mid (\forall x.(\mathbf{girl}wx, (\exists e.\mathbf{walk}wxe)) \rightarrow (\exists e.\mathbf{see}wx\mathbf{m}e)\})^{\uparrow}$$

We end up with an information state, without presuppositions, in which Mary saw every girl that walked in.

Recall that one of the given principles of the framework we are developing is that presupposition triggers can take scope outside of the syntactic contexts in which they are embedded. This has the result that, while presuppositions can become filtered within the local contexts of their triggers, they need not be. Thus we avoid a particular pitfall of the satisfaction account of presupposition, as presented in Heim 1983: what Geurts (1996) calls the "proviso problem". The problem itself is exemplified by contrasts like the following.

(35) a. If Emily is a diver, she brought her wetsuit.

b. If Emily is a diver, she brought her brother.

What (35a) seems to presuppose is that, if Emily is a diver, then she has a wetsuit. This is indeed the presupposition predicted for this sentence within Heim's satisfaction account, which, as noted in Geurts 1996, predicts for conditionals of the form $p \rightarrow q$, where r is the presupposition of p and s is the presupposition of q, that they presuppose $r \wedge (p \rightarrow s)$. The reason the satisfaction account makes this prediction is that it defines a sentence's presuppositions in terms of the potential input contexts whose update with that sentence is defined. As Heim puts it, "S presupposes p iff all contexts that admit S entail p." For sentence S to presuppose proposition p is no less than for S to require that its local context entail p. As pointed out by Francez (2018), a general account of the contexts in which a given presupposition is satisfied (as is given in Karttunen (1974)'s original presentation of the satisfaction account) need not make this identification. One could (as Karttunen did) simply state the conditions under which the presupposition of the consequent of a conditional is satisfied in terms of the entailments of its antecedent and those of its broader syntactic context; crucially, without making commitments as to the presupposition of the conditional sentence itself. The failure of Heim's satisfaction account is rendered all the more poignant, however: it results precisely from attempting to deliver a compositional account of how presuppositions are filtered.

For (35a), the above definition of presupposition does not present a problem, as (35a) indeed appears to presuppose exactly what the contexts it may update are required to entail: that if Emily is a diver, she has a wetsuit. When we look at (35b), however, this prediction is apparently not correct. What (35b) appears to presuppose is that Emily has a brother; not that she has a brother so long as she is a diver. But the latter is exactly what the satisfaction account predicts to be the presupposition of (35b), according to the above definition of a sentence's presuppositions. In the satisfaction account, the presuppositions of (35b) are therefore misconstrued.

In the present account, we avoid predicting the weak presuppositions associated with a conditional like (35b) by allowing the presupposition trigger *her brother* to take scope. In particular, our type shifting operations allow us to treat the presupposition of the consequent in (35b)—i.e., that Emily has a brother—as a side effect. One of the benefits of the monadic approach to side effects is that it allows them to systematically be ignored, even by functions whose values themselves depend on side effects. Schematically, if we have a function $f : \mathbf{D}e\alpha \to \mathbf{D}f\beta$ (which doesn't care about the identity of the effect e registered by its argument), and we have a computation $(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M)) : \mathbf{D}e\alpha$, we then have a choice about how to compose them. Either we do the obvious thing; i.e.,

$$f \triangleright (\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M))$$

but we also have the option to allow the side effect of $(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M))$ to be ignored, by doing

$$\begin{aligned} & f^{\uparrow_{\mathbf{D}}} \triangleright^{*} (\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M))^{\uparrow_{\mathbf{D}_1}} \\ = \; & ((\triangleright)^{\uparrow_{\mathbf{D}}}_{\epsilon,\epsilon} f^{\uparrow_{\mathbf{D}}})^{\mathbf{D}}_{\epsilon,e}((\lambda x.x^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}}}_{\epsilon,e}(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M))) \\ = \; & (\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} (f \triangleright M^{\uparrow_{\mathbf{D}}}))) \end{aligned}$$

The presupposition ϕ associated with the sequent $(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} M))$ has out-scoped the function f and is thus insensitive to its effects. Given the result type of f, the resulting meaning is of type $\mathbf{D}e(\mathbf{D}f\beta)$; however, we may use $\mu_{\mathbf{D}_{e,f}}$ to deliver a meaning with one single side effect:

$$\mu_{\mathbf{D}_{e,f}}(\lambda i, c.(\phi \Vdash_{x_1,\ldots,x_m} (f \triangleright M^{\uparrow_{\mathbf{D}}}))) : \mathbf{D}(e+f)\beta$$

The result of using the μ type shift is that the side effect of $(\phi \Vdash_{x_1,\ldots,x_m} M)$ has both escaped f and been sequenced with its side effects. More concretely,

λ-term	Shorthand	Type
$(\lambda g, w, g'.\mathbf{e})$	e	E
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{diver}w(xgwg)\})$	diver	$E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{wetsuit}w(xgwg)\})$	wetsuit	$E \to T$
$(\lambda x, y, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{bring}w(xgwg)(ygwg)e)\})$	brought	$E \to E \to T$
$(\lambda i, c.(\top \Vdash_x x))$	she	$\mathbf{D}\{E\}E$
$(\lambda P.(\lambda i, c.(c((\lambda g.\{\langle w, g\rangle \mid \mathbf{have}w(ygwg)(xgwg)\}) + Py) = \mathtt{true} \Vdash_{x,y} y)))$	her	$(E \to T) \to \mathbf{D}\{E, E\}E$

Table 4.3: Abbreviations

if we think of f as the context provided by the antecedent of the conditional, and $(\phi \Vdash_{x_1,\ldots,x_m} M)$ as the meaning of the consequent, we observe the presuppositions of the entire conditional to be the presuppositions of the consequent (i.e., the side effect of $(\phi \Vdash_{x_1,\ldots,x_m} M)$) together with the presuppositions of the antecedent (i.e., the side effects of f). This result provides a striking contrast with that obtained by the usual satisfaction account, in which presuppositions are evaluated in the local contexts of their triggers.

For the particular case of (35), we define a function her (provided in Table 4.3), which, like the meaning of *the*, takes a (dynamized) property into the graded monad **D**. This meaning first takes a property, then seeks a possessor and a possessee, and then returns the possessee, granted that the possession relation holds of the two and that the property holds of the possessee. We can then assign the following lexical entry to *her* (ignoring the gender presupposition associated with the possessor).

$$\langle \mathit{her}, \mathtt{her}\rangle :: \mathsf{d/n}$$

We may then analyze (35a) as follows, given the meanings for *Emily*, *diver*, *wetsuit*, *brought*, and *she* provided in Table 4.3. Note the simplified meaning of the subject pronoun *she*; i.e., that it lacks a gender presupposition.

$$\dfrac{\dfrac{\langle \textit{if}, \texttt{if}\rangle :: (\texttt{s}/\texttt{s})/\texttt{s} \quad \dfrac{\langle \textit{Emily}, \texttt{e}\rangle :: \texttt{d} \quad \dfrac{\langle \textit{is}, \texttt{id}\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{d} \quad \dfrac{\langle \textit{a}, \texttt{id}\rangle :: \texttt{d}/\texttt{n} \quad \langle \textit{diver}, \texttt{diver}\rangle :: \texttt{n}}{\langle \textit{a diver}, \texttt{diver}\rangle :: \texttt{d}}\,(FA) \quad \langle \textit{is a diver}, \texttt{diver}\rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{Emily is a diver}, \texttt{divere}\rangle :: \texttt{s}}\,(BA)}{\langle \textit{if Emily is a diver}, \texttt{if(divere)}\rangle :: \texttt{s}/\texttt{s}}\,(FA) \quad \dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{she}, \texttt{she}\rangle :: \texttt{d} \quad \langle \textit{brought her wetsuit}, \texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit})\rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{she brought her wetsuit}, \texttt{she} \triangleleft^* (\texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit}))\rangle :: \texttt{s}}\,\textit{Anaph}}{\langle \textit{she brought her wetsuit}, \texttt{anaph}_0 E\{E\}\texttt{e}(\texttt{she} \triangleleft^* (\texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit})))\rangle :: \texttt{s}}\,\textit{Anaph}}{\langle \textit{she brought her wetsuit}, \texttt{anaph}_0 E \epsilon \texttt{e}(\texttt{anaph}_0 E\{E\}\texttt{e}(\texttt{she} \triangleleft^* (\texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit}))))\rangle :: \texttt{s}}\,(FA) \quad \dfrac{\dfrac{\langle \textit{brought}, \texttt{brought}\rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{brought}, \texttt{brought}^{\overset{\uparrow}{\texttt{D}}}\rangle :: \texttt{d}\backslash\texttt{s}} \uparrow \dfrac{\langle \textit{her}, \texttt{her}\rangle :: \texttt{d}/\texttt{n} \quad \langle \textit{wetsuit}, \texttt{wetsuit}\rangle :: \texttt{n}}{\langle \textit{her wetsuit}, \texttt{herwetsuit}\rangle :: \texttt{d}}\,(FA)}{\langle \textit{she brought her wetsuit}, \texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit})\rangle :: \texttt{d}\backslash\texttt{s}}\,(FA)}{\langle \textit{if Emily is a diver she brought her wetsuit}, \texttt{if(divere)}(\texttt{anaph}_0 E \epsilon \texttt{e}(\texttt{anaph}_0 E\{E\}\texttt{e}(\texttt{she} \triangleleft^* (\texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit}))))))\rangle :: \texttt{s}}\,(FA)$$

$$\begin{aligned}
&\texttt{if(divere)}(\texttt{anaph}_0 E \epsilon \texttt{e}(\texttt{anaph}_0 E\{E\}\texttt{e}(\texttt{she} \triangleleft^* (\texttt{brought}^{\overset{\uparrow}{\texttt{D}}} \triangleright^* (\texttt{herwetsuit})))))\\
=\ &(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{diver}w\mathbf{e} \rightarrow (\mathbf{have}w(ygwg)\mathbf{e} \wedge \mathbf{wetsuit}w(ygwg))\}) = \texttt{true}\\
&\quad \Vdash_y (\lambda g.\{\langle w, g\rangle \mid \mathbf{diver}w\mathbf{e} \rightarrow (\exists e.\mathbf{bring}w(ygwg)\mathbf{e}e)\}))
\end{aligned}$$

In order to satisfy the anaphoric dependency and presupposition of (35a), one must simply find some or other entity which, as long as Emily is a diver, is Emily's wetsuit. If Emily is not a diver, any old individual will do; if she is, the individual should be Emily's wetsuit. Thus any preceding discourse that satisfies the presupposition will indeed entail that, if Emily is a diver, she has a wetsuit, and not necessarily anything stronger. Compare this result to that yielded by the following derivation of (35b).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{if}, \mathtt{if}\rangle :: (\mathtt{s}/\mathtt{s})/\mathtt{s} \quad \dfrac{\dfrac{\langle \textit{Emily}, \mathtt{e}\rangle :: \mathtt{d} \quad \dfrac{\langle \textit{is}, \mathtt{id}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d} \quad \dfrac{\langle \textit{a}, \mathtt{id}\rangle :: \mathtt{d}/\mathtt{n} \;\; \langle \textit{diver}, \mathtt{diver}\rangle :: \mathtt{n}}{\langle \textit{a diver}, \mathtt{diver}\rangle :: \mathtt{d}}\,(FA)}{\langle \textit{is a diver}, \mathtt{diver}\rangle :: \mathtt{d}\backslash\mathtt{s}}\,(BA) \quad \langle \textit{Emily is a diver}, \mathtt{divere}\rangle :: \mathtt{s}}{\langle \textit{if Emily is a diver}, \mathtt{if(divere)}\rangle :: \mathtt{s}/\mathtt{s}}\,(FA)}{\langle \textit{if Emily is a diver}, (\mathtt{if(divere)})^{\uparrow_{\mathtt{D}}}\rangle :: \mathtt{s}/\mathtt{s}}\uparrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{she}, \mathtt{she}\rangle :: \mathtt{d} \quad \dfrac{\dfrac{\dfrac{\langle \textit{brought}, \mathtt{brought}\rangle :: \mathtt{d}\backslash\mathtt{s}}{\langle \textit{brought}, \mathtt{brought}^{\uparrow_{\mathtt{D}}}\rangle :: \mathtt{d}\backslash\mathtt{s}}\uparrow \quad \dfrac{\langle \textit{her}, \mathtt{her}\rangle :: \mathtt{d}/\mathtt{n} \;\; \langle \textit{brother}, \mathtt{brother}\rangle :: \mathtt{n}}{\langle \textit{her brother}, \mathtt{herbrother}\rangle :: \mathtt{d}}\,(FA)}{\langle \textit{brought her brother}, \mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})\rangle :: \mathtt{d}\backslash\mathtt{s}}\,(BA)}{\langle \textit{she brought her brother}, \mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother}))\rangle :: \mathtt{s}}\,\textit{Anaph}}{\langle \textit{she brought her brother}, \mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})))\rangle :: \mathtt{s}}\,\textit{Anaph}}{\langle \textit{she brought her brother}, \mathtt{anaph}_0 E\epsilon\mathtt{e}(\mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother}))))\rangle :: \mathtt{s}}\uparrow}{\langle \textit{she brought her brother}, (\mathtt{anaph}_0 E\epsilon\mathtt{e}(\mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})))))^{\uparrow_{\mathtt{D}_1}}\rangle :: \mathtt{s}}\,(FA)}{\langle \textit{if Emily is a diver she brought her brother}, (\mathtt{if(divere)})^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{anaph}_0 E\epsilon\mathtt{e}(\mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})))))^{\uparrow_{\mathtt{D}_1}}\rangle :: \mathtt{s}}\,\mu$$

$$\langle \textit{if Emily is a diver she brought her brother}, \mu_{\mathbf{D}\{E\},\epsilon}((\mathtt{if(divere)})^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{anaph}_0 E\epsilon\mathtt{e}(\mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})))))^{\uparrow_{\mathtt{D}_1}})\rangle :: \mathtt{s}$$

$$\begin{aligned}
& \mu_{\mathbf{D}\{E\},\epsilon}((\mathtt{if(divere)})^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{anaph}_0 E\epsilon\mathtt{e}(\mathtt{anaph}_0 E\{E\}\mathtt{e}(\mathtt{she} \triangleleft^{*} (\mathtt{brought}^{\uparrow_{\mathtt{D}}} \triangleright^{*} (\mathtt{herbrother})))))^{\uparrow_{\mathtt{D}_1}}) \\
= \; & (\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{have}w(ygwg)\mathbf{e} \wedge \mathbf{brother}w(ygwg)\}) = \mathtt{true} \\
& \quad \Vdash_y (\lambda g.\{\langle w, g\rangle \mid \mathbf{diver}w\mathbf{e} \rightarrow (\exists e.\mathbf{bring}w(ygwg)\mathbf{e}e)\}))
\end{aligned}$$

In this case, the presupposition registered by the conditional as a whole is just the one registered by the expression *her brother*. This result obtains because, as illustrated in the final line of the derivation of *she brought her brother*, the side effect contributed by *her brother* takes scope via an internal application of $(\cdot)^{\overset{\uparrow}{\mathrm{D}}}$. Once the side effect takes scope, the antecedent *if Emily is a diver* may no longer directly interact with the presupposition, e.g., to weaken it. Finally, the side effect contributed by the consequent of the conditional is collapsed together with the trivial side effect contributed by $(\cdot)^{\overset{\uparrow}{\mathrm{D}}}$ in the final line of the derivation, via an application of $\mu_{\mathbf{D}\{E\},\epsilon}$. What prior discourse must imply, on this analysis, is that Emily has a brother tout court.

One may wonder, at this point, whether or not this has been a sufficient explanation of the different presuppositions associated with (35a) and (35b). For example, does the present account not predict that both (35a) and (35b) are each ambiguously associated with two kinds of presuppositions: one wherein the presupposition of the consequent is weakened by the antecedent of the conditional and one wherein it is not? Indeed. Given that we have one presupposition trigger (*her* NP) and one filtering device (*if* S…), each sentence will be two-ways ambiguous. Given the predicted ambiguity, it is worth considering sentences like the following, which semantic theories tend also to predict is ambiguous.

(36) Every swimmer brought a wetsuit.

In particular, (36) should be able to either mean that every swimmer brought a different wetsuit, or that there is some single wetsuit that every swimmer brought. Many speakers, however, find that the second reading involves a certain amount of strain. Should that affect our assessement of theories of quantificational expressions that attribute this reading to (36)? No! There are likely other factors that enter into semantic construal than those which should be formulated grammatically. Conceptual or world knowledge about buying events, for example, might be one of those. Similarly, we don't need to explain within the grammatical account proposed above why the sentences in (35) are associated with different presuppositions. In fact, it seems likely that the explanation would have something to do with the improved likelihood of an individual having a wetsuit if she is a diver, while the likelihood of her having a brother is unlikely to be affected by this contingency.[1] The innovation of the present account is the ability to

[1] See Lassiter (2012) for an account of projection behavior in this vein which argues that

associate different readings with sentences containing presuppositions in the first place. This possibility is absent in, for example, the satisfaction account of Heim 1983, in which presupposition triggers are trapped inside of their local contexts. Nevertheless, it is not too difficult to construct sentences whose presuppositions are ambiguous in exactly the way the present account would predict. Consider the following examples.[2]

(37) a. If Emily likes Italian food, she'll bring her pasta maker.

b. If Ronald were popular, he would probably invite his friends over for dinner.

c. If Ashley crossed the street, it's because she knows the light turned green.

d. If I ran outside, I'd eventually stop getting dizzy.

(37a) presupposes either (a) that Emily has a pasta maker, or (b) that Emily has a pasta maker if she likes Italian food; (37b) presupposes either (a) that Ronald has friends, or (b) that Ronald would have friends if he were popular; (37c) presupposes either (a) that the light turned green, or (b) that the light turned green if Ashley crossed the street; and (37d) presupposes either (a) that I get dizzy, or (b) that I would get dizzy if I ran outside. The two-way ambiguities manifested by these examples are exactly what is predicted by an account of presupposition projection that attributes it to *scope*: because their are two scopal possibilities in each case, there ought to be two possible construals.

---

the presuppositions of conditional sentences are determined by probabilisitically defined utterance usage conditions. An atomic sentence is taken to be utterable if its presuppositions have a high probability of being true, while a conditional sentence is assumed utterable if the presuppositions of its consequent have a high conditional probability of being true, given the truth of its antecedent. Because, in (35b) the conditional and unconditional probabilities are likely (close to) equal, the conditional sentence itself is predicted to be utterable if the stronger, unconditional presupposition is true.

In contrast to Lassiter's account, the present framework maintains that there are genuine semantic ambiguities about the presuppositions of conditional sentences. Here Lassiter's usage conditions may be viewed as constraints on the resolution of ambiguous sentences, perhaps in tandem with a constraint which prefers attributing to ambiguous sentences their strongest possible interpretation; the latter might have an effect similar to that intended by Maximize Presupposition (Heim, 1991).

[2]See Karttunen 1974 and van der Sandt 1992 for examples of the same kind of ambiguity.

4.2.2 Propositional attitude verbs

A second set of phenomena appearing to exhibit presupposition triggers escaping their local contexts is that in which they are embedded in the complements of propositional attitude verbs. The following contrast illustrates.

(38) a. Mary believes a man walked in. She believes the man sat down.

b. A man walked in. Mary believes the man sat down.

As these examples show, the presupposition of a definite description embedded within the complement of *believe* can be satisfied either by the context given by Mary's beliefs, as in (38a), or by the context given by the discourse as a whole, as in (38b). In particular, (38a) does not, in fact, entail the existence of a man, and yet the presupposition of the definite in the second sentence is satisfied: (38a), as an entire discourse, lacks presuppositions. At the same time, the presupposition of the definite of (38b) is satisfied, despite the fact that (38b) does not entail that Mary believes in the existence of a man—it instead entails that some man exists who Mary believes sat down (for all Mary knows, he may be a beast). That both (38a) and (38b) don't presuppose anything shows that the definite description in each has its presupposition satisfied in one of two ways: either within the local context given by Mary's beliefs, or within the global context given by the discourse as a whole. In the first case, the propositional attitude verb takes control of the presupposition contributed by the definite description in its complement, weakening it to a presupposition that depends on Mary's beliefs. In the second case, the definite description takes scope out of the complement of the propositional attitude verb into the matrix clause, where it registers a global presupposition.

To treat the meaning of the verb *believe*, we add a constant $\mathbf{dox} : e \to s \to T$ to the metalanguage. **dox** takes an individual and a world in order to return the individual's doxastically accessible information state in that world. We also define three abbreviations—`slide`, `shift`, and `modalize`—which assist in presenting the semantics of propositional attitude verbs. In the definition of `shift`, the relation symbol $\subseteq_T$ is used as an abbreviation. For each type ending in t, we have such an abbreviating symbol, defined as

follows.

$$
\begin{aligned}
(\subseteq_t) &: t \to t \to t \\
\phi \subseteq_t \psi &:= \phi \to \psi \\
(\subseteq_{\alpha\to\beta}) &: (\alpha \to \beta) \to (\alpha \to \beta) \to t \\
\phi \subseteq_{\alpha\to\beta} \psi &:= (\forall x^{\alpha}.\phi x \subseteq_{\beta} \psi x)
\end{aligned}
$$

The effect of `slide` is to ensure that indefinites introduced outside the intensional context of the propositional attitude verb can still antecede anaphora inside the intensional context. The effect of `shift` is to shift either the asserted content or the presuppositions of a proposition into an intensional context of some kind. The effect of `modalize` is to apply some modal context changer to both the asserted and presupposed part of a proposition with presuppositions.[3]

$$
\begin{aligned}
\texttt{slide} &: i \to (i \to e) \to T \to T \\
\texttt{slide} &:= (\lambda i, g, p, g', w, g''. \\
&\qquad (\exists g'''.pg'wg''' \wedge (\forall j.(j \leq i \to g''j = gj) \\
&\qquad\qquad \wedge (i' \leq j \to g''j = g'''(j - i))))) \\
\texttt{shift} &: (s \to T) \to T \to T \\
\texttt{shift} &:= (\lambda q, p, g.\{\langle w, g\rangle \mid qw \subseteq_T qw + p\}) \\
\texttt{modalize} &: \mathbf{D}eT \to (i \to T \to T) \to \mathbf{D}eT \\
\texttt{modalize} &:= (\lambda\phi, f, i, c.(fi)^{\uparrow\mathbf{P}}_{\mathbf{P}\downarrow\epsilon,e}(\phi(i+i)(c \circ fi)))
\end{aligned}
$$

With these abbreviations, we define the following abbreviation `believe`, which will serve as the meaning of the propositional attidude verb *believe*,

$$
\begin{aligned}
\texttt{believe} &: \mathbf{D}eT \to E \to \mathbf{D}eT \\
\texttt{believe} &:= (\lambda\phi, x.\texttt{modalize}\phi \\
&\qquad (\lambda i, p, g, w, g'.\texttt{shift}(\lambda w'.\texttt{slide}ig \\
&\qquad\qquad (\mathbf{dox}(xgwg')w'))pgwg'))
\end{aligned}
$$

[3]The definition of **slide** uses the constants $(-)$ and $(\leq)$, whose behavior is axiomatized as follows:

$$
\begin{aligned}
\mathbf{i} - n &= \mathbf{i} \\
m' - n' &= m - n \\
(\forall n^{i}.\mathbf{i} &\leq n) \\
m' \leq n' &\leftrightarrow m \leq n
\end{aligned}
$$

and add the following lexical entry.

$$\langle \textit{believe}, \texttt{believe} \rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{s}$$

According to this meaning, the verb takes an information state with presuppositions, along with an individual, and returns a new information state with presuppositions. The returned information state maps any assignment to itself across a world just in case the individual's "slid" doxastic information state in that world, applied to that assignment, is a subset of the world-assignment pairs gotten by updating that "slid" doxastic information state with the information state denoted by the embedded clause and applying the result to the assignment. In other words, updating the individual's doxastic information state with the meaning of the embedded clause is harmless: it doesn't rule out any world-assignment pairs that were in the original doxastic state, once applied to the input assignment. It may add more world-assignment pairs—say, if pairs $\langle w, g \rangle$ are in the individual's doxastic state applied to the input assignment only if g_1 is a boy who walked in and sat down, while the state updated with the meaning of the embedded clause, once applied to the assignment, includes pairs $\langle w, g \rangle$ if g_1 is any boy who walked in. In this case, the update will have weakened the doxastic information state; crucially, however, it is forbidden from strenghthening it.

What the propositional attitude verb does to the at-issue content of its complement clause, it also does to its presuppositions. If the presupposition of the complement of *believe* is that there is a man, then the main clause will presuppose that the agent believes that there is a man. In this way, we may explain the status of examples like (38a). Thus the first sentence of (38a) receives the following interpretation, given the meaning of *believe*.

$$\dfrac{\langle \textit{Mary}, \texttt{m} \rangle :: \texttt{d} \quad \dfrac{\langle \textit{believes}, \texttt{believe} \rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{s} \quad \dfrac{\dfrac{\langle \textit{a}, \texttt{a} \rangle :: \texttt{d}/\texttt{n} \quad \langle \textit{man}, \texttt{man} \rangle :: \texttt{n}}{\langle \textit{a man}, \texttt{aman} \rangle :: \texttt{d}}\,(FA) \quad \dfrac{\langle \textit{walked in}, \texttt{walked\_in} \rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{walked in}, \texttt{walked\_in}^{\uparrow_{\texttt{K}}} \rangle :: \texttt{d}\backslash\texttt{s}}\uparrow}{\langle \textit{a man walked in}, (\texttt{aman}) \triangleleft^{*} \texttt{walked\_in}^{\uparrow_{\texttt{K}}} \rangle :: \texttt{s}}\,(BA)}{\langle \textit{a man walked in}, ((\texttt{aman}) \triangleleft^{*} \texttt{walked\_in}^{\uparrow_{\texttt{K}}})^{\Downarrow} \rangle :: \texttt{s}}\Downarrow}{\langle \textit{believes a man walked in}, \texttt{believe}((\texttt{aman}) \triangleleft^{*} \texttt{walked\_in}^{\uparrow_{\texttt{K}}})^{\Downarrow} \rangle :: \texttt{d}\backslash\texttt{s}}\,(FA)}{\langle \textit{Mary believes a man walked in}, \texttt{m} \triangleleft (\texttt{believe}((\texttt{aman}) \triangleleft^{*} \texttt{walked\_in}^{\uparrow_{\texttt{K}}})^{\Downarrow}) \rangle :: \texttt{s}}\,(BA)$$

λ-term	Shorthand	Type
$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{sit} w (x g w g) e\}))$	sat_down	$E \to T$

Table 4.4: Abbreviations

The resulting term may be expanded as follows.

$$
\begin{aligned}
& \mathtt{m} \blacktriangleleft (\mathtt{believe}((\mathtt{a\,man}) \blacktriangleleft^{*} \mathtt{walked\_in}^{\uparrow_{\mathrm{K}}})^{\Downarrow}) \\
= \; & (\lambda i.((\lambda g.\{\langle w, g\rangle \mid \mathtt{slide} i g (\mathbf{doxm} w) \\
& \qquad \subseteq_T \mathtt{slide} i g (\mathbf{doxm} w) \\
& \qquad\qquad + (\lambda g.\{\langle w', g'\rangle \mid g[i+i]g' \wedge \mathbf{man} w' g'_{i+i} \wedge (\exists e.\mathbf{walk} w' g'_{i+i} e)\}\})))^{\uparrow_{\mathrm{D}}} i)
\end{aligned}
$$

Since the embedded clause lacks presuppositions, so does the main clause, as the update to the context provided by *believe* has simply been thrown out. The meaning of the second sentence of (38a), which has presuppositions, is composed as follows (where the new abbrevation sat_down is defined in Table 4.4).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{the}, \texttt{the}\rangle :: \texttt{d/n} \quad \langle \textit{man}, \texttt{man}\rangle :: \texttt{n}}{\langle \textit{the man}, \texttt{theman}\rangle :: \texttt{d}}\,(FA) \quad \dfrac{\langle \textit{sat down}, \texttt{sat\_down}\rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{sat down}, \texttt{sat\_down}^{\uparrow_{\texttt{D}}}\rangle :: \texttt{d}\backslash\texttt{s}}\uparrow}{}\,(BA)}{\langle \textit{believes}, \texttt{believe}\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{s} \quad \langle \textit{the man sat down}, (\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}}\rangle :: \texttt{s}}\,(FA)}{\langle \textit{believes the man sat down}, \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})\rangle :: \texttt{d}\backslash\texttt{s}}\uparrow}{\langle \textit{she}, \texttt{she}\rangle :: \texttt{d} \quad \langle \textit{believes the man sat down}, \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})^{\uparrow_{\texttt{D}}}\rangle :: \texttt{d}\backslash\texttt{s}}\,(BA)}{\langle \textit{she believes the man sat down}, \texttt{she} \triangleleft^{*} \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})^{\uparrow_{\texttt{D}}}\rangle :: \texttt{s}}\,\textit{Anaph}$$

$$\dfrac{\dfrac{\dfrac{\langle \textit{she believes the man sat down}, \texttt{anaph}_0 E\{E\}\texttt{m}((\texttt{she} \triangleleft^{*} \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})^{\uparrow_{\texttt{D}}}))\rangle :: \texttt{s}}{\langle \textit{she believes the man sat down}, \mu_{\mathbf{D}_{\epsilon,\{E\}}}(\texttt{anaph}_0 E\{E\}\texttt{m}((\texttt{she} \triangleleft^{*} \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})^{\uparrow_{\texttt{D}}})))\rangle :: \texttt{s}}\,\mu}{}\,\textit{Anaph}$$

$$\langle \textit{she believes the man sat down}, \texttt{anaph}_0 E \epsilon \texttt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\texttt{anaph}_0 E\{E\}\texttt{m}((\texttt{she} \triangleleft^{*} \texttt{believe}((\texttt{theman}) \triangleleft^{*} \texttt{sat\_down}^{\uparrow_{\texttt{D}}})^{\uparrow_{\texttt{D}}})))))\rangle :: \texttt{s}$$

Although it is not shown explicitly in the derivation (given the abbreviations), feeding the meaning of *the man sat down* to the meaning of the verb *believe* has the effect that the presupposition registered by the definite description *the man* is modified to depend on the relevant agent's belief state: specifically, it is made into a presupposition that the presupposed individual is a man in the information state corresponding to the agent's beliefs; not in the matrix information state. Thus composing the two sentences of (38a) gives us the result on the next page, If we apply reset to this meaning, we see that the presupposition of the definite description has been filtered.

$$\dfrac{\dfrac{\langle \textit{Mary believes a man walked in}, \mathtt{m} \triangleleft (\mathtt{believe}((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}) \rangle :: \mathtt{s} \quad \langle ;, (\,;) \rangle :: (\mathtt{s} \backslash \mathtt{s})/\mathtt{s}}{\langle \textit{Mary believes a man walked in} \;;, (\lambda\psi.(\mathtt{m} \triangleleft (\mathtt{believe}((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}))\,;\psi) \rangle :: \mathtt{s}/\mathtt{s} \; \langle \textit{she believes the man sat down}, \mathtt{anaph}_0 E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\epsilon,\{E\}} (\mathtt{anaph}_0 E\{E\} \mathtt{m}((\mathtt{she} \triangleleft^* \mathtt{believe}((\mathtt{theman}) \triangleleft^* \mathtt{sat\_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}}}))))) \rangle :: \mathtt{s}}\,(FA)}{\langle \textit{Mary believes a man walked in ; she believes the man sat down}, (\mathtt{m} \triangleleft (\mathtt{believe}((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}))\,;(\mathtt{anaph}_0 E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\epsilon,\{E\}} (\mathtt{anaph}_0 E\{E\} \mathtt{m}((\mathtt{she} \triangleleft^* \mathtt{believe}((\mathtt{theman}) \triangleleft^* \mathtt{sat\_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}}}))))) \rangle :: \mathtt{s}}\,(BA)$$

$$\begin{aligned}
&(\mathtt{reset}(\mathtt{m} \triangleleft (\mathtt{believe}((\mathtt{aman}) \triangleleft^* \mathtt{walked_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow})))\\
&\qquad ;(\mathtt{anaph}_0 E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\epsilon,\{E\}} (\mathtt{anaph}_0 E\{E\} \mathtt{m}((\mathtt{she} \triangleleft^* \mathtt{believe}((\mathtt{theman}) \triangleleft^* \mathtt{sat_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}}})))))\\
=\;& (\lambda g.\{\langle w, g\rangle \mid \mathbf{doxm}\, w \subseteq_T \mathbf{doxm}\, w + (\lambda g.\{\langle w', g'\rangle \mid g[0]g' \wedge \mathbf{man} w' g'_0 \wedge (\exists e.\mathbf{walk} w' g'_0 e) \wedge (\exists e.\mathbf{sit} w' g'_0 e)\})))^{\uparrow_{\mathbf{D}}}
\end{aligned}$$

Thus we correctly predict that the sentence in (38a) lacks presuppositions.

One benefit the above analysis is that we have a way of accounting for entailment patterns like the following ones.

(39) a. Mary believes a man walked in. She believes the man sat down.
$\rightsquigarrow$ Mary believes a man walked in and sat down.

b. Mary believes a man walked in and sat down.
$\rightsquigarrow$ Mary believes a man walked in. She believes the man sat down.

Giving the following conjunctive semantics for *and* will help.[4]

$$\langle and, (\lambda Q, P, x.(Px)^{\uparrow_{\mathbf{D}}} ; (x^{\uparrow_{\mathbf{D}}} \blacktriangleleft^{*} Q))\rangle :: ((\mathsf{d}\backslash\mathsf{s})\backslash(\mathsf{d}\backslash\mathsf{s}))/(\mathsf{d}\backslash\mathsf{s})$$

Then, we can compose (39b) as follows (where we abbreviate the meaning for *and* just proposed as and). Following the derivation, we provide a β-reduced λ-term to which `reset` has been applied, in order to provide a better comparision with the meaning of (39a).

[4] Although not relevant to the example at hand, this meaning for *and* has the effect of filtering whatever presuppositions are incurred in the second conjunct by the first conjunct. Hence, its type is $\mathbf{D}e(E \to T) \to (E \to T) \to E \to \mathbf{D}eT$.

$$\frac{\langle \textit{sat down}, \mathtt{sat\_down}\rangle :: \mathtt{d\backslash s}}{\langle \textit{sat down}, \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}\rangle :: \mathtt{d\backslash s}}\ \uparrow$$

$$\frac{\langle \textit{and}, \mathtt{and}\rangle :: \mathtt{((d\backslash s)\backslash(d\backslash s))/(d\backslash s)} \quad \langle \textit{sat down}, \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}\rangle :: \mathtt{d\backslash s}}{\langle \textit{and sat down}, \mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}\rangle :: \mathtt{(d\backslash s)\backslash(d\backslash s)}}\ (FA)$$

$$\frac{\langle \textit{walked in}, \mathtt{walked\_in}\rangle :: \mathtt{d\backslash s} \quad \langle \textit{and sat down}, \mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}\rangle :: \mathtt{(d\backslash s)\backslash(d\backslash s)}}{\langle \textit{walked in and sat down}, \mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}})\rangle :: \mathtt{d\backslash s}}\ (BA)$$

$$\frac{\langle \textit{walked in and sat down}, \mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}})\rangle :: \mathtt{d\backslash s}}{\langle \textit{walked in and sat down}, (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}}\rangle :: \mathtt{d\backslash s}}\ \uparrow$$

$$\frac{\langle a, \mathtt{a}\rangle :: \mathtt{d/n} \quad \langle \textit{man}, \mathtt{man}\rangle :: \mathtt{n}}{\langle \textit{a man}, \mathtt{aman}\rangle :: \mathtt{d}}\ (FA)$$

$$\frac{\langle \textit{a man}, \mathtt{aman}\rangle :: \mathtt{d} \quad \langle \textit{walked in and sat down}, (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}}\rangle :: \mathtt{d\backslash s}}{\langle \textit{a man walked in and sat down}, (\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}}\rangle :: \mathtt{s}}\ (BA)$$

$$\frac{\langle \textit{a man walked in and sat down}, (\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}}\rangle :: \mathtt{s}}{\langle \textit{a man walked in and sat down}, ((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}\ \Downarrow$$

$$\frac{\langle \textit{a man walked in and sat down}, ((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}{\langle \textit{a man walked in and sat down}, \mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}\ \mu$$

$$\frac{\langle \textit{believes}, \mathtt{believe}\rangle :: \mathtt{(d\backslash s)/s} \quad \langle \textit{a man walked in and sat down}, \mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}{\langle \textit{believes a man walked in and sat down}, \mathtt{believe}(\mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow})\rangle :: \mathtt{d\backslash s}}\ (FA)$$

$$\frac{\langle \textit{Mary}, \mathtt{m}\rangle :: \mathtt{d} \quad \langle \textit{believes a man walked in and sat down}, \mathtt{believe}(\mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow})\rangle :: \mathtt{d\backslash s}}{\langle \textit{Mary believes a man walked in and sat down}, \mathtt{m} \triangleleft (\mathtt{believe}(\mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}))\rangle :: \mathtt{s}}\ (BA)$$

$$\begin{aligned} & \mathtt{reset}(\mathtt{m} \triangleleft (\mathtt{believe}(\mu_{\mathbf{D}_{\epsilon,\epsilon}}((\mathtt{aman}) \triangleleft^{*} (\mathtt{walked\_in} \triangleleft (\mathtt{and} \triangleright \mathtt{sat\_down}^{\overset{\uparrow}{\mathbf{D}}}))^{\overset{\uparrow}{\mathbf{K}}})^{\Downarrow}))) \\ = \ & (\lambda g.\{\langle w, g\rangle \mid \mathbf{doxm}w \subseteq_T \mathbf{doxm}w + (\lambda g.\{\langle w', g'\rangle \mid g[0]g' \wedge \mathbf{man}w'g'_0 \wedge (\exists e.\mathbf{walk}w'g'_0 e) \wedge (\exists e.\mathbf{sit}w'g'_0 e\})\})\})^{\overset{\uparrow}{\mathbf{D}}} \end{aligned}$$

The result is exactly the same meaning as that obtained for (38a)/(39a).

Let's now look at (38b). We derive the first sentence, *a man walked in*, as follows.

$$\dfrac{\dfrac{\dfrac{\langle a, \mathtt{a}\rangle :: \mathtt{d/n} \qquad \langle man, \mathtt{man}\rangle :: \mathtt{n}}{\langle a\ man, \mathtt{aman}\rangle :: \mathtt{d}}\,(FA) \qquad \dfrac{\langle walked\ in, \mathtt{walked\_in}\rangle :: \mathtt{d\backslash s}}{\langle walked\ in, \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}}\rangle :: \mathtt{d\backslash s}}\uparrow}{\langle a\ man\ walked\ in, (\mathtt{aman}) \blacktriangleleft^{*} \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}}\rangle :: \mathtt{s}}\,(BA)}{\langle a\ man\ walked\ in, ((\mathtt{aman}) \blacktriangleleft^{*} \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}\rangle :: \mathtt{s}}\Downarrow$$

In contrast to (38a), therefore, (38b) requires an interpretation of its second sentence whereon it presupposes the existence of a man; not that Mary believes that there is a man. We, therefore, ought to derive the second sentence of (38b) as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{the}, \texttt{the}\rangle :: \texttt{d/n}\ \ \langle \textit{man}, \texttt{man}\rangle :: \texttt{n}}{\langle \textit{the man}, \texttt{theman}\rangle :: \texttt{d}}\,(FA) \quad \dfrac{\langle \textit{sat down}, \texttt{sat\_down}\rangle :: \texttt{d}\backslash\texttt{s}}{\langle \textit{sat down}, \texttt{sat\_down}^{\uparrow \texttt{D}}\rangle :: \texttt{d}\backslash\texttt{s}}\uparrow}{\langle \textit{the man sat down}, (\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}}\rangle :: \texttt{s}}\,(BA)}{\langle \textit{the man sat down}, ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1}\rangle :: \texttt{s}}\uparrow}{\langle \textit{believes the man sat down}, \texttt{believe}^{\uparrow \texttt{D}} \blacktriangleright^{*} ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1}\rangle :: \texttt{d}\backslash\texttt{s}}\,(FA)}{\langle \textit{Mary believes the man sat down}, \texttt{m}^{\uparrow \texttt{D}} \blacktriangleleft^{*} (\texttt{believe}^{\uparrow \texttt{D}} \blacktriangleright^{*} ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1})\rangle :: \texttt{s}}\,(BA)}{\langle \textit{Mary believes the man sat down}, \mu_{\mathbf{D}_{\epsilon,\{E\}}}(\texttt{m}^{\uparrow \texttt{D}} \blacktriangleleft^{*} (\texttt{believe}^{\uparrow \texttt{D}} \blacktriangleright^{*} ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1}))\rangle :: \texttt{s}}\,\mu}{\langle \textit{Mary believes the man sat down}, \texttt{anaph}\,E\epsilon\texttt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\texttt{m}^{\uparrow \texttt{D}} \blacktriangleleft^{*} (\texttt{believe}^{\uparrow \texttt{D}} \blacktriangleright^{*} ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat\_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1})))\rangle :: \texttt{s}}\,\textit{Anaph}$$

$$\begin{aligned}
&\texttt{anaph}\,E\epsilon\texttt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\texttt{m}^{\uparrow \texttt{D}} \blacktriangleleft^{*} (\texttt{believe}^{\uparrow \texttt{D}} \blacktriangleright^{*} ((\texttt{theman}) \blacktriangleleft^{*} \texttt{sat_down}^{\uparrow \texttt{D}})^{\uparrow \texttt{D}_1})))\\
=\ &(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid \mathbf{man} w g_0\}) = \texttt{true} \Vdash (\lambda g.\{\langle w, g\rangle \mid \mathbf{doxm} w \subseteq_T \mathbf{doxm} w + (\lambda g'.\{\langle w', g'\rangle \mid (\exists e.\mathbf{sit} w' g_0 e)\})\})))
\end{aligned}$$

We can then update the first sentence of (38b) with the second. Because the presupposition trigger—the definite description—of the second sentence has taken scope above the intensional context provided by Mary's beliefs, its presupposition is satisfied by the asserted content of the first sentence. Following the derivation, we provide the resulting meaning, to which reset has been applied.

$$\dfrac{\dfrac{\langle \textit{a man walked in}, ((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow} \rangle :: \mathtt{s} \quad \langle ;, (;) \rangle :: (\mathtt{s}\backslash\mathtt{s})/\mathtt{s}}{\langle \textit{a man walked in} ;, (\lambda\psi.((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}; \psi) \rangle :: \mathtt{s}/\mathtt{s}} (BA) \quad \langle \textit{Mary believes the man sat down}, \mathtt{anaph}E\epsilon\mathtt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{believe}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{theman}) \triangleleft^* \mathtt{sat\_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}_1}}))) \rangle :: \mathtt{s}}{\langle \textit{a man walked in ; Mary believes the man sat down}, ((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}; (\mathtt{anaph}E\epsilon\mathtt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{believe}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{theman}) \triangleleft^* \mathtt{sat\_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}_1}}))))) \rangle :: \mathtt{s}} (FA)$$

$$\mathtt{reset}((\mathtt{aman}) \triangleleft^* \mathtt{walked\_in}^{\uparrow_{\mathbf{K}}})^{\Downarrow}; (\mathtt{anaph}E\epsilon\mathtt{pro}_0(\mu_{\mathbf{D}_{\epsilon,\{E\}}}(\mathtt{m}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathtt{believe}^{\uparrow_{\mathbf{D}}} \triangleright^* ((\mathtt{theman}) \triangleleft^* \mathtt{sat\_down}^{\uparrow_{\mathbf{D}}})^{\uparrow_{\mathbf{D}_1}}))))$$

$$= (\lambda g.\{\langle w, g' \rangle \mid g[0]g' \wedge \mathbf{man} w g'_0 \wedge (\exists e.\mathbf{walk} w g'_0 e) \wedge \mathbf{doxm} w \subseteq_T \mathbf{doxm} w + (\lambda g''.\{\langle w', g'' \rangle \mid (\exists e.\mathbf{sit} w' g'_0 e)\})\})^{\uparrow_{\mathbf{D}}}$$

This information state relates an assignment to another across a world if the latter assignment fills register 0 of the former with a man who walked in that world and that Mary believes sat down in that world. These truth conditions result when the definite description embedded inside of the complement of the propositional attitude verb takes scope outside of this complement to satisfy its presupposition in the discourse as a whole.

Before moving forward, let us compare the account presented here with the discussions of the ambiguity of presuppositions generally found within a satisfaction account like Heim's. Within this theory of presupposition, what is observed to be an ambiguity in where presuppositions project tends to be thought of in one of two ways. One way involves thinking of the ambiguity of presupposition projection in terms of opacity with respect to intensional contexts. Taking the above discourses as an example, the definite description *the man* enters into a *de re*/*de dicto* ambiguity, such that when it is interpreted *de re*, it somehow lodges its presuppositional content above the intensional context created by the propositional attitude verb, and when it is interpreted *de dicto*, below. It is the strategy of such a semantic treatment, therefore, to regard the second sentence of (38b) as a paraphrase of the following example.

(40) As for the man, Mary believes that he sat down.

If (38b) can, somehow, be assimilated to (40), its presuppositional behavior can be explained as an effect of whatever mechanism for handling the scopal construal of presupposition triggers assimilates them.

Another way of thinking about the ambiguous behavior of presuppositions involves treating ambiguity as resulting from the availability of a pragmatic strategy of interpretation known as "global accommodation". Under this strategy, presupposition triggers embedded in intensional contexts can require that their presuppositions be satisfied globally if it can be accommodated into the discourse that the intensional context entails the information represented by the discourse. Given an utterance of *Mary believes the man sat down*, what might be accommodated is that Mary's doxastic state entails the information state that the utterance updates (e.g., by, somehow, paring down the latter to make it so). In that case, the satisfaction of the presupposition in (38b) will simply be said to result from accommodating the information that Mary believes that a man walked in; i.e., enough to entail the presupposition that the second sentence of (38b) is predicted to have by the satisfaction account.

The second strategy seems not to work if only because the ambiguities

associated with presupposition triggers are not generally tied to any constraints on the relations between the contexts that they feature. Take the following discourse as an example.

(41) Mary believes that there aren't any men. She believes that the man is not a man, but a dog.

We likely don't accommodate the presupposition that Mary believes there is a man in (41) (or anything entailing it), and yet (41) presupposes the existence of a man.

As for the first strategy, its main flaw is that it is not formulated within the satisfaction account. Although, the task of this dissertation is not to literally assimilate presupposition projection to quantificational scope, we use another scope taking mechanism to achieve *de re*/*de dicto* ambiguities of presuppositional content (see above!). This strategy, as illustrated above, allows presupposition triggers to send their presuppositional requirements outside of the intensional contexts in which they are embedded.

4.3 Modal subordination

Modal subordination (Roberts, 1989) is a phenomenon wherein a sentence is understood as if it were embedded within an intensional context in which it is not, in fact, syntactically embedded, and wherein this context is provided by an antecedent sentence which may itself be syntactically modal. The modal auxiliary *would* can provide modally subordinated intensional contexts for the interpretation of its prejacent. Here is an example.

(42) If John were gardening, a dog would walk in. The dog would be drenched.

The second sentence of (42) appears to comment on the hypothetical context introduced in the sentence which precedes it. It seems to say that if John were gardening and a dog walked in, the dog that walked in would be drenched. What is interesting about examples of modal subordination like this is not only that they involve anaphora to intensional contexts, but that indefinite noun phrases introduced within those contexts can provide antecedents for anaphora in the sentence understood as subordinated. The definite description *the dog* in (42), for example, is anaphoric to the indefinite noun phrase *a dog* of the previous sentence, and, moreover, (42) as a whole lacks presuppositions. The pattern exhibited here is reminiscent

λ-term	Shorthand	Type
$(\lambda R, \phi.\phi R)$	`if`	$(s \to T) \to ((s \to T) \to T) \to T$
$(\lambda P, x, w.\mathbf{mod} w + Px)$	`were`	$(E \to T) \to E \to s \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{garden} w(xgwg)e\}))$	`gardening`	$E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{walk} w(xgwg)e\}))$	`walked_in`	$E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{drenched} w(xgwg)\})$	`drenched`	$E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{dog} w(xgwg)\})$	`dog`	$E \to T$
$(\lambda P, k, i.((\lambda g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g', P(\lambda g, w, g'.g_i)g'wg'')\})^{\uparrow_{\mathbf{D}}}; k(\lambda g, w, g'.g_i))i')$	a	$(E \to T) \to \mathbf{K}\epsilon E$

Table 4.5: Abbreviations

of that observed among propositional attitude verbs: an indefinite introduced in an intensional context provides an antecedent for an anaphor introduced within a semantically equivalent intensional context in later discourse. To analyze modal subordination of this kind, let's say that *would* is itself anaphoric to a function from worlds to information states. In particular, let's assign it the following lexical entry, which makes use of the abbreviation `would`.

$$\texttt{would} : \mathbf{K}e(E \to T) \to \mathbf{K}fE \to (s \to T) \to \mathbf{D}(e+f)T$$
$$\texttt{would} := (\lambda P, x, q.\texttt{modalize}(P^{\mathbf{K}\downarrow}_{e,f} x)^{\Downarrow}(\lambda i, g.\texttt{shift}(\texttt{slide}ig \circ q)g))$$

$$\langle \textit{would}, \texttt{would}\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{v}$$

Thus in general, *would* will have the ability to rely on some previously introduced intensional context to supply this function. We should also refine the meaning we assign to *if* in order to allow it, on the one hand, to be modal, and, on the other hand, to be able to supply its consequent with the meaning of the intensional context provided by its antecedent. This will involve plugging the intensional context argument of its consequent with this intensional context. We should add a new constant, **mod**, in order to provide the appropriate modal context for the conditional.

$$\mathbf{mod} : s \to T$$

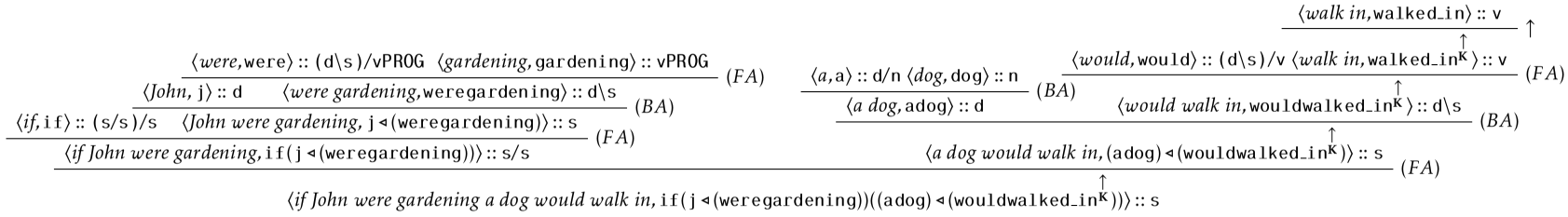
This constant will feature in the analysis of the auxiliary *were*, to which we assign the following lexical entry.

$$\langle \textit{were}, (\lambda P, x, w.\mathbf{mod} w + Px)\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{vPROG}$$

We thus provide for *if* the following lexical entry.

$$\langle if, (\lambda R, \phi.\phi R)\rangle :: (\mathtt{s}/\mathtt{s})/\mathtt{s}$$

We can now analyze the first sentence of (42), which we also compose with ;, so that we may view its `reset` meaning. The derivation given makes use of the abbreviations provided in Table 4.5.


⟨were, were⟩ :: (d\s)/vPROG ⟨gardening, gardening⟩ :: vPROG (FA)
⟨John, j⟩ :: d ⟨were gardening, weregardening⟩ :: d\s (BA)
⟨if, if⟩ :: (s/s)/s ⟨John were gardening, j◃(weregardening)⟩ :: s (FA)
⟨if John were gardening, if(j◃(weregardening))⟩ :: s/s
⟨a, a⟩ :: d/n ⟨dog, dog⟩ :: n (BA)
⟨a dog, adog⟩ :: d
⟨walk in, walked_in⟩ :: v ↑
⟨would, would⟩ :: (d\s)/v ⟨walk in, walked_in^K⟩ :: v (FA)
⟨would walk in, wouldwalked_in^K⟩ :: d\s (BA)
⟨a dog would walk in, (adog)◃(wouldwalked_in^K)⟩ :: s (FA)
⟨if John were gardening a dog would walk in, if(j◃(weregardening))((adog)◃(wouldwalked_in^K))⟩ :: s

To analyze the second sentence of (42), we have to deal with the final, $s \to T$-type argument of *would*. What we would like is to saturate this argument with the same modal relation used to saturate the corresponding argument in the meaning of the first sentence of (42). We should, therefore, make the argument anaphoric. To do this, we can introduce the following function push, which pushes an argument onto a turnstile.

$$\begin{aligned}&\mathsf{push} : (\alpha \to \beta) \to \mathbf{D}\{\alpha\}\beta\\&\mathsf{push} := (\lambda f, i, c.(\top \Vdash_x (f\, x)))\end{aligned}$$

Then, we may add the following lexical entry, which does the relevant work. We use ϵ denotes the empty phonetic form.

$$\langle \epsilon, \mathsf{push} \rangle :: \mathsf{s/s}$$

We now derive the second sentence of (42) as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{the}, \texttt{the}\rangle :: \texttt{d/n} \quad \langle \textit{dog}, \texttt{dog}\rangle :: \texttt{n}}{\langle \textit{the dog}, \texttt{thedog}\rangle :: \texttt{d}}(FA)}{\langle \textit{the dog}, (\texttt{thedog})^{\Uparrow}\rangle :: \texttt{d}}\Uparrow \quad \dfrac{\dfrac{\langle \textit{be}, \texttt{id}\rangle :: \texttt{v/a} \quad \langle \textit{drenched}, \texttt{drenched}\rangle}{\langle \textit{be drenched}, \texttt{drenched}\rangle :: \texttt{v}}(FA)}{\langle \textit{be drenched}, \texttt{drenched}^{\uparrow}\rangle :: \texttt{v}}\uparrow \quad \langle \textit{would}, \texttt{would}\rangle :: \texttt{(d\textbackslash s)/v}}{\langle \textit{would be drenched}, \texttt{woulddrenched}^{\uparrow}\rangle :: \texttt{d\textbackslash s}}(FA) \quad \langle \epsilon, \texttt{push}\rangle :: \texttt{s/s}}{\langle \textit{the dog would be drenched}, (\texttt{thedog})^{\Uparrow} \triangleleft (\texttt{wouldrenched}^{\uparrow})\rangle :: \texttt{s}}(BA) \quad \langle \textit{the dog would be drenched}, \texttt{push}((\texttt{thedog})^{\Uparrow} \triangleleft (\texttt{wouldrenched}^{\uparrow}))\rangle :: \texttt{s}}(FA)}{\mu} \quad \langle \textit{the dog would be drenched}, \mu_{\mathbf{D}\{s\to T\},\{E\}}(\texttt{push}((\texttt{thedog})^{\Uparrow} \triangleleft (\texttt{wouldrenched}^{\uparrow})))\rangle :: \texttt{s}}\mu \quad \langle \textit{the dog would be drenched}, \texttt{anaph}_1 E \epsilon \texttt{pro}_0(\mu_{\mathbf{D}\{s\to T\},\{E\}}(\texttt{push}((\texttt{thedog})^{\Uparrow} \triangleleft (\texttt{wouldrenched}^{\uparrow}))))\rangle :: \texttt{s}}\textit{Anaph}$$

$$\begin{aligned}
&\texttt{anaph}_1 E \epsilon \texttt{pro}_0(\mu_{\mathbf{D}\{s\to T\},\{E\}}(\texttt{push}((\texttt{thedog})^{\Uparrow} \triangleleft (\texttt{wouldrenched}^{\uparrow}))))\\
=\ &(\lambda i, c.(c(\lambda g.\{\langle w, g\rangle \mid qw \subseteq_T qw + (\lambda g'.\{\langle w', g'\rangle \mid \mathbf{dog}\, w'\, g'_0\})\}) = \texttt{true}\\
&\qquad \Vdash_q (\lambda g.\{\langle w, g\rangle \mid qw \subseteq_T qw + (\lambda g'.\{\langle w', g'\rangle \mid \mathbf{drenched}\, w'\, g'_0\})\})))
\end{aligned}$$

If we now update the first sentence of (42) with the second, and do anaphora, we obtain the following result.

$$\dfrac{\dfrac{\langle \textit{if John were gardening a dog would walk in}, \mathtt{if(j \triangleleft (weregardening))((adog) \triangleleft^{*} (wouldwalked\_in^{\uparrow_{\mathbf{K}}}))} \rangle :: \mathtt{s} \qquad \langle \mathtt{;}, (\mathtt{;}) \circ \mathtt{reset} \rangle :: \mathtt{s} \backslash (\mathtt{s}/\mathtt{s})}{\langle \textit{if John were gardening a dog would walk in ;}, (\lambda\psi.(\mathtt{reset(if(j \triangleleft (weregardening))((adog) \triangleleft (wouldwalked\_in^{\uparrow_{\mathbf{K}}}))); \psi}) \rangle :: \mathtt{s}/\mathtt{s} \quad \langle \textit{the dog would be drenched}, \mathtt{anaph}_1\, E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\{s \to T\},\{E\}} (\mathtt{push((thedog)^{\Uparrow} \triangleleft (wouldd renched^{\uparrow_{\mathbf{K}}}))})) \rangle :: \mathtt{s}}\,(FA)}{\langle \textit{if John were gardening a dog would walk in ; the dog would be drenched}, \mathtt{(reset(if(j \triangleleft (weregardening))((adog) \triangleleft (wouldwalked\_in^{\uparrow_{\mathbf{K}}})))); (anaph}_1\, E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\{s \to T\},\{E\}} (\mathtt{push((thedog)^{\Uparrow} \triangleleft (wouldd renched^{\uparrow_{\mathbf{K}}}))})) \rangle :: \mathtt{s}}\,(BA)$$

$$\langle \textit{if John were gardening a dog would walk in ; the dog would be drenched}, \mathtt{anaph}_0 (s \to E) \epsilon (\lambda w.\mathbf{mod} w + \mathtt{gardening\ j})(\mathtt{(reset(if(j \triangleleft (weregardening))((adog) \triangleleft (wouldwalked\_in^{\uparrow_{\mathbf{K}}})))); (anaph}_1\, E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\{s \to T\},\{E\}} (\mathtt{push((thedog)^{\Uparrow} \triangleleft (wouldd renched^{\uparrow_{\mathbf{K}}}))})) \rangle :: \mathtt{s}\,(Anaph)$$

$$\begin{aligned}
&\mathtt{anaph}_0 (s \to E) \epsilon (\lambda w.\mathbf{mod} w + \mathtt{gardening\ j}) \\
&\qquad (\mathtt{(reset(if(j \triangleleft (weregardening))((adog) \triangleleft (wouldwalked_in^{\uparrow_{\mathbf{K}}})))}) \\
&\qquad\qquad \mathtt{; (anaph}_1\, E \epsilon \mathtt{pro}_0 (\mu_{\mathbf{D}\{s \to T,E\}} (\mathtt{push((thedog)^{\Uparrow} \triangleleft (wouldd renched^{\uparrow_{\mathbf{K}}}))})) \\
= \ & (\lambda g.\{\langle w, g \rangle \mid \mathbf{mod} w + (\lambda g'.\{\langle w', g' \rangle \mid (\exists e.\mathbf{garden} w \mathbf{j} e)\}) \\
&\qquad \subseteq_T \mathbf{mod} w + (\lambda g'.\{\langle w', g'' \rangle \mid g'[0] g'', (\exists e.\mathbf{garden} w \mathbf{j} e), \mathbf{dog} w g_0'', (\exists e.\mathbf{walk} w g_0'' e), \mathbf{drenched} w' g_0''\})\}^{\uparrow_{\mathbf{D}}}
\end{aligned}$$

Thus we end up with an information state relating an assignment to itself across a world if, when **mod** is applied to the world and updated with the information that John is gardening, one gets a subset of the information gotten by applying **mod** to the world and updating it with the information that John is gardening and that a dog who walked in drenched is held at register 0. As one can verify, we predict the following entailments between sentences, given the meaning for *and* proposed above.

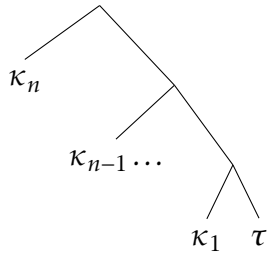
(43) a. If John were gardening, a dog would walk in. The dog would be drenched
$\rightsquigarrow$ If John were gardening, a dog would walk in and be drenched.

b. If John were gardening, a dog would walk in and be drenched.
$\rightsquigarrow$ If John were gardening, a dog would walk in. The dog would be drenched.

In the next chapter, we will survey some ways of applying the general framework proposed to other phenomena. In particular, we first take a look at other kinds of anaphora—specifically, null complement anaphora and ellipsis—and we then apply the framework to an account of the semantics and scopal properties of distributive quantifiers, like *every*. In the meantime, let us conclude by summarizing the key lessons brought out by the phenomena illustrated in this chapter and the general features of the framework which they highlight.

4.4 Conclusions

The phenomena surveyed in this chapter have been meant to illustrate one particular property of presupposition triggers which has not received a treatment by prior work in the vein of the satisfaction account of Karttunen (1974) and Heim (1983). This property is their ability to send their presuppositional requirements outside of the local contexts in which they occur; that is, their ability to take scope. Previous work in this tradition has come up short on this front precisely because of its reliance on the notion of a local context in order to determine whether or not the presuppositions of an expression are satisfied. What the graded monadic approach has on offer, instead, is a much more flexible notion of context. Schematically, given a presupposition trigger, τ, which composes with some number of other expressions—say propositional attitude verbs—each contributing an intensional context of some sort,



we have the choice to register the presuppositions of τ at any of these contexts. To satisfy its presuppositions within the most embedded context, we may do

$$[\![\kappa_n]\!]([\![\kappa_{n-1}]\!] \ldots ([\![\kappa_1]\!][\![\tau]\!]))$$

while, to satisfy them in the context provided by κ_i, we may instead do

$$[\![\kappa_n]\!]([\![\kappa_{n-1}]\!] \ldots [\![\kappa_i]\!](\mu_{\mathbf{D}e,f} \ldots ([\![\kappa_1]\!]^{\uparrow^{\mathbf{D}}_{\mathbf{D}} \downarrow}_{\epsilon,e} [\![\tau]\!]^{\uparrow^{D_1}})))$$

That is, we lift the presuppositions of τ via an internal application of $(\cdot)^{\uparrow^{\mathbf{D}}}$, pass the presuppositions along on an outer effectual layer until we get to the context we want, in order to finally collapse them back down via an application of $\mu_{\mathbf{D}e,f}$. $(\cdot)^{\uparrow^{\mathbf{D}}}$ frees the scope of a presupposition trigger, while $\mu_{\mathbf{D}e,f}$ fixes it. The result is genuine semantic ambiguity about which context a given presupposition is satisfied in and, hence—following Heim (1983)—what an expression's presuppositions are. By endowing the satisfaction account of presupposition with a graded monad, we rescue it from the problem of trapped presupposition triggers.

Chapter 5

Some further applications

5.1 Introduction

This chapter extends the framework to three other phenomena: verb phrase ellipsis, null complement anaphora, and quantification, with particular attention to the quantifier *every*. It is shown that natural analyses are made available for both ellipsis and null complement anaphora that are able to distinguish certain important properties—ones which have been argued to motivate analyses of ellipsis according to which it involves unpronounced syntax (for proponents of this view, see Sag, 1976; Chung et al., 1995; Merchant, 2001, 2019, i.a.). The source of the distinction between the two constructions, on the present analysis, is a difference between whether the operation which suppresses a predicate's argument takes effect "in the lexicon", before any syntactic operations have applied, or after the application of of syntactic combinatory operations and type shifts. To analyze the semantic contribution of quantifier *every*, it is shown how Charlow's notion of an *underlying monad* may be incorporated into the present grammar in order to accommodate exceptionally-scoping indefinite semantics and non-exceptionally-scoping distributive quantifier semantics within the same fragment.

5.2 Complex anaphora

5.2.1 Null complement anaphora

The term 'null complement anaphora', coined in Hankamer and Sag 1976, describes a process exhibited by some predicates part of whose interpretation appears to be controlled by an antecedent or by the context of utter-

ance. These very predicates tend to have syntactic alternant which realize this part overtly as an argument. (44) and (45) exhibit this pattern this with the adjective *ready* and the verb *win*.

(44) a. Peet asked Ashley to eat a hamburger, and she was ready to eat a hamburger.

b. Peet asked Ashley to eat a hamburger, and she was ready.

(45) a. Peet ran a race and won it.

b. Peet ran a race and won.

Hankamer and Sag dedicate much of their text to illustrating differences between predicates exhibiting null complement anaphora and those that have undergone ellipsis. The explanation they adopt for these generalizations appeals to the distinction between deep anaphora, whose dependence on an antecedent is determined by the presence of a lexical anaphor (e.g., a pronoun), and surface anaphora, which require an antecedent in order to license the non-pronunciation of a syntactic constituent bearing some special relation to the antecedent (a relation which they characterize as syntactic identity). An example of ellipsis corresponding to (44b) can be created by adding to it the infinitival morpheme *to*.

(46) Peet asked Ashley to eat a hamburger, and she was ready to.

The kinds of data Hankamer and Sag use to motivate the distinction between deep anaphora (e.g., null complement anaphora) and surface anaphora (e.g., ellipsis) are twofold. First, they note that surface anaphora, as opposed to deep anaphora, tend to require a linguistic antecedent—they are controlled by prior discourse. To see this, consider *ready* again, but given the specified non-linguistic context.

(47) Context: Peet attempts to give Ashley a plate holding a triple cheeseburger with bacon and ranch sauce. Ashley says:

a. I'm not ready.

b. #I'm not ready to.

(47a), but not (47b), can be uttered felicitously in this context to mean that Ashley isn't ready to eat the triple cheeseburger. For Hankamer and Sag,

this effect rides on a grammatical difference between the two processes: only ellipsis requires linguistic control.

Second, Hankamer and Sag argue that certain kinds of linguistic contexts in which surface anaphora can be found expose their underlying syntactic structure. One of these, they dub the "missing antecedent phenomenon". Compare the following two examples respectively containing ellipsis and null complement anaphora with the verb *begin*.

(48) a. Peet ate a triple cheeseburger, and eventually Ashley began to. It was medium rare.

b. Peet ate a triple cheeseburger, and eventually Ashley began. #It was medium rare.

Hankamer and Sag take the difference in acceptability between these sorts of examples to provide evidence for a structurally present antecedent for the pronoun *it* when it follows an ellipsis site—though which is not present when it follows a null complement anaphor. In (48a), the antecedent is plausibly provided by an indefinite noun phrase buried in the ellipsis site, which is itself available given the syntactic identity of the ellipsis site with its antecedent in the preceding clause. The failed antecedent for the pronoun in (48b), however, is argued to obtain its meaning in virtue of its pronominal status: the verb *begin*, like *ready*, acts much like a pronoun, in that, like a pronoun (but unlike ellipsis), it generally allows for pragmatic control by the context of the discourse.

(49) Context: Peet motions toward the cheeseburger in front of Ashley and says:

a. I think I'll begin.

b. #I think I'll begin to.

and, moreover, its failure as an antecedent in (48b) seems plausibly due to the fact that, like a pronoun, it lacks syntactic structure of the kind that would support anaphora to an indefinite, as (48a) does.

Within our present framework, we can regard the distinction between null complement anaphora and ellipsis somewhat differently. It is more suitable, in particular, to consider null complement anaphora to result from a lexical type shift, while we consider ellipsis to be a syntactically driven process. For both types of construction, we can use the function push. In the case of null complement anaphora, we use it to supress the complement

λ-term	Shorthand	Type
$(\lambda g, w, g'.\mathbf{a})$	ash	E
$(\lambda x, y, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{eat} w (xgwg)(ygwg)e\})$	ate	$E \to E \to T$
$(\lambda x, g.\{\langle w, g\rangle \mid \mathbf{tc} w (xgwg)\})$	tc	$E \to T$
$(\lambda P, k, i.((\lambda g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g', P(\lambda g, w, g'.g_i)g'wg'')\})^{\uparrow_{\mathbf{D}}}; k(\lambda g, w, g'.g_i))i')$	a	$(E \to T) \to \mathbf{K}\epsilon E$

Table 5.1: Abbreviations

of the verb *began* by pushing it onto the turnstile, thus rendering the verb anaphoric. To accomplish this, we use the operation push whose definition we repeat here.

$$\begin{aligned}&\mathsf{push} : (\alpha \to \beta) \to \mathbf{D}\{\alpha\}\beta\\&\mathsf{push} := (\lambda f, i, c.(\top \Vdash_x (f\,x)))\end{aligned}$$

Let us have a constant $\mathbf{begin} : s \to (v \to t) \to e \to v \to t$. To state the meanings for both verbal complement *began* and its null complement anaphoric variant, we define the following abbreviation.

$$\begin{aligned}&\mathsf{began} : (V \to T) \to E \to T\\&\mathsf{began} := (\lambda P, x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{begin} w (\mathtt{sta}_{v\to t} P g w g)(xgwg)e)\})\end{aligned}$$

Then, we may add the following lexical entries to our lexicon (where the category vINF is that of infinitival finite verb phrases).

$$\begin{aligned}\langle \mathit{began}, \mathsf{began}\rangle &:: \mathtt{(d\backslash s)/vINF}\\\langle \mathit{began}, \mathsf{pushbegan}\rangle &:: \mathtt{d\backslash s}\end{aligned}$$

To (48b), we give the following analysis, which makes use of the null complement anaphoric lexical entry, as well as abbreviations defined in Table 5.1 (which we refer to for the rest of this chapter).

$$\dfrac{\dfrac{\langle \mathit{Ashley}, \mathsf{ash}\rangle :: \mathtt{d}}{\langle \mathit{Ashley}, \mathsf{ash}^{\uparrow_{\mathbf{D}}}\rangle :: \mathtt{d}}\uparrow \qquad \langle \mathit{began}, \mathsf{pushbegan}\rangle :: \mathtt{d\backslash s}}{\langle \mathit{Ashley\ began}, \mathsf{ash}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathsf{pushbegan})\rangle :: \mathtt{s}}\,(BA)$$

If we expand the result, we obtain the following term.

$$\begin{aligned}&\mathsf{ash}^{\uparrow_{\mathbf{D}}} \triangleleft^* (\mathsf{pushbegan})\\=\ &(\lambda i, c.(\top \Vdash_P (\lambda g.\{\langle w, g\rangle \mid (\exists e.\mathbf{begin} w (\lambda e'.P(\lambda g, w', g'.e')gwg')\mathbf{a}e)\})))\end{aligned}$$

What results is thus a sentence meaning with an anaphoric presupposition: the sentence seeks a dynamized property of events, in order to yield truth conditions according to which Ashley entered into a particular relation with that property (the **begin** relation).

5.2.2 Ellipsis

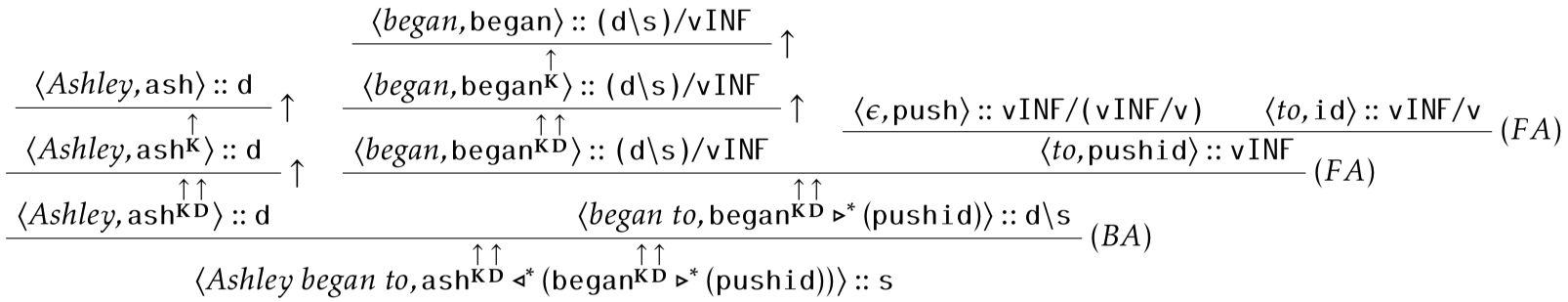
Ellipsis requires a slightly different analysis. Rather than the result of a lexical type shift, it is given by a change that occurs during an expression's syntactic derivation. As a vehicle for this change, we'll add the following two lexical entries, which turn the infinitival morpheme *to*, and a finite auxiliary, respectively, into variants that license ellipsis of their complements.

$$\langle \epsilon, \mathsf{push} \rangle :: \mathtt{vINF/(vINF/v)}$$
$$\langle \epsilon, \mathsf{push} \rangle :: \mathtt{(d\backslash s)/((d\backslash s)/v)}$$

To illustrate the effect that push has in this context, let's look at a derivation of (48a). To the infinitival morpheme *to*, we'll give the following analysis, whereon it simply denotes the identity function.

$$\langle \mathit{to}, \mathsf{id} \rangle :: \mathtt{vINF/v}$$

Then, we can give *Ashley began to* the following derivation.


⟨began, began⟩ :: (d\s)/vINF
⟨Ashley, ash⟩ :: d
⟨began, began^K⟩ :: (d\s)/vINF
⟨ϵ, push⟩ :: vINF/(vINF/v)
⟨to, id⟩ :: vINF/v
⟨Ashley, ash^K⟩ :: d
⟨began, began^KD⟩ :: (d\s)/vINF
⟨to, pushid⟩ :: vINF
(FA)
(FA)
⟨Ashley, ash^KD⟩ :: d
⟨began to, began^KD ▷* (pushid)⟩ :: d\s
(BA)
⟨Ashley began to, ash^KD ◁* (began^KD ▷* (pushid))⟩ :: s

The meaning that results is of type $\mathbf{D}(\mathbf{K}e(V \to T))(\mathbf{K}eT)$, given some effect e. Thus we require an antecedent of type $\mathbf{K}e(V \to T)$—a verb phrase meaning which is capable of, on the one hand, registering a presupposition, and, on the other hand, scoping out indefinites that introduce new registers. We can use the following antecedent, whose type is $\mathbf{K}\epsilon(V \to T)$, corresponding to what might have been used in the derivation of the preceding sentence.

$$\begin{aligned}&\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc})\\ =\ &(\lambda k, i.((\lambda g.\{\langle w, g'\rangle \mid g[i]g', \mathbf{tc}wg'_i\})^{\uparrow}_{\mathbf{D}}; k(\lambda e, g.\{\langle w, g\rangle \mid \mathbf{eat}wg_i(egwg)\}))i')\end{aligned}$$

Given this antecedent, we can resolve the ellipsis using the rule *Anaph*.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\langle \textit{Ashley began to}, \mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid}))\rangle :: \mathtt{s}}{\langle \textit{Ashley began to}, \mathsf{anaph}_0(\mathbf{K}(V \to T))\epsilon(\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc}))(\mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid})))\rangle :: \mathtt{s}}\,\textit{Anaph}}{\langle \textit{Ashley began to}, (\mathsf{anaph}_0(\mathbf{K}(V \to T))\epsilon(\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc}))(\mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid}))))^{\Uparrow}\rangle :: \mathtt{s}}\,\Uparrow}{\langle \textit{Ashley began to}, \mu_{\mathbf{K}\epsilon,\epsilon}(\mathsf{anaph}_0(\mathbf{K}(V \to T))\epsilon(\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc}))(\mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid}))))^{\Uparrow}\rangle :: \mathtt{s}}\,\mu}{\langle \textit{Ashley began to}, (\mu_{\mathbf{K}\epsilon,\epsilon}(\mathsf{anaph}_0(\mathbf{K}(V \to T))\epsilon(\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc}))(\mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid}))))^{\Uparrow})^{\Downarrow}\rangle :: \mathtt{s}}\,\Downarrow$$

If we apply `reset` to the final line, and then expand and β-reduce the result, we obtain the following meaning.

$$\begin{aligned}&\mathtt{reset}(\mu_{\mathbf{K}\epsilon,\epsilon}(\mathsf{anaph}_0(\mathbf{K}(V \to T))\epsilon(\mathtt{eat}^{\uparrow}_{\mathbf{K}} \triangleright^* (\mathtt{atc}))(\mathtt{ash}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleleft^* (\mathtt{began}^{\uparrow\uparrow}_{\mathbf{KD}} \triangleright^* (\mathtt{pushid}))))^{\Uparrow})^{\Downarrow}\\ =\ &(\lambda g.\{\langle w, g'\rangle \mid g[0]g', \mathbf{tc}wg'_0, (\exists e.\mathbf{begin}w(\mathbf{eat}wg'_0)\mathbf{a}e)\})^{\uparrow}_{\mathbf{D}}\end{aligned}$$

We end up with a term of type $\mathbf{D}\epsilon T$ that both introduces a new register and relates an input assignment to an output assignment that differs from it at that register by a triple cheeseburger that Ashley began to eat. Moreover, because of the introduction of a new register, the relevant cheeseburger is available for anaphora, explaining the judgment about (48a).

5.2.3 Summary

The present framework gives rise to a natural account of anaphoric behaviors which are attested across different construction types. In ellipsis, anaphoric meaning may potentially give rise to the introduction of new indefinite antecedents for later anaphora; in null complement anaphora,

this possibility is not observed. The difference is argued to coincide with whether the relevant construction is lexically anaphoric, i.e., in null complement anaphora, or anaphoric in virtue of a syntactic process, i.e., the selection of a potential ellipsis licensor by the silent expression denoting push. In this respect, the present analysis takes after its predecessors which attribtue *syntactic structure* to ellipsis sites (Sag, 1976; Chung et al., 1995; Merchant, 2001, 2013; Kobele, 2015, i.a.): ellipsis licesensing is fed by other inferential processes that allow for it to be governed by complex type-driven constraints. It takes after the work of researchers who regard ellipsis as analogous to other forms of anaphora, however (for a representative example, see Hardt, 1993), in that these constraints are here regarded as properly semantic, effected by type shifts. Note, however, that, in order to derive the different behaviors of null complement anaphora and verb phrase ellipsis, it is not actually necessary, given the assumed compositional scheme, to posit two different lexical entries for the verb *began*, in contrast to the syntactic operation which coincides with ellipsis. We could just as easily have analyzed both null complement anaphora and verb phrase ellipsis as resulting from syntactic operations effected by lexical entries, such as the following.[1]

$$\langle \epsilon, \mathtt{id} \rangle :: \mathtt{vINF/(vINF/v)}$$
$$\langle \epsilon, \mathtt{id} \rangle :: \mathtt{(d\backslash s)/((d\backslash s)/v)}$$
$$\langle \epsilon, \mathtt{id} \rangle :: \mathtt{(d\backslash s)/((d\backslash s)/vINF)}$$

Meanwhile, we could allow push to apply anywhere in a derivation as a freely available inference rule. Whether or not null complement anaphora or ellipsis is observed in a given construction then becomes purely a question of whether or not there is a lexical entry available to endow it with the right syntactic type. In this scenario, missing antecedents will still only be observed among elliptical constructions, due to the different lexical semantics of verbs and the infinitival morpheme *to*: *began* takes an argument of semantic type E, while *to* denotes the identity function at any type. Thus *began* may only be anaphoric to antecedents of semantic type E, while *to* may be anaphoric to an antecedent of any type. It is the possibility of having an antecedent of any semantic type which allows anaphora governed by *to*, i.e., verb phrase ellipsis, to contribute new indefinite antecedents for anaphora.

Because of the flexibility of the types of possible antecedents for anaphora under the present approach, it is worth investigating whether or not con-

[1] It would still be necessary to somehow ensure that only null complement anaphoric verbs are affected by the relevant (third) lexical entry, a nuance I ignore here for simplicity.

straints on anaphora which are usually regarded as syntactic in nature can here be understood as semantic and type-driven. For example, extraction is well known to occur out of ellipsis sites, but not out of sites of null complement anaphora.

(50) a. Peet will eat what he is ready to.

b. *Peet will eat what he is ready.

A difference between the semantic types of the anaphora sites in such examples of ellipsis and null complement anaphora would provide an avenue for describing the difference in anaphoric possibility.[2]

5.3 Quantification

Next, we show how an account of distributive quantifiers, like *every*, may be smoothly integrated with the present framework. As part of the discussion, we consider how the semantics of such quantifiers interacts with expressions which incur presuppositions. As Heim (1983) remarks, the sentence in (51) is taken, on her account (as well as that of Karttunen and Peters (1979)), to presuppose that every nation has a king, as a result of the presence of the quantifier *every nation* and the definite description *its king*.

(51) Every nation cherishes its king.

The analysis of quantifiers presented in this chapter makes consistent predictions, as shown below. First, however, we give an analysis of the basic, at-issue semantics of such quantifiers, from which it will be seen that their presuppositional behavior follows.

Distributive quantifiers are well known to differ from indefinites in that their scopal possibilities are restricted to the smallest finite clause in which they are embedded, as the following pair illustrates.

(52) a. If some triple cheeseburger was delicious, then Ashley is happy.

b. If every triple cheeseburger was delicious, then Ashley is happy.

[2]For instance, according to compositional analyses of *wh*-extraction such as that in Kobele 2018, the types of verb phrases with and without missing arguments are different: verb phrases denote functions of type $e \to t$, while verb phrases which have been extracted out of denote functions of types which have been *lifted* via a parameterized applicative functor.

(52a) may be understood in one of two ways. Either it may mean that Ashley's happiness rides on the deliciousness of some particular triple cheeseburger, or it may mean that any old cheeseburger suffices. (52b), however, may only be understood in the first way: it may mean that Ashley is happy as long as all the triple cheeseburgers were delicious; it is not enough for only one of them to be. Finite clauses thus constitute scope islands; but, curiously, only for certain quantifiers, like *every*. Indefinites, in contrast, generally appear to be unrestricted in where they may be construed as taking scope.

Charlow (2014) provides a an account of the scopal possibilities witnessed for indefinites, in contrast to other quantifiers, by regarding indefiniteness as arising from a monad. In particular, he gives an account of indefinites in which they give rise to non-determinism, which he models using the Powerset monad.[3]

Definition 21 ($\{\cdot\}$)**.**

$$\begin{aligned}\{\alpha\} &= \alpha \to t\\ a^{\overset{\uparrow}{\{\cdot\}}} &= \{a\}\\ u^{\overset{\{\cdot\}}{\downarrow}} &= (\lambda v.\{f\,x \mid f \in u, x \in v\})\\ \mu_{\{\cdot\}}\, m &= \bigcup m\end{aligned}$$

In order to study the contribution of non-exceptionally-scoping quantifiers, like *every*, Charlow uses a continuation monad transformer, i.e., analogous to the graded continuation monad transformer presented earlier in this thesis. The result is a monad ($\{\cdot\}^{\uparrow}$) capable of handling both the non-determinism triggered by indefinites and the distributive semantics of *every*.

[3]This presentation gives a simplification of Charlow's approach, which uses a monad giving rise to both non-determinism and state functionality—the State.Set monad—in order to countenance both indefinite reference and anaphora, including anaphora to indefinites.

Definition 22 ($\{\cdot\}^{\uparrow}$)**.**

$$\begin{aligned}
\{\alpha\}^{\uparrow} &= (\alpha \to \{t\}) \to \{t\} \\
\overset{\uparrow}{a}^{\{\cdot\}^{\uparrow}} &= (\lambda k.ka) \\
\overset{\{\cdot\}^{\uparrow}}{u}\!\!\downarrow &= (\lambda v, k.u(\lambda U.v(\lambda V.k(UV)))) \\
\mu_{\{\cdot\}^{\uparrow}}\, m &= (\lambda k.m(\lambda k'.k'k))
\end{aligned}$$

Given this setup, Charlow is able to state both the meanings of indefinites and distributive quantifiers. The indefinite *some*, he gives the following meaning.

$$[\![some]\!] = (\lambda P.\{x \mid Px\})$$

Its semantic type is thus $(e \to t) \to \{e\}$: it receives a property in order to produce a non-deterministic individual which satisfies that property. To *every*, he gives a meaning which is equivalent to the following.

$$[\![every]\!] = (\lambda P, k.\{(\forall x^{e}.Px \to \top \in kx)\})$$

Thus the type of *every* is $(e \to t) \to \{e\}^{\uparrow}$: it receives a property and a *continuation* in order to return the singleton set of $\top$ just in case every way of feeding an individual of which the property is true to the continuation returns a truth value which is non-deterministically $\top$; $\bot$ (false), otherwise.

To characterize the non-exceptionally-scoping behavior of *every*, Charlow provides a mechanism to reset sentence meanings in the monad $\{\cdot\}^{\uparrow}$. The effect of resetting a sentence meaning is to fix the scope of quantifiers whose meanings are stated in the monad $\{\cdot\}^{\uparrow}$; that is, by nullifying their scopal potential. To capture the generalization that non-exceptioinally-scoping quantifiers have their scope trapped inside the smallest finite clause containing them, Charlow includes in his grammar fragment a mechanism to reset meanings within the continuation monad at every finite clause boundary. The effect is $\{\cdot\}^{\uparrow}$-specific, however: the scopal potential of indefinites remains unaffected by its application.

To reset a meaning $m : \{t\}^{\uparrow}$ is to do the following.

$$(\lambda k.m(\lambda x.\overset{\uparrow}{x}^{\{\cdot\}}) \gg\!=_{\{\cdot\}} k)$$

Given the meaning of the sentence *every dog laughed*, i.e., $(\lambda k.\{(\forall x^e.\mathbf{dog}x \to \top \in k(\mathbf{laugh}x))\})$, resetting it has the following result.

$$\begin{aligned}
&(\lambda k.(\lambda k.\{(\forall x^e.\mathbf{dog}x \to \top \in k(\mathbf{laugh}x))\})(\lambda x.x^{\overset{\uparrow}{\{\cdot\}}}) \gg\!=_{\{\cdot\}} k) \\
=\ &(\lambda k.\{(\forall x^e.\mathbf{dog}x \to \top \in (\mathbf{laugh}x)^{\overset{\uparrow}{\{\cdot\}}})\} \gg\!=_{\{\cdot\}} k) \\
=\ &(\forall x^e.\mathbf{dog}x \to \mathbf{laugh}x)^{\overset{\uparrow}{\{\cdot\}^\uparrow}}
\end{aligned}$$

A scopally inert truth value is returned in the monad $\{\cdot\}^\uparrow$: $\top$ if every dog laughed; $\bot$, otherwise. In summary, Charlow treats both exceptionally scoping indefinites and non-exceptionally-scoping quantifiers within the same monadic framework by first providing a monad to characterize the scopal behavior of indefinites, and then continuizing that monad, yielding another monad to characterize the scopal behavior of other quantifiers. In Charlow's terms, the monad for indefiniteness is the "underlying monad" of the monad for other kinds of quantification. Resetting the meaning of an expression whose scopal potential is provided within the continuation monad fixes its scope, while leaving the scope of expressions whose meanings are provided within the underlying monad unaffected.

Making use of this strategy within the current framework is straightforward. For this purpose, we may employ the graded continuation monad transformer, defined in chapter 3, in order to continuize the graded monad $\mathbf{K}$ (again choosing the result type to be T). The result, $\mathbf{CT}(T)(\mathbf{K})$, which we call $\mathbf{K}^\uparrow$, is shown in the following definition.

Definition 23 ($\mathbf{K}^\uparrow$).

$$\begin{aligned}
\mathbf{K}^\uparrow &: \mathcal{T}^* \to \mathcal{T} \to \mathcal{T} \\
\mathbf{K}^\uparrow e\alpha &= (\alpha \to \mathbf{K}fT) \to \mathbf{K}(e+f)T \\
&= (\alpha \to (T \to \mathbf{D}gT) \to \mathbf{D}(f+g)T) \to (T \to \mathbf{D}hT) \to \mathbf{D}(e+f+h)T \\
a^{\overset{\uparrow}{\mathbf{K}^\uparrow}} &= (\lambda k.ka) \\
u^{\overset{\mathbf{K}^\uparrow}{\downarrow}}_{e,f} &= (\lambda v,k.u(\lambda U.v(\lambda V.k(UV)))) \\
\mu_{\mathbf{K}^\uparrow e,f}m &= (\lambda k.m(\lambda k'.k'k))
\end{aligned}$$

To provide a meaning for the determiner *every*, we first define the following

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{D}_n}} : \ldots \mathbf{D}\epsilon\alpha \ldots}\uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{D}e(\mathbf{D}f\alpha)\ldots}{\Gamma \vdash \mu_{\mathbf{D}_n e,f} m : \ldots \mathbf{D}(e+f)\alpha \ldots}\mu$$

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{K}_n}} : \ldots \mathbf{K}\epsilon\alpha \ldots}\uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}e(\mathbf{K}f\alpha)\ldots}{\Gamma \vdash \mu_{\mathbf{K}_n e,f} m : \ldots \mathbf{K}(e+f)\alpha \ldots}\mu$$

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{K}^{\uparrow}_n}} : \ldots \mathbf{K}^{\uparrow}\epsilon\alpha \ldots}\uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}^{\uparrow}e(\mathbf{K}^{\uparrow}f\alpha)\ldots}{\Gamma \vdash \mu_{\mathbf{K}^{\uparrow}_n e,f} m : \ldots \mathbf{K}^{\uparrow}(e+f)\alpha \ldots}\mu$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{D}e\alpha \ldots}{\Gamma \vdash m^{\Uparrow n} : \ldots \mathbf{K}e\alpha \ldots}\Uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}e\alpha \ldots}{\Gamma \vdash m^{\Uparrow n} : \ldots \mathbf{K}^{\uparrow}e\alpha \ldots}\Uparrow$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{K}eT \ldots}{\Gamma \vdash m^{\Downarrow n} : \ldots \mathbf{D}eT \ldots}\Downarrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}^{\uparrow}eT \ldots}{\Gamma \vdash m^{\Downarrow n} : \ldots \mathbf{K}eT \ldots}\Downarrow$$

$$\frac{\Gamma \vdash m : \mathbf{D}(\mathbf{insert}_n ae)\alpha \qquad \Gamma \vdash t : a}{\Gamma \vdash \mathsf{anaph}_n aetm : \mathbf{D}e\alpha}\ Anaph$$

Figure 5.1: Dynamic inference rules for quantification

convenient abbreviation not.

$$\mathsf{not} : \mathbf{K}eT \to \mathbf{K}eT$$
$$\mathsf{not} := (\lambda m.(\lambda i, c.(\lambda p, g.\{\langle w, g\rangle \mid \neg(\exists g'.pgwg')\})^{\overset{\uparrow}{\mathbf{P}}}_{\epsilon,e}{}^{\overset{\mathbf{P}}{\downarrow}} m^{\Downarrow} ic)^{\Uparrow})$$

The role of this function is to negate an effectful information state, while also destroying its scopal and anaphoric potential. We may use it, in turn, to define an abbreviation every, which is to serve as the meaning of the quantificational determiner *every*, as follows.

$$\mathsf{every} : (E \to T) \to \mathbf{K}^{\uparrow}\epsilon E$$
$$\mathsf{every} := (\lambda P, k.\mathsf{not}(\mathrm{a}P \gg\!=_{\mathbf{K}\epsilon,f} (\lambda x.\mathsf{not}(kx))))$$

$$\langle \textit{every}, \mathsf{every}\rangle :: \mathsf{d/n}$$

In addition, we should define lift and lower operators corresponding to the graded continuation monad $\mathbf{K}^{\uparrow}$.

Definition 24 ($(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$)**.**

$$(\cdot)^{\Uparrow} : \mathbf{K}e\alpha \to \mathbf{K}^{\uparrow}e\alpha$$
$$m^{\Uparrow} = (\lambda k.m \gg\!=_{\mathbf{K}e,f} k)$$
$$(\cdot)^{\Downarrow} : \mathbf{K}^{\uparrow}eT \to \mathbf{K}eT$$
$$m^{\Downarrow} = m(\lambda x.x^{\overset{\uparrow}{\mathbf{K}}})$$

We correspondingly update the definitions of forward and backward application, in order to accommodate meanings in the graded monad $\mathbf{K}^{\uparrow}$.

$$(\triangleright^*) ::= (\triangleright) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{D}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{D}}_{\overset{\downarrow}{e},f} v) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{K}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{D}}_{\overset{\downarrow}{e},f} v) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{K}^{\uparrow}}_{\overset{\downarrow}{e},f} v)$$
$$(\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{D}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{D}}_{\overset{\downarrow}{e},f} v) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{K}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{D}}_{\overset{\downarrow}{e},f} v) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}_{\overset{\downarrow}{e},e} u)^{\mathbf{K}^{\uparrow}}_{\overset{\downarrow}{e},f} v)$$

The definitions of the dynamic inference rules for the present setting are provided in Figure 5.1. Thus the full set of available type shifts is understood to be ammended, in order to accommodate meanings in the graded monad $\mathbf{K}^{\uparrow}$.

$$Op ::= (\cdot)^{\overset{\uparrow}{\mathbf{D}}} \mid \mu_{\mathbf{D}e,f} \mid (\cdot)^{\overset{\uparrow}{\mathbf{K}}} \mid \mu_{\mathbf{K}e,f} \mid (\cdot)^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}} \mid \mu_{\mathbf{K}^{\uparrow}e,f} \mid (\cdot)^{\Uparrow} \mid (\cdot)^{\Downarrow} \mid (\cdot)^{\Uparrow} \mid (\cdot)^{\Downarrow} \mid Op^{\overset{\uparrow}{\mathbf{D}}}_{\overset{\downarrow}{e},e} \mid Op^{\overset{\uparrow}{\mathbf{K}}}_{\overset{\downarrow}{e},e} \mid Op^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}_{\overset{\downarrow}{e},e}$$

To illustrate, we provide a derivation of the following sentence.

(53) Ashley ate every tripple cheeseburger.

To abbreviate the meanings of *Ashley*, *ate*, and *tripple cheeseburger*, we write ash, ate, and tc, respectively.

$$\dfrac{\dfrac{\langle \textit{Ashley}, \mathtt{ash}\rangle :: \mathtt{d}}{\langle \textit{Ashley}, \mathtt{ash}^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}\rangle :: \mathtt{d}} \qquad \dfrac{\dfrac{\langle \textit{ate}, \mathtt{ate}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}}{\langle \textit{ate}, \mathtt{ate}^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}\rangle :: (\mathtt{d}\backslash\mathtt{s})/\mathtt{d}} \uparrow \qquad \dfrac{\langle \textit{every}, \mathtt{every}\rangle :: \mathtt{d}/\mathtt{n} \qquad \langle \textit{triple cheeseburger}, \mathtt{tc}\rangle :: \mathtt{n}}{\langle \textit{every tripple cheeseburger}, \mathtt{everytc}\rangle :: \mathtt{d}}\,(FA)}{\langle \textit{ate every tripple cheeseburger}, \mathtt{ate}^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}} \triangleright^* \mathtt{everytc}\rangle :: \mathtt{d}\backslash\mathtt{s}}\,(FA)}{\langle \textit{Ashley ate every tripple cheeseburger}, \mathtt{ash}^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}} \triangleleft^* (\mathtt{ate}^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}} \triangleright^* \mathtt{everytc})\rangle :: \mathtt{s}}\,(BA)$$

We may then expand the term that results as follows.

$$
\begin{aligned}
& \mathsf{ash}^{\uparrow\mathbf{K}^{\uparrow}} \triangleleft^{*} (\mathsf{ate}^{\uparrow\mathbf{K}^{\uparrow}} \triangleright^{*} \mathsf{everytc}) \\
= \; & \mathsf{ash}^{\uparrow\mathbf{K}^{\uparrow}} \triangleleft^{*} (\mathsf{ate}^{\uparrow\mathbf{K}^{\uparrow}} \triangleright^{*} (\lambda k.\mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,f} (\lambda x.\mathsf{not}(kx))))) \\
= \; & \mathsf{ash}^{\uparrow\mathbf{K}^{\uparrow}} \triangleleft^{*} (\lambda k.\mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,f} (\lambda x.\mathsf{not}(k(\mathsf{ate}x))))) \\
= \; & (\lambda k.\mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,f} (\lambda x.\mathsf{not}(k(\mathsf{ate}x\mathsf{ash})))))
\end{aligned}
$$

We end up with a meaning of type $\mathbf{K}^{\uparrow}\epsilon T$. Given that it is the meaning of a finite clause, we may wish to reset it. To do so, first we lower it, and then we lift it again, as follows.

$$
\begin{aligned}
& (\lambda k.\mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,\epsilon} (\lambda x.\mathsf{not}(k(\mathsf{ate}x\mathsf{ash})))))^{\Downarrow\Uparrow} \\
= \; & \mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,\epsilon} (\lambda x.\mathsf{not}(\mathsf{ate}x\mathsf{ash})^{\uparrow\mathbf{K}}))^{\Uparrow} \\
= \; & \mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,\epsilon} (\lambda x.\mathsf{not}(\lambda g.\{\langle w,g\rangle \mid (\exists e.\mathbf{eat}w(xgwg)\mathbf{a}e)\})^{\uparrow\mathbf{K}}))^{\Uparrow} \\
= \; & \mathsf{not}(\mathsf{atc} \gg\!=_{\mathbf{K}\epsilon,\epsilon} (\lambda x.(\lambda g.\{\langle w,g\rangle \mid \neg(\exists g'',e.\mathbf{eat}w(xgwg'')\mathbf{a}e)\})^{\uparrow\mathbf{K}}))^{\Uparrow} \\
= \; & (\lambda i.(\lambda g.\{\langle w,g\rangle \mid \neg(\exists g'.\mathbf{tc}wg_i' \wedge \neg(\exists g'',e.\mathbf{eat}wg_i'\mathbf{a}e))\})^{\uparrow\mathbf{D}}i)^{\Uparrow\Uparrow} \\
= \; & (\lambda g.\{\langle w,g\rangle \mid (\forall x^{e}.\mathbf{tc}wx \rightarrow (\exists e.\mathbf{eat}wx\mathbf{a}e))\})^{\uparrow\mathbf{D}\Uparrow\Uparrow} \\
= \; & (\lambda g.\{\langle w,g\rangle \mid (\forall x.\mathbf{tc}wx \rightarrow (\exists e.\mathbf{eat}wx\mathbf{a}e))\})^{\uparrow\mathbf{K}^{\uparrow}}
\end{aligned}
$$

Analogous to the result obtained above within the continuized Powerset monad, we end up with a mere information state in which Ashley ate every cheeseburger, returned in the graded monad $\mathbf{K}^{\uparrow}$.

Having integrated an account of distributive quantifiers within the present framework, we are in a position to ask what predictions it makes about the interaction of presupposition and quantification. What happens to presuppositions triggered by expressions within the scope of such quantifiers? Let's explore these predictions with the sentence in (54) as an illustration.

(54) Every linguist stopped smoking.

We provide the following lexical entry for the verb *stopped*, assigning to it as its meaning a dynamized, non-null-complement-anaphoric variant of the one proposed in chapter 2 (see Table 5.2).

λ-term	Shorthand	Type
$(\lambda R,x,i,c.(c(Rxe) \Vdash_e (\lambda g.\{\langle w,g\rangle \mid (\exists e'.\mathbf{stop}(egwg)(xgwg)e')\}))$	stopped	$(E \to V \to T) \to E \to \mathbf{D}\{V\}T$
$(\lambda x,e,g.\{\langle w,g\rangle \mid \mathbf{smoke}w(xgwg)(egwg)\})$	smoking	$E \to V \to T$
$(\lambda x,g.\{\langle w,g\rangle \mid \mathbf{ling}w(xgwg)\})$	linguist	$E \to T$

Table 5.2: Abbreviations

$$\langle \textit{stopped}, \texttt{stopped}\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{vPROG}$$

The derivation of (54) may then be given as follows.

$$\dfrac{\dfrac{\dfrac{\langle \textit{every}, \texttt{every}\rangle :: \texttt{d/n} \quad \langle \textit{linguist}, \texttt{linguist}\rangle :: \texttt{n}}{\langle \textit{every linguist}, \texttt{everylinguist}\rangle :: \texttt{d}}(FA) \quad \dfrac{\langle \textit{stopped}, \texttt{stopped}\rangle :: (\texttt{d}\backslash\texttt{s})/\texttt{vPROG} \quad \langle \textit{smoking}, \texttt{smoking}\rangle :: \texttt{vPROG}}{\langle \textit{stopped smoking}, \texttt{stoppedsmoking}\rangle :: \texttt{vPROG}}(FA)}{\langle \textit{stopped smoking}, (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}}\rangle :: \texttt{vPROG}}(BA)}{\langle \textit{every linguist stopped smoking}, \texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}}\rangle :: \texttt{s}}\Uparrow$$
$$\dfrac{\langle \textit{every linguist stopped smoking}, (\texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1}\rangle :: \texttt{s}}{\langle \textit{every linguist stopped smoking}, \mu_{\mathbf{K}_{\epsilon,\{V\}}}(\texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1}\rangle :: \texttt{s}}\mu$$
$$\dfrac{\langle \textit{every linguist stopped smoking}, \mu_{\mathbf{K}_{\epsilon,\{V\}}}(\texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1})^{\Downarrow}\rangle :: \texttt{s}}{\langle \textit{every linguist stopped smoking}, (\mu_{\mathbf{K}_{\epsilon,\{V\}}}(\texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1})^{\Downarrow\Downarrow}\rangle :: \texttt{s}}\Downarrow$$

Expanding the result, we obtain the following.

$$\begin{aligned}
&(\mu_{\mathbf{K}_{\epsilon,\{V\}}}(\texttt{everylinguist} \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1})^{\Downarrow\Downarrow}\\
=\ &(\mu_{\mathbf{K}_{\epsilon,\{V\}}}((\lambda k.\texttt{not}(\texttt{alinguist} \gg\!=_{\mathbf{K}_{\epsilon,e}} (\lambda x.\texttt{not}(kx)))) \blacktriangleleft^* (\texttt{stoppedsmoking})^{\overset{\uparrow}{\kappa^{\uparrow}}})^{\Uparrow 1})^{\Downarrow\Downarrow}\\
=\ &(\mu_{\mathbf{K}_{\epsilon,\{V\}}}(\lambda k.\texttt{not}(\texttt{alinguist} \gg\!=_{\mathbf{K}_{\epsilon,e}} (\lambda x.\texttt{not}(k(\texttt{stoppedsmoking}x)))))^{\Uparrow 1})^{\Downarrow\Downarrow}\\
=\ &(\mu_{\mathbf{K}_{\epsilon,\{V\}}}(\lambda k.\texttt{not}(\texttt{alinguist} \gg\!=_{\mathbf{K}_{\epsilon,e}} (\lambda x.\texttt{not}(k(\texttt{stoppedsmoking}x)^{\Uparrow})))))^{\Downarrow\Downarrow}\\
=\ &(\lambda i,c.(c(\lambda g.\{\langle w,g\rangle \mid (\forall g'.(g[i]g', \mathbf{ling}wg_i') \to \mathbf{smoke}wg_i'(eg'wg'))\}) = \texttt{true}\\
&\qquad \Vdash_e (\lambda g.\{\langle w,g\rangle \mid (\forall g'.(g[i]g', \mathbf{ling}wg_i') \to (\exists e'.\mathbf{stop}w(eg'wg')g_i'e'))\})))
\end{aligned}$$

What we obtain is thus an information state with presuppositions in the graded monad **D**. The presuppositions we associate with (54) have an anaphoric component: the sentence seeks a dynamic event (i.e., something of type V), in order to give back an information state. What is presupposed about this dynamic event is that every way of updating an incoming assignment at

register i with a linguist results in an outgoing assignment, which, when the dynamic event is applied to it, yields a smoking event of which that linguist was the agent. The way this smoking event is determined, moreover, may vary with the relevant linguist. This is to say that every linguist must have smoked; though for each linguist, there may be a potentially different smoking event associated with that linguist. The at-issue content ascribed to (54), according to this result, is that, given the relevant dynamic event, every linguist stopped participating in the event gotten by applying it to the outgoing assignment; of course, this event will be an event of that linguist smoking, given the presupposition.

There are, perhaps, two things to say about this result. First, from an empirical point of view, it is a good one: (54) has a clear reading on which it presupposes that every linguist smoked. Moreover, because the present approach to presupposition triggers allows them to take scope, we predict ambiguous readings for sentences like (55).

(55) Ashley believes that every linguist stopped smoking.

(55) is predicted to have one of two possible presuppositions. Let's say that the presupposition trigger *stopped smoking* takes scope in the embedded clause; then (55) will presuppose that Ashley believes that every linguist smoked, while its at-issue content consists in the proposition that she believes that every linguist stopped the relevant event which she believes to be a smoking event of which that linguist was the agent. But, what if the presupposition trigger scopes up to the matrix clause? Then, (55) is predicted to presuppose that every linguist really did smoke, while its at-issue content is the proposition that Ashley believes that every linguist stopped the event of which they are presupposed to be the agent outside of the intensional context. In other words, it presupposes that every linguist smoked, and says that Ashley believes every linguist to have stopped an activity which, in actuality, was smoking.[4] This second reading persists, in spite of the fact that the scope of the distributive quantifier itself, *every linguist*, is trapped within the finite clause denoting the intensional context of Ashley's beliefs.

Second, although we appear to make the correct predictions about the presuppositions of sentences like (54) and (55), it is less clear how to treat

[4]There is a caveat to this discussion: that in order for the second reading to be available, *every linguist* must be understood with a *de re*, rather than a *de dicto*, reading. As a result, Ashley's belief is that everyone who is, in reality, a linguist stopped an event which, in reality, was smoking. If *every linguist* is understood *de dicto*, then there is no guarantee that Ashley's beliefs are about smoking events.

discourses in which those presuppostions are resolved, and thus filtered. (54) can be expanded as in (56), for example.

(56) Every linguist smoked previously, and every linguist stopped smoking.

The discourse in (56) appears to be presupposition-free. Given the truth conditions of the first clause, there must be a way of pairing each linguist with a smoking event of which that linguist was the agent. Nevertheless, we presently lack a means of determining a function from linguists to such events with which we may resolve the anaphoric presupposition registered by the second clause. Making such a function available for later anaphora appears to require a strikingly different analysis of eventive predicates, like that of the first clause. Improving the account to accommodate eventive anaphora will have to be left for the future, however.

Chapter 6

Conclusions

The scope-based perspective on natural language presupposition allows for options as to how a given presupposition is satisfied. The presupposition trigger *his brother* may register its existence presupposition inside the intensional context or the consequent of a conditional in which it finds itself; but, it may also hold out for a better context of evaluation—perhaps, the global context—in order to take scope *there*. A lesson that this dissertation has sought to demonstrate is that the problem of trapped presupposition triggers is not inherent to the satisfaction account of pressupposition, in the same way that natural language quantification is not an inherent problem of a general account of meaning composition based on functional application. To accommodate the latter, grammars based on functional application are enriched with new mechanisms that are added *on top of* functional application in a monotonic way, rendering these grammars more powerful; such new mechanisms may be, e.g., Quantifier Raising and predicate abstraction (Heim and Kratzer, 1998), type lifting operations (Jacobson, 1999; Steedman, 2000, i.a.), or enriched compositional schemes in which quantificational force is managed as a side effect (Shan, 2001; Barker, 2002; Shan, 2005; Barker and Shan, 2014, i.a.). Similarly, the satisfaction account of presupposition projection has been shown to provide a flexible, empirically adequate account of presupposition projection once it is enriched with a new mechanism that allows presupposition triggers to take scope.

Given the scopal analysis of presupposition triggers, a question arises about how the proposed semantically available readings of sentences with presupposition triggers eventuate in the readings which are actually observed. Why does the sentence (5b) *only* have a reading according to which it presupposes that Theo has a wetsuit, and not, in addition, another, weaker reading according to which it presupposes that Theo has a wetsuit if he has

a brother?

(5b) If Theo has a brother, he will bring his wetsuit.

This dissertation has not sought to answer this question, but has, instead, left it up to a pragmatic account of semantic ambiguity resolution to decide how a preferred interpretation is arrived at. Consistent with an account of (5b) according to which it features a genuine semantic ambiguity, it is possible to bring out the dispreferred reading using contextual support. Imagine that (5b) is instead utterred with the following context:

(57) Context: everyone's brother plans to give them a wetsuit as a holiday gift.
If Theo has a brother, he will bring his wetsuit.

The question remains, however, why such contextual support is required, and thus why (5b) lacks the weaker presupposition when utterred out of the blue. A pragmatic account according to which different theories of the interlocutors' background knowledge, and thus what they presuppose, may be assigned different probabilities by an interpreter is likely to play a role in guiding the resolution of semantically ambiguous presuppositions (see Beaver, 1999). Although the scope-based perspective argued for in this dissertation has not provided an account of exactly which presuppositions are observed for any given sentence utterred in any given context, it has rendered the presuppositions which actually are observed as grammatically available readings. No pragmatic deus ex machina is required, in order to modify the λ-terms generated by the grammar, thus providing a new λ-term that corresponds to a sentence's observed presuppositions; the λ-terms are already there! The role of pragmatics, given the current perspective, is to select a subset of the (finite) set of λ-terms generated by the grammar as the ones which correspond to available readings.

A topic which has not been touched upon in this dissertation is that of presupposition accommodation. What grammatical mechanism explains the observation that when presupposition-laden sentences are uttered in contexts in which their presuppositions are not satisfied, their presuppositions are treated as true by the interloctors? That is, why are presuppositions accommodated? As mentioned in chapter 1, de Groote and Lebedeva (2010) and Lebedeva (2012) have recently provided a formal account of presupposition accommodation in a grammar equipped with an exception-handling system. According to the account, exceptions are raised by anaphoric

presupposition triggers in discourse contexts in which their presuppositions are unsatisfied. These exceptions are then *handled* by adding an antecedent which satisfies the anaphor's presuppositions to the context. Within the Strawsonian perspective provided in this dissertation, failed presuppositions merely give rise the undifferentiated error #, corresponding to an "undefined" semantic value. A next step is thus to make the perspective more robust by incorporating an accommodation mechanism; such a mechanism will likely be informed by the insights of de Groote and Lebedeva and Lebedeva, by allowing presupposition failure to raise exceptions specific to the particular presupposition at hand. Such a combined approach may allow presupposition triggers to take scope within particular contexts, even while their presuppositions are accommodated in contexts "higher up". The linguistic consequences of such an approach will have to be worked out in future work.

Finally, while the main contribution of this dissertation has been to present a particular version of the satisfaction theory—one unbeset with a difficulty it has faced in the past—perhaps, a more far-reaching side effect of the present proposal is to demonstrate two particular advantages of the monadic approach to semantic composition. First, because the approach is modular, it presents the possibility of studying presupposition from the point of view of other dynamic semantic frameworks. Another approach might attempt to integrate some of the proposals made here for presupposition with an approach to other dynamic phenomena, like indefiniteness; e.g., those found in de Groote 2006 and Charlow 2014. Second, because the crucial property of the compositional scheme introduced here is that it gives rise to a graded monad, one may attempt to instead model presupposition in terms of another (graded) monad, while enjoying the same basic advantages of scope-taking. To regard presupposition as a kind of monadic side effect is to allow presupposition triggers to take scope. Future research within the satisfaction account of presupposition is thus likely to benefit from the guidance of the monad laws.

References

Atlas, Jay. 1976. On the semantics of presupposition and negation: an essay in philosophical logic and the foundations of linguistics. Doctoral Dissertation, Princeton University.

Atlas, Jay. 1977. Negation, ambiguity and presupposition. *Linguistics and Philosophy* 1:321–336.

Atlas, Jay. 1979. How linguistics matters to philosophy: Presupposition, truth, and meaning. In *Syntax and semantics*, ed. Choon-Kyu Oh and David A. Dinneen, volume 11: Presupposition, truth, and meaning, 265–281. New York: Academic Press.

Atlas, Jay, and Stephen Levinson. 1981. *It*-clefts, informativeness and logical form: Radical pragmatics. In *Radical pragmatics*, ed. Peter Cole, 1–61. New York: Academic Press.

Bar-Hillel, Yehoshua. 1953. A quasi-arithmetical notation for syntactic description. *Language* 29:47–58.

Barendregt, Henk Pieter, Wil Dekkers, and Richard Statman. 2013. *Lambda calculus with types*. Perspectives in Logic. New York: Cambridge University Press.

Barker, Chris. 2002. Continuations and the nature of quantification. *Natural Language Semantics* 10:211–242.

Barker, Chris. 2015. Scope. In *Handbook of contemporary semantics*, ed. Shalom Lappin and Chris Fox, 47–87. Wiley-Blackwell, second edition.

Barker, Chris, Raffaella Bernardi, and Chung-chieh Shan. 2010. Principles of interdimensional meaning interaction. In *Proceedings of the 20th Conference on Semantics and Linguistic Theory*, ed. Nan Li and David Lutz,

109–127. Ithaca, NY: Cornell.

Barker, Chris, and Chung-chieh Shan. 2008. Donkey anaphora is in-scope binding. *Semantics and Pragmatics* 1:1–46.

Barker, Chris, and Chung-chieh Shan. 2014. *Continuations and natural language*. Oxford Studies in Theoretical Linguistics. Oxford: Oxford University Press.

Beaver, David, and Emiel Krahmer. 2001. A partial account of presupposition projection. *Journal of Logic, Language and Information* 10:147–182.

Beaver, David I. 1999. Presupposition accomodation: A plea for common sense. In *Logic, language, and computation*, ed. Lawrence S. Moss, Jonathan Ginzburg, and de Rijke 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.

Bekki, Daisuke. 2014. Representing anaphora with dependent types. In *Logical aspects of computational linguistics*, ed. Nicholas Asher and Sergei Soloviev, volume 8535 of *Lecture Notes in Computer Science*, 14–29. Heidelberg: Springer.

Bekki, Daisuke, and E. McCready. 2014. CI via DTS. In *Proceedings of LENLS 11*, 110–123. Tokyo.

Bekki, Daisuke, and Miho Satoh. 2015. Calculating projections via type checking. In *Proceedings of TYTLES in ESSLI 27*. Barcelona.

Bochvar, D. A. 1939. On a three valued calculus and its application to the analysis of contradictories. *Matematicheskii Sbornik* 4:287–308.

Böhm, Corrado, and Alessandro Berarducci. 1985. Automatic synthesis of typed Λ-programs on term algebras. *Theoretical Computer Science* 39:135–154.

Charlow, Simon. 2014. On the semantics of exceptional scope. Doctoral Dissertation, NYU, New York.

Chierchia, Gennaro, and Sally McConnell-Ginet. 1990. *Meaning and gram-*

mar: An introduction to semantics. Cambridge: MIT Press.

Chomsky, Noam. 1981. *Lectures on government and binding*. Dordrecht: Foris.

Chomsky, Noam. 1995. *The minimalist program*. Cambridge: MIT Press.

Chung, Sandra, William A. Ladusaw, and James McCloskey. 1995. Sluicing and logical form. *Natural Language Semantics* 3:239–282.

Coppock, Elizabeth, and David Beaver. 2015. Definiteness and determinacy. *Linguistics and Philosophy* 38:377–435.

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.

von Fintel, Kai. 2004. Would you believe it? the king of France is back! presuppositions and truth-value intuitions. In *Descriptions and beyond*, ed. Marga Reimer and Anne Bezuidenhout, chapter 8, 269–296. Oxford: Oxford University Press.

von Fintel, Kai. 2008. What is presupposition accommodation, again? *Philosophical Perspectives* 22.

Francez, Itamar. 2018. How not to project the satisfaction theory of projection (on Karttunen) or: Who has a proviso problem? In *Tokens of meaning: Papers in honor of Lauri Karttunen*, ed. Cleo A. Condoravdi and Tracy Holloway King, volume 22 of *CSLI Lecture Notes*. Stanford: CSLI Publications.

Frege, Gottlob. 1892 [1952]. On sense and reference. In *Translations from the philosophical writings of gottlob frege*, ed. Peter T. Geach and Max Black, 56–78. Oxford: Blackwell.

Frege, Gottlob. 1967 [1879]. Begriffsschrift, a formula language, modelled upon that of arithmetic, for pure thought. In *From Frege to Gödel: A source book in mathematical logic*, ed. Jean van Heijenoort. Cambridge: Harvard University Press.

Fujii, Soichiro, Shin-ya Katsumata, and Paul-André Melliès. 2016. Towards

a formal theory of graded monads. In *Proceedings of the 20th International Conference on Foundations of Software Science and Computation Structures*, ed. Bart Jacobs and Christof Löding, number 9634 in Lecture Notes in Computer Science, 513–530.

Gazdar, Gerald. 1979. *Pragmatics: Implicature, presupposition and logical form*. New York: Academic Press.

Gentzen, Gerhard. 1964 [1934]. Investigations into logical deduction. *American Philosophical Quarterly* 1:288–306.

Geurts, Bart. 1996. Local satisfaction guaranteed: A presupposition theory and its problems. *Linguistics and Philosophy* 19:259–294.

Giorgolo, Gianluca, and Ash Asudeh. 2012. $\langle M, \eta, \star$ monads for conventional implicature. In *Proceedings of Sinn und Bedeutung 16*, ed. Ana Aguilar Guevara, Anna Chernilovskaya, and Rick Nouwen, MITWPL, 265–278.

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.

Grice, Paul H. 1975. Logic and conversation. In *Syntax and semantics*, ed. Peter Cole and Jerry L. Morgan, volume 3: Speech Acts, 41–58. New York: Academic Press.

Groenendijk, Jeroen A., and Martin B. J. Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy* 14:39–100.

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.

de Groote, Philippe, and Makoto Kanazawa. 2013. A note on intensionalization. *Journal of Logic, Language and Information* 22:173–194.

de Groote, Philippe, and Ekaterina Lebedeva. 2010. Presupposition accommodation as exception handling. In *Proceedings of SIGDIAL 2010: the 11th Annual Meeting of the Special Interest Group on Discourse and Dia-*

logue, 71–74. Tokyo: Association for Computational Linguistics.

Hankamer, Jorge, and Ivan Sag. 1976. Deep and surface anaphora. *Linguistic Inquiry* 7:391–426.

Hardt, Daniel. 1993. Verb phrase ellipsis: Form, meaning, and processing. Doctoral Dissertation, University of Pennsylvania, Philadelphia.

Heim, Irene. 1982. The semantics of definite and indefinite noun phrases. Doctoral Dissertation, University of Massachussetts, Amherst.

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. 1991. Artikel und definitheit [articles and definiteness]. In *Semantik: Ein internationales handbuch der zeitgen ossischen forschung*, ed. Arnim von Stechow and Dieter Wunderlich, 487–535. Berlin: de Gruyter.

Heim, Irene. 1992. Presupposition projection and the semantics of attitude verbs. *Journal of Semantics* 9:183–221.

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.

Jacobson, Pauline. 1999. Towards a variable-free semantics. *Linguistics and Philosophy* 22:117–184.

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.

Kamp, Hans. 1988. Comments on Stalnaker, belief attribution and context. In *Contents of Thought: Proceedings of the 1985 Oberlin Colloquium in Philosophy*, ed. Robert H. Grimm and Daniel D. Merrill, 3, 156–181. University of Arizona Press.

Karttunen, Lauri. 1973. Presuppositions of compound sentences. *Linguistic Inquiry* 4:169–193.

Karttunen, Lauri. 1974. Presuppositions and linguistic context. *Theoretical Linguistics* 1:181–194.

Karttunen, Lauri, and Stanley Peters. 1979. Conventional implicature. In *Syntax and semantics*, ed. Choon-Kyu Oh and David A. Dinneen, volume 11, 1–56. New York: Academic Press.

Katsumata, Shin-ya. 2014. Parametric effect monads and semantics of effect systems. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. San Diego.

Kempson, Ruth. 1975. *Presupposition and the delimitation of semantics*. Cambridge: Cambridge University Press.

Kleene, Stephen Cole. 1938. On a notation for ordinal numbers. *Journal of Symbolic Logic* 3:150–155.

Kleene, Stephen Cole. 1959. *Introduction to metamathematics*. Amsterdam: North Holland.

Kobele, Gregory M. 2015. LF-copying without LF. *Lingua* 166, Part B:236–259.

Kobele, Gregory M. 2018. The Cooper storage idiom. *Journal of Logic, Language and Information* 1–37.

Kubota, Yusuke, and Wataru Uegaki. 2009. Continuation-based semantics for conventional implicatures: The case of Japanese benefactives. In *Proceedings of the 19th Conference on Semantics and Linguistic Theory*, ed. Satoshi Ito and Ed Cormany, 306–323. Ithaca, NY: LSA and CLC Publications.

Langendoen, Donald Terence, and Harris B. Savin. 1971. The projection problem for presuppositions. In *Studies in linguistic semantics*, ed. Charles J.. Fillmore and Donald Terence Langendoen, 55–60. Holt, Rinehart & Winston.

Lassiter, Daniel. 2012. Presuppositions, provisos, and probability. *Seman-*

tics and Pragmatics 5:1–37.

Lebedeva, Ekaterina. 2012. Expressing discourse dynamics through continuations. Doctoral Dissertation, Université de Lorraine, Lorraine.

Lewis, David. 1979. Scorekeeping in a language game. In *Semantics from different points of view*, ed. Rainer Bäurele, Springer Series in Language and Communication, 172–187. Springer.

Martin, John N. 1979. Some misconceptions in the critique of semantic presupposition. *Theoretical Linguistics* 6:235–282.

Martin-Löf, Per. 1984. Intuitionistic type theory. Notes of Giovanni Samblin on a series of lectures given in Padua, June 1980.

May, Robert Carlen. 1978. The grammar of quantification. Doctoral Dissertation, MIT, Cambridge.

McBride, Conor, and Ross Paterson. 2008. Applicative programming with effects. *Journal of Functional Programming* 18:1–13.

Merchant, Jason. 2001. *The syntax of silence: Sluicing, islands, and the theory of ellipsis*. Oxford Studies in Theoretical Linguistics. Oxford: Oxford University Press.

Merchant, Jason. 2013. Voice and ellipsis. *Linguistic Inquiry* 44:77–108.

Merchant, Jason. 2019. Ellipsis: A survey of analytical approaches. In *The Oxford handbook of ellipsis*, ed. Jeroen van Craenenbroeck and Tanja Temmerman, 19–45. Oxford: Oxford University Press.

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.

Montague, Richard. 1970. Universal grammar. *Theoria* 36:373–398.

Montague, Richard. 1974. The proper treatment of quantification in ordinary English. In *Formal philosophy*, ed. Richmond H. Thomason, 247–270. Yale University Press.

Muskens, Reinhard. 1996. Combining Montague semantics and discourse

representation. *Linguistics and Philosophy* 19:143–186.

Orchard, Dominic A., Tomas Petricek, and Alan Mycroft. 2014. The semantic marriage of monads and effects. *CoRR* abs/1401.5391. URL http://arxiv.org/abs/1401.5391.

Potts, Christopher. 2005. *The logic of conventional implicatures*. Oxford: Oxford University Press.

Roberts, Craige. 1989. Modal subordination and pronominal anaphora in discourse. *Linguistics and Philosophy* 12:683–721.

van Rooij, Robert. 2007. Strengthening conditional presuppositions. *Journal of Semantics* 24:289–304.

Rothschild, Daniel. 2011. Explaining presupposition projection with dynamic semantics. *Semantics and Pragmatics* 4:1–43.

Russell, Bertrand. 1905. On denoting. *Mind* 14:479–493.

Sag, Ivan A. 1976. Deletion and logical form. Doctoral Dissertation, MIT, Cambridge.

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.

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.

Schlenker, Philippe. 2009. Local contexts. *Semantics and Pragmatics* 2:1–78.

Schlenker, Phillippe. 2007. Anti-dynamics: Presupposition projection without dynamic semantics. *Journal of Logic, Language and Information* 16:325–356.

Schlenker, Phillippe. 2008. Be articulate: A pragmatic theory of presupposition. *Theoretical Linguistics* 34:157–212.

Schlenker, Phillippe. 2011. The proviso problem: a note. *Natural Language Semantics* 19:395–422.

Shan, Chung-chieh. 2001. Monads for natural language semantics. In *Proceedings of the ESSLLI 2001 Student Session*, ed. Kristina Striegnitz. Helsinski: 13th European Summer School in Logic, Language, and Information.

Shan, Chung-chieh. 2005. Linguistic side effects. Doctoral Dissertation, Harvard University, Cambridge.

Shannon, Benny. 1976. On two kinds of presuppositions in natural language. *Foundations of Language* 14:247–249.

Singh, Raj. 2007. Formal alternatives as a solution to the proviso problem. In *Proceedings of the 17th Conference on Semantics and Linguistic Theory*, ed. Tova Friedman and Masayuki Gibson, 264–281. Ithaca, NY: Cornell.

Singh, Raj. 2008. Modularity and locality in interpretation. Doctoral Dissertation, MIT, Cambridge.

Soames, Scott. 1979. A projection problem for speaker presuppositions. *Linguistic Inquiry* 10:623–666.

Stalnaker, Robert. 1972. Pragmatics. In *Semantics of natural language*, ed. Donald Davidson and Gilbert Harman, 389–408. Dordrecht: Reidel.

Stalnaker, Robert. 1973. Presuppositions. *Journal of Philosophical Logic* 2:447–457.

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.

Stalnaker, Robert. 1978. Context. In *Pragmatics*, ed. Milton K. Munitz and Peter K. Unger, 78–95. New York: New York University Press.

Stalnaker, Robert. 1998. On the representation of context. *Journal of Logic, Language and Information* 7:3–19.

Steedman, Mark. 2000. *The syntactic process*. MIT Press.

Strawson, Peter F. 1950. On referring. *Mind* 59:320–344.

Tanaka, Ribeka, Koji Mineshima, and Daisuke Bekki. 2014. Resolving modal anaphora in dependent type semantics. In *Proceedings of the 11th International Workshop on Logic and Engineering of Natural Language Semantics*, JSAI International Symposia on AI, 43–56. Tokyo.

Tanaka, Ribeka, Koji Mineshima, and Daisuke Bekki. 2015. Factivity and presupposition in dependent type semantics. In *Proceedings of TYpe Theory and LExical Semantics*. ESSLI2015.

Wadler, Philip. 1992a. Comprehending monads. In *Mathematical structures in computer science*, volume 2, 461–493. Cambridge University Press.

Wadler, Philip. 1992b. The essence of functional programming. In *Proceedings of the 19th ACM SIGPLAN-SICACT symposium on Principles of programming languages*, 1–14.

Wadler, Philip. 1994. Monads and composable continuations. *Lisp and Symbolic Computation* 7:39–56.

Wadler, Philip. 1995. Monads for functional programming. In *Advances in functional programming*, ed. J. Jeuring and E. Meijer, volume 925 of *Lecture Notes in Computer Science*. Heidelberg: Springer.

Wilson, Deidre. 1975. *Presupposition and non-truth-conditional semantics*. London: Academic Press.

Zeevat, Henk. 1992. Presupposition and accommodation in update semantics. *Journal of Semantics* 9:379–412.

Appendix A

The full fragment

A.1 The simply typed λ-calculus

Here we present the metalanguage in terms of the simply typed λ-calculus with Church-style typing (Barendregt et al., 2013), in conjunction with logical and non-logical constants. We define a set of types

$$\mathcal{T} ::= \mathsf{At} \mid \mathcal{T} \to \mathcal{T}$$
$$\mathsf{At} ::= e \mid v \mid t \mid s \mid i \mid \#$$

consisting of atomic types (At) and implication types. Simpy-typed λ-terms are elements of the following set Λ,

$$\Lambda = \bigcup_{\alpha \in \mathcal{T}} \Lambda_\alpha$$

where the sets Λ_α are defined via the following typing rules. We have the family of denumerably infinite sets $\{Var_\alpha\}_{\alpha \in \mathcal{T}}$, whose members are pairwise disjoint, along with a set Con and a function $\tau : Con \to \mathcal{T}$.

$$\frac{v^\alpha \in Var_\alpha}{\Gamma, v^\alpha : \alpha \vdash v^\alpha : \alpha}\, Ax \qquad \frac{c \in Con}{\Gamma \vdash \tau(c)}\, Con$$

$$\frac{\Gamma, v^\alpha : \alpha \vdash M : \beta}{\Gamma \vdash (\lambda v^\alpha . M) : \alpha \to \beta} \to I \qquad \frac{\Gamma \vdash M : (\alpha \to \beta) \qquad \Gamma \vdash N : \alpha}{\Gamma \vdash (MN) : \beta} \to E$$

Any Λ_α is defined as $\{t \mid (\exists \Gamma . \Gamma \vdash t : \alpha)\}$. We assume the usual notions of α-, β-, and η-conversion on simply-typed λ-terms. Meanwhile, we also assume the usual shorthands for terms and types, taking application to associate to the left (i.e., $MNO := ((MN)O)$) and implication to associate to the right

(i.e., $\alpha \to \beta \to \gamma := (\alpha \to (\beta \to \gamma))$). We may additionally omit type superscripts from variable names if doing so does not introduce ambiguity.

The logical constants are the following, presented along with their type-signatures. Note that there are denumerably infinite sets of constants $(=)$, $\forall$, and $\exists$, one for each type α. Constants presented as infixes between their two arguments are wrapped in parentheses.

$$\begin{aligned}(=) &: \alpha \to \alpha \to t\\ \forall &: (\alpha \to t) \to t\\ \exists &: (\alpha \to t) \to t\\ (\wedge) &: t \to t \to t\\ (\vee) &: t \to t \to t\\ (\to) &: t \to t \to t\\ \neg &: t \to t\\ \top &: t\\ \bot &: t\end{aligned}$$

We generally adopt the convention of writing terms $\forall(\lambda x.t)$ as $(\forall x.t)$, and likewise for $\exists$.

We define the family of domains $\{D_\alpha\}_{\alpha\in\mathcal{T}}$, so that $\{D_a\}_{a\in\mathsf{At}}$ are non-empty sets, $D_t = \{T, F\}$, and $D_{\alpha\to\beta}$ is the set of total functions from D_α to D_β. A model $\mathcal{M}$ for Λ is a pair $\langle\{D_\alpha\}_{\alpha\in\mathcal{T}}, I\rangle$, where $I : \bigcup_{\alpha\in\mathcal{T}} Con_\alpha \to \bigcup_{\alpha\in\mathcal{T}} D_\alpha$, with the constraint that for each $c \in Con_\alpha$, $I(c) \in D_\alpha$.

An assignment is a function $g : \bigcup_{\alpha\in\mathcal{T}} Var_\alpha \to \bigcup_{\alpha\in\mathcal{T}} D_\alpha$ with the constraint that for each $v^\alpha \in Var_\alpha$, $g(v^\alpha) \in D_\alpha$. Assuming $a \in D_\alpha$, we write $g[a/v^\alpha]$ for the assignment behaving exactly as g, but which takes v^α onto a. Given a model $\mathcal{M}$, and an assignment g, we define the interpretation function $[\![\cdot]\!]_{\mathcal{M},g} : \Lambda \to \bigcup_{\alpha\in\mathcal{T}} D_\alpha$ as follows. If $c \in Con_\alpha$ is a non-logical constant for some $\alpha \in \mathcal{T}$, $[\![c]\!]_{\mathcal{M},g} = I(c)$. If $v^\alpha \in Var_\alpha$ for some $\alpha \in \mathcal{T}$, $[\![v^\alpha]\!]_{\mathcal{M},g} = g(v^\alpha)$. If $v^\alpha \in Var_\alpha$ and $t \in \Lambda_\beta$, $[\![(\lambda v^\alpha.t)]\!]_{\mathcal{M},g}$ is that function $f : D_\alpha \to D_\beta$ such that, for any $a \in D_\alpha$, $f(a) = [\![t]\!]_{\mathcal{M},g[a/v^\alpha]}$. Finally, $[\![(t_1\,t_2)]\!]_{\mathcal{M},g} = [\![t_1]\!]_{\mathcal{M},g}([\![t_2]\!]_{\mathcal{M},g})$.

Logical constants are interpreted as follows. $[\![(t_1 \wedge t_2)]\!]_{\mathcal{M},g} = T$ iff $[\![t_1]\!]_{\mathcal{M},g} = T$ and $[\![t_2]\!]_{\mathcal{M},g} = T$; $[\![(t_1 \vee t_2)]\!]_{\mathcal{M},g} = T$ iff $[\![t_1]\!]_{\mathcal{M},g} = T$ or $[\![t_2]\!]_{\mathcal{M},g} = T$; $[\![(t_1 \to t_2)]\!]_{\mathcal{M},g} = T$ iff $[\![t_1]\!]_{\mathcal{M},g} = F$ or $[\![t_2]\!]_{\mathcal{M},g} = T$. $[\![\neg t_1]\!]_{\mathcal{M},g} = T$ iff $[\![t]\!]_{\mathcal{M},g} = F$. If $t \in \Lambda_{\alpha\to t}$ for some $\alpha \in \mathcal{T}$, then $[\![(\forall t)]\!]_{\mathcal{M},g} = T$ iff $[\![t]\!]_{\mathcal{M},g}(a) = T$ for every $a \in D_\alpha$. And $[\![(\exists t)]\!]_{\mathcal{M},g} = [\![\neg(\forall(\lambda x^\alpha.\neg(t x)))]\!]_{\mathcal{M},g}$ (for any $x^\alpha \in Var_\alpha$ not free in t). If $t_1, t_2 \in \Lambda_\alpha$ for some $\alpha \in \mathcal{T}$, then $[\![(t_1 = t_2)]\!]_{\mathcal{M},g} = T$ iff $[\![t_1]\!]_{\mathcal{M},g} = [\![t_2]\!]_{\mathcal{M},g}$. Finally, $[\![\top]\!]_{\mathcal{M},g} = T$ and $[\![\bot]\!]_{\mathcal{M},g} = F$.

Here we present the non-logical constants, along with their type signatures. We use o to designate an arbitrary result type. We abbreviate the types $(i \to e) \to s \to (i \to e) \to e$, $(i \to e) \to s \to (i \to e) \to v$, and $(i \to e) \to s \to (i \to e) \to t$ as E, V, and T, respectively. Note that we have a denumerably infinite set of constants ($\Vdash$), one for each pair of types α and o.

$$(\Vdash) : t \to \alpha \to (\alpha \to o) \to (\# \to o) \to o$$
$$\# : \#$$
$$(\cdot)' : i \to i$$
$$(+) : i \to i \to i$$
$$(-) : i \to i \to i$$
$$(\leq) : i \to i \to t$$
$$\mathbf{a} : e$$
$$\mathbf{begin} : s \to (v \to t) \to e \to v \to t$$
$$\mathbf{believe} : s \to T \to e \to t$$
$$\mathbf{bring} : s \to e \to e \to v \to t$$
$$\mathbf{brother} : s \to e \to t$$
$$\mathbf{come} : s \to e \to v \to t$$
$$\mathbf{diver} : s \to e \to t$$
$$\mathbf{dog} : s \to e \to t$$
$$\mathbf{dox} : e \to s \to T$$
$$\mathbf{drenched} : s \to e \to t$$
$$\mathbf{e} : e$$
$$\mathbf{eat} : s \to e \to e \to v \to t$$
$$\mathbf{exists} : s \to e \to t$$
$$\mathbf{fem} : s \to e \to t$$
$$\mathbf{garden} : s \to e \to v \to t$$
$$\mathbf{girl} : s \to e \to t$$
$$\mathbf{have} : s \to e \to e \to t$$
$$\mathbf{healthy} : s \to e \to t$$
$$\mathbf{i} : i$$
$$\mathbf{j} : e$$
$$\mathbf{ling} : s \to e \to t$$
$$\mathbf{m} : e$$
$$\mathbf{man} : s \to e \to t$$
$$\mathbf{masc} : s \to e \to t$$
$$\mathbf{mod} : s \to T$$
$$\mathbf{nice} : s \to e \to t$$
$$\mathbf{see} : s \to e \to e \to v \to t$$
$$\mathbf{sit} : s \to e \to v \to t$$
$$\mathbf{sleep} : s \to e \to v \to t$$
$$\mathbf{smkg} : s \to e$$
$$\mathbf{smoke} : s \to e \to v \to t$$
$$\mathbf{stop} : s \to v \to e \to v \to t$$
$$\mathbf{tc} : s \to e \to t$$
$$\mathbf{walk} : s \to e \to v \to t$$
$$\mathbf{wetsuit} : s \to e \to t$$

Given these constants, we have the following axioms, which are meant to

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{D}_n}} : \ldots \mathbf{D} \epsilon \alpha \ldots} \uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{D} e (\mathbf{D} f \alpha) \ldots}{\Gamma \vdash \mu_{\mathbf{D}_n e,f} m : \ldots \mathbf{D}(e+f)\alpha \ldots} \mu$$

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{K}_n}} : \ldots \mathbf{K} \epsilon \alpha \ldots} \uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K} e (\mathbf{K} f \alpha) \ldots}{\Gamma \vdash \mu_{\mathbf{K}_n e,f} m : \ldots \mathbf{K}(e+f)\alpha \ldots} \mu$$

$$\frac{\Gamma \vdash a : \ldots \alpha \ldots}{\Gamma \vdash a^{\overset{\uparrow}{\mathbf{K}^{\uparrow}_n}} : \ldots \mathbf{K}^{\uparrow} \epsilon \alpha \ldots} \uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}^{\uparrow} e (\mathbf{K}^{\uparrow} f \alpha) \ldots}{\Gamma \vdash \mu_{\mathbf{K}^{\uparrow}_n e,f} m : \ldots \mathbf{K}^{\uparrow}(e+f)\alpha \ldots} \mu$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{D} e \alpha \ldots}{\Gamma \vdash m^{\Uparrow n} : \ldots \mathbf{K} e \alpha \ldots} \Uparrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K} e \alpha \ldots}{\Gamma \vdash m^{\Uparrow n} : \ldots \mathbf{K}^{\uparrow} e \alpha \ldots} \Uparrow$$

$$\frac{\Gamma \vdash m : \ldots \mathbf{K} e T \ldots}{\Gamma \vdash m^{\Downarrow n} : \ldots \mathbf{D} e T \ldots} \Downarrow \qquad \frac{\Gamma \vdash m : \ldots \mathbf{K}^{\uparrow} e T \ldots}{\Gamma \vdash m^{\Downarrow n} : \ldots \mathbf{K} e T \ldots} \Downarrow$$

$$\frac{\Gamma \vdash m : \mathbf{D}(\mathbf{insert}_n a e)\alpha \qquad \Gamma \vdash t : a}{\Gamma \vdash \mathsf{anaph}_n a e t m : \mathbf{D} e \alpha} \mathit{Anaph}$$

Figure A.1: Dynamic inference rules

be understood as restricting the class of models in which Λ is interpreted.

$$(\forall x^{\alpha}, s^{\alpha \to o}, f^{\# \to o}.(\top \Vdash x) s f = s x) \qquad \text{(Succeed)}$$

$$(\forall x^{\alpha}, s^{\alpha \to o}, f^{\# \to o}.(\bot \Vdash x) s f = f \#) \qquad \text{(Fail)}$$

$$(\forall m^{i}.\mathbf{i} + m = m) \qquad (+1)$$

$$(\forall n^{i}, m^{i}.n' + m = (n+m)') \qquad (+2)$$

$$(\forall n^{i}.\mathbf{i} - n = \mathbf{i}) \qquad (-1)$$

$$(\forall n^{i}, m^{i}.n' - m' = n - m) \qquad (-2)$$

$$(\forall n^{i}.\mathbf{i} \leq n) \qquad (\leq 1)$$

$$(\forall m^{i}, n^{i}.m' \leq n' \leftrightarrow m \leq n) \qquad (\leq 2)$$

A.2 Grammar

The grammar is given by the inference rules in Figure A.1 and the rules of Forward and Backward Application. We define these by introducing the graded monad **P**, along with the graded monad transformers **RT** and **CT** (through definitions repeated from the main text).

Definition 5 (P).

$$\mathbf{P} : \mathcal{T}^* \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{P}\epsilon\alpha = (\alpha \to o) \to (\# \to o) \to o$$
$$\mathbf{P}(ae)\alpha = a \to \mathbf{P}e\alpha$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{P}}} : \alpha \to \mathbf{P}\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{P}}} = (\lambda s, f.sa)$$

$$(\cdot)^{\overset{\mathbf{P}}{\downarrow}}_{e,f} : \mathbf{P}e(\alpha \to \beta) \to \mathbf{P}f\alpha \to \mathbf{P}(e+f)\beta$$
$$m^{\overset{\mathbf{P}}{\downarrow}}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}} = (\lambda n, x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ mx_1, \ldots, x_m(\lambda u.ny_1, \ldots, y_n(\lambda v.s(uv))f)f)$$

$$\mu_{\mathbf{P}e,f} : \mathbf{P}e(\mathbf{P}f\alpha) \to \mathbf{P}(e+f)\alpha$$
$$\mu_{\mathbf{P}\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}}m = (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ mx_1, \ldots, x_m(\lambda n.ny_1, \ldots, y_n sf)f)$$

Definition 12 (RT).

$$\mathbf{RT} : \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{RT}(r)(\mathbf{E_0})e\alpha = r \to \mathbf{E_0}e\alpha$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} : \alpha \to \mathbf{RT}(r)(\mathbf{E_0})\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} = (\lambda s.a^{\overset{\uparrow}{\mathbf{E_0}}})$$

$$(\cdot)^{\overset{\mathbf{RT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} : \mathbf{RT}(r)(\mathbf{E_0})e(\alpha \to \beta) \to \mathbf{RT}(r)(\mathbf{E_0})f\alpha \to \mathbf{RT}(r)(\mathbf{E_0})(e+f)\beta$$
$$u^{\overset{\mathbf{RT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} = (\lambda v, s.(us)^{\overset{\mathbf{E_0}}{\downarrow}}_{e,f}(vs))$$

$$\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}} : \mathbf{RT}(r)(\mathbf{E_0})e(\mathbf{RT}(r)(\mathbf{E_0})f\alpha) \to \mathbf{RT}(r)(\mathbf{E_0})(e+f)\alpha$$
$$\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}}m = (\lambda s.\mu_{\mathbf{E_0}e,f}((\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\overset{\mathbf{E_0}}{\downarrow}}_{\epsilon,e}(ms)))$$

Definition 17 (**CT**)**.**

$$\mathbf{CT} : \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{CT}(r)(\mathbf{E_0})e\alpha = (\alpha \to \mathbf{E_0} f\, r) \to \mathbf{E_0}(e+f)r$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} : \alpha \to \mathbf{CT}(r)(\mathbf{E_0})\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} = (\lambda k.ka)$$

$$(\cdot)^{\overset{\mathbf{CT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} : \mathbf{CT}(r)(\mathbf{E_0})e(\alpha \to \beta) \to \mathbf{CT}(r)(\mathbf{E_0})f\,\alpha \to \mathbf{CT}(r)(\mathbf{E_0})(e+f)\beta$$
$$u^{\overset{\mathbf{CT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} = (\lambda v, k.u(\lambda U.v(\lambda V.k(U\,V))))$$

$$\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}} : \mathbf{CT}(r)(\mathbf{E_0})e(\mathbf{CT}(r)(\mathbf{E_0})f\,\alpha) \to \mathbf{CT}(r)(\mathbf{E_0})(e+f)\alpha$$
$$\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}}\, m = (\lambda k.m(\lambda k'.k'k))$$

We may then define the graded monad $\mathbf{D}$ as $\mathbf{RT}(i)(\mathbf{RT}(T \to T)(\mathbf{P}))$, $\mathbf{K}$ as $\mathbf{CT}(T)(\mathbf{D})$, and $\mathbf{K}^{\uparrow}$ as $\mathbf{CT}(T)(\mathbf{K})$. Forward and Backward Application are then defined in terms of the following operators and the syntactic categories in Definition 6 (repeated below).

$$(\triangleright^*) ::= (\triangleright) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{D}}}_{\epsilon,e} u)^{\mathbf{D}}_{e,f} v) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{K}}}_{\epsilon,e} u)^{\mathbf{D}}_{e,f} v) \mid (\lambda u, v.((\triangleright^*)^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}_{\epsilon,e} u)^{\mathbf{K}^{\uparrow}}_{e,f} v)$$
$$(\triangleleft^*) ::= (\triangleleft) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{D}}}_{\epsilon,e} u)^{\mathbf{D}}_{e,f} v) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{K}}}_{\epsilon,e} u)^{\mathbf{D}}_{e,f} v) \mid (\lambda u, v.((\triangleleft^*)^{\overset{\uparrow}{\mathbf{K}^{\uparrow}}}_{\epsilon,e} u)^{\mathbf{K}^{\uparrow}}_{e,f} v)$$

Definition 6 (Syntactic categories)**.**

$$\mathcal{C} ::= \mathtt{Prim} \mid (\mathcal{C}/\mathcal{C}) \mid (\mathcal{C}\backslash\mathcal{C})$$
$$\mathtt{Prim} ::= \mathsf{n} \mid \mathsf{d} \mid \mathsf{a} \mid \mathsf{v} \mid \mathsf{vPROG} \mid \mathsf{vINF} \mid \mathsf{s}$$

Definition 7 (Forward and Backward Application)**.**

$$\frac{\langle w, m\rangle :: \mathsf{x/y} \qquad \langle v, n\rangle :: \mathsf{y}}{\langle wv, m \triangleright^* n\rangle :: \mathsf{x}}\,(FA)$$
$$\frac{\langle w, m\rangle :: \mathsf{y} \qquad \langle v, n\rangle :: \mathsf{y\backslash x}}{\langle wv, m \triangleleft^* n\rangle :: \mathsf{x}}\,(BA)$$

Recall, for the following definitions, that $m \gg\!=_{\mathbf{E}e,f} k$ is defined as $\mu_{\mathbf{E}e,f}(k \overset{\uparrow}{\mathbf{E}} \underset{\epsilon,e}{\downarrow} m)$ (for any graded monad $\mathbf{E}$).

Definition 20 ($(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$)**.**

$$
\begin{aligned}
(\cdot)^{\Uparrow} &: \mathbf{D}e\alpha \to \mathbf{K}e\alpha \\
m^{\Uparrow} &= (\lambda k.m \gg\!=_{\mathbf{D}e,f} k) \\
(\cdot)^{\Downarrow} &: \mathbf{K}eT \to \mathbf{D}eT \\
m^{\Downarrow} &= m(\lambda x.x \overset{\uparrow}{\mathbf{D}})
\end{aligned}
$$

Definition 24 ($(\cdot)^{\Uparrow}$ and $(\cdot)^{\Downarrow}$)**.**

$$
\begin{aligned}
(\cdot)^{\Uparrow} &: \mathbf{K}e\alpha \to \mathbf{K}^{\uparrow}e\alpha \\
m^{\Uparrow} &= (\lambda k.m \gg\!=_{\mathbf{K}e,f} k) \\
(\cdot)^{\Downarrow} &: \mathbf{K}^{\uparrow}eT \to \mathbf{K}eT \\
m^{\Downarrow} &= m(\lambda x.x \overset{\uparrow}{\mathbf{K}})
\end{aligned}
$$

As in the main text, the inference rules stated in Figure A.1 are defined so that for any operation op, op_n is $(\ldots op \overset{\uparrow}{\mathbf{E}_1} \underset{e_1,f_1}{\downarrow} \ldots) \overset{\uparrow}{\mathbf{E}_n} \underset{e_n,f_n}{\downarrow}$, where $\mathbf{E}_1, \ldots, \mathbf{E}_n$ are any of $\mathbf{D}$, $\mathbf{K}$, or $\mathbf{K}^{\uparrow}$, and $e_1, f_1, \ldots, e_n, f_n$ are any effects in $\mathcal{T}^*$. (Thus op_0 is just op.)

A.3 Lexicon

We repeat the dynamization procedure from chapter 3, in order to appropriately abbreviate the meanings assigned to certain lexical entries; in particular, nouns, adjectives, and some verbs. First, recall the definition of `lift`.

Definition 9 (`lift`)**.**

$$
\begin{aligned}
&\texttt{lift} : (s \to \alpha) \to (i \to e) \to s \to (i \to e) \to \alpha \\
&\texttt{lift} := (\lambda x, g, w, g'.xw)
\end{aligned}
$$

Then, following de Groote and Kanazawa (2013), we define a function $\overline{(\cdot)}$.

$$\begin{aligned}\overline{e} &= E\\ \overline{v} &= V\\ \overline{t} &= T\\ \overline{\alpha \to \beta} &= \overline{\alpha} \to \overline{\beta}\end{aligned}$$

The dynamization type shift is presented in terms of two functions dyn and $\mathtt{sta}$, each indexed by a type. At any given type a, these functions have the following type signatures (where A is defined as $(i \to e) \to s \to (i \to e) \to a$).

$$\begin{aligned}\mathsf{dyn}_a &: A \to \overline{a}\\ \mathtt{sta}_a &: \overline{a} \to A\end{aligned}$$

At all atomic types a, except for t, we then have that $\mathsf{dyn}_a x = \mathtt{sta}_a x = x$, as well as that $\mathtt{sta}_t \phi = \phi$. dyn_t, we define as

$$\mathsf{dyn}_t \phi = (\lambda g, w, g'.g = g' \wedge \phi g w g')$$

At complex types, we define the functions using mutual recursion as follows.

$$\begin{aligned}\mathsf{dyn}_{\alpha\to\beta} M &= (\lambda x^{\overline{\alpha}}.\mathsf{dyn}_\beta(\lambda g, w, g'.M g w g'(\mathtt{sta}_\alpha x g w g')))\\ \mathtt{sta}_{\alpha\to\beta} M &= (\lambda g, w, g', x^\alpha.\mathtt{sta}_\beta(M(\mathsf{dyn}_\alpha(\lambda h, w', h'.x)))g w g')\end{aligned}$$

Finally, note that many of the meanings for lexical items are presented using abbrevations, all of which are defined in Table A.1. Recall the definition of $(\subseteq)$ from the main text, i.e.,

$$\begin{aligned}(\subseteq_t) &: t \to t \to t\\ \phi \subseteq_t \psi &:= \phi \to \psi\\ (\subseteq_{\alpha\to\beta}) &: (\alpha \to \beta) \to (\alpha \to \beta) \to t\\ \phi \subseteq_{\alpha\to\beta} \psi &:= (\forall x^\alpha.\phi x \subseteq_\beta \psi x)\end{aligned}$$

Thus $p \subseteq_T q$ abbreviates $(\forall g, w, g'.p g w g' \to q g w g')$.

λ-term	Shorthand	Type
$(\lambda g, w, g''.(\exists g'.\phi g w g' \wedge \psi g' w g''))$	$(+)$	$T \to T \to T$
$(\lambda \phi, \psi, g.\{\langle w, g\rangle \mid (\forall g'.\phi g w g' \to (\exists g''.\psi g' w g''))\})$	$(\rightrightarrows)$	$T \to T \to T$
$(\lambda \phi, \psi.\phi \gg\!=_{\mathbf{D}e,f} (\lambda p, i, c.\psi i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}f,e} (\lambda q.(p+q)^{\uparrow_{\mathbf{P}}})))$	$(;)$	$\mathbf{D}eT \to \mathbf{D}fT \to \mathbf{D}(e+f)T$
$(\lambda P, k, i.((\lambda g.\{\langle w, g''\rangle \mid (\exists g'.g[i]g', P(\lambda g, w, g'.g_i)g'wg'')\})^{\uparrow_{\mathbf{D}}}; k(\lambda g, w, g'.g_i))i')$	a	$(E \to T) \to \mathbf{K}\epsilon E$
$(\lambda m.(\lambda i, c.(\lambda p, g.\{\langle w, g\rangle \mid \neg(\exists g'.pgwg')\})^{\uparrow_{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,e}} m^{\Downarrow} i c)^{\Uparrow})$	not	$\mathbf{K}eT \to \mathbf{K}eT$
$(\lambda g.\{\langle w, g\rangle \mid \top\})$	true	T
$(\lambda P.(\lambda i, c.(c((\lambda g.\{\langle w, g\rangle \mid \mathbf{have} w(ygwg)(xgwg)\}) + Py) = \mathtt{true} \Vdash_{x,y} y)))$	her	$(E \to T) \to \mathbf{D}\{E, E\}E$
$(\lambda R, x, i, c.(c(Rxe) \Vdash_e (\lambda g.\{\langle w, g\rangle \mid (\exists e'.\mathbf{stop}(egwg)(xgwg)e')\})))$	stopped	$(E \to V \to T) \to E \to \mathbf{D}\{V\}T$
$(\lambda f, i, c.(\top \Vdash_x fx))$	push	$(\alpha \to \beta) \to \mathbf{D}\{\alpha\}\beta$
$(\lambda P, x, g.\{\langle w, g\rangle \mid (\exists e.\mathbf{begin} w(\mathtt{sta}_{v\to t} Pgwg)(xgwg)e)\})$	began	$(V \to T) \to E \to T$
$(\lambda i, g, p, g', w, g''.(\exists g'''.pg'wg''' \wedge (\forall j.(j \le i \to g''j = gj) \wedge (i' \le j \to g''j = g'''(j-i)))))$	slide	$i \to (i \to e) \to T \to T$
$(\lambda q, p, g.\{\langle w, g\rangle \mid qw \subseteq_T qw + p\})$	shift	$(s \to T) \to T \to T$
$(\lambda \phi, f, i, c.(fi)^{\uparrow_{\mathbf{P}}\downarrow^{\mathbf{P}}_{\epsilon,e}}(\phi(i+i)(c \circ fi)))$	modalize	$\mathbf{D}eT \to (i \to T \to T) \to \mathbf{D}eT$
$(\lambda \phi, x.\mathtt{modalize}\phi(\lambda i, p, g, w, g'.\mathtt{shift}(\lambda w'.\mathtt{slide}ig(\mathbf{dox}(xgwg')w'))pgwg'))$	believe	$\mathbf{D}eT \to E \to \mathbf{D}eT$
$(\lambda P, x, q.\mathtt{modalize}(P^{\downarrow^{\mathbf{K}}_{e,f}} x)^{\Downarrow}(\lambda i, g.\mathtt{shift}(\mathtt{slide}ig \circ q)g))$	would	$\mathbf{K}e(E \to T) \to \mathbf{K}fE \to (s \to T) \to \mathbf{D}(e+f)T$

Table A.1: Abbreviations

A.3.1 Determiners

$$\langle a, \mathsf{a} \rangle :: \mathsf{d/n}$$
$$\langle a, \mathtt{id} \rangle :: \mathsf{d/n}$$
$$\langle every, (\lambda P, k.\mathtt{not}(\mathsf{a}P \gg\!=_{\mathbf{K}\epsilon,f} (\lambda x.\mathtt{not}(kx)))) \rangle :: \mathsf{d/n}$$
$$\langle her, \mathtt{her} \rangle :: \mathsf{d/n}$$
$$\langle the, (\lambda P, i, c.(c(Px) = \mathtt{true} \Vdash_x x)) \rangle :: \mathsf{d/n}$$

A.3.2 Nouns

$$\langle brother, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{brother}) \rangle :: \mathsf{n}$$
$$\langle diver, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{diver}) \rangle :: \mathsf{n}$$
$$\langle dog, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{dog}) \rangle :: \mathsf{n}$$
$$\langle girl, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{girl}) \rangle :: \mathsf{n}$$
$$\langle linguist, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{ling}) \rangle :: \mathsf{n}$$
$$\langle man, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{man}) \rangle :: \mathsf{n}$$
$$\langle triple\ cheeseburger, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{tc}) \rangle :: \mathsf{n}$$
$$\langle wetsuit, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{wetsuit}) \rangle :: \mathsf{n}$$

A.3.3 Adjectives

$$\langle drenched, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{drenched}) \rangle :: \mathsf{a}$$
$$\langle healthy, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{healthy}) \rangle :: \mathsf{a}$$
$$\langle nice, \mathsf{dyn}_{e\to t}(\mathtt{lift}\mathbf{nice}) \rangle :: \mathsf{a}$$

A.3.4 Verbs

$$\langle \mathit{ate}, \mathsf{dyn}_{e\to e\to t}\,\mathtt{lift}(\lambda w,x,y.(\exists e.\mathbf{eat}wxye))\rangle : (\mathsf{d}\backslash\mathsf{s})/\mathsf{d}$$
$$\langle \mathit{began}, \mathsf{began}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{vPROG}$$
$$\langle \mathit{began}, \mathsf{pushbegan}\rangle :: \mathsf{d}\backslash\mathsf{s}$$
$$\langle \mathit{believe}, \mathsf{believe}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{s}$$
$$\langle \mathit{brought}, \mathsf{dyn}_{e\to e\to t}\,\mathtt{lift}(\lambda w,x,y.(\exists e.\mathbf{bring}wxye))\rangle : (\mathsf{d}\backslash\mathsf{s})/\mathsf{d}$$
$$\langle \mathit{exists}, \mathsf{dyn}_{e\to t}\,\mathtt{lift}(\lambda w,x.\mathbf{exists})\rangle :: \mathsf{d}\backslash\mathsf{s}$$
$$\langle \mathit{gardening}, \mathsf{dyn}_{e\to t}\,\mathtt{lift}(\lambda w,x.(\exists e.\mathbf{garden}wxe))\rangle :: \mathsf{vPROG}$$
$$\langle \mathit{saw}, \mathsf{dyn}_{e\to e\to t}\,\mathtt{lift}(\lambda w,x,y.(\exists e.\mathbf{see}wxye))\rangle : (\mathsf{d}\backslash\mathsf{s})/\mathsf{d}$$
$$\langle \mathit{sat\ down}, \mathsf{dyn}_{e\to t}\,\mathtt{lift}(\lambda w,x.(\exists e.\mathbf{sit}wxe))\rangle :: \mathsf{d}\backslash\mathsf{s}$$
$$\langle \mathit{stopped}, \mathsf{stopped}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{vPROG}$$
$$\langle \mathit{walked\ in}, \mathsf{dyn}_{e\to t}\,\mathtt{lift}(\lambda w,x.(\exists e.\mathbf{walk}wxe))\rangle :: \mathsf{d}\backslash\mathsf{s}$$

A.3.5 Copulas, auxiliaries, and the infinitival morpheme

$$\langle \mathit{is}, \mathsf{id}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{a}$$
$$\langle \mathit{to}, \mathsf{id}\rangle :: \mathsf{vINF}/\mathsf{v}$$
$$\langle \mathit{did}, \mathsf{id}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{v}$$
$$\langle \mathit{were}, (\lambda P,x,w.\mathbf{mod}w + Px)\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{vPROG}$$
$$\langle \mathit{would}, \mathsf{would}\rangle :: (\mathsf{d}\backslash\mathsf{s})/\mathsf{v}$$

A.3.6 Coordinators and sentential connectives

$$\langle ;, (;)\rangle :: \mathsf{s}\backslash(\mathsf{s}/\mathsf{s})$$
$$\langle \mathit{and}, (\lambda q,p.p^{\uparrow_{\mathbf{D}}};q)\rangle :: (\mathsf{s}\backslash\mathsf{s})/\mathsf{s}$$
$$\langle \mathit{and}, (\lambda Q,P,x.(Px)^{\uparrow_{\mathbf{D}}};(x^{\uparrow_{\mathbf{D}}} \blacktriangleleft^{*} Q))\rangle :: ((\mathsf{d}\backslash\mathsf{s})\backslash(\mathsf{d}\backslash\mathsf{s}))/(\mathsf{d}\backslash\mathsf{s})$$
$$\langle \mathit{if}, (\lambda R,\phi.\phi R)\rangle :: (\mathsf{s}/\mathsf{s})/\mathsf{s}$$
$$\langle \mathit{if}, (\lambda p,\psi,i,c.\psi i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}\epsilon,e} (\lambda q.(p \rightrightarrows q)^{\uparrow_{\mathbf{P}}}))\rangle :: (\mathsf{s}/\mathsf{s})/\mathsf{s}$$

A.3.7 Names and pronouns

$$\langle \mathit{Ashley}, (\lambda g, w, g'.\mathbf{a}) \rangle :: \mathtt{d}$$
$$\langle \mathit{Emily}, (\lambda g, w, g'.\mathbf{e}) \rangle :: \mathtt{d}$$
$$\langle \mathit{John}, (\lambda g, w, g'.\mathbf{j}) \rangle :: \mathtt{d}$$
$$\langle \mathit{Mary}, (\lambda g, w, g'.\mathbf{m}) \rangle :: \mathtt{d}$$
$$\langle \mathit{she}, \mathtt{push}\,\mathtt{id}_E \rangle :: \mathtt{d}$$
$$\langle \mathit{smoking}, \mathtt{lift}\,\mathbf{smkg} \rangle :: \mathtt{d}$$

A.3.8 Miscellaneous type shifters

$$\langle \epsilon, \mathtt{push} \rangle :: \mathtt{s/s}$$
$$\langle \epsilon, \mathtt{push} \rangle :: \mathtt{vINF/(vINF/v)}$$
$$\langle \epsilon, \mathtt{push} \rangle :: \mathtt{(d\backslash s)/((d\backslash s)/v)}$$

Appendix B

Proofs

Here we prove the theorems stated in the main text (see subsection B.0.2), as well as Fact 1 of chapter 2 and Fact 2 of chapter 3 (see subsection B.0.3).

B.0.1 Equivalence between sets of laws

To simplify the proofs of the theorems, we present graded monads in terms of $(\cdot)^{\uparrow\mathbf{E}}$ and $\gg\!=_{\mathbf{E}\,e,f}$, rather than $(\cdot)^{\uparrow\mathbf{E}}$, $(\cdot)^{\mathbf{E}}_{\downarrow e,f}$, and $\mu_{\mathbf{E}\,e,f}$ (as in the main text). These presentations are equivalent, the first allowing that we prove only a simpler, alternative variant of the Graded Monad Laws (that of Figure B.1). Thus, in order to prove the theorems of the main text, we need to show this equivalence.

Theorem 4. The Graded Monad Laws of Figure B.1 are equivalent to the Graded Monad Laws of Figure 2.7 in conjunction with the Graded Applicative Functor Laws of Figure 2.6

Proof. We first show that the alternative presentation is strong enough for the presentation in the main text. To prove those laws involving $(\cdot)^{\mathbf{E}}_{\downarrow e,f}$, we identify $m^{\mathbf{E}}_{\downarrow e,f}$ with $(\lambda n.m \gg\!=_{\mathbf{E}\,e,f} (\lambda f.n \gg\!=_{\mathbf{E}\,f,\epsilon} (\lambda x.(f\,x)^{\uparrow\mathbf{E}})))$. To prove those laws involving $\mu_{\mathbf{E}\,e,f}$, we identify $\mu_{\mathbf{E}\,e,f}\,m$ with $m \gg\!=_{\mathbf{E}\,e,f} \mathrm{id}$. Recall that $m \gg\!=_{\mathbf{E}\,e,f} k$ is defined as $\mu_{\mathbf{E}\,e,f}(k^{\uparrow\mathbf{E}}{}^{\mathbf{E}}_{\downarrow \epsilon,e} m)$.

Identity

$$
\begin{aligned}
& \mathtt{id}^{\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f} \\
&= (\lambda n.\mathtt{id}^{\overset{\mathbf{E}}{\uparrow}} \gg\!=_{\mathbf{E}e,f}(\lambda f.n \gg\!=_{\mathbf{E}f,\epsilon}(\lambda x.(f\,x)^{\overset{\mathbf{E}}{\uparrow}}))) && \text{(def.)} \\
&= (\lambda n.n \gg\!=_{\mathbf{E}f,\epsilon}(\lambda x.x^{\overset{\mathbf{E}}{\uparrow}})) && (\equiv_\beta\text{, Left Identity 2)} \\
&= \mathtt{id} && \text{(Right Identity 2, } \equiv_\eta)
\end{aligned}
$$

Composition

$$
\begin{aligned}
& u^{\overset{\mathbf{E}}{\downarrow}}_{e,f+g} \circ v^{\overset{\mathbf{E}}{\downarrow}}_{f,g} \\
&= (\lambda n.u \gg\!=_{\mathbf{E}e,f+g}(\lambda u'.n \gg\!=_{\mathbf{E}f+g,\epsilon}(\lambda x.(u'x)^{\overset{\mathbf{E}}{\uparrow}}))) && \text{(def.)} \\
&\qquad \circ (\lambda n'.v \gg\!=_{\mathbf{E}f,g}(\lambda v'.n' \gg\!=_{\mathbf{E}g,\epsilon}(\lambda x.(v'x)^{\overset{\mathbf{E}}{\uparrow}}))) \\
&= (\lambda n'.u \gg\!=_{\mathbf{E}e,f+g}(\lambda u'.(v \gg\!=_{\mathbf{E}f,g}(\lambda v'.n' \gg\!=_{\mathbf{E}g,\epsilon}(\lambda x.(v'x)^{\overset{\mathbf{E}}{\uparrow}}))) && (\equiv_\beta) \\
&\qquad \gg\!=_{\mathbf{E}f+g,\epsilon}(\lambda x.(u'x)^{\overset{\mathbf{E}}{\uparrow}}))) \\
&= (\lambda n'.u \gg\!=_{\mathbf{E}e,f+g}(\lambda u'.v && \text{(Associativity 2, Left Identity 2)} \\
&\qquad \gg\!=_{\mathbf{E}f,g}(\lambda v'.n' \gg\!=_{\mathbf{E}g,\epsilon}(\lambda x.(u'(v'x))^{\overset{\mathbf{E}}{\uparrow}})))) \\
&= (\lambda n'.(\circ)^{\overset{\mathbf{E}}{\uparrow}} \gg\!=_{\mathbf{E}\epsilon,e+f+g}(\lambda c.u \gg\!=_{\mathbf{E}e,f+g}(\lambda u'.v && (\equiv_\beta\text{, Left Identity 2)} \\
&\qquad \gg\!=_{\mathbf{E}f,g}(\lambda v'.n' \gg\!=_{\mathbf{E}g,\epsilon}(\lambda x.(cu'v'x)^{\overset{\mathbf{E}}{\uparrow}}))))) \\
&= (\lambda n'.(((\circ)^{\overset{\mathbf{E}}{\uparrow}} && \text{(Left Identity 2, Associativity 2)} \\
&\qquad \gg\!=_{\mathbf{E}\epsilon,e}(\lambda c.u \gg\!=_{\mathbf{E}e,\epsilon}(\lambda u'.(cu')^{\overset{\mathbf{E}}{\uparrow}}))) \\
&\qquad\qquad \gg\!=_{\mathbf{E}e,f}(\lambda u''.v \gg\!=_{\mathbf{E}f,\epsilon}(\lambda v'.(u''v)^{\overset{\mathbf{E}}{\uparrow}}))) \\
&\qquad\qquad\qquad \gg\!=_{\mathbf{E}e+f,g}(\lambda v''.n' \gg\!=_{\mathbf{E}f,\epsilon}(\lambda x.(v''x)^{\overset{\mathbf{E}}{\uparrow}}))) \\
&= (((\circ)^{\overset{\mathbf{E}}{\uparrow}}{}^{\overset{\mathbf{E}}{\downarrow}}_{\epsilon,e}u)^{\overset{\mathbf{E}}{\downarrow}}_{e,f}v)^{\overset{\mathbf{E}}{\downarrow}}_{e+f,g} && \text{(def., } \equiv_\eta)
\end{aligned}
$$

Homomorphism

$$
\begin{aligned}
& (f\,a)^{\uparrow^{\mathbf{E}}} \\
=\ & f^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,\epsilon} (\lambda g.\, a^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,\epsilon} (\lambda x.\, (g\,x)^{\uparrow^{\mathbf{E}}})) && \text{(Left Identity 2)} \\
=\ & f^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}}} a^{\uparrow^{\mathbf{E}}} && \text{(def.)}
\end{aligned}
$$

Interchange

$$
\begin{aligned}
& u^{\mathbf{E}\downarrow_{e,\epsilon}} a^{\uparrow^{\mathbf{E}}} \\
=\ & u \gg\!=_{\mathbf{E}e,\epsilon} (\lambda u'.\, a^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,\epsilon} (\lambda x.\, (u'\,x)^{\uparrow^{\mathbf{E}}})) && \text{(def.)} \\
=\ & u \gg\!=_{\mathbf{E}e,\epsilon} (\lambda u'.\, (u'\,a)^{\uparrow^{\mathbf{E}}}) && \text{(Left Identity 2)} \\
=\ & u \gg\!=_{\mathbf{E}e,\epsilon} (\lambda u'.\, ((\lambda f.\, f\,a)\,u')^{\uparrow^{\mathbf{E}}}) && (\equiv_\beta) \\
=\ & (\lambda f.\, f\,a)^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e} (\lambda g.\, u \gg\!=_{\mathbf{E}e,\epsilon} (\lambda u'.\, (g\,u')^{\uparrow^{\mathbf{E}}})) && \text{(Left Identity 2)} \\
=\ & (\lambda f.\, f\,a)^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} u && \text{(def.)}
\end{aligned}
$$

Left Identity

$$
\begin{aligned}
& \mu_{\mathbf{E}\epsilon,f}\, m^{\uparrow^{\mathbf{E}}} \\
=\ & m^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,f} \mathtt{id} && \text{(def.)} \\
=\ & m && \text{(Left Identity 2)}
\end{aligned}
$$

Right Identity

$$
\begin{aligned}
& \mu_{\mathbf{E}e,\epsilon} ((\lambda x.\, x^{\uparrow^{\mathbf{E}}})^{\uparrow^{\mathbf{E}}} {}^{\mathbf{E}\downarrow_{\epsilon,e}} m) \\
=\ & ((\lambda x.\, x^{\uparrow^{\mathbf{E}}})^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e} (\lambda f.\, m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda x.\, (f\,x)^{\uparrow^{\mathbf{E}}}))) \gg\!=_{\mathbf{E}e,\epsilon} \mathtt{id} && \text{(def.)} \\
=\ & (\lambda x.\, x^{\uparrow^{\mathbf{E}}})^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e} (\lambda f.\, m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda x.\, (f\,x)^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}e,\epsilon} \mathtt{id})) && \text{(Associativity 2)} \\
=\ & m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda x.\, x^{\uparrow^{\mathbf{E}}}) && \text{(Left Identity 2, } \equiv_\beta\text{)} \\
=\ & m && (\equiv_\eta\text{, Right Identity 2})
\end{aligned}
$$

Associativity

$$
\begin{aligned}
& \mu_{\mathbf{E}e+f,g}(\mu_{\mathbf{E}e,f} m) \\
= \; & (m \gg\!=_{\mathbf{E}e,f} \mathrm{id}) \gg\!=_{\mathbf{E}e+f,g} \mathrm{id} && \text{(def.)} \\
= \; & m \gg\!=_{\mathbf{E}e,f+g} (\lambda x.x \gg\!=_{\mathbf{E}f,g} \mathrm{id}) && \text{(Associativity 2)} \\
= \; & m \gg\!=_{\mathbf{E}e,f+g} \mu_{\mathbf{E}f,g} && \text{(def., } \equiv_\eta\text{)} \\
= \; & \mu_{\mathbf{E}f,g}{}^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e+f+g} (\lambda g.m \gg\!=_{\mathbf{E}e,f+g} (\lambda x.gx)) && \text{(}\equiv_\eta\text{, Left Identity 2)} \\
= \; & \mu_{\mathbf{E}f,g}{}^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e+f+g} (\lambda g.m \gg\!=_{\mathbf{E}e,f+g} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,f+g} \mathrm{id})) && \text{(Left Identity 2)} \\
= \; & \mu_{\mathbf{E}f,g}{}^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e+f+g} (\lambda g.(m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}})) \gg\!=_{\mathbf{E}e,f+g} \mathrm{id}) && \text{(Associativity 2)} \\
= \; & (\mu_{\mathbf{E}f,g}{}^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e} (\lambda g.m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}}))) \gg\!=_{\mathbf{E}e,f+g} \mathrm{id} && \text{(}\equiv_\eta\text{, Associativity 2)} \\
= \; & \mu_{\mathbf{E}e,f+g} (\mu_{\mathbf{E}f,g}{}^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,e} m) && \text{(def.)}
\end{aligned}
$$

Naturality

$$
\begin{aligned}
& f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,e+f} (\mu_{\mathbf{E}e,f} m) \\
= \; & f^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e+f} (\lambda g.(m \gg\!=_{\mathbf{E}e,f} \mathrm{id}) \gg\!=_{\mathbf{E}e+f,\epsilon} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}})) && \text{(def.)} \\
= \; & f^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e+f} (\lambda g.m \gg\!=_{\mathbf{E}e,f} (\lambda y.y \gg\!=_{\mathbf{E}f,\epsilon} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}}))) && \text{(Associativity 2)} \\
= \; & m \gg\!=_{\mathbf{E}e,f} (\lambda y.y \gg\!=_{\mathbf{E}f,\epsilon} (\lambda x.(fx)^{\overset{\uparrow}{\mathbf{E}}})) && \text{(Left Identity 2)} \\
= \; & m \gg\!=_{\mathbf{E}e,f} (\lambda y.f^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,f} (\lambda g.y \gg\!=_{\mathbf{E}f,\epsilon} (\lambda x.(gx)^{\overset{\uparrow}{\mathbf{E}}}))) && \text{(Left Identity 2)} \\
= \; & m \gg\!=_{\mathbf{E}e,f} (\lambda y.f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f} y) && \text{(def.)} \\
= \; & m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda y.(f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f} y)^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}e,f} \mathrm{id}) && \text{(Left Identity 2)} \\
= \; & (m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda y.(f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f} y)^{\overset{\uparrow}{\mathbf{E}}})) \gg\!=_{\mathbf{E}e,f} \mathrm{id} && \text{(Associativity 2)} \\
= \; & (f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f}{}^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,e} (\lambda g.m \gg\!=_{\mathbf{E}e,\epsilon} (\lambda y.(gy)^{\overset{\uparrow}{\mathbf{E}}}))) \gg\!=_{\mathbf{E}e,f} \mathrm{id} && \text{(Left Identity 2)} \\
= \; & \mu_{\mathbf{E}e,f} (f^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,f}{}^{\overset{\uparrow}{\mathbf{E}}\overset{\mathbf{E}}{\downarrow}}_{\epsilon,e} m) && \text{(def.)}
\end{aligned}
$$

We now show that the presentation in the main text is strong enough for the alternative presentation.

Left Identity 2

$$
\begin{aligned}
& a^{\uparrow^{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,f} k \\
=\ & \mu_{\mathbf{E}\epsilon,f}(k^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}}} a^{\uparrow^{\mathbf{E}}}) && \text{(def.)} \\
=\ & \mu_{\mathbf{E}\epsilon,f}(ka)^{\uparrow^{\mathbf{E}}} && \text{(Homomorphism)} \\
=\ & ka && \text{(Left Identity)}
\end{aligned}
$$

Right Identity 2

$$
\begin{aligned}
& m \gg\!=_{\mathbf{E}e,\epsilon} (\cdot)^{\uparrow^{\mathbf{E}}} \\
=\ & \mu_{\mathbf{E}e,\epsilon}((\cdot)^{\uparrow^{\mathbf{E}}\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} m) && \text{(def.)} \\
=\ & m && (\equiv_\eta, \text{Right Identity})
\end{aligned}
$$

Associativity 2

$$
\begin{aligned}
& (m \gg\!=_{\mathbf{E}e,f} n) \gg\!=_{\mathbf{E}e+f,g} o \\
=\ & \mu_{\mathbf{E}e+f,g}(o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e+f}}}(\mu_{\mathbf{E}e,f}(n^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} m))) && \text{(def.)} \\
=\ & \mu_{\mathbf{E}e+f,g}(\mu_{\mathbf{E}e,f}(o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,f}}\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}}(n^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} m))) && \text{(Naturality)} \\
=\ & \mu_{\mathbf{E}e,f+g}(\mu_{\mathbf{E}f,g}{}^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}}(o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,f}}}{}^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}}(n^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} m))) && \text{(Associativity)} \\
=\ & \mu_{\mathbf{E}e,f+g}((\mu_{\mathbf{E}f,g}{}^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} \circ o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,f}}}{}^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} \circ n^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}})m) && (\equiv_\beta) \\
=\ & \mu_{\mathbf{E}e,f+g}((((\circ)^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}}}\mu_{\mathbf{E}f,g}{}^{\uparrow^{\mathbf{E}}}){}^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}}(((\circ)^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}}} o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,f}}}{}^{\uparrow^{\mathbf{E}}}){}^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,\epsilon}} n^{\uparrow^{\mathbf{E}}}))^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}} m) && \text{(Composition)} \\
=\ & \mu_{\mathbf{E}e,f+g}((\lambda v.\mu_{\mathbf{E}f,g}(o^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,f}}}(nv)))^{\uparrow^{\mathbf{E}}_{\mathbf{E}\downarrow_{\epsilon,e}}} m) && \text{(Homomorphism, } \equiv_\beta) \\
=\ & m \gg\!=_{\mathbf{E}e,f+g} (\lambda v.nv \gg\!=_{\mathbf{E}f,g} o) && \text{(def.)}
\end{aligned}
$$

□

We now prove the theorems of the main text.

$$a^{\overset{\uparrow}{\mathbf{E}}} \gg\!=_{\mathbf{E}\epsilon,f} k = ka \qquad \text{(Left Identity 2)}$$

$$m \gg\!=_{\mathbf{E}e,\epsilon} (\cdot)^{\overset{\uparrow}{\mathbf{E}}} = m \qquad \text{(Right Identity 2)}$$

$$(m \gg\!=_{\mathbf{E}e,f} n) \gg\!=_{\mathbf{E}e+f,g} o = m \gg\!=_{\mathbf{E}e,f+g} (\lambda v.nv \gg\!=_{\mathbf{E}f,g} o) \qquad \text{(Associativity 2)}$$

Figure B.1: The Graded Monad Laws (alternative presentation)

B.0.2 Theorems from the main text

Definition 5 (P).

$$\mathbf{P} : \mathcal{T}^* \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{P}\epsilon\alpha = (\alpha \to o) \to (\# \to o) \to o$$
$$\mathbf{P}(ae)\alpha = a \to \mathbf{P}e\alpha$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{P}}} : \alpha \to \mathbf{P}\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{P}}} = (\lambda s, f.sa)$$

$$(\cdot)^{\overset{\mathbf{P}}{\downarrow}}_{e,f} : \mathbf{P}e(\alpha \to \beta) \to \mathbf{P}f\alpha \to \mathbf{P}(e+f)\beta$$
$$m^{\overset{\mathbf{P}}{\downarrow}}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}} = (\lambda n, x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ mx_1, \ldots, x_m(\lambda u.ny_1, \ldots, y_n(\lambda v.s(uv))f)f)$$

$$\mu_{\mathbf{P}e,f} : \mathbf{P}e(\mathbf{P}f\alpha) \to \mathbf{P}(e+f)\alpha$$
$$\mu_{\mathbf{P}\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}} m = (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f. \\ mx_1, \ldots, x_m(\lambda n.ny_1, \ldots, y_n sf)f)$$

Theorem 1. **P** is a graded monad.

Proof.

Left Identity 2

$$
\begin{aligned}
& a^{\overset{\uparrow}{\mathbf{P}}} \gg\!=_{\mathbf{P}\epsilon,f} k \\
=\ & \mu_{\mathbf{P}\epsilon,f}(k^{\overset{\uparrow}{\mathbf{P}}\overset{\mathbf{P}}{\downarrow}}_{\epsilon,\epsilon}(\lambda s, f.sa)) && \text{(def.)} \\
=\ & \mu_{\mathbf{P}\epsilon,f}((\lambda n, s'', f''.(\lambda s', f'.s'k)(\lambda u.n(\lambda v.s''(uv))f)f)(\lambda s, f.sa)) && \text{(def.)} \\
=\ & \mu_{\mathbf{P}\epsilon,f}(\lambda s'', f''.s''(ka)) && (\equiv_\beta) \\
=\ & (\lambda y_1, \ldots, y_n, s, f.(\lambda s'', f''.s''(ka))(\lambda n.ny_1, \ldots, y_n sf)f) && \text{(def.)} \\
=\ & ka && (\equiv_{\beta,\eta})
\end{aligned}
$$

Right Identity 2

$$
\begin{aligned}
& m \gg\!=_{\mathbf{P}e,\epsilon} (\cdot)^{\overset{\uparrow}{\mathbf{P}}} \\
=\ & \mu_{\mathbf{P}e,\epsilon}((\cdot)^{\overset{\uparrow}{\mathbf{P}}\overset{\uparrow}{\mathbf{P}}\overset{\mathbf{P}}{\downarrow}}_{\epsilon,e} m) && \text{(def.)} \\
=\ & \mu_{\mathbf{P}e,\epsilon}((\lambda n, y_1, \ldots, y_n, s, f.(\lambda s', f'.s'(\cdot)^{\overset{\uparrow}{\mathbf{P}}}) && \text{(def.)} \\
& \qquad (\lambda u.ny_1, \ldots, y_n(\lambda v.s(uv))f)f)m) \\
=\ & \mu_{\mathbf{P}e,\epsilon}(\lambda y_1, \ldots, y_n, s, f.my_1, \ldots, y_n(\lambda v.sv^{\overset{\uparrow}{\mathbf{P}}})f) && (\equiv_\beta) \\
=\ & (\lambda x_1, \ldots, x_n, s, f.(\lambda y_1, \ldots, y_n, s', f'.my_1, \ldots, y_n(\lambda v.s'v^{\overset{\uparrow}{\mathbf{P}}})f') && \text{(def.)} \\
& \qquad x_1, \ldots, x_n(\lambda n.nsf)f) \\
=\ & (\lambda x_1, \ldots, x_n, s, f.mx_1, \ldots, x_n(\lambda v.v^{\overset{\uparrow}{\mathbf{P}}}sf)f) && (\equiv_\beta) \\
=\ & (\lambda x_1, \ldots, x_n, s, f.mx_1, \ldots, x_n(\lambda v.(\lambda s', f'.s'v)sf)f) && \text{(def.)} \\
=\ & m && (\equiv_{\beta,\eta})
\end{aligned}
$$

Associativity 2

$$
\begin{aligned}
& (m \gg\!=_{\mathbf{P}\,e,f} n) \gg\!=_{\mathbf{P}\,e+f,g} o \\
=\ & \mu_{\mathbf{P}\,e+f,g}(o^{\uparrow^{\mathbf{P}}_{\epsilon,e+f}}(\mu_{\mathbf{P}\,e,f}(n^{\uparrow^{\mathbf{P}}_{\epsilon,e}} m))) && \text{(def.)} \\
=\ & \mu_{\mathbf{P}\,e+f,g}(o^{\uparrow^{\mathbf{P}}_{\epsilon,e+f}}(\mu_{\mathbf{P}\,e,f}(\lambda x_1,\ldots,x_m,s,f.mx_1,\ldots,x_m(\lambda u.s(nu))f))) && \text{(def., } \equiv_\beta) \\
=\ & \mu_{\mathbf{P}\,e+f,g}(o^{\uparrow^{\mathbf{P}}_{\epsilon,e+f}}(\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,s,f. && \text{(def., } \equiv_\beta) \\
& \qquad mx_1,\ldots,x_m(\lambda u.nuy_1,\ldots,y_n sf)f)) \\
=\ & \mu_{\mathbf{P}\,e+f,g}(\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,s,f. && \text{(def., } \equiv_\beta) \\
& \qquad mx_1,\ldots,x_m(\lambda u.nuy_1,\ldots,y_n(\lambda v.s(ov))f)f) \\
=\ & (\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,z_1,\ldots,z_o,s,f. && \text{(def., } \equiv_\beta) \\
& \qquad mx_1,\ldots,x_m(\lambda u.nuy_1,\ldots,y_n(\lambda v.ovz_1,\ldots,z_o sf)f)f) \\
=\ & \mu_{\mathbf{P}\,e,f+g}((\lambda u,y_1,\ldots,y_n,z_1,\ldots,z_o,s,f. && \text{(def., } \equiv_\beta) \\
& \qquad nuy_1,\ldots,y_n(\lambda v.ovz_1,\ldots,z_o sf)f)^{\uparrow^{\mathbf{P}}_{\epsilon,e}} m) \\
=\ & \mu_{\mathbf{P}\,e,f+g}((\lambda u.\mu_{\mathbf{P}\,f,g}(o^{\uparrow^{\mathbf{P}}_{\epsilon,f}}(nu)))^{\mathbf{P}}_{\epsilon,e} m) && \text{(def., } \equiv_\beta) \\
=\ & m \gg\!=_{\mathbf{P}\,e,f+g}(\lambda u.nu \gg\!=_{\mathbf{P}\,f,g} o) && \text{(def.)}
\end{aligned}
$$

□

Definition 12 (RT).

$$\mathbf{RT} : \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{RT}(r)(\mathbf{E_0})e\alpha = r \to \mathbf{E_0}e\alpha$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} : \alpha \to \mathbf{RT}(r)(\mathbf{E_0})\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} = (\lambda s.a^{\overset{\uparrow}{\mathbf{E_0}}})$$

$$(\cdot)^{\mathbf{RT}(r)(\mathbf{E_0})}_{\downarrow e,f} : \mathbf{RT}(r)(\mathbf{E_0})e(\alpha \to \beta) \to \mathbf{RT}(r)(\mathbf{E_0})f\alpha \to \mathbf{RT}(r)(\mathbf{E_0})(e+f)\beta$$
$$u^{\mathbf{RT}(r)(\mathbf{E_0})}_{\downarrow e,f} = (\lambda v, s.(us)^{\mathbf{E_0}}_{\downarrow e,f}(vs))$$

$$\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}} : \mathbf{RT}(r)(\mathbf{E_0})e(\mathbf{RT}(r)(\mathbf{E_0})f\alpha) \to \mathbf{RT}(r)(\mathbf{E_0})(e+f)\alpha$$
$$\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}}m = (\lambda s.\mu_{\mathbf{E_0}e,f}((\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\mathbf{E_0}}_{\downarrow\epsilon,e}(ms)))$$

Theorem 2. If $\mathbf{E_0}$ is a graded monad with monoid of parameters $\mathcal{E}$, then so is $\mathbf{RT}(r)(\mathbf{E_0})$, for any type r.

Proof.

Left Identity 2

$$\begin{aligned}
& a^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} \gg\!=_{\mathbf{RT}(r)(\mathbf{E_0})_{\epsilon,f}} k \\
=\ & \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{\epsilon,f}}(k^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\mathbf{RT}(r)(\mathbf{E_0})}_{\downarrow\epsilon,\epsilon}(\lambda s.a^{\overset{\uparrow}{\mathbf{E_0}}})) && \text{(def.)} \\
=\ & \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{\epsilon,f}}(\lambda s.k^{\overset{\uparrow}{\mathbf{E_0}}\mathbf{E_0}}_{\downarrow\epsilon,\epsilon}a^{\overset{\uparrow}{\mathbf{E_0}}}) && \text{(def., } \equiv_\beta) \\
=\ & \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{\epsilon,f}}(\lambda s.(ka)^{\overset{\uparrow}{\mathbf{E_0}}}) && \text{(Homomorphism)} \\
=\ & (\lambda s.\mu_{\mathbf{E_0}\epsilon,f}((\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\mathbf{E_0}}_{\downarrow\epsilon,\epsilon}(ka)^{\overset{\uparrow}{\mathbf{E_0}}})) && \text{(def., } \equiv_\beta) \\
=\ & (\lambda s.\mu_{\mathbf{E_0}\epsilon,f}(kas)^{\overset{\uparrow}{\mathbf{E_0}}}) && \text{(Homomorphism, } \equiv_\beta) \\
=\ & ka && \text{(Left Identity, } \equiv_\eta)
\end{aligned}$$

Right Identity 2

$$
\begin{aligned}
& m \gg\!= _{\mathbf{RT}(r)(\mathbf{E_0})_{e,\epsilon}} (\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}} \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,\epsilon}} ((\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\mathbf{RT}(r)(\mathbf{E_0})}{\underset{\epsilon,e}{\downarrow}}} m) && \text{(def.)} \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,\epsilon}} (\lambda s.(\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (ms)) && \text{(def., } \equiv_\beta) \\
= \;& (\lambda s.\mu_{\mathbf{E_0}\,e,\epsilon} ((\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (ms))) && \text{(def.)} \\
= \;& (\lambda s.(\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} ((\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (ms))) && \text{(def., } \equiv_\beta) \\
= \;& (\lambda s.((\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} \circ (\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}})(ms)) && (\equiv_\beta) \\
= \;& (\lambda s.(((\circ)^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,\epsilon}{\downarrow}}} (\lambda n.ns)^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}})(ms)) && \text{(Composition)} \\
= \;& (\lambda s.((\lambda n.ns) \circ (\cdot)^{\overset{\uparrow}{\mathbf{RT}(r)(\mathbf{E_0})}})^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (ms)) && \text{(Homomorphism)} \\
= \;& (\lambda s.(\cdot)^{\overset{\uparrow}{\mathbf{E_0}}\,\overset{\uparrow}{\mathbf{E_0}}\,\overset{\mathbf{E_0}}{\underset{\epsilon,e}{\downarrow}}} (ms)) && \text{(def., } \equiv_\beta) \\
= \;& m && \text{(Right Identity, } \equiv_\eta)
\end{aligned}
$$

Associativity 2

$$
\begin{aligned}
& (m \gg\!=_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}} n) \gg\!=_{\mathbf{RT}(r)(\mathbf{E_0})_{e+f,g}} o \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e+f,g}}(o^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,e+f}}(\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f}}(n^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,e}} m))) && \text{(def.)} \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e+f,g}}(o^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,e+f}}(\lambda s.\mu_{\mathbf{E_0}_{e,f}}((\lambda u.us)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(n^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms))))) && \text{(def.)} \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e+f,g}}(o^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,e+f}} && \text{(Composition, Homomorphism, } \equiv_\beta) \\
& \qquad (\lambda s.\mu_{\mathbf{E_0}_{e,f}}((\lambda u.nus)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms)))) \\
= \;& (\lambda s.\mu_{\mathbf{E_0}_{e+f,g}}((\lambda v.ovs)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e+f}} && \text{(def., Composition, Homomorphism, } \equiv_\beta) \\
& \qquad (\mu_{\mathbf{E_0}_{e,f}}((\lambda u.nus)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms))))) \\
= \;& (\lambda s.\mu_{\mathbf{E_0}_{e+f,g}}(\mu_{\mathbf{E_0}_{e,f}}((\lambda v.ovs)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,f}}{}^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}((\lambda u.nus)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms))))) && \text{(Naturality)} \\
= \;& (\lambda s.\mu_{\mathbf{E_0}_{e,f+g}}(\mu_{\mathbf{E_0}_{f,g}}{}^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}} && \text{(Associativity)} \\
& \qquad ((\lambda v.ovs)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,f}}{}^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}((\lambda u.nus)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms))))) \\
= \;& (\lambda s.\mu_{\mathbf{E_0}_{e,f+g}}((\lambda u.\mu_{\mathbf{E_0}_{f,g}} && \text{(Composition, Homomorphism, } \equiv_\beta) \\
& \qquad ((\lambda v.ovs)^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,f}}(nus)))^{\uparrow^{\mathbf{E_0}}_{\mathbf{E_0}}\downarrow^{\mathbf{E_0}}_{\epsilon,e}}(ms))) \\
= \;& \mu_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f+g}}((\lambda u.\mu_{\mathbf{RT}(r)(\mathbf{E_0})_{f,g}}(o^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,f}}(nu)))^{\uparrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\mathbf{RT}(r)(\mathbf{E_0})}\downarrow^{\mathbf{RT}(r)(\mathbf{E_0})}_{\epsilon,e}} m) && \text{(def.)} \\
= \;& m \gg\!=_{\mathbf{RT}(r)(\mathbf{E_0})_{e,f+g}}(\lambda u.nu \gg\!=_{\mathbf{RT}(r)(\mathbf{E_0})_{f,g}} o) && \text{(def.)}
\end{aligned}
$$

□

Definition 17 (CT).

$$\mathbf{CT} : \mathcal{T} \to (\mathcal{E} \to \mathcal{T} \to \mathcal{T}) \to \mathcal{E} \to \mathcal{T} \to \mathcal{T}$$
$$\mathbf{CT}(r)(\mathbf{E_0})e\alpha = (\alpha \to \mathbf{E_0} f\, r) \to \mathbf{E_0}(e+f)r$$

$$(\cdot)^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} : \alpha \to \mathbf{CT}(r)(\mathbf{E_0})\epsilon\alpha$$
$$a^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} = (\lambda k.ka)$$

$$(\cdot)^{\overset{\mathbf{CT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} : \mathbf{CT}(r)(\mathbf{E_0})e(\alpha \to \beta) \to \mathbf{CT}(r)(\mathbf{E_0})f\,\alpha \to \mathbf{CT}(r)(\mathbf{E_0})(e+f)\beta$$
$$u^{\overset{\mathbf{CT}(r)(\mathbf{E_0})}{\downarrow}}_{e,f} = (\lambda v, k.u(\lambda U.v(\lambda V.k(U\,V))))$$

$$\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}} : \mathbf{CT}(r)(\mathbf{E_0})e(\mathbf{CT}(r)(\mathbf{E_0})f\,\alpha) \to \mathbf{CT}(r)(\mathbf{E_0})(e+f)\alpha$$
$$\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}} m = (\lambda k.m(\lambda k'.k'k))$$

Theorem 3. If $\mathbf{E_0}$ is a graded monad with monoid of parameters $\mathcal{E}$, then so is $\mathbf{CT}(r)(\mathbf{E_0})$, for any type r.

Proof.

Left Identity 2

$$\begin{aligned}
& a^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} \gg\!=_{\mathbf{CT}(r)(\mathbf{E_0})_{\epsilon,f}} k \\
=\ & \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{\epsilon,f}}(k^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}}{}^{\overset{\mathbf{CT}(r)(\mathbf{E_0})\uparrow}{\downarrow}}_{\epsilon,\epsilon} a^{\mathbf{CT}(r)(\mathbf{E_0})\uparrow}) && \text{(def.)} \\
=\ & \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{\epsilon,f}}(\lambda k'.k'(ka)) && \text{(def., } \equiv_\beta) \\
=\ & (\lambda k''.kak'') && \text{(def., } \equiv_\beta) \\
=\ & ka && (\equiv_\eta)
\end{aligned}$$

Right Identity 2

$$
\begin{aligned}
& m \gg\!= _{\mathbf{CT}(r)(\mathbf{E_0})_{e,\epsilon}} (\cdot)^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} \\
= \;& \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,\epsilon}} ((\cdot)^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}})^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} {}^{\mathbf{CT}(r)(\mathbf{E_0})}_{\epsilon, e} m) && \text{(def.)} \\
= \;& \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,\epsilon}} (\lambda k.m(\lambda x.kx^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}})) && \text{(def., } \equiv_\beta) \\
= \;& (\lambda k.m(\lambda x.kx)) && \text{(def., } \equiv_\beta) \\
= \;& m && (\equiv_\eta)
\end{aligned}
$$

Associativity 2

$$
\begin{aligned}
& (m \gg\!= _{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}} n) \gg\!= _{\mathbf{CT}(r)(\mathbf{E_0})_{e+f,g}} o \\
= \;& \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e+f,g}} (o^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} {}^{\mathbf{CT}(r)(\mathbf{E_0})}_{\epsilon, e+f} (\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f}} (n^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} {}^{\mathbf{CT}(r)(\mathbf{E_0})}_{\epsilon, e} m))) && \text{(def.)} \\
= \;& (\lambda k.m(\lambda x.nx(\lambda y.oyk))) && \text{(def., } \equiv_\beta) \\
= \;& \mu_{\mathbf{CT}(r)(\mathbf{E_0})_{e,f+g}} ((\lambda x.\mu_{\mathbf{CT}(r)(\mathbf{E_0})_{f,g}} (o^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} {}^{\mathbf{CT}(r)(\mathbf{E_0})}_{\epsilon, f} (nx)))^{\overset{\uparrow}{\mathbf{CT}(r)(\mathbf{E_0})}} {}^{\mathbf{CT}(r)(\mathbf{E_0})}_{\epsilon, e} m) && \text{(def., } \equiv_\beta) \\
= \;& m \gg\!= _{\mathbf{CT}(r)(\mathbf{E_0})_{e,f+g}} (\lambda x.nx \gg\!= _{\mathbf{CT}(r)(\mathbf{E_0})_{f,g}} o) && \text{(def.)}
\end{aligned}
$$

□

B.0.3 Facts from the main text

Fact 1.

$$
\begin{aligned}
& a^{\overset{\uparrow}{\mathbf{P}}} && \text{(Equiv. 1)}\\
=\ & (\top \Vdash a)
\end{aligned}
$$

$$
\begin{aligned}
& (\phi \Vdash_{x_1,\ldots,x_m} M)\overset{\mathbf{P}}{\downarrow}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}}(\psi \Vdash_{y_1,\ldots,y_n} N) && \text{(Equiv. 2)}\\
=\ & (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN)
\end{aligned}
$$

$$
\begin{aligned}
& \mu_{\mathbf{P}\{\alpha_1 \ldots \alpha_m\},\{\beta_1,\ldots,\beta_n\}}(\phi \Vdash_{x_1,\ldots,x_m} (\psi \Vdash_{y_1,\ldots,y_n} M)) && \text{(Equiv. 3)}\\
=\ & (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M)
\end{aligned}
$$

Proof.

Equiv. 1

$$
\begin{aligned}
& a^{\overset{\uparrow}{\mathbf{P}}}\\
=\ & (\lambda s, f.sa) && \text{(def.)}\\
=\ & (\lambda s, f.(\top \Vdash a)sf) && \text{(Succeed)}\\
=\ & (\top \Vdash a) && (\equiv_\eta)
\end{aligned}
$$

Equiv. 2

$$
\begin{aligned}
& (\phi \Vdash_{x_1,\ldots,x_m} M)\overset{\mathbf{P}}{\downarrow}_{\{\alpha_1 \ldots \alpha_m\},\{\beta_1 \ldots \beta_n\}}(\psi \Vdash_{y_1,\ldots,y_n} N)\\
=\ & (\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,s,f.(\phi \Vdash M)(\lambda u.(\psi \Vdash N)(\lambda v.s(uv))f)f) && \text{(def., } \equiv_\beta)
\end{aligned}
$$

There are four cases to consider, depending on the values of ϕ and ψ.

Case 1: $\phi = \psi = \top$

$$
\begin{aligned}
=\ & (\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,s,f.(\top \Vdash M)(\lambda u.(\top \Vdash N)(\lambda v.s(uv))f)f) && \text{(def.)}\\
=\ & (\lambda x_1,\ldots,x_m,y_1,\ldots,y_n,s,f.s(MN)) && \text{(Succeed, } \equiv_\beta)\\
=\ & (\top \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(Succeed, } \equiv_\eta)\\
=\ & (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(def.)}
\end{aligned}
$$

Case 2: $\phi = \psi = \bot$

$$\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.(\bot \Vdash M)(\lambda u.(\bot \Vdash N)(\lambda v.s(uv))f)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Fail, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(def.)}
\end{aligned}$$

Case 3: $\phi = \top$; $\psi = \bot$

$$\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.(\top \Vdash M)(\lambda u.(\bot \Vdash N)(\lambda v.s(uv))f)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Fail, Succeed, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(def.)}
\end{aligned}$$

Case 4: $\phi = \bot$; $\psi = \top$

$$\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.(\bot \Vdash M)(\lambda u.(\top \Vdash N)(\lambda v.s(uv))f)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Succeed, Fail, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} MN) && \text{(def.)}
\end{aligned}$$

Equiv. 3

$$\begin{aligned}
&\mu_{\mathbf{P}\{\alpha_1\ldots\alpha_m\},\{\beta_1,\ldots,\beta_n\}}(\phi \Vdash_{x_1,\ldots,x_m} (\psi \Vdash_{y_1,\ldots,y_n} M))\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, sf.(\phi \Vdash (\psi \Vdash M))(\lambda n.nsf)) && \text{(def., } \equiv_\beta)
\end{aligned}$$

There are four cases to consider, depending on the values of ϕ and ψ.

Case 1: $\phi = \psi = \top$

$$\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, sf.(\top \Vdash (\top \Vdash M))(\lambda n.nsf)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.sM) && \text{(Succeed, } \equiv_\beta)\\
&= (\top \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n,s,f} M) && \text{(Succeed, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M) && \text{(def.)}
\end{aligned}$$

Case 2: $\phi = \psi = \bot$

$$\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, sf.(\bot \Vdash (\bot \Vdash M))(\lambda n.nsf)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Fail, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n,s,f} M) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M) && \text{(def.)}
\end{aligned}$$

Case 3: $\phi = \top$; $\psi = \bot$

$$
\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, sf.(\top \Vdash (\bot \Vdash M))(\lambda n.nsf)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Fail, Succeed, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n,s,f} M) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M) && \text{(def.)}
\end{aligned}
$$

Case 4: $\phi = \bot$; $\psi = \top$

$$
\begin{aligned}
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, sf.(\bot \Vdash (\top \Vdash M))(\lambda n.nsf)f) && \text{(def.)}\\
&= (\lambda x_1, \ldots, x_m, y_1, \ldots, y_n, s, f.f\#) && \text{(Succeed, Fail, } \equiv_\beta)\\
&= (\bot \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n,s,f} M) && \text{(Fail, } \equiv_\eta)\\
&= (\phi, \psi \Vdash_{x_1,\ldots,x_m,y_1,\ldots,y_n} M) && \text{(def.)}
\end{aligned}
$$

□

Fact 2.

$$\phi^{\uparrow_{\mathbf{D}}}; \psi^{\uparrow_{\mathbf{D}}} = (\phi + \psi)^{\uparrow_{\mathbf{D}}}$$

Proof.

$$
\begin{aligned}
&\phi^{\uparrow_{\mathbf{D}}}; \psi^{\uparrow_{\mathbf{D}}}\\
&= \phi^{\uparrow_{\mathbf{D}}} \gg\!=_{\mathbf{D}e,f} (\lambda p, i, c.\psi^{\uparrow_{\mathbf{D}}} i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}f,\epsilon} (\lambda q.(p + q)^{\uparrow_{\mathbf{P}}})) && \text{(def.)}\\
&= (\lambda i, c.\psi^{\uparrow_{\mathbf{D}}} i(\lambda t.c(p \rightrightarrows t)) \gg\!=_{\mathbf{P}f,\epsilon} (\lambda q.(\phi + q)^{\uparrow_{\mathbf{P}}})) && \text{(Left Identity 2)}\\
&= (\lambda i, c.\psi^{\uparrow_{\mathbf{P}}} \gg\!=_{\mathbf{P}f,\epsilon} (\lambda q.(\phi + q)^{\uparrow_{\mathbf{P}}})) && \text{(def., } \equiv_\beta)\\
&= (\lambda i, c.(\phi + \psi)^{\uparrow_{\mathbf{P}}}) && \text{(Left Identity 2)}\\
&= (\phi + \psi)^{\uparrow_{\mathbf{D}}} && \text{(def.)}
\end{aligned}
$$

□